An Evolutionary Signaling Model of the Emergence of Norms

Robert Lindgren

December 14, 2015

1 Introduction

Social norms are often modeled as equilibria in coordination games. This works well for describing norms such as “Drive on the right side of the road,” for which there is a clear and direct connection between an agent’s decision and the outcome of that decision. If a person drives on the right side of the road, they will likely make it to their destination safely. Otherwise, the likely outcome is a head-on collision. Many norms are far more arbitrary, however, and do not fit this model. It is not clear what direct benefits are to be had by dressing in a particular style, using the correct fork at a nice dinner, or making a display of one’s patriotism, and these phenomena do not lend themselves to being modeled as coordination games.

Eric Posner proposes in *Law and Social Norms* [3] that we think of these norms instead in terms of their signaling benefits. He presupposes that much of our social existence can be modeled successfully as an iterated Prisoner’s Dilemma, such that a person’s reputation as cooperative or uncooperative may significantly impact their ability to get along in life. Posner argues that our adherence to norms like fashion and manners signals our cooperativeness to others and enhances our prospects in the daily situations that we encounter. In this paper, we formalize a variation on Posner’s signaling model and explore what initial conditions are necessary for the emergence of what we will call “signaling norms.”

Building on Posner’s approach, we propose that signaling norms originate with arbitrary correlations between actions in a particular situation and the status of the actors as good or bad cooperative partners. We formalize this idea as an evolutionary game in which a single population engages in two distinct subgames. There are two types of players, “good” and “bad” cooperators. The first subgame is a complete information coordination game in which all actions are public. While agents are not aware of other agents' types as would typically be the case in a complete information game, types are irrelevant and do not affect payoffs in this subgame. The second is an incomplete information game in which all actions are private. In this second subgame, "good" types have a payoff matrix that encourages cooperation, while "bad" types have a payoff matrix that discourages it. All agents are aware of the population-level correlations between actions in the perfect information game and agent types. Making this information available to agents allows for an action in the complete information game to serve as a signal of the agent's type.

We analyze a set of games with randomized initial parameters to determine whether arbitrary correlations between types and $\mathbb{G}_A$ actions can affect the game's dynamic. We are unable to distinguish between the effect of arbitrary correlation and the effect of the overall distribution of strategies, so the model will have to be revised.

Several papers have been crucially important to the development of this project, both in its current and in future iterations. Mengel [2] studies the process by which two agents, engaged in many games simultaneously, learn the actions of these games and learn to partition this set of games. He finds that learning across games may destabilize Nash equilibria, even when the reasoning costs are made arbitrarily small. We draw from this paper the notion that learning may be approximated by the standard replicator dynamic. Sawa and Zusai [4] investigate imitative evolutionary dynamics in a population with heterogenous aspiration levels. They find that, in the long run, the distribution of strategies in the population becomes independent of the distribution of aspiration levels, allowing them to import tools commonly used to analyze games with homogenous aspiration levels into the analysis of games with heterogenous aspiration levels. Future work will utilize this approach, seeking properties of more well-understood games that might be shared by our model, allowing for the development of an analytic model. Finally, Isaac's paper [1] on the writing of agent-based simulations proved to be an enormously helpful tutorial on the use of Python for this application.

2 Model

A population consists of 1000 agents who are randomly paired in each round to play one of two subgames, $\mathbb{G}_A$ and $\mathbb{G}_B$. Agents are either type θ_1 or type θ_2 with μ indicating the proportion of all players who are type θ_1. $\mathbb{G}_A$ is a symmetric, complete-information game with the following payoff matrix:

$$v : \begin{Bmatrix} e_{a_1a_1} & e_{a_1a_2} \\ e_{a_2a_1} & e_{a_2a_2} \end{Bmatrix} = \begin{Bmatrix} 3 & 0 \\ 0 & 3 \end{Bmatrix}. \tag{1}$$

Note that $e_{a_1a_1}$ indicates the payoff an agent receives when they play action a_1 against an opponent also playing action a_1. The action set of $\mathbb{G}_A$ is $\{a_1, a_2\}$, with $x_1 \in [0,1]$ giving the distribution of type θ_1 agents playing a_1 and $x_2 \in [0,1]$ giving the distribution of type θ_2 agents playing a_1. Agents choose the best reply as determined by the following payoff function

$$v^{\theta_k}_{a_ib_i} = [(x_1 + x_2)e_{a_ia_1} + ((1 - x_1) + (1 - x_2))e_{a_ia_2}] + u^{\theta_k}_{a_ib_i}. \tag{2}$$

This sum describes the total payoff that a type θ_k agent with a current $\mathbb{G}_B$ action of b_i will expect to get from choosing action a_i. The first term of the sum is the direct payoff from $\mathbb{G}_A$, while the second term, $u^{\theta_k}_{a_ib_i}$, is the indirect payoff an agent can expect from $\mathbb{G}_B$, given their choice in $\mathbb{G}_A$. For the sake of describing the indirect payoff, let $z^{\theta_k}_{a_i}$ be the proportion of type θ_k agents for whom the following two statements are true: 1) their most recent opponent in $\mathbb{G}_B$ was an agent whose most recent action in $\mathbb{G}_A$ was a_i, and 2) they played b_1 against this opponent. The following equation gives the indirect payoff of action a_i in $\mathbb{G}_A$ for an agent of type θ_k

$$u^{\theta_k}_{a_ib_i} = \mu[z^{\theta_1}_{a_i} f_{\theta_kb_ib_1} + (1 - z^{\theta_1}_{a_i})f_{\theta_kb_ib_1}] + (1 - \mu)[z^{\theta_2}_{a_i} f_{\theta_kb_ib_1} + (1 - z^{\theta_2}_{a_i})f_{\theta_kb_ib_1}]. \tag{3}$$

The indirect payoff function is crucial to the signaling aspect of the game. It allows agents to use information about past population behavior ($z^{\theta_k}_{a_i}$) to draw conclusions about how others will react to them in the future, depending on the action they choose in the current $\mathbb{G}_A$ round. As will be described below, agents playing $\mathbb{G}_B$ use their opponent's past behavior in $\mathbb{G}_A$ to determine the probability of their opponent being of one type or the other. Therefore, an agent's choice of action in $\mathbb{G}_A$ can affect future $\mathbb{G}_B$ payoffs. The indirect payoff function is designed so that agents can predict what this effect will be.

$\mathbb{G}_B$ is an incomplete-information game. The payoff matrix for type θ_1 agents is:

$$u_{\theta_1} : \begin{Bmatrix} f_{\theta_1b_1b_1} & f_{\theta_1b_1b_2} \\ f_{\theta_1b_2b_1} & f_{\theta_1b_2b_2} \end{Bmatrix} = \begin{Bmatrix} 3 & 0 \\ 0 & 3 \end{Bmatrix}. \tag{4}$$

while the payoff matrix for type θ_2 agents is:

$$u_{\theta_2} : \begin{Bmatrix} f_{\theta_2b_1b_1} & f_{\theta_2b_1b_2} \\ f_{\theta_2b_2b_1} & f_{\theta_2b_2b_2} \end{Bmatrix} = \begin{Bmatrix} 0 & 3 \\ 3 & 0 \end{Bmatrix}. \tag{5}$$

The action set of $\mathbb{G}_B$ is $\{b_1, b_2\}$, with $y_1 \in [0,1]$ giving the distribution of type θ_1 agents playing b_1 and $y_2 \in [0,1]$ giving the distribution of type θ_2 agents playing b_1. The payoff function for $\mathbb{G}_B$ is

$$\begin{aligned} u_{b_i}^{\theta_k} &= P(\theta_1|a_i)[y_1(f_{\theta_k b_i b_1}) + (1-y_1)(f_{\theta_k b_i b_2})] \\ &\quad + P(\theta_2|a_i)[y_2(f_{\theta_k b_i b_1}) + (1-y_2)(f_{\theta_k b_i b_2})]. \end{aligned} \tag{6}$$

In the equation above, $P_{\theta_1 a_i}$ and $P_{\theta_2 a_i}$ are the inferred conditional probabilities that an opponent is type θ_1 or θ_2, respectively, given that their most recent action in $\mathbb{G}_A$ was a_i. These probabilities are calculated using one of the following formulas, all based on Bayes' Theorem. Which formula is used depends on the agent's type and their opponent's most recent action in $\mathbb{G}_A$

$$\begin{aligned} P(\theta_1|a_1) &= \frac{\mu x_1}{\mu x_1 + (1-\mu)x_2} \qquad & P(\theta_1|a_2) &= \frac{\mu(1-x_1)}{\mu(1-x_1)+(1-\mu)(1-x_2)} \\ P(\theta_2|a_1) &= \frac{(1-\mu)x_2}{\mu x_1 + (1-\mu)x_2} \qquad & P(\theta_2|a_1) &= \frac{(1-\mu)(1-x_2)}{\mu(1-x_1)+(1-\mu)(1-x_2)}. \end{aligned} \tag{7}$$

We use the following *pairwise proportional imitation* revision protocols:

$$\begin{aligned} r_{a_1}^{\theta_k} &= (1-x_k)[\tilde{v}_{a_2}^{\theta_k} - v_{a_1}^{\theta_k}] \qquad & r_{b_1}^{\theta_k} &= (1-y_k)[\tilde{u}_{b_2}^{\theta_k} - u_{b_1}^{\theta_k}] \\ r_{a_2}^{\theta_k} &= x_k[\tilde{v}_{a_1}^{\theta_k} - v_{a_2}^{\theta_k}] \qquad & r_{b_2}^{\theta_k} &= y_k[\tilde{u}_{b_1}^{\theta_k} - u_{b_2}^{\theta_k}]. \end{aligned} \tag{8}$$

The probability of revision for a type θ_k agent whose most recently played action in $\mathbb{G}_A$ is a_1 is denoted by $r_{a_1}^{\theta_k}$. All other cases are similarly indicated.

3 Simulation Protocol

In order to test the model, we programmed an Agent-Based Simulation (ABS) in Python. What distinguishes an ABS from other simulations is that "the outcomes of interest result from the repeated interaction of autonomous actors," [1] what we are calling here "agents." These agents have their own feasible sets from which to choose an action, and they do so on the basis of both their internal states and the state of the aggregate population.

Python is a high-level, object-oriented, general purpose programming language. The code is naturally readable and amenable to designing the kind of autonomous agents necessary for an ABS [1].

3.1 Agents

Each agent has a state and a set of "move methods." An agent's state tracks the agent's type, most recent actions in each subgame, most recent payoffs in each subgame, and the last $\mathbb{G}_A$ action of their most recent $\mathbb{G}_B$ opponent at the time of their matchup. This last component of the agent's state is necessary to calculate indirect payoffs in $\mathbb{G}_A$.

Agents have one move method for each subgame, $\mathbb{G}_A$ and $\mathbb{G}_B$. In each move method, the deciding agent determines the probability of revision by sampling a random agent of the same type. If the sampled agent's most recent strategy in the subgame being played is the same as the deciding agent's most recent strategy, then the deciding agent will not revise. Otherwise, the difference between the two agents' most recent payoffs is used to calculate the probability of revision according to the second term of the appropriate equation in (7).

This process of random selection instantiates the first term of the pairwise proportional imitation protocol as an algorithm rather than a calculation. For example, let the subgame be $\mathbb{G}_A$ and the deciding agent be a type θ_k player whose most recent action in $\mathbb{G}_A$ was a_1. If the deciding agent randomly samples a player from a list of type θ_k agents, the probability that this player's most recent action in $\mathbb{G}_A$ was a_2 is equal to $(1 - x_k)$, the first term of the relevant revision probability function.

When an agent chooses to change a subgame action according to the revision protocol above, they do so using a best reply method. The deciding agent calculates the expected payoff of each action using the appropriate subgame payoff function, (2) or (5), and chooses the action that gives the maximum payoff.

3.2 The Game

At the beginning of each game initial values are assigned to the population state parameters: μ, x_1, x_2, y_1, and y_2. These values are chosen from a uniform distribution between 0 and 1 using Python's pseudo-random number generator. Dictionaries (unordered sets of "key: value" pairs) are generated to store the set of all players and the sets of all values that describe the population state at the end of each round. A population of 1000 agents is generated using the initial parameter values. The simulation then runs for 1000 rounds.

In each round, 5% of the population is randomly selected, paired with an opponent, and assigned a subgame. There are two reasons for the low rate of revision in the population. First, it avoids large and abrupt changes to the population state that would make it difficult to develop a continuous-time version of the model, a goal of future research. Second, it reflects the reality of the social phenomenon being modeled. Individuals in a society do not continually reconsider every behavior in which they engage.

Agents then play their assigned subgames using the appropriate move method (as described in the section, Agents). The variables that describe the game state, μ, x_1, x_2, y_1, y_2, are calculated, their values at the end of the round are stored, and another round begins.

4 Analysis

4.1 Pure Equilibria

Out of 1000 simulations, pure equilibria were reached in 38.4% of cases. These simulations converged to four different pure equilibria, given in the table below.

	x_1	x_2	y_1	y_2
Eq. 1	0	0	0	1
Eq. 3	0	0	1	0
Eq. 4	1	1	0	1
Eq. 2	1	1	1	0

The equilibria reached were primarily determined by two initial values, the proportion of all agents playing action a_1 in $\mathbb{G}_A$ $(x_1 + x_2)$ and the proportion of all agents playing action b_1 in $\mathbb{G}_B$ $(y_1 + y_2)$. Let $X_1 = (x_1 + x_2)$ and $Y_1 = (y_1 + y_2)$. The table below describes which initial values lead to which equilibria. These results are consistent with what one would expect from $\mathbb{G}_A$ and $\mathbb{G}_B$ being played in separate populations.

Equilibria	Initial Values
Eq. 1	$X_1 < 0.5$ and $Y_1 < 0.5$
Eq. 3	$X_1 < 0.5$ and $Y_1 > 0.5$
Eq. 4	$X_1 > 0.5$ and $Y_1 < 0.5$
Eq. 2	$X_1 > 0.5$ and $Y_1 > 0.5$

The primary question explored by our model is this: can an arbitrary correlation between agent types and $\mathbb{G}_A$ actions determine the equilibrium to which the population converges? We sought to test this by asking whether particular ranges of x_k

initial values (the proportion of type θ_k agents who are playing action a_1) correlate with particular equilibrium outcomes. We are unable to give a satisfactory answer to this question with the simulation data that has been collected so far.

There is a strong correlation between values of x_1 and particular equilibrium outcomes. 90% of games that are initialized with a value of $x_1 < 0.5$ result in either Eq. 1 or Eq. 3, both equilibria in which the entire population plays a_2 in $\mathbb{G}_A$. Similarly, 90% of games that are initialized with a value of $x_1 > 0.5$ result in either Eq. 2 or Eq. 4, both equilibria in which the entire population plays a_1 in $\mathbb{G}_A$.

However, x_1 seems to determine these outcomes merely through its effect on X_1. 99% of games that are initialized with a value of $X_1 < 0.5$ result in either Eq. 1 or Eq. 3. 99% of games that are initialized with a value of $X_1 > 0.5$ result in either Eq. 2 or Eq. 4.

5 Conclusion

No conclusions can be drawn at this time regarding the hypothesis motivating this project. Because all initial values were randomly assigned, the initial values of X_1 correlated strongly with the initial values of x_1, on which X_1 depends. This made it impossible to distinguish between the effects of x_1 and X_1 on the resulting equilibria.

However, the work done thus far has prepared all the tools necessary for future analysis to proceed quickly. With not only the simulation complete, but also a number of scripts designed specifically for the analysis of the data this simulation produces, further work requires nothing more than the time necessary to run the simulation with the required initial parameter values and to analyze the results.

5.1 Future Research

Future research will focus on testing a variety of permutations of the initial game parameters. First, we will test the effect of varying x_1 while holding X_1 constant. This will allow us to distinguish the effects of the initial value of x_1 from those of X_1. We will also test the effect of weighting the payoffs of $\mathbb{G}_B$ as greater than those of $\mathbb{G}_A$. This will reveal at what weight the effect of the indirect payoff in the $\mathbb{G}_A$payoff function overpowers the effect of the direct payoff. Lastly, we will double the rate of revision in the population to ensure that equilibrium is reached in 1000 rounds or less in each game.

References

[1] A. Isaac, Simulating Evolutionary Games: A Python-Based Introduction, Journal of Artificial Societies and Social Simulation 11 (2008).

[2] F. Mengel, Learning across games, Games and Economic Behavior 74 (2012) 601-619.

[3] E. Posner, *Law and Social Norms*. Harvard University Press, 2000.

[4] R. Sawa, D. Zusai, Evolutionary imitative dynamics with population-varying aspiration levels, Journal of Economic Theory 154 (2014) 562-577.