Influencing convergence speed in Lewis signalling games

Marcel Ruland

hand-in date: January 21, 2019

Contents

1 Introduction

Lewis signalling games (henceforth LSG) were first introduced by Lewis (1969). Following Barrett (2007, p. 530 ff.), they works as follows: There is a set of states of the world S, a set of signals (or terms) T, and a set of acts A. There is a mapping from acts to states of the world, such that every act corresponds to a state of the world.[1] There is a *sender*, who can observe the current state of the world and a *receiver*, who

[1]Barrett (2007) leaves unclear if that mapping must be bijective (for every a there is exactly one corresponding s and for every s there is at least one corresponding a), but every signalling game involved in this essay fulfils this condition.

cannot observe the current state of the world. In each round exactly one state of the world $s \in S$ holds, i.e. is the current state of the world. The sender will observe the current state of the world and then choose a signal $t \in T$ and send it to the receiver. The receiver will observe the signal and then perform an act $a \in A$. A round is won if the act a matches the current state of the world s and lost if it does not. Both sender and receiver know about whether the round was a success or a failure and adapt their strategies for choosing signals and acts according to some learning function. Commonly, states of the world are distributed uniformly and sender and receiver start out with randomly choosing a signal and an act respectively, but this is not a formal requirement for the game to count as an LSG.

Agent-based modelling (henceforth ABM) is a modelling paradigm that has its roots in cellular automata and complexity theory (Heath and Hill, 2014). Following Grimm and Railsback (2005); Railsback and Grimm (2011) in an ABM, there are individual *agents* representing individual entities of one or several kinds (such as for instance cars, sheep, or viruses). These entities interact with each other, with an *environment*, or (typically) both. The environment itself has characteristics which influence the agents. It can play a major role in the simulation (such as a forest providing food and shelter to a population of animals) or be virtually inexistent (as is the case in the present essay, see section 2). Typical applications include (but are by far not limited to) the simulation of traffic flow in a road network, spreading of a virus in a population (human or non-human), and changes in real estate prices in a city. The strength of ABM, in comparison to other modelling paradigms, is that by modelling each individual agent, one can observe how the sum of individual actions of agents give rise to phenomena which were not explicitly programmed into the model (such as the creation of ant corridors, which are not intentionally created by any individual ant, but are a byproduct of the fact that ants directly follow each other in straight lines, Wilensky (1997)).

This essay combines both paradigms. An ABM will be created where there is a population of senders and a population of receivers. Initially the number of signals available will be less than the number of states of the world and actions, but will be increased with time. The aim is to determine how the time at or the condition on which more symbols will be made available influences the speed of convergence towards *perfect communication*. Perfect communication in an LSG is achieved if "each state of the world corresponds to a term in the language and each term corresponds to an act that matches the state of the world, so each signal leads to a successful action" (Barrett, 2007, p. 530, there referred to as "perfect Lewis signalling system"). The model, described in detail in section 2, has some functionality for choosing parameters which will not be used in this essay, for simple reasons of scope.

2 Model description

The description of the model follows the *Overview, Design concepts, and Details* protocol (Grimm et al., 2006, 2010, henceforth ODD).

2.1 Purpose

The model is run to understand how, based on elapsed time and/or quality of communication, increasing the number of symbols available in an LSG accelerates or slows down convergence towards perfect communication.

2.2 Entities, state variables, and scales

There are two entities, `senders` and `receivers`, the populations of which are equal in size. The `senders` have two state variables. `urns` is a list of integer lists for implementing the urns in an urn learning function, see subsubsection **??**. Every nested list represents one urn, the integers within it represent balls to be drawn from it. `chosen-signal` is an integer corresponding to the signal the sender will send in the current round. `receivers` are structured almost identically. The urns are also lists of integer lists, but are structured differently (see again subsubsection **??**). Instead of a `chosen-signal` variable, they have a `chosen-action` variable. In addition, they also have a `received-signal` variable, which stores the signal received from a sender. Senders and receivers are connected by `links`, which carry a signal in their `signal` variable. This `signal` variable can be read only by the sender and the receiver corresponding to the link. The environment is simply a state of the world, implemented as a global integer variable, and nothing else. The following parameters of the model can be modified:

- number of possible states of the world (the number of possible actions is always equal to this value)
- number of available signals
- population size (equal for senders and receivers)
- number of balls added in case of success
- number of balls removed in case of failure

2.3 Process overview and scheduling

Every iteration begins with randomly choosing a state of the world and creating a randomised one-to-one mapping from senders to receivers. Links are then created and connected to senders and receivers according to this mapping. Senders have access to the state of the world and choose a signal based on it, which they then pass on to their respective link. Once the links carry the sent signal, receivers have access to it and read it. They choose an action based on the received signal and check whether the action matches the state of the world. Receivers then pass on an arbitrary, conventional value (here 42) on to the links, which the receivers can read and interpret as *success*. Lastly, senders and receivers run a learning function according to whether the round was a success or a failure.

The reader will notice that the previous paragraph seems to contradict the above given definition for an LSG. Receivers cannot observe the state of the world, yet in this model description it is the receivers who check whether their performed action corresponds to the current state of the world. While this may seem a violation of the rules of an LSG, it is in fact not. Once signals and actions have been chosen, the simplest way of implementing the passing on of the correct success/failure information to the correct senders/receivers just happened to be letting the receivers check the global state of the world variable and then informing the correct senders via the still-existent links. At no other time do the receivers access the state of the world variable, thus not violating the definition of an LSG.

2.4 Design concepts

Emergence The number one (and only) emergent behaviour predicted here is that of perfect communication. From an initially random distribution of chosen signals and chosen actions emerges (ideally) a perfect one-to-one mapping from states of the world to signals and to actions corresponding to the states of the world.

Adaptation Senders have as many urns as there are possible states of the world. Receivers have as many urns as there are available signals. A sender chooses a signal by randomly drawing a ball from the urn corresponding to the current state of the world. A receiver chooses an action by randomly drawing a ball from the urn corresponding to the received signal.

Objectives Senders and receivers aim to maximise the percentage of rounds in which communication is successful, i.e. in which the current state of the world matches the action chosen by the receiver.

Learning Senders start off with one ball corresponding to every possible signal in each of their urns. In a game with three possible states of the world and two available signals, the initial configuration will look like this: `[[0 1] [0 1] [0 1]]` Receivers start off with one ball corresponding to every possible action in each of their urns. In the same game, their initial urn configuration looks as follows: `[[1 2 3] [1 2 3]]` In case of success, senders and receivers will add more balls with the chosen signal/action to the consulted urn. In case of failure, they will remove balls with the chosen signal/action. The number of added or removed balls is determined by the value of two parameters.

Prediction Agents cannot predict the future values of any variables.

Stochasticity The state of the world of a round is determined by randomly drawing a value from a uniform distribution. Furthermore, within one of the steps described in subsection 2.3, the order in which senders and receivers perform their actions is also randomised.

Observation A game is considered to converge to perfect communication if 100 consecutive rounds result in a communication quality > 0.8 within one million rounds. The 0.8 threshold is in line with Barrett (2006, 2007, p. 533). In Barrett (2006, sec. 2, unpaginated preprint), the author states that "After 10^6 plays the ratio of successful actions to the number of plays, the signal success rate, is typically better than 0.999", but leaves open how this is measured. In trial runs, the condition of 100 consecutive rounds has proven to be a reliable indicator for convergence and has therefore been chosen.

2.5 Initialisation

The model is initialised by creating a population, setting the number of available actions equal to the number of possible world states, and resetting the counter keeping track of the number of rounds played.

3 Results

The results section is focused on two main questions. The most obvious one, and the focus of subsection 3.1 is how population size influences convergence speed in an LSG, as this parameter is the main novelty that ABM brings to the game. A second question is the introduction of signals into an ongoing game. Subsection 3.2 investigates how beginning a game with just one signal, and then introducing more signals as the rounds progress, influences convergence speed.

3.1 Population size

In the first experiment, to investigate the effect of population size on convergence speed, the model was run with the following parameter settings:

```
num-signals = 10
num-remove-balls = 1
num-add-balls = 5
num-world-states = 10
```

`population-size` was incremented step-wise from 1, 2, 3, ..., 98, 99, 100 and ten runs were performed for every value.[2]

The value for `num-add-balls` may seem unusually high. Small populations tend to not converge if the value for this parameter is too high, whereas large populations tend to not converge if it is too low. Assigning a value of 5 to `num-add-balls` has proven to reliably lead to convergence with smaller and larger populations. In fact, out of the one thousand runs for this experiment, only four runs did not converge within one million iterations. The respective population sizes were 3, 4, 12, and 48.

The predicted result here is that with increasing population size the number of iterations until convergence will also increase. If a given sender and a given receiver are paired in the random mapping and communicate successfully, then they will adapt their strategies only to each other. This is not an issue if population size is equal to one, because the same sender and receiver will be paired over and over again. But as population size increases, the likelihood of being paired with the same sender/receiver decreases and therefore iterations are needed not only for convention to emerge within a single pair but within many pairs and eventually across an entire population.

Figure 1 shows the mean and median convergence speed for all values of `population-size` respectively. One can see that the mean values consistently are at least as high, but usually higher, than their median counterparts. This indicates rare runs with very late convergence, increasing the means more than the medians.[3] Visually, the plot seems to affirm the predicted statistical relationship between the two variables and indeed there appears to be a correlation of $R^2 = 0.55$. One would obviously expect this relation to be logarithmic, because for every pair added to the population the population grows by a factor of $\frac{1}{n+1}$, but the simulation is not nearly extensive enough to affirm or deny this hypothesis.

[2] 10 runs are, admittedly, less than one would have liked given the extent of variation shown in figure reffig:pop; unfortunately the complexity of running the model with high values of population-size was such that e.g. 100 runs for every value were just not feasible.

[3] All other experiments have shown similar characteristics with rare, high outliers. I have therefore decided to only consider medians as a measure of centrality from now on, which are much less affected by outliers as compared to means.

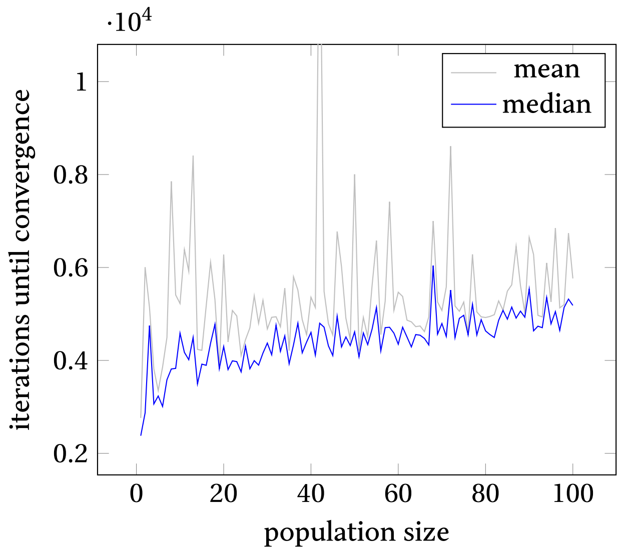

Figure 1: Effect of population size on convergence speed; x-axis indicates population size, y-axis indicates mean and median number of iterations until convergence to perfect communication; the outlier not in view for a population size of 42 has a value of 14450; $R^2 = 0.55$.

3.2 Introducing signals into an ongoing LSG

Macro scale

A second experiment is concerned with the effects of delaying the introduction of signals into the game. The model was run with the following parameter settings:

```
population-size = 10
num-remove-balls = 1
num-add-balls = 3
num-world-states = 10
```

The number of available signals at the beginning of a game was always 1. More signals were then introduced after a certain amount of iterations–given by the parameter `signals-interval`–had passed. The values tested for `signals-interval` were 100, 200, 300, ..., 4000, 4100, 4200. For every value, 100 runs were performed.

Figure 2 shows a somewhat dull linear relationship ($R^2 = 0.97$). This is, in hindsight, no surprise as figure 1 shows a convergence median of 4585 rounds for a population of size 10. Introducing signals to the game long after the 4585th round has passed is bound to slow convergence down. The linear relationship one can see here is simply representing the linear fashion in which the value of `signals-interval` has been increased. A truly interesting result would be a later introduction of signals

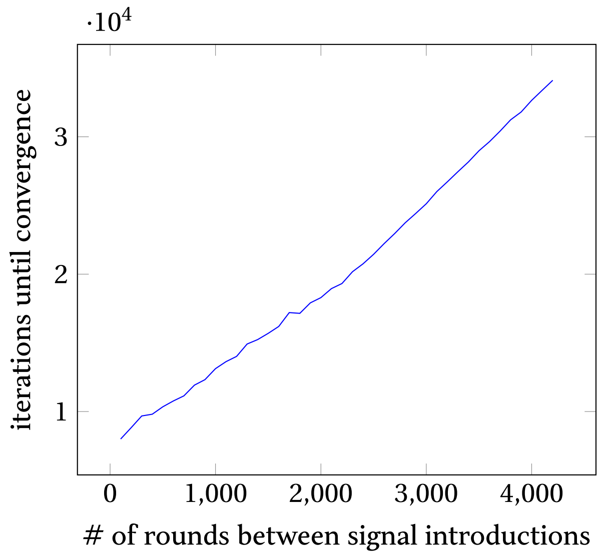

Figure 2: Effect of later signal introduction on convergence speed; x-axis indicates the number of iterations that passed before another signal was introduced; y-axis indicates the median number of iterations until convergence; $R^2 = 0.97$

that speeds up median convergence speed, instead of slowing it down. In order to potentially produce such results it is necessary to zoom in and choose much smaller values for `signals-interval`.

But all is not lost. Figure 3 shows how communication quality evolved along on of the games played as part of the second experiment, and brings much less expected results to light. In the game displayed, a new signal was introduced every 4200 rounds. One can see that communication quality remains close to 0.1 without significant variation up until around the 25000th iteration. Around this iteration, communication quality experiences a sudden increase.

Micro scale

The model was run with the following parameters:

```
population-size = 1
num-remove-balls = 1
num-add-balls = 2
num-world-states = 10
```

`signals-interval` was again increased step-wise, but this time on much smaller scales. First, from 1 to 100, with 100 runs being performed for every value. Second,

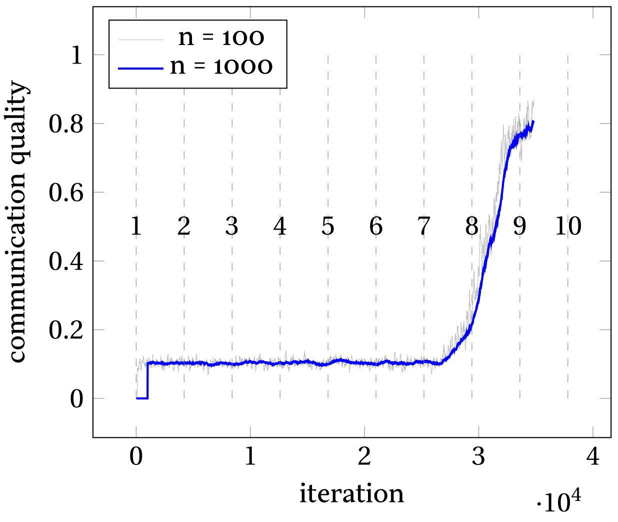

Figure 3: Communication quality in a game where new signals were introduced with a frequency of 1 signal per 4200 rounds played; x-axis indicates iterations, y axis indicates communication quality; grey line indicates a moving average over 100 rounds, red line a moving average over 1000 rounds; vertical dashed lines indicate when new signals were introduced; for simplicity only values for every 10th round are shown. Notice how communication quality lingers somewhere around 0.1 until the seventh signal is introduced, where it suddenly increases dramatically.

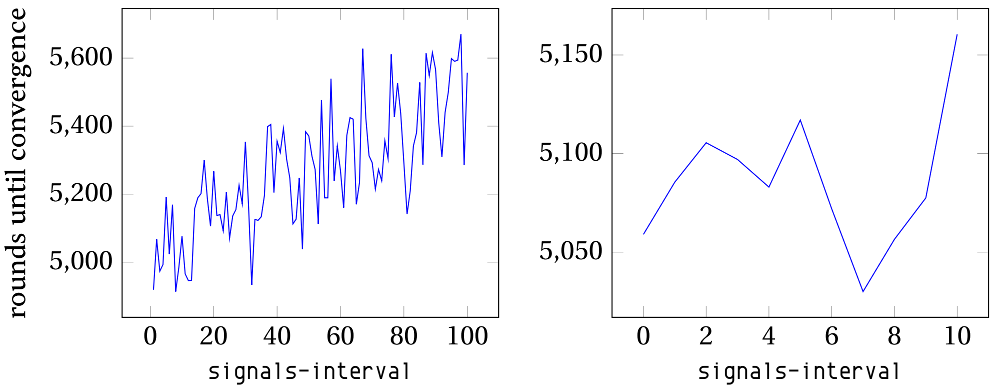

Figure 4: Effects of later introduction of signals on a micro scale; x-axis indicates value of `signals-interval`, y-axis indicates median iterations until convergence to perfect communication. *Left:* `signals-interval` incremented from 0 to 100 with 100 rounds played per value (a value of 0 indicating all signals being available from the beginning on), $R^2 = 0.76$. *Right:* `signals-interval` incremented from 1 to 10 with 1000 rounds played per value, $R^2 = 0.04$.

from 1 to 10, with 1000 runs being performed for every value. The results are plotted in figure 4 on the left and right respectively. On the left-hand plot, a linear relation between signal introduction and convergence speed is once again visible. Statistics confirm this visual hunch with $R^2 = 0.76$. The right-hand plot shows a somewhat more interesting picture. The increase in convergence time experiences a drop for `signals-interval = 7` that even goes below the convergence speed for `signals-interval = 0` (i.e. all signals are available from the beginning of the game on). Unfortunately, testing the distribution of convergence speeds with `signals-interval = 7` for significantly low values against the distribution with `signals-interval = 0` results in $p = 1.0$ with both a Student-t test and a Wilcoxon signed-rank test (the latter does not assume the data to be normally distributed; for the data to be in fact pass tests of normality, e.g. Shapiro-Wilk, one would have to exclude several high outliers).

3.3 Reproducing ? results

3.4 Concluding remarks

4 Discussion

(Wilensky, 1999)

References

BARRETT, Jeffrey A. (2006): "Numerical simulations of the lewis signaling game: Learning strategies, pooling equilibria, and the evolution of grammar." *UC Irvine Institute for Mathematical Behavioral Sciences* preprint.

BARRETT, Jeffrey A. (2007): "Dynamic partitioning and the conventionality of kinds." *Philosophy of Science* **74** (527–546).

GRIMM, Volker; BERGER, Uta; BASTIANSEN, Finn; ELIASSEN, Sigrunn; GINOT, Vincent; GISKE, Jarl; GOSS-CUSTARD, John; GRAND, Tamara; HEINZ, Simone K.; HUSE, Geir; HUTH, Andreas; JEPSEN, Jane U.; JØRGENSEN, Christian; MOOIJ, Wolf M.; MÜLLER, Birgit; PE'ER, Guy; PIOU, Cyril; RAILSBACK, Steven F.; ROBBINS, Andrew M.; ROBBINS, Martha M.; ROSSMANITH, Eva; RÜGER, Nadja; STRAND, Espen; SOUISSI, Sami; STILLMAN, Richard A.; VABØ, Rune; VISSER, Ute; and DEANGELIS, Donald L. (2006): "A standard protocol for describing individual-based and agent-based models." *Ecological Modelling* **198**: 115–126.

GRIMM, Volker; BERGER, Uta; DEANGELIS, Donald L.; POLHILL, J. Gary; GISKEE, Jarl; and RAILSBACK, Steven F. (2010): "The ODD protocol: A review and first update." *Ecological Modelling* **221**: 2760–2768.

GRIMM, Volker and RAILSBACK, Steven F. (2005): *Individual-Based Modeling and Ecology*. Princeton, NJ: Princeton University Press.

HEATH, B. L. and HILL, R. R. (2014): "Some insights into the emergence of agent-based modeling." In: TAYLOR, Simon Je, ed., "Agent-based modeling and simulation," 32–44. New York, NY, USA: Palgrave MacMillan.

LEWIS, David (1969): *Convention*. Cambridge, MA: Harvard University Press.

RAILSBACK, Steven F. and GRIMM, Volker (2011): *Agent-based and individual-based modeling: A practical introduction*. Princeton, NJ: Princeton University Press.

WILENSKY, Uri (1997): *NetLogo ant lines model*. Evanston, IL: Center for connected learning and computer-based modeling, Northwestern University.
URL: `http://ccl.northwestern.edu/netlogo/models/AntLines`

WILENSKY, Uri (1999): *NetLogo*. Evanston, IL: Center for connected learning and computer-based modeling, Northwestern University.
URL: `http://ccl.northwestern.edu/netlogo/`