JOSS
The Journal of Open Source Software

APGG - A Modular C++ Framework for Asymmetric Public Goods Games

Mirko Rosenthal[1], **David J. Richter** [1,2,5], **Falk Hübner**[1], **Jochen Staudacher** [1], **and Arend Hintze** [3,4]

1 Kempten University of Applied Sciences, Germany **2** Purdue University Northwest, USA **3** MicroData Analytics, Dalarna University, Sweden **4** BEACON Center for the Study of Evolution in Action, Michigan State University, USA **5** Chonnam National University, South Korea

Summary

The Asymmetric Public Goods Game (APGG) C++ framework offers an easy to use environment to study game theoretical questions. Specifically, it is designed to address questions in the domain of asymmetric public goods games. The modular architecture allows for a vast amount of scenarios and setups for experimenting with different public goods games, using easy to change parameters. Users can experiment with well mixed and structured populations as well as with symmetric and asymmetric payoffs. APGG also features group level payoffs and individual payoffs, and different evolutionary selection mechanisms (Miller et al., 1995) and replication schemes. Results are automatically saved in semantic and descriptive structures and can be easily visualized with the included Python scripts. This paper aims to explain the functionality and the structure of the framework, to show the workflow that APGG follows, to present the different modules that are available, and to show how APGG can be used to run experiments with public goods games on example scenarios.

Figure 1: APGG Logo.

Introduction / Literature Review

Public Good Games illustrate the tragedy of the commons, see (Hardin, 1968) for the standard reference which is both widely cited and still controversially discussed (Mildenberger, 2019). These games have been intensively studied before, with an increasing interest in the asymmetric variant of the game (Hintze et al., 2020; McGinty & Milam, 2013). Typically, game theoretical problems are solved using rigorous mathematical analysis, but that approach reaches its limits when it comes to the stochastic and random behavior of the evolutionary process (Adami et al., 2016). Consequently, computational models, such as this one, are used.

One problem of using computational models in research, is their oftentimes limited expandability. APGG remedies this problem by providing a modular framework that is designed to be easily extended in a vein similar to Bohm et al. (2017) and Richter (2019).

What is a Public Goods Game

The tragedy of the commons describes an important social and economical phenomenon that pitches self interest against the interests of a group. Players in a Public Goods Game (PGG) can either contribute to a common pool (cooperate) or withhold their contribution (defect). The money collected in the pool is increased by a multiplicative synergy factor, and then equally distributed amongst the players. It becomes immediately clear that the defecting players will always receive the same as the cooperators, but end up having more money than the cooperators due to the amount they withheld before. The tragedy specifically describes the dilemma, that if all players would cooperate the total amount received by everyone would be higher, but the greed (or self interest) of the defectors prevents that favourable outcome.

The question is how to overcome the tragedy of the commons. In social societies, often institutions, regulations, and incentives are used (Fehr & Gächter, 2002; Hintze et al., 2020). In the theoretical context, all of this becomes abstracted as costly punishment (Hardin, 1968). This costly punishment has been shown to affect human behavior (Fehr & Gächter, 2002) and can indeed lead to the evolution of cooperation (Hintze & Adami, 2015). Another option that alters the outcome towards cooperation could be asymmetric distribution of resources, as it can be found in many animal hierarchies. Hyenas for example have a steep despotic index (Smith et al., 2007) while also cooperating with each other, supporting the idea of asymmetric payoffs potentially leading to cooperation.

Statement of Need

Studying evolution in biological systems is cumbersome, to say the least (Lenski, 2017). Consequently, using computational models becomes a viable alternative. However, for new experiments to build on previous results, the modeling software needs to be extendable. This leads to a challenging problem. Future users will independently modify the software to suit their own needs, with no regard for other users. This could create an ever growing tree of alternative versions, that might not be compatible with each other. Here, a modular design (see Figure 2) approach is used, such that possible future users can define custom modules. However, those modules will remain interoperable, because interfaces are well defined.
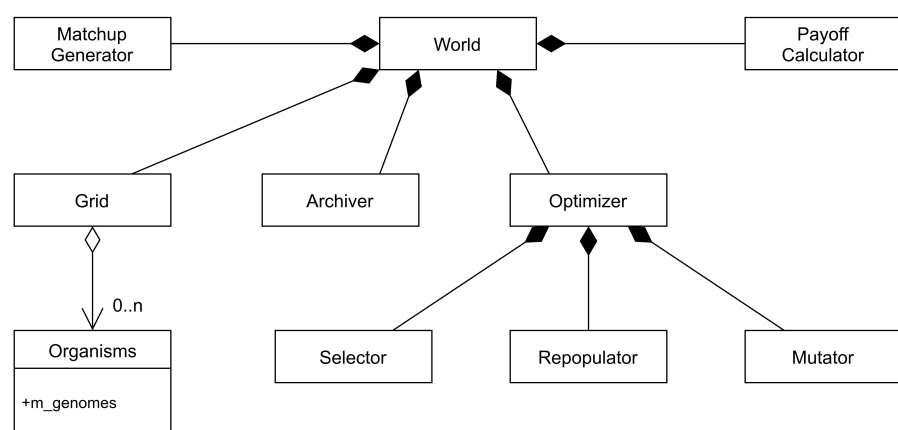
Code Overview

Figure 2: A Figure Displaying the Class Structure of APGG.

World: Encapsulates a single experiment. The World Class is divided into 4 methods, init, tick (plays the game), evolve and fini. The main loop (tick and evolve) is being run within the world class and controls the entire game, by calling all the relevant classes and methods. All

other classes are being initialized within the world class. World has no subclasses and is used for every experiment.

Grid: The grid is home to all organisms (the population), and functions as the playing field/game-board for the model. Grid does not necessarily have to be a "grid" in a geometric sense. All organisms are initialized onto the grid and the population will play, compete and evolve on the grid. The grid can either be used as a "grid" (spatialGrid), just like the word suggests, where the spatial location of the organisms matters and is taken into consideration. In the "defaultGrid" mode, where the grid only stores the organisms, the placement has no effect on the games and evolution (well mixed population). Extending this module allows users to define arbitrary population topologies.

Organism: The organisms in APGG are the class for the agents that participate in the evolutionary public goods games. Depending on the grid size, a fixed number of organisms will be spawned. Organisms contain a Genome, which in turn holds the information that determines how agents act during the game. The Genomes are also used for the selection and repopulation process, as offspring will be derived from those Genomes by copying them. If users want to change how strategies are encoded this class should be modified. See Mutator for changing how Genomes are copied.

Archiver: The Archiver Class is used to collect data during the runtime of the model. It has 2 subclasses. The SimpleArchiver, which will write the status of the current generation (number of cooperators, defectors) as well as the LODArchiver, which will print the whole Line of Descent (for the best organism and his ancestors) to a file at the end of each experiment.

Mutator: The mutator is responsible for mutating the genome of organisms during the selection process. The chance for the organisms to mutate is set by the mutationRate parameter. If a random value between 0 and 1 exceeds the mutationRate, the genomes of the organism will be changed. In addition to the normal (default) mutator, there is a threshold mutator, which will reassign the genome a new value within a specific range of the current value. Overloading this class allows users to implement other mutation schemes or modes.

PayOffCalculator: The payoff calculator is the calculation class of the APGG framework. It is used to calculate the payoff each organism receives based on its decisions. For example, in a PGG with punishment, the punishment cost and the punishment fine as well as the rewards from cooperation or defection are calculated. Other game modes can be implemented by overloading this class.

The payoff calculation for the PGG with punishment is a two step process. In the first step, the base costs, fines, and payoff are calculated. The second step calculates the individual payoff based on the organism's decision. Depending on its decision, the organism has to pay a punishment cost or a punishment fine out of its individual payoff. There are payoff calculators for different scenarios: The AsymmetricPayoffCalculator is a calculator, which calculates the payoff based on a spatial tier list. Everyone transfers his individual payoff into a group pool. After that, everyone gets his individual payoff out of this pool based on his rank in the group (Smith et al., 2007). The GroupLevelPayoffCalculator (Hintze et al., 2020) is a calculator where everyone transfers a predefined part of his payoff into a group pool for group members to share rewards with each other. After everyone has paid into this pool, the pool will be split by the number of group members and the value will be added to the individual payoff.
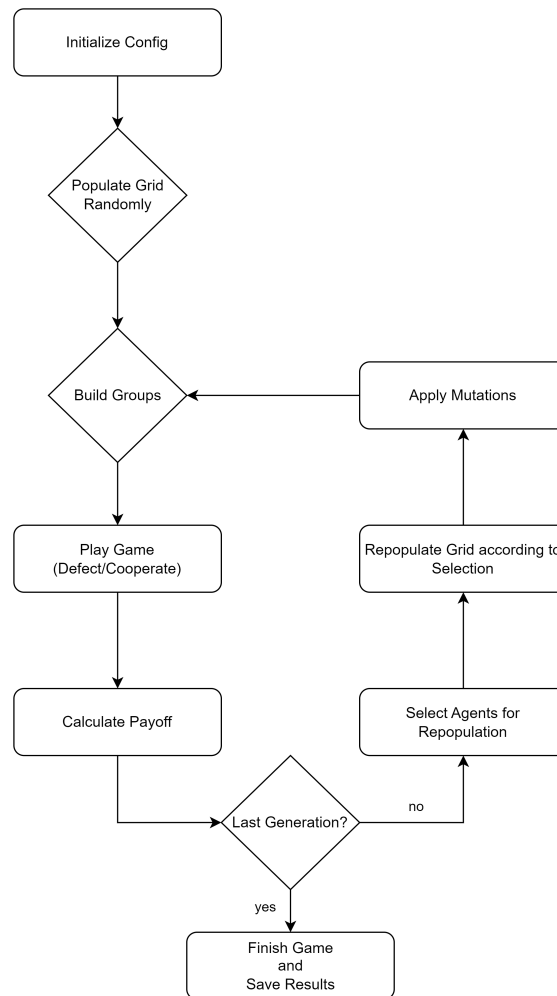
Workflow

Figure 3: A Figure displaying the Workflow that APGG follows.

APGG was designed to be an easy-to-use framework for making experiments with Public Goods Games available to a wide range of users. Hence, the workflow Figure 3 has been designed to not only please computer scientists but to also invite social and biological researchers with basic programming skills. To set up and configure an experiment or a set of them, all users have to do is edit a csv file in a spreadsheet software of their choice. To expand the framework by adding or changing modules only basic C++ programming knowledge is recommended.

The computer model runs using the following steps:

1. Initialize the grid and create all organisms randomly
2. Start the main loop for number of generations
3. Generate all sets of players interacting in games during one generation
4. Evaluate all sets of games
5. Select those organisms who will reproduce based on their payoff
6. Repopulate the grid using the organisms identified in the previous step
7. Go to step 2, until the total number of desired generations is reached
8. Save all data

Proof of Concept

Parameters	r	μ	Pop. Size	Group Size	γ	β	i
Value:	1.5 to 7.0	0.02	1000	5	0.2	0.8	1

Table 1: Parameters used to replicate our first proof of concept experiment, where r is the synergy factor, μ is the mutation rate, γ is the punishment cost, β is the punishment fine, and i is the payoff individualism (with $i = 1$ denoting the maximal level of selfishness and thus the classical public goods game). The experiment was run for 100,000 generations with 100 replicates. The Random Mutator was picked, as was the Random Selector. The Repopulator used was the Proportionate Repopulator.

To illustrate that the software is capable of reproducing scientific results, we first repeated an experiment from Hintze & Adami (2015). In that experiment the synergy factor in the public goods game was varied and the outcome of evolution, given that factor, determined. When the synergy factor is low, we expect defectors to win, while cooperators should thrive when synergy is high. However, in this variant of the game, agents can also punish, and punishment can modulate the response of agents to the synergy factor. The expectation was that punishment should allow agents to cooperate at lower synergy factors, and one should observe punishment to increase around this critical point. Interestingly, agents' probabilities to cooperate and to punish evolved as expected, and punishment lowered the critical point, but the chance to punish went from 0.0 to 0.5 (drifting). Figure 4 confirms both the phenomenon of drifting as well as the fact that at a synergy level of around 4.0 there are hardly any defectors left.

Let us look at the shift of the critical point from 5.0 (without punishment) down to a lower level (with punishment) in a little more detail. The critical point is defined as the smallest synergy factor for which the payoff of a cooperator equals or exceeds the payoff of a defector. Hintze & Adami (2015) first observed that this critical point depends upon the density of punishers ρ_P and Hintze et al. (2020) pointed out that in the absence of mutations the critical point can be expressed as

$$r_C = i(1 - \beta\rho_P)(k + 1) - i + 1 - (1 - i)(\beta + \gamma)\rho_P \tag{1}$$

with $k + 1$ standing for the group size. For example, for $\rho_P = 0.4$ and our setting of parameters, expression (1) predicts $r_C = 1 \cdot (1 - 0.8 \cdot 0.4) \cdot 5 = 3.4$ without mutations. As expected we observe a transition from defection to cooperation for synergy factors between 2.5 and 4.0 in Figure 4.
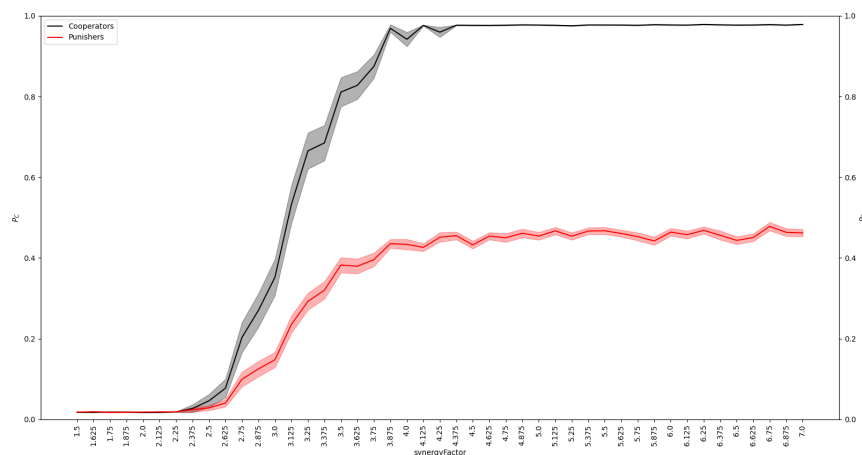
Figure 4: Evolved agent behavior (y-axes) in the public goods game with punishment for different synergy factors (x-axis). The evolved likelihood to cooperate (in black, left y-axis) and to punish (in red, right y-axis) is shown for 100 replicate experiments per tested synergy factor value. Shadows in red and black show the 95% confidence intervals respectively. The expected critical point for agents not being able to punish is at a synergy factor of 5.0, and here we find this critical point to be shifted to a lower level as expected.

Parameters	r	μ	Pop. Size	Group Size	γ	β	i
Value:	1.5 to 7.0	0.02	1000	5	0	0	0.5

Table 2: Parameters used to replicate our second proof of concept experiment, where r is the synergy factor, μ is the mutation rate, $\gamma = \beta = 0$ denote that neither a punishment cost nor a punishment fine is imposed, and $i = 0.5$ is the payoff individualism (which is called the level of selfishness ζ in Hintze et al. (2020)). The experiment was run for 100,000 generations with 100 replicates. The Random Mutator was picked, as was the Random Selector. The Repopulator used was the Proportionate Repopulator.

We also confirm a result from Hintze et al. (2020). Again, the synergy factor in the public goods game was varied and the outcome of evolution, given that factor, determined. In this variant of the game, there are neither fines nor costs for punishment. However, we impose a payoff individualism of $i = 0.5$, meaning that players keep only half of their individual payoffs whereas the other half is transferred into a pool and then distributed evenly among the group members. Expression (1) predicts $r_C = 0.5 \cdot 5 - 0.5 + 1 = 3$ implying that as soon as the synergy factor exceeds 3.0 cooperators will benefit from a larger payoff than defectors. Figure 5 confirms the shift of the critical point from 5.0 (classical public goods game, $i = 1$) down to a synergy factor of 3.0 with a probability to cooperate of 0.5 for that case. It also displays the gradual transition from defection to cooperation in the presence of mutations reported in Hintze et al. (2020), Figure 1. The probability to punish is drifting throughout the experiment as expected.
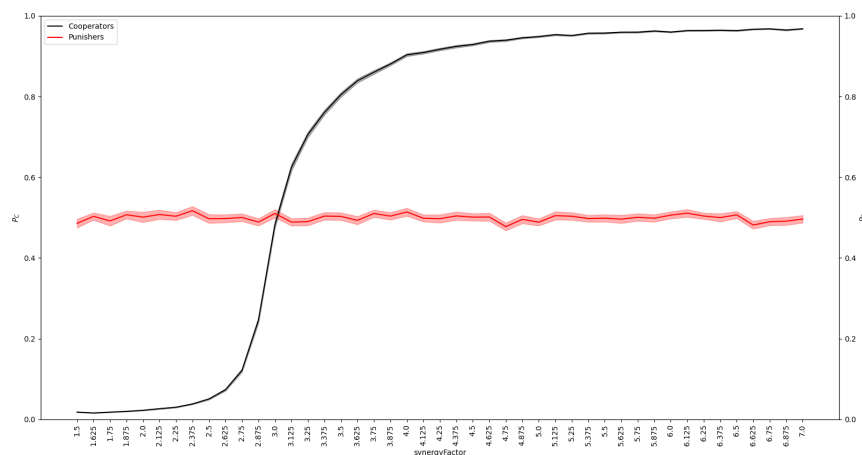
Figure 5: Evolved agent behavior (y-axes) in the public goods game with punishment for different synergy factors (x-axis). The evolved likelihood to cooperate (in black, left y-axis) and to punish (in red, right y-axis) is shown for 100 replicate experiments per tested synergy factor value. Shadows in red and black show the 95% confidence intervals respectively. The expected critical point for agents in the classical public goods game ($i = 1$) is at a synergy factor of 5.0, and here we find this critical point to be shifted to a lower level around 3.0 as expected.

Conclusion

After the initial development of the APGG software different groups of students performed additional experiments with ease, supporting the usability design goal. Extensions to other population structures like graphs were also attempted and proved to be easily achievable in practice, thus underlining the quality of the modular design of the tool. Further, we could accurately replicate already existing results from previous publications using our tool (see Figure 4 and Figure 5). Lastly, we used our tool to conduct original research on another public goods game featuring asymmetric payoff redistribution (Hintze et al., 2020). We look forward to further extending our software and hope other researchers will join our endeavor and benefit from our tool.

Outlook

Since APGG is written in a modular fashion, adding to and expanding on the existing code base should be easy to do, and therefore our tool allows anyone who wants to conduct experiments to write their specific use cases into APGG. Creating these extentions has already been tested (Hintze et al., 2020), and shown to work easily. Further additions can also be contributed to the github repository via pull requests. APGG will remain under further development for future experiments by Jochen Staudacher (Kempten University, Germany) and Arend Hintze (Dalarna University, Sweden).

Acknowledgements

References

Adami, C., Schossau, J., & Hintze, A. (2016). Evolutionary game theory using agent-based methods. *Physics of Life Reviews*, *19*, 1–26. https://doi.org/10.1016/j.plrev.2016.08.015

Bohm, C., Hintze, A., & others. (2017). MABE (modular agent based evolver): A framework for digital evolution research. *ECAL 2017, the Fourteenth European Conference on Artificial Life*, 76–83. https://doi.org/10.7551/ecal_a_016

Fehr, E., & Gächter, S. (2002). Altruistic punishment in humans. *Nature*, *415*(6868), 137–140. https://doi.org/10.1038/415137a

Hardin, G. (1968). The tragedy of the commons: The population problem has no technical solution; it requires a fundamental extension in morality. *Science*, *162*(3859), 1243–1248. https://doi.org/10.1126/science.162.3859.1243

Hintze, A., & Adami, C. (2015). Punishment in public goods games leads to meta-stable phase transitions and hysteresis. *Physical Biology*, *12*(4), 046005. https://doi.org/10.1088/1478-3975/12/4/046005

Hintze, A., Staudacher, J., Gelhar, K., Pothmann, A., Rasch, J., & Wildegger, D. (2020). Inclusive groups can avoid the tragedy of the commons. *Scientific Reports*, *10*(1), 1–8. https://doi.org/10.1038/s41598-020-79731-y

Lenski, R. E. (2017). Experimental evolution and the dynamics of adaptation and genome evolution in microbial populations. *The ISME Journal*, *11*(10), 2181–2194. https://doi.org/10.1038/ismej.2017.69

McGinty, M., & Milam, G. (2013). Public goods provision by asymmetric agents: Experimental evidence. *Social Choice and Welfare*, *40*(4), 1159–1177. https://doi.org/10.1007/s00355-012-0658-2

Mildenberger, M. (2019). *The tragedy of the tragedy of the commons*. https://blogs.scientificamerican.com/voices/the-tragedy-of-the-tragedy-of-the-commons/.

Miller, B. L., Goldberg, D. E., & others. (1995). Genetic algorithms, tournament selection, and the effects of noise. *Complex Systems*, *9*(3), 193–212. https://doi.org/10.1162/evco.1996.4.2.113

Richter, D. J. (2019). *Evolutionary computational modeling in a 3D physics environment* [Bachelor's Thesis, Kempten University]. https://doi.org/10.13140/RG.2.2.27100.72322/2

Smith, J. E., Memenis, S. K., & Holekamp, K. E. (2007). Rank-related partner choice in the fission–fusion society of the spotted hyena (crocuta crocuta). *Behavioral Ecology and Sociobiology*, *61*(5), 753–765. https://doi.org/10.1007/s00265-006-0305-y