DeFinetti: A Mathematica program to analyze the replicator dynamics of 3-strategy collective interactions

Marco Archetti
Department of Biology, Pennsylvania State University, State College, PA, USA

ARTICLE INFO
Article history:
Received 18 June 2019
Received in revised form 1 January 2020
Accepted 3 February 2020

Keywords:
Game theory
Replicator dynamics
Public goods
Cooperation
Ternary plot
Ternary graph
Triangle plot
Simplex

ABSTRACT
DeFinetti is a Mathematica program that enables the user to visualize the replicator dynamics of three-strategy games with collective interactions, to study cooperation at different levels of biological complexity. The program computes the replicator dynamics of a game for multiple initial frequencies and shows trajectories, fitness and end points on a ternary plot (a DeFinetti diagram). The user only needs to specify the payoffs, and can change the parameters, visualize their effect, customize the style of the graphs and export the results.

© 2020 The Author. Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).
While many types of PGGs assume that individuals can be either cooperators or defectors (hence use two-strategy games), many cases involve three strategies. Three-way symbioses [18–20] are obvious examples in which DeFinetti diagrams are useful. Another general case in which two strategies are not enough to describe cooperation for the provision of public goods occurs when the public good derives from a third party. For example, cooperators among a population of symbionts can provide diffusible nutrients to a host (whereas non-cooperative symbionts free-ride on the contributions of the cooperators) which in turn provides a benefit to the symbionts. The host here is a third type whose fitness and frequency changes over time, hence a three-player game must be used. Examples of indirect provision of public goods exist in cell biology too: in tumors, for example, cancer cells that produce or do not produce diffusible growth factors can engage in a public goods game with two strategies (cooperators and defectors); more commonly, however, cancer cells secrete molecules that induce other “stromal” cells to provide beneficial growth factors to the tumor — hence again, a three-way system that is described by DeFinetti diagrams [21,22].

1. Motivation and significance

1.1. DeFinetti diagrams

A DeFinetti diagram (also known as ternary plot, ternary graph, triangle plot, simplex plot or simplex) is a plot of three variables that sum to a constant. The position of each point within the plot, which is bounded by an equilateral triangle, shows the values of the three variables. A ternary plot therefore enables the user to represent three variables in a two-dimensional graph. While not strictly necessary, since there are effectively only two variables (as the three variables sum to a constant), an equilateral triangle shows graphically the ratios of the three variables in an intuitive way: the concentration of each variable is 100% (pure type) at each corner of the triangle and decreases linearly with increasing distance from this corner, reaching 0% at the line opposite it (by Viviani’s theorem: the sum of the distances from any interior point to the sides of an equilateral triangle equals the length of the triangle’s altitude) [Fig. 1].

Ternary plots are widely used, from chemistry and physics to geology and biology, to show the status of systems composed of three types. In biology, they are often used to describe not only static systems but also the dynamics of populations over time. In population genetics ternary plots have traditionally been used to show the changes in genotypic frequencies of systems with one locus and two alleles (resulting in three genotypes) [1]; ternary plots have also been of interest to evolutionary biologists and behavioral ecologists since the inception of evolutionary game theory [2] to study interactions between individuals with three strategies.

1.2. Public goods games

Public goods games (PGGs) are multiplayer games with collective interactions (payoffs are derived from fitness functions that depend on the behavior of multiple other players, not simply on the outcome of pairwise interactions). Examples of PGGs range from microbes secreting diffusible molecules [3] and cancer cells secreting growth factors [4] to cooperative hunting in mammals [5,6], birds [7] and arachnids [8,9], sentinels in cooperative mammals [10] and birds [11]. The exploitation of shared natural resources [12] is a classic case of PGG, and examples of collective interactions in environmental sciences and conservation biology range from land management [13] to problems related to global climate change [14,15]. Analogous collective action problems arise in most human social interactions and interventions for public goods, such as contribution to herd immunity in vaccination campaigns [16]. Hence, collective action problems for the production of public goods occur in many cases from microbiology to ecology and environmental sciences and are thought to be crucial in most of the major transitions in evolution [17].

While extensions exist for certain programming languages to plot ternary diagrams, they do not allow to plot changes over time, which is often essential in biology. Dynamo [23,24], as well as its more recent companion EvoDyn-3s [25], are notable exceptions that enable the user to visualize different types of dynamics (including the replicator dynamics) of 3-strategy games on ternary plots. The recent Python package egtplot [26] has similar capabilities. They can be used to model three or four-strategy games, two-population, 3 × 2 strategy games and three-population, 2 × 2 × 2 strategy games but not multiplayer games with collective interactions (public goods games: PGGs).

I have developed a Mathematica program to plot the replicator dynamics of three-strategy multiplayer games with collective interactions. Here I describe the software, which is also available on https://github.com/archetti/DeFinetti. Examples of figures produced by the software can be found in references [22] and [27]; further examples are reported here.

2. Software description

DeFinetti is an executable program written in Mathematica. It contains the source code and intuitive instructions on how to use it. There is no need to compile it — it can be used directly by simply opening it with Mathematica. Parameters and style can be modified using a series of intuitive input boxes, sliders and menus, and the results are updated and exported either automatically and in real time or by pressing a button, without rerunning the program. The code, hidden in the interface, can be revealed (and modified, if necessary) by clicking on each section.

2.1. Software architecture

DeFinetti conducts the following operations:

- Produces a three-strategy multiplayer public goods game with collective interactions, as described in the Appendix, using the following input provided by the user: names of the three types; group size; the cost $c_i$ for each type $i$ in Eqs. (2)–(4); the values of the maximum benefits $B_{ij}$ of the inflection points $h_{ij}$ and of the steepness values $s_{ij}$ in Eq. (10); the sign of $B_{ij}$; the number of starting points and the number of generations can also be controlled by the user.
- Computes the replicator dynamics numerically for the given starting points.

Fig. 1. In a ternary plot, three quantities $a$, $b$, $c$ corresponding to three types $A$, $B$ and $C$ are described by the position of a point within an equilateral triangle. The distance of this point from each side of the triangle is proportional to the quantity corresponding to the opposite vertex. Vertices denote pure types.
Fig. 2. Curves show the frequencies of the three types and the average fitness of the population over time on a logarithmic scale. Parameters: \( n = 10; \quad c = [0.1, 0.2, 0.3]; \quad b = [[0, 1, 1.1], [1, 0, 0], [1.1, −0.3, 0]]; \quad h = [[0.3, 0.3, 0.3], [0.3, 0.3, 0.3], [0.3, 0.3, 0.3]]; \quad s = s = [[20, 20, 20], [20, 20, 20], [20, 20, 20]]; \quad inc = [[True, True, True], [True, True, True], [True, True, True]]. \) The corresponding ternary plots are shown in Figs. 3 and 5.

Fig. 3. Ternary plots for the parameters in Fig. 2. The starting points are fixed, but the user can choose how many of these starting points to use and the style of the trajectories (number of arrows and color gradient), and add further starting points by clicking directly on the simplex.

- Shows the changes in frequency of the three strategies over time in a 2-axis plot.
- Shows a series of trajectories and arrows showing the dynamics in a ternary plot.
- Shows the end points of the system in the ternary plot.
- Colors the trajectories in the ternary plot according to the average fitness of the population, using a color gradient selected by the user.
- Enables the user to add further starting points directly on the ternary plot.

2.2. Software functionalities

The user can specify the parameters of the model and the style of the plots using interactive input fields, sliders and popup menus. The results are updated either automatically in real time as the controls are changed, or they are generated upon pressing specific buttons, enabling a fast update of the results that are affected by the changes, with no need to run the program multiple times.

Files generated by the program are named \( \text{path/time/filename} \), where \( \text{path} \) is the path of the directory where the \( \text{DeFinetti.nb} \) file is located; \( \text{time} \) is the time and date the file is generated (in this format: \( \text{year-month-day-hours-minutes-seconds} \)), which identifies each result; \( \text{filename} \) is the type of the file generated.

A text file (\( \text{filename} = \text{Parameters.txt} \)) listing the values of the parameters can be generated when the parameters are modified. The content of the file is the following: \( \text{Types:} \) a vector with the names of the three types; \( \text{n:} \) group size \( N \); \( \text{c:} \) a vector with the cost \( c_i \) for each type \( i \) in Eqs. (2)–(4); \( \text{b:} \) a matrix with the maximum value \( B_{i,j} \) in Eq. (10); \( \text{h:} \) a matrix with the inflection point \( h_{i,j} \) in Eq. (10); \( \text{s:} \) a matrix with the steepness \( s_{i,j} \) in Eq. (10); \( \text{inc:} \) a matrix describing whether the sign of \( B_{i,j} \) is positive.

The first graphical results are a series of plots showing the benefit functions (the effect of each type on each other) described by Eq. (10) given these parameters. These plots are exported to a file with \( \text{filename} = \text{Benefits.jpg} \).

The first plot of the dynamics shows the frequencies and the average fitness (normalized between its minimum and maximum values) of the population as a function of time (generation...
Fig. 4. Examples of ternary plots obtained using different parameters (all values of \( \text{inc} \) are positive). A: \( n = 10; c = [0,0.1,0.2]; b = [[0,0.0], [0.5,0.5,0.5], [1,1]]; h = [[0.5,0.5,0.5], [0.5,0.5,0.5], [0.5,0.5,0.5]]; s = [[20,20,20],[20,20,20],[20,20,20]]; \) B: \( n = 10; c = [0.1,0.2,0.3]; b = [[0,1,1],[1,0,0],[1,1,1]]; h = [[0.3,0.3,0.3],[0.3,0.3,0.3],[0.3,0.3,0.3]]; s = [[20,20,20],[20,20,20],[20,20,20]]; \) C: \( n = 20; c = [1,2,1,1,1]; b = [[1,1,1,1,1],[0.95,0.95,0.95],[1.8,-0.35,0.35]]; h = [[0,0.0],[0.0,0.0],[0.2,0.2,0.2]]; s = [[4,4,4,4],[4,4,4],[6,6,1500]]; \) D: \( n = 10; c = [0.1,0.1,0.1]; b = [[1,1,1],[1,1,1],[1,1]]; h = [[0.4,0.4,0.4],[0.7,0.4,0.4],[0.4,0.4,0.4]]; s = [[20,20,20],[20,20,20],[20,20,20]]; \) E: \( n = 20; c = [0.1,0.1,0.1]; b = [[0.45,0.9,2],[0.9,0.3,0],[2,-0.3,0]]; h = [[0.05,0.05,0.05],[0.05,0.05,0.05],[0.5,0.5,0.5]]; s = [[10,10,10],[10,10,10],[50,50,50]]; \) F: \( n = 10; c = [0.1,0.1,0.1]; b = [[1,1,1],[0.7,0.7,0.7],[0.9,0.9,0.9]]; h = [[0.4,0.7,0.1],[0.7,0.4,0.2],[0.4,0.3,0.7]]; s = [[20,20,20],[20,20,20],[20,20,20]]; \)

Fig. 5. Examples of results using different styles (number of arrows, number of starting points, rescaling of colors) for the same parameters \( n = 10; c = [0.1,0.2,0.3]; b = [[0,1,1,1],[1,0,0],[1,1,-0.3,0]]; h = [[0.3,0.3,0.3],[0.3,0.3,0.3],[0.3,0.3,0.3]]; s = [[20,20,20],[20,20,20],[20,20,20]]; \)

number, on a logarithmic scale to highlight the initial, usually wider, changes) for given initial frequencies [Fig. 2]. The initial frequencies of the three types can be changed using sliders and the results are produced in real time. The number of generations is initially set to 200, but can be changed. The user can also specify the colors of the plot. This plot is not exported by default, but can be saved by the user.

The core of the program produces ternary plots using the following coordinate transformation: if the pure \( A \) type has Cartesian coordinates \((0,0)\) and pure \( B \) has coordinates \((1,0)\), then the
Cartesian coordinates of the pure C type in a ternary plot are \((1/2, \sqrt{3}/2)\); hence a point \((a, b, c)\) has Cartesian coordinates:

\[((2b + c)/(2(a + b + c)), (\sqrt{3}/2)c/(a + b + c))\]

In the ternary plot, each point represents the three frequencies at each generation; points are not actually shown; instead, they are joined by segments that form a trajectory describing the dynamics of the population over time. The color of each segment shows the average fitness of the population at that generation. The frequency of arrows, as well as the number of initial starting points and the colors representing average fitness can be customized using sliders.

In addition to choosing among a variety of color schemes, it is possible to normalize the fitness values (that is, rescale them between the observed fitness minimum and maximum). This is useful to highlight variations in fitness within the plot, which might be too small for some parameters; on the other hand, not rescaling the values (that is, using absolute fitness values) enables an easier comparison between plots with different parameters.

Finally, arrows are added to the trajectory to show the direction of the dynamics, and it is possible to choose whether to show the end points or not (which is useful, for example, in the case of cyclical dynamics) [Fig. 3].

The plot is exported to a file with filename = Simplex.pdf. A separate file with a legend showing the values corresponding to the colors for average fitness is also produced and exported automatically (filename = Legend.pdf).

Additional starting points can be added in the final section of the software, by clicking on any point in the simplex [Fig. 3]. This part of the code uses the Locator command in Mathematica, and requires an inverse transformation, from Cartesian to Simplex coordinates: given Cartesian coordinates \((x,y)\), the corresponding coordinates on the simplex \((a, b, c)\) are:

\[\left(1 - x - y/\sqrt{3}, x - y/\sqrt{3}, 2y/\sqrt{3}\right)\]

Showing individual trajectories is sometimes useful in order to highlight peculiar dynamics from specific starting points that are not included in the default set. The user can decide whether to overlap the new trajectory with the ones already obtained or not. This plot is not exported by default, but can be saved by the user, and added (using any image processing software) to the ones already shown in the original standard plot.

3. Illustrative examples

Examples of ternary plots produced using different parameters are shown in Fig. 4. Further examples using different styles for the same parameters are shown in Fig. 5. Fig. 7 shows how DeFinetti enables the user to visualize the results of changing one set of parameters in the study of tumor-stroma interactions in multiple myeloma [22]: the three strategies are the three interacting cell phenotypes: osteoblasts (OB), osteoclasts (OC) and malignant plasma cells (MM). A healthy equilibrium of OB and OC cells can be destabilized by the introduction of MM cells. Fig. 7 shows how changing the position of the inflection points of the cytokines produced by the three types of cells (which simulates targeted therapy against those cytokines) changes the dynamics and end points of the model.

4. Impact

4.1. Differences with previous software

The main difference between DeFinetti and alternate programs such as Dynamo [23,24], EvoDyn-3s [25] and egtplot [26] is that DeFinetti plots three-strategy games with collective interactions (the payoff of each player depends on the strategy of many other players) rather than pairwise interactions (the payoff of each player depends only on the strategy of its single opponent). Additional, minor differences include the ability to visualize the evolution of average population fitness (although at the expense of the visualization of the speed of the dynamics—a notable feature of Dynamo and EvoDyn-3s which is absent here); and the ability to add further starting points directly on the ternary plot, enabling the study of very specific initial starting frequencies. A notable shortcoming of DeFinetti is that it is limited to the replicator dynamics (Dynamo and EvoDyn-3s can plot different types of dynamics).

4.2. Limitations and potential extensions

Determining the equilibria of three-strategy multiplayer non-linear games with collective interactions is not feasible [27]. The end points in DeFinetti are inferred by looking at the values of the frequencies at the last generation. Because of this, two problems may arise. First, if the number of generations used is not high enough, multiple end points might seem to arise (because each trajectory will be truncated too early); this problem can be easily solved by choosing a high number of generations, and does not usually lead to any issue. Second, if not enough starting points are used, some end points may be missing if their basin of attraction is small (because no starting points are used within those basins of attraction); this is a more serious issue, which can be solved by choosing a large number of starting points—which however leads to the following possible other issue.

The only known serious issue with the current version is that under certain conditions (a high number of starting points and using normalized values for fitness colors), on some machines, the “update simplex” button does not actually produce a new simplex after updating the parameters. This seems to happen because the simultaneous use of these two options requires a high computational effort. Three solutions are possible: choosing a lower number of starting points (and if necessary eventually adding individual trajectories as explained above); using absolute values for fitness colors (instead of normalizing them); and plotting the dynamics without colors. These options are all easily selected using interactive commands. In addition, if normalized fitness colors for many trajectories are required, the problem can sometimes be solved by evaluating manually (selecting and pressing shift-enter) the line of code corresponding to the “update simplex” button (rather than simply pressing the button). This issue, when it happens, does not have an impact on the other parts of the program.
The program allows to change the parameters that specify the benefit functions used for the three types, allowing to study the dynamics of multiplayer public goods games with monotonic nonlinear benefits ranging from linear, concave and convex to sigmoid and step functions. It does not include more complex, non-monotonic functions. Users with a basic knowledge of *Mathematica* could easily modify the source code in order to extend it to more complex benefit functions or other types of three-strategy games. Alternatives to the replicator dynamics could also be introduced, like *Dynamo* [23,24] and *EvoDyn-3s* [25] do for games with pairwise interactions.

### 5. Conclusion

*DeFinetti* enables to visualize the replicator dynamics of three-strategy games with collective interactions, an advance over existing programs that only allow pairwise interactions. The most obvious use of *DeFinetti* is in the study of public goods games, which is useful in many fields of biology, from microbiology and cancer research to behavioral ecology and human social behavior. I would like to encourage colleagues to suggest additional improvements and extensions.

### Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

### Appendix. Replicator dynamics

In evolutionary game theory, the change in frequency over time is often assumed to be based on the replicator dynamics [28]

\[
x'_{i} = x_{i} \left[ W(i) - \sum_{j=a,b,c} x_{j} W(j) \right]
\]

where \(x_{i}\) and \(W(i)\) are, respectively, the frequency and the fitness of type \(i\). Hence the rate of change \(x'_{i}/x_{i}\) is equal to the difference between the fitness of type \(i\) and the average fitness of the population. As is standard in models of large, well-mixed populations, fitness is calculated by considering the payoffs obtained in randomly formed groups of size \(N\) weighted by the probability that such groups occur.

In a system with three types (\(i = A, B, C\)) we can define \(x_{A} = a; x_{B} = b; x_{C} = c\) (with \(a + b + c = 1\)); in a well-mixed population, the probability that a group contains \(n_{A}, n_{B}\), and \(N - n_{A} - n_{B}\) individuals \(n_{i}\) of type \(A, B\) and \(C\), respectively, is given by

\[
\left( \begin{array}{c} N - 1 \\ n_{A}, n_{B} \end{array} \right) \frac{n_{A}^{a} \cdot n_{B}^{b} \cdot n_{C}^{c}}{N - n_{A} - n_{B} - 1}.
\]

and the payoffs for \(A, B\) and \(C\) are, respectively:

\[
\begin{align*}
V_{A} &= b_{A,A}(n_{A} + 1) + b_{B,A}(n_{B}) + b_{C,A}(N - 1 - n_{A} - n_{B}) - c_{A} \\
V_{B} &= b_{A,B}(n_{A}) + b_{B,B}(n_{B} + 1) + b_{C,B}(N - 1 - n_{A} - n_{B}) - c_{B} \\
V_{C} &= b_{A,C}(n_{A}) + b_{B,C}(n_{B}) + b_{C,C}(N - n_{A} - n_{B}) - c_{C}
\end{align*}
\]

where \(c_{i}\) is the cost for type \(i\) and \(b_{ij}\) describes the effect of type \(i\) on type \(j\). Note that each player’s strategy has also an effect on itself. The fitness of type \(i\) is given by

\[
W_{i} = \sum_{n_{A}=0}^{N-1} \sum_{n_{B}=0}^{N-1-n_{A}} \left( \begin{array}{c} N - 1 \\ n_{A}, n_{B} \end{array} \right) \frac{n_{A}^{a} \cdot n_{B}^{b} \cdot n_{C}^{c}}{N - n_{A} - n_{B} - 1} \cdot V_{i}
\]

The frequencies change based on the replicator dynamics:

\[
\begin{align*}
x_{A}' &= x_{A}(W_{A} - W^{*}) \\
x_{B}' &= x_{B}(W_{B} - W^{*}) \\
x_{C}' &= x_{C}(W_{C} - W^{*})
\end{align*}
\]

where \(W^{*} = x_{A}W_{A} + x_{B}W_{B} + x_{C}W_{C}\).

The benefit function, which describes the effect of each type on each other, is defined by

\[
b_{ij}(n_{i}) = \frac{B_{ij}}{1 + e^{a_{i}/h_{ij} - n_{i}/N}}
\]

The parameters define the effect of type \(i\) on type \(j\): \(B_{ij}\) defines the maximum benefit; \(h_{ij}\) defines the position of the inflection point: \(h_{ij} \rightarrow 1\) gives increasing returns and \(h_{ij} \rightarrow 0\) diminishing returns; \(s_{ij}\) defines the steepness of the function at the inflection point (\(s_{ij} \rightarrow \infty\) models a threshold public goods game; \(s_{ij} \rightarrow 0\) models a linear benefit [Fig. 6]; and the normalization in (9) in prevents the logistic function (10) from becoming constant for \(s_{ij} \rightarrow 0\). This framework enables to analyze a wide range of non-linear multiplayer games with collective interactions [27,29].
References

[1] Cannings C, Edwards AWF. Natural selection and the de Finetti diagram. Ann Hum Genet 1968;31:421–8.

[2] Smith Maynard. Evolution and the theory of games. Cambridge, UK: Cambridge University Press; 1982.

[3] West SA, Diggle SP, Buckling A, Gardner A, Griffin AS. The social lives of microbes. Annu Rev Ecol Evol Syst 2007;38:53–77.

[4] Axelrod R, Axelrod DE, Pienta KJ. Evolution of cooperation among tumor cells. Proc Nat Acad Sci USA 2006;103:13474–9.

[5] Heinsohn R, Packer C. Complex cooperative strategies in group-territorial African lions. Science 1995;269:1260–2.

[6] MacNulty DR, Smith DW, Mech LD, Vucetich JA, Packer C. Nonlinear effects of group size on the success of wolves hunting elk. Behav Ecol 2011;23:75–82.

[7] Bednarz JC. Cooperative hunting in Harris’ Hawks (Parabuteo unicinctus). Science 1988;239:1525–7.

[8] Kim KW, Krafft B, Choe JC. Cooperative prey capture by young subsocial spiders. Behav Ecol Sociobiol 2005;59:92–100.

[9] Yip EC, Powers KS, Avilés L. Cooperative capture of large prey solves scaling challenge faced by spider societies. Proc Natl Acad Sci USA 2008;105:11818–22.

[10] Clutton-Brock TH. Selfish sentinels in cooperative mammals. Science 1999;284:1640–4.

[11] Pulliam HR, Pyke GH, Caraco T. The scanning behavior of juncos: A game-theoretical approach. J Theoret Biol 1982:95:89–103.

[12] Hardin G. The tragedy of the commons. Science 1968;162:1243–8.

[13] Rustagi D, Engel S, Kosfeld M. Conditional cooperation and costly monitoring explain success in forest commons management. Science 2010;330:961–5.

[14] Ricke KL, Moreno-Cruz JB, Schewe J, Levermann A, Caldeira K. Policy thresholds in mitigation. Nat Geosci 2016;9:5–6.

[15] Steffen W, et al. Trajectories of the earth system in the Anthropocene. Proc Natl Acad Sci USA 2018;2:10141–8.

[16] Reluga TC, Galvani AP. A general approach for population games with application to vaccination. Math Biosci 2011;230:67–78.

[17] Maynard Smith J, Szathmáry E. The major transitions in evolution. San Francisco, CA, USA: Freeman; 1995.

[18] von Dohlen CD, Kohler S, Alisp ST, McManus WR. Mealybug beta-proteobacterial endosymbionts contain gamma-proteobacterial symbionts. Nature 2001;412:433–6.

[19] Márquez LM, et al. A virus in a fungus in a plant: three-way symbiosis required for thermal tolerance. Science 2007;315:513–5.

[20] van der Heide T. A three-stage symbiosis forms the foundation of seagrass ecosystems. Science 2012;336:1432–4.

[21] Sartakhti JS, Manshaei MH, Bateni S, Archetti M. Evolutionary dynamics of tumor-stroma interactions in multiple myeloma. PLoS One 2016;11(12). e0168856.

[22] Sartakhti JS, Manshaei MH, Archetti M. Game theory of tumor–stroma interactions in multiple myeloma: effect of nonlinear benefits. Games 2018;9(32).

[23] Sandholm WH, Dokumaci E, Franchetti F. Dynamo: Diagrams for evolutionary game dynamics. 2012. http://www.ssc.wisc.edu/whs/dynamo.

[24] Franchetti F, Sandholm WH. An introduction to dynamo: diagrams for evolutionary game dynamics. Biol Theory 2013;8(2):167–78.

[25] Izquierdo LR, Izquierdo SS, Sandholm WH. EvoDyn-3s: A mathematica computable document to analyze evolutionary dynamics in 3-strategy games. SoftwareX 2018;7:226–33.

[26] Mirzaev I, Williamson DFK, Scott JG. Egtplot: A python package for three-strategy evolutionary games. J. Open Source Softw 2018;3(26):735.

[27] Archetti M. How to analyze models of nonlinear public goods. Games 2018;9(17).

[28] Taylor PD, Jonker LB. Evolutionary stable strategies and game dynamics. Math Biosci 1978;40:145–56.

[29] Archetti M, Scheuring I. Review: Evolution of cooperation in one-shot social dilemmas without assortment. J Theoret Biol 2011;299:9–20.