Norms Evolution under Multi-Agent Systems

THESIS

Submitted in partial fulfilment
of the requirements for the degree of
DOCTOR OF PHILOSOPHY

by

ANKUR
ID NO. 2018PHXF0507H

Under the Supervision of
Dr. DUSHYANT KUMAR

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI
2024

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

CERTIFICATE

This is to certify that the thesis entitled **"Norms Evolution under Multi-Agent Systems"** and submitted by **ANKUR**, ID No **2018PHXF0507H** for award of Ph.D. of the Institute embodies original work done by him/her under my supervision.

Signature of the Supervisor:
Name in capital letters: DR. DUSHYANT KUMAR
Designation: Assistant Professor

Date: 11/04/2024

Acknowledgements

This work was only possible with the valuable guidance my supervisor, Dr. Dushyant Kumar provided. He has frequently supported me through the many challenges I have faced while conducting the work related to this thesis. I would also like to acknowledge the supportive environment in the Department of Economics and Finance and specifically Dr. Sudatta Banerjee, Dr. Dushyant Kumar, and Dr. Rishi Kumar, who served as heads of the Department during my tenure here. I also express my deep gratitude towards Dr. Bheemeshwar Reddy, Convenor, Doctoral Research Committee and Dr. Bheemeshwar Reddy, Dr. Durgesh Chandra Pathak, members of my Doctoral Advisory Committee. I thank the Academic-Graduate Studies and Research Division, specifically Prof. Venkata Vamsi Krishna Venuganti, Dean, and Prof. Alivelu Manga Parimi, Associate Dean. I also thank Prof. V. R. Rao, Group Vice Chancellor of BITS Pilani, and Prof. Soumyo Mukherji, Director of BITS Pilani Hyderabad Campus, for continuing to enable work even during the very trying pandemic-ridden times. Finally, I would like to acknowledge the support and patience of my family in dealing with my frequent absence during the time spent developing this work. I sincerely hope that the insights and frameworks developed from this work will help make a difference in computational economics literature.

Ankur

Place: Hyderabad

Date: 11/04/2024

Abstract

Social norms are the societal rules that govern how individuals should behave in a society. Some examples of norms include following traffic manners/etiquettes, e.g., driving on the left/right side of the road, mannerisms regarding greeting or handshaking, exchanging greetings/gifts on festivals or specific occasions, etc. Multiple arguments are given in favour of following norms like rationality of following norms, peer pressure, the threat of social disapproval, signal to the larger audience to comply with the norm, etc. (Young, 2015).

The focus of our research is on norm evolution or emergence. Norm emergence occurs when a significant proportion of agents, say a minimum of *x%*, follow the norm. The value of *x* varies depending on the context and is seen between 35 to 100 across different simulation studies of norms (Savarimuthu & Cranefield, 2011).

The existing literature has focused on how norms evolve and depend upon various parameters. Some of these parameters include the payoff structure of the game, population size, memory length, strength of relations with other agents, time taken to reach a norm, methods which agents follow to update their actions during each period of the game, and randomness in agents' actions (Young & Foster, 1991; Kandori et al., 1993; Young, 1993; Alexander, 2007; Young, 2015). The focus of our research is on norms evolution using computational methods.

Norms evolution from a computational standpoint has been described in the following three different ways in the literature:

A. First, some finite normal-form games are defined along with their payoff matrix. Agents play the game repeatedly for a certain period. Agents' decision on what strategy to choose at any given time depends upon past strategies played. The frequently played strategy is considered a norm strategy (Axelrod, 1986; Shoham & Tennenholtz, 1992; Young, 1993). In this case, the norm strategy is not necessarily a single strategy but an action pair from the payoff matrix, which captures row-player and column-player strategies.
B. Secondly, there is a finite game with specific strategies and the corresponding payoff values. In this approach, we also define population size and the agents' distribution following these strategies before starting the game. Agents interact with other agents over a period, calculate payoffs and possibly adapt their strategies. Agents can use different methods to decide what strategy to choose. It could be a simple imitation of other agents' strategies if following that strategy results in a higher payoff, or it could be more complex, like using neural networks or genetic algorithms (Nishizaki et al., 2009). At the end of the simulation period, we get the information on the agents' revised strategy distribution. The frequently played strategy is considered a norm (Young & Foster, 1991; Axtell et al., 1999; Tesfatsion, 2003; Nande et al., 2020).
C. Another school of thought belongs to incorporating agents' social networks into consideration. This approach defines agents' social networks, and their impact on agents' decisions is evaluated (Alexander, 2007; Young, 2015).

Most of the existing literature performed simulations using a hypothetical game, and it is difficult for the reader to replicate the same results or to check the impact of tweaking parameters. There has been less visibility around how the results have been derived and what tool or programming language has been used to perform simulations. Against this background, we have tried to bring

transparency into the process of simulations and how this can help answer the question of norms evolution.

In response to the literature divided into sections A, B, and C above, we have created a simulation framework addressing all three approaches used in these papers. This thesis aims to create a reusable tool or framework that can then be used to test different possibilities concerning norms evolution computationally. This would help create opportunities for many researchers in economics, social science, computer science, and other related fields. We have executed the reusability objective by creating open-source Python libraries that anyone can use without building and writing the code from scratch. This would also help expand the libraries to make them broader and more accessible by incorporating suggestions from different researchers. To the best of our knowledge, these are the first open-source Python libraries that can be used for this purpose by the end user with very little programming language. These libraries support customization with respect to different games, payoff matrices, response functions, memory length, network types, agent types, simulation time periods etc. In addition, the output of these libraries is in the form of graphs and Excel files which provide detailed and granular information on agent's choices at every step of the simulation exercise. We believe that every user can make use of Excel files and hence is simpler to use and analyze. This also increases the transparency of the simulations performed, which we have found somewhat lacking in the current literature. We believe that these libraries are our humble contribution to the advancement of more computational and agent-based modeling in economics where lack of suitable software tools has been pointed out as one of the challenges lately (Axtell & Farmer, 2022).

Firstly, we created a norms evolution framework using two different response functions or approaches agents can use to decide what action to take. In the first approach, agents choose the strategy having the maximum payoff in response to all possible opponents' strategies from historical interactions. The second approach considers the relative frequencies of different strategies played in the past and uses these as weights to calculate an expected payoff. In this scenario, a finite normal-form game is defined along with its payoff matrix. The action or strategy pair from the payoff matrix being played or proposed most frequently is considered a norm strategy pair. We have created *game-simulator* library, which can be used to assess evolution for any combinations of m*n payoff matrix, memory length, time period, initial states etc. This produces output in the frequency distribution of pairs of the payoff matrix.

In the second method to approach norm evolution, we have considered the scenario where strategies are defined along with their payoff values. We also have the population with a defined percentage share following those strategies. Agents are connected via a specific social network. Agents update their actions or strategies according to the strategies followed by their neighbours. We explain this with the help of the Nash demand game of Axtell et al. (1999). We have created a second library named *multi-agent-decision* which can be used to get the choice distribution in the population. For example, in the Nash demand game with three actions, High, Medium, and Low, if we start the game with some initial percentage distribution of agents following these strategies (say 33% each), we can get the answer on how this distribution would change as agents interact with their neighbours and possibly change their choices. Agents' decision depends upon the number of strategies, initial state, population size, agents' neighbourhood size etc. The output

generated from the library includes information on strategy frequency distribution during each period of the game.

The third possibility where norm evolution is applicable includes when no fixed actions or strategies are defined to start with, and agents need to formulate these choices while interacting with other agents in their neighbourhoods. We explain this scenario with the help of the Naming game of Young (2015). In the Naming game, two agents are selected randomly and shown a picture of a face. Agents need to suggest names for the faces and get rewarded with positive payoffs if they suggest the same names. We have shown that this results in convergence towards 2 or 3 names when agents play the game repeatedly and are connected in the ring network. These selected few names which emerged from the plausible multiple names with agents' interactions are called norms. To generalize the results from this approach to different possibilities of norm evolution where cooperation is rewarded, we created a third python library called *multi-agent-coordination* , when no fixed choices are defined at the beginning of the game.

Once a specific norm is established, it can be displaced by other norms over time. We have shown this with the help of the Nash demand game, where there is an incentive provided to agents to follow low (L) or high (H) strategy to displace medium (M) strategy. We captured the norm displacement possibility in the *multi-agent-decision* library created with the help of a few additional parameters.

The three possibilities and corresponding Python libraries discussed above approach norm evolution using individual agent-based approaches. This implies that the norms that emerged are

derived from agents' interactions with their neighbours, which translates to macro/population-level norms. To assess if following individual agent-based approaches bring in different results compared to results from macro-based approach, we have compared individual agent-based evolution results with macro/aggregative approaches like replicator dynamics. This is explained with the help of a migration game where some agents migrate from a domestic country (following X norms) to a foreign country (following Y norms). We are interested in knowing which norm out of X and Y survive when domestic and foreign country agents coexist in the population in a foreign country. Results show that different social network types influence agents' decisions to follow domestic or foreign country norms. The replicator and other dynamics results show the convergence towards one outcome, either X or Y when one agent type is more social than another. However, the individual agent-based response function approach shows the possibility of the emergence of both the outcomes, X and Y, under some parameter restrictions. We also assessed the impact of network structures on the outcomes. A higher dense network usually leads to convergence towards one outcome. Results are sensitive to the fat-tailedness and clustering of the network in case of lower dense networks. Medium to heavy fat-tailedness can lead to convergence towards one outcome in the absence of high densities. When agents are connected in Barabasi-Albert network structures, results primarily dependent on the strategies followed by "hub" agents.

With these different norm evolution and displacement methods and techniques, we have tried to capture multiple scenarios where agents interact with other agents in their neighbourhood and define the norm. This has wider applications in multiple areas of economics and beyond, like sociology, computer science, anthropology etc.

Table of Contents

Chapter/ Section	Topic	Page No
	List of Tables	12
	List of Figures	13
	List of Abbreviations/Symbols	18
1	**Introduction**	20
1.1	Research Gaps	27
1.2	Objectives	28
1.3	Norms Evolution Frameworks	30
2	**Literature Review**	40
2.1	Norm definition	40
2.2	Norm characteristics	44
2.3	Classical game theory approach	45
2.4	Static approach towards evolution	46
2.5	Dynamic approach towards evolution	49
2.6	Applications of the evolutionary game theory approach	56
2.7	Norms evolution	57
2.8	Role of sanctions in the effective implementation of social norms	65
2.9	Role of randomness in agents' decision making	72
2.10	Role of social networks in evolution and norms	83
2.11	Other factors impacting the evolution of norms	94
2.12	Agent-based modeling	100
3	**Response Functions and Norms Evolution**	105
3.1	Introduction	105
3.2	Literature review	108
3.3	Methodology	110
3.3.1	Exhaustive best response approach	111
3.3.2	Expected payoff approach	118
3.4	Simulation results	125
3.4.1	Prisoner's dilemma	127
3.4.2	Battle of sexes	129
3.4.3	Matching pennies	133
3.4.4	Stag hunt	135
3.4.5	Coordination	139
3.5	Impact of memory size	140
3.6	Applications	152
3.7	A note on the python library created	155
3.8	Conclusion	157
4	**Social Networks and Norms Evolution**	158
4.1	Introduction	158
4.2	Literature review	164
4.3	Norm definition	168

4.4	Naming game and evolution of norms	169
4.4.1	Impact of fixed agents	185
4.4.2	Impact of population size	188
4.4.3	Impact of positive probability of suggesting new name (pN)	189
4.5	Nash demand game and norm evolution	191
4.5.1	Impact of fixed agents	201
4.5.2	Impact of population size	203
4.6	Norm displacement and emergence of new norm	204
4.7	Random networks and norm evolution	210
4.8	A note on python libraries created	228
4.9	Conclusion	230
5	**Dynamic Evolution and Networks**	232
5.1	Introduction	232
5.2	Static and dynamic evolution	236
5.3	Migration game	239
5.4	Best response function framework	242
5.5	Results - Replicator and other dynamics	245
5.6	Results – Best response function approach	258
5.7	Random networks and norm evolution	268
5.7.1	Erdos-Renyi (ER) network	269
5.7.2	Barabasi-Albert (BA) random network	282
5.8	Conclusion	290
	Conclusions	292
	Specific Contributions	297
	Future Scope of Work	302
	References	304
	List of Publications and Presentations	307
	Brief Biography of the Candidate	308
	Brief Biography of the Supervisor	309

List of Tables

Table No	Details
1.1	Coordination game
2.1	Prisoner's dilemma game
2.2	Stag hunt game
2.3	A hypothetical game with evolutionary stable set
2.4	Prisoner's dilemma game with hypothetical payoffs
2.5	A hypothetical game with a weakly dominated strategy
2.6	Rock-Scissors-Paper game with an additional strategy
2.7	Prisoner's dilemma game with 0 payoffs for both defects
2.8	Prisoner's dilemma game for IPD
2.9	Prisoner's dilemma game with a tit-for-tat strategy
2.10	A hypothetical 2*2 symmetric game
3.1	A hypothetical 2*2 game
3.2	History and action pairs for exhaustive best response approach
3.3	Expected payoffs in period 3
3.4	History and action pairs for expected payoff approach
3.5	Prisoner's dilemma
3.6	Battle of sexes
3.7	Battle of sexes (modified 1)
3.8	Battle of sexes (modified 2)
3.9	Matching pennies
3.10	Stag hunt
3.11	Stag hunt (modified 1)
3.12	Stag hunt (modified 2)
3.13	Stag hunt (modified 3)
3.14	Stag hunt (modified 4)
3.15	Coordination game
3.16	Parenting style game
3.17	Labour unions game
4.1	Nash demand game
4.2	Agent's payoffs till period 35 (Nash demand game)
4.3	Agent's payoffs from time period 36 till time period 70 (Nash demand game)
4.4	Agent's payoffs from time period 71 (Nash demand game)
5.1	Migration game. X agents less social compared to Y agents
5.2	Migration game. X agents more social compared to Y agents
5.3	Migration game. Both X and Y agents are more social
5.4	Migration game. Both X and Y agents are anti- social

List of Figures

Figure No	Details
2.1	Replicator dynamics in prisoner's dilemma game (Alexander, 2021)
2.2	The dynamical system with s = 0.2 (Young & Foster, 1991)
2.3	Another possibility of the dynamical system with a marginal background mutation rate (Young & Foster, 1991)
2.4	Probability of population composition of 90% T-players (alternatively, 90% D players) with varying population sizes (Young & Foster, 1991)
2.5	Degree 4 network (Young, 2015)
2.6	Star network (Young, 2015)
2.7	Probability of using family planning. Obisa. (Young, 2015)
2.8	Probability of using family planning. Owich, Kawadhgone and Wakula South. (Young, 2015)
3.1	Prisoner's dilemma convergence with exhaustive best response approach (including randomness) and initial history of {(1,1), (1,1)}
3.2	Battle of sexes (Table 3.6) convergence with exhaustive best response approach and initial history of {(0,0), (1,1)}
3.3	Battle of sexes (Table 3.8) convergence with exhaustive best response approach and initial history of {(1,0), (1,1)}
3.4	Battle of sexes (Table 3.8) convergence with exhaustive best response approach. Results are averaged across different initial 2 period histories.
3.5	Battle of sexes (Table 3.8) convergence with expected payoff approach. Results are averaged across different initial 2 period histories.
3.6	Battle of sexes (Table 3.8) convergence with expected payoff approach and initial history of {(1,1), (1,1)}
3.7	Matching pennies (Table 3.9) convergence with exhaustive best response approach and initial history of {(1,1), (1,1)}
3.8	Stag hunt (Table 3.10) convergence with expected payoff approach and initial history of {(1,1), (0,1)}
3.9	Stag hunt (Table 3.11) convergence with expected payoff approach. Results are averaged across all initial histories.
3.10	Stag hunt (Table 3.13) convergence with exhaustive best response approach. Results are averaged across all initial histories.
3.11	Exhaustive best response approach using memory size of 3 with initial state as ([1, 1], [1, 0], [0, 0])
3.12	Exhaustive best response approach using memory size of 4 with initial state as ([0, 1], [1, 1], [0, 0], [1, 1])
3.13	Exhaustive best response approach using memory size of 5 with initial state as ([0, 1], [0, 1], [0, 1], [1, 0], [0, 1])
3.14	Exhaustive best response approach using memory size of 3 with initial state as ([1, 0], [0, 0], [0, 1]) and randomness probability 5%
3.15	Expected payoff approach using memory size of 3 with initial state as ([1, 1], [1, 1], [1, 0])

3.16	Expected payoff approach using memory size of 4 with initial state as ([1, 0], [1, 0], [0, 0], [1, 1])
3.17	Average maximum norm pair % share at the end of simulation
3.18	Action pairs count over 100 iterations for memory length 1 to 12.
3.19	Average maximum norm pair % share at the end of simulation (with randomness of 5%)
3.20	Action pairs count over 100 iterations for memory length 1 to 12 (with randomness of 5%)
3.21	Row and column agents' 0 strategy % share across 100 iterations
3.22	Row and column agents' 1 strategy % share across 100 iterations
3.23	Row and column agents' 0 strategy % share across 100 iterations (randomness 5%)
3.24	Row and column agents' 1 strategy % share across 100 iterations (randomness 5%)
4.1	Ring network with 20 agents and 2 neighbours
4.2	Norm emergence in Newman-Watts-Strogatz small-world network (SW2)
4.3	Newman-Watts-Strogatz small-world network (SW2) at the end of 50^{th} time period
4.4	2d Grid network where norm emergence is not observed
4.5	2d Grid network at the end of 300 time periods
4.6	Emergence of norms in SW2 network with fixed agents
4.7	SW2 network at 50th time period with fixed agents
4.8	Complete network at the 50^{th} timeperiod with fixed agents
4.9	Convergence in complete network with fixed agents
4.10	Most frequent names proposed by agents in SW1 network with 5% probability of new name
4.11	Convergence in ring network (Nash demand game)
4.12	Ring network at the beginning of Nash demand game
4.13	Ring network at 50th time period (Nash demand game)
4.14	Convergence in complete network (Nash demand game)
4.15	Constant oscillation in small world network (SW1) with fixed agents (Nash demand game)
4.16	Initial state of small world network (SW1) with fixed agents (Nash demand game)
4.17	Small world network (SW1) with fixed agents at 300^{th} time period (Nash demand game)
4.18	Norm displacement in SW3 network (Nash demand game)
4.19	Norm displacement in complete network (Nash demand game)
4.20	ER network initial state with $p = 0.9$
4.21	ER network end state with $p = 0.9$
4.22	ER network norm evolution with $p = 0.9$
4.23	ER network initial state with $p = 0.17$
4.24	ER network end state with $p = 0.17$
4.25	ER network norm evolution with $p = 0.17$
4.26	ER network initial state with $p = 0.1$

4.27	ER network end state with p = 0.1
4.28	ER network norm evolution with p = 0.1
4.29	ER network initial state with p = 0.13 and heavy fat-tailed network
4.30	ER network end state with p = 0.13 and heavy fat-tailed network
4.31	ER network norm evolution with p = 0.13 and heavy fat-tailed network
4.32	BA network initial state with m = 2 and star network as base
4.33	BA network end state with m = 2 and star network as base
4.34	BA network norm evolution with m = 2 and star network as base
4.35	BA network initial state with m = 18 and star network as base
4.36	BA network end state with m = 18 and star network as base
4.37	BA network norm evolution with m = 18 and star network as base
4.38	BA network initial state with m = 1 and ER network (p=0.84) as base
4.39	BA network end state with m = 1 and ER network (p=0.84) as base
4.40	BA network norm evolution with m = 1 and ER network (p=0.84) as base
4.41	BA network initial state with m = 2 and ER network (p=0.64) as base
4.42	BA network end state with m = 2 and ER network (p=0.64) as base
4.43	BA network norm evolution with m = 2 and ER network (p=0.64) as base
4.44	BA network initial state with m1 = 8, m2= 2 and ER network (p=0.31) as base
4.45	BA network end state with m1 = 8, m2= 2 and ER network (p=0.31) as base
4.46	BA network norm evolution with m1 = 8, m2= 2 and ER network (p=0.31) as base
4.47	BA network initial state with m1 = 3, m2= 1 and ER network (p=0.81) as base
4.48	BA network end state with m1 = 3, m2= 1 and ER network (p=0.81) as base
4.49	BA network norm evolution with m1 = 3, m2= 1 and ER network (p=0.81) as base
5.1	Replicator dynamic - Case 1
5.2	BNN dynamic - Case 1
5.3	Smith dynamic - Case 1
5.4	Logit dynamic ($\eta = 0.01$) - Case 1
5.5	Logit dynamic ($\eta = 0.3$) - Case 1
5.6	Logit dynamic ($\eta = 0.7$) - Case 1
5.7	Logit dynamic ($\eta = 0.99$) - Case 1
5.8	Replicator dynamic - Case 2
5.9	BNN dynamic - Case 2
5.10	Smith dynamic – Case 2
5.11	Logit dynamic ($\eta = 0.01$) – Case 2
5.12	Logit dynamic ($\eta = 0.3$) – Case 2
5.13	Logit dynamic ($\eta = 0.7$) - Case 2
5.14	Logit dynamic ($\eta = 0.99$) - Case 2
5.15	Replicator dynamic - Case 3
5.16	BNN dynamic - Case 3
5.17	Smith dynamic – Case 3
5.18	Logit dynamic ($\eta = 0.01$) - Case 3
5.19	Logit dynamic ($\eta = 0.3$) - Case 3
5.20	Logit dynamic ($\eta = 0.7$) – Case 3

5.21	Logit dynamic ($\eta = 0.99$) - Case 3
5.22	Replicator dynamic - Case 4
5.23	BNN dynamic - Case 4
5.24	Smith dynamic – Case 4
5.25	Logit dynamic ($\eta = 0.01$) - Case 4
5.26	Logit dynamic ($\eta = 0.3$) - Case 4
5.27	Logit dynamic ($\eta = 0.7$) – Case 4
5.28	Logit dynamic ($\eta = 0.99$) - Case 4
5.29	SW3, $p_{edge} = 0.99$, Case 1
5.30	SW3, $p_{edge} = 0.99$, Network initial state, Case 1
5.31	SW3, $p_{edge} = 0.99$. Network state by the end of 50 periods, Case 1
5.32	SW1, $p_{edge} = 0.99$, Case 2
5.33	SW1, $p_{edge} = 0.99$, Network initial state, Case 2
5.34	SW1, $p_{edge} = 0.99$, Network state by the end of 50 periods, Case 2
5.35	SW2, $p_{edge} = 0.5$. Case 3
5.36	SW2, $p_{edge} = 0.5$, Network initial state, Case 3
5.37	SW2, $p_{edge} = 0.5$, Network state by the end of 50 periods, Case 3
5.38	2d-grid network, Case 3
5.39	2d-grid network. Initial state, Case 3
5.40	2d-grid network. Network state by the end of 50 periods, Case 3
5.41	SW1, $p_{edge} = 0.99$, Case 4
5.42	SW1, $p_{edge} = 0.99$, Initial state, Case 4
5.43	SW1, $p_{edge} = 0.99$, Network state by the end of 50 periods, Case 4
5.44	Ring network, Neighbourhood size = 4, Case 3
5.45	Ring network, Neighbourhood size = 4, Initial state, Case 3
5.46	Ring network, Neighbourhood size = 4, Network state by the end of 50 periods, Case 3
5.47	SW3, $p_{edge} = 0.99$, Neighbourhood size = 4, Case 4.
5.48	SW3, $p_{edge} = 0.99$, Neighbourhood size = 4, Initial state, Case 4
5.49	SW3, $p_{edge} = 0.99$, Neighbourhood size = 4, Network state by the end of 50 periods, Case 4
5.50	ER network initial state with p=0.08. Case 1
5.51	ER network end state with p=0.08. Case 1
5.52	ER network norm evolution with p=0.08. Case 1
5.53	ER network initial state with p=0.96. Case 1
5.54	ER network end state with p=0.96. Case 1
5.55	ER network norm evolution with p=0.96. Case 1
5.56	ER network initial state with p=0.14. Case 1
5.57	ER network end state with p=0.14. Case 1
5.58	ER network norm evolution with p=0.14. Case 1
5.59	ER network initial state with p=0.11. Disconnected network. Case 2
5.60	ER network end state with p=0.11. Disconnected network. Case 2
5.61	ER network norm evolution with p=0.11. Disconnected network. Case 2
5.62	ER network initial state with p=0.11. Connected network. Case 2
5.63	ER network end state with p=0.11. Connected network. Case 2

5.64	ER network norm evolution with p=0.11. Connected network. Case 2
5.65	ER network initial state with p=0.11. Case 3
5.66	ER network end state with p=0.11. Case 3
5.67	ER network norm evolution with p=0.11. Case 3
5.68	ER network initial state with p=0.44. Case 3
5.69	ER network initial state with p=0.44. Case 3
5.70	ER network norm evolution with p=0.44. Case 3
5.71	BA network initial state with p=0.31, m1=3, m2= 1. Case 1
5.72	BA network end state with p=0.31, m1=3, m2= 1. Case 1
5.73	BA network norm evolution with p=0.31, m1=3, m2= 1. Case 1
5.74	BA network initial state with p=0.22, m1=19, m2= 19. Case 1
5.75	BA network end state with p=0.22, m1=19, m2= 19. Case 1
5.76	BA network norm evolution with p=0.22, m1=19, m2= 19. Case 1
5.77	BA network initial state with p=0.36, m1=4, m2= 1. Case 2
5.78	BA network end state with p=0.36, m1=4, m2= 1. Case 2
5.79	BA network norm evolution with p=0.36, m1=4, m2= 1. Case 2
5.80	BA network initial state with p=0.36, m1=4, m2= 1. Case 3
5.81	BA network end state with p=0.36, m1=4, m2= 1. Case 3
5.82	BA network norm evolution with p=0.36, m1=4, m2= 1. Case 3
5.83	BA network initial state with p=0.16, m1=19, m2= 4. Case 4
5.84	BA network end state with p=0.16, m1=19, m2= 4. Case 4
5.85	BA network norm evolution with p=0.16, m1=19, m2= 4. Case 4

List of Abbreviations/Symbols

Abbreviation/Symbol	Details
ESS	Evolutionary stable strategy
$\pi(p1\|p2)$	Payoff which agent receives when playing strategy p1 against opponent agent playing p2
W_C	Expected fitness of cooperate strategy agents
W_D	expected fitness of defect strategy agents
p_C	Percentage share of cooperators
p_D	Percentage share of defectors
$\overline{W}$	Average fitness of entire population
$\frac{dpc}{dt}$	Change in p_C respect to time t
$\frac{dpD}{dt}$	Change in p_D with respect to time t
η	Logit dynamic parameter to control for randomness in agents' decision making
e	Probability of agents proposing names randomly
ACE	Agent-based computational economics
μ	Invading (or new) strategy
σ	Current strategy which population usually follows
BNN	Brown-Nash-von Neumann
Δ_1	Rate at which agents start using strategy p_i
Δ_2	Rate at which agents stop using strategy p_i
m	Memory length
N	Population size
NRF	naming response function
pN	Probability of suggesting new name.
SW1	Small world network 1 (Watts-Strogatz small world network)
SW2	Small world network 2 (Newman-Watts-Strogatz small-world network)
SW3	Small world network 3. Same as SW1 except with the added property of network to be connected
p_{edge}	Probability of rewiring existing edges or adding new edges
pR	Perturbation probability (Probability of agents taking decisions randomly)
RF	Response function
$\frac{dX}{dt}$	Rate of change of X with respect to time t
DyPy	A python library for simulating matrix-form games
network_simulations	Function name in multi-agent-coordination python library

simulation_function_neighbors	Function name in multi-agent-decision python library
EvolutionaryGames	An R library to illustrate the core concepts of evolutionary game theory.
simulation_function	Function name in game-simulator python library leveraging exhaustive best response approach.
simulation_function_payoff	Function name in game-simulator python library leveraging expected payoff approach.
simulation_function_random	Same as simulation_function with the additional parameters on randomness
simulation_function_payoff_random	Same as simulation_function_payoff with the additional parameters on randomness
random_multiplier	Parameter to specify randomness in functions simulation_function_random and simulation_function_random
IPD	Iterated prisoners' dilemma
ER	Erdos-Renyi random network
BA	Barabasi-Albert random network

Chapter 1: Introduction

Social norms are the societal rules that govern how individuals should behave in a society. Norms improve individual cooperation and coordination (Savarimuthu & Cranefield, 2011). Norms are patterns of behaviour which are self-enforcing in nature (Young, 1993). Everybody prefers to conform, if others will conform (Young, 1993). Some examples of norms include:

- Following traffic manners/etiquettes, e.g., driving on the left/right side of the road.
- Exchange of greetings/gifts on festivals or specific occasions.
- Not littering in a public place.
- Dinner table etiquette, etc.

The question arises why norms are followed. Young (2015) talks about the following mechanisms which support normative behavior.

- Coordination. There is an inherent threat of not complying with cooperative behaviour. That threat could imply exclusion from society or sanctions imposed by other agents. Hence, the agents' best interest is to comply with cooperative behaviour.
- Social pressure. Norms are complied due to the threat of social disapproval, or a possible abandonment from the society.
- Providing signals. Following the norms can indicate giving signals to indicate belonging to a certain group.
- Benchmarks and reference points. Norms serve the purpose of being reference points or targets. For example, a sharecropping agreement wherein 50% share is shared with the landlord and tenant.

Savarimuthu and Cranefield (2011) identified five expanded stages of the norm life cycle: norm creation, identification, spreading, enforcement, and emergence. The focus of our research is on norm evolution or emergence. Norm emergence occurs when some significant x% (minimum) agents start following the norm. The value of x varies depending on the context and is seen between 35 to 100 across different simulation studies of norms.

The existing literature on norms evolution can be divided into multiple branches depending upon the approaches followed. There is literature that solved the norms evolution problem using the deterministic /analytical approach and some literature that has used the computational/simulation approach or a combination of these two. The existing literature has focused on how norms evolve and depend upon various parameters. Some of these parameters include the payoff structure of the game, population size, memory length, strength of relations with other agents, time taken to reach a norm, methods which agents follow to update their actions during each period of the game, and randomness in agents' actions (Young & Foster, 1991; Kandori et al., 1993; Young, 1993; Young, 2015; Alexander, 2007).

The focus of our research is on norms evolution using computational methods. Norms evolution from a computational standpoint has been described in the following three different ways:

- First, some finite normal-form games are defined along with their payoff matrix. Agents play the game repeatedly for a certain period. Agents' decision on what strategy to choose at any given time depends upon past strategies played. The strategy being played most frequently is considered a norm strategy (Axelrod, 1986; Shoham & Tennenholtz, 1992;

Young, 1993). In this case, the norm strategy is not necessarily a single strategy but an action pair from the payoff matrix that captures row and column player strategies.

- Secondly, there is a finite game with specific strategies and the corresponding payoff values. In this approach, we also define population size and the agents' distribution following these strategies before starting the game. Agents interact with other agents over a period, calculate payoffs and possibly adapt their strategies. Agents can use different methods to decide what strategy to choose. It could be a simple imitation of other agents' strategies if following that strategy results in a higher payoff, or it could be more complex, like using neural networks or genetic algorithms (Nishizaki et al., 2009). At the end of the simulation period, we get the information on the agents' revised strategy distribution. The strategy being played most frequently is considered a norm (Young & Foster, 1991; Axtell et al., 1999; Tesfatsion, 2003; Nande et al., 2020).
- Another school of thought belongs to incorporating agents' social networks into consideration. This approach defines agents' social networks and evaluates their impact on agents' decisions (Alexander, 2007; Young, 2015).

Axelrod (1986) was one of the first papers which used the evolutionary approach to norms evolution. Young (1993) talks about any specific equilibrium can be understood as a norm and it need not be necessarily due to that being inherently prominent. It can be explained by the dynamics of the process which selected the particular outcome. Therefore, there is a path dependency. Similarly, Axtell et al. (1999) demonstrate that emerging social norms may not necessarily have convincing apriori justification and could have arisen due to random events.

Norm emergence also depends upon the parameters specified, and the process followed for evolution. The dynamics are defined by the model specified (agent-based discrete or aggregative), payoff values, population size, and the level of noise present in agents' decision-making (Young & Foster, 1991; Kandori et al., 1993). Young and Foster (1991) showed that stochastic and deterministic variants of the evolutionary model give significantly different results. These results are supported by Kandori et al. (1993), which showed that long-run equilibrium results depend upon the payoff structure of the game. It also raises the importance of noise present in agents' decision-making. Randomness helps to reach coordination on a particular equilibrium which do need require to be Pareto dominant. Nishizaki et al. (2009) demonstrates the possibility of cooperative outcomes in prisoners' dilemma games when agents make decisions based on learning mechanisms using neural networks and genetic algorithms. The input to the neural network-based model includes parameters on agents' choices in the prior period, population obedience rate in the previous period, personal taste and preferences of agents, individual agents' utility in prior periods, the aggregate utility of all agents in the prior period, degree of belief of individual agent for the social norm, among others. The discrete model investigates the agent-level information and assesses how it impacts agents' decision-making (Alexander, 2007). Axtell et al. (1999), Sen and Airiau (2007), and Martinez et al. (2021) are some of the papers which used a discrete model approach to show norm evolution computationally. It shows results being dependent upon initial states, agents' memory length, population size, and the learning framework agents use to make decisions during each simulation state. Our research focuses on norm evolution using a discrete model approach.

In the traditional game theory, expected utility involves making assumptions like continuity, substitutability, transitivity, and monotonicity. These conditions may not hold in all situations. In reality, heuristics incorporate a common body of beliefs acquired through participation in a common culture. Humans have bounded rationality and do not generally rely on complex calculations. These calculations are generally derived from heuristics and rules of thumb (Alexander, 2007). In agent-based computational economics (ACE), agents are assumed to be bounded rational (Tesfatsion, 2003). Evolutionary models minimally require two things (Alexander, 2007). First, representation of the current state of population and second, the dynamical laws that explain how the state of the population changes over time. There are two models, the continuous model and the discrete model, which are used to represent the population. The continuous model uses global statistics to represent the population, implying that agents' choices and frequencies are represented at an aggregate level. The discrete model investigates the agent-level information and assesses how it impacts agents' decision-making.

The literature has discussed two approaches to the evolutionary game theory (Alexander, 2021). The first approach revolves around the evolutionary stable strategy (ESS) as the principal analysis tool. This approach can be considered static because it does not explicitly model the underlying process by which strategy changes. For example, assume that σ is the strategy the population usually follows and μ is the invading (or new) strategy. For σ to maintain its stability, it should do well against μ. If other agents follow σ, and if a new agent is evaluating to follow σ or μ, the new agent will follow σ if the following holds.

$\pi(\sigma \mid \sigma) >= \pi(\mu \mid \sigma)$

where $\pi(p1|p2)$ is payoff which agent receives when playing strategy p1 against opponent agent playing p2.

Suppose if $\pi(\sigma \mid \sigma) = \pi(\mu \mid \sigma)$, which means the new agent is indifferent between σ and μ, then the following condition should hold for a new agent to have an incentive to choose σ

$$\pi(\sigma \mid \mu) > \pi(\mu \mid \mu)$$

The above condition implies that the payoff from following σ given others follow μ should be strictly higher than the payoff from following μ given others follow μ.

A strategy σ is called an evolutionary stable strategy (ESS) if it satisfies both these conditions. Over time, multiple variants of evolutionary tools developed like uniform invasion barrier, locally superior, evolutionary stable set, equilibrium evolutionary stable set, etc. This led to competing definitions of stability where in some cases a strategy can satisfy the stability criteria laid out by *locally superior* and *equilibrium evolutionary stable set* but do not satisfy the criteria for ESS. Therefore, evolutionary game theory encounters similar selection problems that traditional game theory encountered over the years. Due to this, the focus of research shifted towards the second approach, a dynamic approach to evolution.

In the dynamic approach, an explicit model showing the process around dynamics is constructed, which shows how the frequency of strategies changes. Moreover, it further studies the properties of dynamics. Some interesting results emerged from literature focusing on dynamic approaches to evolution. For example, Harms and Skyrms (2008) show the possibility of a ‘cooperate’ outcome

in a prisoner's dilemma game, an equal share in a symmetric bargaining game, cooperate outcome ('stag') in a stag hunt game rather than risk-dominant outcome ('hunt') etc. Hofbauer and Sandholm (2011) opine on the possibility of agents playing strictly dominated strategies in the population.

The first tool used predominantly in the context of dynamic evolution is replicator dynamics. Suppose a prisoner's dilemma game with two strategies: cooperate (C) and defect (D). W_C denotes the expected fitness of cooperate strategy agents, while W_D represents the same for defect strategy agents. p_C and p_D denote the percentage share of cooperators and defectors in the population in the current generation. $\overline{W}$ shows the average fitness of the entire population. We can write the expected fitness equation as follows:

$W_C = p_C * \pi(C|C) + p_D * \pi(C|D)$

$W_D = p_C * \pi(D|C) + p_D * \pi(D|D)$

$\overline{W} = p_C * W_C + p_D * W_D$

Using the above formulation, we can construct the two equations below: replicator dynamics.

$$\frac{dpc}{dt} = \frac{pc * (Wc - \overline{W})}{\overline{W}}$$

$$\frac{dpD}{dt} = \frac{pD * (WD - \overline{W})}{\overline{W}}$$

where $\frac{dpc}{dt}$ and $\frac{dpD}{dt}$ represents the change in p_C and p_D with respect to time t, respectively. With these equations, we can compute the rate of change of increase /decrease in co-operators/defectors across various time points, which also depends upon the divergence between individual strategy fitness value and the average fitness value of the entire population.

Researchers propose other dynamics over time. Other dynamics, like the BNN dynamic, require agents to consider all alternatives available along with their payoff matrices so that new strategies can also enter the population if it is not already entered. A further refinement of the BNN dynamic is the Smith dynamic. BNN dynamic compares the expected payoff of alternative strategies with the average population payoff. However, the Smith dynamic compares the expected payoff of current individual strategy with the expected payoff from following other strategies. Hence, the individual payoff of strategies is compared with each other and not with the population average. The logit dynamic involves an additional parameter (η). As the name suggests, the logit dynamic involves taking the exponential of fitness value associated with individual strategy multiplied by η^{-1}. As η approaches 0, the probability of playing any strategy that is not the best response goes to zero.

1.1 Research Gaps

The literature on simulations-based evolution has been developed considering the specific problem. Most of the existing literature performed simulations using a hypothetical game, and it is difficult for the reader to replicate the same results or to check the impact of tweaking parameters. There has been less visibility around how the results have been derived and what tool

or programming language has been used to perform simulations. It is not easy to expand the results presented in the literature with different parameters or different payoffs of the game. There is less clarity on the comparative statics of simulation models presented in the literature with the possibility of assessing the impact of changing different values. We have not come across any extensive reusable framework to address the norm evolution problem using agent-based decisions. We believe that the computational approach followed, and its visibility is required particularly for the research which has shown some interesting not-so-obvious results from the classical game theoretical constructs standpoint. Some of these results include possibility of 'cooperate' outcome in prisoner's dilemma game, equal share in symmetric bargaining game, cooperate outcome ('stag') in stag hunt game rather than risk-dominant outcome ('hunt') or possibility of agents playing strictly dominated strategies in the population (Harms & Skyrms, 2008; Hofbauer & Sandholm, 2011; Alexander, 2007; Alexander, 2021).

We have also seen limited research on the agent-based norms evolution considering impact of agents' social networks except the more recent ones done by Alexander (2007), Young (2015) or Chatterjee et al. (2023). Hence, we believe that there is some more scope to explore in this area. A lot of research on norms using simulations has considered the evolutionary dynamics approach towards evolution. The research has used replicator dynamics and related approaches towards evolution which we believe is more top down or aggregative in nature. This leads us to approach this problem from an agent-based perspective to try and test how the individual agent-based approach differs from the existing dynamics-based approaches.

1.2 Objectives

Against this background, we have tried to bring transparency into the process of simulations and how this can help answer the question of norms evolution. In response to the current literature divided into three sections listed at the beginning of this chapter, we have created a simulation framework addressing all three approaches used in these papers. These reusable frameworks can be used to test different possibilities concerning norms evolution computationally. These frameworks help to perform comparative statics with respect to multiple parameters like initial state, memory length, population size, response functions, agents' neighbourhood size, number of trials, randomness in agents' decisions, agent's types, among others. We have considered different social networks and their impact on agents' decisions. These computational frameworks help explain the results which are in conformity with respect to classical game theory constructs, but it also explores the possibilities of explaining conflicting outcomes.

We also attempt to compare results from evolutionary dynamics with agent-based modeling to check the differences in outcomes. This will help answer what insights can each model provide about individual agent behavior. Policymakers can use this knowledge to design interventions that promote desirable norms.

We have executed the reusability objective by creating three open-source Python libraries that anyone can use without building and writing the code from scratch. This would also help expand the libraries to make them broader and more accessible by incorporating suggestions from different researchers.

We have presented insights leveraging these norm evolution frameworks built with the help of a few important games like prisoner's dilemma, stag hunt, battle of sexes, coordination, matching pennies, naming game (Young, 2015), Nash demand game (Axtell et al., 1999) and migration game.

1.3 Norms Evolution Frameworks

Firstly, we created a norms evolution framework using two different response functions or approaches agents can use to decide what action to take. In the first approach, agents choose the strategy having the maximum payoff in response to all possible opponents' strategies from historical interactions. The second approach considers the relative frequencies of different strategies played in the past and uses these as weights to calculate an expected payoff. In this scenario, a finite normal-form game is defined along with its payoff matrix. The action or strategy pair from the payoff matrix being played or proposed most frequently is considered a norm strategy pair.

This type of evolution framework is more applicable in scenarios where we assume there are two broad categories of agents in the form of row players and column players. Agents are uniform, and they are differentiated only with respect to the choices they make. The agent's social network is assumed to be complete, as every agent knows the prior history of interactions. Agents are concerned only about the choices made in previous periods, irrespective of who has made those choices. This scenario applies in close-knit communities with a higher degree of social connection.

We can apply this framework where we have a row player, a column player, a defined payoff matrix, and their strategies or choices. We want to know which choices would be more frequently proposed by row and column players. We are looking for a combination of actions that are being proposed more frequently. It can be used to answer questions like which of the Nash equilibrium can emerge as the norm under different parameter restrictions when multiple pure strategy Nash equilibria exist. It can also decide parameter restrictions for Pareto efficient and inefficient outcomes. For example, in a coordination game below in Table 1.1, there are two pure strategy Nash equilibria, (2,2) and (1,1)

Table 1.1 Coordination game

	Left	Right
Left	2,2	0,0
Right	0,0	1,1

We can think of (2,2) as both agents driving on the left side of the road and (1,1) as both agents driving on the right side. We want to assess if we start the game with initial states as {(2,2), (0,0)}, what outcome we can expect to achieve assuming agents have limited memory and consider the past two periods of history. There could be a situation where some agents in certain regions drive on the left side of the road and some on the right side due to poor implementation of the laws or agents not following morally appropriate behavior. The results would vary depending on the initial state with which the game is started and the agents' memory length. We want to answer which pair from the above payoff matrix would be observed more frequently than others. The framework developed would also answer questions on if the initial state involves one Nash equilibrium, is

possible to achieve some other Nash equilibrium during the game, e.g., the game started with (2,2) as one of the initial states, but eventually (1,1) is being observed more frequently after the simulation as compared to (2,2). This can help formulate relevant incentive structures or sanctions which can change the direction of outcomes. We have tested this framework on a few popular 2*2 normal form games like Prisoner's dilemma, Battle of Sexes, Stag Hunt, etc. We have also assessed the impact of increased memory length on the outcomes. Results show that a higher memory length leads to convergence towards actions pairs which are more in alignment with pure and mixed strategies Nash equilibria. We have created *game-simulator* library, which can be used to assess evolution for any combinations of m*n payoff matrix, memory length, time period, initial states etc. This is the first library or tool we are aware of which can be used for this analysis by the reader without writing the code or program from scratch. This produces output in the frequency distribution of pairs of the payoff matrix.

In the second method to approach norm evolution, we have considered the scenario where strategies are defined along with their payoff values. We also have the population with a defined percentage share following those strategies. Agents are connected via a specific social network. Agents update their actions or strategies according to the strategies followed by their neighbours. We explain this with the help of the Nash demand game. Agents play the Nash demand game of Axtell et al. (1999), where there is a fixed pie (say 100 dollars), and agents are expected to distribute that among themselves. Agents can make three demands, namely High (H) with a payoff of 70, Medium(M) with a payoff of 50, and Low (L) with a payoff of 30. These three choices are agents' options or strategies which they can choose during the game. At any given point, two agents are selected, and they make their demands. The rule of the game is that agents get these

payoffs only if the demands made by both the agents playing the game are lower than or equal to 100. If the total demanded payoffs exceed 100, none of the agents receives anything. Therefore, if both the agents demand H or one of them demands H and another M, both would receive a zero payoff. Agents' initial distribution of strategies is defined before starting the game, implying how many % of agents follow H, M, and L. We are interested in knowing how the initial percentage share of agents following these strategies changes over a period of time. The strategy played by more agents and for a higher duration is a likely candidate for the norm.

There are multiple scenarios where this can be applicable. Examples include when agents try to decide which education course their kids should pursue. A degree of social connection is attached to the education agents' kids pursue. The payoffs are different if agents in small and close-knit communities pursue the same education course followed in their neighbourhood versus something else. Those payoffs could also be non-monetary in the form of respect, embarrassment, or a feeling of belonging or exclusion. We can think of a game with two strategies, pursuing the same education course generally followed in the community versus something else along with its payoffs. Some agents in the community pursue the norm, while others choose not to do so. There is an initial state with, say, 50% of agents following the norm while 50% not following the norm. We are interested in knowing when these agents are connected and play the game over time, which of the norms outweighs the other. We have created a second library named *multi-agent-decision* which can be used to get the choice distribution in the population. For example, in the Nash demand game with three actions, High, Medium, and Low, if we start the game with some initial percentage distribution of agents following these strategies (say 33% each), we can get the answer on how this distribution would change as agents interact with their neighbours and possibly change their

choices. Agents' decision depends upon the number of strategies, initial state, population size, agents' neighbourhood size etc. The output generated from the library includes information on strategy frequency distribution during each period of the game.

The third possibility where norm evolution is applicable includes when no fixed actions or strategies are defined, and agents need to formulate these choices while interacting with other agents in their neighbourhoods. We explain this scenario with the help of the Naming game of Young (2015) in chapter 4. In the Naming game, two agents are selected randomly and shown a picture of a face. Agents do not know the identity of other agents, and they independently suggest a name for the face. If agents suggest the same name, they get a positive reward; otherwise, they get a negative reward. At the end of each iteration, agents get to know the names proposed by opponent agents, and this keeps them updated with the names currently popular at any given time. There is no constraint on the names agents can propose, and it is left to their imagination. Agents use a perturbed response function to decide what names to propose, implying agents propose names randomly with certain probability e and propose the most frequent names with probability 1-e. 'e' here can be interpreted as the probability of committing an error by the agents. A fixed population size is defined along with the agents' neighbourhood size. No constraint exists on the number of unique names that can evolve after the simulation. We have shown that this framework results in convergence towards 2 or 3 names when agents play the game repeatedly when agents are connected in the ring network. These selected few names which emerged from the plausible multiple names with agents' interactions are called norms. We have seen higher chances of norms emergence in case of denser networks. However, if the network is heavy fat-tailed, it can

compensate for lower network density. A lower network diameter and higher clustering coefficient correlates positively with norm emergence.

There are different situations where this kind of norm evolution behavior emerges. For example, parents can decide on any reasonable names for their kids. However, before choosing any name, they generally consider what names have been used by other parents for their kids. This way, they can reduce their potential name choices to a manageable number. They can either choose the name other parents have used or they can still propose a new name. And, this cycle continues where no fixed set of names is defined, but still parents narrow their choices to a few. Another example could be designing a new compensation strategy in an organization. A private organization can design any compensation composition for their employees regarding the split between base pay and variable pay, assuming no regulatory restrictions exist. However, before deciding the optimal split, they consider the compensation strategies of other organizations to remain employee-friendly employer in the market. When a new organization enters the market, it can either follow the composition other companies are using or tweak it marginally to remain competitive. This leads to a fewer number of potential splits, which can happen between base and variable pay when, to begin with, there could be multiple possibilities.

To generalize the results from this approach to different possibilities of norm evolution where cooperation is rewarded, we created a third Python library called *multi-agent-coordination*, when no fixed choices are defined at the beginning of the game. For example, there are no fixed defined names in the Naming game, but these emerged endogenously as agents interact with each other. The library can get results with custom user-defined parameters on agents' social networks and

associated parameters. The output generated from the library include information on strategies proposed by agents at different points in time, fixed agent distribution, how quickly agents reach the norm, and how agents' strategy choices influenced other agents' choices connected via the network.

Once a specific norm is established, it can be displaced by other norms over time. We have shown this with the help of the Nash demand game in chapter 4, where there is an incentive provided to agents to follow the 'L' or 'H' strategy to displace the 'M' norm. This norm evolution behavior is relevant in multiple scenarios. For example, we can assume two major telecom providers are in the country, and a new player is trying to enter the market. Most of the population sticks to the two dominant providers at any time. The new entrant incentivizes users to switch to their services to gain market share. Moreover, those incentives could be in the form of additional broadband services at no extra cost. There are also incentives involved if an agent forwards this information in their network and brings their network to use the new entrant services. Hence, there is some boost in payoff for the agents, which could be a notional gain also for those trying to switch to the new player. Another example could be when the Government is trying to push any agenda and wants the general public to follow that. For example, to promote women's empowerment in the country, Indian Government provides certain savings and investment avenues for women providing higher risk-free returns. This translates to a shift in savings and investment options for the women population from existing avenues like bank account savings to more lucrative Government-backed bonds and securities. Hence, this can be interpreted as changes in payoff structure for the women population evaluating different savings and investment avenues with

higher payoffs for Government-backed bonds and securities. We captured the norm displacement possibility in the *multi-agent-decision* library created as one of the additional parameters.

So far, all three possibilities and their corresponding Python libraries explained above approach norm evolution using individual agent-based approaches. This implies that the norms that emerge are derived from agents' interactions with their neighbours, which translates to norms at the macro/population level. In Chapter 5, we explored evolutionary dynamics, like replicator dynamics which consider a macro or aggregative approach towards evolution. We compare evolution results from individual agent-based approaches versus replicator dynamics and other dynamics. This is explained with the help of the migration game. We formulated a migration game that entails some agents migrating from a domestic country to a foreign country where most agents follow certain norms X and Y, respectively. Agents' payoffs vary when domestic country agents interact with foreign country agents. We are interested in knowing which norm out of X and Y survive when domestic and foreign country agents coexist in the population in a foreign country. Results show that the individual agent-based evolution methods show insights more in alignment with real-life situations than results from replicator dynamics. We also assessed the impact of network structures on the outcomes. A higher dense network usually leads to convergence towards one outcome. Results are sensitive to the fat-tailedness and clustering of the network in case of lower dense networks. Medium to heavy fat-tailedness can lead to convergence towards one outcome in the absence of high densities. When agents are connected in Barabasi-Albert network structures, results primarily dependent on the strategies followed by "hub" agents.

With these different norm evolution and displacement methods and techniques, we have tried to capture multiple scenarios where agents must interact with other agents and define the norm. That norm could be interpreted as the action which is followed most frequently. This has wide applications in multiple areas of economics and beyond, like sociology, computer science etc. We have attempted to share this knowledge broadly with a larger audience by creating three open-source Python libraries. These libraries support customization with respect to different games, payoff matrices, response functions, memory length, network types, agent types, simulation time periods etc. In addition, the output of these libraries is in the form of graphs and Excel files which provide detailed and granular information on agent's choices at every step of the simulation exercise. We believe that every user can make use of Excel files and hence is simpler to use and analyze. This also increases the transparency of the simulations performed, which we have found somewhat lacking in the current literature. We believe that these libraries are our humble contribution to the advancement of more computational and agent-based modeling in economics where lack of suitable software tools has been pointed out as one of the challenges lately (Axtell & Farmer, 2022).

The rest of the thesis is divided into four chapters. The next chapter talks about the literature review in detail. We also provide a brief literature review in respective chapters relevant to the topic or approach discussed in that chapter to maintain continuity. Chapter 3 details the two different response functions and norm pair evolution, describing which pair from payoff matrix has the potential to be observed more frequently. Chapter 4 details social networks and norm evolution with the help of Nash demand and Naming games. Chapter 5 compares the results from individual agent-based approaches established in Chapter 4 with the replicator dynamics

and other different dynamics variants with the help of the migration game. The thesis concludes by providing conclusive remarks along with limitations and future research areas.

Chapter 2: Literature Review

In this chapter, we first start with the definition of norms. We explain some of the characteristics or properties of norms. Then we will explain different approaches followed to explain the evolution of norms. This includes both static approaches to evolution and dynamic approaches towards evolution. We then differentiate the dynamic approach towards evolution into the continuous/aggregative model approach and the discrete agent-based modeling approach. We discuss the results of norms evolution leveraging these two different approaches. We next talk about the transition from static to dynamic approaches and the reasons behind it. The comparative statics results showing how different parameters impact the evolution process is also being discussed. These parameters include different methods or response functions that agents follow and how they impact the evolution process. The agents' social networks and their impact on the simulation results are also being discussed. The chapter concludes by providing an overview of the agent-based modeling approach toward evolution and its characteristics.

Norms are an active area of research in sociology, economics, biology, philosophy, law, and computer science. In economics, agent-based computational economics (ACE) studies how economies evolve with the decentralized individual interactions of autonomous agents (Tesfatsion, 2003). Software agents are modeled borrowing behaviour from human societies in multi-agent systems within computer science literature. Researchers in multi-agent systems have used the social construct of norms to study how cooperation and coordination can be achieved.

2.1 Norm definition

Norms are the societal rules that govern how individuals should behave in a society. Norms improve individual cooperation and coordination (Savarimuthu & Cranefield, 2011). Norms are self-enforcing patterns of behaviour (Young, 1993). It is in everybody's interest to conform, given the expectation that others will conform (Young, 1993). The question arises why norms are followed. The reasons include fear of authority or power, rational appeal to norms, emotions, e.g., shame, guilt, embarrassment, and willingness to follow the crowd (Savarimuthu and Cranefield, 2011). Examples of norms include the exchange of gifts on Christmas, style of dress, dinner table etiquette, driving on the left/right side of the road, following traffic rules, not littering in a public place etc. Young (2015) talks about the following mechanisms which support normative behavior.

- Coordination. There is an inherent threat of not complying with cooperative behaviour. Hence the best thing for agents is to comply with cooperative behaviour.
- Social pressure. Norms are complied due to the threat of social disapproval, or a possible abandonment from the society.
- Providing signals. Following the norms can indicate giving signals to indicate belonging to a certain group.
- Benchmarks and reference points. Norms serve the purpose of being reference points or targets. For example, a sharecropping agreement wherein 50% share is shared with the landlord and tenant.

Savarimuthu and Cranefield (2011) identified five expanded stages of the norm life cycle: norm creation, identification, spreading, enforcement, and emergence. We explain this briefly below.

Norm Creation: One of the three approaches creates Norms in multi-agent systems. The first approach is offline design, wherein a designer specifies norms. In this approach, norms are hard-

wired into agents and are being used in a context wherein norms are beneficial for the entire society. Another approach is where a norm leader specifies norms. This leadership approach can be authoritarian or based upon democratic relations, and the leader can dictate norms to follower agents. The third approach is entrepreneur driven, wherein norm creators are not necessarily leaders. However, they create a proposed norm and have the potential to influence other agents to follow the norm.

Norm Identification: If norms are created by one of the abovementioned approaches, then norms may spread in society. However, if norms have not been explicitly created, it implies norms would require to be created or derived based upon agents' interactions with each other. In this case, agents would need some mechanism to identify norms from their environment. One such mechanism is game theory wherein agents have a limited set of actions available, and they choose the action that maximizes their utility as the norm. This mechanism is then supplemented with various computer simulations to derive conclusive insights. The second approach considers the cognitive abilities of an agent to infer norms. Here norms are based on the observations of interactions among agents, and agents have normative expectations, beliefs, and goals. There is a higher possibility of norms divergence as these are based upon the observations and hence an agent can create its own notion of the norms.

Norm spreading relates to distributing norms to larger audiences using different mechanisms and methods like leadership, entrepreneurship, and cultural and evolutionary methods.

Norm enforcement: It involves punishing agents who violate norms and rewarding those who follow them. Punishment and rewards could be monetary or non-monetary.

Norm emergence: Norm emergence is when some significant x% (minimum) agents start following the norm. The value of x varies depending on the context and is seen between 35 to 100 across different simulation studies of norms. Also, emergence can be viewed from a global or a local viewpoint, e.g., an agent might only look for agents one block away from him/her in all directions in a grid environment. In addition, it is also possible that spreading can directly lead to emergence without much enforcement. Once a norm emerges, it is superseded by another new norm created either using an offline design approach or through agents' interactions among themselves. Moreover, this cycle continues.

A few advantages and disadvantages are associated with different mechanisms being followed during the norm lifecycle. For example, offline design models are simpler to implement, and the designer has greater control over system functionality. However, if the system is complex, wherein agents' goals keep on changing, it would be costly and inefficient to keep reprogramming agents. Also, the more complex the system becomes, the less likely it will translate to effective social laws. Another drawback of the offline design model is that it assumes all agents will follow the norm, which might be unrealistic, particularly in open societies where competing norms might be present at a given time. The leadership and entrepreneurship approach assumes that a powerful authority is present in society and that all agents acknowledge the power of such agents. Also, it is inherently assumed that the authority present is *all knowledgeable*.

2.2 Norm characteristics

Young (2015) talks about some of the key features of norms, which are explained below:

Persistence. Norms tend to be followed for a longer period and generally respond slowly to changes in extraneous conditions.

Tipping. When norms shift happens, the transition is quick. Once a specific threshold is reached and enough people start following new norms, this results in new ways of doing things, and the transition to new norms completes rapidly.

Punctuated equilibrium. This reflects the phase when there are sufficiently large periods of no changes followed by a sudden shift in activity in which new norms are displacing an old norm. This is called the punctuated equilibrium effect.

Compression. This implies that individual choices have less variation than otherwise expected had there been no norm. For example, if landowners and tenants did not follow the conventional norm of 50-50, there would have been more diversity in contract terms.

Local conformity/ Global diversity. There could have been different norms present in local communities, while at the global level, there could be a single norm. Norms in each community at any given time are usually unpredictable. When the interaction among communities is occasional, they may follow different norms despite being similar in other respects.

2.3 Classical game theory approach

Classical game theory assumes agents as perfectly rational. A good theory of the evolution of norms should explain the following situations.

- Evolution of altruism in Prisoner's dilemma (Table 2.1), where players are encouraged to cooperate instead of defect, is the outcome of rational choice theory.

Table 2.1 Prisoner's Dilemma game

	Cooperate	Defect
Cooperate	3,3	1,4
Defect	4,1	2,2

- Evolution of cooperative outcome (Stag) in stag hunt game, rather than risk-dominant outcome (hunt). According to rational choice theory, if agents do not trust each other, they would play hunt, which is a safer option. However, playing hunt makes their payoffs lower than playing stag (Table 2.2).

Table 2.2 Stag Hunt game

	Stag (Cooperate)	Hunt (Go it alone)
Stag (Cooperate)	4,4	0,3
Hunt (Go it alone)	3,0	3,3

- Equal share in Nash bargaining game. If the shares demanded exceed 100%, then no one gets anything. Otherwise, each would get what each agent demands. In this scenario, there

exists an infinite number of Nash equilibria. This also includes where one agent demands everything, and the other agent gets nothing. The challenge is determining conditions where evolution propels agents to distribute equally.

Classical game theory lacked the dynamics required to explain some of these scenarios. Although the extensive form of the game and repeated game literature can capture some of the dynamics compared to normal form representation, it becomes unmanageable with more complex games. The action the agent will take is decided beforehand at every possible stage of the game. This representation does not consider what agents would learn during the game and how it responds to the newly available information about their opponents at each point of the game. Evolutionary game theory considers learning and dynamics (Alexander, 2021). This leads to the rise of game theory's evolutionary variants, which can explain how a particular outcome evolves over a period (Uyttendaele, 2015). According to Alexander (2021), three main reasons explain the emergence of evolutionary game theory. First is the equilibrium selection problem in games having multiple pure strategy Nash equilibria in classical game theory. Second is the assumption of perfect rationality of agents in classical game theory, which evolutionary game theory does not consider. Third is the lack of dynamics in traditional game theory.

2.4 Static approach towards evolution

Two approaches to evolutionary game theory have been discussed in the literature. In this section, we talk about the static approach towards evolution. The first approach revolves around the evolutionary stable strategy (ESS) as the principal analysis tool. This approach can be considered static because it does not explicitly model the underlying process by which strategy changes. For

example, assume that σ is the population's strategy usually follows, and μ is the invading (or new) strategy. For σ to maintain its stability, it should do well against μ. If other agents follow σ, and if a new agent is evaluating to follow σ or μ, the new agent will follow σ if the following holds.

$$\pi(\sigma \mid \sigma) >= \pi(\mu \mid \sigma)$$

where π(p1|p2) is payoff which agent receives when playing strategy p1 against opponent agent playing p2.

Suppose if $\pi(\sigma \mid \sigma) = \pi(\mu \mid \sigma)$, which means the new agent is indifferent between σ and μ, then the following condition should hold for a new agent to have an incentive to choose σ

$$\pi(\sigma \mid \mu) > \pi(\mu \mid \mu)$$

The above condition implies that the payoff from following σ given others follow μ should be strictly higher than the payoff from following μ given others follow μ.

A strategy σ is called an evolutionary stable strategy (ESS) if it satisfies both these conditions. From the above two conditions, we can say that every ESS is also a Nash equilibrium. Also, every strict Nash equilibrium is evolutionary stable. Hence, we can say that ESS is a stronger version of Nash equilibrium. Any game with finite strategies and finite players has at least one Nash equilibrium assuming presence of mixed strategies. However, every game does not need to have an ESS. E.g., Rock- Scissors-Paper game does not have any ESS.

Over time, there are other related concepts also emerge. A strategy σ satisfies the conditions for a *uniform invasion barrier,* provided there exists an $\overline{\epsilon}>0$ for all $\sigma \neq \mu$ and given $\epsilon \in (0, \overline{\epsilon})$,

$$\pi(\sigma \mid \epsilon\mu + (1- \epsilon)\, \sigma) > \pi(\mu \mid \epsilon\mu + (1- \epsilon)\, \sigma)$$

It says that when all agents in the population follow the same strategy σ, and a small percentage of the population follows only mutant strategy μ, then the incumbent strategy σ has strictly higher expected fitness consisting of the incumbent and invading strategy than the mutant strategy μ. Assuming a sufficiently large population size with a small number of invaders leads to selection against the invading strategy, and σ would become evolutionary stable.

Another related concept is a strategy being *locally superior*. This comes from the understanding that a stable strategy should not have the possibility of drifting. A strategy σ satisfies the conditions of *locally superior* if there is a neighbourhood U around σ such that

$\pi(\sigma \mid \mu) > \pi(\mu \mid \mu)$ for all strategies $\mu \in U$ where $\mu \neq \sigma$.

From above, we can say that these three statements, namely σ is ESS, a uniform invasion barrier and is also locally superior, are all equivalent (Hofbauer et al.1979). Over time there are other evolutionary concepts evolved. One such alternative is the *evolutionary stable set* (Thomas 1984, 1985). Suppose the game is as shown below (Table 2.3):

Table 2.3 A hypothetical game with evolutionary stable set

	S1	S2	S3	S4
S1	(1,1)	(1,1)	(1,1/2)	(1,1/2)
S2	(1,1)	(1,1)	(1,1/2)	(1,1/2)
S3	(1/2,1)	(1/2,1)	(1/2,1/2)	(1/2,1/2)
S4	(1/2,1)	(1/2,1)	(1/2,1/2)	(1/2,1/2)

The above game has no evolutionary stable strategies because the payoffs from playing strategy S1 and S2 are the same when agents play these against each other. However, any population which follows a mix of S1 and S2 is stable. Swinkels (1992) coined the *equilibrium evolutionary stable set*, further refining the idea of an evolutionary stable set. It implies that equilibrium evolutionary stable set is a subset of evolutionary stable set but not vice versa.

Hence, we can see that multiple competing concepts of static evolutionary stability evolved over time. Therefore, evolutionary game theory encounters similar selection problems that traditional game theory encountered over the years. Due to the above reasons, the focus of the research shifted to a second approach which is more dynamic. The dynamic approach can explain how the strategies change over a period of time. The focus of our research is on this second approach.

2.5 Dynamic approach towards evolution

The first tool used predominantly in the context of evolutionary game theory is replicator dynamics. We explain this with the help of the Prisoner's dilemma game. Consider the following prisoner's dilemma game (Table 2.4):

Table 2.4 Prisoner's dilemma game with hypothetical payoffs

	Cooperate	Defect
Cooperate	(R,R)	(S,T)
Defect	(T,S)	(P,P)

We are trying to answer how the agent's composition from playing the above game evolves. The traditional game theory says agents should always play Defect. We try to answer this question using replicator dynamics if that answer stays the same.

We assume that the population size is large and the probability of an agent interacting with the cooperator type or defector type agent equals their percentage share in the population respectively. . There are two strategies, cooperate (C) and defect (D). W_C denotes the expected fitness of cooperating strategy agents, while W_D represents the same for defect strategy agents. p_C and p_D denote the percentage share of cooperators and defectors in the population in the current generation. $\overline{W}$ shows the average fitness of the entire population. We can write the expected fitness equation as follows:

$W_C = p_C * \pi(C|C) + p_D * \pi(C|D)$

$W_D = p_C * \pi(D|C) + p_D * \pi(D|D)$

$\overline{W} = p_C * W_C + p_D * W_D$

π (a|b) is the payoff when playing strategy 'a' against someone using strategy 'b'. There is an assumption that The percentage share of the population following cooperate or defect strategy in the future generation is defined by the following rule:

$$p_C' = \frac{pC * WC}{\overline{W}}$$

$$p_D' = \frac{pD * WD}{\overline{W}}$$

The above two equations imply that if the fitness from cooperating strategy (W_C) is lower than the aggregate average fitness ($\overline{W}$), it is more beneficial to defect than cooperate. Hence the percentage share of agents following cooperate is expected to reduce in the next period.

Therefore, if $W_C/\overline{W} < 1$, it follows that $p_C' < p_C$

p_C' is the percent share of cooperators in the next period. Similar reasoning holds for defectors also.

Rewriting the above equations yields the following:

$$p_C' - pc = \frac{pC * (WC - \overline{W})}{\overline{W}}$$

$$p_D' - p_D = \frac{pD * (WD - \overline{W})}{\overline{W}}$$

Assuming that the change in the percentage of strategies followed is small from one generation to the next, we can approximate the above equations with differential equations.

$$\frac{dpc}{dt} = \frac{\mathrm{pc} * (\mathrm{Wc} - \overline{W})}{\overline{W}}$$

$$\frac{dpD}{dt} = \frac{\mathrm{pD} * (\mathrm{WD} - \overline{W})}{\overline{W}}$$

where $\frac{dpc}{dt}$ and $\frac{dpD}{dt}$ represents the change in p_C and p_D with respect to time t, respectively. With these equations, we can compute the rate of change of increase /decrease in co-operators/defectors across various time points, which also depends upon the divergence between individual strategy fitness value and the average fitness value of the entire population. The above equations are known as replicator dynamics. This is used for continuous dynamics for evolutionary game theory. The replicator model for the prisoner's dilemma can be shown below (Alexander, 2021):

Figure 2.1 Replicator dynamics in prisoner's dilemma game (Alexander, 2021)

The leftmost point in the above diagram represents the state where all agents defect. The rightmost state shows the state where all agents cooperate. Since $W_C < \overline{W}$, for any value of p_C and p_D, we expect fewer cooperators in future generations compared to defectors. The rightmost state is unstable, while the leftmost state is stable. Defect is considered as a stable equilibrium which implies if some population starts following cooperate strategy, the evolutionary dynamics will push the system towards all defect states. The arrow on the diagram reflects the evolutionary path agents follow over time.

Below are other dynamics that researchers have proposed over time.

- Suppose each player selects any other player randomly from the population and compares the payoffs in the last round of play with that of the selected player. If the payoff of the other agent is higher, then the agent will switch to that strategy with the probability of a payoff difference. Schlag (1998) shows this equates to the replicator dynamics.
- Brown-Nash-von Neumann (BNN) dynamics. The replicator dynamics rely heavily on imitation. The expected payoffs being greater than the average population might indicate current population dynamics and have nothing to do with the superiority or inferiority of a particular strategy. Moreover, if a strategy is not currently being used in the current population, this is being ruled out from consideration to be adopted using imitation. The alternative to this could be to consider strategy S_i if the expected payoff from following S_i becomes higher than the average payoff of the population. BNN dynamics require agents to consider all alternatives available along with their payoff matrices so that new strategies can also enter the population if it is not being entered already.
- Smith dynamic. BNN dynamic does the comparison between the expected payoff of strategies not being used with the average population payoff. Smith dynamic on the other hand compares the expected payoff of the current strategy with the respective payoffs from following other strategies. Strategies with higher expected payoffs are expected to be adopted.

Both BNN and Smith dynamics are considered to employ more rationality than replicator dynamics, which depend heavily on imitation.

Suppose G is a symmetric two-player game and assume p(0) is a vector of initial strategy shares in the population. We assume p(0) to represent all the strategies, meaning they appear with a

frequency greater than 0. Given this context, Akin (1980) shows that replicator dynamics will push all strictly dominated strategies to zero over time. However, this does not negate that if all agents cooperate in the prisoner's dilemma game, replicator dynamics show that the population will remain in an all-cooperate state. This is despite cooperating being strictly dominated by defects in the prisoner's dilemma game. However, the same cannot be said for weakly dominated strategies. Consider the following game (Table 2.5).

Table 2.5 A hypothetical game with a weakly dominated strategy

	S_1	S_2
S_1	(1,1)	(100,0)
S_2	(0,100)	(100,100)

In the above game, S_1 weakly dominates S_2. A weakly dominated strategy can be a Nash equilibrium. In the above game, if both players adopt S_2, agents do not have an incentive to deviate from it.

In some cases, there is a divergence between the outcomes received from static approaches to evolutionary game theory (like ESS) and the dynamic approach to evolutionary game theory (like replicator dynamics). Weibull (1995) argues that a weakly dominated strategy cannot qualify as ESS. At the same time, Skyrms (1996) shows that weakly dominated strategies can appear as the outcome in replicator dynamics. This disagreement also extends to other forms of replicator dynamics. Hofbauer and Sandholm (2011) show a possibility of strictly dominated strategies

emerging as the outcome in the case of BNN and Smith dynamics. Consider the following game (Table 2.6).

Table 2.6 Rock-Scissors-Paper game with an additional strategy

	Rock	Scissors	Paper	Twin
Rock	(0,0)	(1, -1)	(-1,1)	(-1, 1- ϵ)
Scissors	(-1, 1)	(0,0)	(1, -1)	(1, - 1- ϵ)
Paper	(1, -1)	(-1, 1)	(0, 0)	(0, -ϵ)
Twin	(1 - ϵ, -1)	(-1 - ϵ, 1)	(- ϵ, 0)	(-ϵ , -ϵ)

The above game is Rock-Scissors-Paper with an additional Twin strategy. The twin strategy is exactly like the Paper strategy except that payoffs are reduced by some small amount $\epsilon > 0$. Twin is strictly dominated by Paper. However, it is possible that when using the Smith dynamic, some agents follow Twin strategy under certain conditions.

Another interesting result emerges about the divergence between ESS and stable states of the evolutionary dynamic model when we assume the population to be non-continuous. For example, assume agents interact in a local interaction framework, implying agents interact with other agents only in their neighbourhood and play Prisoner's dilemma game. Nowak and May (1992, 1993) show that in the local interaction model, the stable states depend on the payoff matrix. The paper shows that with values of T, R, P and S being 2.8, 1.1, 0.1 and 0, respectively, the local interaction model results agree with the replicator dynamics results, where the system converges towards an all-defect strategy. With the payoff values of T, R, P and S being 1.2, 1.1, 0.1 and 0, respectively,

the evolutionary results lead to a stable cycle oscillating between cooperators and defectors. This shows the state where cooperators and defectors coexist. With these payoff values, the local interaction model results vary significantly with the corresponding results from ESS and replicator dynamics. With these payoff values, the stable states achieved in the local interaction model have no corresponding analog in ESS or replicator dynamics. When we change payoff values to T, R, P and S to 1.61, 1.01, 0.01, 0, respectively, the local interaction model results show a continuous flux. This implies that defectors can invade regions earlier occupied by cooperators and vice versa. Therefore, there is no stable strategy found as defined in the classical dynamic theory.

2.6 Applications of the evolutionary game theory approach

Researchers have used evolutionary game theory to explain different aspects of human behavior. These include explaining altruistic behavior, agents' behavior when the nature of games is public goods, empathy, moral behavior, and social norms (Alexander, 2021). The focus of our research is on social norms evolution.

The static and dynamic approaches towards evolution take the macro/aggregative view towards evolution. Replicator dynamics and other dynamics start with specific strategies' percentage share in the population and try to assess how that percentage share changes depending upon its fitness value over time. However, there can be a connection between agents making decisions individually at the micro level and the population dynamics. Sandholm (2010) lays out a framework where it is assumed that individuals make decisions based upon two things: first, the current state of the population, which means the percentage share of strategies followed in the population, and second

the expected payoff from playing each strategy given the current state of the population. An individual learning rule or *revision protocol* can be created which uses these two pieces of information as input and creates a matrix of *conditional switch rates*. These conditional switch rate provides the probability of switching from one strategy to another conditional upon the current population state and expected payoff. Using this information, we can make inferences about the population-level dynamics as follows:

$$\frac{dpi}{dt} = \Delta_1 p_i - \Delta_2 p_i$$

The above equation shows the rate at which strategy p_i changes with time, and it depends upon the rate at which agents start using strategy p_i (Δ_1) subtracted from the rate at which agents stop using strategy p_i (Δ_2). The above equation shows the relationship between learning rules at the individual agent level and their impact at the population level. We leveraged the same thinking in formulating the best response function approach to dynamic evolution.

2.7 Norms evolution

This section briefly mentions some papers showing norms evolution in the context of different games. We talk about the structure of the game and the results achieved. We also clarify some parameters the authors have considered and the results from comparative statics. We would highlight those parameters in **bold** in this section.

Shoham and Tennenholtz (1992) deal with designing a multi-agent system in a way that promotes rapid convergence towards conventions. It is impossible to achieve this immediately because of the inability to impose norms through a central broadcast. The paper considers the bottom-up approach of norm formation wherein individual agents meet and update their strategies. It also talks about the impact of various factors like update functions, memory length, frequency of update, and amount of information exchanged in pairwise interactions on the convergence of conventions.

The paper considers convention as a single bit (1 or 0), and each agent starts with a randomly generated bit. In the repeated iterations, a randomly selected pair of agents meet and observe each other's bit. If it matches, it is called a success; otherwise, failure. A convention convergence is said to have been established when at least 85% of the agents reached a convention in multiple trails (800 – 4000). Further simulations are done to see the impact on convention convergence when there is a change in the parameter values.

Results show that when the **update frequency** decreases, there is a decrease in the efficiency of norm evolution. To measure the impact of **memory size**, they have explored two possibilities of limited memory (by default, in the base model, it is assumed full memory). Firstly, the model checks the case when the memory is restarted from time to time. It implies agents do not forget the current strategies (the ones they start within a particular iteration), but they forget the previous history. Results show that the convention evolution is higher when the **memory restart** happens less frequently. The second case is for when the agents consider a limited memory window. Results

show that convention evolution decreases as the memory length increases, implying *it pays to forget*.

The authors then consider how the results change when combining memory size and update frequency. Results show that when the updates become infrequent, implying there exists a long-time lag between agents' decision to perform strategy updates, the convention evolution is more efficient when agents' refresh their memory at periodic intervals than relying on the whole memory. This result is opposite from the authors' results when looking at these two parameters individually.

The paper also explores the results when agents communicate their current strategy and the full prior history of its interactions. The concept of *extroversion radius* is introduced, implying agents are comfortable sharing prior history to a certain extent. Results show that a small **extroversion radius** increases convention evolution efficiency compared to the scenario wherein only the current strategy is shared.

Axtell et al. (1999) discuss norms that can emerge at the social level with the micro and decentralized interactions of the agents. These interactions are self-reinforcing in nature. These social expectations once formed, can remain in the society for a longer duration and they may have no *apriori* justification. It means these norms resulted could be a result of purely random events. The paper talks about norms that govern the distribution of property and posits three possibilities. Equity norm is a state in which property is distributed equally among agents with no distinctions on agent types or what the paper calls as agents' "class". The second norm, discriminatory norm

is when the agents receive different amounts based on factors which might be fundamentally irrelevant. Examples include giving property to the eldest son in the house, and women/black should receive lower compensation, etc. The third possibility is the fractious state which shows when distributions have failed to coalesce.

The model assumes that two agents have equal claims to the property. Each agent can make three possible demands (Low 30% share), Medium (50% share), and High (70 % share). The rule of the game is that if the sum of two demands is more than 100%, each gets nothing, while each gets his/her share if the sum is less than 100%. In this case, there are three pure strategy Nash equilibria exist – namely (Low, High), (Medium, Medium), and (High, High).

In the first case, agents are assumed to be indistinguishable from each other. The paper used the terminology of *tags* like dark skin/ light skin, brown eyes/ blue eyes to distinguish agents. The game is played iteratively, and in each iteration, one pair of agents is drawn randomly from the population of N agents. Agents use their past histories to make the demand (Low/Medium/High) in each iteration which is impacted by their **memory length** m. Agents' decision to make a choice is also determined by some random noise (e).

Simulation results show that when N, m, and e are 100, 10 and 0.2 respectively, given that the initial state is random about the three possible strategies, then after 80 periods, Medium is the best response. However, inequitable regimes can also arise from a different initial state. Results show that with both **m** and **N/m** sufficiently high, agents are more likely to be in the equity region than in the fractious region assuming a small error rate e. Also, equity norms are *stochastically stable*,

implying once these are established, it is difficult to displace them. On the other hand, it is relatively easier to move out of the fractious regime if agents are in that state.

In the second model type, agents carry a **distinguishable tag,** e.g., dark skin/ light skin. The difference here is that tag also impacts the agent's decision to choose a strategy. Simulation results show that after 225 periods, equity state holds within each group, but discriminatory norms govern the relation between the two groups. Results show that when a dark agent type interacts with a light agent type, the dark agent demands High and light agent demand Low. This leads to higher payoff of dark agents (70) as compared to of light agents (30). This leads to class distinctions among agents, and these have emerged due to their own actions and strategies. In other words, it is *endogenous*. Light agent types realized that dark agent types will be dominating, so it is in their own interest to meet their demands instead of demanding High or Medium and get nothing. The similar reasoning holds for dark agent types too, where dark agents know the light agents will act passively, so it is in their own interest to utilize this opportunity and demand High.

Another interesting simulation result emerges after 260 periods when starting with a different **initial random state**. This tells equity norms to prevail within the darks (intra-type dark, dark), but lights observe a fractious state within the intra-type (light, light) society. This result shows the emergence of a divided underclass oppressed by a few elites. The inter-type result remains the same, i.e., dark dominates the light.

Epstein (2001) mentioned two features of social conventions. First, there are self-enforcing behavioural regularities, and second once norms become entrenched, we conform to them without

thinking much about it. Although thoughtless conformity is useful in some contexts, but not in others, for example, when norms of discrimination are prevalent in the society. This paper is focused on the second feature. It shows that agents engage in little mental calculations to think less about following the actions which are more prevalent as norm in the society.

The model assumes that each agent is represented at a fixed position on the ring and has two attributes. The first attribute is norms, and the second attribute is search radius. Two agents play a game in which (left, left) and (right, right) have a payoff of 1 for both agents. We can think of this as driving on the left or right. If one agent plays left and other plays right, the payoff is zero for both agents. So, there are two pure strategy Nash equilibria, left, left, and right, right. So, in this context, norms are either play left or right.

Agents update their norms according to a parameter called **search radius**, which implies how many agents to look for in the vicinity to decide what action to take. This parameter is typically heterogeneous across agents. For example, if its value is 5, agents would look for five agents to the left and five agents to the right. Agents update their radius according to a defined rule based on the relative frequency of the left norm and right norm to the left and to the right of the agent. Agents look for the majority within the radius and update norms. If, at the updated search radius, the count of left norms is higher than that of right norms, then they adopt the left norm. Hence, the combination of the norm updating procedure and search radius updating procedure is called the *best reply to adaptive sample evidence*. There is a noise element which shows the probability of agent adopting a random norm (left or right).

Results show that when we initially set all agents to the left norm and **noise** to zero, the left social norm remains entrenched, individual computing/thinking dies out, and the search radius is reduced to 1. Another result shows that when agents are being set to random norms (with noise still at zero), agents present at the point in the ring where there is equal probability of choosing any norm, need to introspect which norm to choose. Individual thinking and computing are very high at norm borders, with a noise level of 0.15 across all cycles. The average search radius settles at two. This shows that the question of looking how many agents to look for is answered *endogenously* through local agents' interaction. Higher noise levels increase the regions of local conformity amidst globally diverse patterns, and the range of average search radius is becoming higher than the previous runs of low noise levels. An interesting result is that when the noise is set at 1 (the maximum) for all the cycles, the average search radius confidence interval is between 4.53 and 4.63. This result is interesting since theoretically the maximum its value can go is very high in the total random world.

Sen and Airiau (2007) proposed a model in which agents learn from interaction experiences. Each agent learns its policy over interactions with other agents, implying agents interact with other agents repeatedly in a given scenario. Authors call this *social learning* different from when agents learn from repeated interactions playing against the same player. Multiple action combinations can result in the same payoff. The key question is whether the entire population can converge to a consistent norm.

The paper considers a scenario where each agent interacts with another agent. The agent selection is done randomly in each iteration. This way, every agent learns simultaneously with random

agents over a period. The paper focused upon experiments which have multiple pure strategy Nash equilibria with the additional condition of resulting in the same payoff. Also, in a similar work done in the literature, the agents interact with fixed set of opponents, but in this paper, agents interact with different opponents at each iteration. Also, the opponents may use different learning algorithm. Given this context, the paper tries to find an explanation on how a social norm can emerge in this social learning framework.

The social learning framework they consider is of rules of the road. Agents need to decide on which side of the road to drive in and who will yield if two agents arrive at an intersection simultaneously from neighbouring roads. An n person, m action stage game is defined with n=2 and m=2 depicting each interaction between two drivers. These stage games typically have multiple pure strategy Nash equilibria. Any agent can be randomly chosen as row player or column player. There is an assumption that agents know the payoffs, but they do not know the identities of agents. Therefore, each agent creates a pair of actions which they choose when they act like row player or when they act like column player. Each agent uses learning algorithms to decide what action to take and agents learn this independently, one as row player and another as column player. The first simulation result was to resolve the *social dilemma*. Simulation results of 1000 runs show that the population choose “left” norm 482 times and “right” norm 518 times. Hence, it shows that agents do not need to know other agents for norm emergence when agents learn.

To take it further, the paper shows how results change when a few parameters like **population size**, **number of actions** and **learning algorithm** change. Results show that it takes more time to for norm emergence with increased population size. The paper also considers the impact of **fixed**

agents, the agents following a specific norm and not having learning capability and, hence repeating the predetermined action. Results show that (with 3000 agents), when an equal number of agents play a fixed strategy (choose left or right), one of these two choices, drive towards Left or Right, emerges as a norm with almost equal percentage share.

The authors also look at the feasibility of social learning frameworks in isolated societies. It is usually believed that isolated societies follow norms that do not conform to global norms. The authors tried to investigate the extent or magnitude of isolation which can result in divergent norms to emerge. The experiment assumes two equal size population groups with the **probability of interaction** among two groups as p. Results show that the value of 0.3 or higher is enough for single norm to emerge in the entire population. When the probability is less (0.2 or less), this leads to divergent norms in society.

2.8 Role of sanctions in the effective implementation of social norms

There is sometimes a difference between conventions, social norms, and laws (Savarimuthu & Cranefield, 2011). The convention can be understood as a common expectation amongst agents regarding the agent's action or behaviour in each circumstance. With the rise of conventions, violations of these can be sanctioned, e.g., if driving on the right is sanctioned, then left-hand driving becomes a norm. The sanctions are mostly informal and distributed in social norms. Eventually, the social norms can transform into laws when imposed by an institution, e.g., laws governing driving behaviour.

To investigate the role of sanctions, Axelrod (1986) formulated a game in which players have the option of defecting. They can punish agents who they caught as defecting. The goal is to understand under what conditions cooperation can be promoted when there is a difficulty of achieving that. The paper assumes agents are bounded rational and used an evolutionary approach. This implies strategies which worked well in the past are more likely to be used again in future, while the ones which performed poorly are likely to be extinguished. Players play the game with each other agents and achieve a payoff that depends upon the agent's actions and the actions of others. In an evolutionary approach, there is no rationality assumption involved means the best strategy does not need to be always chosen. Effective strategies are more likely to be retained and reused by agents than ineffective ones at any given time. Players learn by trial and error, and in this process try to filter out strategies which they like to use in future and discard those which they don't want to use.

An individual (i) can defect, say, cheating on an exam. The opportunity of being caught (or seen) is given by a known chance, S. If player i does indeed defect, she will get a payoff of T (3) while each of the other players gets hurt slightly due to this and get a payoff of H (= -1). In the case of player not defecting, no one gets anything. If the other players see the defection, they may punish the defector. This punishment (P) is very painful, with a payoff of – 9. Also, since punishment is usually costly, the punisher pays the enforcement cost (E) equal to -2.

A player has two dimensions. The paper names first dimension as *boldness (B)* which indicates when the agent can defect, and this happens when the chances of being seen are less than the boldness level (i.e., $S < B$). The second dimension is the probability of an agent punishing other

agents seen as defecting which the paper calls as player's *vengefulness (V)*. The higher the V, the higher the possibility of being punished by other players. These two dimensions can take one of eight levels from 0/7 to 7/7 in the simulations.

Results show that norms collapse when players do not punish the players who have seen other players defect. To extend this further, in the next stage, players need to punish those who do not punish the defecting player. The model assumes that a player's vengefulness against non-punishment is the same as that against an original defection. A norm against defection is being observed in this case.

Voss (2001) opined that agents who apply sanctions should be sufficiently compensated for the associated costs involved in sanctioning. A behaviour conforming to social norms is self-enforcing if it results from Nash equilibrium. Hence, the first requirement for the effective enforcement of social norms is to identify the conditions under which an equilibrium of universal cooperation in social dilemma situations can be sustained. In addition, the requirement of identifying the conditions which lead to the emergence of threats or sanctions which can be perceived as credible. This translates to identifying the Nash equilibrium and subgame perfect Nash equilibrium.

Suppose there is a prisoner's dilemma game as below (Table 2.7):

Table 2.7 Prisoner's dilemma game with 0 payoffs for both defects

	Cooperate(C)	Defect(D)
Cooperate(C)	R,R	S,T
Defect(D)	T,S	0,0

In the above payoff matrix, T > R > 0 > S. In the case of repeated interactions, Pareto optimal Nash equilibria may exist in the prisoner' dilemma case where both players agree to cooperate if the discount parameter or "shadow of the future" (Axelrod, 1984) is high. This can also mean the probability of the game continuing for another period. If it is assumed that both players play cooperate strategy forever, every player will be expected to receive the following payoffs with 'a' representing the discount rate:

$$R + aR + a^2R + \ldots\ldots\ldots = \frac{R}{1-a}$$

Trigger strategies enforce cooperation in repeated games. Trigger strategies are in equilibrium if it satisfies the following conditions:

$$a >= a^* = \frac{T-R}{T-0} = 1 - \frac{R}{T}$$

Where T = maximum payoff from unilateral deviation.

This pair of trigger strategies is subgame perfect equilibrium if a>= a*. For examples when T = 10 and R = 5, this leads to a* = 10-5 / 10-0 = 5/10 = 0.5. With R = 7, a* becomes 10-7/10-0 = 3/10 = 0.3. Hence, the higher difference between R and T results in a higher threshold for a*.

Under appropriate constraints and conditions, repeated interactions are shown to generate sanctions endogenously which can enforce cooperation. There is a large number of equilibrium (sub-game perfect) in repeated interactions for different trigger strategies, so there is also an equilibrium selection problem. The traditional game theory treats the selection problem exogenously, while evolutional game theory explains how coordination equilibria are determined endogenously.

Given the coordination and bargaining problems in the context of repeated games, it implies that path dependencies matter and that including information on historical data is important. This may result in situations where norms that emerged are stable but suboptimal. The following hypothesis is suggested from a theoretical standpoint:

1. Probability of agents following cooperation behavior increases with the shadow of the future (a).
2. An increase in the cost of cooperation which implies (T – R) decreases cooperation.
3. An increase in the cost of conflict (T – 0) increases cooperation.

Other hypotheses can also be derived which impact the emergence of norms like the size of the group, social network and linkages, multiplexity (implies two agents are connected in more than one relation) etc. A larger group size, a more connected social network, and a higher multiplexity leads to requirement of lower threshold or critical value of the discount factor compared to what is required to support universal cooperation.

Trigger strategy is not the only kind of sanctioning mechanism which is being used in real-life interactions. Another example of sanctioning mechanism is exit threats, where norm deviators are barred from engaging in future interactions. “Incremental” sanctions are another mechanism where every agent bears a small burden of sanction. However, the sum of these small incremental sanctions is a huge punishment to the norm deviator.

The paper then references Axelrod (1986) on norms and meta-norms. There are two stages in the norms game. In the first stage, players choose between C (cooperate) and D (defect) to the prisoners’ dilemma. The second stage involves sanctioning or punishments which involves players reacting to the decisions made during the first stage. Both players can punish their partners with a negative sanctions ‘s’. Also, they can select s*, i.e., not to employ a sanction. The target actor incurs a punishment cost of -p while the cost of sanctioning equals – k. The actor, whom his/her partner punishes, receives the payoff of T – p while the partner’s payoff is s – k. The following three propositions arise:

1. A Nash equilibrium of cooperation exists, i.e., p >= (T – R). T – R is the cost of cooperation. This condition implies punishment cost is equal to or higher than the cost of cooperation.
2. Nash equilibrium of cooperation is subgame perfect provided k <= 0.
3. Subgame perfect equilibria of universal cooperation exist in infinitely repeated norms game with discounting even when k > 0 provided the discount factor is large enough.

The justification for costless sanctions (k < 0) can be explained in three aspects. In this case, the condition is that the norm game is not repeated. Assume that the partner has defected, and the

defect is irreversible. This would imply that punishing the defector would further decrease the partner's payoff if $k > 0$

I. In interactions involving social approval, the refusal to provide social approval to a group member can be taken as a positive reward ($k < 0$) if there is a higher possibility of provisioning the social approval being costly.
II. Second, sanctioning the defector may involve some "emotional" rewards, which can outweigh the material cost associated with punishing.
III. Third, the norm game can be a part of an extended larger game wherein the punisher might want to acquire a particular personal reputation. With this, the punisher would like to apply sanctions even if the material costs are high enough. This is done to build reputation which might be helpful in future interactions with other agents.

The third possibility above implies that in infinitely repeated games, the punishment costs will be outweighed by the rewards associated with cooperation in the future. Therefore, even with positive sanction costs ($k > 0$), the threat to punish defection is optimal and credible. It should be noted that in equilibrium, these sanctions need not be implemented. A credible and optimal threat can ensure universal conformity in equilibrium.

Norms are typically dependent on the paths taken in previous iterations. Learning processes for cultural beliefs are modeled in evolutional game theory. Some of the recent approaches are the following:

1. A population of bounded rational agents is represented as a stochastic dynamical system wherein agents are engaged in social interactions iteratively.

2. Bounded rationality implies that recent history and not the entire history is being used.
3. Agents do not always select the best response toward their expectations about others' behaviour. There is some element of randomness involved.
4. There is a possibility of stochastic shocks corresponding to mutations within the biological evolution.

The evolutional approach treats the evolution of conventions as endogenous compared to exogenous, which has been treated in the traditional game theoretic approach.

2.9 Role of randomness in agents' decision making

Young (2015) studied norms with the help of stochastic evolutionary game theory. It is based on the perturbed best response approach proposed in the paper.

Suppose G is a two-person coordination game and is symmetric. Agents are chosen randomly from the population. Populations are assumed to be sufficiently large, and this translates to action of any single agent having a small impact on the whole dynamics. In addition,

- Agents are expected to have insufficient information about what is happening in society. They have information about their own past experience and some information about their neighbours' experiences in a social group.
- Agents choose the best response given the information available which is myopic, but they sometimes deviate.
- Agents interact randomly with others, which usually involves some bias coming from their geographical or social neighbours.

In the small group's interactions, a single agent has the potential to change the evolutionary dynamics toward a new norm. This is especially true if that single agent is the leader. However, norm entrepreneurship is risky because they lose a lot in terms of their reputation if their efforts fail, and if they succeed, it can significantly enhance their status. In the case of large group settings and where the evolutionary time frame is sufficiently large, myopic best response behaviour is usually the baseline assumption. In the works by Axelrod (1984) and Axelrod (1986), there is even less emphasis on the rationality of the agents. Agents do not optimize or form beliefs by design. They are given strategies, and agents choose those strategies where the reproductive success is high depending on how well they are fared with other strategies in the competition regarding their fitness values.

The paper uses the perturbed best response framework to address the following questions:

- Determine the conditions or states which can show convergence into a social norm using interactions of many dispersed agents.
- Which norms are more likely to emerge than others?
- Will the outcome vary in the intermediate and in long run? Or are there any unique features of the dynamics?
- Welfare implications of the resultant outcomes.

The symmetric two-person game is G, and a single population of agents interact and play G. Agents' actions are denoted by X. Given a pair of actions (x,x'), u(x,x') denotes payoff to the first player and u(x',x) denote payoff to the second player. Each pair of players (i, j) has an importance weight $w_{ij} >= 0$. Weights are assumed to be symmetric, i.e., $w_{ij} = w_{ji}$ for all $i \neq j$. And w_{ii} value is assumed to be 0.

We assume that there is a population of n agents having identical utility functions. Time period is t = 1,2, 3, and is assumed to be discrete. At the end of period t, there is a state which is an n vector x^t where x^t belongs to X and is the choice of action by each player i where i is between 1 and n. X = Xn denotes the state space. At the start of period t+1, one agent i is randomly selected to update (agents update their strategies asynchronously). Provided the choice of all other agents except i as $X^t{}_{-i,}$ the utility of agent i from choosing action x is as follows:

$$U_i\,(x, X^t{}_{-i}) = \Sigma_j\, w_{ij}\, u(x, x^t_j)$$

When the agent updates her strategy, a new action is chosen $x_i^{t+1} = x$ with a probability which is increasing in U_i. i's choice maximizes U_i with high probability and chooses other actions with low probability. This is called the perturbed best response model.

The paper gave a reference to two benchmark perturbed best response models. First is the uniform error model wherein agent chooses an action which maximizes U_i with probability 1-e. Agent chooses the action randomly with probability of e.. The second model is the logit or log-linear response model. In this case, i's probability of action x at time t+1 is given below.

$$P[x^{t+1}{}_i = x] = e^{\,\beta\, Ui\,(x,\, Xt\,-i)} / \Sigma_{y \in Xi}\, e^{\,\beta\, Ui\,(y,\, Xt\,-i)}$$

In the above equation, e refers to the exponential function. β is a non-negative real number. This equation shows that the probability of deviating from the best response decreases with the increase

in the magnitude of loss in utility. In the limiting case of β value as infinity, the best response is chosen with a probability of 1.

In the case of asymmetric games, there are two action spaces, X for row player and Y for column player. Let (x^t, y^t) belong to X * Y pair of actions agents choose in period t. History through period t is written as:

$h^t = ((x^1, y^1), (x^2, y^2), ….. (x^t, y^t))$

The recency aspect is considered by looking at the history of the last m periods. When an agent plays at time t+1, a random sample is drawn from the historical actions of opponent members in past m periods. Agent uses a perturbed best response to the actions taken by opponent agents. The agent chooses action using either a uniform error or logit model.

Young (1993) talks about the dynamics of the process involved, which evolve expectations and behaviours using theoretical game theoretic tools. When agents experiment and make mistakes, there are no absorbing states. The paper showed that it depends upon the stationary distribution, which shows the frequency of different states observed in the long run. The absorbing state is defined as a state with repetitions of pure strategy Nash equilibria in succession. The paper shows that this stationary distribution is concentrated towards a subset of pure strategy Nash equilibria in case of lower probability of a mistake. This is called *stochastically stable equilibrium*. The paper showed that this outcome will be observed with 100% probability assuming the noise/mistake is small.

Young and Foster (1991) address why we can expect cooperation to result among unrelated individuals when there is an option to cheat among players, and cheating would have resulted in higher payoffs for any of the players. The paper shows that if payoff rates are variable, tit-for-tat may be favoured in the short run and not necessarily in the long run. The goal is to show that results are sensitive to how the model is specified, and it requires focusing on the evolutionary processes involved. With the presence of stochastic effects in the dynamical system, it's long-run behaviour can be altered, and paradoxically as the noise level reduces, the alteration may become larger. The paper used a prisoner's dilemma game to demonstrate these results. The paper showed different results from stochastic and deterministic versions of the evolutionary model. Consider the following prisoner's dilemma (PD) game (Table 2.8).

Table 2.8 Prisoner's dilemma game for IPD

	Cooperate (C)	Defect (D)
Cooperate (C)	(3,3)	(0,5)
Defect (D)	(5,0)	(1,1)

In an iterated prisoner's dilemma (IPD), two players play PD game repeatedly. There is a stopping probability of s and both players are aware that engagement will end with probability s after the current round. The first round happens with 100% probability; the probability of second round occurring is 1 – s. The expected number of rounds per engagement equals 1/s. Suppose there are three strategies: first strategy where players always cooperate (C), second strategy where players always defect (D), and third strategy with players playing tit-for-tat (T). At each time t, n^t =

(n_C^t,n_D^t,n_T^t) represents the count of C, D, and T players, respectively. N^t is the total number of agents. $p^t = (p_C^t,p_D^t,p_T^t) = n^t / N^t$ represents the proportion of C, D, and T players respectively.

In the first possibility, the paper examines the behaviour of a deterministic system where expected values of C, D and T strategies are used. Suppose the D player engages with the T player. In round 1, D gets 5 and T gets 0 from the payoff matrix mentioned earlier. After round 1, both get 1 (using tit-for-tat, D will be played after that). The remaining expected number of rounds after the first round = 1/s – 1. Expected payoff from D's strategy equals 5 + (1/s-1) * 1 = 4 + 1/s. Similarly, we can compute other expected payoff values, resulting in a below-expected payoff matrix (Table 2.9).

Table 2.9 Prisoner's dilemma game with a tit-for-tat strategy

	C	D	T
C	3/s	0	3/s
D	5/s	1/s	4 + 1/s
T	3/s	1/s - 1	3/s

When C and C type agents interact, the payoff in first round equals 3. The expected number of rounds in this case becomes 1/s, which results in expected payoff value of 3 + (1/s – 1) * 3 = 3/s. When D and C-type agents interact, this results in a payoff of 5 for D-type agents and a payoff of 0 for C-type agents. This leads to an expected payoff value of 5 + (1/s – 1) * 5 = 5/s.

The population evolves with the following equation:

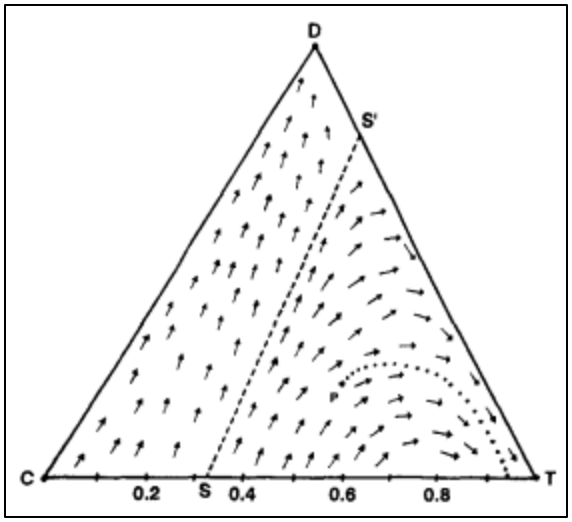
$$n_i^{t+1} = n_i^t A_i n^t$$

Where i = C, D, T. A_i is i^{th} row of the payoff matrix, A. Similarly,

$$p_i^{t+1} = p_i^t A_i p^t / [p^t A p^t]$$

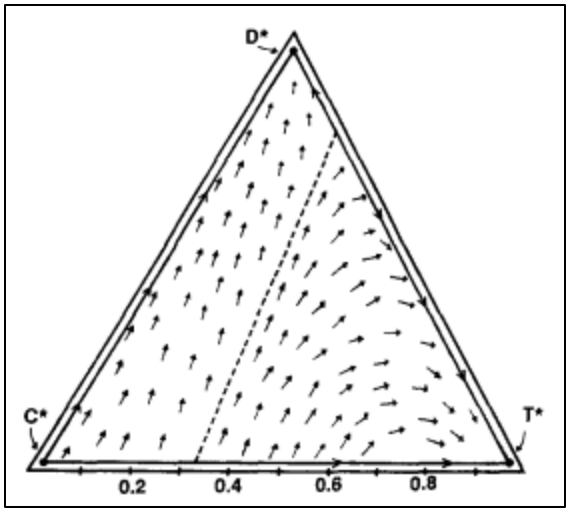
State space is diagrammed in Figure 2.2 below. Results show that points to the right of line SS' leads to evolution towards T outcome, and towards the left it leads to evolution towards D. If there are more T players relative to D players, T will be evolutionary favoured. Point D is locally stable means any path in the vicinity of D leads to convergence towards D. Point D is also evolutionary stable, implying the system will revert towards D. However, T is not necessarily evolutionary stable, implying once there is a divergence from point T, there will be a push towards T but it may not reach it. This is possible when D players wiped out before it happens with C players. In this scenario, the evolutionary path can end near point T on the CT line. Any point on the CT line is considered stationary, implying C and T players are considered equally fit in the absence of any D players to differentiate.

Figure 2.2 The dynamical system with s = 0.2 (Young & Foster, 1991)



The above argument assumes that strategies can die permanently, which is not quite true in the long run. Firstly, most populations have the characteristics of immigrating agents from outside, and secondly, mutations in the background ensure that new and previously existing strategies are constantly being pushed into the system. If we assume the mutation rate is constant (across C, D, and T types), then the evolutionary process wound not touch the edges and is bounded away from edges, as shown in Figure 2.3. Point T* indicates the situation wherein all, excluding the newcomers, are playing T.

Figure 2.3 Another possibility of the dynamical system with a marginal background mutation rate (Young & Foster, 1991)



The above results are based on the deterministic evolution approach. The paper then analyses the stochastic system approach toward evolution. This leads to revised payoff of 5+X to the D player when they interact with T player where X is any geometric random variable with mean = (1 – s)/s and standard deviation = $\sqrt{(1-s)}/s$. The payoff to the D player and T player will be larger if the engagement is longer (if s is small, (1 – s / s) or 1/s – 1 will be higher, hence 5 + any positive number will be higher). Here the payoffs are variable between each pair of players. It is assumed that populations change according to the following dynamic equation:

$n_i^{t+1} = n_i^t A_i^{\wedge} n^t + rN^t$

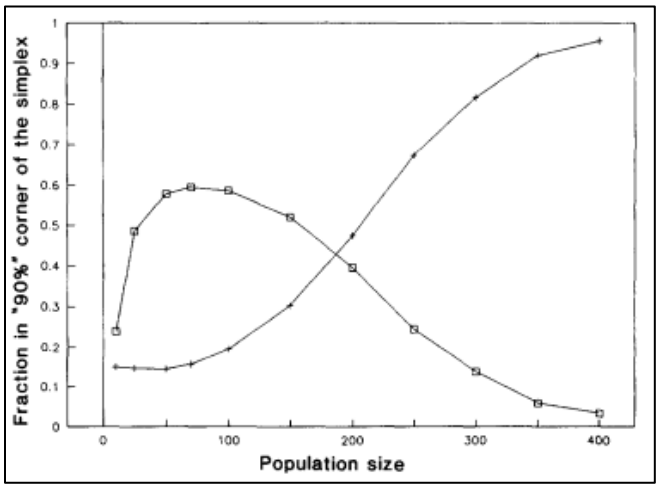
Where r is the mutation rate, $A^{\wedge}$ is the payoff matrix with random component. And the expected payoff matrix equals A.

The paper investigates the behaviour when the population is fixed at N. With small N, the stochastic term becomes large, and the evolutionary process is found to be erratic around the state

space. The opposite results are obtained, when N is large enough, and with small stochastic variation, the system is found to be closer to the deterministic path.

The authors assume a mutation rate of 0.1% implying new C, D, and T players are being added into the existing population at 1 per thousand of the existing population in each period. In the case of N = 100, simulation results show that population would consist of at least 90% T players with a probability of 0.59. It also shows that the population would consists of at least 90% D players with a probability of 0.18. This implies that tit-for-tat is favoured 59% of the time. Also, it is shown that the evolutionary process is either near D* or near T*, which has a higher possibility. However, the intermediate regimes or co-operators show a very low probability. Results are reversed when N increases, and noise decreases. In the case of N = 300, D is favoured (81%) over T. Almost everyone became a defector with the further increase in N (Figure 2.4). Figure 2.4 shows the tit-for-tat graph with a square symbol on the line and the defect graph with a + sign on the line.

Figure 2.4 Probability of population composition of 90% T-players (alternatively, 90% D players) with varying population sizes (Young & Foster, 1991)



Results show that when there are higher number of players near T*, the D and C players become almost non-existent, and pure cooperation becomes almost as fit as tit-for-tat. There is a possibility that agents following C are doing better than agents following T for a certain duration if CC engagements happen to be larger than TT engagements. The effort required to climb out of the D* basin is more difficult and less likely than moving away from T* in a given time frame. This is explained when computing the path of least resistance between D* and T*. The resistance along the path from T* to D* is relatively lower than that of the corresponding resistance from D* to T*. This leads to higher expected transition time from D* to T* as compared to from T* to D*. Therefore, as the population increases, there will be more certainty with respect to agents following D strategy.

The above argument shows that tit-for-tat may be unviable in the long run, but similar arguments can also be said about noncooperative behaviour (defect). Suppose agents follow the strategy where there is a defection on the first round and agents continue to follow that until the opponent starts cooperating. Post that, agents play tit-for-tat. This is called disguised tit-for-tat (DT). With three strategies, D, T, and DT, the evolution will happen towards tit-for-tat in the presence of low positive mutation rate and noise. Therefore, the defect may also be subject to degeneration.

The paper assumes that the length of engagements between different players is random, and this is considered as a primary source of randomness. Another possibility of noise could be noisy channels of communication, which implies a small probability associated with the player misreading the action taken by his/her opponent.

2.10 Role of social networks in evolution and norms

Young (2015) talks about the role of social networks in the agent-based framework in norms evolution with the help of two case studies, the naming game and the bargaining game.

The naming game case study shows how naming conventions can arise through trial-and-error learning. There is a pure coordination game wherein two people need to suggest names for the pictures shown independently and simultaneously. There is a positive reward to agents if both agents propose the same name and pay a penalty in the form of negative reward when they propose different names. There are no restrictions on the potential names that can be proposed.

There are 25 rounds in each trial. In each round, the same face is shown to all agents, and this is repeated in all rounds. The total number of agents are from 24 to 96. At the beginning of a given round, each agent is paired with some other agent, and they are given 20 seconds to respond. Agents do not know the identities of the other agents. Agent pairs are drawn randomly from the edges of a fixed and undirected network. Agents do not know with whom they are being paired or the network structure they are part of. Agents are aware of the names their opponents have provided in the preceding rounds. In this way, agents gather information about the names in vogue among the people they are being paired with.

Each agent is placed at the node of a fixed network. Two agents are selected randomly in each round and play the naming game. The names proposed by opponent agents are known to the partner agents at the end of a given round. The history till round t is written in pairs as follows,

$h^t = ((x^1, \delta^1), (x^2, \delta^2), ….. (x^t, \delta^t))$

Where x^t_i is the name proposed by player i in round t, δ^t shows an indicator function. It implies $\delta_{ij}{}^t$ = 1 when players i and j were paired together in round t; otherwise, its value equals 0. Each player i knows the names their opponent agents have provided in all previous rounds at the end of round t..

The paper considered two different network types. First, a ring network in which each player is connected to four agents in the neighbourhood. Second, a complete graph in which each agent is connected to all other agents. Ring network results show that by round 10, there are several distinct local norms emerged. This is explained by nearby agents proposing same name in small groups. There are five total local conventions emerged. Complete graph results show the possibility of dominant convention establishment by round 10, and by round 24, this has displaced all other conventions, despite facing more failures in the initial rounds. These patterns emerge without agents knowing the network structure or the system's state.

The second case study (Bargaining) is around the evolution of bargaining norms. Some agents are buyers who interact with a single seller and repeatedly play the Nash demand game. In each period, each buyer and seller can only be paired once. There is a fixed pie and buyers and seller demand their shares from that pie. If the total demanded pie exceeds the total pie size, both will not get anything. Otherwise, they will get their share proportional to their demands. The buyer receives information about the historical shares demand by seller from other buyers. The seller uses the

perturbed best response function to sample the historical demands made by buyers. Buyers have full knowledge of this information.

A trial consists of a fixed group of 6 buyers and a seller. Buyers are located at the nodes of a fixed network. Each trail contains a total of 50 rounds. In every round, each buyer and seller play the Nash demand game once. Buyer is not being informed on what is being demanded by the seller; she is only told whether her demand is compatible with what is being demanded by the seller. Demands are non-negative numbers, and the total size of the pie is 17 units.

When buyer b expects to make a demand, she is having information of what the seller demanded from a random sample of previous matches with b's neighbours in the preceding six rounds. So, the buyer gets some initial information about the seller's demands through her network. The size of the buyer's sample is $2d_{b,}$ where d_b is b's neighbour's count. The buyer also has information from her experience with the seller on which of her demands led to successful outcomes. On the other hand, the seller chooses perturbed best response and takes samples from the historical demands made by buyers in the preceding six rounds. The seller chooses the myopic best response in response to the frequency distribution of buyer demands with a 95% probability and a random response with a 5% probability.

The process converges into a bargaining norm when buyers and sellers demand the same amount for at least 5 out of 6 buyers and this should repeat for at least four consecutive periods. Results show that the convergence happens in 11 out of 13 trails provided the network is regular of degree 4. (Figure 2.5) The number of buyers ranges from 9 to 14. The stochastically stable norm is

dependent on the minimum sample size when different subjects have different sizes. With all else being equal, the smaller the minimum sample size, the less its members can expect to get. This is supported by the results achieved considering the star network (Figure 2.6). This network has five agents with one neighbour each; hence, the sample size is 2. The central agent connects with five other agents, so the degree is five, and the sample size is ten, as shown below in Figure 2.6.

Figure 2.5 Degree 4 network (Young, 2015)

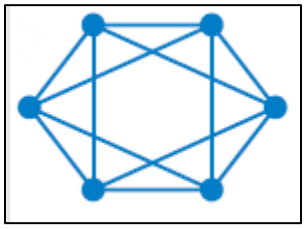
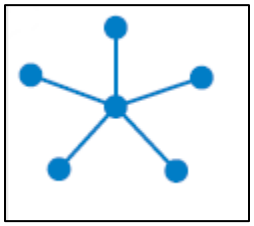


Figure 2.6 Star network (Young, 2015)



Gallo (2014) finds that the buyers' demand approximately 10% more in the regular network as compared to the star network. The explanation for this is due to the role of random distractions in the long-run stability of norms which can cause a shift in agents' expectations. A buyer is expected to lower her expectations given only a few instances of higher demands made by the seller, which happens in the case of star network. However, an agent with a large sample size (in the case of a regular network) expects more high demands by the seller to lower her expectations which

anyways is less likely to occur given the nature and rules of the bargaining game, which states no agent will get anything if the demanded pie exceeds the available pie.

Kohler et al. (2001) studied the impact of network structure on the usage of contraceptives. Birth rates in developing countries have declined significantly. However, the decline observed in not uniform. It is observed that the timing and speed of fertility transition varies in different communities. Researchers investigated other factors which can explain this phenomenon, including the disparity in social norms.

A woman's network is defined as the set of women with whom she has interactions and discuss family planning matters. The network's density is the proportion of links members have with each other. So, for example, a woman network with n individuals can have a maximum of n * (n -1)/2 links. The authors propose that keeping the network size fixed and increasing the density creates more opportunities for applying social pressure on the group members. The authors argue that it does not provide any significant new information and is redundant. This will eventually lead to a decrease in the probability of contraceptive adoption when there are other members in the population who have adopted. This result is explained by redundant information keeping network size and other social and economic characteristics as fixed. On the other hand, women in the denser network will have a higher probability to follow the dominant norm, whether the dominant norm is using contraceptives or against it.

The paper further tested these predictions from a survey done in rural Kenya for approximately 500 women. They found a statistically significant difference in behaviour between those living in

isolated villages and those living nearby market activity. Results show that the probability of adoption is higher in sparser networks when the proportion of existing adopters is less than 2/3. This is explained by the sparser network providing more independent information than the denser network for a given level of adoption. Relatively lower density values define a sparse network. In the non-market regions, the trend is reversed. When the proportion of existing adopters is less than 2/3, higher densities reduce the probability of adoption. However, when the existing adopters' proportion is more than 2/3, higher densities increase the chances of adoption, all else being equal. While the trend is consistent in market regions, more sparse networks result in more adoption. Below, the Figure 2.7 graph is for the market region, and the Figure 2.8 graph is for the non-market region.

Figure 2.7 Probability of using family planning. Obisa. (Young, 2015)

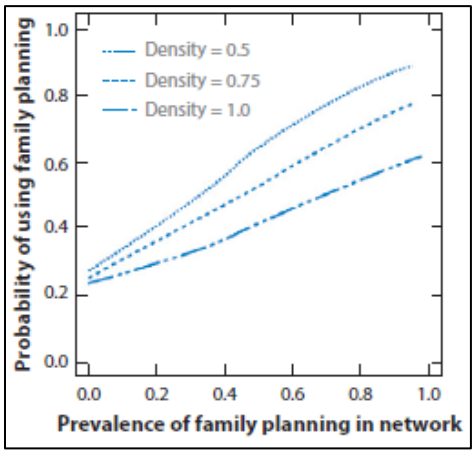
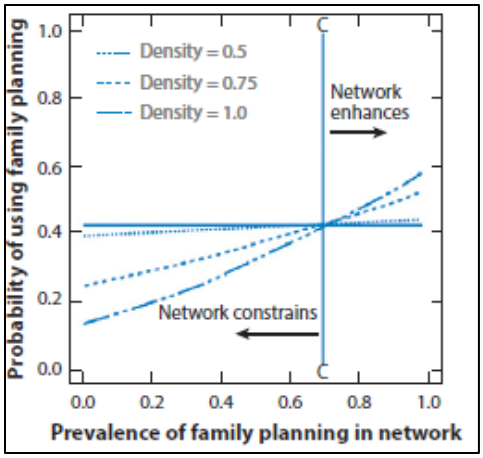


Figure 2.8 Probability of using family planning. Owich, Kawadhgone and Wakula South. (Young, 2015)



Bowles and Gintis (1998) talk about how communities help to promote cooperative behavior in agents. The paper defines community as a structure of social interaction that fosters frequent interaction among the same agents. Examples include residential neighbourhoods, ethnic groups, and other business or trade-related groupings. The paper argues that communities promote pro-social norms. Pro-social norms are those norms whose increased adoption in the population leads to an increase in the average well-being of the overall population. For example, truth-telling, choosing a 'cooperate' outcome in a prisoner's dilemma, etc. Communities impact the norms evolution because they impact agents' interactions with each other. This, in turn, influences the benefits and costs associated with actions which are norm conducive.

The paper studies how communities promote social norms by adopting an evolutionary view. The paper defines *differential replication,* which implies certain aspects of behaviours or traits are copied, retained, and followed by the other durable agents in the population while other traits are discarded. Differential replication can occur from agents who copy norms with higher frequency in the population, or it can come from some privileged culture models like the following which parents or teachers have done, or it can be by the exercise of power by nations.

The paper argues that communities foster frequent interactions among agents of the same type, resulting in low-cost access to information about other agents. There is a tendency in which agents prefer to interact with their community members as compared to interacting with outsider agents. Moreover, lastly, there is a restricted migration of agents to and from other communities. These four characteristics of agents who belong to communities promote pro-social behavior. The paper explains this with the help of the Prisoner's dilemma game, where a community faces a coordination problem. The paper provides three ways in which the structure of a community can helps induce agents to cooperate. First, the cost of getting information about other agents is low when frequent interactions among community members take place. This results in agents in actions that build a reputation with other agents. Authors call this the reputation effect. Second, since there is a higher chance that agents will engage with the same agents in the future, there is an incentive to act, keeping in mind other agents' interests. This is referred to as the retaliation effect. In a large population where many different communities are involved, pro-social agents are more likely to interact with communities following pro-social norms rather than anti-social norms. This is referred to as the segmentation effect. All three effects, reputation, retaliation, and segmentation, allow communities to support pro-social traits. There is another effect, which authors call as parochialism effect which enhances these three effects. The parochialism effect does not directly influence the agent's behavior, but it does so by increasing the three effects under appropriate conditions.

The paper shows that an equilibrium condition for a trait to be evolutionary stable is equal payoffs from the different traits. This means a small increase in population following the x trait; it should

increase the replication propensity of alternate trait y more than the x trait, leading to the y trait becoming more favourable in replication and lowering the x trait population.

The authors then showed how different effects observed in communities help in pro-social norms evolution and the different parameter restrictions using the prisoner's dilemma game. The paper showed four different reputation effects based on the capacity of the community to provide low-cost information about the agents with which one interacts. First, if the information cost is low, this results in trusting behavior equilibrium. Second, the amount of trust behavior is also of higher intensity when the cost is low. Third, the average payoff is generally higher for all the agents in the population when the cost of accessing information is low. Moreover, lastly, the percentage of agents choosing to defect would be higher when the information is costly to obtain and vice-versa.

For the retaliation effect, the authors show that the tit-for-tat strategy is an evolutionary stable strategy when there exists at least a certain proportion of defectors in the population, say ϵ, and if the proportion of defectors goes below this proportion, the differential replication will lead to its extinction from the population.

The segmentation of the population is explained by the high entry and exit costs that characterize communities. It implies that agents in a particular community prefer to interact with other agents who belong to the same community more often than with agents who belong to different communities. This leads to the clustering of agents who cooperate in a prisoner's dilemma situation and prefer to interact with agents who also play cooperate as it would be advantageous to other agents while playing defection would be costly to the other agents. This leads to biased pairing,

which pairs likes with likes and increases the payoffs for agents exhibiting pro-social traits. The paper considers a parameter, σ, which shows the degree of segmentation in the population. The paper then shows some value of $\sigma < 1$, which proves universal cooperation to be an equilibrium even in the case of a single-shot prisoner's dilemma. This is called the critical value of the degree of segmentation, σ', which happens when the percentage share of cooperators is 100%. Similarly, there is another level of segmentation degree, which is σ'', such that for $\sigma > \sigma''$, some cooperation level may be sustained as an equilibrium, resulting in when the percentage share of cooperators is 0%. If the proportion of cooperators in the population is stable, an increase in segmentation degree will lead to increase cooperation in the population.

To explain the parochialism effect, the paper reconsiders the retaliation effect. It now assumes that a fraction μ of each group relocates in each period. μ value be lower when there is higher entry and exit cost. Interactions take place only within groups. Migration changes the composition of an updated population where some existing agents move out and are replaced by new agents. The paper shows that when tit-for-tat players are lower than the population average, an increase in migration leads to decreased frequency of tit-for-tat players and increased defectors in equilibrium.

In a recent development, Chatterjee et al. (2023) introduced a different variant of ESS which is adaptive to games played on networks. In the original ESS, it is assumed that a certain percentage of agents follow a specific strategy, and all these agents constitute the population. This paper argues that when agents' network information is taken into account, it results in a different variant of ESS which is a refinement of Nash equilibrium but with the additional properties of being robust to the agents' degree of the network. Agents are assumed to play a bimatrix game with other agents

in their neighbourhood. Agents' payoffs are an aggregation of the payoffs they receive when interacting with all their neighbours.

The paper first defines a payoff function when agents are connected in a network. Suppose G represents finite, simple, and undirected graphs and assume g ∈ G. There is a set of nodes V = {1, 2, n} and a set of edges represented by E ⊂ V*V. Assume g_{ij} = 1 if there is an edge connecting agents i and j, else it is assumed to be 0. Suppose S represents the different strategies that agents can choose. The strategy profile is denoted as s = (s_1, s_2, … s_n). The network payoff of agent i is defined as below:

$$\pi_i (s \mid g) = \sum_{j=1}^{n} gij\ \pi (si, sj \mid g) = \sum_{j \in Ni} \pi (si, sj \mid g)$$

A strategy profile s* is the Nash equilibrium of the game when played on a network if it satisfies the following condition:

$$\pi_i (s^* \mid g) >= \pi_i (s_i; s_{-i}^* \mid g)$$

for each i = 1, 2, ……, n

The paper argues that a strategy profile s* is evolutionary stable (ESS) when played on a network g if it satisfies the following two conditions:

1. s* is Nash equilibrium of the network game, and
2. For every node i ∈ V and any sk (≠ si*) ∈ S, the following should hold:

$$\left(\frac{1}{di}\right)\pi_i\,(s_i^*\,;\,(s_k, s_k, \ldots\, s_k)) + \left(1 - \frac{1}{di}\right)\pi_i\,(s_i^*;\, s^*_{-i}) >= \pi_i\,(s_k;\, s^*_{-i})$$

In the above equation, d_i is the degree of player i, which implies the number of edges connected with agent node i. This shows that ESS is weighted by the degree of players in the presence of networks. If d_i value is large (approaches ∞), the second condition resembles the Nash equilibrium condition. If we rearrange the second inequality condition, we can rewrite the same condition as follows:

$$\sum_{j \in Ni}(\,\pi\,(sk, sj*) - \,\pi\,(si*, sj*)) <= \left(\frac{1}{di}\right)\sum_{j \in Ni}(\,\pi\,(si*, sk) - \,\pi\,(si*, sj*))$$

The left-hand side of the above inequality shows the agent i's change in payoff when the agent switches from s*, and the right-hand side shows the average change in payoffs if the change happens in the agent i's neighbourhood. This implies that agent i's prefers to stick to the strategy s*, and if the change is inevitable, it prefers to have that change come from the neighbourhood rather than changing its own strategy. Therefore, this condition can be interpreted as testing the agent's resistance toward an invading strategy in the network game.

2.11 Other factors impacting the evolution of norms

Kandori et al. (1993) talk about various factors that impact the actions/strategies that would evolve and the speed with which the evolution would take place. The paper assumes there is limited ability on the player's part to receive, decode and act upon the information they get while playing games. Comparing this with Foster and Young (1990)'s paper, where authors argued that the

stochastically stable equilibrium depends on the additional factors of dynamics like, the speed of adjustment. The difference is due to the *source of randomness*. In this paper, randomness is introduced at the agent level, while Foster and Young (1990) assume randomness at the population level.

Another factor that impacts results is the payoff structure of the game. Large jumps are likely, but these results are independent of the deterministic dynamic. On the other hand, gradual small changes play a critical role in Foster and Young (1990), and the results produced are not independent of the speed of adjustment.

Upsetting the good or efficient equilibrium is more challenging compared to upsetting the bad one. This is due to the former requires more mutations. When the mutation rate is small, the evolutionary system is usually at a good equilibrium.

The focus of the paper is on 2*2 symmetric games as follows (Table 2.10):

Table 2.10 A hypothetical 2*2 symmetric game

a,a	b,c
c,b	d,d

There are three types of games considered:

- Dominant strategy equilibrium which satisfies this condition of (a-c) * (d-b) < 0
- Coordination game scenario which satisfies these conditions (a > c, d > b)

- Games involving unique symmetric equilibrium, with mixed strategies satisfying these parameter restrictions ($a < c, d < b$)

The paper shows that expected wait times matter for reaching the particular strategy in addition to the probabilities assigned by limiting distribution. Upsetting the wrong equilibrium takes less time compared to upsetting the right equilibrium.

In this paper, randomness is at the individual player level, generating aggregate randomness. At the same time, Foster and Young (1990) assume continuous time, continuous state space formation, and aggregate randomness. The paper assumes that population is finite, and mutations are independent. Equilibrium outcome is challenged by large shifts from equilibrium to the basin of attraction of the other equilibrium. Brownian motion models are used in Foster and Young (1990) which have continuous sample paths. In those models the only way to move out of the equilibrium are the gradual local movements. Therefore, speed of adjustment is not important in this paper's formulation.

In the case of more general games, the long-run equilibrium is dependent on the Darwinian adjustment process. We may observe a sub-optimal equilibrium depending upon the initial state and on the relative speeds of adjustment of row players versus column players. Therefore, the selection of a long-run equilibrium might be determined by the Darwinian adjustment process specification, and not just solely by the payoff structure of the game. Randomness leads to coordination on a particular equilibrium which may or may not be Pareto dominant.

Young (1993) posits that any specific outcome/action can emerge as equilibrium and becomes entrenched conventionally. It need not be because of this being prominent but because it is chosen by the dynamics followed for the evolution. The convergence takes place asymptotically, provided that the game has properties of acyclic best reply structure and agents take decisions with some randomness. Majority of the time, agents would be playing Nash equilibrium. However, some Nash equilibriums are more likely to be selected than others. If one Nash equilibrium is observed with high probability; this equilibrium is said to be stochastically stable.

The paper assumes that agents play a fixed n-person game. Agents play the game once each period. A small portion of the history is known to every player. Players occasionally make mistakes. The paper assumes that there is no learning at the individual level. It implies agents are being replaced with similar agents of the same type once the agent plays the game. The paper defines the concept of *adaptive* play which means agents take decisions based on other agents' actions in the recent past, and these decisions are not necessarily optimized.

Results show that for weakly acyclic games adaptive play converges to a pure strategy Nash equilibrium with 100% probability, provided the samples are sufficiently incomplete, and the players take decisions with zero randomness. However, this need not necessarily hold true for general n-person games. Adaptive play need not converge to a pure or mixed Nash equilibrium. If players make mistakes, then the process has no absorbing states. This leads to a stationary distribution which shows the relative frequency of different states observed in the long run. If the probability of agents making mistakes is small, then this stationary distribution revolves around a particular subset of pure strategy Nash equilibrium. Usually, this equilibrium is being given most

of the weight. This stochastically stable equilibrium is expected to be observed with a 100% probability when the noise element is very small. This concept differs from evolutionary stable strategies (ESS). ESS is a strategy (or frequency distribution of strategies) which gets restored if it is hit by small one-time shock. A stochastically stable equilibrium is a distribution repeatedly restored when small random shocks constantly buffet the evolutionary process.

The methodology deployed in the paper is similar to Foster and Young (1990) and Kandori et al. (1993). The paper considered an evolutionary learning framework defined by 2*2 symmetric games. In each period, every agent interacts every other. The strategies which show higher payoffs are adopted with a higher probability than the ones which produce lower payoffs, and this is based on the assumption that there is a small positive probability of agents making mistakes.

The paper shows that for 2*2 coordination games, the risk-dominant equilibrium is the unique stochastically stable equilibrium. For the coordination games having more than two strategies, the stochastically stable equilibrium need not be risk-dominant or Pareto optimal. For games where cycling is built into the best reply structure of the game, adaptive play need not converge. Many games do not have a cyclic best reply structure, e.g., coordination game, common interests game etc.

Incomplete sampling, which implies stochastic variations into the player's responses helps to break out of suboptimal cycles. If agents make mistakes, the stochastic process does not converge to an absorbing state because there are no absorbing states. Mistakes constantly divert the process towards outcome which are away from equilibrium. If we assume that agents make all mistakes

and that the probabilities of making mistakes are time-independent, then the process leads to a unique stationary distribution; and its asymptotic behaviour can be studied. When the probability of agents making mistakes is small, this leads to a stationary distribution which is concentrated around a particular convention (or a subset of conventions). These are stochastically stable conventions which will be observed with positive probability in the long run when the noise is small but positive.

The paper shows that risk dominance and stochastic stability differ in these two concepts. There are different notions of resistance which are employed. Strategy i is said to risk dominates strategy j if the mistakes required to go from j to i are lower than the corresponding mistakes to go from i to j within the subgame consisting of only these two strategies. Stochastic stability takes a broader perspective and looks at all transitions from i to j, including those that involve other strategies. Another distinction is that risk dominance is defined only when there is one strategy that risk dominates every other in pairwise comparisons. Stochastic stability relies on global criteria. Risk dominance selects the equilibrium, assuming such equilibrium exists, which is easier to achieve from every other equilibrium considered individually (). Stochastic stability, on the other hand, selects the equilibrium that is easiest to get into from all other states, which includes both equilibrium and non-equilibrium states.

Nishizaki et al. (2009) demonstrates the possibility of using a neural network model to decide how agents make decisions and adapt over time depending upon their past experiences. The neural network model is specific to each agent and outputs the action the agent should follow. The inputs to the model include individual factors specific to the agent and external or macro factors specific

to the society or environment in which the agent operates. The input to the neural network includes agents' choices in the prior period, populations obedience rate in the previous period, personal taste and preferences of agents, individual agents' utility in prior periods, aggregate utility of all agents in the prior period, degree of belief of individual agent for the social norm. Agent's choice in the prior period is represented as a binary flag where 1 represents if agent i obeys the social norm and 0 otherwise. The population obedience rate, scaled from 0 to 1, can be interpreted as an index of the society's social norm, which is determined by aggregate information of actions taken by all agents. Personal taste or preferences of an individual agent can be interpreted as an agent's preference towards reputation if they obey or disobey the social norm. The agent is assumed to behave adaptively; hence individual agent utilities and the utilities of all other agents in previous periods are considered in the decision-making. In the model, there is a penalty for disobeying the social norm; hence the degree of belief of the individual agent is also used as one of the inputs. Agents who are believers are assigned a score of 1.

2.12 Agent-based modeling

Tesfatsion (2003) uses the term *agent-based computation economics (ACE)* to describe the phenomenon of autonomous interacting agents who learn from their interactions and experiences. ACE researchers use computational frameworks to understand the evolution of decentralized market economies under certain controlled experimental and parameter restrictions. The paper mentioned conditions or concerns which drive ACE research. First, this area can explain the emergent global behaviour in terms of how global regularities evolved without top-down planning and control and why few specific regularities and why not others. The second focuses on the

implications of following a mechanism and understanding what social outcomes will evolve from the repeated attempts of self-seeking agents to exploit a mechanism to their advantage. The paper mentioned a couple of research areas conducive to ACE modeling, including the evolution of behavioural norms, formation of economic networks, bottom-up modelling of market processes, etc.

In this section, we highlight some of the characteristics of agent-based modeling and how these are different from classical game theory. We also briefly talk about evolutionary models, their aggregative or discrete types, and how best these reflect human behaviour in the real world. We end the section briefly by mentioning the importance of social structures or networks in the evolution of norms (Alexander, 2007).

- Heuristics incorporate a common body of beliefs in humans acquired through participation in a common culture.
- Humans are generally bounded rational and rely on less-than-perfect calculations derived from heuristics and rules of thumb. There is an element of dynamic aspect in which people's aspirations vary over time.
- People's reactions to our choices can significantly affect the resultant outcome.
- Problems of strategic choice tend to characterize better the choice problems people face in social contexts.
- Most interdependent choice problems in society have a structure where the outcome reflects a mutual agreement among rational persons.

- The expected utility concept under traditional game theory involves assumptions like continuity, substitutability, transitivity, and monotonicity. These assumptions may not hold in all contexts.
- The social structure of society is important for the cultural evolution of norms.
- Social norms can be viewed as a general heuristic whose adoption ensures that an individual will generally do better if they are followed than if they are not. These norms are culturally evolved responses to repeated interdependent decision problems in a socially structured environment.
- Evolutional game theory assumes that agents are bounded rational and interact repeatedly. This is similar to the repeated games literature in traditional game theory. This setup provides a better way of analyzing interdependent decision problems.
- Evolutionary models contain a minimum of two things. First, the representation of the current state of the population. And second, the dynamical rules or policies which explain how the current state of the population changes over time. Population representation is done through two models, which include a continuous/ aggregative model or a discrete model. Continuous/aggregative models use global statistics to represent the population using frequency distribution of specific genotypes. An example of a continuous model is replicator dynamics. Replicator dynamics represent the state of the population by the frequency of each genotypes or phenotypes.
- Aggregative models assume that individuals agents' characteristics are unimportant in the population since differences between individual agents are lost when global statistics is used to represent the population state.

- Discrete methods like agent-based models keep track of the individuals' identities, including information such as phenotype, spatial position, location in a social network and so on.
- Replicator dynamics is represented in differential equations, ensuring that exact solutions can be found and expressed in mathematical equations. On the other hand, agent-based models rely on computer simulations.
- Aggregate models cannot differentiate between two agents, and these models cannot represent the structure of the society and social interactions. To incorporate social structure into agents' decision making, it is required to specify the agents' relations and social network.
- Replicator dynamics assumes agents engage in random interactions, i.e., the probability of any two agents meeting is equally likely. This assumption is false for human interactions. Usually, interactions with friends influence behaviour more easily than with strangers.
- Including structure in evolutionary game theoretic models makes a huge difference in the long-term behaviour of the model. Population states which are unstable in replicator dynamics can attain stability in structured agent-based models. The other reason to understand its importance is that incorporating structure into agent-based models enables us to model situations where long-term convergence behaviour more closely approximates the behaviour found in real human interactions. Including structure can show cooperation outcome to persist in Prisoner's Dilemma game, selection for universal stag hunting in the Stag Hunt game, and fair division in the Nash bargaining game, among others. Including structure is necessary to explain outcomes because many of these results are not obtainable

under replicator dynamics. The nature and kind of structure embedded also plays an important part in the evolution of norms.

- In agent-based modeling, the number of agents is defined along with their neighbourhood and how these agents are connected with a specific social network. The number of strategies is assumed to be finite. Each agent interacts with their neighbour agents and calculates a payoff score against each strategy. Agents can use different tactics or learning rules to decide what strategy to follow. Some of these include imitating the best neighbour, imitating with probability proportional to success, imitating the best average payoff, best-expected payoff (highest payoff in the next generation) etc. Another unique characteristic of agent-based modeling is the possibility of key agents. Key agent is one whose adoption of a different strategy sparks a large-scale shift in the strategy frequencies found in the populations. On the other hand, replicator dynamics cannot model key agents. Replicator dynamics provide a deterministic dynamical model which is insufficient to capture the intricacies of cultural evolutionary models.

Chapter 3: Response Functions and Norms Evolution

3.1 Introduction

There has been much interest lately in the dynamic aspects of social norms evolution. As observed in the previous chapter, there have been two approaches for building normative behaviour in agents. In the first approach, institutions force certain behaviours which are required to be abided by each agent, called the prescriptive approach. The second approach is the bottom-up approach, wherein individual agents need to interact with each other, leading to the emergence of norms that govern agents' behavior. The bottom-up approach has been studied by Sen and Airiau (2007), Aydogmus et al. (2020) which explain how cooperation can evolve among self-centered non-related agents through repeated interactions with no explicit forward-looking expectations on the part of the agents. Our research is also focused on leveraging the bottom-up approach of norm evolution.

Some of the earlier literature on norms evolution has tried to approach this topic using a theoretical model, while the recent literature is more focused on using simulations. Simulation-based approaches lead to a larger number of different possibilities for social connections, which can be tested and may not be possible with the traditional approach (Alexander, 2007). Most of the existing literature performed simulations using a hypothetical game, and it is difficult for the reader to replicate the same results or to check the impact of tweaking parameters. There has been less visibility around how the results have been derived and what tool or programming language has

been used to perform simulations. Against this background, we have tried to bring transparency into the process of simulations and how this can help answer the question of norms evolution.

To explain how agents make decisions, we use two approaches in this chapter that agents can follow to select action at any given point depending upon the prior history of their interactions. In the first approach, agents choose the strategy having the maximum payoff in response to all possible opponents' strategies from historical interactions. This is called an exhaustive payoff approach. The second approach, called the expected payoff approach, considers the relative frequencies of different strategies played in the past and uses these as weights to calculate an expected payoff. We define the strategy followed most frequently as the most likely candidate for being called a norm. We provide a computational framework which can potentially answer the questions raised in the literature, like cooperate outcome ('stag') in stag hunt game rather than risk-dominant outcome ('hunt') or possibility of 'cooperate' outcome in prisoner's dilemma game or equal share in symmetric bargaining game or possibility of agents playing strictly dominated strategies in the population etc. (Harms & Skyrms, 2008; Hofbauer & Sandholm, 2011; Alexander, 2007; Alexander, 2021). The framework can also reinforce the view from Young (1993) that any action can be perceived as a conventional way of playing the game once agents have been used to playing that action. This chapter answers research questions like how does the sequence of actions impact norm selection. It also answers the impact of historical events shaping norms. The expected payoff approach addresses the impact of payoff functions and its values on norms. We also investigate the impact of memory length on agents' decision-making. The impact of agents taking decisions randomly is also discussed.

To generalize and to ensure the reusability of these approaches for the reader, we have created an open-source Python library, *game-simulator*, which can be used to assess evolution for any combinations of m*n payoff matrix, memory length, time period, initial states etc. This is the first library or tool we are aware of which can be used for this analysis by the reader without writing the code or program from scratch. Some of the applications of this library include explaining prisoners' dilemma situations where there is a choice between cooperating and defecting and a higher incentive to defect. Suppose one firm is evaluating whether to spend money advertising a new product. This decision about spending and how much money to spend would depend on the competitor firm's decision to spend on advertising. If both firms decide not to spend on advertising, it will lead to savings, and both can maintain their market share. However, if one firm decides to spend money on advertising and the other does not, this would lead to the firm losing the market share that decides not to advertise. Hence, both firms may eventually end up spending money on advertising. However, depending on history, firms can choose to take different actions. We can create and test multiple if else scenarios depending upon different initial states and memory lengths with this library.

We tested the two response functions on widely applicable 2*2 finite normal-form games: prisoner's dilemma, matching pennies, battle of sexes, coordination, and stag hunt. Some of these games have multiple pure strategy Nash equilibria, some have strictly dominant strategies, and some do not have any pure strategy Nash equilibrium. A simulation exercise is performed to evaluate which of the strategies is played more frequently compared to others when agents play these games iteratively.

Prisoner's dilemma results show that when agents are allowed to make mistakes, it means choosing non-recommended actions randomly with a certain probability e and taking recommended actions with probability (1 – e), then (cooperate, cooperate) or (cooperate, defect) also has chances of emerging as the norm. This posits the possibility of sustaining a non-Nash and Pareto superior outcome as a norm. Games involving multiple pure strategies Nash equilibria like stag hunt, battle of sexes, and coordination game results depend upon the initial history, memory length of the agents, payoff values etc. Matching pennies game results did not show any clear trend. In general, results show there is a possibility of outcomes that can emerge as a norm that are neither Nash equilibria nor Pareto efficient.

The rest of the chapter is divided into seven sections. Section 2 discusses a brief literature review. The following section discusses methodology, which explains how agents decide what action to take at any given period. We then discuss the simulation results of the five games considered. Section 5 provides a detailed explanation of the impact of memory size. Section 6 lists some applications where this computational framework can be applied. The chapter concludes by briefly mentioning how to use the Python library.

3.2 Literature review

The existing literature on norms evolution can be divided into multiple branches depending upon the approaches followed. There is literature that solved norms evolution problem using the deterministic /analytical approach and some literature that has used the computational/simulation approach or a combination of these two. The existing literature has focused on how norms evolve and depend upon various parameters. Some of these parameters include the payoff structure of the

game, population size, memory length, strength of relations with other agents, time taken to reach a norm, methods which agents follow to update their actions during each period of the game, and randomness in agents' actions (Young & Foster, 1991; Kandori et al., 1993; Young, 1993; Young, 2015; Alexander, 2007).

The focus of our research is on norms evolution using computational methods. As described in Chapter 2, norms evolution from a computational standpoint has been described in three different ways. First, some finite normal-form games are defined along with their payoff matrix. The strategy pair being played most frequently is considered a norm pair strategy (Axelrod, 1986; Shoham & Tennenholtz, 1992; Young, 1993). Secondly, there is a finite game with specific strategies and the corresponding payoff values. At the end of the simulation period, we get the information on the agents' revised strategy distribution. The strategy being played most frequently is considered a norm (Young &Foster, 1991; Axtell et al., 1999; Tesfatsion, 2003; Nande et al., 2020). The third approach considers agents' social networks (Alexander, 2007; Young, 2015). In this chapter, we confined ourselves to the first computational approach of norm evolution out of these three.

The first approach can be used to answer questions like which of the Nash equilibrium can emerge as the norm under different parameter restrictions when multiple pure strategy Nash equilibria exist. It can also decide parameter restrictions for Pareto efficient and inefficient outcomes. Axelrod (1986) was one of the first papers which used the evolutionary approach to norms evolution. Young (1993) talks about any specific equilibrium can be seen as the norm. It may not

be due to its prominence but is simply the result of the dynamics of the process. Therefore, there is a path dependency.

Norm emergence also depends upon the process followed for evolution. The dynamics are defined by the model specified (agent-based discrete or aggregative), payoff values, population size, and the level of noise present in agents' decision-making. Young and Foster (1991) showed different results from stochastic and deterministic versions of the evolutionary model. These results are supported by Kandori et al. (1993), which showed that long-run equilibrium results depend upon the game's payoff structure. It also raises the importance of noise present in agents' decision-making. Randomness might lead to promote coordination for a particular equilibrium which may not satisfy the conditions of Pareto dominancy. The discrete model investigates the agent level information and assesses how it impacts agents' decision-making (Alexander, 2007). Axtell (1999), Sen and Airiau (2007) are some papers that used a discrete model approach to show norm evolution computationally. It shows results being dependent upon initial states, agents' memory length, population size, and the learning framework agents use to make decisions during each simulation state. This chapter focuses on norm evolution using a discrete model approach.

3.3 Methodology

We have tried two approaches to evaluate how agents make decisions in the existing period based on the history of previous periods. We assume that agents have complete information about the previous history. In the first approach, agents respond by doing what is best for them against the strategy played by their opponents. In this approach, the agent follows every feasible path available

at any given time to determine the best response action (Young, 1993). The second approach takes into consideration the payoff values corresponding to different strategies. In this approach, agents compute the expected payoff values using the relative frequency of strategies played by opponents during the memory window as the weights. Agents select the strategy which has the maximum expected payoff. These two approaches are detailed further below with the help of an example. We call the first approach an exhaustive best response approach and the second approach an expected payoff approach.

3.3.1 Exhaustive best response approach.

We start by defining the following 2*2 payoff matrix (Table 3.1).

Table 3.1 A hypothetical 2*2 game

	Column player B		
Row player A		S_{1B}	S_{2B}
	S_{1A}	2,2	0,2
	S_{2A}	1,0	1,1

We can see two pure-strategy Nash equilibrium of the above game (2,2) and (1,1). We assume the memory length of 2 periods. This is a kind of recency bias where only the immediate previous two periods' outcomes are considered to compute the best response in the next period. It has been

shown that higher memory length results in a decrease in the efficiency of norm evolution. Shoham and Tennenholtz (1992) described the efficiency of norm evolution in terms of the number of trials where at least 85% of agents reach a convention (follow the same strategy) compared to the total number of trials. A decrease in norm evolution efficiency implies a decrease in the number of trials where the condition of at least 85% of agents reaching a convention is satisfied.

In Table 3.1 above, the row player is Player A, and the column player is Player B. Player A has two strategies to choose from, S_{1A} and $S_{2A,}$ while Player B has S_{1B} and S_{2B}. We refer to strategies with their index, so the first row in the above payoff matrix for row player (A) is strategy 0 (S_{1A}), and the second row for row player is strategy 1 (S_{2A}). Similar reasoning holds for column players also. Column player (B) strategy 0 is $S_{1B,}$ and strategy 1 is S_{2B}. The row player payoffs from playing strategy 0 are 2 or 0, corresponding to column player playing strategy 0 or 1, respectively. Column player payoffs from playing strategy 1 are 2 or 1, corresponding to row player playing strategy 0 or 1, respectively.

We assume that no history is available when the game starts, and agents select each action randomly with equal probability (Axtell et al., 1999; Epstein, 2001). Suppose the outcomes are (0,1) and (1,1) in periods 1 and 2, respectively. These numbers refer to the position of strategies for both row and column players, not the actual payoff values associated with those strategies. From these initial history pairs, the row player has played 0 and 1 while column player has only 1. The row player would play 1, which would fetch a higher return when the column player plays 1. The column player can play 0 and 1 since the row player has played both 0 and 1 in the previous two periods. Therefore, in period 3, there could be two potential outcomes, (1,0) or (1,1).

With this at the beginning of period 4, the history available would be {(1,1), (1,0)} or {(1,1), (1,1)}, which is represented as period two outcomes followed by period three outcomes. If we assume the history at the beginning of period four as {(1,1), (1,0)}, this implies row player has played 1 in both periods while the column player has played both 0 and 1. With this column player would play 1 in period 4, while row player can play both 0 and 1. This results in a potential outcome of (0,1) or (1,1). If we repeat the same process assuming history as {(1,1), (1,1)}, this will mean both row and column players would have an incentive to play 1 in period four also. This translates to (1,1) as the outcome.

At the beginning of Period 5, period three and period four outcomes constitute history. Period 3 potential outcomes include (1,0) or (1,1). Period 4 potential outcomes include (0,1) or (1,1) using history of {(1,1), (1,0)} and include (1,1) using history of {(1,1), (1,1)}. Based upon the above potential outcomes in periods 3 and 4, the potential three histories are available at the beginning of period 5.

{(1,0), (0,1)}, {(1,0), (1,1)}, {(1,1), (1,1)}.

In the above histories, the potential outcomes (0,1) or (1,1) in period 4 came from using {(1,1), (1,0)} as the history at the beginning of period 4; hence we have used (1,0) as the 3rd period history candidate for first two potential histories at the beginning of period 5. The standalone outcome (1,1) in period 4 was derived from using {(1,1), (1,1)} as the history at the beginning of that period; hence we have used (1,1) as the 3rd period history candidate for the outcome (1,1) in the third potential history for period 5. In period 5, we would again look at the best response outcome of

row, and column players using each of these histories as the input and construct the potential outcomes and histories pairs. This process is also explained in Table 3.2.

Table 3.2 History and action pairs for exhaustive best response approach

Period	History available at the beginning of the period	Potential/Actual outcome
1		(0,1)
2		(1,1)
3	{(0,1), (1,1)} →	(1,0), (1,1)
4	{(1,1), (1,0)} →	(0,1), (1,1)
	{(1,1), (1,1)} →	(1,1)
5	{(1,0), (0,1)}	..
	{(1,0), (1,1)}	
	{(1,1), (1,1)}	
6	…	…

The arrows in Table 3.2 point toward the potential outcomes achieved using the history pairs available at the beginning of the corresponding period. The same process goes for the number of time periods we specify for the simulation run. At the end of the simulation run, we take the frequency count of the pairs in the 3rd column of the above table. For 2*2 games, there could be four potential pairs. The pair with the highest frequency count among all four possibilities is considered a norm candidate.

Below we explain the functioning of this approach in an algorithmic form. It provides individual row and column player strategies against the history defined by memory length. In the context of 2*2 game, the output of this function is row player and column player strategies out of 0 and 1.

Algorithm 1 Exhaustive best response approach

Require: memorylength, initialhistory,timeperiod,payoffmatrix
Ensure: bestresponserowplayer,bestresponsecolumnplayer

```
for t 1 to timeperiod do
    distinctcolumnchoices = ∅
    for k IN memorylength do
        distinctcolumnchoices = unique(initialhistory) ∋ columnplayerchoices
    end for
    distinctrowchoices = ∅
    for k IN memorylength do
        distinctrowchoices = unique(initialhistory) ∋ rowplayerchoices
    end for
    bestresponserowplayer = ∅
    for k IN distinctcolumnchoices do
        bestresponserowplayer = max(payoffmatrix)[rowplayerpayoff] ∋
k=columnplayerstrategy
    end for
    bestresponsecolumnplayer = ∅
    for k IN distinctrowchoices do
        bestresponsecolumnplayer = max(payoffmatrix)[columnplayerpayoff] ∋
k=rowplayerstrategy
    end for
    bestresponserowplayer = unique(bestresponserowplayer)
    bestresponsecolumnplayer = unique(bestresponsecolumnplayer)
end for
```

The inputs required are *memorylength*, *initialhistory*, *timeperiod* and *payoffmatrix*. *memorylength* indicates how many periods agents need to look back. *initialhistory* specifies the choices made during the preceding periods defined by memory length. *timeperiod* indicates how many periods we need to run simulations for. And *payoffmatrix* specifies the payoff values of row and column players. The function returns the output as best response strategies for row (*bestresponserowplayer*) and column player (*bestresponsecolumnplayer*).

- The loop starts with time period t.
- An empty set *distinctcolumnchoices* is defined.
    - *distinctcolumnchoices* is filled with distinct strategies played by column player during the initial history.
- An empty set *distinctrowchoices* is defined.
    - *distinctrowchoices* is filled with distinct strategies played by row player during the initial history.
- An empty set *bestresponserowplayer* is defined.
    - *bestresponserowplayer* is filled with row player strategies which have a maximum payoff for row player against the strategies played by column player in previous periods.
- An empty set *bestresponsecolumnplayer* is defined.
    - *bestresponsecolumnplayer* is filled with column player strategies which have a maximum payoff for column player against the strategies played by row player in previous periods.
- The unique row and column player strategies are returned using *bestresponserowplayer* and *bestresponsecolumnplayer* respectively.

The below function provides all the possible combinations of individual choices resulting from the individual row and column player strategies. It creates a cartesian product of row and column player strategies in pairs. The example outputs look like (0,1), (1,0) etc.

Algorithm 2 Strategy selection

Require: timeperiod, responsefunction()
Ensure: cartprod

```
for t 1 to timeperiod do
    rowbestresponse = responsefunction().bestresponserowplayer
    columnbestresponse = responsefunction().bestresponsecolumnplayer
    cartprod = cartesianproduct(rowbestresponse,columnbestresponse)
end for
```

The inputs required for algorithm 2 are *timeperiod* and *responsefunction()*. *timeperiod* indicates the time for which we want to run the simulations for. *responsefunction()* is the output of the response function, providing individual row and column player strategies. The output of algorithm 1 is used as an input in this function. The function provides output in action pairs for different row and column player individual strategies, which implies a cartesian product (*cartprod*) of different row and column player strategies.

- The loop starts with time period t.
- *rowbestresponse* is the row player strategies from algorithm 1.
- *columnbestresponse* is the column player strategies from algorithm 1.
- A cartesian product is created from the individual row and column player strategies and is the output of this function (*cartprod*).

The output of this algorithm is then used to construct history and action pairs as defined in Table 3.2.

We can see multiple applications where this approach can be used. In this approach, agents are essentially simulating “what if” scenarios to inform their decisions. They engage in counterfactual reasoning. Below are a few characteristics of agents where they may like to follow this approach.

- Agents care about their reputation. By exploring counterfactuals, they aim to establish consistent patterns of behavior.
- Agents weigh potential gains and losses. Considering counterfactuals helps them assess risks associated with different actions.
- Agents learn from past experiences and adapt their strategies. Counterfactual thinking helps them update their beliefs about opponents’ preferences and behavior.
- Agents may use heuristics (mental shortcuts) to simplify decision-making. Considering counterfactuals could be a heuristic to navigate complex strategic environments.
- Agents are myopic and consider only recent histories while making decisions.

Below are few applications, where this thinking might be applicable:

- Investment Decisions: Investors consider counterfactual scenarios when evaluating investment options. They assess potential gains and losses under different market conditions.
- Contract Design: When designing contracts, parties consider various contingencies. What if one party breaches the contract? How would the other party respond?
- Antitrust Cases: Antitrust authorities analyze counterfactual scenarios to assess the impact of mergers or monopolistic behavior. They compare actual outcomes with what might have happened in the absence of anticompetitive practices.

3.3.2 Expected payoff approach.

The second approach calculates agents' expected payoffs against different strategies. We use the relative frequencies of opponents' actions during the memory window as weights to compute the expected payoff. The strategy resulting in a higher expected payoff value is chosen. Consider again the hypothetical game as defined in Table 1 above.

Suppose the initial 2 period history is the same as previously (0,1), and (1,1). The row player has played 0 and 1 once. Hence, the probability of a row player playing 0 and 1 in the next period is 0.5. The column player has played 1 in both periods; hence the probability of the column player playing 1 in period 3 is 1, and the probability of playing 0 equals 0. Both row and column players compute expected payoffs using the probability values as weights. These weights indicate the agent's expectations about opponent agents' playing 0 or 1 strategy. With these weights, we can create below a matrix of expected payoffs for row and column players against both strategies in period 3.

Table 3.3 Expected payoffs in period 3

Strategy	Expected payoff (row player)	Expected payoff (column player)
0	2* 0 + 0* 1 = 0	2* ½ + 0 * ½ = 1
1	1 * 0 + 1 * 1= 1	2* ½ + 1* ½ = 1.5

Since the expected payoff values are higher with strategy 1 for both players, both players would choose 1 in period 3. This results in an outcome of (1,1) in period 3. The history available at the beginning of period 4 becomes {(1,1), (1,1)}. Since both row and column players have chosen 1 in 2 preceding periods, the probability of playing 1 becomes 1 for both row and column players. This translates to the probability of playing 0 becoming 0 for both players. Using the same computation as followed in Table 3.3 above also leads to (1,1) outcome in period 4. We can consolidate the information in a tabular form as below (Table 3.4)

Table 3.4 History and action pairs for expected payoff approach

Period	History available at the beginning of the period	Potential/Actual outcome
1		(0,1)
2		(1,1)
3	{(0,1), (1,1)} →	(1,1)
4	{(1,1), (1,1)} →	(1,1)
5	{(1,1), (1,1)}	..

The interpretation of this table and its columns is similar to Table 3.2. If the expected payoff is the same from both strategies, we consider both strategies for successive time periods. And the table would look something similar to the exhaustive function approach with multiple potential histories and outcomes across different time periods. At the end of the simulation run, we consider the frequency count of the pairs in the 3rd column of the above table. The pair with the maximum frequency count is considered the norm candidate.

Below we explain the functioning of this approach in an algorithmic form. The algorithm below is a replica of algorithm 1 and has a similar objective but creates output using the expected payoff approach.

Algorithm 3 Expected payoff approach

Require: memorylength, initialhistory,timeperiod,payoffmatrix
Ensure: bestresponserowplayer,bestresponsecolumnplayer

```
for t 1 to timeperiod do
    columnchoices = ∅
    for k IN memorylength do
        columnchoices = (initialhistory) ∋ columnplayerchoices
    end for
    rowchoices = ∅
    for k IN memorylength do
        rowchoices = (initialhistory) ∋ rowplayerchoices
    end for
    rowplayerpayoff = ∅
    for k IN unique(columnchoices) do
        rowplayerpayoff = payoffmatrix[rowplayerpayoff] ∋
k=columnplayerstrategy
    end for
    strategycountdb = table(columnchoices,count)
    strategycountdb(prob) = strategycountdb(count)/memorylength
    rowplayerexpectedpayoff = ∅
    for k IN len(rowplayerpayoff) do
        rowplayerexpectedpayoff=rowplayerpayoff(k)*strategycountdb(prob)
    end for
    bestresponserowplayer = max(rowplayerexpectedpayoff)
    columnplayerpayoff = ∅
    for k IN unique(rowchoices) do
        columnplayerpayoff = payoffmatrix[columnplayerpayoff] ∋
k=rowplayerstrategy
    end for
    strategycountdb = table(rowchoices,count)
    strategycountdb(prob) = strategycountdb(count)/memorylength
    columnplayerexpectedpayoff = ∅
    for k IN len(columnplayerpayoff) do
        columnplayerexpectedpayoff=columnplayerpayoff(k)*strategycountdb(prob)
    end for
    bestresponsecolumnplayer = max(columnplayerexpectedpayoff)
    bestresponserowplayer = unique(bestresponserowplayer)
    bestresponsecolumnplayer = unique(bestresponsecolumnplayer)
end for
```

The inputs required for algorithm 3 include *memorylength*, *initialhistory*, *timeperiod* and *payoffmatrix*. *Memorylength* is the duration of history in which agents look back to decide the best response. *Initialhistory* is the preceding period choices made by row and column players.

Timeperiod is the duration for which simulations are run. *Payoffmatrix* is the row and column player payoff values. The output of this function is row and column player strategies which are the best responses against the strategies played by their opponents in the preceding periods (*bestresponserowplayer, bestresponsecolumnplayer* respectively)

- The loop starts with time period t.
- An empty set *columnchoices* is created.
    - *columnchoices* is filled with the strategies played by the column player in the preceding periods.
- An empty set *rowchoices* is created.
    - *rowchoices* is filled with the strategies played by row player in the preceding periods.
- An empty set *rowplayerpayoff* is created.
    - *rowplayerpayoff* is filled with row player payoff values against the strategies played by column players (*columnchoices*).
- *strategycountdb* is the table of column player choices and the count of the respective strategy being played.
- A new column is created in *strategycountdb* with name *prob*, which shows the probability or percent of times the column player played that specific strategy in preceding periods.
- An empty set *rowplayerexpectedpayoff* is created.
    - *rowplayerexpectedpayoff* is filled with dot product of row player payoff values (*rowplayerpayoff*) with the probability value (*prob*) derived in the previous step.

- The best response for row player (*bestresponserowplayer*) is the strategy that has maximum expected payoff values (*rowplayerexpectedpayoff*).
- An empty set *columnplayerpayoff* is created.
    - *columnplayerpayoff* is filled with column player payoff values against the strategies played by row players (*rowchoices*).
- *strategycountdb* is reformatted as the table of row player choices and count of the respective strategy being played.
- A new column is created in *strategycountdb* with the name *prob*, which shows the probability or percent of times a row player played that specific strategy in preceding periods.
- An empty set *columnplayerexpectedpayoff* is created.
    - *columnplayerexpectedpayoff* is filled with dot product of column player payoff values (*columnplayerpayoff*) with the probability value (*prob*) derived in the previous step.
- The best response for column player (*bestresponsecolumnplayer*) is the strategy that has maximum expected payoff values (*columnplayerexpectedpayoff*).
- In case multiple strategies are repeated, we take the distinct row player and column player strategies and stored in *bestresponserowplayer* and *bestresponsecolumnplayer*, respectively.

The output of the above algorithm is then used along with algorithm 2 and computations described in Table 3.4 to decide the norm candidate. Below are a few characteristics of agents who may follow this approach.

- The expected payoff approach assumes that agents are rational decision-makers who weigh probabilities and outcomes. This behavior aligns with the fundamental principle of rational choice theory.
- Agents behave in a risk-neutral manner. They do not consider risk aversion or risk-seeking preferences. Expected payoffs provide a balanced view of gains and losses.
- Agents' choices are more deterministic. Maximizing expected payoffs often leads to consistent choices. Agents commit to strategies that yield the highest average expected gains. Commitment helps build reputation and trust.
- Agents become predictable. Consistent choices based on expected payoffs makes the environment stable, this predictability can be advantageous to establish norms and stable patterns.

Below are a few applications of this approach.

- Firm Behavior: Profit-maximizing firms make decisions based on expected returns. They choose production levels, pricing, and investment strategies to maximize long-term profits.
- Household Behavior: Households consider expected utility when making consumption and saving decisions. They allocate resources to maximize overall well-being.
- Investment Decisions: Investors evaluate expected returns and risks when choosing assets. Rational investors diversify portfolios to optimize expected gains.

In the next section, we show simulation results in the context of five different 2*2 games to check which strategies evolve when agents are expected to select responses corresponding to these two approaches in each period.

3.4 Simulation results

The following subsections show results for different games assuming a memory size of 2 and total 10 time periods. We consider the impact of increasing memory size in detail in the following section. To keep the discussion simple and maintain brevity, the results shown below assume agents do not make mistakes while taking actions unless explicitly stated to show any unique findings. It means agents proceed with actions recommended by the two approaches listed above. However, we will explain how the framework can be tweaked to incorporate the possibility of allowing agents to make mistakes during the discussion on using the Python library towards the end of this section. We present the output of simulation results in the form of graphs which are presented in the corresponding sub-sections below. These graphs represent time period on X-axis which we assume is 10. On Y-axis, we show the percentage of times the specific outcome is played during the simulation run (Count %). In some cases, we have shown "Cumulative Count" on the Y-axis, where we have observed that any single outcome is chosen at any given time. This leads to its Count % number to 100 at that time period. Hence, we have shown the cumulative count to show how this single outcome is being increased or decreased or stayed constant throughout the game. This cumulative count starts with 0 for all four action pairs and gets incremented with 1 if that action pair is being played during that period. If a specific action pair is not being played during that period, its count remains constant as was observed in the preceding period.

We assume the time period starts from period 0. The graphs show the game's outcome after initial state choices, which are assumed. So, for example, if the memory length is 2, this implies that at period 0, the agents' choices after considering the past 2-period initial history are shown. We have

not considered simulations beyond 10 time periods as we did not observe any trigger which can change the results observed drastically after 10 periods. In cases where we considered different variants of the same game, we have appended the keyword ‘modified’ in the names of those payoff matrices tables. This is done due to two reasons. First, we want to assess how results change when the payoff ranking differs from the original game as this might impact the results from exhaustive payoff approach. Second, we can assess the impact of payoff values as changing payoff values impacts results from expected payoff approach. For some graphs, results are averaged across different 2-period initial histories. These initial histories are assumed to be 16 in total with 2*2 game and 2-period memory size. These 16 different initial states are as follows:

([0, 0], [0, 0])

([0, 0], [0, 1])

([0, 0], [1, 0])

([0, 0], [1, 1])

([0, 1], [0, 0])

([0, 1], [0, 1])

([0, 1], [1, 0])

([0, 1], [1, 1])

([1, 0], [0, 0])

([1, 0], [0, 1])

([1, 0], [1, 0])

([1, 0], [1, 1])

([1, 1], [0, 0])

([1, 1], [0, 1])

([1, 1], [1, 0])

([1, 1], [1, 1])

The above pair say ([0, 0], [0, 1]) is interpreted as first-period outcome of (0,0), the second period outcome of (0,1) etc. As can be seen, we have considered each pair say (0,0), as the first period outcome and paired it with all four pairs in the payoff matrix as the second-period outcome. And this is repeated for the rest of the pairs (0,1), (1,0) and (1,1) as well, where each pair starts as the first-period outcome, and we pair it with other pairs as second-period outcomes. We report results with some initial state to demonstrate the output.

3.4.1 Prisoner's dilemma

We first consider the prisoner's dilemma game with the following payoff matrix.

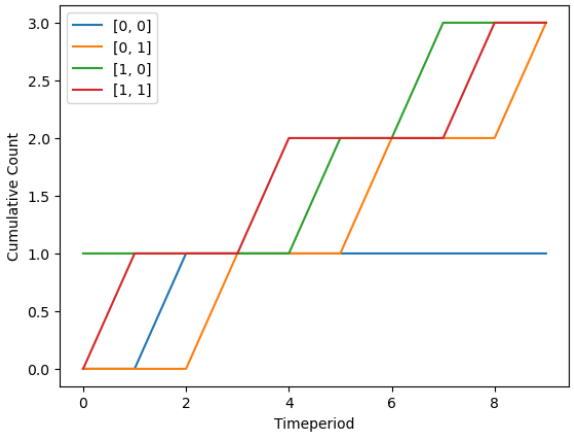
Table 3.5 Prisoner's dilemma

	Cooperate	Defect
Cooperate	2,2	0,3
Defect	3,0	1,1

In the above prisoner's dilemma game, (1) is a strictly dominant strategy for both row and column players; hence it is also the unique Nash equilibrium. It is observed in all the simulation results irrespective of its initial history. Results are the same with both approaches. There are 3 Pareto efficient actions in the game, (0,0), (0,1), and (1,0). Once we modify the response function and

allow agents to make mistakes with a certain probability, these Pareto efficient outcomes can also emerge as the norm, as shown in Young and Foster (1991), Voss (2001), etc. This includes the strictly dominated outcome (0,0) as well. When we assume agents choose non-recommended actions and recommended actions from the exhaustive best response approach with equal probability, we get the following distribution of action pairs as one of the possible outcomes, (0,1): 30%, (0,0): 30%, (1,0): 30% and (1,1): 10% with the initial 2-period history of {(1,1) and (1,1)}. This distribution is achieved with only one outcome observed out of four during any of the given time period. Figure 3.1 demonstrates this result which shows the time period on the X-axis and the cumulative count on the Y axis. Since at any given time, agents chose only one outcome out of four, hence the count of that outcome stayed constant for future time periods if it was not played again.

Figure 3.1. Prisoner's dilemma convergence with exhaustive best response approach (including randomness) and initial history of {(1,1), (1,1)}



Therefore, with enough randomness in agents' decisions, we can expect Pareto-dominated outcomes also emerge as the norm.

3.4.2 Battle of sexes

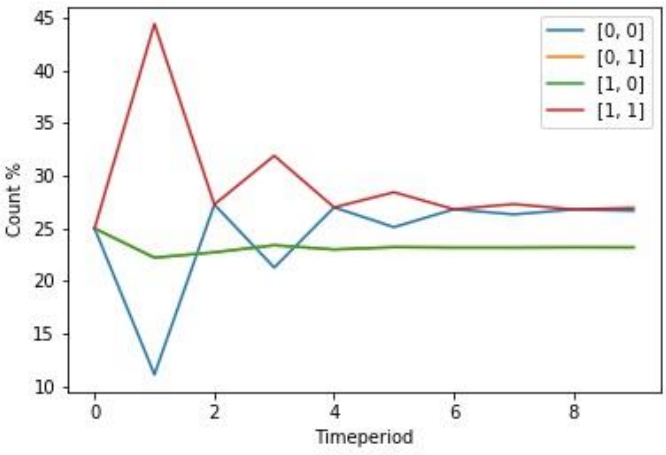
We start with the following battle of sexes game.

Table 3.6. Battle of sexes

	Prize Fight	Ballet
Prize Fight	2,1	0,0
Ballet	0,0	1,2

This game has 2 pure strategy Nash equilibria (0,0) and (1,1). It also has one mixed strategy Nash equilibrium ((0.67,0.33), (0.33,0.67)). None of the strategies is weakly or strictly dominant. Results using the exhaustive best response approach show that all four possibilities have similar chances of emerging as the norm. No significant difference exists in the percentage of times a particular strategy combination is being played. This shows the possibility of non-Nash equilibrium strategy combinations emerging as the norm. However, when the initial 2-period history is {(0,0), (0,0)} or {(1,1), (1,1)}, in that case, the outcome is (0,0) and (1,1), respectively. Figure 3.2 below show one of the possible convergence trend of this game with initial history of {(0,0), (1,1)}.

Figure 3.2. Battle of sexes (Table 3.6) convergence with exhaustive best response approach and initial history of {(0,0), (1,1)}



Trends are different when we look at results from the expected payoff approach. When the initial 2-period history contains both Nash equilibria {(0,0), (1,1)} or {(1,1), (0,0)}, the recent outcome emerges as the most frequent strategy, (1,1) and (0,0) respectively. In the case of the initial 2-period history of {(0,0), (1,0)}, (0,1) emerges as the most frequent strategy (70%) followed by (1,0) (30%). Both strategies are non-Nash equilibrium strategies. The payoffs from both the outcomes, (0,1) and (1,0), are lower than those from the rest of the strategies and hence are inefficient.

We then modified the original game to evaluate the impact of scaled payoff values, as shown in Table 3.7 below.

Table 3.7 Battle of sexes (modified 1)

	Prize Fight	Ballet
Prize Fight	2,1	0,0
Ballet	0,0	2,4

The above payoff matrix retains the original ranking of different possible strategy combinations. Results using the expected payoff approach show that the (1,1) outcome is achieved irrespective of the initial history, except when (0,0) is played in both the initial periods. On the other hand, results are the same when using the first approach because it does not consider the actual payoff values. It just considers the relative ranking of payoffs which is still the same across all four cells in the matrix compared with the original game of Table 3.8. Next, we check how these results vary when we make any strategy weakly dominant for row or column players with the below payoff matrix (Table 3.8).

Table 3.8 Battle of sexes (modified 2)

	Prize Fight	Ballet
Prize Fight	2,1	0,0
Ballet	0,0	0,2

The above payoff matrix still has two pure strategy Nash equilibria (0,0) and (1,1), as in the original battle of sexes game (Table 3.6), except for the row player, strategy 0 becomes weakly dominant strategy. Results from the exhaustive best response approach show that there is not much difference in terms of how frequently some action pairs are being played. (0,0) is played relatively more frequently, and (1,1) is played less frequently. The percentage distribution is as follows: (0,0)

comes to approx. 35% of the time, followed by (0,1) approx. 26% of times, (1,0) approx. 22% and (1,1) 20% of the time. (Figure 3.4). The corresponding percent distribution from the expected payoff approach is (0,0) comes to 67%, followed by (0,1) - 32%, (1,1) - 29%, and (1,0) - 11% (Figure 3.5).

Figure 3.4 Battle of sexes (Table 3.8) convergence with exhaustive best response approach. Results are averaged across different initial 2 period histories.

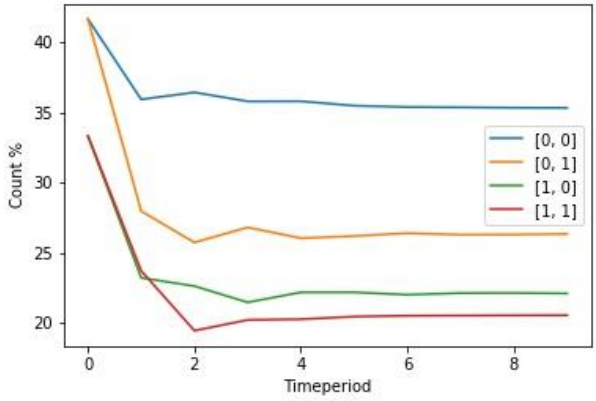
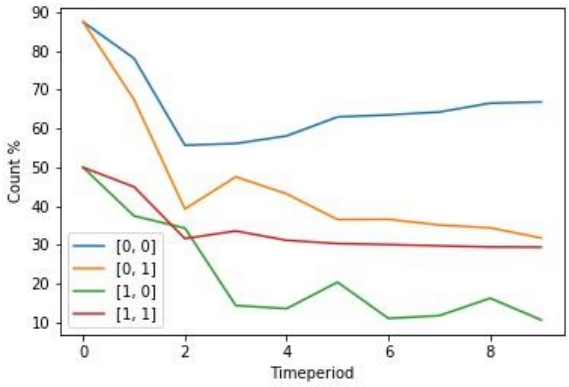
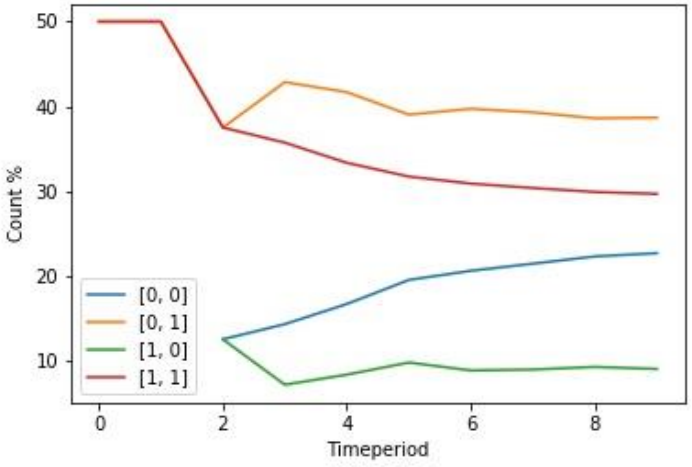


Figure 3.5 Battle of sexes (Table 3.8) convergence with expected payoff approach. Results are averaged across different initial 2 period histories.



Results from both approaches show the preference for weakly dominant strategy for row player (strategy 0). One iteration result from the payoff approach also shows that if the initial history contains pure strategy Nash equilibria strategies {(1,1), (1,1)}, it can result in non-Nash equilibrium strategies being played relatively more frequently. It is shown in Figure 3.6 where (0,1) comes approx. 39% of times compared to any pure strategy Nash equilibrium strategies (1,1): 30% and (0,0): 23%. For instance, this phenomenon can be explained by the generational gap between parents and their kids. In certain societies, kids have been expected to care for their old-age parents. This is the norm that parents and their parents' generations have established for ages. But people born in the 21st century are seen to be not following that norm very judiciously.

Figure 3.6 Battle of sexes (Table 3.8) convergence with expected payoff approach and initial history of {(1,1), (1,1)}



3.4.3 Matching pennies

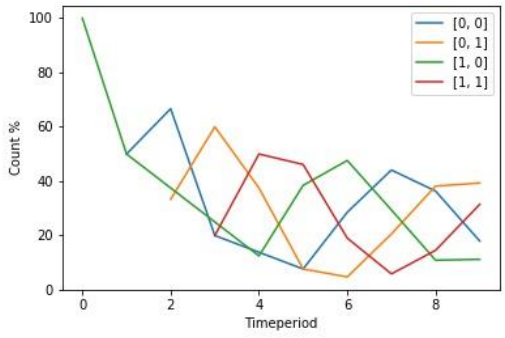
The payoff matrix of the game is defined in Table 3.9.

Table 3.9. Matching pennies

	Heads	Tails
Heads	1,-1	-1,1
Tails	-1,1	1,-1

In this game, there is no weakly or strictly dominant strategy. There is no pure strategy Nash equilibrium. It has one mixed strategy Nash equilibrium, ((0.5,0.5), (0.5,0.5)). Results do not show any clear pattern. A zigzag pattern is observed over time. All the strategies have similar chances of emergence as we go further down towards the end of the simulation period. Results are the same using both approaches (Figure 3.7). Matching pennies is an application of a zero-sum game, and it applies to any scenario where losses from one player are gains for the other player. Examples include games like poker, chess, tennis etc.

Figure 3.7. Matching pennies (Table 3.9) convergence with exhaustive best response approach and initial history of {(1,1), (1,1)}



3.4.4 Stag hunt

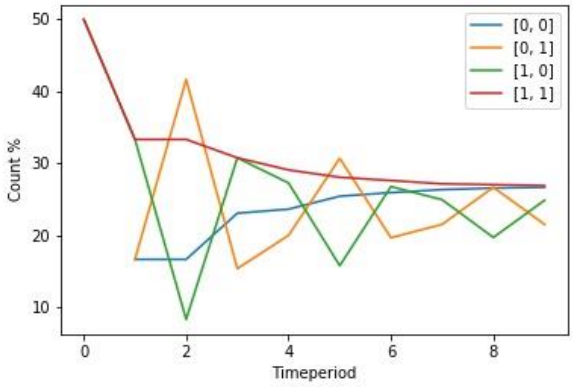
Stag hunt game is defined with payoff matrix as per Table 3.10.

Table 3.10. Stag hunt

	Stag	Hare
Stag	2,2	0,1
Hare	1,0	1,1

This game has two pure strategy Nash equilibria, (0,0) and (1,1). It also has one mixed strategy Nash equilibrium ((0.5,0.5), (0.5,0.5)). None of the strategies is weakly or strictly dominant for either of the players. When the initial 2 period history is {(0,0) and (0,0)}, then results also converge to (0,0). Similar is the case when the initial 2 period history is {(1,1) and (1,1)}. All other cases when the initial 2 period history is not {(0,0), (0,0)} or {(1,1), (1,1)} results do not converge to any specific strategy. All strategies have similar chances of emergence by the end of 10 period simulations. Results are similar using both approaches (Figure 3.8).

Figure 3.8. Stag hunt (Table 3.10) convergence with expected payoff approach and initial history of {(1,1), (0,1)}



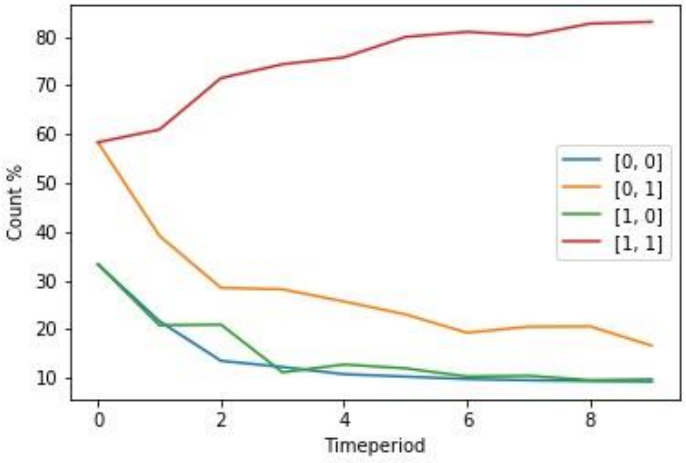
The above payoff matrix is modified as per below (Table 3.11) when we make strategy 1 as weakly dominant for column player.

Table 3.11 Stag hunt (modified 1)

	Stag	Hare
Stag	2,2	0,2
Hare	1,0	1,1

With this payoff matrix, the strategy outcome (1,1) is the most frequent, and (0,0) is the least frequent. Results from the exhaustive best response function did not give much conclusive evidence of any specific strategy emerging as the norm. Results from the expected payoff approach are more conclusive, with on average 80% of the time (1,1) emerging as the most frequent outcome and (0,0) coming on average 10% of the time. The next highest strategy is (0,1) in approx—17% of the times (Figure 3.9).

Figure 3.9 Stag hunt (Table 3.11) convergence with expected payoff approach. Results are averaged across all initial histories.



We also investigate results when strategy 1 is being made weakly dominant for row player instead of column player (Table 3.12)

Table 3.12 Stag hunt (modified 2)

	Stag	Hare
Stag	2,2	0,1
Hare	2,0	1,1

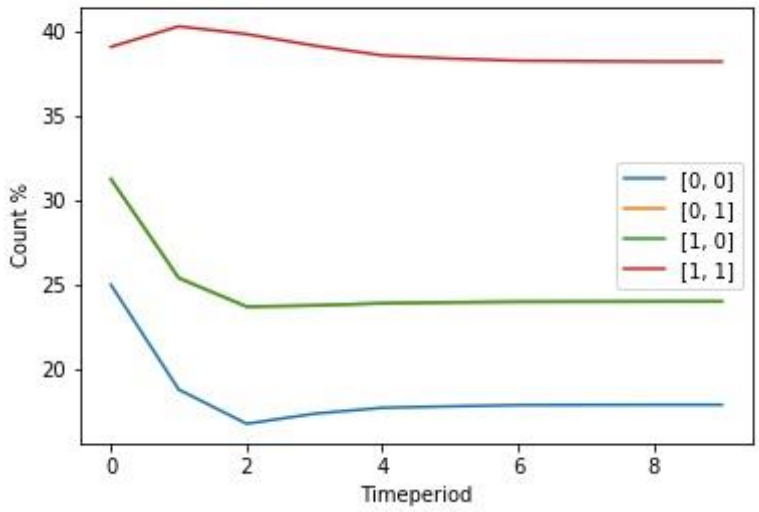
Results with this payoff matrix are similar to the ones achieved from the previous payoff matrix (Table 3.11), except that the second highest outcome is (1,0) now as compared to (0,1) earlier. We also evaluate how results would change when both row and column players have strategy 1 as the weakly dominant strategy as per Table 3.13

Table 3.13 Stag hunt (modified 3)

	Stag	Hare
Stag	2,2	0,2
Hare	2,0	1,1

According to the first approach, we get the following outcome distribution: (1,1): 34%, (0,0): 18%, (0,1) and (1,0): 24 % each except for the case when the initial strategy is {(1,1), (1,1)} (Figure 3.10). In the second approach, (1,1) is the most frequent outcome, with 99% occurrence.

Figure 3.10 Stag hunt (Table 3.13) convergence with exhaustive best response approach. Results are averaged across all initial histories.



An alternative version of the original stag hunt payoff matrix (Table 3.11) is tried as per the below revised matrix (Table 3.14)

Table 3.14 Stag hunt (modified 4)

	Stag	Hare
Stag	3,3	0,2
Hare	2,0	1,1

We got similar results with this payoff matrix if compared with results using the payoff matrix in Table 3.10. Results are the same when we modify the above matrix to make a particular strategy weakly dominant for row or column players compared with the corresponding changes made in the original payoff matrix of Table 3.10. Therefore, these results point towards every possibility of inefficient outcomes, given that the game has only one Pareto efficient outcome (0,0).

3.4.5 Coordination

We start with the payoff matrix for the coordination game (Table 3.15).

Table 3.15 Coordination game

	Left	Right
Left	2,2	0,0
Left	0,0	1,1

There are two pure strategy Nash equilibria (0,0) and (1,1). The mixed strategy Nash equilibrium of the game is ((0.33,0.67), (0.33,0.67)). None of the strategies is weakly or strictly dominant. Results are more conclusive from the expected payoff approach where (0,0) emerges as the

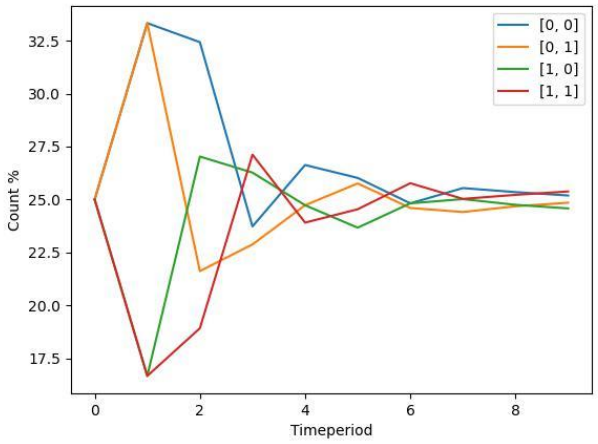
outcome except for the case when the initial history is {(1,1), (1,1)} where (1,1) emerges as the outcome.

A typical example of a coordination game is choosing which side of the road to drive. If drivers on both sides of the road follow the same rule (either drive towards the left side or the right), this reduces the chances of collision and is beneficial for both. However, in certain sections of societies where law enforcement is not stringent, particularly in rural areas, driving rules are usually not followed judiciously. The alternative of driving towards the left or the right is a risky choice for the agents but still has the potential to emerge as the local norm and be ingrained as the conventional way of driving.

3.5 Impact of memory size

In this section, we take Battle of Sexes game as an example, and increase the memory size. We assume battle of sexes game with payoffs as defined in Table 3.6. We first start with results from the exhaustive payoff approach. Below graph shows one of the results using memory size 3.

Figure 3.11 Exhaustive best response approach using memory size of 3 with initial state as ([1, 1], [1, 0], [0, 0])



The following figure shows one of the results using memory size of 4.

Figure 3.12 Exhaustive best response approach using memory size of 4 with initial state as ([0, 1], [1, 1], [0, 0], [1, 1])

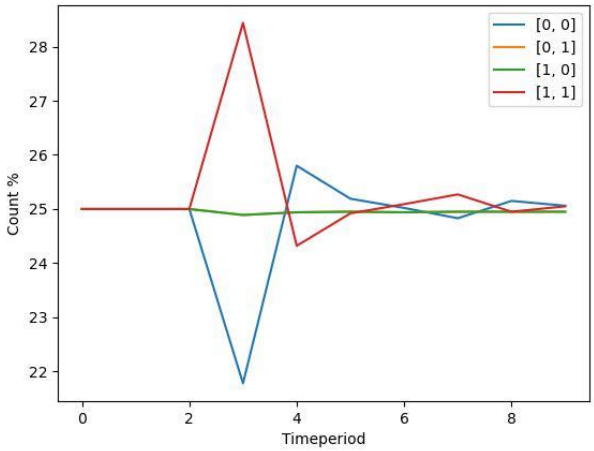
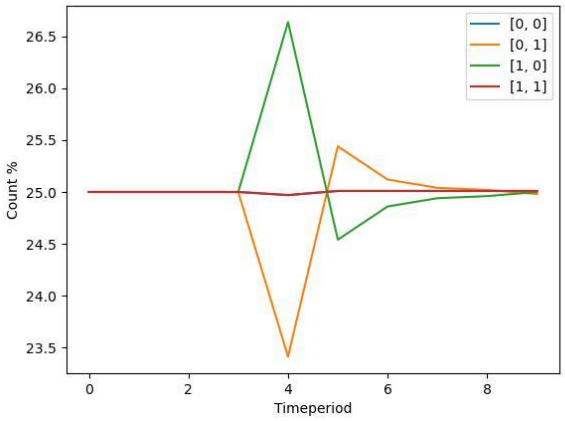


Figure 3.13 shows one of the results using memory size of 5 with initial state as initial state as ([0, 1], [0, 1], [0, 1], [1, 0], [0, 1]).

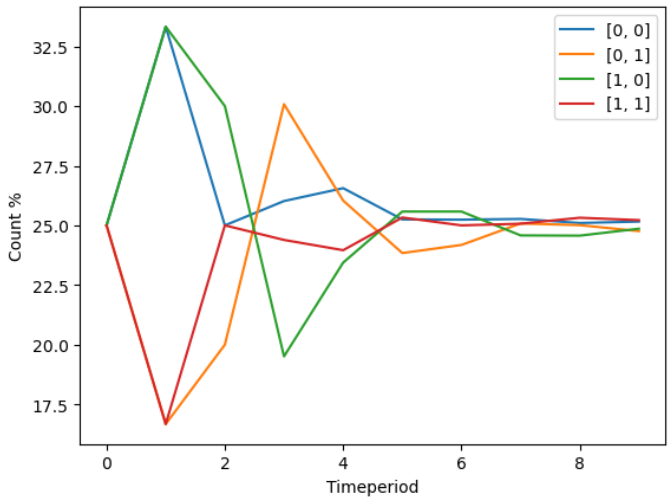
Figure 3.13 Exhaustive best response approach using memory size of 5 with initial state as ([0, 1], [0, 1], [0, 1], [1, 0], [0, 1])



With increased memory size, we have seen that the results do not change much. They all converge towards 25% for each of the action pair. Since there are only 2 choices possible for both the agents in a 2*2 game, when we ran the iterations with higher memory length, we did not see much difference in convergence results since exhaustive approach by design considers all potential alternatives at each time period. Hence, we observed a convergence towards 25% for all the pairs in majority of times if memory length is higher than 1 and the game is played for sufficient time periods. The time period threshold at which action pairs start convergence towards 25% is also shifting with an increase in memory length.

We also tried introducing randomness in agent's decision making by assuming agents take decisions randomly with 5% probability. Below figure shows one such result with memory size of 3.

Figure 3.14 Exhaustive best response approach using memory size of 3 with initial state as ([1, 0], [0, 0], [0, 1]) and randomness probability 5%

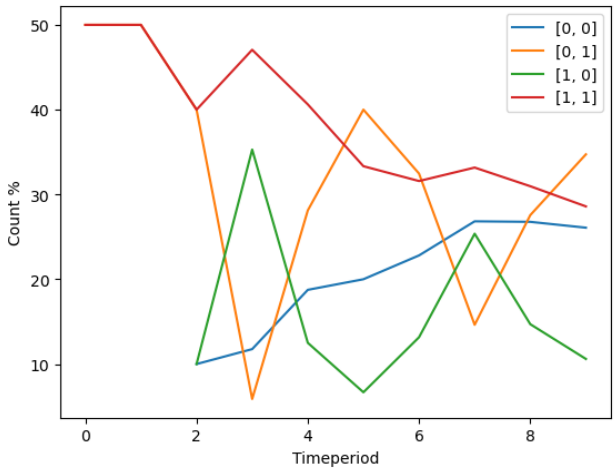


By including randomness in agents' decision making, we can see variations in agents' choices during the early time periods. Similar trend is observed for higher memory lengths also. As in the case of zero randomness, the time period when the convergence takes place also shifts with higher memory lengths. Therefore, we did not see much impact of randomness in the long term when the game is played for sufficiently longer duration. We also assessed the possibility when randomness is reduced during the game after 5 periods, we did not observe much difference in outcomes. We observed some volatility during certain periods, but the overall trend remains the same, which is convergence towards 25% share.

On the other hand, in the expected payoff approach, with memory lengths higher than 1, we can see a distribution that may not necessarily converge towards 25% for all the choices. In some cases, only a few action pairs are chosen, which can increase this percentage to higher than 25% for these

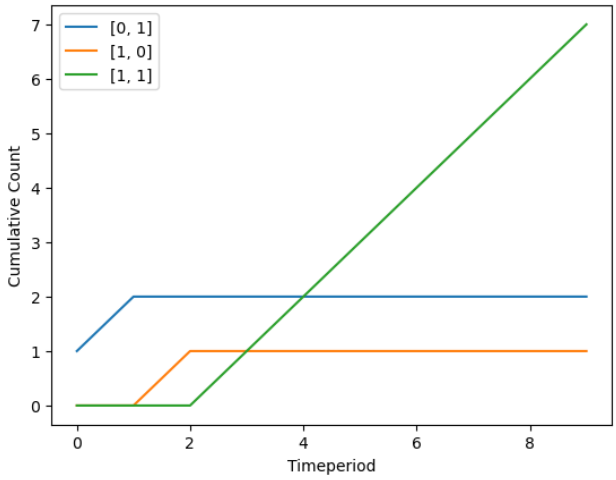
pairs over a period. Below figure shows one of the results from expected payoff approach with the initial state of ([1, 1], [1, 1], [1, 0]) and memory size of 3.

Figure 3.15 Expected payoff approach using memory size of 3 with initial state as ([1, 1], [1, 1], [1, 0])



In the above figure, the share of (1,0) action pair is reduced over time. Below figure shows one of the results from memory size 4. When the memory length is increased to 4, the convergence towards one outcome is observed in higher % of cases as compared to memory length of 3. Also, there is more preference towards single outcome with memory length of 4 at any given time period.

Figure 3.16 Expected payoff approach using memory size of 4 with initial state as ([1, 0], [1, 0], [0, 0], [1, 1])



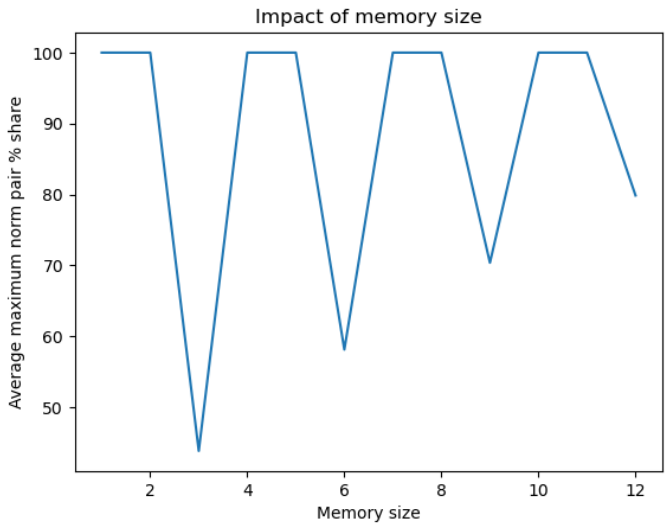
Above figure shows (1,1) outcome is being played more by the end of simulation period. Also, in any of the given time period, a single action pair is being played. This is different from the results using memory length 3, when we had multiple action pairs at any given time period.

To further assess the impact of memory length, we ran 100 iterations using randomly initiated initial state for each of the memory lengths. We considered memory length from 1 to 12. For memory lengths up to 10, we ran iterations till 10 time periods. For memory lengths 11 and 12, we considered the time period as 11 and 12 respectively. This is performed using the expected payoff response function.

For each memory size, we ran 100 iterations with different random seeds and initial states. After running 100 iterations, we checked the output at the end of the simulation period for each of these iterations and assessed which of the action pairs are being played. We extracted the action pair which is being observed the most by at the end of simulation period for each iteration. We took

the average of the frequency value of the most played action pair and plots this average value for each memory size (Figure 3.17). We repeated this exercise for memory lengths from 1 to 12.

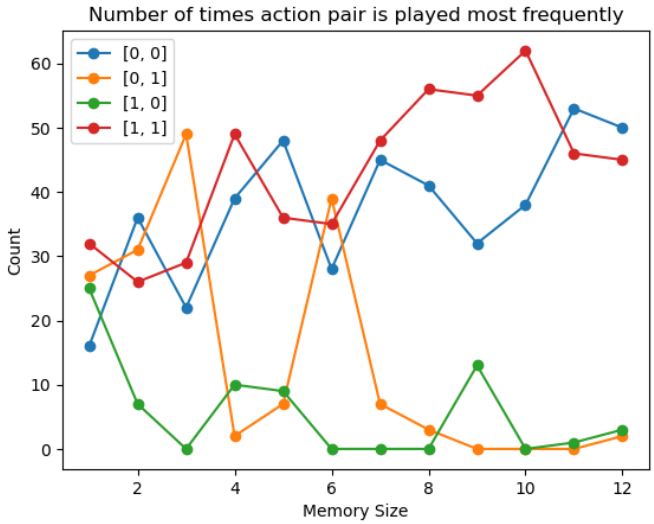
Figure 3.17 Average maximum norm pair % share at the end of simulation



The above graph shows that the maximum % share of norm pair for a given memory size is either at the maximum 100 or increasing with the memory size. This is irrespective of which pair was being chosen as a norm pair. We have seen that at the memory size of 3 or a multiple of 3, there is more than 1 choice agent playing in the equilibrium. For memory lengths of 3 or a multiple of 3, the threshold for average maximum % share increases with memory size.

To further explore, we have considered which action pairs are being chosen more frequently at each of the memory lengths. Below figure is the count of action pairs from 100 iterations for each memory length.

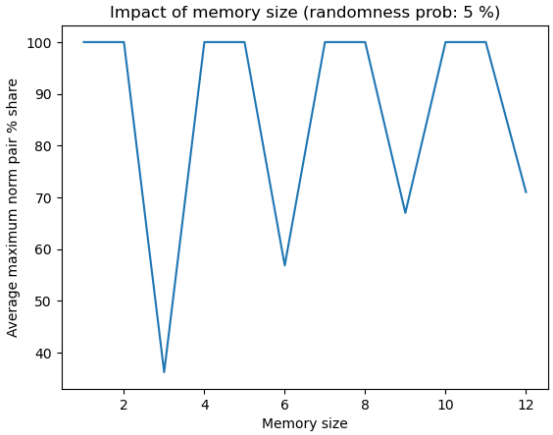
Figure 3.18 Action pairs count over 100 iterations for memory length 1 to 12.



The above graph shows how different action pairs are being played when memory size is increased. This is the count of different action pairs out of 100 iterations. For each memory size, we looked at the pairs played most frequently during that iteration. We then calculated the count of these pairs for each of the 100 iterations across each of the memory size. We can see that action pairs [0,0] and [1,1] contribution increased with higher memory size. At a lower memory size, [0,1] and [1,0] pairs are also being frequently played but this reduces with higher memory size.

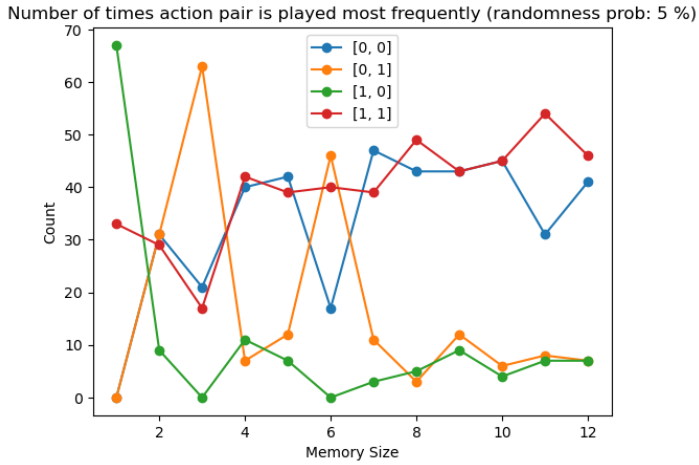
We then repeated the same exercise assuming agents take actions with some randomness. We assume the randomness probability to be 5%. Below figure shows the impact of memory size with 5% randomness.

Figure 3.19 Average maximum norm pair % share at the end of simulation (with randomness of 5%)



The impact of randomness is felt more when the memory size is lower. The average maximum norm % share at the end of simulation period is relatively lower in the presence of randomness. However, the impact is reduced at higher memory size. Below figure shows the individual action pairs count for each memory length in the presence of randomness.

Figure 3.20 Action pairs count over 100 iterations for memory length 1 to 12 (with randomness of 5%)



The trend for individual action pairs is similar to when randomness is not present. The variations are high in case of lower memory lengths but are averaged out as memory length increases.

To further analyze the reasons behind the patterns with memory lengths of 3 or multiple of 3, we have considered the % of times row and column players play strategy 0 or 1 in 100 iterations for each of the memory lengths. This is also attributable to the mixed strategy Nash equilibrium where each agent expects the opponent agent to play the higher payoff agent's strategy with at least 33% probability. This translates to row agent playing 0 when they expect column agent to play 0 with at least 33% probability, and column agent playing 1 when they expect row agent to play 1 with at least 33% probability. Figures 3.21 and 3.22 demonstrate these results for row and column agents playing 0 and 1 strategy respectively.

Figure 3.21 Row and column agents' 0 strategy % share across 100 iterations

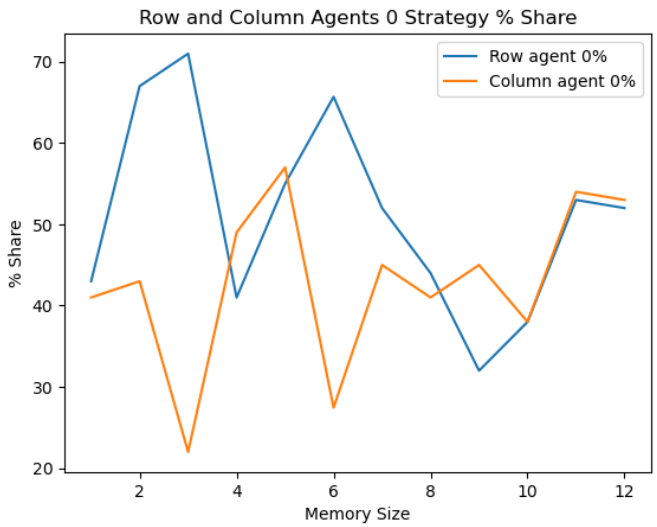
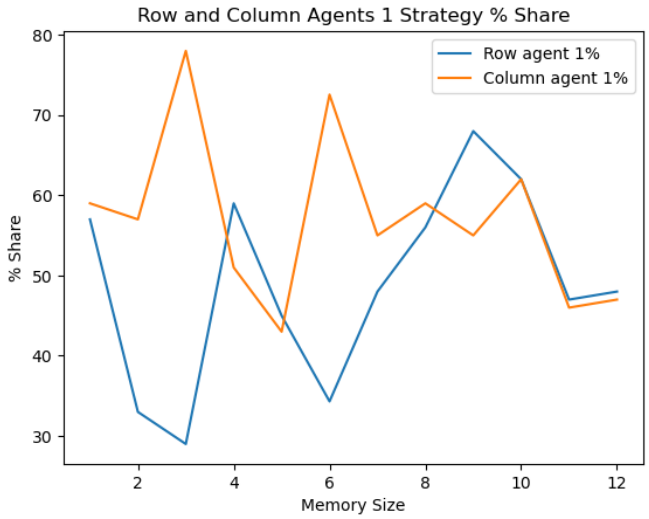


Figure 3.22 Row and column agents' 1 strategy % share across 100 iterations



Above graphs show the % of times row player and column player played 0 and 1 strategy. It shows that row player playing 0 strategy more is preceded by column player playing 0 strategy in previous periods. Similarly, column player playing strategy 1 is preceded by row player playing 1 strategy in previous periods. As memory length increases, there is a convergence towards row and column players playing equal % of times 0 and 1 strategy. This is reflected in the above graphs where there is convergence towards (0,0) and (1,1) strategy. We repeated the same analysis when agents have a 5% probability of making decisions randomly. Below Figures 3.23 and 3.24 show these trends.

Figure 3.23 Row and column agents' 0 strategy % share across 100 iterations (randomness 5%)

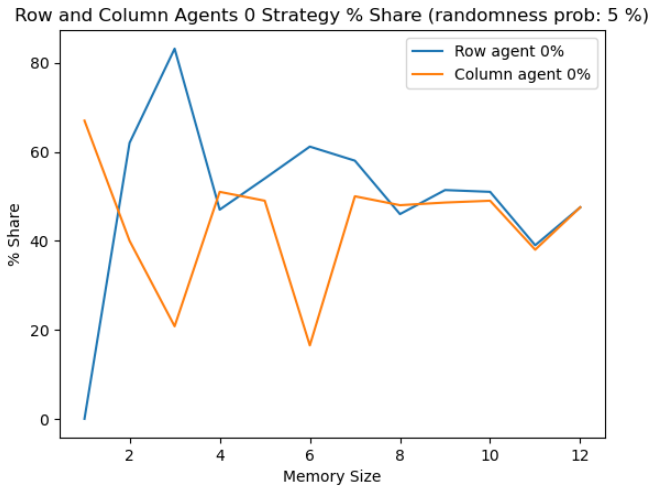
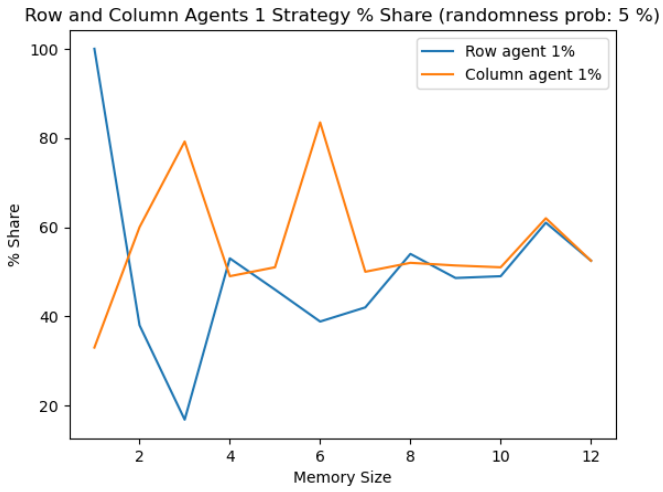


Figure 3.24 Row and column agents' 1 strategy % share across 100 iterations (randomness 5%)



These graphs represent similar results with the difference of more variation in agents' choices with lower memory size. These results with respect to longer memory lengths are in alignment with Block et al. (2019) where its shown larger memory size results in convergence towards Nash equilibrium and mixed strategy equilibria is more favored compared to pure strategy Nash equilibrium.

To summarize, when the memory length is 1, the exhaustive approach gives direction on any specific action pair. When the memory length is increased to 2 or more, and the game is being played for longer, this results in convergence towards a nearly equal share of different action pairs in case of exhaustive payoff approach. On the other hand, the expected payoff approach shows the possibility of different results depending on payoff values. With higher memory length, we can expect convergence towards fewer choices, where some action pairs are being shown higher frequency than others. Randomness facilitates convergence or coordination towards a certain action pair. The following section discusses some of the applications where this computational framework can be used.

3.6 Applications

The framework proposed is helpful in scenarios where we want to measure the path to reach a particular outcome and assess which agents' choices in transit led to those choices. This can also answer how long it takes to reach a particular outcome. For example, we can try different payoff values of the game along with different values of memory length and randomness parameters to answer these questions. We have shown this with the help of two response functions. This also has the potential to explain some results which are surprising like the 'cooperate' outcome in a prisoner's dilemma game, or 'stag' outcome in a stag hunt game etc (Harms & Skyrms, 2008; Hofbauer & Sandholm, 2011; Alexander, 2007; Alexander, 2021).

One example is when a society tries to decide which parenting style couples should adopt. For example, there is a below game with two strategies, Strict or Lenient.

Table 3.16 Parenting style game

	Strict	Lenient
Strict	5,5	1,4
Lenient	4,1	3,3

If both parents adopt a strict parenting style, it is assumed that it leads to more disciplined kids, and hence, both parents enjoy higher payoffs. The opposite is when both parents are lenient; both would get a lower payoff. However, if one is strict and another is lenient, this would lead to a better payoff for lenient but less for strict.

We can use the framework proposed to answer the question of which parenting style is expected to prevail in society. If the government wants to see a more disciplined environment in society, it can study the path dependency of this outcome and intervene in the society. This intervention could provide additional incentives to adopt strict behaviour towards children or create an environment where disciplined behaviour is rewarded, which can help reduce randomness in agents' decision-making. Some examples could include giving scholarships to students who demonstrate disciplined behaviour and perform well in school exams and extracurricular activities. Another intervention could be directed towards employers where the government can enforce following strict work timings which indirectly can make the parents conform to more disciplined behaviour. Another example where it could have applications include labour unions. We can assume there are two unions, A and B, in an organization. If both unions join hands, that will increase the efficacy of the union, and both would be able to create better working conditions for their workers and put pressure on the management to increase their wages. Workers have two choices: to cooperate

(demand better wages and working conditions) or to defect (do not make any additional demands). There is a risk involved that workers who demand higher wages may face retaliation from the employer in the form of their job elimination. We can have the following payoff matrix with two choices.

Table 3.17 Labour unions game

	Union B cooperates	Union B defects
Union A cooperates	2,2	-1, 3
Union A defects	3, -1	0,0

We can see that when some workers demand additional wages and others do not, they risk losing their jobs and hence face a negative payoff of –1. Moreover, the ones who do not demand they enjoy a higher loyalty bonus from the employer, resulting in a higher payoff of 3. When unions A and B demand nothing, there is no additional payout, and they remain at the 0 payoff. If both can trust each other and cooperate, that results in a better outcome for both sides of workers with the payoff of 2.

We can ascertain the path dependency of this game by leveraging the framework provided in this paper. This helps formulate the paths if one side defects, how would it change the course of action. This can help employers simulate scenarios where they expect to see an outcome of unions cooperating and accordingly adjust loyalty bonuses. It can help formulate better policies in the organization where management proactively contacts workers and discusses their concerns. This might help them save from the eventual breakdown when unions ask for their demands and go on

strike until their demands are met. From the Government's standpoint, it helps them formulate better labour laws in the country. They can simulate different scenarios where they expect convergence toward societal defect outcomes. The Government can formulate laws or policies that encourage workers to participate in unions, which can eventually prevent society from getting stuck in defect outcomes. A good governance and laws-abiding nation empower workers to have less fear in raising their concerns with their employers and enables them to have better working conditions.

3.7 A note on the python library created.

We have created the *game-simulator* Python library to replicate the results for any payoff matrix. There are two main functions in the library, *simulation_function* and *simulation_function_payoff*. The first function generates results using the exhaustive best response approach, while the second function creates results using the expected payoff approach. Results discussed above in the previous section are derived using these two functions. The other two functions, *simulation_function_random*, and *simulation_function_payoff_random*, are replicas of these functions, allowing agents to make mistakes and choose actions randomly.

In these functions, the parameter *random_multiplier* allows users to specify how recommended actions from the abovementioned approaches will be weighed against the rest of the choices. The parameter value of 1 indicates equal weightage to recommended and non-recommended actions. A value higher than 1 implies more weightage to recommended actions than non-recommended ones, while a value lower than 1 implies the opposite. For instance, there are 2 strategies, A and

B, for an agent to choose from, and A is the recommended action. A *random_multiplier* value of 5 implies A weights 5 while B is weighted 1. Agents choose one action randomly from these actions, A and B, based on the relative weights of recommended and non-recommended actions. Actions with higher weights are more likely to be selected compared to lower weights.

These functions produce output in different Excel files, which contain information about the strategy outcome played each time during the simulation window. These Excel files can further be used for analysis and producing different graphs. Additional details about installing the library, the parameters required, and output interpretation are provided in the library homepage's description section (https://pypi.org/project/game-simulator/).

We are aware of few libraries which already exist like ‘DyPy’ or ‘egttools’ in Python, ‘EvolutionaryGames’ in R or a full-fledged software like ‘NetLogo’. The majority of these libraries approach evolution from a macro perspective meaning leveraging replicator dynamics and its variants and focuses more on evolutionary dynamics rather than individual agent level interactions. The Python library built focuses on individual agent level decisions and how it changes as agents interact over time. The library built requires minimal knowledge of the language and its output is generated in the form of Excel files which are easy to understand and widely appliable and used across a larger user base. It is reusable and doesn’t require writing code from scratch. Writing code from scratch may take more time and being the open-source nature of Python language, the library code can be customized to solve specific problems if required. Although we acknowledge that 'Stata’ commands are generally used in economics literature, it has a limited user base and is generally used for econometric analysis. We prefer to do this in Python, because of its

open-source nature and a large user base. Also, there is limited functionality to perform agent-based modeling in ‘Stata’ language as compared to Python.

3.8 Conclusion

The literature on the emergence of social norms shows that the prevalent norms at any given time need not necessarily be Pareto efficient or satisfy a Nash equilibrium. We have shown this with the help of simulations on a few standard 2*2 normal-form games, which can be extended to any finite normal-form game. We have used two response functions for agents to decide what action to take. These two response functions are used in the literature, and we have integrated these into a reusable framework that can increase its applicability and usability. We provide a computational framework which shows that any strategy or action has the potential to emerge as a norm. It is driven by the dynamics followed and not necessarily due to that strategy being superior to others. A higher memory length leads to convergence towards actions pairs which are more in alignment with pure and mixed strategies Nash equilibria. We believe the Python library created will be useful to researchers interested in the evolutionary aspects of finite normal-form games and has vast applications in many areas of social science and beyond.

Chapter 4: Social Networks and Norms Evolution

4.1 Introduction

In the second chapter on literature review, we have seen that the existing literature has analyzed norms evolution using deterministic or analytical approaches and computational or simulation approaches. The literature shows norm evolution is dependent upon various parameters like payoff structure of the game, population size, memory length, agent's network and its strength, time taken to reach a norm, methodology which agents follow to update their actions during each period of the game, and randomness in agents' actions (Young & Foster, 1991; Kandori et al., 1993; Young, 1993; Young, 2015; Alexander, 2007). This chapter focuses on norm evolution using simulation-based methods considering agents' social networks.

We have attempted to solve norm evolution from the ground up, where agents interact with other agents in their neighbourhood over a period and decide what is best for them. We defined the social network of agents who interact with other agents in their neighbourhood over time. These networks represent agent relationships in the form of nodes and edges where each agent is placed in a single node and is connected with other agents placed in different nodes through edges. These networks could be of different types. In particular, we consider ring, small-world, complete, and 2-dimensional networks. To establish the definition of norm, we have considered two dimensions, the number of agents following the norm at any given time in the population and the longevity represented by how long agents follow the norm. A strategy played by a larger number of agents and longer periods is a potential candidate for the norm.

We have explained this with the help of two games, the Naming game (Young, 2015) and the Nash demand game (Axtell et al., 1999). In the Naming game, two agents are selected randomly and shown a picture of a face. Agents do not know the identity of other agents, and they independently suggest a name for the face. If agents suggest the same name, they get a positive reward; otherwise, they get a negative reward. At the end of each iteration, agents get to know the names proposed by opponent agents, and this keeps them updated with the names currently popular at any given time. There is no constraint on the names agents can propose, and it is left to their imagination. Agents use a perturbed response function to decide what names to propose, implying agents propose names randomly with certain probability e and propose the most frequent names with probability 1-e. 'e' here can be interpreted as the probability of committing an error by the agents. A fixed population size is defined along with the agents' neighbourhood size. No constraint exists on the number of unique names that can evolve after the simulation.

Results show the majority of the population propose 1 or 2 names in the case of a ring network with at least 1 name satisfying the norm criteria. In the case of small-world networks where there are *shortcuts* available, norms that emerge are not necessarily locally concentrated. This implies agents following norms proposed by those agents where there is no direct linkage among agents exist. The complete network results show convergence towards one name. Results vary when we incorporate different agent types and assume few agents as fixed. Fixed agents continue to use fixed strategy (propose the same name) every time, irrespective of what other agents propose. In general, results show fewer names satisfying the criteria of the norm due to the larger number of unique names proposed compared to the case when no fixed agent exists. To generalize the results, we created a Python library, *multi-agent-coordination* , which can be used to get results with any

custom user-defined parameters on agents' social network and its associated parameters. The output generated from the library include information on strategies proposed by agents at different points in time, fixed agent distribution, how quickly agents reach the norm, and how agents' strategy choices influenced other agents' choices connected via the network. We considered naming game as an example to demonstrate how agents' social networks and agent types impact the norm evolution when agents interact and decide what action to take. In general, we can extend this analogy to any coordination game scenario where we expect agents to receive a positive reward if they follow the same strategy and a zero or negative reward if they follow a different strategy. There could be multiple applications of naming game. For example, this could explain the regional differences in career paths in different societies. In certain regions, engineering education is preferred, while management education might be given more precedence in others. One can think of a positive reward if agents follow the same career path their network has followed, while a negative reward if agents follow different paths. A reward could be interpreted in terms of reputation, not being isolated, helping each other out if others follow a similar career path, etc. Another example could be learning a new language if agents migrate from their home country to a new foreign country where the language of communication is different. It is in the agents' interest to communicate in the language of communication prevalent in foreign countries to build social connections in the new place.

In the second game, we considered the Nash demand game. The game starts with specifying agents' distribution in terms of their strategies. Agents play the Nash demand game of Axtell et al. (1999), where there is a fixed pie (say 100 dollars), and agents are expected to distribute that among themselves. Agents can make 3 demands, namely High (H) with a payoff of 70, Medium(M) with

a payoff of 50, and Low (L) with a payoff of 30. These 3 choices are agents' options or strategies which they can choose during the game. At any given point, two agents are selected, and they make their demands. The rule of the game is that agents get these payoffs only if the demands made by both the agents playing the game are lower than or equal to 100. If the total demanded payoffs exceed 100, none of the agents receives anything. Therefore, if both the agents demand H or one of them demands H and another M, both would receive a zero payoff. Agents' initial distribution of strategies is defined before starting the game, implying how many % of agents follow H, M, and L. This distribution changes as agents play the game repeatedly over a period and change their strategies. Agents choose their neighbour's strategy if the payoff from playing that strategy is higher than the payoff from following their current strategy.

Results show convergence towards M outcome across most network structures when we assume the initial state of agents is distributed in the ratio of 40/40/20 (H/M/L). This approach is open-sourced in the form of a Python library named *multi-agent-decision*. The library can get results with any user-defined parameters on the number of strategies, agents' distribution, population size, agents' neighbourhood size etc. The output generated from the library includes information on strategy frequency distribution during each period of the game. This library can be used to answer questions on the impact of agents' neighborhood on their choices. For example, agents in specific regions prefer engineering or medical education because agents in the neighborhood prefer similar education. Agents need not necessarily be related to physical geographical distance. An example is a small world network, where the young or teenage population in developing economies follow western world trends due to the relatively cheaper internet availability along with various social media channels and avenues. We can analyze these networks' impact on the decisions that agents

in developing economies make with respect to their educational or professional careers and how it influences the payoffs corresponding to different choices. Another example could be that agents' style and way of talking or greeting other agents is different when they are in their domestic country than when they are in a foreign country. Some agents may prefer to talk in the native language of the foreign country where they are, while some prefer to talk in an almost universally acceptable language like English. The network determines this decision and corresponding payoffs they are a part of at different points in time and their past interactions. We can make use of the multi-agent-decision library to answer these questions.

To extend this further, we investigated how the norms that emerged get replaced by other choices or actions that did not qualify the norm criteria earlier. We proposed a framework where agents are incentivized to move to a different choice, leading to the displacement of norms. The incentives provided depend upon factors like the current norm strategy, how far agents are from meeting the norm criteria, what alternative option we want to see as the norm, etc. We assume a fixed delta payout that agents receive to move from the established norm to something else which does not satisfy the norm criteria. We assume this delta payout to be fixed to maintain simplicity, but in reality, this is expected to depend on many other factors like population size, agents' neighbourhood size, network structure etc. Results show that delta payout 20 is sufficient to move agents from M outcome to H or L when agents are connected in a small world network.

This chapter aims to provide a computational framework for the evolution of norm in the muti-agent framework. We want to provide a base structure of how norms can evolve and how it changes with respect to changes in various parameter configurations. We have provided results

with respect to changes in crucial parameters like network structure, fixed agents' percentage, population size etc., and a direction on the impact of applying changes in these parameters on the evolution of optimal strategies. These results are broadly consistent with the existing literature. The broad insights presented here have been discussed briefly in some of the earlier literature; we provide a detailed and extensive analysis here. We acknowledge that the simulations exercise of such nature cannot be exhaustive since there could be multiple possibilities involved concerning different permutations around multiple parameter changes. We provide an easy-to-access computational framework to bridge this gap as well. To maintain brevity, we would leave it to the readers to check the impact of applying multiple parameter changes on evolution.

We have also provided tentative areas where this computational framework can be applied. The intent of putting these two games together derives from the thinking that we can convert a wide range of strategic situations involving strategic complementarity or substitutability into one of these forms. The naming game is essentially a coordination game. Conforming or matching behaviour is rewarded. On the other hand, in the Nash demand game, conforming or matching behaviour is not optimal often. If one's opponent opts for high demand, it is optimal to be submissive and demand low; If one's opponent opts for low demand, it's optimal to go aggressive and demand high. Hence combining these two types of games gives us a wide range of coverage in terms of applicability. The Nash demand game computational framework can be applied to any other game with different payoffs as long as finite actions are defined along with their payoffs. The framework provided would work in other symmetric games too. It can be applied to non-symmetric games also, but in that case, there would be a need to run different iterations, one corresponding to payoffs for the row player and another corresponding to payoffs for the column

player. And if we want to see the shift in norms after being in place for a certain period, we can then use the delta payout approach to see the norms shifting from one to another. And we expect this cycle to continue once old norms become obsolete and replaced by new norms. We can extend this thinking more broadly, where the intent of naming game is to identify potential choices or actions in terms of names. Then we provide some meaningful explanation for these names (e.g., High, Medium, or Low as in the Nash demand game) and define some payoffs associated with these choices. The Nash demand game framework decides which of these choices will emerge. And the delta payout approach completes the loop with norm displacement.

The rest of the chapter is structured into 7 sections. The next section provides a brief literature review followed by defining the norm criteria. The following two sections explain the Naming and Nash demand games in detail and the simulation results obtained. It is followed by a brief explanation of how existing norms can be displaced by new norms using the delta payoff argument. The chapter also briefly mentions how to use the Python libraries and some limitations of the study. The last section provides concluding remarks along with some tentative future research areas.

4.2 Literature review

The existing literature has analyzed norms evolution using analytical and computational methods. As discussed in the preceding chapters, the computational methods literature, in turn, can be divided into three categories. One which specifies finite normal-form games, strategies, and their payoffs, and agents choose the best strategies as they interact with other agents over a period. The strategy pair being played most frequently is considered a norm strategy (Axelrod, 1986; Shoham

& Tennenholtz, 1992; Young, 1993). In the second approach, agents are distributed according to their strategy. Agents' population size and their strategies are defined before starting the game. Agents interact with other agents over a period of time and adapt their strategies if following other strategies results in a higher payoff. Agents can use different methods to decide what strategy to choose. It could be a simple imitation of other agents' strategies or a more complex one, like using neural networks or genetic algorithms (Nishizaki et al., 2009). The strategy being played most frequently is considered a norm (Young &Foster, 1991; Axtell et al., 1999; Tesfatsion, 2003; Nande et al., 2020). And lastly, some literature has studied norms evolution in the context of agents' social networks. Agents are connected to a social network, interact with their neighbouring agents, and decide the optimal strategy. The nature and type of social network influence what strategy emerges as a norm (Alexander, 2007; Young, 2015). This chapter considers the second and third computational approaches of norm evolution out of these three. These approaches incorporate agents' connection and network information of agents into their decision framework. It also provides an opportunity to explore multiple possibilities, which can be easier to test in the computational framework than in the analytical framework (Alexander, 2007).

In the traditional game theory, expected utility involves making assumptions like continuity, substitutability, transitivity, and monotonicity. These conditions may not hold in all situations. In reality, heuristics incorporate a common body of beliefs acquired through participation in a common culture. Humans are bounded rational and take decisions based on less-than-perfect calculations derived from heuristics and rules of thumb (Alexander, 2007). In agent-based computational economics (ACE), agents are assumed to be bounded rational (Tesfatsion, 2003). Evolutionary models minimally require two things (Alexander, 2007). First, how to represent the

current state of population and second, the dynamical laws that explain how that state changes over time. Two models, the continuous and discrete models, represent the population. The continuous model uses global statistics to represent the population, implying that agents' choices and frequencies are represented at an aggregate level. The discrete model investigates the agent-level information and assesses how it impacts agents' decision-making. Axtell et al. (1999), Sen and Airiau (2007), and Martinez et al. (2021) are some of the papers which used a discrete model approach toward agent-based modeling. This paper primarily focuses on norm evolution using a discrete model approach.

Norm emergence depends upon the parameters specified and the process followed for evolution. The dynamics are defined by the model specified (agent-based discrete or aggregative), payoff values, population size, memory length, and the level of randomness present in agents' decision-making (Young & Foster, 1991; Kandori et al., 1993). Norms evolution results can vary in the short run versus the long run and are sensitive to how the model is specified. It calls for giving attention to the evolutionary processes (Young & Foster, 1991). Kandori et al. (1993) show that randomness or noise helps to reach coordination on a particular equilibrium that need satisfy being Pareto dominant. The paper shows that expected wait times matter for achieving the norm evolution in addition to the frequency distribution of strategies (percentage of times a particular strategy is followed). The authors show that upsetting the wrong equilibrium takes less time than the right one. Nishizaki et al. (2009) demonstrates the possibility of achieving a cooperative outcome in prisoners' dilemma game when agents make decisions based on learning mechanisms using neural networks and genetic algorithms. The input to the neural network-based model includes parameters on agents' choices in the prior period, population obedience rate in the

previous period, personal taste and preferences of agents, individual agents' utility in prior periods, aggregate utility of all agents in the prior period, degree of belief of individual agent for the social norm, among others.

Stochastic evolutionary game theory can be used to study the evolution of norms. In this context, Young (2015) defines five key elements of the evolution of social norms: persistence, tipping, punctuated equilibrium, compression, and local conformity or global diversity. Persistence implies that existing norms stay for long periods and generally adapt very slowly to changes in external conditions. Tipping indicates that when norms shift happens, the transition is quick. Once a specific threshold is reached and enough people start following new norms, this results in new perspective on actions which agents are going to take, and the transition to new norms completes rapidly. Punctuated equilibrium reflects the phase when there are sufficiently large periods of no changes followed by a sudden shift in activity in which new norms are displacing an old norm. This is called the punctuated equilibrium effect. Compression implies that individual choices have less variation than otherwise expected had there been no norm. For example, if landowners and tenants do not follow the conventional norm of 50-50, contract terms would have more diversity. Local conformity or global diversity indicates the possibility of different unique norms in local communities when agents do not interact or have very limited interactions with agents outside their local communities. This leads to norms followed in local communities that are different and unique to the specific local community. This contributes to a globally diverse pattern of social norms.

Social norms that emerged may not necessarily have convincing *apriori* justification, and they could have arisen due to random events (Axtell et al., 1999). The self-reinforcing nature of the process drives it. Any specific equilibrium can be considered as the conventional way of playing the game, not because it being prominent or focal but because it is selected by the dynamics of the process due to path dependency (Young, 1993). Results show there is endogenous creation of structures or equilibrium. Axtell et al. (1999) show with the help of the Nash demand game that when agents carry a distinguishable tag, e.g., dark skin or light skin, then equity state holds within each group, but discriminatory norms govern the relation between two groups. Agents can make one of three choices (high 70, medium 50, or low 30), and when a dark type agent meets a light type agent, a dark agent acts aggressively (demanding high or medium), and the light acts passively (demanding low). This results in the payoff to dark agents as over twice that of light agents. In other words, there is an endogenous creation of class distinctions. Light agents expect that dark agent will be demanding, so it is rational to submit to their demands instead of demanding high or medium and ended up getting nothing. Similarly, darks also come to realize that lights will submit, so it is rational to take advantage of them.

4.3 Norm definition

Different papers have considered different definitions of norms. For example, Shoham and Tennenholtz (1992) considered different thresholds from 55% to 95% of agents to validate the convention definition. Axtell et al. (1999) considered anything above 33%, given that agents had 3 choices, while Sen and Airiau (2007) used 50% as the cutoff.

We have considered two dimensions to check the norm establishment, the number of agents following the norm and the number of times it has been practiced. For any action or strategy to be called a norm, it should satisfy the following two conditions:

- At least 70% of agents should use it at any given time.
- It should be used at least 50% of the time.

If the population size is 20 and we run the simulation for 10 time periods, we expect a strategy ‘s’ to be called a norm if at least 14 agents play it for at least 5 time periods. The thresholds of 70 % and 50 % have relevance here because the strategy will qualify as a norm if a significant share of the population experience it for a longer period. This definition is used for both games.

In what follows, we study two games as discussed earlier. We have tried to maintain parity regarding the approach to both games. We have used similar response functions, network structures, and other parameter restrictions across both games.

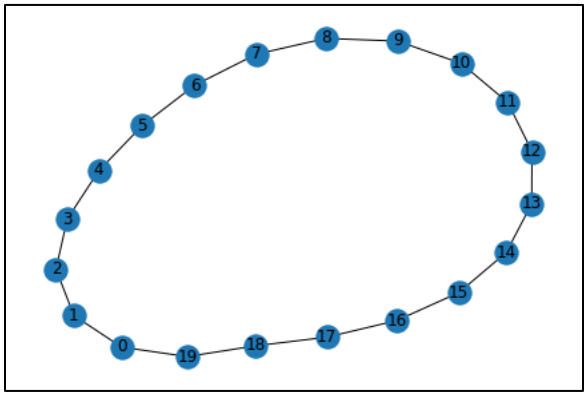
4.4 Naming game and evolution of norms

We explain norm evolution with the help of a *naming game* from Young (2015). Naming is a coordination game wherein two agents are shown a picture of a face. They require to simultaneously and independently suggest names for it. If agents provide the same name, there is a positive reward and if they provide different names there is a small penalty in terms of negative reward. There is no restriction on agents' names; this is left to their imagination. Agents do not know the identity of the other agents with whom they are being paired. At the end of a given time period, agents get to know the names proposed by other agents. Once agents have a history of

names proposed by other agents, they consider the names proposed by their opponents in the next successive periods. This way, agents get to know the names which are popular among the agents.

We start with a base model wherein we assume there are a total of 20 agents connected via the ring network. Agents are represented as nodes in the graph and are numbered from 0 to 19. Each agent is connected with 2 agents in the neighbourhood, as shown in Figure 4.1. For instance, agent 0 is connected to 2 agents, agents 1 and 19. Similarly, agent 5 is connected with agents 4 and 6. We assume the network to be undirected.

Fig 4.1. Ring network with 20 agents and 2 neighbours



We can represent the ring network with an adjacency matrix using the equations below and notations. Suppose N is total number of agents in the network, A is Adjacency matrix, a square matrix of size N x N and i is index for rows and columns of the matrix (representing individual agents). We define a function modulo (j, N) to ensure neighbor indices stay within the matrix range (0 to N-1) as follows:

modulo (j, N) = j % N

Define left and right neighbor indices for each agent i:

- Left neighbors: L_i = [modulo(i - 2, N), modulo(i - 1, N)]
- Right neighbors: R_i = [modulo(i + 1, N), modulo(i + 2, N)]

The element at row i and column j of the adjacency matrix, denoted by A_ij, is defined as follows:

A_ij = {

1: j ∈ L_i or j ∈ R_i,

0: otherwise

}

This notation captures the logic of connecting each agent with its two neighbors on the left and two neighbors on the right, considering the ring structure.

Agents play a Naming game assuming that the positive reward value from suggesting the same name and negative reward value from suggesting a different name remains constant throughout the game. We start the game assuming that agents do not have any prior history to look into; hence they suggest random names. A single edge of the network is selected at any given point, and agents associated with those edges play the game. The network's edge is represented in parenthesis as (0,1) or (5,4) etc., implying agents 0 and 1 are connected, and agents 5 and 4 are connected. We assume that all network edges participate in the simulation run during each time period. However, agents make their decisions based on their knowledge of names proposed by opponents till the last period. By the end of the current time period, agents update their knowledge about names proposed by opponents during the current period and use this revised knowledge along with the knowledge accumulated so far from the next periods onwards. Agents do not update their knowledge during

the time period which is currently running. This is done to ensure all network edges get an equal chance of being represented in the simulation run across all periods. Also, we assume full memory implies agents use their knowledge about names proposed by opponents from previous periods.

To begin with, agents do not have history to look back into; hence agents propose names randomly. We assume the name to be any 4-character keyword, a combination of 26 alpha (A-Z) and 10 numeric (0-9) characters, generated randomly, e.g., AS58, or XGH7. Agents keep track of names proposed by other agents and update their actions in successive rounds of play. We assume agents have bounded rationality and use limited calculations to decide what action to take.

We assume that agents use perturbed best response, implying agents can act randomly with a certain probability (Young, 2015). When agents require to take action at any given period, they consider the names proposed by their opponents so far and decide what names to propose in the current period. There could be many ways that agents can follow to decide what action to choose. We consider the following four possibilities agents can employ in their decision-making. We call this naming response function (NRF), and its different variants are labeled as NRF_i.

- NRF_1: Agents propose names that occurred most frequently in the past, and if there is more than one such name that was proposed most frequently, agents select one name randomly out of those. Agents do this 95% of the time. In the rest, 5% of the time, agents select names randomly from the ones that were not frequent. This 5 % represents the error possibility.
- NRF_2: Secondly, agents select names with a probability in the ratio of the corresponding frequency distribution of names proposed by opponents. Therefore, the names proposed

more frequently by opponents in the past have higher chances of being picked by the agent compared to the names which were relatively less frequent.

- NRF_3: This is similar to what agents do in the first possibility. However, the difference lies in how randomness is treated. Here we assume 5% randomness is with respect to all the names agents have observed in their past interactions, not just the ones that are not observed most frequently, as in the first case.
- NRF_4: In the fourth possibility, we assume no perturbation probability. Agents select the name which came most frequently in their interactions with 100% probability. However, if multiple names satisfy this condition, agents select any one name randomly.

We explain the functioning of the first response function with the help of a pseudo-code. Algorithm 1 below shows how the NRF_1 function works.

Algorithm 1 Naming response function1 (NRF1)

Require: agentI, strategydb,pR,namelen,pN
Ensure: bestname

```
if agentI IN strategydb then
    strategydb = strategydb ∋ agent = agentI
    frequencydb = table(opponentnameproposed,percentfreq) ∋ strategydb
    mostfrequent = frequencydb.opponentnameproposed ∋ percentfreq = max(percentfreq)
    Notmostfrequent = frequencydb.opponentnameproposed ∋ opponentname-proposed ∉ mostfrequent
    bestreponse = random((mostfrequent,Notmostfrequent)∝(1- pR, pR))
else
    bestresponse = random((a-z0-9),length=namelen)
end if
newrandomname = random((a-z0-9),length=namelen)
bestname = random(bestresponse,newrandomname)∝(1- pN, pN)
```

Below are the steps performed in Algorithm 1.

- Requires input on individual agent($agent_I$), *strategydb* which is the table containing agent name, and the names proposed by them and their opponents in each time period. This is one of the outputs from Algorithm 3 explained later in the chapter. *pR* is the perturbation probability or probability of error, *namelen* is the length of name that we want to generate, and *pN* is the probability of the agent recommending a new name.
- Provides recommended name as the output (*bestname*)
- Check if $agent_I$ is present in *strategydb*. If yes, perform below steps:
    - Filter *strategydb* to get records associated with $agent_I$ only.
    - Create a frequency table that contains two columns, names proposed by opponent agents, and the percentage of times it occurred in the simulation. This uses filtered *strategydb* created in the previous step as the input.
    - Get the most frequent name proposed by opponent agents.
    - Get all other names that are not most frequent.
    - Select one name randomly from the most frequent name and the rest of the others in the ratio of ($1 - pR$, pR).
- If $agent_I$ is not present in *strategydb*, then perform below steps:
    - Generate any one alpha numeric name randomly with the length of *namelen*
- Get a new alpha numeric random name with *namelen* size.
- The best-recommended name is one of the names used most frequently by opponent agents or the new random name. It is selected with the probability of ($1 - pN$, pN). The eventual name selected depends upon the value of *pN*.

We assume that agents follow NRF_1 response function with 5% randomness in their decisions. We let agents play the game for 50 time periods and check the norm emergence trend in the results. If we observe any name satisfying the norm criteria in the initial 50 periods, we stop running iterations beyond 50 periods. However, in cases where we observe a positive trend of the emergence of norms for any names but did not yet meet the norm criteria, we extended the iterations to 300 time periods to check if any name evolves into a norm later in the time period.

With the assumption of 20 agents, we can expect a maximum of 20 unique names in any given simulation iteration, provided *pN* equals zero. This is due to the assumption that agents select names randomly initially, but once they have history, they suggest names that opponents have already proposed more frequently. These 20 names may not be unique when we run iterations across different instances and can have many possibilities. This is due to the nature of these names, which are generated randomly using a combination of alpha and numeric characters. We are not interested in the specifics of what these 20 names represent but in how agents converge to a few of these names out of the maximum possible.

We have tried simulation iterations across four different network structures. We have considered ring, small world, complete, and 2-dimensional grid networks. The ring network is considered a circular network where each agent is connected with specific agents towards the left and right, as shown in Figure 1. In the case of a small-world network, a 'shortcut' is available. The presence of a 'shortcut' implies that the agent can directly connect from one node to another without traversing through the agents in transit. The significance of the small world here means each agent knows limited agents in the population and not all the agents. We have considered different variants of

small-world network, and we call them SW1, SW2, and SW3. The first variant (SW1) is Watts-Strogatz small world network, where if short-cut edges are added, it would remove the existing ring network edges and replace them with short-cut edges. In this case, the number of edges remains constant, as in the ring network. The resulting network need not necessarily be connected. A connected network implies the presence of at least one edge to reach from one node to another. The short-cut edges are added in the second case (SW2), but the existing edges are not removed. So, the number of edges in this network is higher than the other two small-world network variants. This produces the Newman-Watts-Strogatz small-world network. The third variant (SW3) is the same as SW1 except with the added property of network to be connected. All three small world networks are impacted by a parameter on probability (p_{edge})of rewiring existing edges or adding new edges. This probability is interpreted as the probability of rewiring existing edges in the case of SW1 and SW3 networks, while it is interpreted as the probability of adding new edges in the case of SW2 network. If this probability value is high, this indicates more short-cut edges are present in the network, while the opposite holds in the case of a low probability value. In the complete network, each node is connected to every other node. A lattice network is a spatial network where each node represents agents' spatial position. It is also called a grid network. This chapter considers the 2-dimensional grid network where agents are placed in a 2-dimensional grid-like structure with a specified number of row and column agents.

Below we explain the rationale of using these specific networks.

- Watts–Strogatz Small-World Network. This includes both SW1 and SW3 networks.

    - Small-World Property: SW1 and SW3 exhibit the small-world property, meaning that the average distance between nodes is relatively short. This property is essential for studying how information and influence spread quickly through a network.
    - High Local Clustering: Despite having short average distances, small world networks maintain high local clustering. This combination of short paths and local connections is relevant for understanding how social norms propagate within tightly connected communities.
    - Real-World Relevance: Small world networks have been shown to capture the structure of real-world networks, making it a suitable choice for modeling social interactions.
- Newman–Watts–Strogatz Small-World Network (SW2 network)
    - Enhanced Structure: SW2 builds upon the original SW1/ SW3 by incorporating additional features (such as community structure) to better represent real-world networks. This is done by introducing additional short-cut edges into the network.
    - Community Detection: SW2 allows for the study of community dynamics and how social norms differ within and between communities by adding additional edges.
    - Robustness and Resilience: Investigating how social norms evolve in the presence of community boundaries and inter-community interactions can provide insights into network robustness and resilience.
- Complete Network (Fully Connected):
    - Extreme Case: A complete network represents the extreme case where every agent is directly connected to every other agent. It provides a baseline for understanding how social norms evolve when information spreads instantly and universally.

    - Homogeneity: Complete networks assume homogeneity, which simplifies the analysis. It's useful for studying scenarios where everyone has equal influence and access to information.
    - Emergence of Consensus: In a complete network, social norms may converge more rapidly due to the absence of information bottlenecks.
- 2D Grid Network:
    - Spatial Structure: 2D grid networks introduce spatial constraints, mimicking physical proximity. Agents interact primarily with their immediate neighbors, which can impact the diffusion of social norms.
    - Local Influence: In a grid network, agents' interactions are limited to their local neighborhood. This localized influence can lead to the emergence of distinct norms in different regions.
    - Boundary Effects: Investigating how norms evolve near the network boundaries provides insights into how spatial constraints affect norm adoption and persistence.

We have explained the network graph selection in Algorithm 2 below:

Algorithm 2 Network graph

Require: network, numagents, numneighbours , pEdge, seed, gridnetworkM, gridnetworkN
Ensure: graph,alledges

```
if network = 'smallworld' then
    graph= wattsstrogatz(numagents,numneighbours,pEdge,seed)
    alledges = graph.edges
else if network = 'newmansmallworld' then
    graph= newmanwattsstrogatz (numagents,numneighbours,pEdge,seed)
    alledges = graph.edges
else if network = 'complete' then
    graph= complete(numagents)
    alledges = graph.edges
else if network = 'grid2d' then
    graph= grid2d(m=gridnetworkM,n=gridnetworkN)
    alledges = graph.edges
end if
```

Below are the steps performed in Algorithm 2. To generate graphs, we have used *networkx* python library.

- It requires input parameters on network structure (*network*), the number of agents (*numagents*), neighbourhood size (*numneighbours*), probability of edge rewiring or adding new edges (*pEdge*), random seed number (*seed*), 2-dimensional grid network parameters (*gridnetworkM, gridnetworkN*).
- It produces output in terms of graph object(*graph*) and the edges associated with the network(*alledges*).
- If the network type is '*smallworld'*, it generates Watts-Strogatz network. If the *pEdge* value is zero, it generates a ring network.
- The Newman-Watts-Strogatz network is generated by specifying the network type of *'newmansmallworld'*.

- Similar is the explanation for other network types, complete and grid networks.

Algorithm 3 below explains the underlying steps in creating the dataset or table, which would be used later to determine if any name satisfies the norm criteria.

Algorithm 3 Norm dataset creation

Require: numtrials, responsefunction(), alledges
Ensure: strategydb,normdb

```
for timeperiod 1 to numtrials do
    for edge in alledges do
        agent = edge[0]
        opponentagent = edge[1]
        nameproposed = responsefunction().bestname
        opponentnameproposed = responsefunction().bestname
        strategydb1 = table(agent,opponentagent,nameproposed, opponent-
nameproposed,timeperiod)
        agent = edge[1]
        opponentagent = edge[0]
        nameproposed = opponentnameproposed
        opponentnameproposed = nameproposed
        strategydb2 = table(agent,opponentagent,nameproposed, opponent-
nameproposed,timeperiod)
        strategydb = table(union(strategydb1,strategydb2)
    end for
    normdb = table(nameproposed,timeperiod,agentpercentshare) ∋ strategydb

end for
```

Below are the steps involved in Algorithm 3.

- It requires input parameters on the number of trials (*numtrials*), response function (*responsefunction()*), and edges of the network (*alledges*). The response function is the output from Algorithm 1, and edges are one of the outputs from Algorithm 2.
- It creates two output tables. *strategydb* contains information about agents, their opponents, and the names proposed by them during each time period. *normdb* is a table that takes

strategydb table as input and contains information on the percentage distribution of agents across different names proposed in each period.

- The algorithm starts from time period 1 till *numtrials*.
    - It then iterates through all the network edges (*alledges*), taking one edge at any given time.
    - Each edge of the network has two agents, an agent and an opponent agent.
    - Agents selected through edges play the naming game and suggest names according to the *responsefunction()*.
    - This information is stored in a table, *strategydb1,* which contains the agent, opponent agent, the name proposed by both the agents, and the time period when these names were proposed.
    - *strategydb2* is a replica of *strategydb1* table except that this table is from the perspective of opponent agent. The opponent agent from *strategydb1* is the main agent in this table, and the main agent from *strategydb1* is the opponent agent here. This ensures we store the main and opponent agents' choices in the same structure.
    - *strategydb* is the union of both *strategydb1* and *strategydb2* tables.
    - *normdb* is the table that takes *strategydb* table as the input and provides information on the percentage of agents who played different norms at different time periods.

Algorithm 4 below shows how the game tests the norm conditions internally.

Algorithm 4 Testing norm conditions

Require: normdb, agentfreqcutoff, timefreqcutoff
Ensure: normname

```
potentialnormcandidates = unique(normdb.nameproposed) ∋ normdb.agentpercentshare ≥ agentfreqcutoff
if potentialnormcandidates ≠ ∅ then
    for name IN potentialnormcandidates do
        normdbfiltered = normdb ∋ nameproposed=name
        timeratiofrequency = length(unique(normdbfiltered.timeperiod))/length(unique(normdb.timeperiod))
        if timeratiofrequency ≥ timefreqcutoff then
            normname = name
        else
            print('name does not satisfy the norm criteria')
        end if
    end for
end if
```

Below are the steps involved in Algorithm 4.

- It requires input parameters on *normdb*, one of the outputs from Algorithm 3. In addition, it requires norm-determining cutoffs, *agentfreqcutoff*, which is the minimum percentage of agents required to follow a specific name to be called a norm. Another parameter, *timefreqcutoff* determines the minimum percentage of time periods for which the specific name is to be played sufficiently among all time periods (*numtrials).*
- It gives the output name (*normname*), which satisfies the norm conditions.
- The algorithm first checks if any name satisfies the *agentfreqcutoff* condition. It checks the unique names proposed by agents in *normdb* table and filters those records where the *agentpercentshare* value from *normdb* table is higher than or equal to *agentfreqcutoff* value. *agentpercentshare* column in *normdb* table shows the percentage of agents who proposed the respective name during a given time period.

- If there is at least one name that satisfies the *agentfreqcutoff* condition from the previous step (*potentialnormcandidates* is a non-empty set), then it checks the second condition of *timefreqcutoff*
    - First, a new table *normdbfiltered* is created, which takes *normdb* table as input and filters only those records where the name satisfies the *agentfreqcutoff* condition.
    - A new variable *timeratiofrequency* is created, which is the ratio of the number of distinct timeperiods where the name satisfying the *agentfreqcutoff* condition is played with respect to all time periods.
- If *timeratiofrequency* value is higher than or equal to *timefreqcutoff* value, then the name satisfies the norm criteria. Else the name does not satisfy the norm conditions.

Results vary with respect to different network structures. Ring network results show at least one name satisfying the norm criteria majority of the time. Results from small world networks vary with network types and probability p_{edge}. In the case of SW1 and SW3 networks, the lower probability of rewiring edges translates to a higher possibility of norm emergence. However, in the case of SW2 we have seen the opposite trend with respect to probability value. The higher probability of adding new edges translates to a higher possibility of norm emergence. Figure 4.2 shows the norm trend in the case of SW2 network with a 50% probability of adding new edges. This figure shows time period on X-axis and the percentage of agents who proposed the respective name on Y-axis. The values on the Y axis are shown in ratio form, ranging from 0 to 1, implying 0% agents to 100% agents, respectively. ‘Q4WP’ and ‘TQH8’ are random names where ‘Q4WP’ satisfies the norm criteria. Figure 4.3 shows the network graph at the 50^{th} time period. The network graph colours the nodes in 2 distinct colours. This implies that agents with the same colour nodes proposed the same name more frequently during the simulation run. For example, agents 17, 18,

and 19 proposed the same names more frequently, while the rest proposed some other names more frequently. The colour itself has no significance here; it is used to demonstrate which agents often propose the same name. The network graph colours the agent nodes based on the most frequent name proposed by agents in the entire time period for which the game has been played.

Fig 4.2. Norm emergence in Newman-Watts-Strogatz small-world network (SW2)

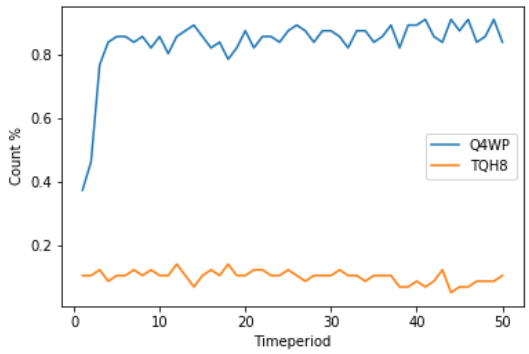
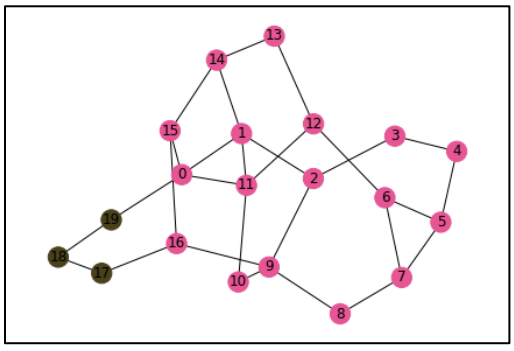


Fig 4.3. Newman-Watts-Strogatz small-world network (SW2) at the end of 50th time period



In the case of a complete network, one name is being proposed, and norm emergence is also the fastest, which is similar to the result obtained by Young (2015). In the 2d grid network, we have taken 5*4 grid to ensure the total number of agents remains 20. We have seen mixed results in the case of 2d grid networks, where we observed norm emergence in some cases while, in others, we

did not see it. Figure 4.4 shows one of the results from the grid network where none of the names satisfies the norm criteria by the end of 50 periods, and the trend remained the same till 300 time periods. Figure 4.5 shows the network state by the end of 300 time periods.

Fig 4.4 2d Grid network where norm emergence is not observed

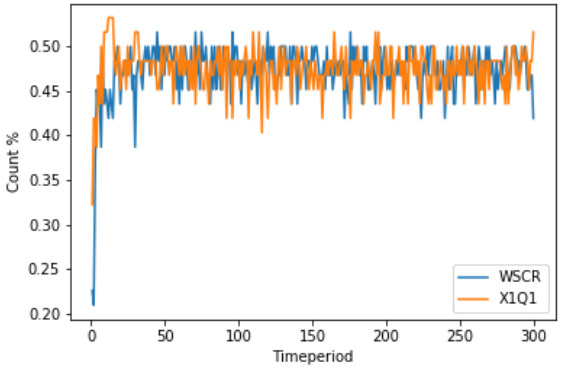
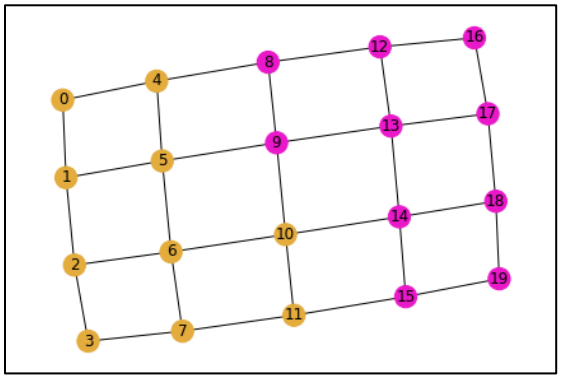


Fig 4.5. 2d Grid network at the end of 300 time periods



4.4.1 Impact of fixed agents

So far, we have assumed all agents observe what their opponents have done and accordingly update their actions. Next, we modified the base model and assessed the impact of fixed agents. We assume fixed agents as agents who continue to use one name throughout the simulations run and

do not change in response to their opponents. We assume 3 agents as fixed out of a total of 20 agents. We assume agents 5, 10, and 15 are fixed to ensure they are spread throughout the network. We keep the rest of the parameters constant as was in the base model.

Ring network results did not show any emergence of norms with fixed agents. The trend is similar with respect to SW1 and SW3 small-world networks. In the SW2 small world network, the norm emergence happens with a higher p_{edge}. It also takes longer to satisfy the norm criteria. Figure 4.6 below shows the emergence of the norm ('43AJ') at the 49th time period in the SW2 network with p_{edge} value of 75%. Figure 4.7 shows the network state at the 50th period.

Fig 4.6. Emergence of norms in SW2 network with fixed agents

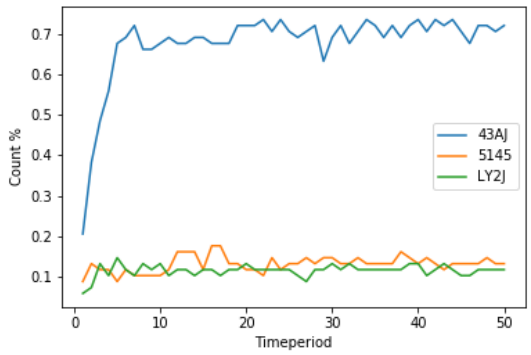


Fig 4.7. SW2 network at 50th time period with fixed agents

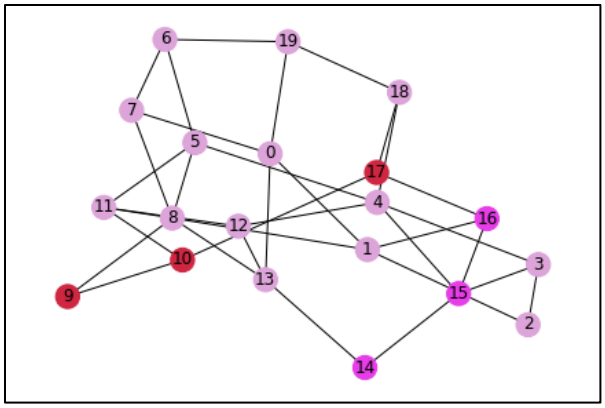


Figure 4.7 shows more agents follow Agent 5's fixed name than agents' 10 and 15 fixed names. The network has 4 edges corresponding to agent 5, {(5,8), (5,11), (5,6), (4,5)}, 3 edges for agent 10 {(9,10), (10,11), (10,17)} and 4 edges for agent 15 {(14,15), (15,16), (3,15), (4,15}. In the case of a complete network, we can see the emergence of norms, but the names satisfying the norm criteria are not the ones that fixed agents follow. Figure 4.8 shows a complete network where agents 5, 10, and 15 follow their own fixed names, but the rest follow the same name, which differs from any of the fixed agents' names.

Fig 4.8. Complete network at the 50th timeperiod with fixed agents

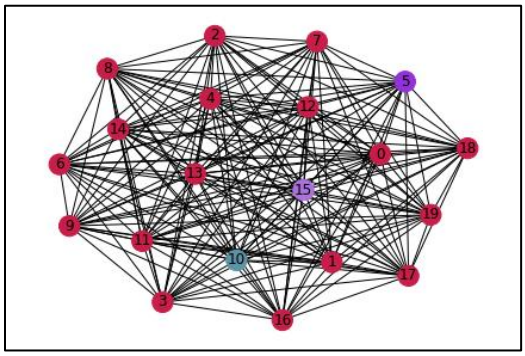


Fig 4.9. Convergence in complete network with fixed agents

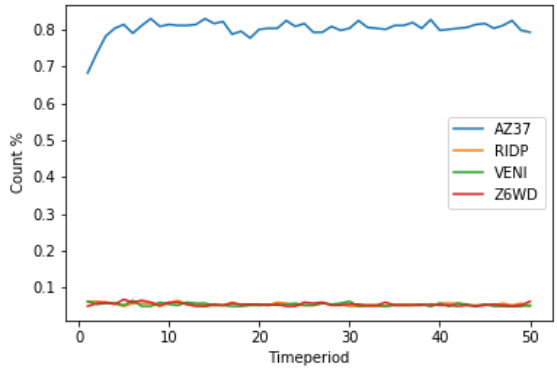


Figure 4.9 shows that the name which emerged as a norm ('AZ37') differs from what the rest of the fixed agents propose in the case of a complete network. In the case of the grid 2d network case, we did not observe any name satisfying the norm criteria.

4.4.2 Impact of population size

So far, we have assumed the population size as 20. Next, we assessed the impact of population size on the evolution of norms. We increased the agent count to 40, keeping other parameters constant, as in the base model. We did not assume any fixed agents for assessing the impact of population size. With the increased agent count, the number of distinct names which can be present in the overall population is increased. Since agents have full memory, the chances of a single name satisfying the norm criteria are generally reduced. However, the results are mixed when analyzed in the context of different network structures.

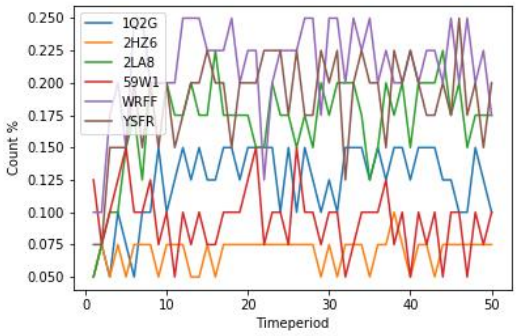
We have not seen any impact of increased population size on the evolution of norms in the case of the ring network. In the case of SW1 and SW3 networks, the chances are lower for a name to

emerge as the norm with increased population size. In the SW2 network, there is not much impact on the emergence of the norm with the higher p_{edge} value. We have also not seen much impact for complete networks with higher population sizes. The trend is mixed with the grid network.

4.4.3 Impact of positive probability of suggesting new name (pN)

We have assumed that agents can suggest new names only at the beginning of the game when they do not have any past interactions. We relax this assumption now and assess the impact of the probability of suggesting a new name later in the game. We assume this probability to be constant at 5%. We keep the rest of the parameters as constant as in the base model with 20 population size and no fixed agents. Results, in general, show a lower possibility of any single name satisfying the criteria of the norm due to more new names being present in the population. Ring network, SW1, and SW3 network results support this view. Figure 4.10 below shows the most common names proposed by agents by the end of 50 time periods from the SW1 network with p_{edge} value of 75%. The total number of unique names proposed by the end of the 50 periods turned out to be 136.

Fig 4.10. Most frequent names proposed by agents in SW1 network with 5% probability of new name



We can see from Figure 4.10 that the probability of any name meeting the criteria for the norm is lower as any specific name is followed by at most 20-25% of agents. However, in the case of the SW2 and complete networks, we can see a single name satisfying the norm criteria despite having more new names available. The trend in the case of 2d grid network is mixed.

To summarize, results from the naming game show that few names are emerging to satisfy the norms' criteria. The number of names satisfying the norm criteria is significantly lower than the overall names present in the population. This result is similar to Epstein (2001), where there is an endogenous emergence of the average optimal search radius resulting from agents' interactions over a period**.** Hence, there is the endogenous formation of a finite optimum number of strategies (names in our context) even when to begin with, there are no fixed strategies defined. Results vary to a certain extent based on network structures, network properties, population size, neighbourhood size, fixed agents etc. For instance, in the case of small-world networks, the norms that emerged are not necessarily locally concentrated compared to ring network, implying we can have two agents following the same norm without any direct linkage.

This result is significant in contexts where agents do not have defined strategies to choose from. Agents are expected to interact in an environment with other agents and create these strategies. For example, parents can decide on reasonable names for their kids. However, before choosing any name, they generally consider what names have been used by other parents for their kids. This way, they can reduce their potential name choices to a manageable number. They can either choose the name other parents have used or they can still propose a new name. And this cycle continues where no fixed set of names is defined, to begin with, but still parents narrow their choices to a

few. Another example could be designing a new compensation strategy in an organization. A private organization can design any compensation composition for their employees regarding the split between base pay and variable pay, assuming no regulatory restrictions exist. But before deciding the optimal split, they consider the compensation strategies of other organizations to remain employee-friendly employers in the market. When a new organization enters the market, they can either follow the other companies' composition or tweak it marginally to remain competitive. This leads to a fewer number of potential splits, which can happen between base and variable pay when, to begin with, there could be multiple possibilities.

4.5 Nash demand game and norm evolution

In this game, we assume agents can make 3 demands, High with a payoff of 70, Medium with a payoff of 50, or Low with a payoff of 30. The rules specify that agents would get their demanded shares if the total demands made by two agents in any given iteration is less than 100. If it exceeds 100, no agent will get anything. Below is the payoff matrix of the row versus column player (Table 4.1).

Table 4.1 Nash demand game

	H	M	L
H	0,0	0,0	70,30
M	0,0	50,50	50,30
L	30,70	30,50	30,30

There are 3 pure strategy Nash equilibria, (H, L), (M, M), and (L, H). There are four mixed strategy Nash equilibria, {(0.4, 0.6, 0), (0, 0.29, 0.71)}, {(0.57, 0, 0.43), (0.57,0 ,0.43)}, {(0, 0.29, 0.71), (0.4, 0.6, 0)} and {(0.4, 0.17, 0.43), (0.4, 0.17, 0.43)}. There is no strictly or weakly dominant strategy for any row or column player. We assume agents are placed in a specific network wherein each agent is connected with other agents via an edge. We call H, M and L three distinct strategies or actions agents can follow.

We assume there is a total of 20 agents in the population with neighbourhood size of 2, implying each agent is connected with two other agents. However, this does not necessarily hold for some networks like SW2, where additional edges are added to the existing network. As in the case of the Naming game, we have run all the iterations initially for 50 time periods. If we have seen any strategy satisfying the norm criteria laid out, we have not proceeded with further time periods. However, in cases where we have seen 50 time periods, there is no emergence of norms, and at the same time, if we observe a trend that indicates the possibility of a strategy satisfying the criteria for norm, in those cases, we have run the iterations further up to 300 time periods.

To keep consistent with the previous Naming game, we have formulated four different response functions which agents can deploy while making decisions. These response functions take inputs on the number of agents, agents' current actions, payoffs, and error probability. Error probability is interpreted as the probability of taking action randomly, which we assume is 5%. Agents calculate their payoffs according to the actions followed by them and their network. For example, if the agent with action H is connected with agents following M and L, the payoff of the H agent from meeting with the M agent equals 0 as the total demand (70+50 = 120) exceeds 100. Similarly,

the payoff of the H agent from meeting with the L agent equals 70; hence the total payoff equals 70 +0 = 70. Agents update their strategy if any of their neighbours are having higher payoff, or they stick to their own strategy they have been following. In case of a tie, where payoffs are the same with following the current strategy versus payoffs from neighbours' strategy, agents choose to stick to what they have been following already. We call below different variants of response functions as RF_i:

- RF_1: Agents select actions that have a maximum payoff. If that action is the agent's current action, they continue to use that in the next period also. If that action is something else, they will choose that action. If there is more than 1 action that results in maximum payoff and is not the action the agent is currently following, agents choose any one action randomly. Agents choose the maximum payoff action with a 95% probability and choose any from the remaining actions with a 5% error probability.
- RF_2: Agents assign weights to different strategies or actions. These weights represent the percentage of times the corresponding strategy has been chosen in the last period. Therefore, the actions chosen more frequently by agents' networks in the past have higher chances of being chosen by the agent in the current period than the relatively less frequent ones.
- RF_3: This response function is similar to what agents do in the first response function (RF_1). However, the difference lies in how randomness is treated. Here we assume 5% randomness is with respect to all the actions, not just those that do not have a maximum payoff.

- RF_4: In this function, we assume there is no randomness. Agents select the action which has the maximum payoff with 100% probability. However, if multiple actions have equal and maximum payoff, agents select any one action randomly. If an agent's current action is resulting maximum payoff, agents will continue to use that action.

In this game also, we have only considered the RF_1 function with a 5% error probability for all the simulation iterations. We also assume that agents consider the most recent period actions while calculating their payoffs. We assume that agents play Nash demand game where they require to choose any one strategy out of three (H, M and L) in any given period.

The simulation exercise starts with specifying the initial state. The initial state specifies the agents' distribution of strategies out of H, M, and L. We assume 20 agents are distributed in the ratio of 40/40/20 following High (H)/Medium (M)/Low (L) strategies. This implies that out of 20 agents, 40% (8) agents follow H, another 40% (8) follow M, and 20% (4) agents follow L. There is no specific reason to select this distribution; we wanted to give enough scope for an equitable distribution to emerge; hence we assigned equal weightage to both H and M. There are many ways in which these 20 agents can be represented in the network and still satisfy the criteria of H, M, L following 40/40/20 distribution. We can represent these different ways/states in the form of a list, e.g. [H,M,H,H,H,H,H,H,M,H,M,M,M,M,M,L,L,L,L]. There could be multiple possibilities in terms of how H, M, and L are distributed in that ratio. This list representation is considered as the initial state of the population, which is interpreted as the first agent following H, the second agent following M, the last agent following L etc.

We start the game with a randomly chosen initial state, distributed in the ratio of H/M/L as (40/40/20). Agents are connected with other agents according to a specific network structure. At each time, all agents need to decide whether to continue using the existing strategy or change to a different one. Agents' decision on what to follow depends on their payoffs from different strategies. We explain the payoff calculation with the help of a small example.

Suppose the initial state is [H,M,L,L,H,M,L], and agents are placed in a circular network. The first agent follows the H strategy, and the last agent follows the L strategy. Suppose each agent is connected with one agent towards the left and another towards the right. The first agent following H is connected with neighbours following M and L strategies. The payoff for the first agent is, therefore, 70. Similarly, for the second agent following M, its neighbours are agents who follow H and L. Therefore, the payoff for the second agent following the M strategy is 50. And similarly, the rest of the agents compute their payoffs. Once these payoffs are computed, we create a list of payoffs, for example, [70,50, 60,…60]. Here the first agent's payoff value is 70; the second agent's payoff value is 50, and so on. The payoff for the first agent is 70, and the neighbours' payoff is 60 (first agent payoff from last) and 50 (second agent payoff from the start). Since the agent's current payoff of 70 is higher than the other two payoffs, they stick to their original strategy H. For the second agent, the payoff is 50, but the neighbours' payoff is 60 and 70. Hence, the second agent decides to switch strategy from M to H as the 70 payoff is associated with the first agent who follows H. Similarly, the rest of the agents decide whether to continue to follow the existing strategy or shift to strategies followed by neighbouring agents if that results in higher payoffs. This leads to a new state of agents like [H, H,….]. This new state becomes an input in time period 2 for agents to perform payoff computation and again decide what strategy to follow in the

following periods. And this process goes on till the time period for which the game is played. Agents consider only the immediately preceding period state as input to decide what strategy to follow in the next period. This is a deviation from what is done in the Naming game where we assume agents have a full memory and consider all previous periods' history. The strategies are already fixed and defined in the Nash demand game like H, M, and L. We are interested in how the initial state distribution of agents changes as the game is played repeatedly. Hence, we consider the most recent state of agents as an input to decide the next state of agents.

Algorithms 5 and 6 provide details about the computation of payoffs and agents response function (RF_1) under the Nash demand game in algorithmic form.

Algorithm 5 Calculate payoff

```
Require: numagents, agentstrategydb, alledges,payoffmatrix
Ensure: agentpayofflist
  agentpayofflist = ∅
  for agentI IN numagents do
      agentIneighbours = alledges.neighbours ∋ agent = agentI
      strategyagentI = agentstrategydb.strategy ∋ agent = agentI
      datatocheck = agentstrategydb ∋ agent IN agentIneighbours
      agentpayoff = 0
      for agentJ IN agentIneighbours do
          agentpayoff += payoffmatrix(strategyagentI,datatocheck.strategy)
      end for
      agentpayofflist = (agentpayofflist,agentpayoff)
  end for
```

Below are the steps involved in Algorithm 5.

- It requires input on the number of agents (*numagents*), *agentstrategydb*, a table containing 2 columns, agent number, and the strategy they followed in the most recent period. *alledges*

is the edges of the network, which is one of the outputs from Algorithm 2. *payoffmatrix* is a table containing payoff values associated with different row versus column player strategies.

- It produces outputs in the form of list (*agentpayofflist*) containing each agent's total payoffs realized when they interact with their neighbours in the network.
- The algorithm starts with an empty set of *agentpayofflist*.
- It iterates through each agent and performs following:
    - Get all the neighbours of the respective agent (*agentIneighbours*)
    - Get the strategy for the agent selected (*strategyagentI*)
    - Filter the *agentstrategydb* table and get all the records of the agent's neighbours (*agentIneighbours*). Name this table as *datatocheck*.
    - Initialize *agentpayoff* variable to zero value.
    - Iterates through all agent's neighbours (*agentIneighbours*)
        - Get the payoff value from the *payoffmatrix* where row strategy is agentI's strategy (*strategyagentI*) and column strategy is one of agentI's neighbours' strategy. *datatocheck* table contains the agent's neighbour's strategy.
        - Add the payoff value to the *agentpayoff* variable created in the previous step.
        - Repeat this for all agentI's neighbours (*agentIneighbours*).
    - Fill the *agentpayofflist* empty set created in the first step with the value from *agentpayoff* variable.

Algorithm 6 Response function1 (RF1)

Require: numagents, agentstrategydb2, alledges,pR,uniquestrategies
Ensure: agentbestresponselist

```
agentbestresponselist = ∅
for agentI IN numagents do
    agentandneighbours = alledges.allagents ∋ agent = agentI
    datatocheck = agentstrategydb2 ∋ agent IN agentandneighbours
    if datatocheck.payoff ∋ (agent = agentI) = max(datatocheck.payoff) then
        recomm = datatocheck.strategy ∋ agent = agentI
    else
        recomm = datatocheck.strategy ∋ payoff = max(payoff)
    end if
    nonrecomm = uniquestrategies ∋ strategy ≠ recomm
    bestresponse = random(recomm,nonrecomm)∝(1- pR, pR)
    agentbestresponselist = (agentbestresponselist,bestresponse)
end for
```

Below are the steps involved in Algorithm 6.

- It requires inputs on the number of agents (*numagents*), *agentstrategydb2,* which is a table containing agent number, strategy, and total payoffs from the most recent period. The total payoff column in this table is the output from Algorithm 5. *alledges* is the network edges which is one of the outputs from Algorithm 2. *pR* is the probability of an agent choosing strategy randomly. *uniquestrategies* is the list of all unique strategies available for the agents to choose from.
- It produces output in the form of list containing each agent's best response (*agentbestresponselist*). The strategy column in *agentstrategydb2* table represents this list output in the preceding period.
- Initialize *agentbestresponselist* to empty set.

- Iterate the following for all agents (*numagents*).
    - Get the respective agent and its neighbours from all the edges (*alledges*). Name this as *agentandneighbours*.
    - Filter *agentstrategydb2* table to contain only those records containing information about the respective agent and its neighbours. Name this table as *datatocheck*.
    - If agentI's payoff is the maximum among all its neighbours, stick to the strategy that agentI is already following (*recomm*).
    - If agentI's payoff is not the maximum among all its neighbours, choose the neighbours' strategy, which results in maximum payoff (*recomm*).
    - Get all the remaining strategies from *uniquestrategies* that do not belong to the strategy selected in the above step (*nonrecomm*).
    - Select the best response randomly from the recommended strategy (*recomm*) and non-recommended strategies (*nonrecomm*) in the proportion of $(1 - pR, pR)$. Name this as *bestresponse*.
    - Fill the *agentbestresponselist* empty set created in the first step with the value from *bestresponse* variable.

The algorithmic formulations for the norm dataset creation and testing norm conditions for the Nash demand game are not shown separately as there is not much deviation from what has been followed in the Naming game, except that name is replaced by H, M, or L strategy.

Results show that medium strategy M satisfies the criteria laid out for the norm across all four network structures. In the case of a complete network though, we have observed higher volatility

during the initial time periods before convergence towards M strategy. Figures 11, 12, and 13 represent the results from the ring network with the initial state of [‘M’, ‘M’, ‘L’, ‘H’, ‘M’, ‘H’, ‘L’, ‘M’, ‘M’, ‘H’, ‘M’, ‘H’, ‘M’, ‘H’, ‘H’, ‘M’, ‘L’, ‘H’, ‘H’, ‘L’]. Figure 4.11 shows the percentage of agents who played H, M, and L on Y -axis across the different time periods from 1 to 50. The values on Y-axis are shown in ratio form ranged 0 to 1, implying 0% agents to 100% agents, respectively. Figure 4.12 shows the network state at the beginning of the game, which depicts 8 M agents (40%), 8 H agents (40%), and 4 L-type agents (20%). Figure 4.13 shows the network state at the 50th time period where we can see 18 agents following M and 2 agents following H.

Fig 4.11. Convergence in ring network (Nash demand game)

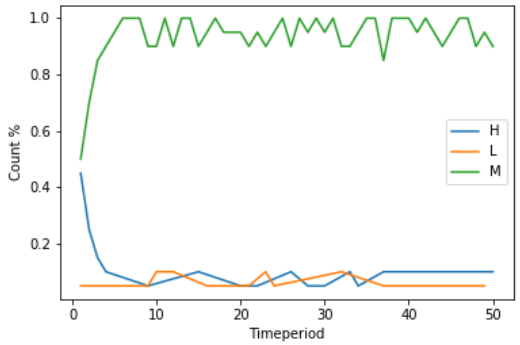


Fig 4.12. Ring network at the beginning of Nash demand game

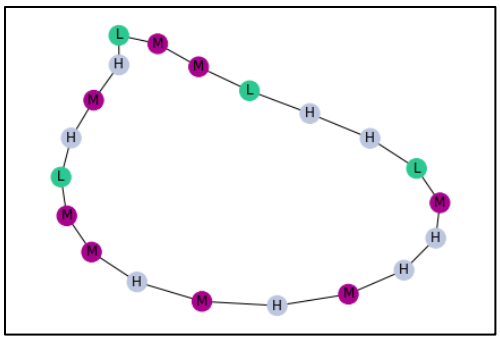


Fig 4.13. Ring network at 50th time period (Nash demand game)

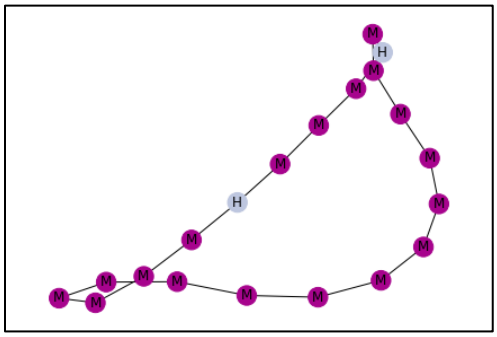
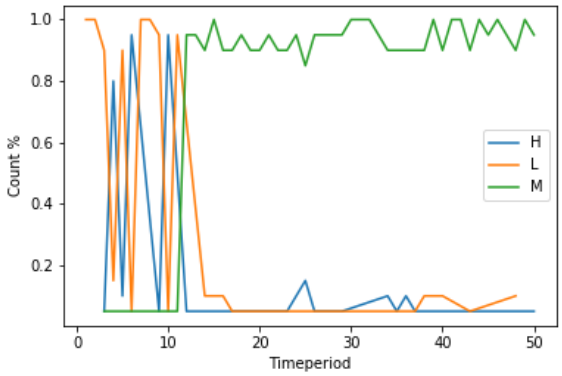


Figure 4.14 shows the complete network results with the initial state of ['M', 'L', 'H', 'H', 'H', 'H', 'H', 'M', 'L', 'L', 'M', 'H', 'M', 'L', 'H', 'M', 'M', 'H', 'M', 'M']. We can see the delay in the emergence of the M norm in the case of a complete network.

Fig 4.14. Convergence in complete network (Nash demand game)



4.5.1 Impact of fixed agents

Next, we assessed the impact of fixed agents. As in the case of the Naming game, we assume 3 fixed agents and are assumed to be on nodes 5, 10 and 15. We assume all fixed agents follow H

strategy irrespective of what their neighbours follow. This resulted in either delaying in emergence of M strategy as a norm or in some cases, we have seen this results in oscillation between agents following M and H strategy frequently. Results from the ring and small-world networks (SW1, SW3) show similar patterns where we see medium not satisfying the norm criteria. Figure 4.15 shows the results from the SW1 network with the initial state of ['H', 'H', 'M', 'M', 'M', 'H', 'M', 'M', 'H', 'M', 'H', 'L', 'H', 'M', 'H', 'L', 'L', 'M', 'H', 'L'] with p_{edge} value of 0.75. Here we can see that agents keep oscillating between H, M and L strategy throughout the period. Figure 4.16 shows the initial state of the network, and Figure 4.17 shows the end state of the network.

Fig 4.15. Constant oscillation in small world network (SW1) with fixed agents (Nash demand game)

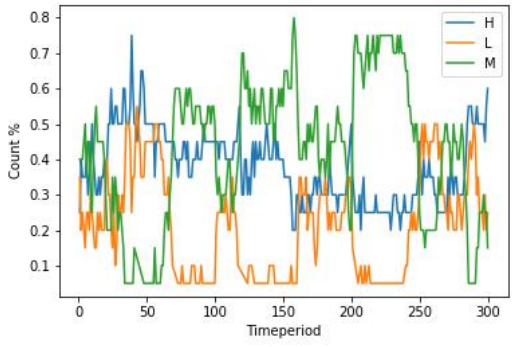


Fig 4.16. Initial state of small world network (SW1) with fixed agents (Nash demand game)

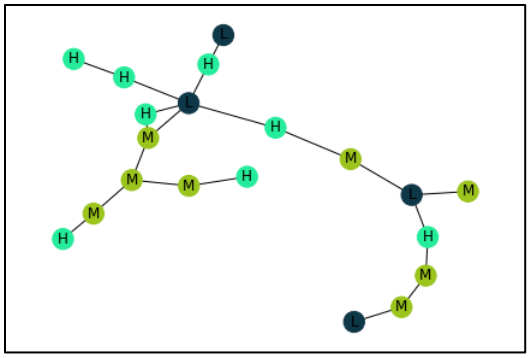
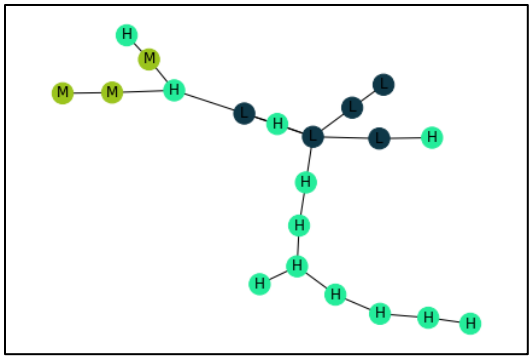


Fig 4.17. Small world network (SW1) with fixed agents at 300th time period (Nash demand game)



The lower value of p_{edge} in SW1 and SW3 networks increases the chances of M as the norm but not high enough to satisfy the norm criteria. However, SW2 results show that M still dominates despite fixed agents. The probability of adding new edges does not significantly impact evolution. Complete network and grid network results also show similar results where M is still frequently chosen in most cases.

4.5.2 Impact of population size

The iterations so far assume the number of agents as 20. We try to assess if increased population size impacts the evolution of norms in the Nash demand game. We assume the rest of the

parameters are the same as in previous iterations with no fixed agents. When we increased the agent count to 40, we did not observe many changes in norm evolution, which means we see M as the outcome in most cases. Although in some cases, we have seen the emergence of norms is delayed slightly. This result is similar to Axtell et al. (1999), which show that equity norms once established, are stochastically stable, implying it takes much longer to undo the equity norm once it is established than to undo the fractious regime once it is in place.

Equitable norms emerge when we allow for enough M agents in the population. This trend is sustained in the presence of few agents always following a non-equitable strategy (H) under some network structures like Newman-Watts-Strogatz small world, complete network etc.

4.6 Norm displacement and emergence of new norm

So far, we have assessed that when agents do not have any defined choices or actions to begin and follow the coordination game dynamics, this leads to a few strategies or choices with the help of context from the naming game. And once we have some defined choices like H, M, or L in the case of the Nash demand game, we have seen how the distribution of agents following these strategies changes as agents play the game repeatedly with their neighbours. Now, we analyze how these norms are displaced and replaced by some other strategy like H or L once these norms are established, like M in the case of the Nash demand game. We continue to use the Nash demand game as an example to prove this point.

The norm established implies a majority of the population follows that strategy or action, but there is still a minority population who follow something which is not a norm. We assume that once any norm is established, there is an incentive provided to move to alternative options by some external force. That external force could be in the form of a government or an entity that is trying to excite the population to move to other alternatives. That incentive could be in monetary or non-monetary terms. For example, we can assume two major telecom providers are in the country, and a new player is trying to enter the market. Most of the population sticks to the two dominant providers at any time. The new entrant incentivizes users to switch to their services to gain market share. And those incentives could be in the form of additional broadband services at no extra cost. There are also incentives involved if an agent forwards this information in their network and brings their network to use the new entrant services. Hence, there is some boost in payoff for the agents, which could be a notional gain too who are trying to switch to the new player. Another example could be when the Government is trying to push any agenda and wants the general public to follow that. For example, to promote women's empowerment in the country, Indian Government provides certain savings and investment avenues for women providing higher risk-free returns from time to time. This translates to a shift in savings and investment options for the women population from existing avenues like bank account savings to more lucrative Government-backed bonds and securities. Hence, this can be interpreted as changes in payoff structure for the women population evaluating different savings and investment avenues with higher payoffs for Government-backed bonds and securities.

With this background, we assume that once any strategy reaches the norm state, external force provides an incentive to use the alternative strategies. We assume this incentive is represented in

an additional agent payoff. To keep things simple, we assume the additional delta payoff value as a constant value of 20. We consider the delta payoff value as 20 to keep the agents indifferent between choosing M and L strategy. This 20-payoff value is the difference between M and L strategy payoffs. The results from the Nash demand game in the last section showed M as the norm most times, and we want to assess if the additional delta payoff breaks that trend.

In this case, we ran iterations for 100 time periods to check the transition from one norm to another instead of 50 time periods in the previous two sections. We continue to assume other parameters as in the Nash demand game without fixed agents. However, we have one additional parameter to choose here: the minimum time period. This minimum period is the minimum duration for which the game should be played before attempting to add the delta payoff. We have set the minimum time period as 20. Once the game is started, it will run for a minimum of 20 time periods. From the 21st period onwards, it will check if any strategy satisfies the norm criteria laid out as shown in the norm definition section. If it is true, then the delta payoff is added to the rest of the strategies payoffs that do not satisfy the norm criteria. The delta payoff 20 is added for all the strategies that do not satisfy the norm criteria. Once the delta payoff is added, agents will consider the revised payoff matrix in future periods to decide what strategy to follow. For example, say at period 35, strategy M satisfies the norm criteria laid out. Table 4.2 shows the agent's payoff values from the start of the game till period 35.

Table 4.2. Agent's payoffs till period 35 (Nash demand game)

	H	M	L
H	0	0	70
M	0	50	50
L	30	30	30

The payoff matrix in Table 4.2 changes to the below payoff matrix in Table 4.3 from period 36 onwards. We have added a payoff value of 20 for H and L-type agents while keeping the payoff of M-type agents the same as in Table 4.2. This incentivizes agents to follow H and L strategy more compared to M.

Table 4.3. Agent's payoffs from time period 36 till time period 70 (Nash demand game)

	H	M	L
H	20	20	90
M	0	50	50
L	50	50	50

Agents would continue to use Table 4.3 payoffs as long as no other strategy satisfies the norm criteria. For example, at the time period 70, the L strategy satisfies the norm criteria. In that case, the payoff matrix would transform from Table 4.3 to Table 4.4 as below:

Table 4.4 Agent's payoffs from time period 71 (Nash demand game)

	H	M	L
H	20	20	90
M	20	70	70
L	30	30	30

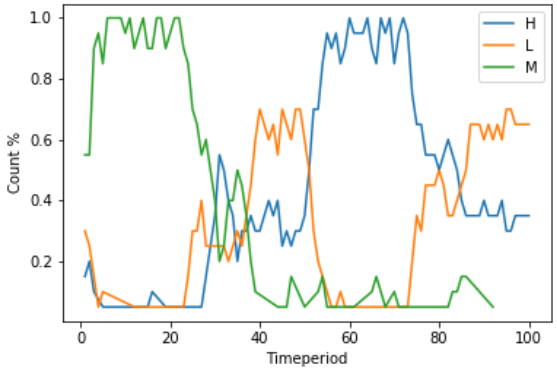
In Table 4.4 we added the delta payoff of 20 to H and M type agents and kept the payoff of L-type agents as in Table 4.2. This is to be noted that we have added the delta payoff to the base payoffs as listed in Table 4.2 and not Table 4.3. We assume that the payoffs listed in Table 4.3 or Table 4.4 are transitory and are available for a certain duration until a new norm displaces the old norm.

Agents would continue to use the payoff matrix of Table 4.4 till the time a new strategy satisfies the norm criteria. And this process goes on for 100 periods. We have set the minimum time period as 20, meaning changes to the payoff matrix, like from Table 4.2 to Table 4.3 or subsequent, if any, would happen only after the game is played for at least 20 periods. This, in turn, implies we want at least 70% of agents (14 out of 20) to use the specific strategy for at least 10 time periods before making any changes to the payoff matrix according to the norm criteria which we have defined. We can set this minimum time period cut-off to a high value if we want a specific strategy to be played for sufficiently longer time periods.

Results vary with respect to network structure. Ring network results show delta payoff of 20 is not sufficient for agents to move from M to any other strategy. In the case of a small-world network, a shift is happening from M to other strategies. Figure 4.18 shows the convergence in the case of

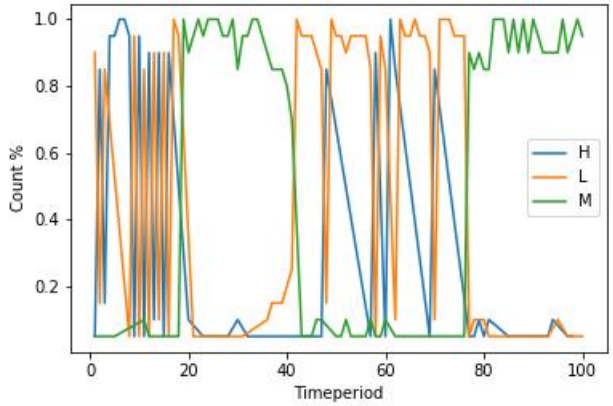
the SW3 network with the initial state of ['M', 'M', 'H', 'M', 'L', 'M', 'H', 'M', 'H', 'M', 'H', 'M', 'H', 'H', 'L', 'L', 'L', 'H', 'H', 'M'] and p_{edge} value of 25%. The M strategy is being replaced by H and L strategy over time.

Fig 4.18. Norm displacement in SW3 network (Nash demand game)



In the case of a complete network, a constant oscillation is observed between H, M and L strategies. Figure 4.19 shows the complete network results with the initial state of ['M', 'L', 'H', 'M', 'M', 'H', 'M', 'L', 'H', 'H', 'M', 'M', 'M', 'L', 'M', 'H', 'L', 'H', 'H', 'H']. We observed a shift from M to H strategy in the case of 2d grid network.

Fig 4.19. Norm displacement in complete network (Nash demand game)



To summarize, results show that the speed and pace of equitable norm displacement depend upon additional incentives provided to shift towards other non-equitable strategies along with agents' network structure. The small world network structure is more adaptive to changes.

4.7 Random networks and norm evolution

The results from previous sections show that norm evolution depends on the network structure. In this section, we looked at two random networks, Erdos-Renyi (ER) and Barabasi-Albert (BA) and assessed its network properties. Specifically, we take a look at network density, diameter, clustering coefficient, network's fat-tailedness and evaluate how these correlates with norm evolution. Network density is 1 for complete graph implying all possible edges are present, while 0 indicates completely disconnected nodes showing graph with no edges. Network diameter can take a value from 1 to infinity. The diameter of a network is the maximum shortest distance between any pair of nodes in the network. It represents the longest chain one would need to traverse to get from one node to another in the worst-case scenario. Social networks with a small diameter are more connected, allowing faster dissemination of news, ideas, or trends. A diameter of 1 occurs in a complete graph (all nodes directly connected to each other). The clustering coefficient in a network measures the degree to which nodes tend to cluster together. It is of two types, local clustering coefficient and global clustering coefficient. We considered the global clustering coefficient for our analysis. The global clustering coefficient provides an overall indication of clustering in the entire network. It ranges from 0 to 1. A value of 0 indicates that nodes are not connected in clusters while a value of 1 means that every node is part of a tightly knit group. To compute fat-tailedness of the network, we considered power-law exponent. If the exponent value

is < 2, it is considered heavy fat-tailed, if its value is between 2 and 3, it is moderate or medium fat-tailed, and a value of > 3 show low fat-tailed. In simple terms, heavy fat-tailedness implies networks have few highly connected nodes or hubs which significantly impact the overall network structure. For example, a social network where a few individuals have many connections, or internet with popular websites.

We generate different random networks and assess under what network structures or properties norm evolution is more or less likely to happen. We present these results using the Naming game as an example. For each result, we would show three sets of graphs, first the network initial state, second network end state by the end of 50 time periods and then the frequency distribution in percentages of different names proposed during the simulation run as a line chart. We continue to assume total agents count as 20 with no fixed agent.

We start with ER network. The input parameter to generate ER network is probability parameter (p) which represents the probability for edge creation. We start with a significantly high probability value of 0.9. Below figures 4.20 till 4.22 show one such result.

Fig 4.20. ER network initial state with p = 0.9

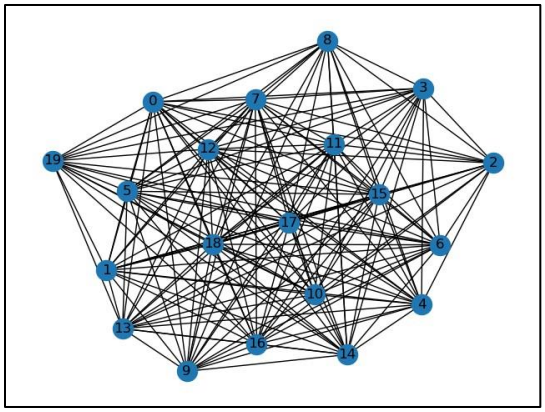


Fig 4.21. ER network end state with p = 0.9

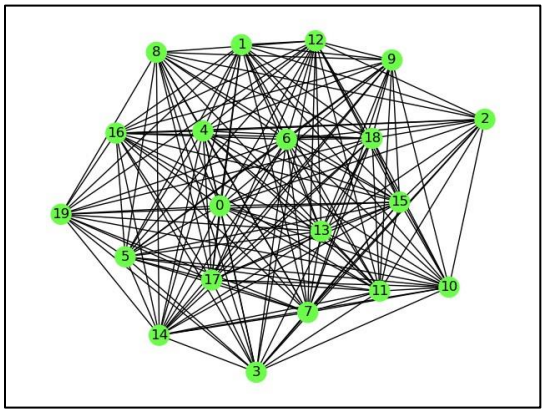
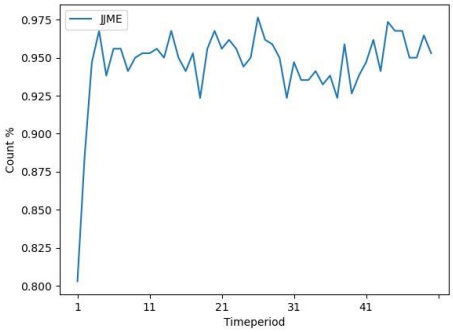


Fig 4.22. ER network norm evolution with p = 0.9



With a significant high probability for edge creation, the network looks like a complete network. This leads to agents following a single name predominantly and satisfying the norm criteria. Below figures show one such result when we lower the probability value to 0.17.

Fig 4.23. ER network initial state with p = 0.17

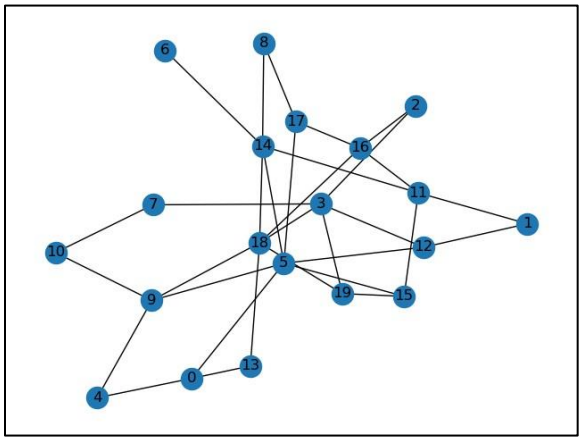


Fig 4.24. ER network end state with p = 0.17

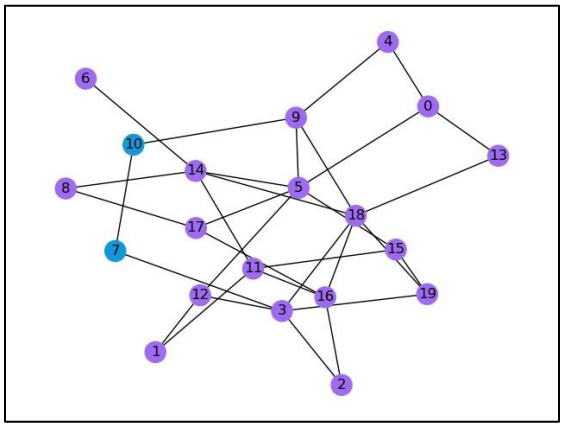


Fig 4.25. ER network norm evolution with p = 0.17

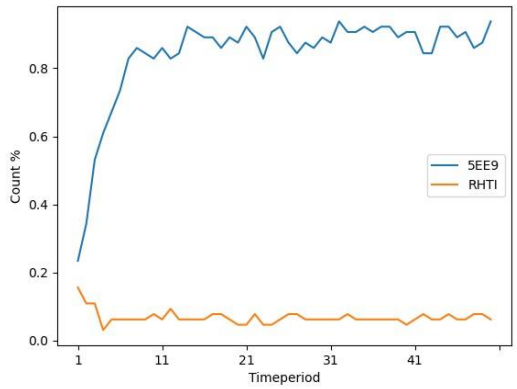


Figure 4.23 demonstrates that the p value of 0.17 is sufficient enough to generate a connected network. Figure 4.25 shows one of the results where the name proposed is satisfying the norm criteria. If we reduce the probability value even further to 0.1 this most likely leads to disconnected network. With this we don't have any name satisfying the norm criteria.

Fig 4.26. ER network initial state with p = 0.1

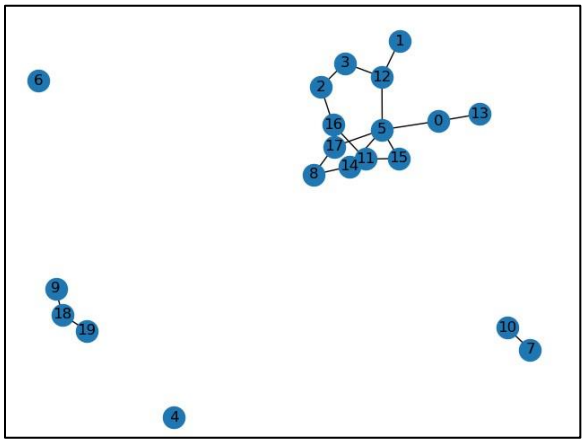


Fig 4.27. ER network end state with p = 0.1

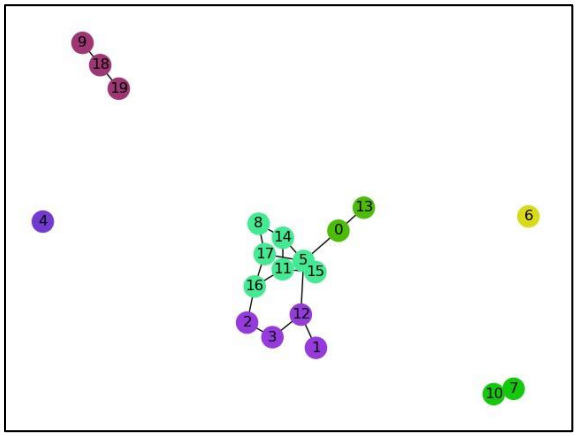
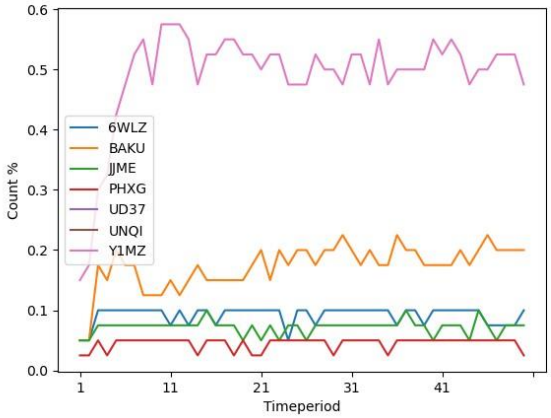


Fig 4.28. ER network norm evolution with p = 0.1



Figures 4.26 and 4.27 represent disconnected networks which get generated with lower probability value for edge creation. There are multiple names being proposed by agents and no name satisfying the norm criteria of 70%.

We have seen that higher probability of edge creation translates to higher network density. A higher dense network has higher chances of norm emergence and vice-versa. In the case of ER network, we observed density and clustering coefficient are mostly in the same direction, implying higher density networks also has higher clustering coefficients and vice-versa. And network density is directly related to probability for edge creation. A higher probability value is related to higher network density and vice versa. We also observed that lower diameter value is associated with higher density and clustering coefficient. And higher diameter values are associated with lower density and clustering coefficient values. Hence, the impact of lower and higher diameter values is similar to higher and lower density values respectively.

A heavy fat-tailed network can help emergence of norm despite its lower density. Figures 4.29 till 4.31 show one such example which has density of 0.14 and clustering coefficient of 0.23. Figure 4.31 shows one name which is closer to satisfying the norm threshold criteria.

Fig 4.29. ER network initial state with p = 0.13 and heavy fat-tailed network

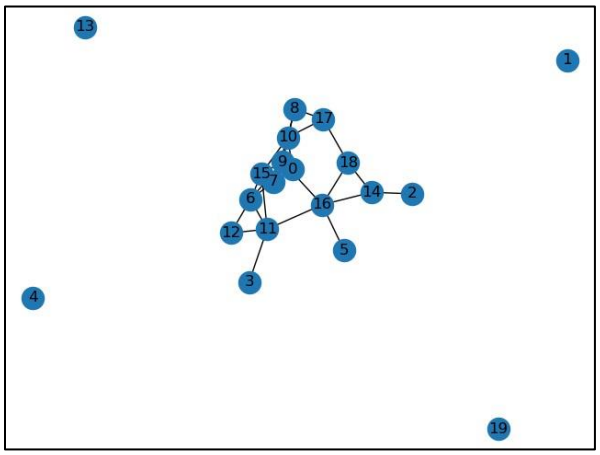


Fig 4.30. ER network end state with p = 0.13 and heavy fat-tailed network

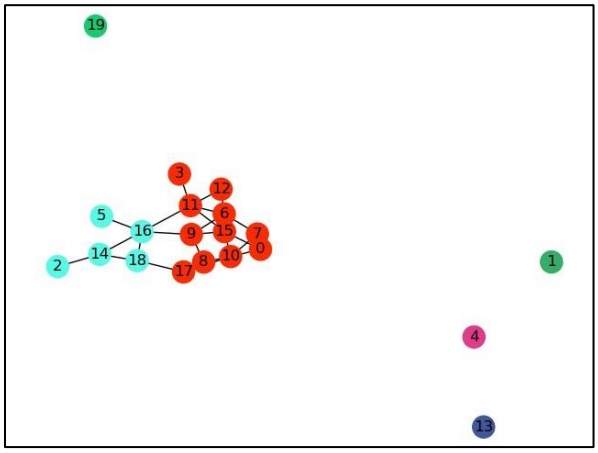
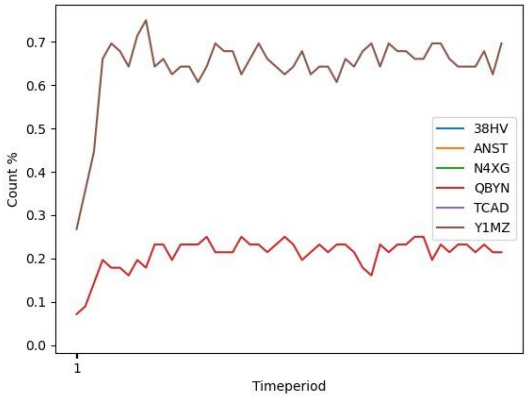


Fig 4.31. ER network norm evolution with p = 0.13 and heavy fat-tailed network



Next, we analyzed BA random network. It has two parameters, n and m. n represents the number of nodes and m denotes the number of edges to attach from a new node to existing nodes. In this network, an initial base network is created, and new edges are preferentially attached to nodes with a higher number of edges. In our case n equals 20 which is the total number of agents with each

agent representing one node. The initial network can be specified as any network which should be connected. We have considered two different scenarios, where in one case we have star network as the initial network and in another we have taken ER random network as the initial network. In the case of star network, we change m parameter and observe results. In the case of ER random network as initial network, we change two parameters, m and p and observe the results.

Fig 4.32. BA network initial state with m = 2 and star network as base

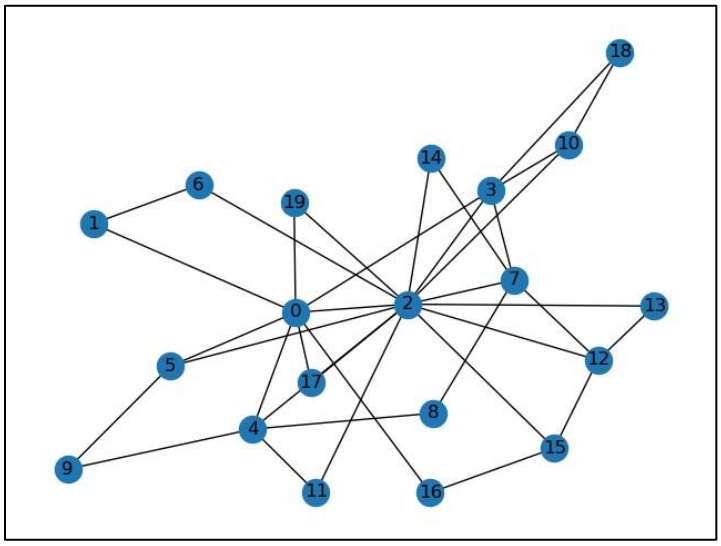


Fig 4.33. BA network end state with m = 2 and star network as base

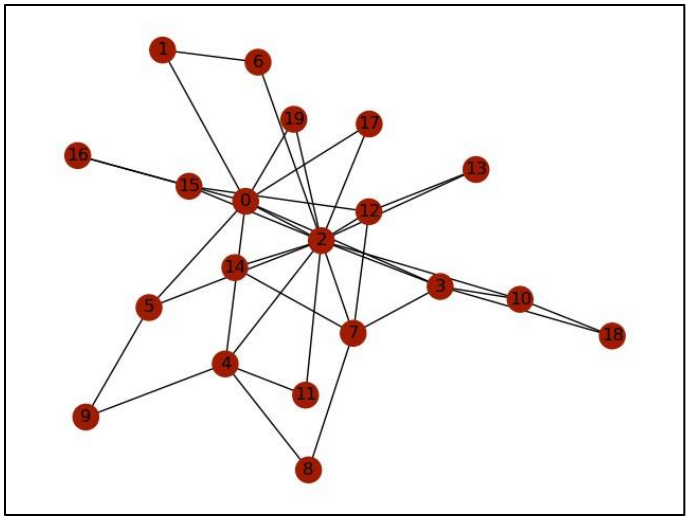
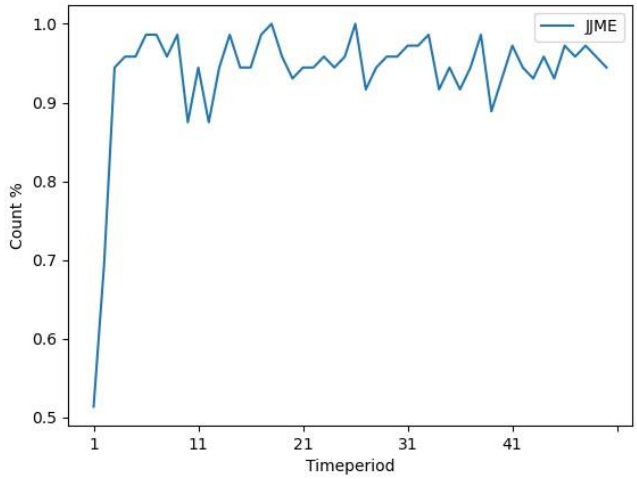


Fig 4.34. BA network norm evolution with m = 2 and star network as base



Figures 4.32 till 4.34 show BA random network with star network as the base network and 2 number of edges. The star network is created with m+1 = 3 nodes which serve as the initial

network. With these network configurations, the norm criteria are getting satisfied. Below 3 figures show the graphs with higher number of edges (m=18).

Fig 4.35. BA network initial state with m = 18 and star network as base

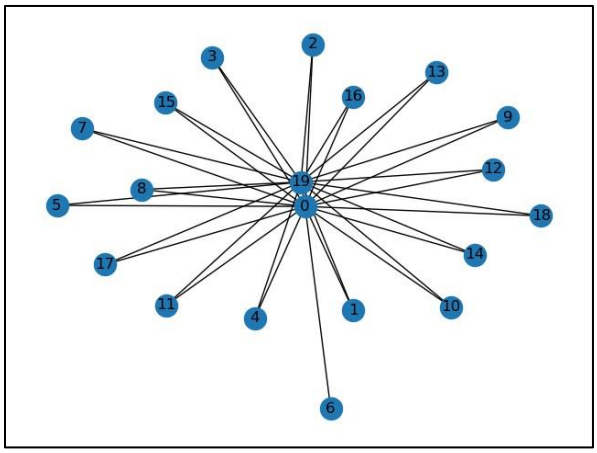


Fig 4.36. BA network end state with m = 18 and star network as base

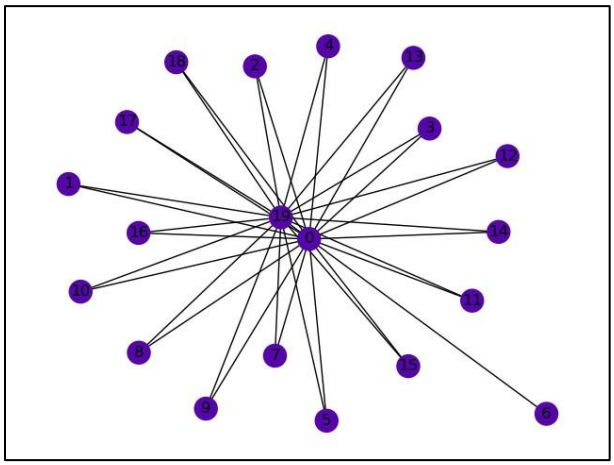
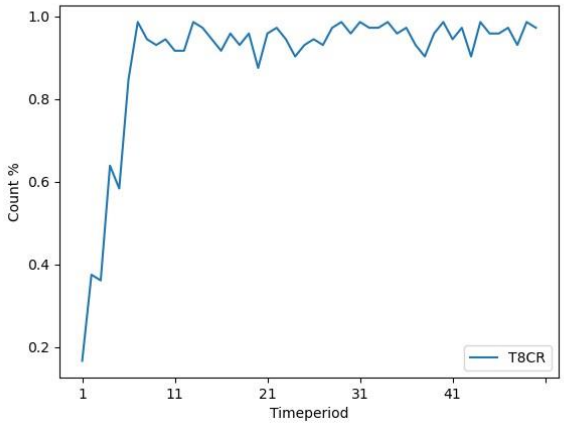


Fig 4.37. BA network norm evolution with m = 18 and star network as base



BA network with higher count of edges (18) and star network as the base initial network also being shown to satisfy the norm criteria. Now, we replaced the star network with ER random network as the base network to assess the impact of base network on the norm evolution. The ER network is created using m+1 nodes. Figures 4.38 till 4.40 consider the ER network with a p value of 0.84 and m value of 1. This network has a density value of 0.1 and clustering coefficient of 0. There is no name satisfying the norm criteria.

Fig 4.38. BA network initial state with m = 1 and ER network (p=0.84) as base

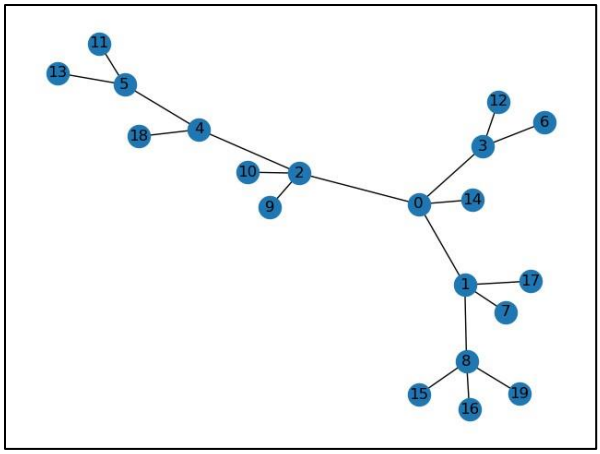


Fig 4.39. BA network end state with m = 1 and ER network (p=0.84) as base

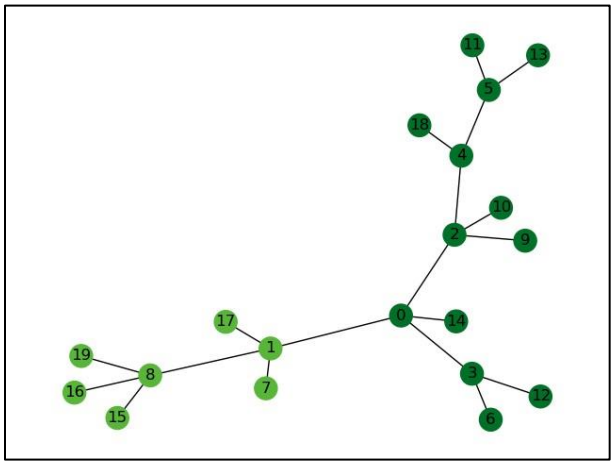
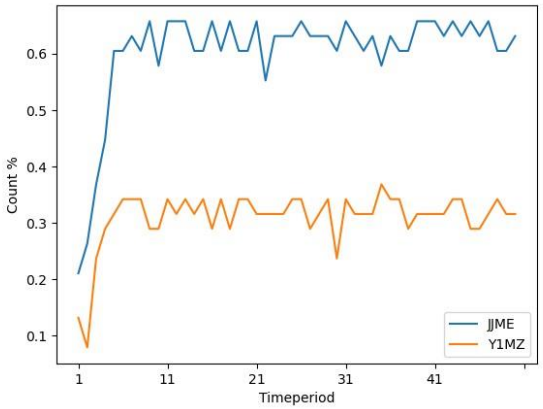


Fig 4.40. BA network norm evolution with m = 1 and ER network (p=0.84) as base



Another example below where the clustering coefficient is high along with number of edges, this leads to norm evolution. Figure 4.41 network has density of 0.19 and clustering coefficient of 0.39.

Fig 4.41. BA network initial state with m = 2 and ER network (p=0.64) as base

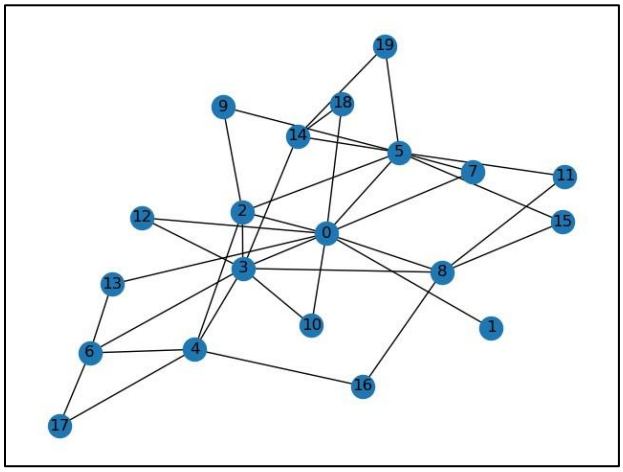


Fig 4.42. BA network end state with m = 2 and ER network (p=0.64) as base

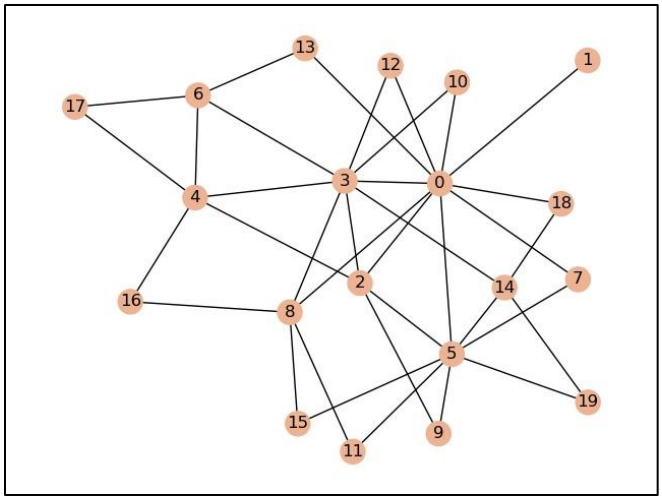
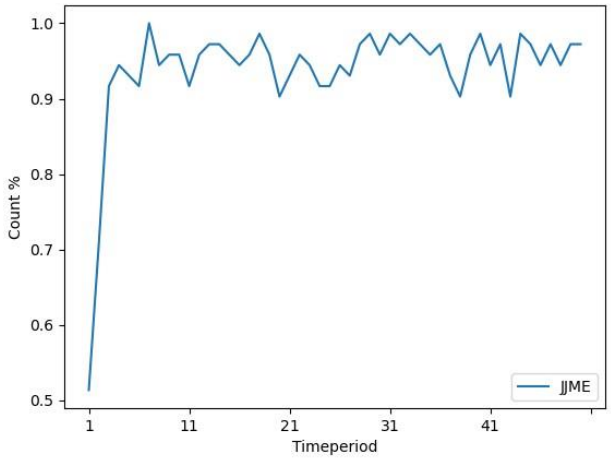


Fig 4.43. BA network norm evolution with m = 2 and ER network (p=0.64) as base



Above iterations with random network considers the same number of edges (m) value for both the initial base random network (ER) and the outcome random network (BA). Now, we relax this and make the two different. We assume that the base network ER starts with m1 nodes, and it is expanded by adding m2 edges in the BA random network. Figures 4.44 till 4.46 depict one such

possibility. The network in Figure 4.44 has a density of 0.17 and clustering coefficient of 0.2 and it satisfies the norm criteria.

Fig 4.44. BA network initial state with m1 = 8, m2= 2 and ER network (p=0.31) as base

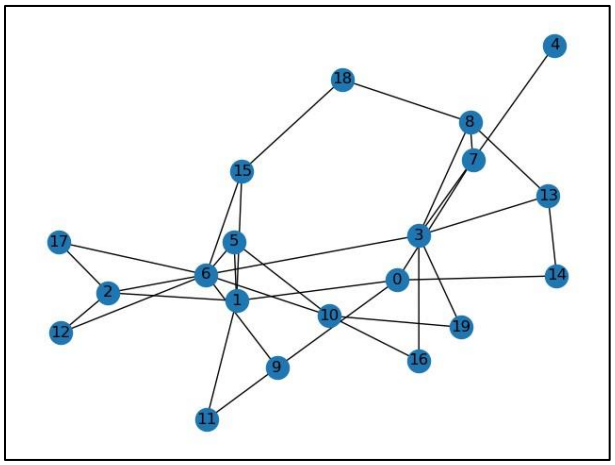


Fig 4.45. BA network end state with m1 = 8, m2= 2 and ER network (p=0.31) as base

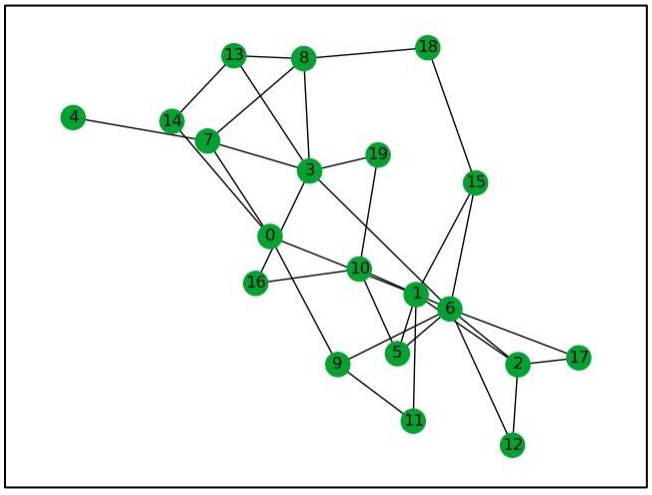
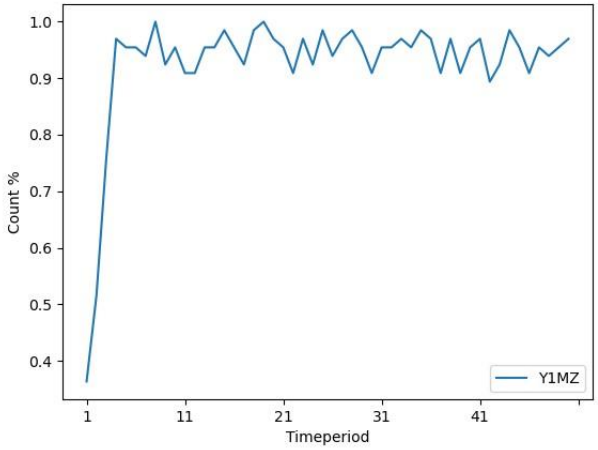


Fig 4.46. BA network norm evolution with m1 = 8, m2= 2 and ER network (p=0.31) as base



Below is one of the examples of where norm criteria are not being satisfied when m1 and m2 are different. Figure 4.47 network has density of 0.11, clustering coefficient of 0.02 and heavy fat-tailedness.

Fig 4.47. BA network initial state with m1 = 3, m2= 1 and ER network (p=0.81) as base

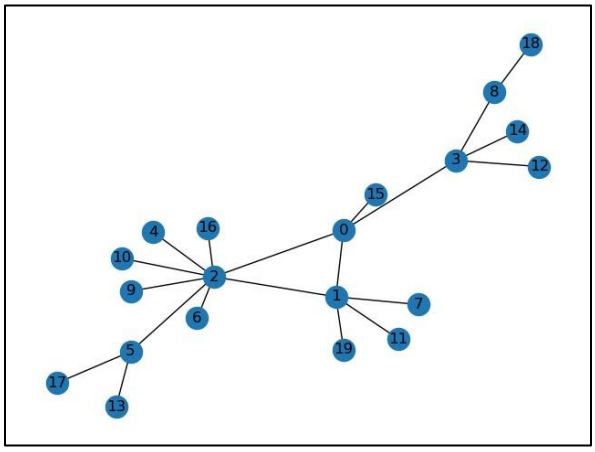


Fig 4.48. BA network end state with m1 = 3, m2= 1 and ER network (p=0.81) as base

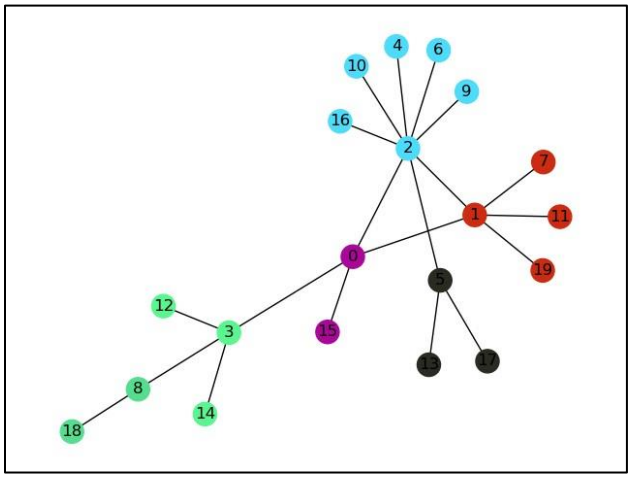
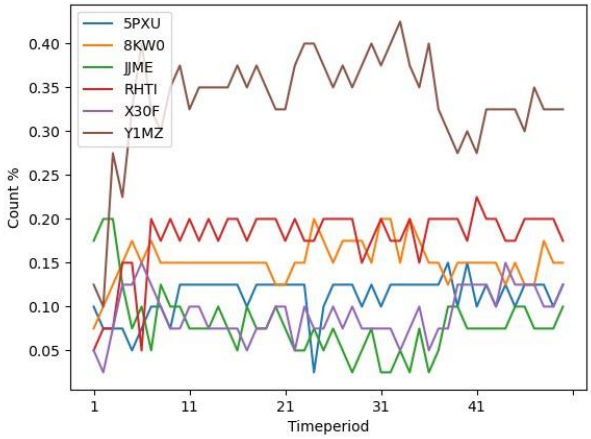


Fig 4.49. BA network norm evolution with m1 = 3, m2= 1 and ER network (p=0.81) as base



We can summarize the findings below.

- When the initial graph starts with star network having nodes as number of edges + 1 (m+1), the norm evolution happens generally.
- When the initial graph starts with ER random network, there are chances of names not satisfying the norm criteria in following scenarios.

- The number of edges count is low.
- The probability of adding edge is low.
- With lower number of edges and lower probability of edge, this generally translates to lower density.
- Clustering coefficient value is low.
- Medium to Heavy fat-tailedness coincides with a lower number of edges.
- Network diameter value is high.

4.8 A note on python libraries created

We have created two Python libraries associated with the two sets of games shown in the chapter, the Naming and Nash demand games. The results shown in the chapter are based on certain parameters, assumptions, and restrictions. This implies it need not necessarily explore all the potential insights which can be derived from these games. Also, we have taken these games as an example to set the stage for many strategic decision games where this computational framework can be applied.

We created a Python library, *multi-agent-coordination* , for the Naming game, which can be used to generalize results with any user-defined network and agent combinations. The function *network_simulations* have input parameters on the number of agents, neighbourhood size, network name, fixed agent ratio, probability of agents taking random response, and norm parameters. The output generated contains details on strategies being played and its frequency. It also has information on fixed agents' strategies and where they are placed in the network. The strategies

which satisfy the norm criteria are shown in the network graph by the end of the simulation period. A detailed explanation of expected parameters and output interpretation is provided in the project description section of the library homepage (https://pypi.org/project/multi-agent-coordination/). To the best of our knowledge, this is the first of its kind open-source library which can be used for this purpose.

We created a Python library, *multi-agent-decision*, for the Nash demand game, which can be used to generate results with any user-defined strategies and payoff combinations. The function *simulation_function_neighbors* has input parameters on the number of agents, neighbourhood size, number of strategies, initial agents' strategy distribution, response function, network structure, and norm parameters. The output of executing this function is the revised agents' strategy distribution by the end of the simulation period. It can also be seen if the trend is reversed during the simulation period where one strategy outweighs the other during the period. The library can also explore the possibility of adding delta payoff and fixed agents. A detailed explanation of the required parameters and the output interpretation is provided on the library homepage (https://pypi.org/project/multi-agent-decision/). The functionality of this library is somewhat similar to the existing library *DyPy* (Nande et al.,2020), but the main difference lies in the approach taken for norm evolution. We have used the bottom-up approach of norm creation where agents interact with other agents within the defined neighbourhood and decide what action to take, but the *DyPy* library requires specifying dynamics (one of Moran, Wright-Fisher or Replicator) and takes the aggregative approach of norm creation. Also, the *DyPy* library does not consider the agents' social network information. There is a somewhat similar library to DyPy, named

EvolutionaryGames (Gebele & Staudacher, 2022), but it is written in R language, which also considers the evolutionary stable strategy and evolutionary stable set approach of evolution.

The intent of creating these libraries is to test different permutations of parameter combinations which can create several unique insights. These libraries can also serve as a base to expand further and make it more robust in terms of incorporating more complexities around different response functions, network structures etc. We believe that these libraries can be useful in numerous evolution applications under multi-agent framework which will add value to researchers working in social science, anthropology, computer science etc.

However, there are some limitations to this study. We have assumed payoffs are constant across the simulation period, which can vary depending on the strength of the agent's relations with other agents. We assume all agents as equal, but, in reality, we value some peoples' opinions more than others. The four possibilities we have considered for the response function can be further tweaked to incorporate this aspect of the agent's preferences. Naming game results assume full memory length, which can be changed to limited memory. The libraries created support four social network structures which can be further extended to incorporate more complex social network types and their associated parameters.

4.9 Conclusion

The decisions or choices taken in real life are impacted by the environment in which opinions are formulated and transformed. People are generally bounded rational and rely on heuristics and their

acquaintances' opinions to act. We have shown this computationally with the help of the Nash demand and Naming games, which can be further extended to other custom-defined games and payoffs. Any strategy potentially emerging as a norm depends upon agents' network, their belief system, and how frequently they transform their beliefs. We have seen higher chances of norms emergence in case of denser networks. However, if the network is heavy fat-tailed, it can compensate for lower network density. A lower network diameter and higher clustering coefficient correlates positively with norm emergence. We have also shown the possibility of local concentration of norms under certain parameter restrictions, which may not generalize well to the norms at the global level. Over a period, norms established can be displaced by other strategies, which, in turn, leads to the emergence of new norms. We believe that the Python libraries created can immensely help researchers interested in the computational aspects of evolution.

Chapter 5: Dynamic Evolution and Networks

5.1 Introduction

Classical game theory assumes agents as perfectly rational. Some examples that rational choice theory has some difficulty explaining include the evolution of cooperate outcome in Prisoner's dilemma, agents playing 'stag' in stag hunt game rather than risk-dominant outcome ('hunt'), equal pie in Nash bargaining game etc. Although the extensive form of the game and repeated game literature can capture some of the dynamics compared to normal form representation, it becomes unmanageable with more complex games. The action the agent will take is decided beforehand at every possible stage of the game. This representation does not consider what agents would learn during the game and how it responds to the newly available information about their opponents at each point of the game. Evolutionary game theory considers learning and dynamics (Alexander, 2021). This leads to the rise of game theory's evolutionary variants, which can explain how a particular outcome evolves over a period (Uyttendaele, 2015). Researchers have used evolutionary game theory to explain different aspects of human behavior, like explaining altruism, empathy, moral behaviour, social norms etc. This chapter focuses on explaining the evolution and sustenance of social norms.

Replicator dynamics is one of the first dynamics followed in dynamic evolutionary game theory (Taylor & Jonker, 1978). But other dynamics have been proposed by researchers over time like *Brown-Nash-von Neumann* or BNN (Brown & von Neumann, 1950), *Smith* (Smith, 1984), logit dynamic (Fudenberg & Levine, 1998) etc. These dynamics assume that decisions are taken at the

macro/aggregate level, which increases or decreases the percentage share of specific strategies or agents in the population. However, there can be a connection between agents making decisions individually at the micro level and the population dynamics. Sandholm (2010) lays out a framework where it is assumed that individuals make decisions based upon two things: first, the current state of the population, which means the percentage share of strategies followed in the population, and second the expected payoff which can result when agents follow these strategies given the current state of the population. An individual learning rule or *revision protocol* can be created which uses these two pieces of information as input and creates a matrix of *conditional switch rates*. These conditional switch rates provide the probability of switching from one strategy to another conditional upon the current population state and expected payoff.

Against this background, we have approached dynamic evolution from the micro perspective. We assume agents make decisions individually. These individual agents' decisions culminate in the evolution of specific strategies at the macro level, which can increase or decrease depending on how other agents in the population respond. We assume that agents are connected with other agents with a social network, and they interact with other agents and decide the optimal strategy to follow. Agents follow the best response function approach, continuing to follow the existing strategy if that results in a higher payoff. And if their neighbours' strategy brings in a higher payoff, they switch to that strategy in future time periods. We assume that the strategy played by a larger number of agents in the population and played for a longer duration is a candidate for a stable strategy or a norm in society. In replicator dynamics terminology, this implies agent types and shares in the population are defined by their actions or choices made by them. We compared the results from the best response function approach with results from the replicator and other

dynamics, which follows an aggregative or deterministic approach towards an increase or decrease in the agents' share in the population. The best response function approach considered in this paper is similar to the one used in the previous chapter's Nash demand game.

Assessing the difference between aggregate behavior from replicator and other dynamics with micro-level outcomes from agent-based modeling is important for various reasons. Some of these reasons include understanding how well these two approaches capture real-world dynamics. It is interesting to understand whether one model is more suitable for large-scale simulations and is more computationally efficient as compared to others. What insights do each model provide about individual behavior? Is there any scenario where agent -based model explains better human decision-making compared to evolutionary dynamics? We have seen in previous chapter that social network structures influence the game outcomes. It is interesting to gauge how evolutionary dynamics and agent-based models can incorporate the network information and compare results. The comparison also involves how robust these results are with respect to sensitivity analysis which implies how results change when we change different model parameters and initial states. In the presence of heterogeneous agents which have different tastes and preferences, the comparison is on how well dynamics-based models and agent-based models capture this information and its influence on norm evolution. Finally, which model can provide more actionable insights for policymakers to design policy interventions. We have tried to address some of these aspects in this chapter.

We assessed the dynamic evolution problem with the help of a migration game. Migration as a subject area within the game theory and microeconomics has been widely studied in literature

(Rocha, 2012; Özgönül & Kaplan, 2013; Barnett-Howell, 2017). We formulated a migration game that entails agents of certain types migrating from a domestic country to a foreign country where most agents follow certain norms X and Y, respectively. Agents' payoffs vary when domestic country agents interact with foreign country agents. We want to know which norm out of X and Y survives when domestic and foreign country agents coexist in the population. We used the dynamics-based methods and the agent-based best response function approach to answer this question. We also wanted to explore if the results from replicator dynamics hold when evaluated against the best response approach, where agents are allowed to decide what choices to make and are connected via social networks. We took the migration game as an example to prove the point, but this can be replicated in any symmetric finite normal-form game.

Results show that different social network types influence agents' decisions to follow domestic or foreign country norms. The replicator and other dynamics results show the convergence towards one outcome, either X or Y when one agent type is more social than another. But the best response function approach shows the possibility of the emergence of both the outcomes, X and Y, under some parameter restrictions.

The rest of the chapter is divided into six sections. The next section reiterates some key concepts of ESS and replicator dynamics shown in the chapter on literature review. The third section talks about the construct of the migration game and the four possibilities of the game we have considered. The best response function framework is explained with the help of an example in the following section. The fourth and fifth sections present the results of the four migration game possibilities using different dynamics and best response function approaches, respectively. The

sixth section provides detailed analysis of random networks. The chapter concludes by providing some tentative future areas of scope and work.

5.2 Static and dynamic evolution

Two approaches to evolutionary game theory have been discussed in literature. The first approach revolves around the evolutionary stable strategy (ESS) as the principal analysis tool. This approach can be considered static because it does not explicitly model the underlying process by which strategy changes. For example, assume that σ is the strategy that the population usually follows and μ is the invading (or new) strategy. For σ to maintain its stability, it should do well against μ. If other agents follow σ, and if a new agent is evaluating to follow σ or μ, the new agent will follow σ if the following holds.

$$\pi(\sigma \mid \sigma) >= \pi(\mu \mid \sigma)$$

where π(p1|p2) is payoff which agent receives when playing strategy p1 against opponent agent playing p2.

Suppose if $\pi(\sigma \mid \sigma) = \pi(\mu \mid \sigma)$, which means the new agent is indifferent between σ and μ, then the following condition should hold for a new agent to have an incentive to choose σ

$$\pi(\sigma \mid \mu) > \pi(\mu \mid \mu)$$

The above condition implies that the payoff from following σ given others follow μ should be strictly higher than the payoff from following μ given others follow μ.

A strategy σ is called an evolutionary stable strategy (ESS) if it satisfies both these conditions. Over time, multiple variants of evolutionary tools developed like uniform invasion barrier, locally superior, evolutionary stable set, equilibrium evolutionary stable set, etc. This led to competing definitions of stability where in some cases a strategy can satisfy the stability criteria laid out by *locally superior* and *equilibrium evolutionary stable set* but do not satisfy the criteria for ESS. Therefore, evolutionary game theory encounters similar selection problems that traditional game theory encountered over the years. Due to this, the focus of research shifted towards the second approach, a dynamic approach to evolution.

In the dynamic approach, an explicit model showing the process around dynamics is constructed, which shows how the frequency of strategies changes. And it further studies the properties of dynamics. Some interesting results emerged from literature focusing on dynamic approaches to evolution. For example, Harms and Skyrms (2008) show the possibility of a 'cooperate' outcome in a prisoner's dilemma game, an equal share in a symmetric bargaining game, cooperate outcome ('stag') in a stag hunt game rather than risk-dominant outcome ('hunt') etc. Hofbauer and Sandholm (2011) opine on the possibility of agents playing strictly dominated strategies in the population.

The first tool used predominantly in dynamic evolution is replicator dynamics. Suppose a prisoner's dilemma game with two strategies: cooperate (C) and defect (D). W_C denotes the

expected fitness of cooperate strategy agents, while W_D represents the same for defect strategy agents. p_C and p_D denote the percentage share of cooperators and defectors in the population in the current generation. $\overline{W}$ shows the average fitness of the entire population. We can write the expected fitness equation as follows:

$W_C = p_C * \pi(C|C) + p_D * \pi(C|D)$

$W_D = p_C * \pi(D|C) + p_D * \pi(D|D)$

$\overline{W} = p_C * W_C + p_D * W_D$

Using the above formulation, we can construct the two equations below: replicator dynamics.

$$\frac{dpc}{dt} = \frac{pc * (Wc - \overline{W})}{\overline{W}}$$

$$\frac{dpD}{dt} = \frac{pD * (WD - \overline{W})}{\overline{W}}$$

where $\frac{dpc}{dt}$ and $\frac{dpD}{dt}$ represents a change in p_C and p_D with respect to time t, respectively. With these equations, we can compute the rate of change of increase /decrease in co-operators/defectors across various time points, which also depends upon the divergence between individual strategy fitness value and the average fitness value of the entire population.

Replicator dynamics is one of the important dynamics used in literature. Other dynamics, like the BNN dynamic, require agents to consider all alternatives available along with their payoff matrices so that new strategies can also enter the population if it is not already entered. A further refinement

of the BNN dynamic is the Smith dynamic. BNN dynamic does the comparison of alternate strategies' expected payoff with the average payoff of the population. However, the Smith dynamic does the comparison of the current strategy's expected payoff in the current population state with the respective expected payoff of other strategies in the present state. Hence, the individual payoff of strategies is compared with each other and not with the population average. The logit dynamic involves an additional parameter (η). As the name suggests, the logit dynamic involves taking the exponential of fitness value associated with individual strategy multiplied by η^{-1}. As η approaches 0, the probability of playing any strategy that is not the best response goes to zero.

Sandholm (2010) draws out a link between actions at the individual agent level and these being reflected at the macro level. It proposed the following equation to make inferences about the population-level dynamics as follows:

$$\frac{dpi}{dt} = \Delta_1 p_i - \Delta_2 p_i$$

The above equation shows the rate at which strategy p_i changes with time, and it depends upon the rate at which agents start using strategy p_i (Δ_1) subtracted from the rate at which agents stop using strategy p_i (Δ_2). From the above equation, we can determine the relationship between particular learning rules at the individual agent level and their impact at the population level. We leveraged the same thinking in formulating the best response function approach to dynamic evolution.

5.3 Migration game

Consider two regions, A and B. Majority of agents in region A follow some norm, say X, and most agents in region B follow norm Y. We assume that some agents from region A migrate to region B in search of better employment and quality of life. Agents' payoffs vary when domestic country agents (following X norms) interact with foreign country agents (following Y norms). We are interested in knowing which norm out of X and Y survives when both domestic and foreign country agents coexist in the population, which requires interaction. Table 5.1 below shows X and Y agents' payoffs when interacting. We assume the game to be symmetric.

When X agents interact with other X agents, each gets a payoff of 1. Similar is the case when Y agents interact with other Y agents. We normalize the payoff values on a scale of 1. We assume the base payoff value as 1. When agents interact with others, they get the payoff value in a multiple of this base payoff value. We assume this multiple varies between agents when agents interact with the same type of agents or with other types of agents. This multiple can equal 1, implying they get the same payoff value as the base payoff value. If this multiple value is lower than 1, agents get lower than the base payoff value. A value higher than 1 implies that agents get higher payoff values than base payoff values. We assume agents to be more social when they like interacting with opposite agent types and vice versa. This results in the following four possibilities: Assuming X agents to be more self-centered or less social translates to X agents getting lower payoff values when interacting with Y agent types than X agent types. If Y agents are more outgoing and social, it is assumed that it results in a higher payoff for Y agents while interacting with X agent types compared to interacting with its own types. The rationale for these payoffs is the assumption that by being more social, one enjoys interacting with other agent types and hence fetches higher rewards. This translates to the below payoff matrix for X and Y agents.

Table 5.1 Migration game. X agents less social compared to Y agents

	X	Y
X	1	0.9
Y	1.1	1

The analytical solution for Table 5.1 is (Y, Y). The second possibility arises when we assume X agents are more social and open to other agents and Y agents are relatively less social (Table 5.2).

Table 5.2 Migration game. X agents more social compared to Y agents

	X	Y
X	1	1.1
Y	0.9	1

The analytical solution for Table 5.2 is (X, X). Table 5.3 below shows the payoff matrix when both X and Y agents are more social.

Table 5.3 Migration game. Both X and Y agents are more social

	X	Y
X	1	1.1
Y	1.1	1

Table 5.3 has 2 pure strategy Nash equilibrium, (X, Y), (Y, X) and one mixed strategy Nash equilibrium, ((0.5,0.5), (0.5,0.5)). And lastly, the case when both X and Y agents are anti-social results in a payoff matrix of Table 5.4.

Table 5.4 Migration game. Both X and Y agents are anti- social

	X	Y
X	1	0.9
Y	0.9	1

Similarly, Table 5.4 also has 2 pure strategy Nash equilibrium, (X, X), (Y, Y) and one mixed strategy Nash equilibrium, ((0.5,0.5),(0.5,0.5)).

In reality, the population comprises a combination of social and anti-social agents. And it is applicable for both agents, X and Y. We assume that when agents interact with their own types, their payoffs remain the same at 1 irrespective of their degree of social behaviour. The next section explains the best response function framework in detail. The sections following that report result with respect to all four variants of the migration game shown in Tables 5.1 to 5.4 and referred to as Case 1 through Case 4, respectively.

5.4 Best response function framework

We assume agents play the game defined in Case 1 through Case 4 above. We assume agents are placed in a specific social network. These agents are connected with other agents via a network edge. X and Y are two distinct strategies or actions which agents can follow.

We have assumed four different network structures: ring network, small world network, complete network, and 2-dimensional (2d) grid network. A ring network can be considered a circular network where each agent is connected with some agents towards the left and right. The small world network has a 'shortcut' available which implies agents can directly move from one node to another node without the involvement of other agents in the transit. There are three different forms of small-world networks, and we call them SW1, SW2, and SW3. Small world network 1 (SW1) is the Watts-Strogatz network where the total number of edges remains constant as in the ring network. However, in the case of adding short-cut edges into the network, the existing edges are removed and replaced by short-cut edges. In the second small-world network (SW2), the existing edges are not removed when the short-cut edges are added, resulting in more edges than the other two small-world network variants. This network is called the Newman-Watts-Strogatz small-world network. The third network (SW3) has similar characteristics as SW1 with the additional property of network to be connected. The probability of rewiring existing edges or adding new edges (p_{edge}) is one of the parameters which impacts all three small world network structures. This parameter represents the probability of rewiring existing edges in the SW1 and SW3 networks, while it implies the probability of adding new edges in the SW2 network. The higher probability value indicates a larger presence of short-cut edges, while a lower probability value shows smaller short-cut edges. The complete network ensures that each node is connected with every other node.

A Lattice or grid network is a spatial network where each node represents agents' spatial position and is placed in a grid-like structure. Specifically, we consider the 2-dimensional grid network.

We assume the population size is 20, with each agent connected with 2 other agents. In the case of the SW2 network structure, each agent can be connected with more than 2 agents depending upon the value of p_{edge} parameter. We ran the simulations for 50 time periods. Agents start the game with actions specified by the initial state in period 1. The initial state specifies the agents' distribution of strategies out of X and Y. We assume that X and Y agents have a 50% share of the population. This can be represented in different ways, e.g. [X,X,X,X,X,X,X,X,X,X,Y,Y,Y,Y,Y,Y,Y,Y,Y,Y] or [X,X,X,X,X,X,X,Y,X,X,X,Y,Y,Y,Y,Y,Y,X,Y,Y,Y] etc. This list representation indicates that the first agent follows X, the second agent follows X, and so on for the rest of the 18 agents. Agents calculate their payoffs according to the actions followed by them and their network agents. Suppose the initial state is [X,X,X,X,X,X,X,X,X,X,Y,Y,Y,Y,Y,Y,Y,Y,Y,Y], and they play the game as specified in Table 1. For the first agent, the neighbour towards the right (second agent from left) follows X, and the neighbour towards the left (last agent from the right) follows Y. The payoff of the first agent equals 1 (due to X) + 0.9 (due to Y) = 1.9. For the second agent (from the left), both neighbours follow X, so the payoff equals 1+1 = 2. The last agent has one neighbour following X and one neighbour following Y, resulting in 1 + 1.1 = 2.1 payoff. Similarly, the rest of the agents compute their payoffs. At the end of period 1, agents evaluate if they continue to follow the same strategy in period 2 or if they should change. Agents switch to their neighbours' strategy if that increases payoff. If payoffs are the same with following the existing strategy versus their neighbours' strategy, then agents continue to follow their existing strategy. The first agent's

payoff is 1.9, while the right neighbour agent has a payoff of 2 and the left neighbour agent has a payoff of 2.1. Hence, the first agent decides to follow the left neighbour agent strategy Y in period 2, which has a higher payoff. Similarly, the rest of the agents decide their optimum strategy for period 2, creating a new population state in period 2, which becomes an input for period 3. And the cycle continues for the rest of the periods. We assume that there is no randomness in agents' decision-making to maintain parity with the replicator and other dynamics and to have a more deterministic approach to the best response. This is to be noted that agents consider only the immediately preceding period state to decide the optimum strategy for the next period. We are interested in how the initial state distribution of agents, which contains 50% X and 50% Y agents, changes over time as agents interact with their neighbours and play the game repeatedly.

The best response framework described above is similar to what has been followed in the Nash demand game in the previous chapter but with the assumption of no random component in the agent's decision-making. In this chapter, we are refraining from defining the explicit norm criteria to primarily assess the similarities and differences in the outcomes from macro-based evolution methods like replicator dynamics and micro-based evolution methods like best response framework.

5.5 Results - Replicator and other dynamics

We report results from the replicator, BNN, Smith, and logit dynamic across all 4 cases of the migration game. The logit dynamic results are computed with four different values of the η parameter (0.01,0.3,0.7,0.99). The graphs below represent the rate of change of X with respect to

time t ($\frac{dX}{dt}$) on the Y axis and population share of strategy X on the X axis. The X-axis value can range from 0 to 1, which implies 0% X agents (or 100% Y agents) and 100% X agents (or 0% Y agents), respectively. The negative value of $\frac{dX}{dt}$ on the Y axis indicate negative growth or a decrease in X share (or an increase in Y share), while a positive value indicates an increase in X share (or decrease in Y share). The direction of the arrows in the graph represents the convergence trend. The arrows towards the right direction indicate an increase in X share in the population (or decrease in Y share), while arrows towards the left direction indicate a decrease in X share (or increase in Y share). To generate these graphs, we have leveraged the open-source R library called '*EvolutionaryGames*' (Gebele & Staudacher, 2022). To assess the stability of output from replicator dynamics, we have computed eigenvalues of the Jacobian matrix. The Jacobian matrix approximates the partial derivatives of the respective dynamics with respect to each state at the equilibrium point. The eigenvalues of this matrix provide information about the stability of the equilibrium point. If the real part of all the eigenvalues is negative, this shows the equilibrium point is stable. We report these eigenvalues at the respective places in the following text.

In Case 1, strategy X is strictly dominated by strategy Y. This is reflected in the results where most dynamic results show the extinction of X agents from the population. In the case of logit dynamic, results are dependent on the value of η parameter. As η value increases, which implies more randomness, the percentage share of X agents converges towards 50%. These results are shown in Figures 5.1 through 5.7. Results from the replicator, BNN, and Smith dynamic are the same, except the difference lies with the rate of change of decrease in X agents from the population. Logit results show that η value of 0.3 is sufficient to have an equitable representation of both X

and Y agents in the population, with higher values not resulting in any significant difference in the outcomes.

Figure 5.1 Replicator dynamic - Case 1

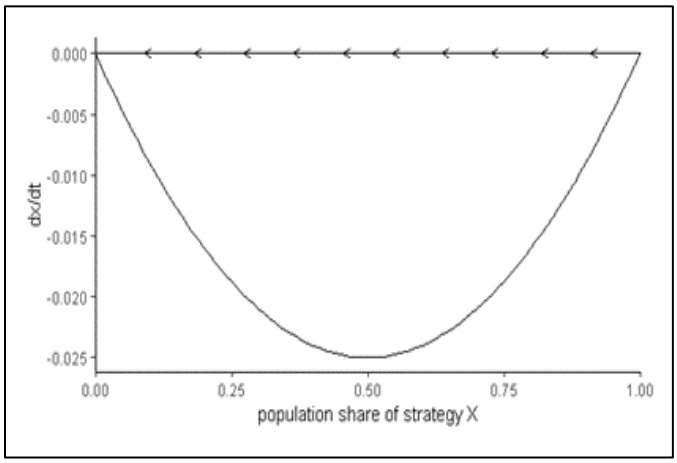


Figure 5.2: BNN dynamic - Case 1

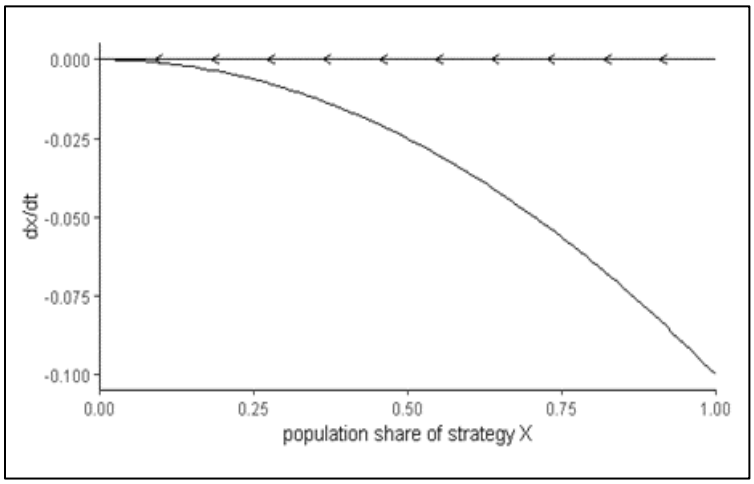


Figure 5.3: Smith dynamic - Case 1

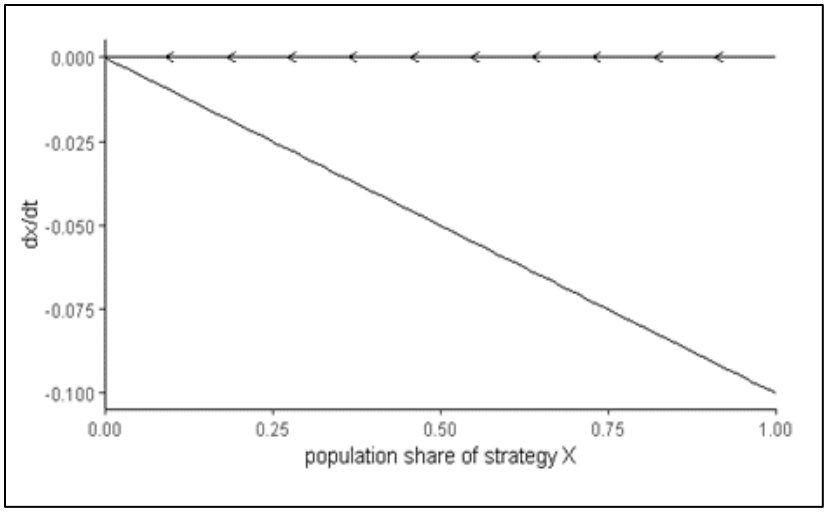


Figure 5.4: Logit dynamic (η = 0.01) - Case 1

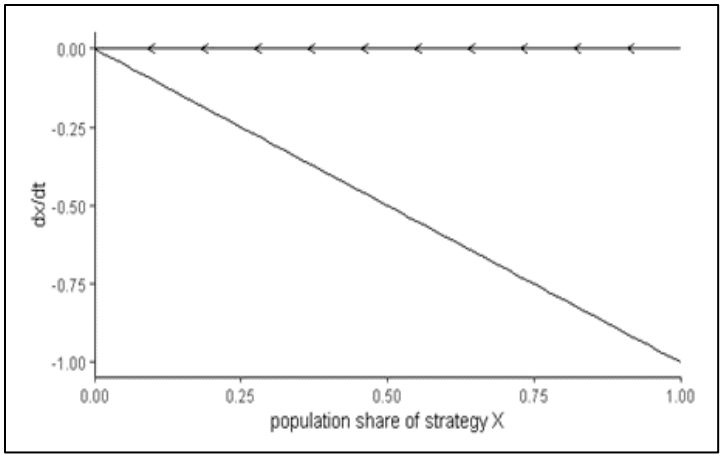


Figure 5.5: Logit dynamic (η = 0.3) - Case 1

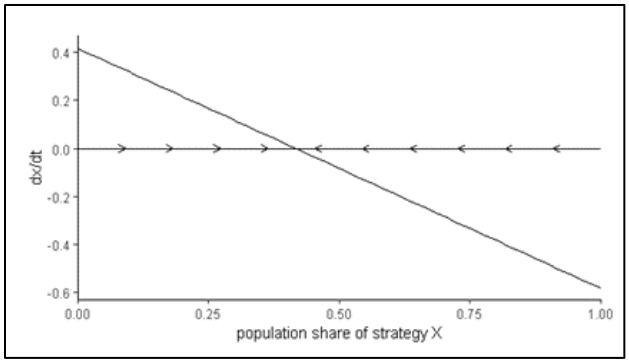


Figure 5.6: Logit dynamic (η = 0.7) - Case 1

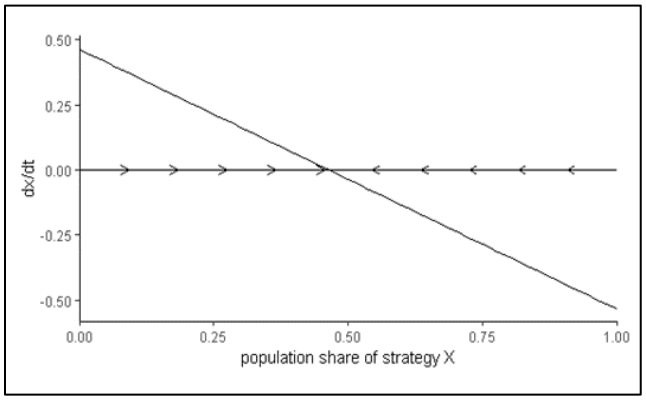
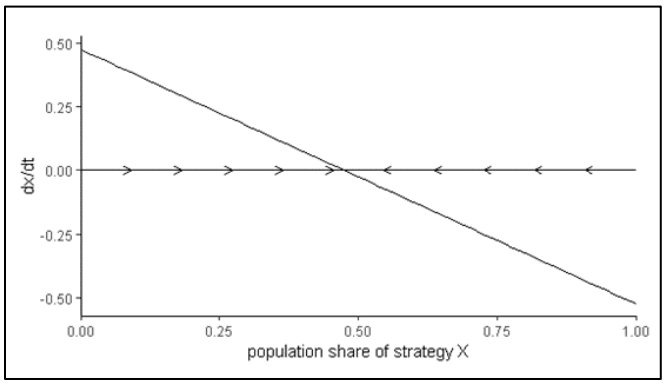


Figure 5.7: Logit dynamic (η = 0.99) - Case 1



The eigenvalues for outcome (0,1) implying 0% X agents and 100% Y agents are -1 and -0.10. For (0.5,0.5) outcome (50% X, 50% Y), eigenvalues are -1+0.000001i and -1-0.000001i. Since the real part of eigenvalues is negative, these outcomes are stable.

In Case 2, X agents are more social, and the X strategy strictly dominates the Y strategy. This is reflected in results where most dynamics results point towards an increase in the percentage share of X agents. An increase in η value (0.3 and above) in logit dynamic results in a deviation from

the optimum outcome, and the dynamics results settle at around 50% X share. These results are the exact opposite of what is achieved in Case 1. These results are shown in Figures 5.8 to 5.14.

Figure 5.8. Replicator dynamic - Case 2

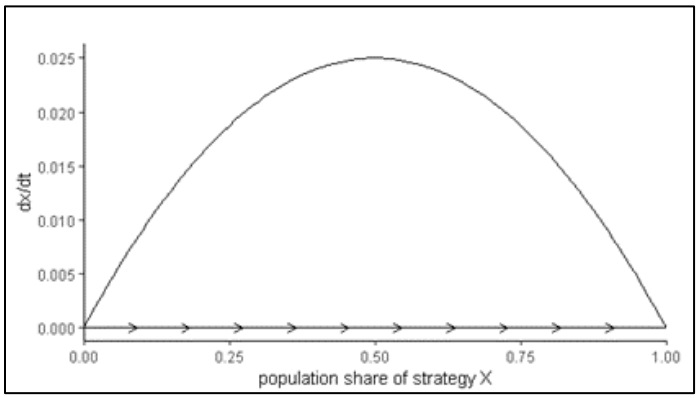


Figure 5.9: BNN dynamic -Case 2

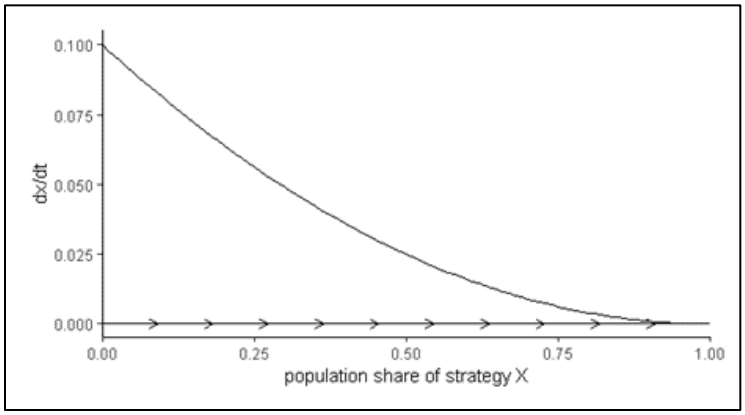


Figure 5.10: Smith dynamic – Case 2

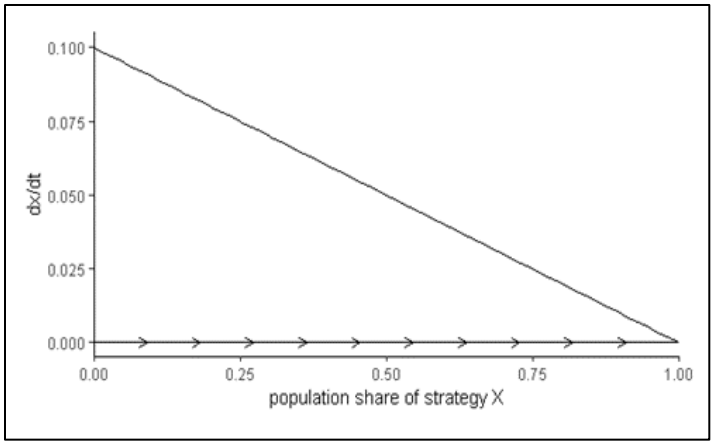


Figure 5.11: Logit dynamic (η = 0.01) – Case 2

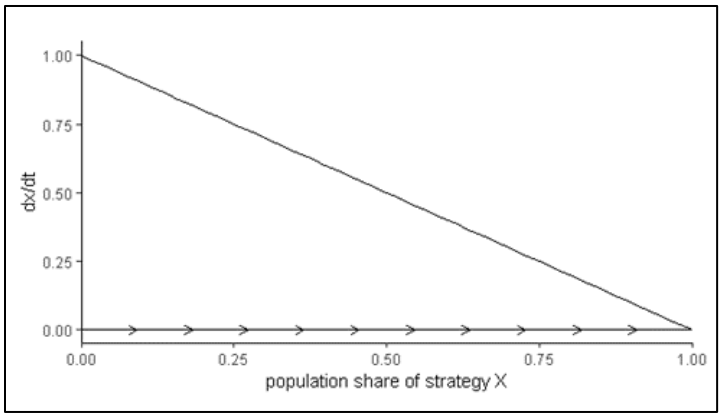


Figure 5.12: Logit dynamic (η = 0.3) – Case 2

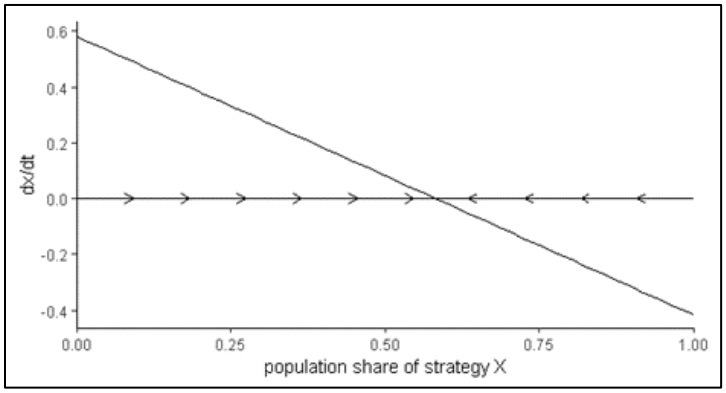


Figure 5.13: Logit dynamic (η = 0.7) - Case 2

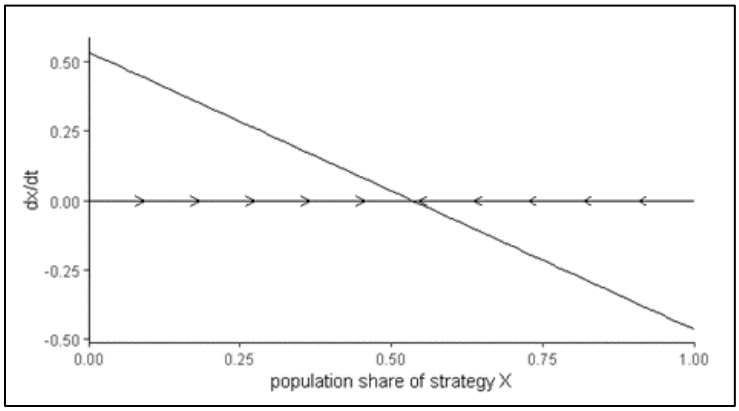
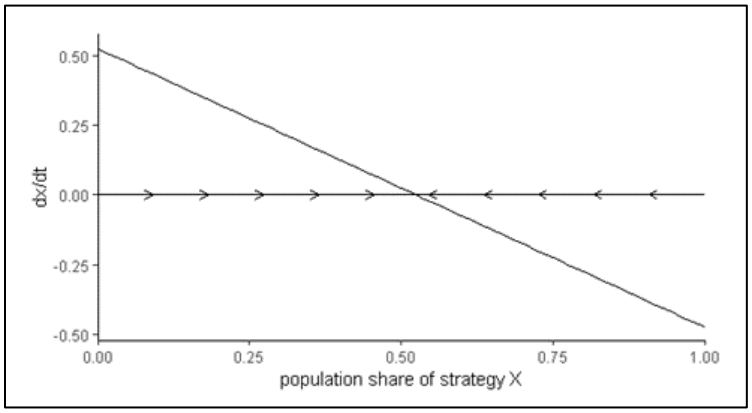


Figure 5.14: Logit dynamic (η = 0.99) - Case 2



The eigenvalues for outcome (1,0) implying 100% X agents and 0% Y agents are -1 and -0.10. For (0.5,0.5) outcome (50% X, 50% Y), eigenvalues are -1+0.000001i and -1-0.000001i. Since the real part of eigenvalues is negative, these outcomes are stable.

In Case 3, none of the strategies strictly/weakly dominates the other. All the dynamic results show convergence towards equal representation of X and Y agents in the population. The logit results also do not vary much with changes in the η parameter. Figure 5.15 shows the replicator dynamics

result from Case 3 payoff matrix. BNN and Smith's dynamic results show a similar trend (Figure 5.16, 5.17). Figures 5.18 and 5.19 show the logit dynamic result with η values of 0.01 and 0.3, respectively. The higher values of logit dynamic results show a similar graph trend as observed with η value of 0.3 (Figures 5.20, 5.21).

Figure 5.15: Replicator dynamic - Case 3

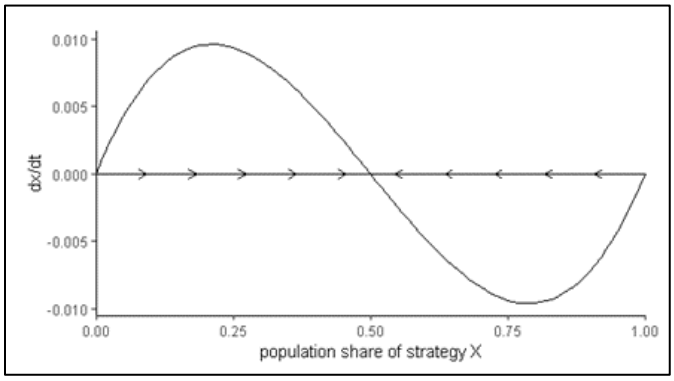


Figure 5.16: BNN dynamic - Case 3

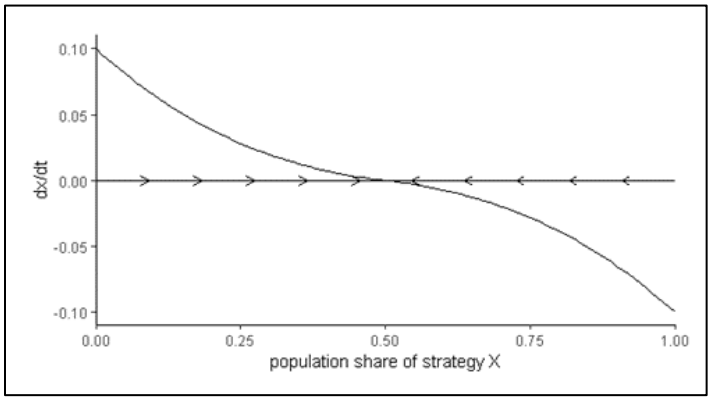


Figure 5.17: Smith dynamic – Case 3

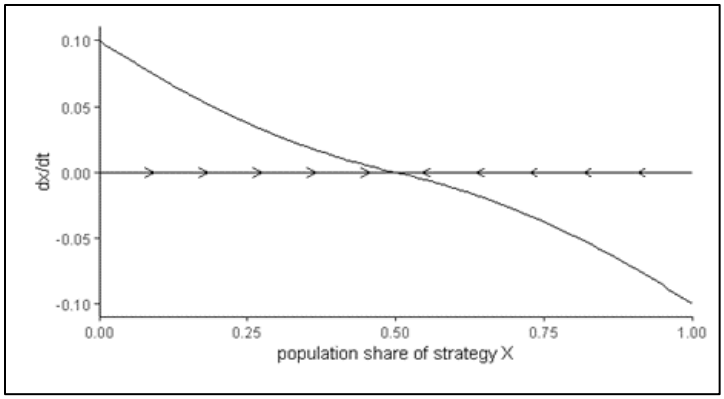


Figure 5.18: Logit dynamic (η = 0.01) - Case 3

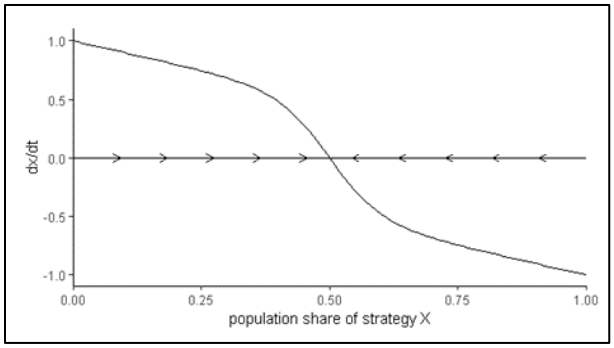


Figure 5.19: Logit dynamic (η = 0.3) - Case 3

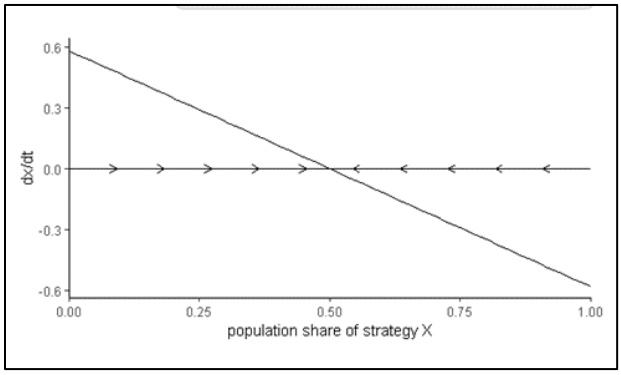


Figure 5.20: Logit dynamic (η = 0.7) – Case 3

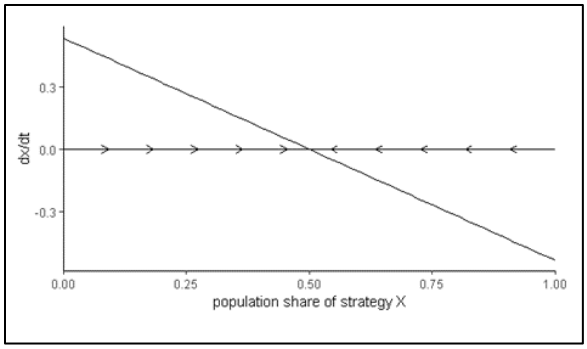
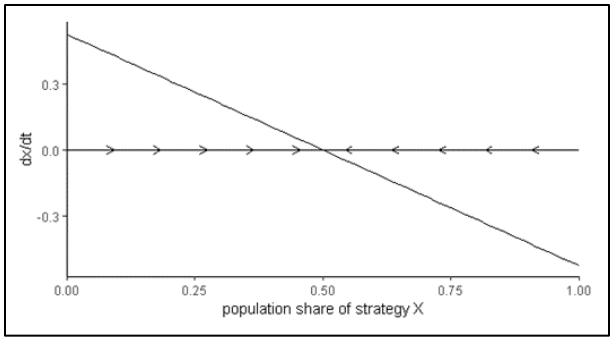


Figure 5.21: Logit dynamic (η = 0.99) - Case 3



The eigenvalues for outcome (0.5,0.5) implying 50% X and 50% Y agents, are -0.05 and -1.05. Since the real part of eigenvalues is negative, the outcome is stable.

When both X and Y agents are less social (Case 4), results show convergence towards one of the strategies, X or Y. When X share is lower than 50% in the population, the dynamic pushes towards the extinction of X agents (or 100% Y share). If X share is more than 50%, the convergence happens towards 100% X share (or 0% Y share). In the case of the logit dynamic, the higher η value leads to an equal percentage share of X and Y agents. Figure 5.22 shows the replicator dynamic results. BNN and Smith's dynamic results show a similar pattern (Figure 5.23, 5.24). Figures 5.25 to 5.28 show logit dynamic results with different η values.

Figure 5.22: Replicator dynamic - Case 4

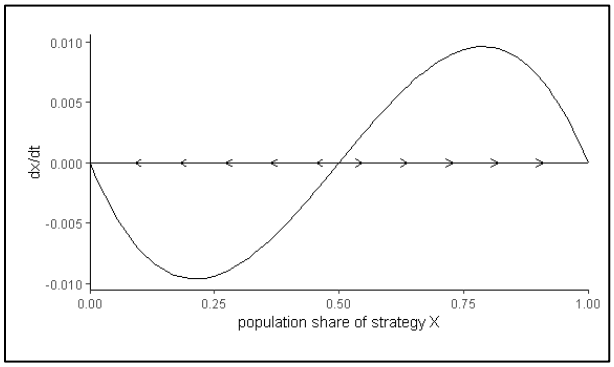


Figure 5.23: BNN dynamic - Case 4

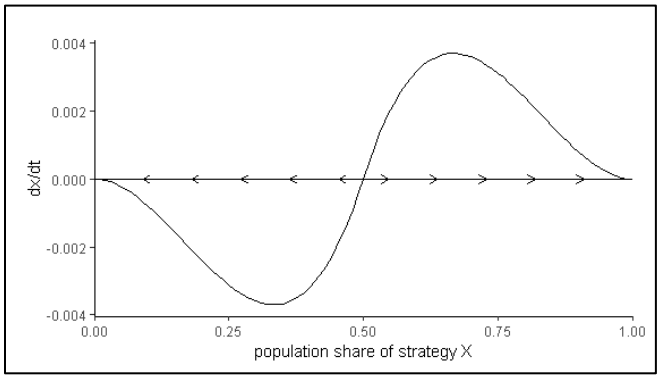


Figure 5.24: Smith dynamic – Case 4

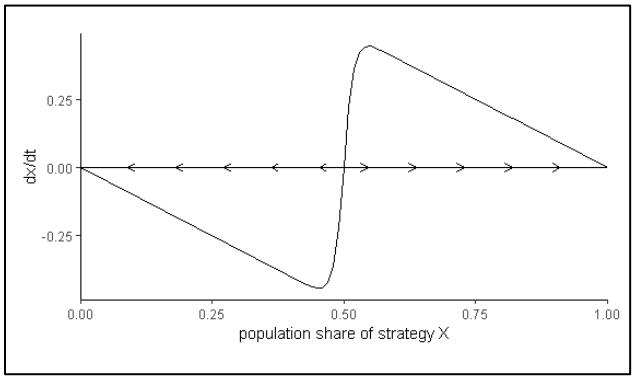


Figure 5.25: Logit dynamic ($\eta = 0.01$) - Case 4

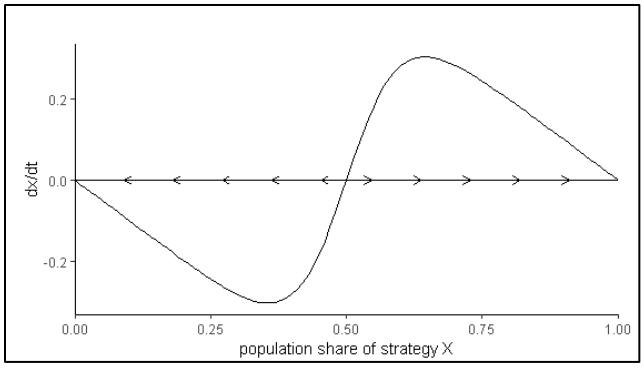


Figure 5.26: Logit dynamic (η = 0.3) - Case 4

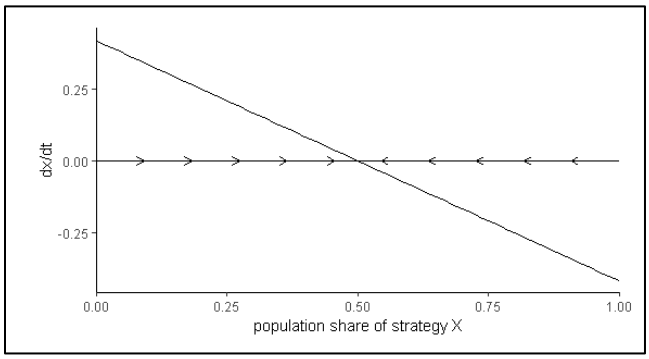


Figure 5.27: Logit dynamic (η = 0.7) – Case 4

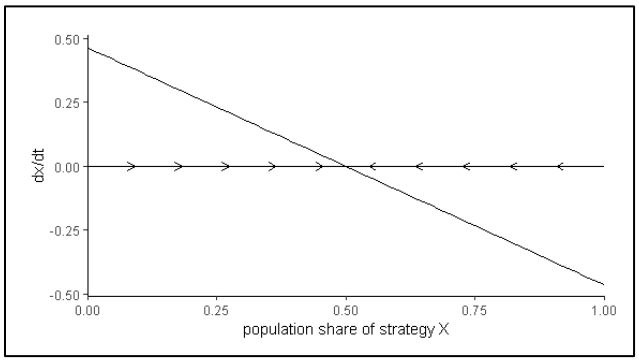
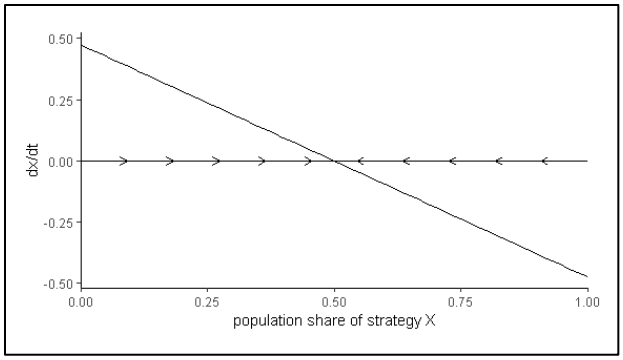


Figure 5.28: Logit dynamic (η = 0.99) - Case 4



The eigenvalues for outcome (1,0) and (0,1) are -1 and -0.10. For (0.5,0.5) outcome eigenvalues are -0.95 and -1. Since eigenvalues are negative, these outcomes are stable.

5.6 Results – Best response function approach

We present results from the best response function approach in the form of line graphs and network graphs. Line graphs show the percentage of X and Y strategy share on Y-axis in the ratio form ranging from 0 (0% agents) to 1 (100% agents) while X-axis shows time period. The network graph shows the distribution of X and Y agents on the specific network nodes at any given point in time during the simulation run. To keep the discussion simple, we primarily focus on results where we observed deviation in results using the best response approach compared to results from replicator dynamics.

Results from the Case 1 payoff matrix show the possibility of both X and Y outcomes when agents are connected in small-world networks. In some cases, % of agents following X outweighs Y. On the other hand, the replicator dynamic results show convergence towards Y agents. Figure 5.29 shows the percentage of times X and Y are played over time when agents are connected with the

SW3 network and p_{edge} value of 0.99. This shows that about 90% of agents are playing the X strategy after the initial few periods. Figure 5.30 shows the initial state of the population with which the game is started, and Figure 5.31 shows the population state by the end of 50 time periods.

Figure 5.29: SW3, p_{edge} = 0.99, Case 1

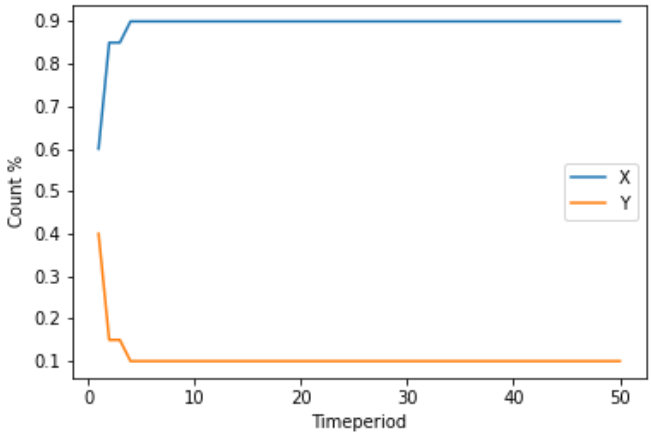


Figure 5.30: SW3, p_{edge} = 0.99, Network initial state, Case 1

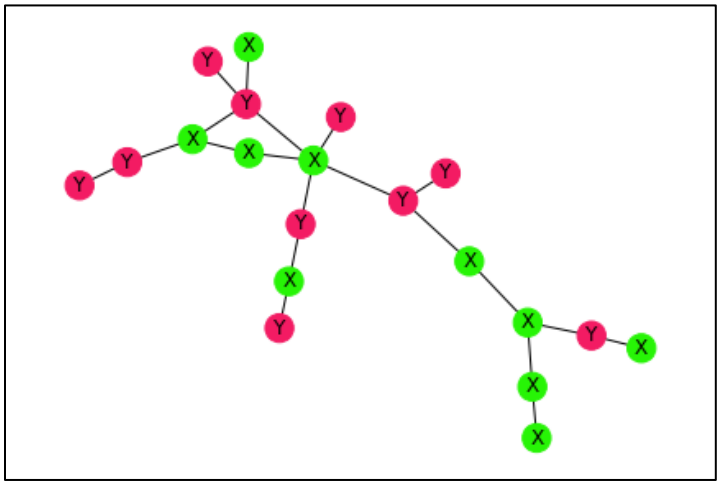
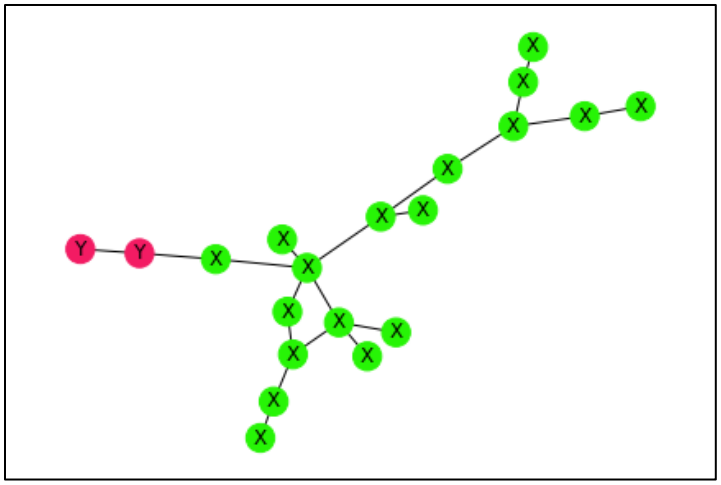


Figure 5.31: SW3, p_{edge} = 0.99. Network state by the end of 50 periods, Case 1



Case 2 results also show the possibility of both X and Y being played when agents are connected in small-world networks. In some cases, Y was played higher than X, whereas replicator dynamics showed a unilateral X outcome. Figure 5.32 shows one possibility where X and Y norms co-exist in the population. Figures 5.33 and 5.34 show the corresponding network state at the game's beginning and end, respectively.

Figure 5.32: SW1, p_{edge} =0.99, Case 2

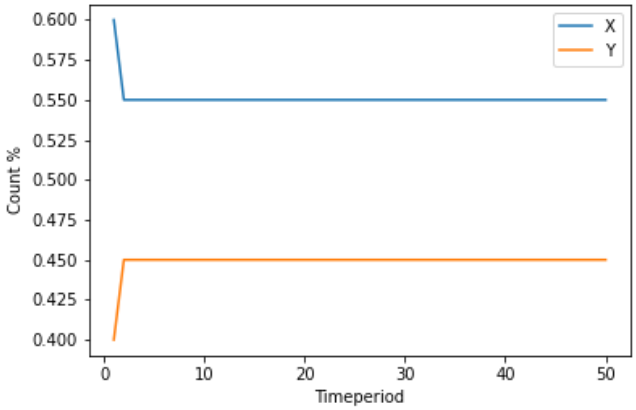


Figure 5.33: SW1, p_{edge} =0.99, Network initial state, Case 2

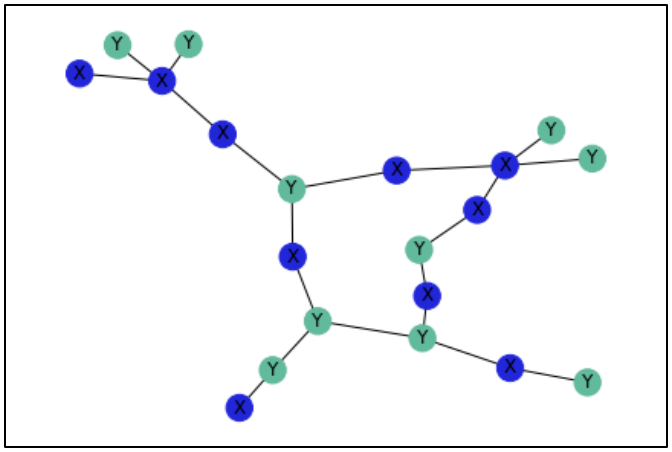
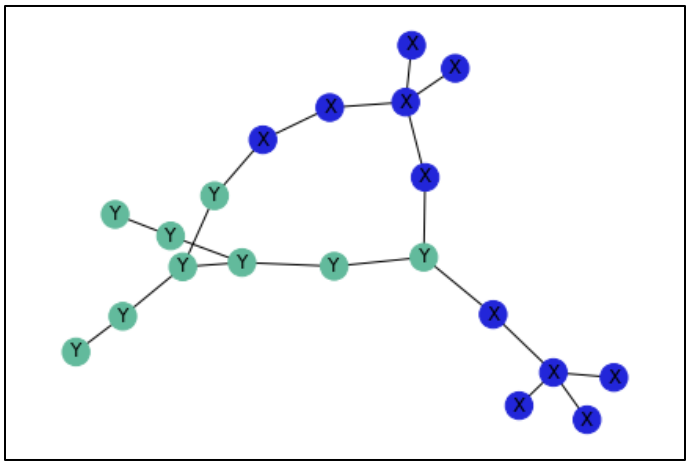


Figure 5.34: SW1, p_{edge} =0.99, Network state by the end of 50 periods, Case 2



When both agents are more social (Case 3), results show the presence of both X and Y agents most of the time, as found in the case of replicator dynamics. However, the percentage share of X and Y may not necessarily be in the ratio of 50% each, which the replicator dynamics results show. Some results show a 60/40 split, while some indicate a 90/10 or 70/30 split among X and Y. Grid network results also suggest the possibility of convergence towards a single outcome. Figures 5.35 to 5.37 show one result where the % share of X and Y is not in the 50/50 split. Figure 5.38 shows another possibility where only the Y norm is being followed after a certain time period in the case

of a 2d grid network. Figures 5.39 and 5.40 show the corresponding initial state and end state of the network.

Figure 5.35: SW2, p_{edge} = 0.5. Case 3

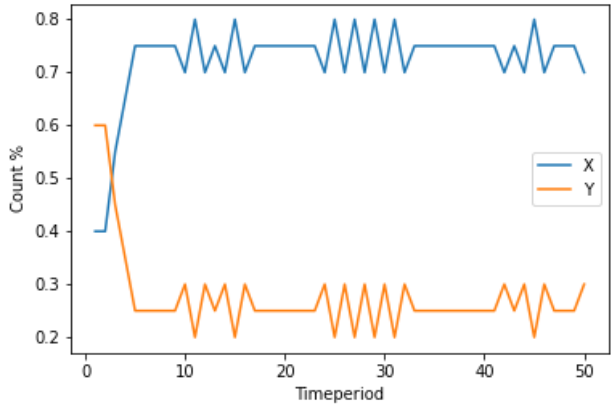


Figure 5.36: SW2, p_{edge} = 0.5, Network initial state, Case 3

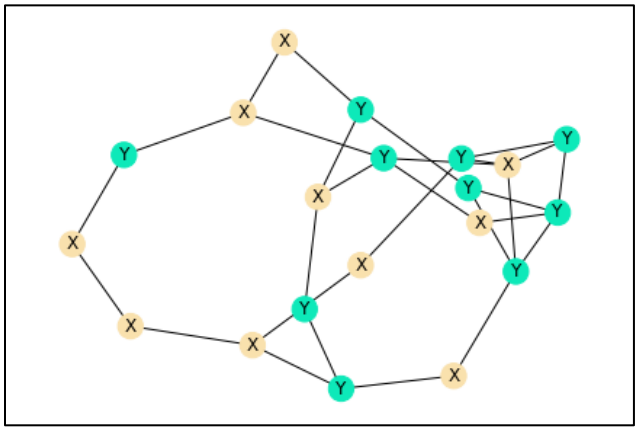


Figure 5.37: SW2, p_{edge} = 0.5, Network state by the end of 50 periods, Case 3

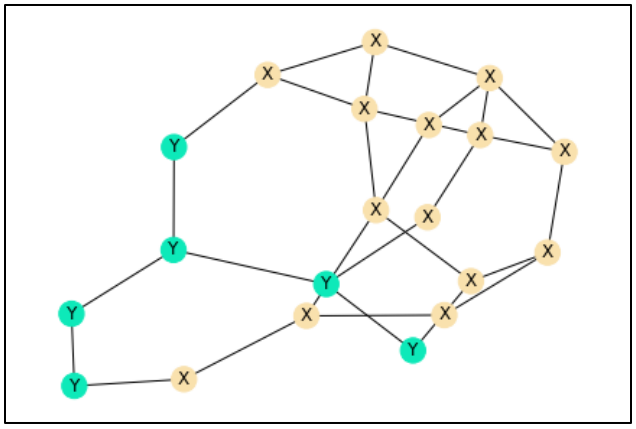


Figure 5.38: 2d-grid network, Case 3

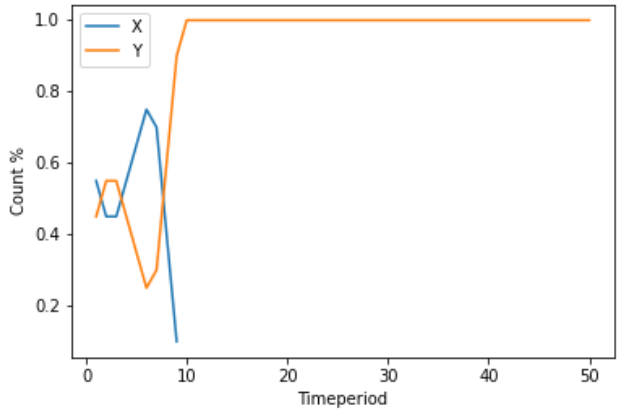


Figure 5.39: 2d-grid network. Initial state, Case 3

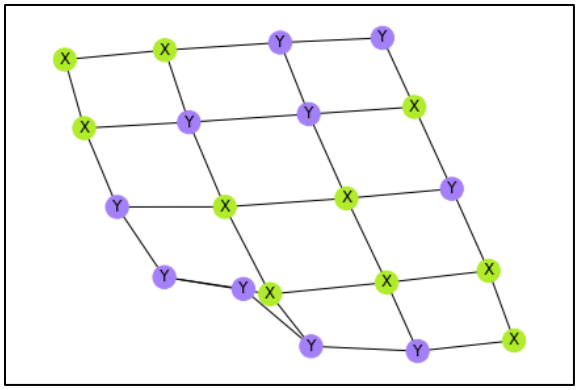
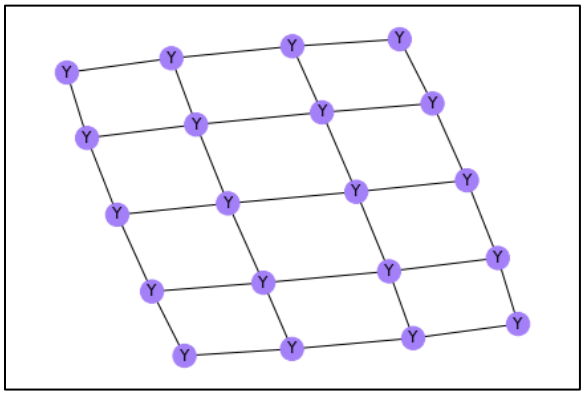


Figure 5.40: 2d-grid network. Network state by the end of 50 periods, Case 3



Case 4 results also show the possibility of the coexistence of X and Y agents in the population. Figure 5.41 demonstrates one such possibility. This contrasts with results achieved from replicator dynamics which shows the existence of either X or Y agents in the population. Figures 5.42 and 5.43 show the network's corresponding initial and end states.

Figure 5.41: SW1, p_{edge} = 0.99, Case 4

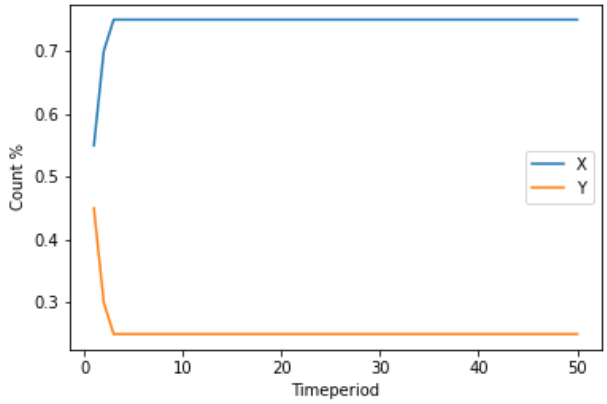


Figure 5.42: SW1, p_{edge} = 0.99, Initial state, Case 4

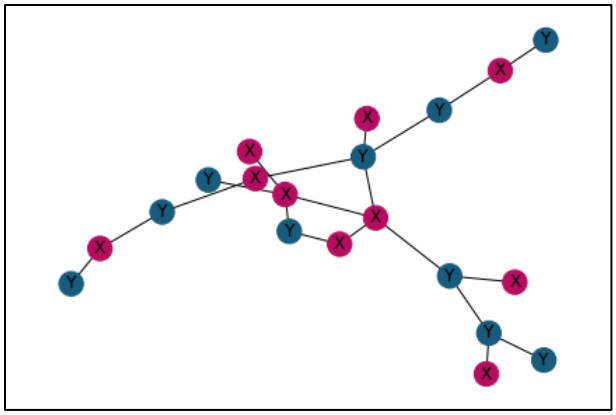
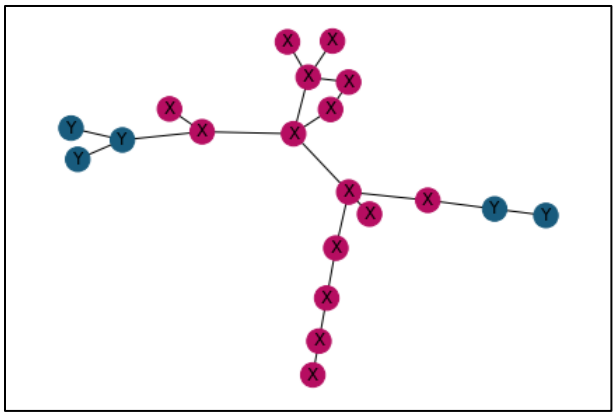


Figure 5.43: SW1, p_{edge} = 0.99, Network state by the end of 50 periods, Case 4



We then assessed the impact of increased neighbourhood size from 2 to 4, keeping the rest of the parameters constant. We did not observe many changes in Case 1 and Case 2 results with the increased neighbourhood size. Case 3 results show higher volatility is observed in the case of ring network structure in most cases, which results in a constant oscillation between X and Y. Figure 5.44 shows one such possibility. The larger neighbourhood size provides more opportunities for a single outcome to emerge in Case 4 compared to a lower neighbourhood size of 2 (Figure 5.47).

Figure 5.44: Ring network, Neighbourhood size = 4, Case 3

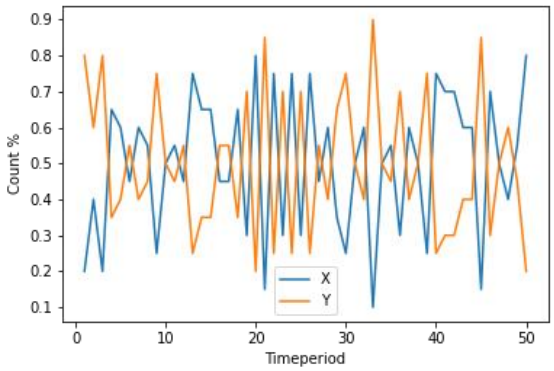


Figure 5.45: Ring network, Neighbourhood size = 4, Initial state, Case 3

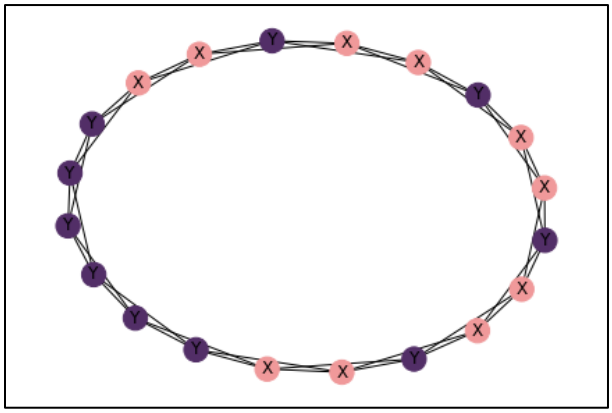


Figure 5.46: Ring network, Neighbourhood size = 4, Network state by the end of 50 periods, Case 3

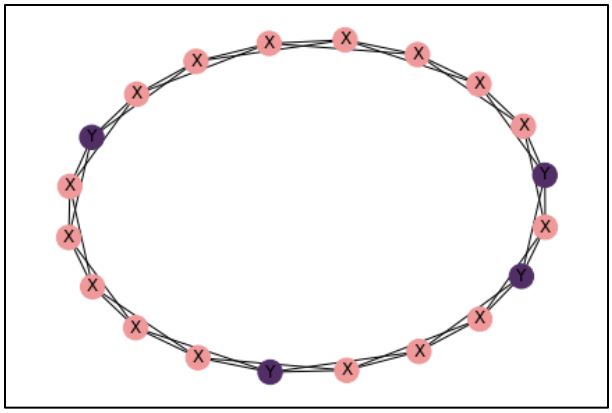


Figure 5.47: SW3, p_{edge} = 0.99, Neighbourhood size = 4, Case 4.

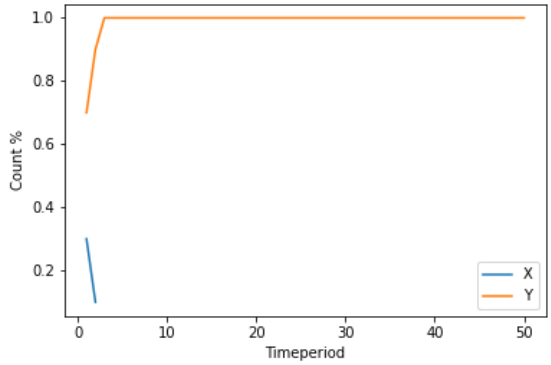


Figure 5.48: SW3, p_{edge} = 0.99, Neighbourhood size = 4, Initial state, Case 4

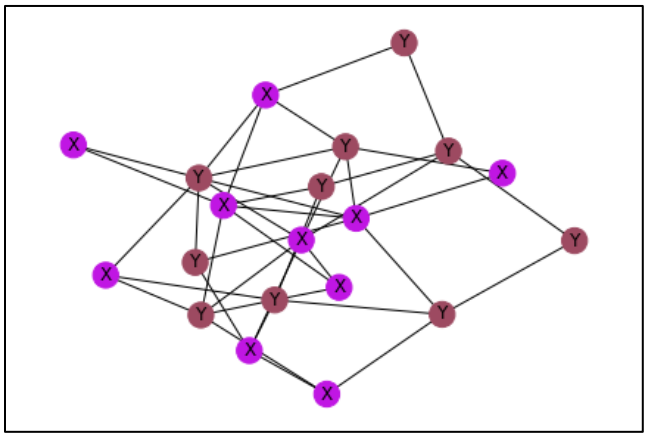
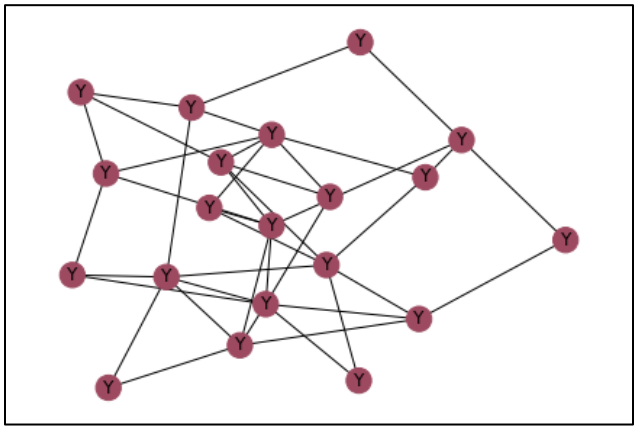


Figure 5.49: SW3, p_{edge} = 0.99, Neighbourhood size = 4, Network state by the end of 50 periods, Case 4



The graphs presented for the best response function approach are produced using *multi-agent-decision* Python library, which we have created and discussed in the previous chapter. This library can produce these graphs for any finite normal-form game. It can also be used to assess results with different combinations of neighbourhood size, network type, number of agents, agents' initial state etc. A detailed explanation of the input parameters requirements and the output produced by the library is provided on the library homepage (https://pypi.org/project/multi-agent-decision/).

5.7 Random networks and norm evolution

In this section, we considered two random networks, Erdos-Renyi (ER) network, and Barabasi-Albert (BA) random network. In alignment with what was done in preceding chapter, we want to assess how network density, clustering coefficient, network diameter, and network fat-tailedness impact the norm evolution. We assess this using the same response function as was followed in the previous section of this chapter where we assume agents follow the strategy which can bring

maximum payoff. In case there is more than one strategy having equal payoffs, agents randomly pick one of the strategies from those. We report results using all the four possibilities of migration game considered in the chapter, marked as Case 1,2,3,4 results respectively. We continue to assume that there are 20 agents in the population and the game starts with 50% X agents and 50% Y agents. The game runs for 50 time periods.

5.7.1 Erdos-Renyi (ER) network

Case 1 results

We start with ER network using probability of edge creation (p) as 0.08. In continuation of the pattern followed as in preceding chapter, we will report three sets of graphs for each individual result. First, the network initial state, second the network end state by the end of 50 time periods, and third the % of times X or Y is followed during the simulation run. Figure 5.50 till 5.52 shows there is not much deviation from the initial state which is 50% share for both X and Y. This network has a density value of 0.04 and clustering coefficient of 0.

Figure 5.50: ER network initial state with p=0.08. Case 1

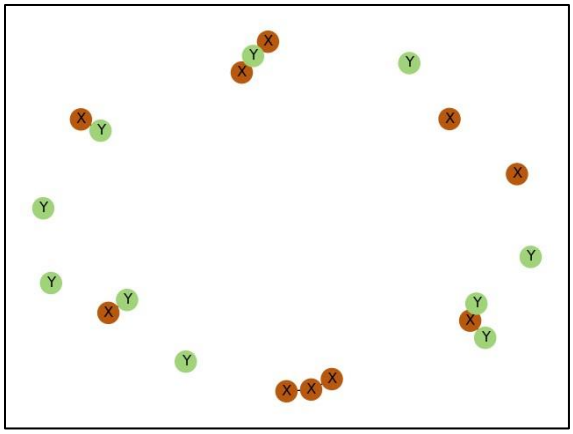


Figure 5.51: ER network end state with p=0.08. Case 1

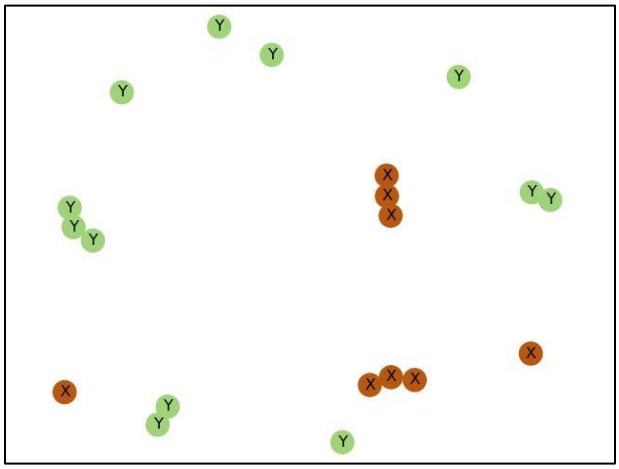


Figure 5.52: ER network norm evolution with p=0.08. Case 1

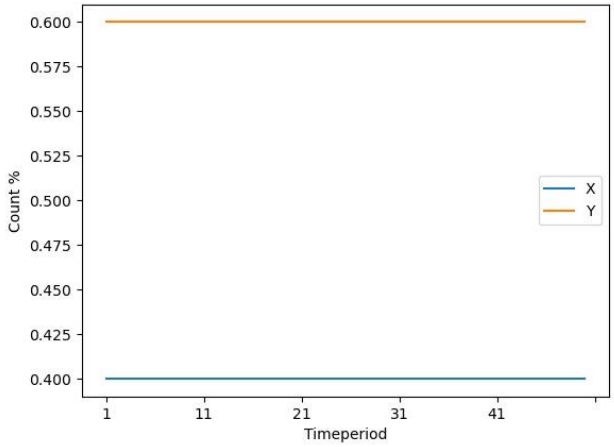


Figure 5.52 shows that when agents are not well connected, then there is a possibility of significant % of agents following strictly dominated strategy (X). When we increase the probability value to 0.96, this leads to a high-density network. Figures 5.53 till 5.55 show one such possibility. In a high-density environment, Y is followed by the majority of agents.

Figure 5.53: ER network initial state with p=0.96. Case 1

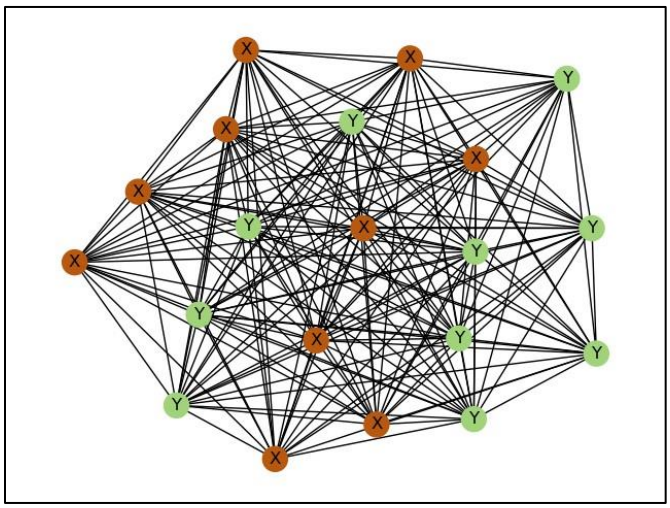


Figure 5.54: ER network end state with p=0.96. Case 1

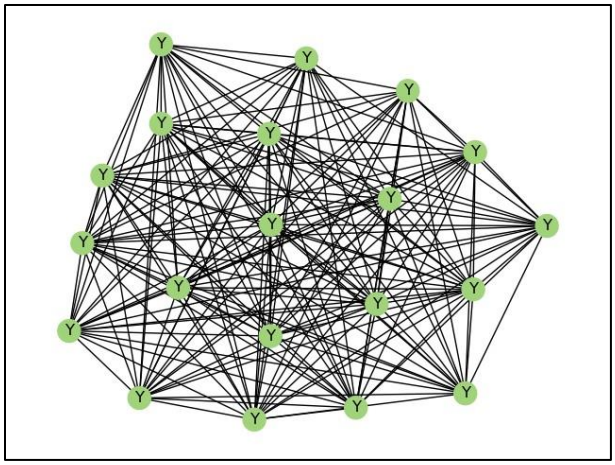
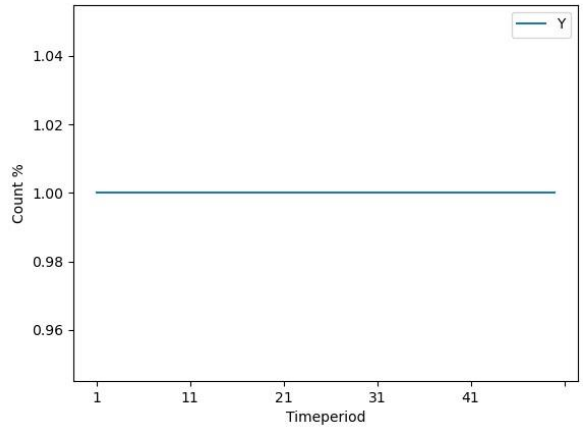


Figure 5.55: ER network norm evolution with p=0.96. Case 1



Therefore, higher density leads to Y outcomes in the majority of cases. However, there are few scenarios when the probability value is high enough to have connected graph along with high diameter and medium-to-heavy fat-tailedness. This can lead to X outcome in the simulations. Figure 5.56 till 5.58 presents one of these scenarios. Figure 5.56 uses a p value of 0.14 and has density of 0.17, clustering of 0.24, diameter of 5 and medium fat-tailedness. As shown in Figure 5.58, only X outcome is being played here.

Figure 5.56: ER network initial state with p=0.14. Case 1

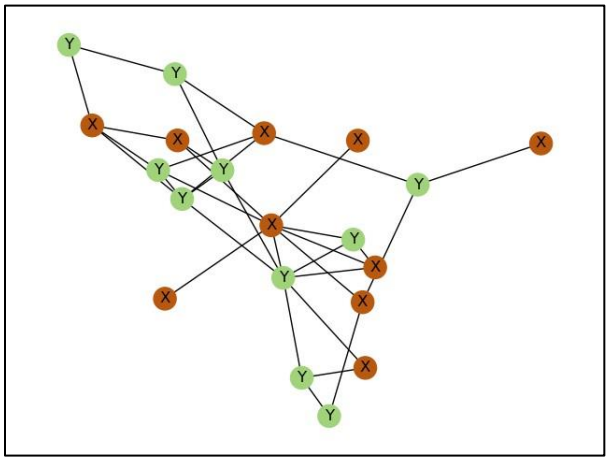


Figure 5.57: ER network end state with p=0.14. Case 1

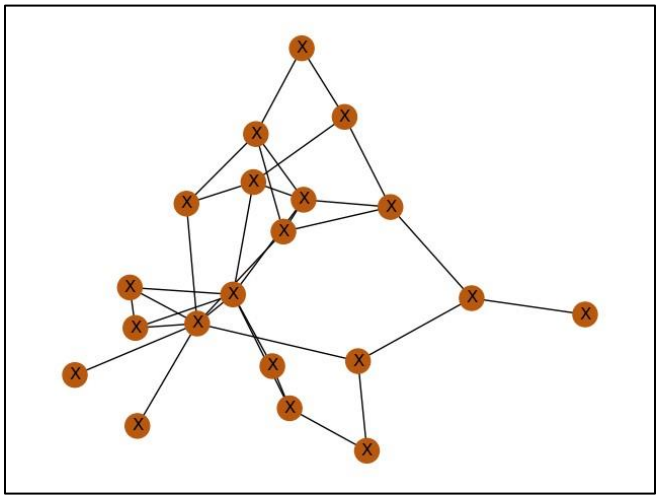
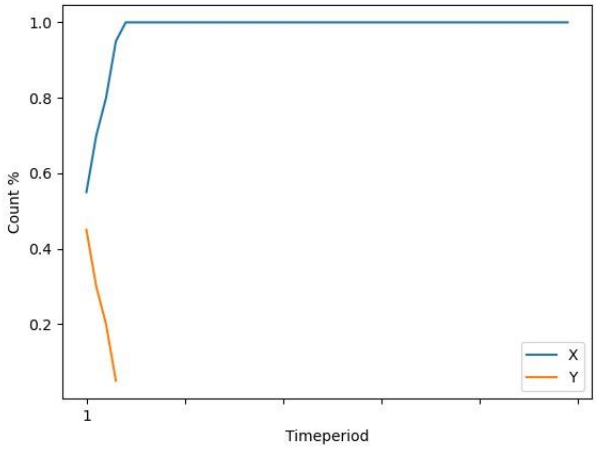


Figure 5.58: ER network norm evolution with p=0.14. Case 1



Case 2 results

In this payoff matrix, X is strictly dominant strategy. Results on this payoff matrix are similar to case 1 directionally, implying as p value increases, there are higher chances of X outcome to emerge as the norm compared to Y outcome. However, at lower probability values, the fat-tailedness of the network and network connectedness can influence results. Figures 5.59 till 5.61 represent one of the scenarios where p value is 0.11, network is not connected, density of 0.11, clustering of 0.14 and low fat-tailedness network.

Figure 5.59: ER network initial state with p=0.11. Disconnected network. Case 2

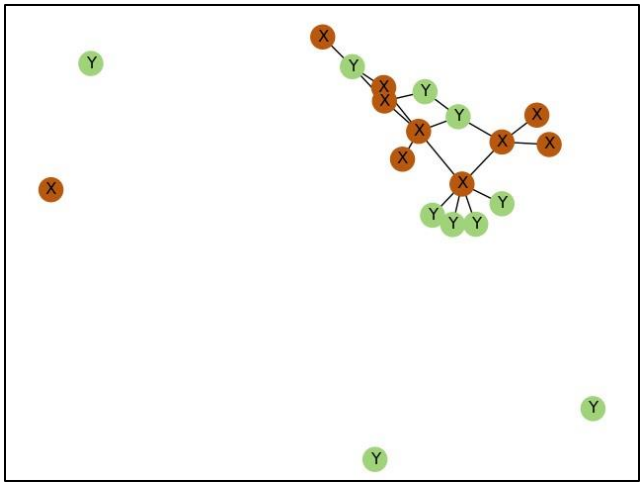


Figure 5.60: ER network end state with p=0.11. Disconnected network. Case 2

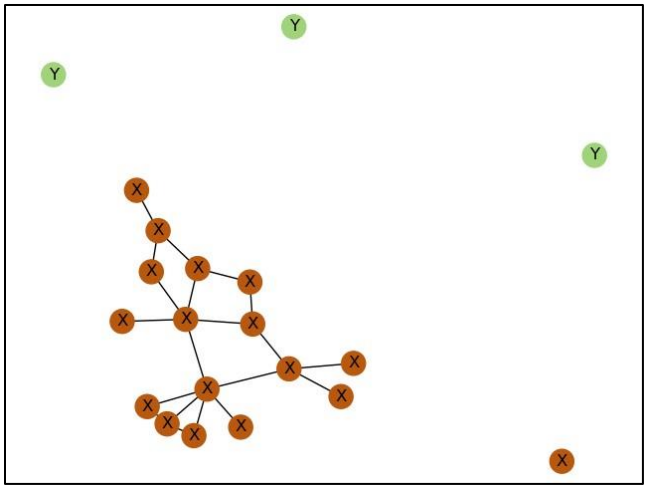


Figure 5.61: ER network norm evolution with p=0.11. Disconnected network. Case 2

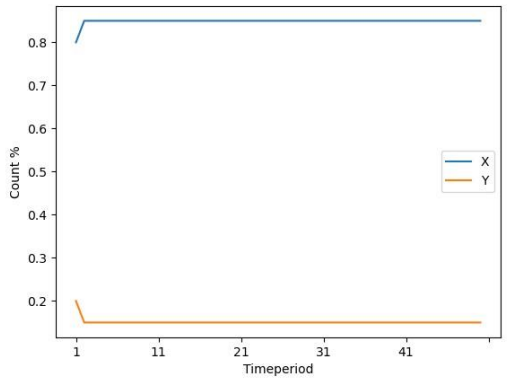


Figure 5.61 shows that when the graph is not connected and the network is low fat-tailed, there is a chance of other small local norms followed by agents who are not connected with other agents. On the other hand, when network is connected, and network has medium to heavy fat-tailedness this has higher chances of agents playing a single norm. The following three figures show one of these scenarios with the same probability value of 0.11. Figure 5.62 has network density of 0.12, diameter of 10, clustering of 0.1 and medium fat-tailedness.

Figure 5.62: ER network initial state with p=0.11. Connected network. Case 2

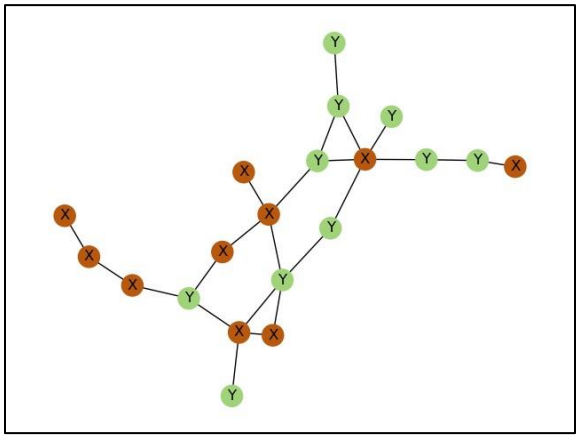


Figure 5.63: ER network end state with p=0.11. Connected network. Case 2

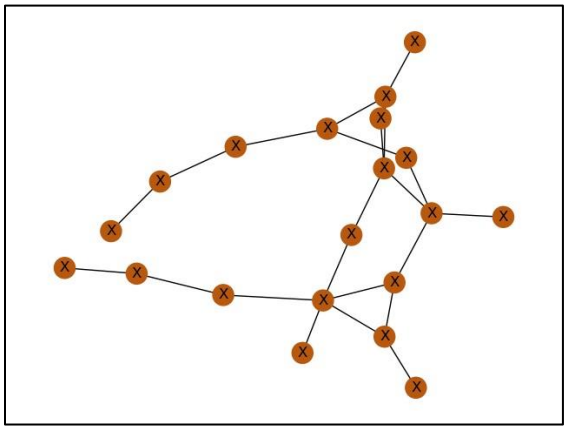
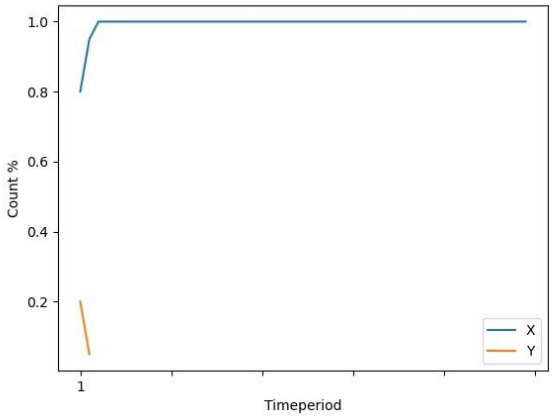


Figure 5.64: ER network norm evolution with p=0.11. Connected network. Case 2



A lower probability value in conjunction with lower clustering and low fat-tailed network can result in both the outcomes X and Y despite network being connected. A lower clustering coefficient means a significantly lower value as compared to its network density. Therefore, medium to heavy fat-tailedness generates one outcome or closer to one outcome with skewed results. Low fat-tailedness generally generates diverse outcomes for a given probability value. Connected graphs have higher potential to generate single norm compared to disconnected graphs.

At higher probability values of edge creation, fat-tailedness of the network does not matter much.

Case 3 results

In case 3 payoff matrix, none of the strategy is strictly or weakly dominant. X agents get higher payoffs when they interact with Y agents and vice-versa. In this scenario, a lower probability implying low density network resulting in both the outcomes, X and Y. Figures 5.65 till 5.67 show one such result with p = 0.11. Figure 5.65 has density of 0.09, clustering of 0 and low fat-tailedness.

Figure 5.65: ER network initial state with p=0.11. Case 3

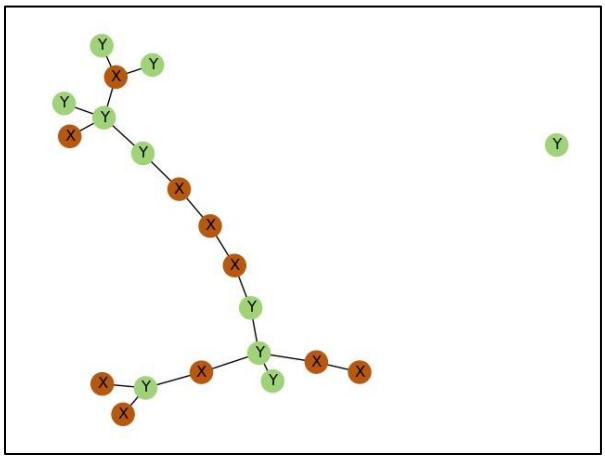


Figure 5.66: ER network end state with p=0.11. Case 3

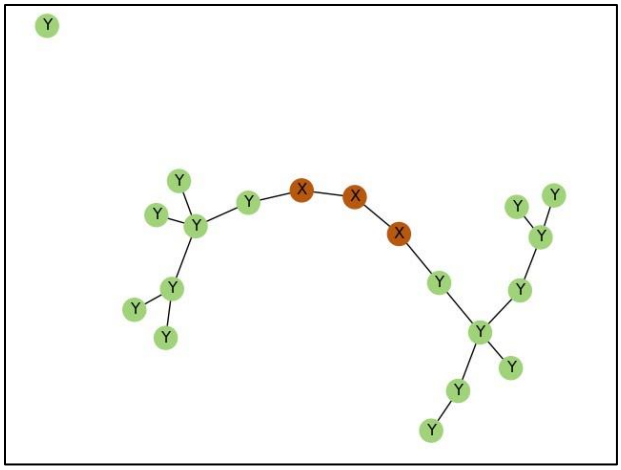


Figure 5.67: ER network norm evolution with p=0.11. Case 3

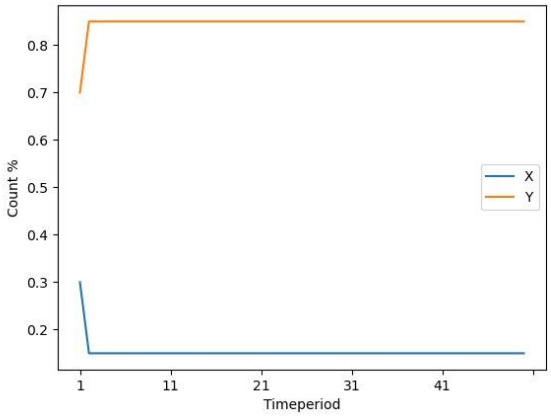


Figure 5.67 shows both the outcomes, X and Y being observed. A higher density network combined with medium to heavy fat-tailedness showing convergence towards a single outcome, sometimes X and sometimes Y. Figures 5.68 till 5.70 represent one such scenario. Figure 5.68 has network density of 0.49, clustering coefficient of 0.5, diameter of 3 and heavy fat-tailedness.

Figure 5.68: ER network initial state with p=0.44. Case 3

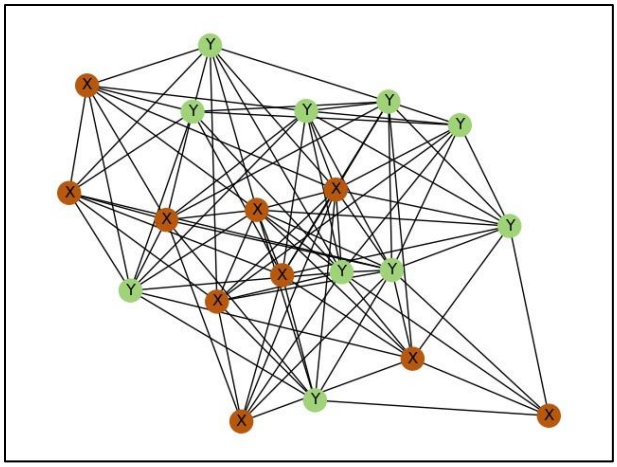


Figure 5.69: ER network initial state with p=0.44. Case 3

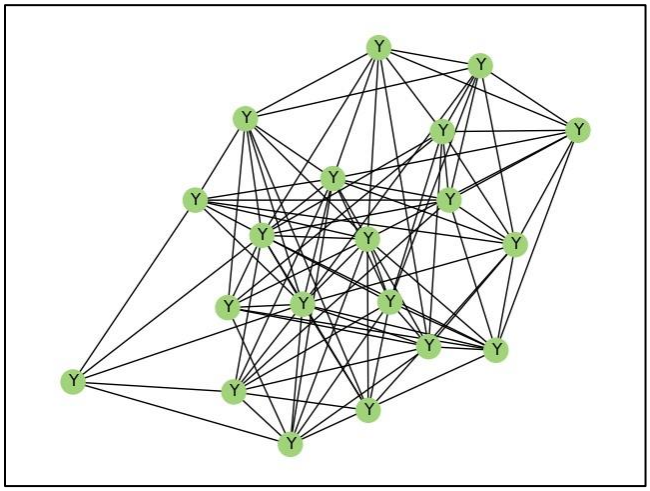


Figure 5.70: ER network norm evolution with p=0.44. Case 3

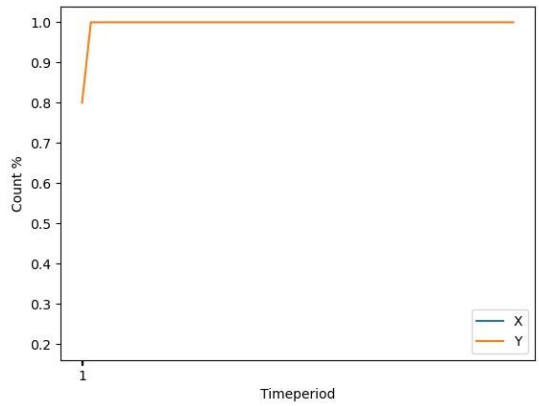


Figure 5.70 shows single outcome emergence of Y. Therefore, in Case 3 we have observed medium density (~0.5) combined with heavy fat-tailedness, or a relatively higher density (~0.7 and above) irrespective of fat-tailedness resulting in convergence towards one outcome in majority of times. However, when the probability value is lower (~0.3 and below), we have observed its leading to both X and Y as outcomes and do not have 100% convergence towards a single outcome in most of the cases.

Case 4 results

In this case, none of the strategies is strictly or weakly dominant. X agents would get lower payoffs if paired with Y agents and vice versa. In this case, the result is similar to case 3 for larger probability values. However, for lower probability values or lower dense networks, there is a higher probability of getting one outcome in this case as compared to previous case. This is explained by the payoff structure of this case which states agents would get higher payoffs when they are paired with their own types. A connected network with heavy fat-tailedness increases the chances of a single outcome in low density network.

To summarize, we can conclude below results from ER random networks.

- A higher probability for edge creation or dense network leads to one outcome in all the four cases.
- Disconnected networks generate more local norms compared to connected networks.
- Medium to heavy fat-tailedness generates one outcome or closer to one outcome with skewed results. Low fat-tailedness generally generates diverse outcomes for a given probability value.
- In Case 3, there is a higher probability of two outcomes for lower probability values while Case 4 shows higher convergence towards single outcome for lower probability values also. However, exceptions do exist with specific fat-tailedness, clustering and connectedness of the network.

5.7.2 Barabasi-Albert (BA) random network

In the case of BA network, we start with specifying an initial base graph. We assume the base graph to be ER graph. The properties of BR network require the initial graph to be a connected graph, hence we would mostly observe cases with sufficiently high probability for edge creation which can produce a connected graph. In addition to p, we have two more parameters, m1 and m2. M1 is the number of nodes with which the initial graph (ER) is generated, and m2 is the number of edges that are preferentially attached from new node to existing nodes with high degree. Therefore, we have 3 parameters in total which we would change.

Case 1 results

In this case, Y is strictly dominant strategy. We have seen Y as the outcome in BA network for the majority of cases. Figure 5.71 shows one such example with p of 0.31, m1 of 3 and m2 of 1. Network density is 0.1 for this network, diameter of 5 and 0 clustering coefficient. Y is the dominant outcome observed.

Figure 5.71: BA network initial state with p=0.31, m1=3, m2= 1. Case 1

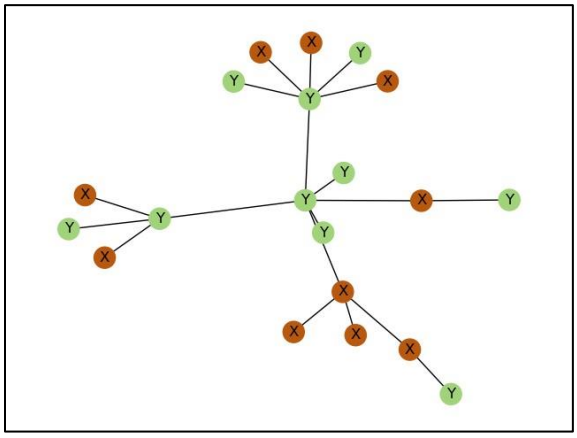


Figure 5.72: BA network end state with p=0.31, m1=3, m2= 1. Case 1

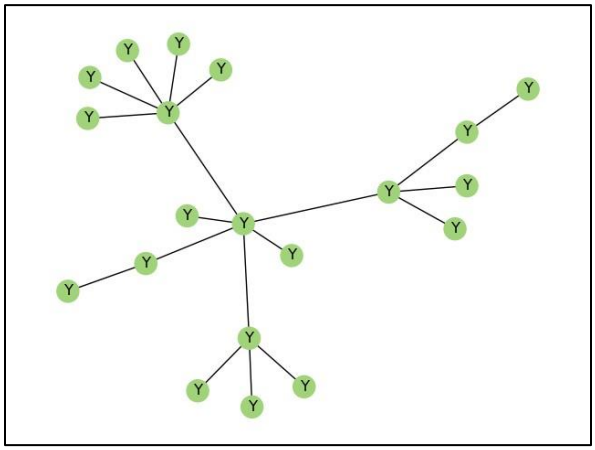
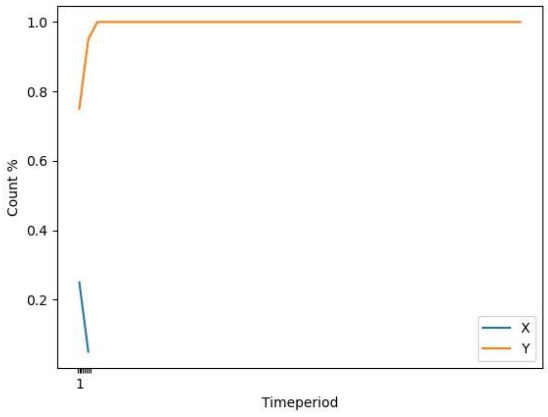


Figure 5.73: BA network norm evolution with p=0.31, m1=3, m2= 1. Case 1



There are few cases where we have seen the emergence of X as the norm. Results are shown to depend upon the "hub" nodes agents' strategies and their neighbours' strategies. Figures 5.74 till 5.76 show one such example. This network has density of 0.3, clustering coefficient of 0.57, diameter of 2 and low fat-tailedness.

Figure 5.74: BA network initial state with p=0.22, m1=19, m2= 19. Case 1

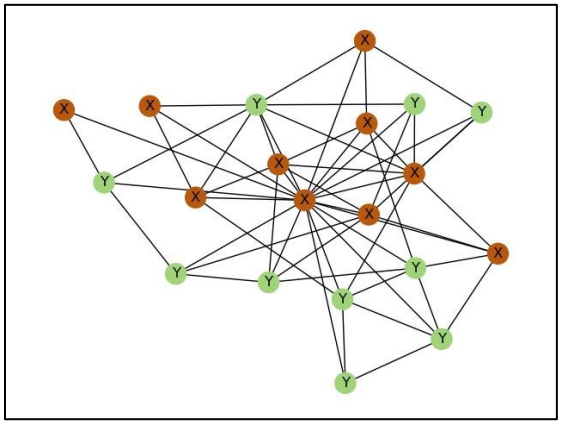


Figure 5.75: BA network end state with p=0.22, m1=19, m2= 19. Case 1

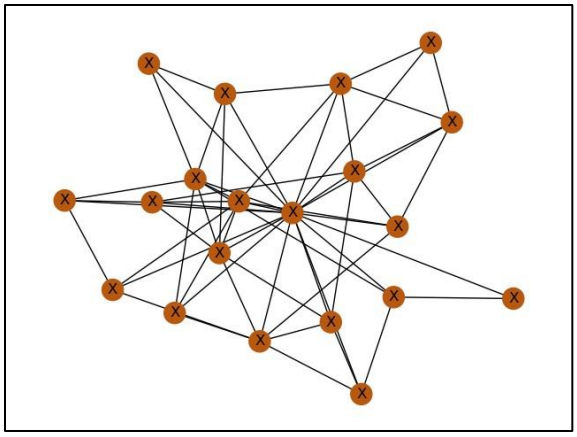
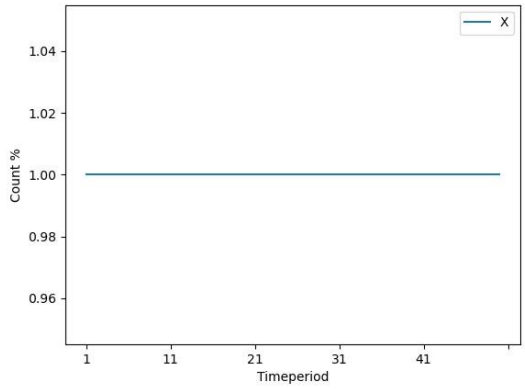


Figure 5.76: BA network norm evolution with p=0.22, m1=19, m2= 19. Case 1



Case 2 results

In case 2, X is strictly dominant outcome. Results show in the majority of cases, X outcome is observed. However, in a few cases with relatively lower values of m1, and m2 we have observed Y outcome. Figures 5.77 till 5.79 show one such possibility. Figure 5.77 has 0.1 network density, 0 clustering coefficient, 4 diameter and low fat-tailedness.

Figure 5.77: BA network initial state with p=0.36, m1=4, m2= 1. Case 2

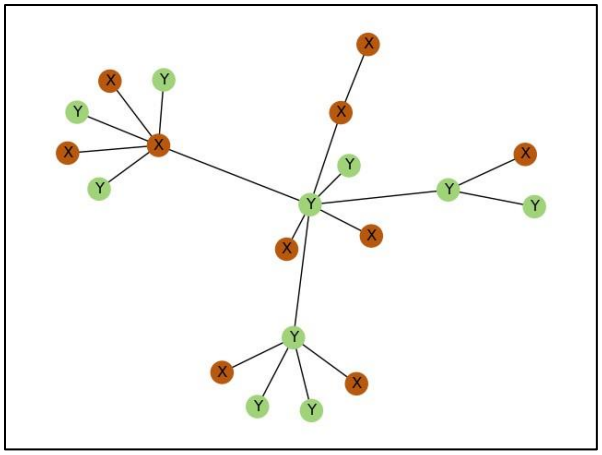


Figure 5.78: BA network end state with p=0.36, m1=4, m2= 1. Case 2

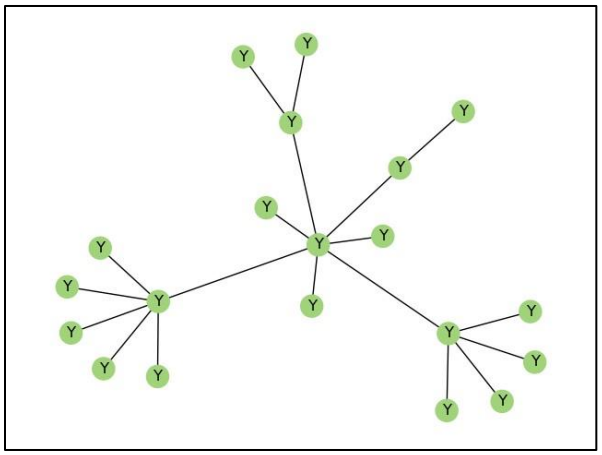
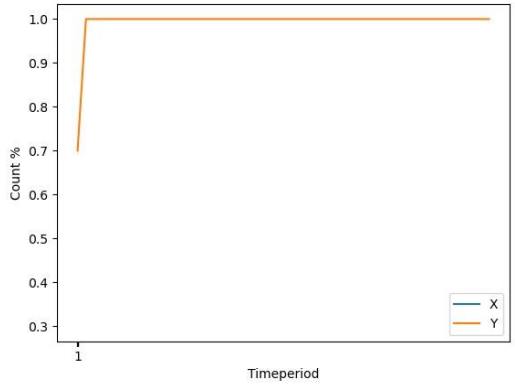


Figure 5.79: BA network norm evolution with p=0.36, m1=4, m2= 1. Case 2



As Figure 5.79 shows Y outcome (strictly dominated) is played more frequently.

Case 3 results

In this case, X agents get higher payoff when they pair with Y agents and vice-versa. In this case, the majority of times there is convergence towards 1 outcome, either X or Y. It's observed across all ranges of probability values. Figures 5.80 till 5.82 represent one such example where there is convergence towards Y outcome.

Figure 5.80: BA network initial state with p=0.36, m1=4, m2= 1. Case 3

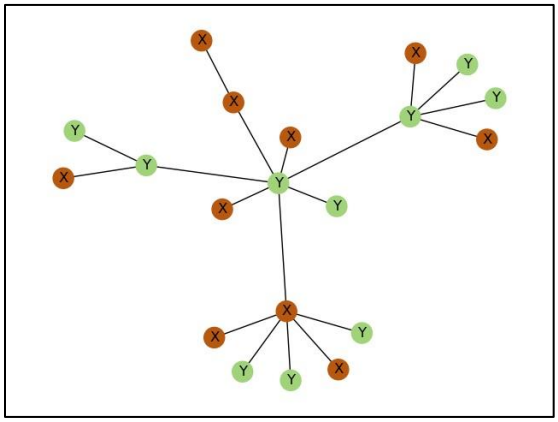


Figure 5.81: BA network end state with p=0.36, m1=4, m2= 1. Case 3

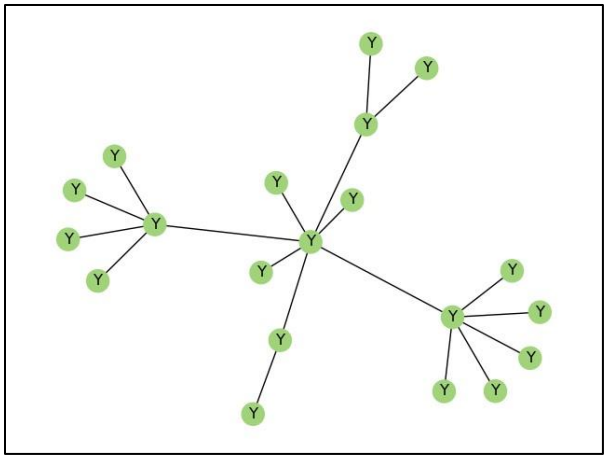
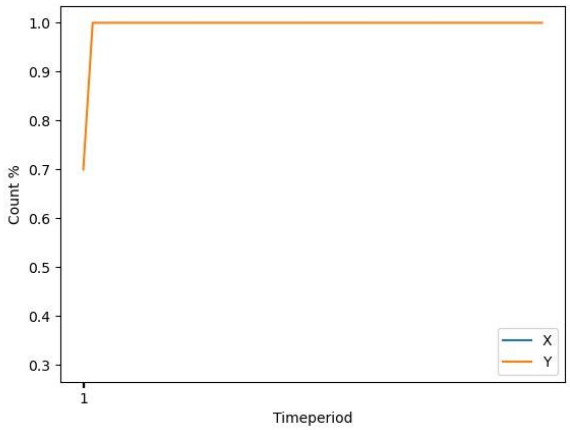


Figure 5.82: BA network norm evolution with p=0.36, m1=4, m2= 1. Case 3



Case 4 results

In this case too, results are similar to what observed under Case 3. There is a convergence towards single outcome, X or Y most of the time. There are a few exceptions, where we can see both the outcomes. One such example is shown in Figures 5.83 till 5.85 where network density is 0.15, clustering coefficient is 0.07, diameter is 6, and heavy fat-tailedness.

Figure 5.83: BA network initial state with p=0.16, m1=19, m2= 4. Case 4

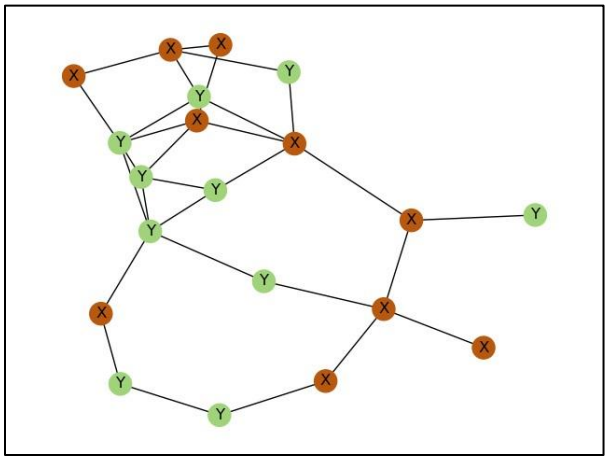


Figure 5.84: BA network end state with p=0.16, m1=19, m2= 4. Case 4

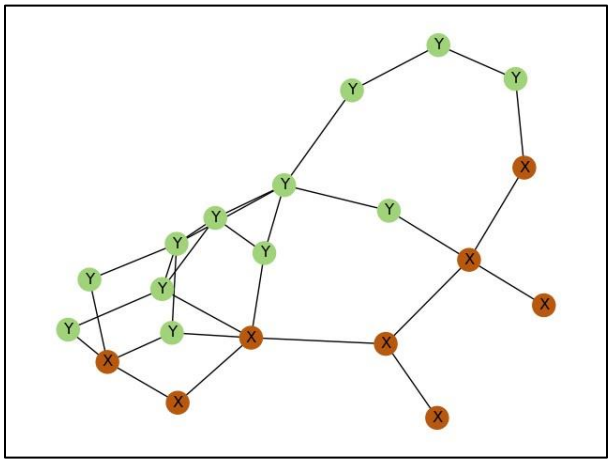
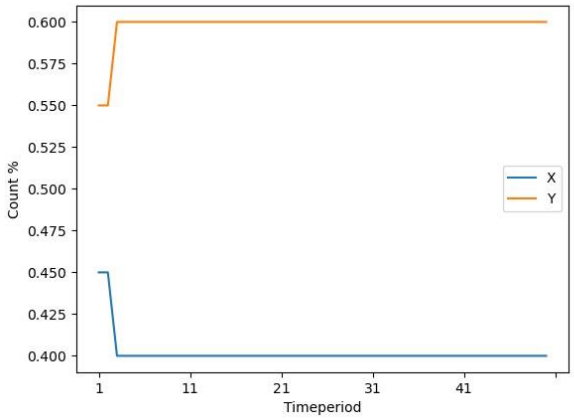


Figure 5.85: BA network norm evolution with p=0.16, m1=19, m2= 4. Case 4



We can summarize findings from BA network in below points:

- For case 1 and case 2 results, these are mostly in alignment with the analytical results. There is convergence towards strictly dominant outcome. There are few exceptions observed when m1 and m2 values are lower.
- In case 3 and 4, there is largely a convergence towards 1 outcome, X or Y.
- In case 4, there are few exceptions observed for lower dense and heavy fat-tailed networks where two outcomes are observed.
- Results are primarily dependent on the strategies followed by "hub" nodes and the number of edges connected to it.

5.8 Conclusion

Different dynamic results show that when both domestic and foreign agents are less social, the convergence happens towards extremes; all agents follow domestic norms or foreign norms. If both agent types are more social, the population would have an equitable region of domestic and

foreign norms agents. If one of the agent types is more social than the others, the social agent types dominate the population. On the other hand, when agents decide what actions to follow using the best response framework, we can expect to see both domestic and foreign norms co-existence in most situations when agents are connected in small-world networks. Network structures play an important role in the emergence of norms. A higher dense network usually leads to convergence towards one outcome. Results are sensitive to the fat-tailedness and clustering of the network in case of lower dense networks. Medium to heavy fat-tailedness can lead to convergence towards one outcome in the absence of high densities. When agents are connected in Barabasi-Albert network structures, results primarily dependent on the strategies followed by "hub" agents. Results from this network are primarily aligned with analytical results but provide different results when the network is less dense and has heavy fat-tailedness. Networks which are disconnected generate more local norms compared to connected networks. When both agents are social, Erdos-Renyi network shows possibilities towards two outcomes which is in alignment with replicator dynamics for lower probability values, while Barabasi-Albert random network results shows the convergence towards single outcome. In case of both agents being less social, Erdos-Renyi network shows possibilities towards one outcome for lower probability values which is in alignment with dynamics results, while Barabasi-Albert random network results shows the possibility towards two outcomes in case of lower dense and heavy fat-tailed networks.

Conclusions

The thesis touched upon the norm evolution problem using the evolutionary approach. We identified three areas in which context norm evolution could be applied. The first context involves having a row and column player with defined strategies and corresponding payoff values. Agents play the game repeatedly and decide what strategy to choose depending on their opponents' strategies in the previous periods. We defined two approaches agents can choose to decide what strategy to choose, the exhaustive best response approach and the expected payoff approach. We applied this framework in five games: prisoner's dilemma, battle of sexes, matching pennies, stag hunt, and coordination game. Prisoner's dilemma results show that when agents are allowed to make mistakes, means choosing non-recommended actions randomly with a certain probability e and taking recommended actions with probability (1 – e), then (cooperate, cooperate) or (cooperate, defect) also has chances of emerging as the norm. This posits the possibility of sustaining a non-Nash and Pareto superior outcome as a norm. Games involving multiple pure strategy Nash equilibria like stag hunt, battle of sexes, and coordination game results depend upon the initial history, memory length of the agents, payoff values etc. Matching pennies game results did not show any clear trend. In general, results show there is a possibility of outcomes that can emerge as a norm that are neither Nash equilibria nor Pareto efficient. We provide a computational framework which reinforces the view from Young (1993) that any strategy or action has the potential to emerge as a norm. It is driven by the dynamics followed and not necessarily due to that strategy being superior to others. We also assess the importance of memory length along with randomness on agent's decision making. A higher memory length leads to convergence towards actions pairs which are more in alignment with pure and mixed strategies Nash equilibria.

Randomness facilitates convergence or coordination towards a certain action pair which otherwise may not have been possible. To generalize and to ensure the reusability of these approaches for the reader, we have created an open-source python library, *game-simulator*, which can be used to assess evolution for any combinations of m*n payoff matrix, memory length, time period, initial states etc.

The second area where norm evolution can be applicable is where we have strategies defined along with their payoffs. There is a population, and a specific percentage of agents follow these strategies. We explained this with the help of the Nash demand game of Axtell et al. (1999) where there is a fixed pie (say 100 dollars), and agents are expected to distribute that among themselves. Agents can make three demands, namely High (H) with a payoff of 70, Medium(M) with a payoff of 50, and Low (L) with a payoff of 30. These three choices are agents' options or strategies which they can choose during the game. At any given point, two agents are selected, and they make their demands. The rule of the game is that agents get these payoffs only if the demands made by both the agents playing the game are lower than or equal to 100. If the total demanded payoffs exceed 100, none of the agents will receive anything. Therefore, if both the agents demand H or one of them demands H and another M, both would receive a zero payoff. Agents' initial distribution of strategies is defined before starting the game, implying how many % of agents follow H, M, and L. This distribution changes as agents play the game repeatedly over a period and change their strategies. Agents choose their neighbour's strategy if the payoff from playing that strategy is higher than the payoff from following their current strategy. Results show convergence towards M outcome across most network structures when we assume the initial state of agents is distributed in the ratio of 40/40/20 (H/M/L). We also investigated how the emerging norms get replaced by

other choices or actions that did not qualify the norm criteria earlier. We proposed a framework where agents are incentivized to move to a different choice, leading to the displacement of norms. The incentives provided depend upon multiple factors like the current norm strategy, how far agents are from meeting the norm criteria, what alternative option we want to see as norm, etc. We assume a fixed delta payout that agents receive to move from the established norm to something else which does not satisfy the norm criteria. We assume this delta payout to be fixed to maintain simplicity. However, in reality, this is expected to depend on many other factors like population size, agents' neighbourhood size, network structure etc. Results show that delta payout 20 is sufficient to move agents from M outcome to H or L when agents are connected in a small world network. This approach is open-sourced in the form of a Python library named *multi-agent-decision*.

The third and last area where we expect the scope of evolution exists when no fixed strategy or agents' choices are defined. We explained this with the help of the Naming game of Young (2015). In the Naming game, two agents are selected randomly and shown a picture of a face. Agents do not know the identity of other agents, and they independently suggest a name for the face. If agents suggest the same name, they get a positive reward; otherwise, they get a negative reward. At the end of each iteration, agents get to know the names proposed by opponent agents, and this keeps them updated with the names currently popular at any given time. There is no constraint on the names agents can propose, and it is left to their imagination. Agents use a perturbed response function to decide what names to propose, implying agents propose names randomly with certain probability e and propose the most frequent names with probability 1-e. 'e' here can be interpreted as the probability of committing an error by the agents. A fixed population size is defined along

with the agents' neighbourhood size. No constraint exists on the number of unique names that can evolve after the simulation. Results show that most of the population proposes 1 or 2 names in the case of a ring network with at least 1 name satisfying the norm criteria. In the case of small-world networks where there are *shortcuts* available, norms that emerge are not necessarily locally concentrated. This implies agents following norms proposed by those agents where there is no direct linkage among agents exist. The complete network results show convergence towards one name. Results vary when we incorporate different agent types and assume few agents as fixed. Fixed agents continue to use fixed strategy (propose the same name) every time, irrespective of what other agents propose. In general, results show fewer names satisfying the norm's criteria due to the larger number of unique names proposed compared to the case when no fixed agent exists. We have seen higher chances of norms emergence in case of denser networks. However, if the network is heavy fat-tailed, it can compensate for lower network density. A lower network diameter and higher clustering coefficient correlates positively with norm emergence. To generalize the results, we created a Python library, *multi-agent-coordination* , which can be used to get results with any custom user-defined parameters on agents' social networks and its associated parameters.

The three areas defined above consider individual agent-based approaches to norm evolution. We have compared results from this approach with the aggregative/macro approach towards evolution, which leverages replicator dynamics and others. We performed this comparison with the help of a migration game. The migration game entails agents of certain types migrating from a domestic country to a foreign country where most agents follow certain norms X and Y, respectively. Agents' payoffs vary when domestic country agents interact with foreign country agents. We want

to know which norm out of X and Y survives when domestic and foreign country agents coexist in the population. We used the dynamics-based methods and the agent-based best response function approach to answer this question. We explored if the results from replicator dynamics hold when evaluated against the best response approach, where agents can decide what choices to make and are connected via social networks. Different dynamic results show that when both domestic and foreign agents are less social, the convergence happens towards extremes; all agents follow domestic norms or foreign norms. If both agent types are more social, the population would have an equitable region of domestic and foreign norms agents. If one of the agent types is more social than the others, the social agent types dominate the population. On the other hand, when agents decide what actions to follow using the best response framework, we can expect to see both domestic and foreign norms co-existence in most situations when agents are connected in small-world networks. Network structures play an important role in the emergence of norms. A higher dense network usually leads to convergence towards one outcome. Results are sensitive to the fat-tailedness and clustering of the network in case of lower dense networks. Medium to heavy fat-tailedness can lead to convergence towards one outcome (X or Y) in the absence of high densities. When agents are connected in Barabasi-Albert network structures, results primarily dependent on the strategies followed by "hub" agents.

Specific Contributions

Norms as a subject area have been discussed across multiple disciplines, like economics, sociology, and anthropology. The existing literature has focused on how norms evolve and depend upon various parameters. Some of these parameters include the payoff structure of the game, population size, memory length, strength of relations with other agents, time taken to reach a norm, methods which agents follow to update their actions during each period of the game, and randomness in agents' actions (Young & Foster, 1991; Kandori et al., 1993; Young, 1993; Young, 2015; Alexander, 2007). The literature on norm evolution has been divided into two broad categories, one which approaches norm evolution using theoretical or analytical models and another which uses simulation or computational methods. The focus of our research is on norms evolution using computational methods. The computational methods provide the ability to test different possibilities with respect to multiple parameter combinations and can be easily scalable to incorporate other suggestions from different researchers.

In the third chapter, we tested the norm evolution in the context of a few selected finite normal-form games. Prisoner's dilemma results show that when agents are allowed to make mistakes, then (cooperate, cooperate) or (cooperate, defect) also has chances of emerging as the norm pair. This posits the possibility of sustaining a non-Nash and Pareto superior outcome as a norm. Games involving multiple pure strategy Nash equilibria like stag hunt, battle of sexes, and coordination game results depend upon the initial history, memory length of the agents, payoff values etc. Matching pennies game results did not show any clear trend. In general, results show there is a possibility of outcomes that can emerge as a norm that are neither Nash equilibria nor Pareto

efficient. It reinforces the view from Young (1993) that any strategy or action has the potential to emerge as a norm. It is driven by the dynamics followed and not necessarily due to that strategy being superior to others. This framework helps to explore different potential path dependencies depending on the initial states, agents' payoffs which can help formulate and design appropriate policy interventions.

The fourth chapter outlines the norm evolution framework when agents are connected in a social network. We helped explain this with the help of the Naming game of Young (2015) and the Nash demand game of Axtell et al. (1999). Results from the Naming game show that most of the population proposes 1 or 2 names in the case of a ring network with at least 1 name satisfying the norm criteria. In small-world networks with *shortcuts* available, emerging norms are not necessarily locally concentrated. This implies agents following norms proposed by those agents where there is no direct linkage among agents exist. The complete network results show convergence towards one name. Nash demand game results show convergence towards medium (M) outcome across most network structures when we assume the initial state of agents is distributed in the ratio of 40/40/20 (High/Medium/Low). These results outline the importance of network structures in determining norms. In the case of Barabasi-Albert network which has a presence of "hub" nodes, it implies leveraging influential nodes in the society (e.g., community leaders) to spread desired norms. It helps revealing individual behavior varies depending on the network they are part of at any given time. It can also simulate the impact of policy interventions before implementation when there is a need to displace norms.

The fifth chapter compares the norm evolution results from the replicator and other dynamics with the agent-based individual payoff function approach using the migration game. Results show that different social network types influence agents' decisions to follow domestic or foreign country norms. The replicator and other dynamics results show the convergence towards one outcome, domestic or foreign country norms when one agent type is more social than another. But the best response function approach shows the possibility of sustaining domestic and foreign country norms under some parameter restrictions. These results show agents' interactions with their neighbours and a continuous evolution of their beliefs based on their interactions with other agents influences the norms outcomes. The incorporation of social networks, agents having bounded rationality, and the flexibility of having somewhat unexpected agent behaviors makes the agent-based modeling approach more appealing to design policy interventions.

To ensure reusability, we have created three open-source Python libraries, which we believe are a significant contribution to computational economics literature. The intent of creating these libraries is to test different permutations of parameter combinations which can create several unique insights. These libraries can also serve as a base to expand further and make it more robust in terms of incorporating more complexities around different response functions, network structures etc. We believe that these libraries can be useful in numerous evolution applications under multi-agent framework which will add value to researchers working in economics, social science, anthropology, computer science etc. To the best of our knowledge, these are the first open-source Python libraries that can be used for this purpose by the end user with very little programming language. These libraries support customization with respect to different games, payoff matrices, response functions, memory length, network types, agent types, simulation time

periods etc. In addition, the output of these libraries is in the form of graphs and Excel files which provide detailed and granular information on agent's choices at every step of the simulation exercise. We believe that every user can make use of Excel files and hence is simpler to use and analyze. This also increases the transparency of the simulations performed, which we have found somewhat lacking in the current literature.

game-simulator python library is created to replicate the results for any finite normal form game payoff matrix. There are two main functions in the library, *simulation_function* and *simulation_function_payoff*. The first function generates results using the exhaustive best response approach, while the second function creates results using the expected payoff approach. The other two functions, *simulation_function_random,* and *simulation_function_payoff_random*, are replicas of these functions, allowing agents to make mistakes and choose actions randomly. In these functions, the parameter *random_multiplier* allows users to specify how recommended actions from the abovementioned approaches will be weighed against the rest of the choices. These functions produce output in different Excel files, which contain information about the strategy outcome played each time during the simulation window. These Excel files can further be used for analysis and producing different graphs.

We created a Python library, *multi-agent-coordination*, for the Naming game, which can be used to generalize results with any user-defined network and agent combinations. The function *network_simulations* have input parameters on the number of agents, neighbourhood size, network name, fixed agent ratio, probability of agents taking random response, and norm parameters. The output generated contains details on strategies being played and their frequency. It also has

information on fixed agents' strategies and where they are placed in the network. The strategies which satisfy the norm criteria are shown in the network graph by the end of the simulation period.

multi-agent-decision library, for which its functionality is demonstrated using the Nash demand game, can be used to generate results with any user-defined strategies and payoffs combinations. The function *simulation_function_neighbors* has input parameters on the number of agents, neighbourhood size, number of strategies, initial agents' strategy distribution, response function, network structure, and norm parameters. The output of executing this function is the revised agents' strategy distribution by the end of the simulation period. It can also be seen if the trend is reversed during the simulation period, where one strategy outweighs the other. The library can also explore the possibility of adding delta payoff and fixed agents.

Future Scope of Work

Before discussing the future scope of work, we list down some of the limitations of the study. First, we list some general limitations of using the agent-based modeling approach. Some of these limitations include difficulty in presenting results with respect to all permutations and combinations of multiple parameters, difficulties associated with calibrating and validating agent-based models using real-world data, the sensitivity of results with respect to parameter changes, and the challenge of restrictive assumptions to use the same in addressing the real-world problems. However, this approach is flexible and can be easily scalable to incorporate multiple parameter combinations, multiple agents, larger/denser social networks, etc., with enough computing power.

The literature has shown that multiple parameters impact norm evolution like payoff structure of the game, population size, memory length, the strength of relations with other agents, time taken to reach a norm, methods which agents follow to update their actions during each period of the game, randomness in agents' actions, among others (Young & Foster, 1991; Kandori et al., 1993; Young, 1993; Young, 2015; Alexander, 2007). We have tried to touch upon some of these parameters and included them in the simulation framework. We have listed below some of the limitations of this study by chapter and its corresponding future areas of scope.

The third chapter considers two response functions, exhaustive best response and expected payoff approach, which agents can use to decide what action to take. These response functions can be further expanded and enriched by adding more parameters that can impact agents' decisions.

Another area to explore is the feasibility of incorporating agents' social networks into this framework.

In the fourth chapter, we assumed agents have full memory in the Naming game. We have considered the limited functionality of social networks in terms of the number of parameters in the Python libraries created. We can expand these networks by adding more parameters on network complexity, agents' weights, and other types of social networks. We have assumed payoffs are constant across the simulation period, which can be variable depending on the strength of the agent's relations with other agents. We assume all agents are equal in assessing opponent agents' choices, but, in reality, we value some peoples' opinions more than others. We can tweak this and have different weights. We can relax the assumption of agents having full memory in the Naming game and assess results where agents have limited memory to check its impact. The four possibilities we have considered for response function can be further tweaked and enhanced to incorporate agent preferences.

In the fifth chapter, we assume that when agents meet with their own types, their payoff values remain the same as 1 in all the four possibilities presented. We can relax the assumption of equal payoffs when agents meet their own types and assess the impact on results. We also assume that agents do not make decisions randomly in the best response approach to have a fair comparison with results from replicator dynamics. We can also analyze results when agents make decisions with some randomness in their decision-making in the best response approach.

References

Akin, E. (1980). Domination or equilibrium. *Mathematical Biosciences*, 50(3-4), 239–250.

Alexander, J.M. (2007). *The structural evolution of morality*. Cambridge University Press.

Alexander, J. M. (2021). Evolutionary game theory. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy*. Summer 2021 ed. Metaphysics Research Lab, Stanford University.

Axelrod, R. (1984). *The Evolution of Cooperation*. New York: Basic Books.

Axelrod, R. (1986). An evolutionary approach to norms. *The American Political Science Review*, 80(4), 1095–1111.

Axtell, R. L., Epstein, J. M., & Young, H. P. (1999). The emergence of classes in a multi-agent bargaining model. In H. P. Young & S. Durlauf (Eds.), *Social dynamics* (pp. 191–211). Cambridge, MA: The MIT Press.

Axtell, R. L. & Farmer, J. D. (2022). "Agent-based modeling in economics and finance: Past, present and future" (No. 2022-10). Institute for New Economic Thinking. Working Paper. Retrieved March 8, 2024, from https://www.inet.ox.ac.uk/files/JEL-v2.0.pdf

Aydogmus, O., Cagatay, H. & Gurpinar, E. (2020). Does social learning promote cooperation in social dilemmas? *Journal of Economic Interaction and Coordination*, 15(3), 633–648.

Barnett-Howell, Z. (2017). "Should I stay or should I go? Microeconomic determinants of migration". University of Wisconsin – Madison. Working Paper. Retrieved June 9, 2023, from https://aae.wisc.edu/wp-content/uploads/2017/11/micromodel.pdf

Block, J. I., Fudenberg, D. & Levine, D.K. (2019). Learning dynamics with social comparisons and limited memory. *Theoretical Economics*, 14, 135–172.

Bowles, S. & Gintis, H. (1998). The moral economy of communities: Structured populations and the evolution of pro-social norms. *Evolution and Human Behavior*, 19, 3-25.

Brown, G. W. & von Neumann, J. (1950). Solutions of games by differential equations. In Kuhn, H. W. & Tucker, A. W. (Eds.), *Contributions to the Theory of Games I*. (pp 73-79). Princeton University Press.

Chatterjee, A., Rao, K.S.M. & Sarangi, S. (2023). Evolutionary stability for games played on networks. *Economics Letters*, 229, 111222.

Epstein, J. M. (2001). Learning to be thoughtless: Social norms and individual computation. *Computational Economics*, 18, 9-24.

Foster, D. & Young, P. (1990). Stochastic evolutionary game dynamics. *Theoretical Population Biology*, 38(2), 219-232.

Fudenberg, D. & Levine, D. K. (1998). *The theory of learning in games*. MIT Press.

Gallo E. (2014). “Communication networks in markets” (No. 1431). University of Cambridge. Working Paper. Retrieved May 9, 2023, from https://api.repository.cam.ac.uk/server/api/core/bitstreams/34ba0ff0-8446-41a3-823d-11c3a62dc004/content

Gebele, D., & Staudacher, J. (2022). *Evolutionary Games: Importance Concepts of Evolutionary Game Theory*. CRAN. Retrieved June 4, 2023, from https://cran.r-project.org/web/packages/EvolutionaryGames/index.html

Harms, W. & Skyrms, B. (2008). Evolution of moral norms. In M. Ruse (Ed.), *The Oxford Handbook of Philosophy of Biology* (pp 434 - 450). Oxford: Oxford University Press.

Hofbauer, J. P. & Sandholm, W. H. (2011). Survival of dominated strategies under evolutionary dynamics. *Theoretical Economics*, 6, 341–377.

Hofbauer, J., Schuster, P. & Sigmund, K. (1979). A note on evolutionary stable strategies and game dynamics. *Journal of Theoretical Biology*, 81, 609–12.

Kandori, M., Mailath, G. J. & Rob, R. (1993). Learning, Mutation, and Long Run Equilibria in Games. *Econometrica*, 61(1), 29-56.

Kohler, H. P., Bereman, J. R. & Watkins S. C. (2001). The density of social networks and fertility decisions: Evidence from south Nyanza District, Kenya. *Demography*, 38, 43- 58.

Martinez, R. G., Espallier, B. D., & Mersland, R. (2021). Bifurcations in business profitability: An agent-based simulation of homophily in self-financing groups. *Journal of Business Research*, 129, 495-514.

Nande, A., Ferdowsian, A., Lubin, E., Yoeli, E., & Nowak, M. (2020). *DyPy: A python library for simulating matrix-form games*. Arxiv. Retrieved July 24, 2022, from https://arxiv.org/pdf/2007.13815.pdf

Nishizaki, I., Katagiri, H., & Oyama T. (2009). Simulation analysis using multi-agent systems for social norms. *Computational Economics*, 34, 37-65.

Nowak, M. A. & May, R. M. (1992). Evolutionary Games and Spatial Chaos. *Nature*, 359 (6398), 826–829.

Nowak, M. A. & May, R. M. (1993). The Spatial Dilemmas of Evolution. *International Journal of Bifurcation and Chaos*, 3, 35–78.

Özgönül, M., & Kaplan, A. (2013). Immigration and unemployment application of game theory on Diyarbakir: Istanbul samples. In Stavrinides, S., Banerjee, S., Caglar, S. & Ozer, M. (Eds.), *Chaos and Complex Systems*. Springer, Berlin, Heidelberg.

Rocha, A.B.d.S. (2012). “Evolutionary dynamics of nationalism and migration” (No. 12/11). University of Leicester. Working Paper. Retrieved June 9, 2023, from https://www.le.ac.uk/economics/research/RePEc/lec/leecon/dp12-11.pdf

Sandholm, W. (2010). *Population Games and Evolutionary Dynamics*, MIT Press.

Savarimuthu, B., & Cranefield, S. (2011). Norm creation, spreading and emergence: A survey of simulation models of norms in multi-agent systems. *Multiagent and Grid Systems*, 7(1), 21–54.

Schlag, K. H. (1998). Why imitate, and if so, how? A Boundedly rational approach to multi-armed bandits. *Journal of Economic Theory*, 78, 130–156.

Sen, S., & Airiau, S. (2007). Emergence of norms through social learning. *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI)* (pp. 1507–1512). AAAI Press.

Shoham, Y., & Tennenholtz, M. (1992). Emergent conventions in multi-agent systems: Initial experimental results and observations. *Proceedings of the Third International Conference on the Principles of Knowledge Representation and Reasoning (KR)* (pp. 225 –231). Morgan Kaufmann Publishers.

Skyrms, B. (1996). *Evolution of the Social Contract*, Cambridge: Cambridge University Press.

Smith, M. J. (1984). The stability of a dynamic model of traffic assignment – An application of a method of Lyapunov. *Transportation Science*, 18, 245–252.

Swinkels, J. (1992). Evolutionary stability with equilibrium entrants. *Journal of Economic Theory*, 57, 306–332.

Taylor, P. D. & Jonker, L. B. (1978). Evolutionary stable strategies and game dynamics. *Mathematical Biosciences*, 40 (1-2), 145–156.

Tesfatsion, L. (2003). Agent-based computational economics: Modeling economies as complex adaptive systems. *Information Science*, 149 (4), 262-268.

Thomas, B. (1984). Evolutionary stability: states and strategies. *Theoretical Population Biology*, 26, 49–67.

Thomas, B. (1985a). Evolutionary stable sets in mixed-strategist models. *Theoretical Population Biology*, 28, 332–341.

Thomas, B. (1985b). On evolutionary stable sets. *Journal of Mathematical Biology*, 22, 105–115.

Uyttendaele, P. & Thuijsman, F. (2015). Evolutionary games and local dynamics. *International Game Theory Review*, 17(2),1540016.

Voss, T. (2001). Game-theoretical perspectives on the emergence of social norms. In Hechter, M. & Opp, K.D. (Eds.), *Social Norms* (pp. 105-138). Russell Sage.

Weibull, J.W. (1995). *Evolutionary Game Theory*, Cambridge, MA: The MIT Press.

Young, H. P. (1993). The evolution of conventions. *Econometrica*, 61(1), 57 – 84.

Young, H. P. (2015). The evolution of social norms. *Annual Review of Economics*, 7, 359 – 387.

Young, H.P., & Foster, D. (1991). Cooperation in the short and in the long run. *Games and Economic Behavior*, 3, 145-156.

List of Publications and Presentations

A. Conferences and Presentations

International Conference on Network Science in Management, 2022. Indian Institute of Management, Ahmedabad, India. December 17 – 18, 2022. Paper presentation entitled "Social Networks and Norms Evolution".

B. Publications

Title	Journal	Indexed	Current state	Publication link (if any)
Social Networks and Norms evolution	Computational Economics	Scopus	Published	Social Networks and Norms Evolution | SpringerLink
A Computational Approach to the Evolution of Social Norms	International Journal of Computational Economics and Econometrics	Scopus	Under Review	
Evolution and Stability of Social Norms				

Python Libraries created.

- game-simulator: https://pypi.org/project/game-simulator/
- multi-agent-coordination: https://pypi.org/project/multi-agent-coordination/
- multi-agent-decision: https://pypi.org/project/multi-agent-decision/

Brief Biography of the Candidate

Ankur is an experienced data science professional with over ten years of experience solving business problems for multiple Indian and multinational clients using data and machine learning. He has had the opportunity to work with multiple information technology (IT) service and product-based companies during his professional career so far.

Ankur holds a bachelor's degree in economics from the University of Delhi, a master's degree in economics from Jamia Millia Islamia, and an MPhil in economics (International Trade and Development) from Jawaharlal Nehru University. His research interests include computational economics, social science, machine learning, and artificial intelligence applications across disciplines and business problems.

Brief Biography of the Supervisor

Dr. Dushyant Kumar is a faculty member of the Department of Economics and Finance at the BITS Pilani, Hyderabad campus. Before joining BITS, he was a post-doctoral fellow at the Centre for Development Economics of the Delhi School of Economics (DSE). He has completed Ph.D. in Quantitative Economics from the Indian Statistical Institute, Delhi Centre. His research and teaching interests lie in game theory and microeconomics. Currently, he is working in the areas of conflict theory, organization theory, and behavioral economic theory. He has published papers in leading journals like Economic Modelling, International Game Theory Review etc. He has taught courses like microeconomics, game theory, industrial economics, behavioral economics, etc., at undergraduate and postgraduate levels.