

Learning is Not Always Good:
A Study of the Effect of Learning Within a
Simple Model of a Communication System

MSc. Dissertation

Juan Pablo Calderón
jpc22@cogs.susx.ac.uk
MSc. Evolutionary and Adaptive Systems
COGS-GRC

September 5, 2003

Abstract

Learning was implemented in the simple communication system developed by Werner & Dyer (1991). Our results show that although plastic individuals reach more complex and efficient communication protocols —than rigid individuals— which should aid them in the reproductive task, evolution selects for rigidity. Experiments were done varying the selective pressure on the artificial agents to study some of the costs and benefits of being plastic and how these affect the evolutionary process.

Acknowledgement

I will like to thank Boris Searles for lending me his laptop which was very useful in completing this work, also Inman Harvey, Ezequiel Di Paolo, Andy Phillipides and Emmet Spiers for sharing with me some of their knowledge throughout the course.

Contents

1	Introduction & Background	4
1.1	Baldwin Effect	4
1.1.1	The Cost of Learning	6
1.1.2	The Benefit of Learning	7
1.2	Werner & Dyer	7
2	Implementation	10
2.1	Model Details	10
2.1.1	Algorithm	10
2.1.2	Agents Structure	11
2.1.3	Agent's Input & Output	12
2.1.4	Genetic Algorithm	13
2.1.5	Varying the Selection Pressure	13
2.2	Reinforcement Learning	14
2.2.1	Q-Values Table	14
2.2.2	Exploration	14
2.2.3	Q-learning	15
3	Experiments	16
3.1	Selection Pressure	16
3.2	Evolution without Learning	17
3.3	Learning and Evolution	17
4	Results	18
4.1	Communication Protocols	19
4.2	Agents Reproduce Every Encounter	20
4.2.1	No Exploration	27
4.2.2	Exploration Without Learning	28
4.2.3	Plastic Agents	29
4.2.4	Genetically Specified Plasticity	29
4.2.5	Male and Female Agents Learning	29
4.3	Agents reproduce after 5 encounters	30
4.3.1	Non-Exploring Rigid Population	30
4.3.2	Exploring agents without Learning	30

4.3.3	Plastic Population	30
4.3.4	Genetically specified plasticity	37
4.3.5	Male and Female agents Learn	38
4.4	Agents reproduced after 15 encounters	38
4.4.1	Rigid Non-exploring Population	38
4.4.2	Exploring Population	38
4.5	Plastic Population	38
4.5.1	Plasticity is genetically specified	38
5	Analysis	45
6	Discussion	55
A	PROGRAM CODE	61
B	MATLAB CODE	87
B.1	plot_protocols.m	87
B.2	sameKind.m	89
B.3	scatter_plot.m	89

Chapter 1

Introduction & Background

This dissertation is a study of the effect of learning within a simple model of a communication system. The basic communication model we use here (without any prejudice) is Werner & Dyer's (1991) (hereafter abbreviated to W&D).

This project is an investigation of the way phenotypic plasticity affects evolution. By modifying an artificial world first developed by Werner and Dyer (1991) (W&D) to include plastic agents it is going to be shown that there are tradeoffs between the cost and benefits of being plastic.

In this introductory chapter the Baldwin effect is going to be presented, some of the costs and benefits of learning are going to be mentioned, and the artificial world developed by Werner and Dyer as an attempt to study the evolution of languages will be briefly described. The second chapter will explain the details of the model and what changes had to be made to adjust Werner and Dyer (W&D) (1991) to the aim of this project. Chapter 3 is a description of the experiments, chapter 4, 5 and 6 correspond to results, analysis and discussion respectively. Hopefully by then the reader will have better insight of how the interaction between learning and evolution works.

1.1 Baldwin Effect

At the beginning of the 20th century there was still some controversy whether Darwin's theory or Lamarck's better explained biological evolution.

In Lamarckian evolution traits acquired during the individual's lifetime are directly inherited to subsequent generations. After the work by August Weismann (1883) this line of thought lost its popularity. He refuted that acquired characteristics, like somatic characteristics, were transmitted physiologically through the germ line¹.

¹According to Weismann (1893) an organism consists of a soma which comprises most body parts and organs, and the germ plasm which contains the cells that give rise to the gametes. The fact that germ plasm segregates from the soma at early stages of the egg made Weismann to assert that the inheritance of acquired characteristics was impossible.

James Mark Baldwin (Baldwin, 1896) proposed “a new factor in evolution” to explain how characteristics initially acquired by the individual during its lifetime could become genetically specified in subsequent generations without recurring to a Lamarckian mechanism. Osborn (1896) and Morgan (1896) described similar mechanisms at the same time, unfortunately for them Baldwin got most of the credit.

Phenotypic plasticity is the “new factor”. It refers to the ability that an organism has to adapt to the environment around it during its own lifetime.. Possible adaptations include lifetime learning and any physical change (adaptation) that an individual could have obtained during its lifetime as well, e.g. the ability to form a callus when the skin is constantly rubbed.

For the purpose of this essay the terms phenotypic plasticity, ontogenetic adaptations, lifetime learning, acquired characteristics, acquired behaviours are going to be used interchangeably since they all have a similar relation with the evolutionary process, and almost everything said here applies to all of them equally.

Baldwin introduced the term *organic selection* for the process in which individuals that are able to acquire a beneficial trait will be selected for. Thus, this ability of acquiring the trait is going to be inherited into subsequent generations.

The Baldwin effect is divided into two steps. In the first step individuals capable of acquiring a beneficial trait during their lifetime will have a better chance of inheriting that ability to a later generation, hence there is going to be an increase of plastic individuals on the population that have the ability to acquire that particular beneficial trait. What is selected for in this first step is the ability to acquire the trait and not the trait itself. The second step begins once the majority of the population is able to acquire the trait, then individuals that can obtain the trait with less evolutionary cost will start taking over the population until the trait becomes genetically assimilated (Waddington, 1942). “Thus, credit for the origin of adaptations could thus be given to creative individuals and not to inheritable adaptations” (Mayley, 1996).

The Baldwin effect is purely Darwinian. It might be Lamarckian in its results but not in its mechanisms.

Turney in a good position paper (Turney,1996) explains some of the myths and misinterpretations of the Baldwin Effect. He says that since Hinton & Nowlan’s (1987) work computer scientists have focused on studying the benefits of learning, and how this learning speeds up the evolutionary process, but they have forgotten about the second part of the Baldwin Effect where because of the cost of learning evolution selects for rigidity and phenotypic acquired features can become assimilated.

The Baldwin effect is concerned with the evolutionary costs and benefits of phenotypic plasticity. Also the interaction between learning and evolution is not beneficial if the learning task is not correlated with the evolutionary task. (Menczer & Belew (1994)). Learning doesn’t always speed up the evolutionary process.

Phenotypic plasticity has a cost as well as a benefit, and it is this cost that is responsible for selecting the trait in the second step of the Baldwin

Effect. Sometimes an instinct behavior is more reliable than a learned behavior. The right examples might simply not be available to the individual. However, learning can increase the variety which can facilitate evolution under certain circumstances.

Phenotypic plasticity has some evolutionary advantages but it also incurs some cost. Sometimes having an instinct behavior might turn out to be a better strategy than having to acquire the behavior during your lifetime.

Turney (1996) points out that evolution might not always follow the path suggested by Baldwin and that Baldwin's lesson is that some trade-offs between plasticity and rigidity do exist. Evolution seeks an optimal balance between learning and instinct by trying to keep a net evolutionary benefit. Further, this balance point changes throughout the evolutionary process. Costs and benefits change because the environment, the individual's behaviours and the relation between them also change.

A good review of the costs and benefits of having phenotypic plasticity can be found on Johnston (1982) or Mayley (1996a). I'm going to mention some of the most important ones.

1.1.1 The Cost of Learning

Learning is not always good because it can be expensive in different ways. Some of the costs involved in phenotypic plasticity are going to be mentioned below. For a more detail explanation refer to the references above.

There is a time-wasting cost during the learning process where the individual is not performing at its best possible way because he has not yet reached its optimum level. If another individual posses the same innate behavior it will perform better in overall because it doesn't spend time learning the specific behavior but can start performing at its maximum level right away.

Sometimes individuals are not able to reproduce during their learning period which is obviously an evolutionary disadvantage. The longer the learning period is, the greater the disadvantage.

Learning also implies some energy consumption that could be better used on other beneficial tasks.

Learning has also a stochastic element. Learning a specific behavior depends on the individual's experiences within the environment. If an individual is unlucky and doesn't find the right example he might learn an inappropriate behavior or even cause some damage to itself affecting its performance during an undefined period of time. In biology this types of costs are reduced by child play or parental care (Fagen, 1981).

There is an energy cost in developing a complex regulatory system for the plasticity of learning (Johnston, 1982). This cost is independent of the length of the learning period.

There are some costs that don't affect the plastic individual directly, but do affect the evolutionary process. An example of this type of cost is the time and energy wasted during parental care or guidance. The time parents spend teaching their offspring could be used in other beneficial tasks such as feeding

or reproducing. Another example is that genotypes have to increase their complexity to be able to encode for all the regulatory changes that take place during the learning period (Johnston, 1982). This increases the dimensionality of the search space making it harder for evolution to find a good solution.

On artificial evolution there is a computational cost. Programmers spend time developing and testing their learning algorithms, also including lifetime learning (or local search) in evolutionary algorithms increases the computational time which in some cases might not be worth doing.

1.1.2 The Benefit of Learning

Some of the benefits of learning are going to be mentioned below.

One benefit of being plastic is that plastic individuals are able to adapt to changes of the environment on an individual's life time scale. Evolution works on an evolutionary time scale and can't notice changes of the environment over an individual's time scale. (Johnston, 1982., Todd & Miller, 1991)

Adaptive individuals can also exploit local environments. Each individual can take advantage of local environmental circumstances (Johnston, 1982). Evolution alone works on a global spatial scale.

Information can be stored on the environment. Individuals can exploit environmental regularities to acquire the required beneficial behaviours instead of having them encoded on their genotypes. By exploiting this regularities plastic individuals might end up with more complex behaviours for a given genotype size (Todd & Miller, 1991).

Plasticity also helps maintain some genetic diversity. Same behaviours can be achieved from genetically different individuals, which will reduce the selection pressure and keep a less converged population (Maynard Smith, 1993).

Evolution is guided by the actions that the individuals of the population take. This is Baldwin's idea of organic selection, where adaptive individuals that acquire an advantageous trait during their lifetime would be selected for their ability to acquire that specific trait, and can later be assimilated by the population.

Phenotypic plasticity can help maintain a particular beneficial trait that is not genetically present on the population. Evolution depends on stochastic methods to maintain some genetic variation, and if this genetic variation is not present on the population for the evolution of that particular trait then learning can help maintain that trait.

1.2 Werner & Dyer

In an influential paper Gregory M. Werner and Michael G. Dyer (1991) evolved simple communication protocols for mate finding in an artificial world using artificial evolution.

The world consisted of a 2-dimensional toroidal grid (200 x 200) with blind male and stationary female agents. Male agents received signals from nearby

females and could perform one of four possible actions, move forward, turn right, turn left or stand still, while stationary females had the ability to detect the position and orientation of nearby males and emit one of four different signals.

Every agent carried genetic information of both the male and female parts. The genetic algorithm used by them was such that whenever two agents of opposite sex meet i.e. the two agents stand on the same grid square, they mate and produce 2 offspring which will replace one random male and one random female from the population —possibly one of the two parents. Offspring’s genotype is determined by the common genetic operators — every gene on their genome can be mutated with a 0.01 % chance and 2 % crossover per gene between the parents. The four agents are randomly placed back on the grid.

This world was design to favour individuals that agree upon the signals meaning i.e. use the same communication protocol. The task is to co-evolve a population who agree on the same signal interpretation. Agents that follow the same signal protocol are going to be able to find a partner and reproduce more efficiently, and hence their genetic information has more chance of passing into the next generation.

In this world communication is not a requirement for reproduction, but those individuals that do communicate are going to be more efficient in finding a mate.

On a first experiment agents’ actions are controlled by a recurrent neural network with genetically determined weights.

Males’ neural network receive a binary input signal coming from the nearest female. The network’s output is then traduced into one of the four possible actions. There are 4 output neurons and a winner-takes-all mechanism is used to pick the male’s action.

Females receive as input the position and orientation of the closest male within her visual field which consists of 5x5 squares centred at the female. Their output is a 3-digit binary signal which can be interpret as a sound and is copied directly into the inputs of all nearby males. However, males only pay attention to the closest female. The paper is not more specific about the network composition, nor about how the females’ input was encoded.

On a second experiment agents are turned into simple pattern transducers i.e. a lookup table. In this case the output that an agent produced upon receiving a certain input is encoded directly on the genome. Agents carry genetic information of both the female and male parts and it is the agent’s sex which determines which part of the genome is interpreted to construct the table.

Some of their reasons for designing the model in the way they did are:

- Males can’t locate the sound source so that they need to interpret the signal and not just follow the direction from where it came from.
- There should be no direct pressure on the animals to communicate. Agents should be able to solve the task without using communication. Communication should only aid the agents gain a better efficiency.
- It is important in the evolution of language to allow overlapping generations for inter-generational communication.

- Agents chose their mate directly by signalling each other i.e. no fitness function.
- The signals emitted by the females should have a meaning for the males to interpret.

Werner & Dyer's work has been criticized of not being biologically plausible. Hence, it can't tell us much about life itself (Bullock, 1997., Noble, 1998.). This project is focusing on the problem of how phenotypic adaptations affect the process of evolution. It is not meant to 'exactly' illustrate the evolution of communication in biological systems, but is an investigation on some of the tradeoffs between the costs and benefits of phenotypic plasticity.

Chapter 2

Implementation

A model very similar to Werner & Dyer's (1991) second experiment was implemented to investigate the effect that allowing plasticity among the individuals of an evolving population has over the outcome of the evolutionary process.

The W&D model was implemented as it was interpreted, some changes that were introduced will be described and explained along this chapter. The world in which agents move is pretty much the same as the one used by them. Changes were made solely upon the nature of the agents to allow exploration and plasticity, also to vary the selection pressure or 'aggressiveness' of the environment.

2.1 Model Details

2.1.1 Algorithm

The pseudo-code for a typical simulation run is illustrated below. Some of the important details are going to be mentioned along the chapter.

```
initialize all agent's position, genomes, orientations, etc;

run till stopping criteria

  /* female loop */
  for every female
    check distance to every male;
    if distance = 0 (male is over female)
      run GA and reward both agents, female's input=0, exit loop;
    else
      get input according to the closest male;
      compute female's output;
      if learning is on update Q-values;

  /* male loop */
```

```

for every male
  check distance to every female;
  if distance = 0
    \* male has just randomly appeared over female
      by product of a GA */
    male's input=0, exit loop;
  else
    copy closest female's output into male's input;
  compute male's output;
  move or rotate male according to its output;
  if learning is on, update Q-values;

```

2.1.2 Agents Structure

The agents used in our model have similar properties as the ones used in W&D (1991).

They are structured in the following way:

- Each agent has a two coordinate *position* on the grid world.
- Male agents have an *orientation* which takes values of [0, 1, 2, or 3] standing for north, east, south, or west, respectively.
- It is the agent's *sex* what determines which part of the genome is interpreted to build the agent's Q-value table that is used to compute the agent's behavior. (section 2.2.1).
- Agents genetic information as in W&D is encoded in a *genome* that consists of 2 parts, a female part and a male part. Genes on the initial population are random float numbers in the range [0,12]¹. On some of the experiments agents were also given a learning gene so that learning could be genetically specified. It is an on-off binary gene determining if a newborn agent had the ability to adapt during its lifetime or if it is going to be rigid.
- Agents grow older every time step. They are born with an *age* of 1 and every time they take an action i.e. every turn, their age increases by one. This feature was first included in the model to determine the amount of exploration that each agent does on the environment, but it is now used for analysis purposes.
- To determine the amount of exploration the agent's age was replaced by the number of encounters with an opposite sex agent. The number of times that an agent has run into a mate can be analogous to the *maturity* of

¹This range was chosen arbitrarily, taking into account that it was big enough to differentiate between the four possible actions for the individuals to perform.

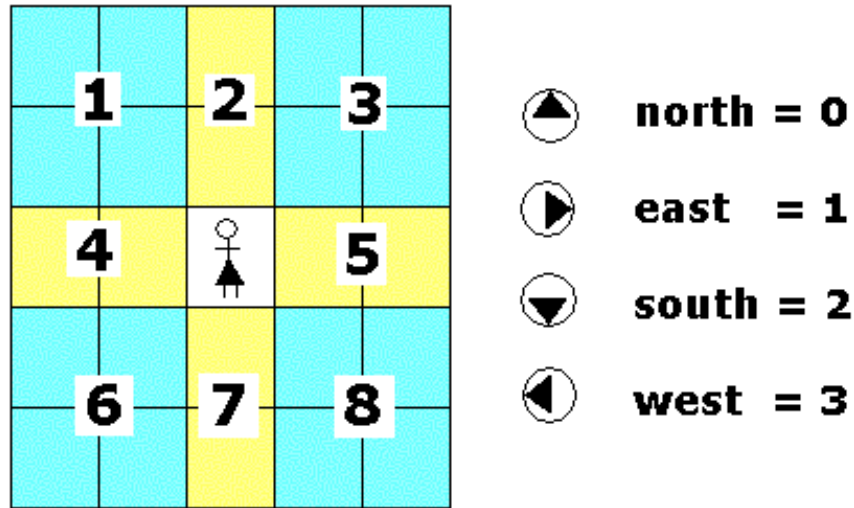


Figure 2.1: Diagram showing the different sections of the female’s visual field where she can detect a male agent, and the possible orientations that a male agent can have.

the agent and is the variable used to determine the amount of exploration that the agent performs. This will be explained in detail in section 2.2.2.

- Every turn an agent receives an *input* which is used to compute an *output* (Section 2.1.3).

2.1.3 Agent’s Input & Output

A female agent detects the direction and orientation of the closest male agent that lies within her visual field — a 5x5 square grid centered at the female. Given that 2 agents are at the same distance, the first one in the agents array is going to be selected.

A female’s input is an integer number between 0 and 32 that is computed with the direction and orientation of the detected male in the following way

$$input = direction + (8 * orientation) \quad (2.1)$$

where direction and orientation take the integer values shown in figure 2.1. If there is no male agent within the female’s visual field the input is 0 and the female agent still emits an output.

A female’s output is an integer number from 0 to 3, where 0 is interpreted as a silence —since it is the default male’s input— and the other 3 values are up to the evolutionary process to interpret.

Sex	State / Inputs	Action / Outputs
Male	4 Possible states corresponding to each one of the signals it can receive [0,1,2,or 3].	They can take one of four possible actions. Move forward, turn left, turn right or stand still.
Female	33 Possible states calculated using equation 2.1 with the direction and orientation of the closest male. They receive 0 if the visual field is empty.	4 different signals can be emitted [0, 1, 2, or 3]. 0 is interpret as silence.

Table 2.1: Summary of the inputs and outputs that agents take each turn.

A male agent is going to hear the signal that the closest female within its hearing range is emitting and will output an integer number (0, 1, 2, or 3) standing for move forward, turn right, stay still or turn left respectively. When a male turns no new ground is covered, it consists of a 90 degrees rotation. The default signal that a male agent receives when there isn't a female nearby is 0 (a silence). Male's hearing range is the same as a female's.

Table 2.1 gives a summary of the inputs and outputs that agents can have.

2.1.4 Genetic Algorithm

W&D's world is very dynamic and aggressive since any "fit" agent can randomly die any instant. The genetic algorithm (which W&D call XGA) generates 2 new offspring—one of each sex—every time 2 agents of the opposite sex meet, and kills 2 random individuals among the population—probably one of the parents—so that the population size doesn't increase or decrease.

The common genetic operators are applied to every offspring's genome. There is 0.01% chance of mutating each gene, and a crossover rate of 5% per gene. A mutation consists of tossing the gene again.

2.1.5 Varying the Selection Pressure

In order for phenotypic adaptations to be successful some time is needed for the individual to organize its internal structure. To give some time for learning to take place the selection pressure was reduced by allowing agents to produce an offspring only if they had previously run into a partner a certain number of times i.e. mature enough. This number is a parameter that will be used to vary the selection pressure within the experiments.

Given the case that only one of the two agents involved in reproduction was mature enough to mate, only one offspring would be produced of the same sex as the 'mature' parent and replaced a random agent of the same sex within the

population

2.2 Reinforcement Learning

“Reinforcement Learning is the learning of a mapping from situations to actions so as to maximize a scalar reward or reinforcement signal.” (Sutton & Barto, 1998). In reinforcement learning the agent must discover the actions that yield the most reward by trial and error. The agent has to search the environment for good actions and needs some sort of memory to remember which actions work well in what situations.

There is a trade-off between *exploitation* and *exploration* in every reinforcement learning algorithm. On the one hand, to get rewards the agent must exploit what he already knows about the environment, and on the other hand, it has to explore the environment and try out actions he hasn’t tried before so as to discover which of these actions are beneficial and which are not.

Q-Learning—a form of reinforcement learning—was implemented in the model to allow agents to be plastic by replacing the simple pattern transducer used by Werner & Dyer with a Q-value table.

2.2.1 Q-Values Table

Each agent is born with a Q-value table that is constructed by directly mapping the agent’s active part of the genome into the rows and columns of the table. For female agents the table has 33 rows—one for each of the states—and 4 columns—one for each action. For males the table has 4 rows and 4 columns, since it can be in one of four different states and perform one of four actions.

Every position of the table gives the Q-value due to following a particular action a given that the agent is in a particular state s . A Q-value $Q(s, a)$ is a measure of the expected benefit or satisfaction that the agent would obtain from following a respective action a for a given state s . A Q-value by itself doesn’t tell you anything. You have to compare all the Q-values for a particular state to be able to decide which action brings you the most benefit.

2.2.2 Exploration

To partially solve the trade-off between exploitation and exploration, an agent in a state s has a probability of following a particular action a given by a Boltzman distribution function of the Q-values for that particular state s .

$$p_s(a) = \frac{e^{\frac{Q(s,a)}{T}}}{\sum_b e^{\frac{Q(s,b)}{T}}} \quad (2.2)$$

where the temperature T is a variable used to reduce exploration as the agent gets older and acquires more experience of the environment. Each agent has its own temperature which decreases each time he finds a mate.

Temperature T varies according to

$$T = (T_{max} - \Delta T(M)) \quad (2.3)$$

where M is the number of rewards that the agent has accumulated so far, $T_{max} = 10$ the maximum temperature and $\Delta T = 1$ is the change in temperature. Values were chosen so that each agent would receive 10 reinforcement signals during the exploration period. When the temperature reaches zero the action taken by the agent would be the one of highest Q-value. Temperature will not decrease below zero but remain there.

2.2.3 Q-learning

The Q-values of plastic individuals are updated accordingly to the typical Q-learning algorithm (Watkins (1989), Sutton & Barto (1998)).

Every time the agent is in a state s and executes an action a the respective Q-value $Q(s, a)$ is updated according to

$$Q(s, a) = (1 - \alpha)Q(s, a) + \alpha(r + \gamma \max_{b \in A} Q(s', b)) \quad (2.4)$$

$$\Delta Q(s, a) = \alpha([r + \gamma \max_{b \in A} Q(s', b)] - Q(s, a)) \quad (2.5)$$

where α is the learning rate, γ is a parameter which specifies the weight that the maximum next Q-value has, r is the reward given for following the action a , s is the current state, s' is the state that the agent is going to be in given that he follows action a , and $\max_{b \in A} Q(s', b)$ is the maximum Q-value of the next state s' .

Chapter 3

Experiments

Several experiments were carried out varying the selection pressure and the nature of the agents involved in the evolutionary process to investigate the effect that plasticity has on the population performance. The number of individuals is 1600 males and 1600 females. It is the same number that W&D use on their second experiment ¹. With this number each female will have roughly 25 squares for herself, which is equal to the size of her visual field as in W&D (1991).

The world has exactly the same topology and size. Other parameters used during the experiments will be listed in table 3.1.

Simulations were stopped as soon as the total number of mates in the whole population reached 5000000 (or 3125 on average per individual), to allow enough time for the population to converge. Stopping the simulation at a constant number of time steps is not a good stopping criterion since the selection pressure varies among the experiments making it harder for the agents to reproduce.

Parameters within the physical world, the GA, the exploration and the Q-learning algorithm were frozen during experimentation, values are listed in table 3.1.

3.1 Selection Pressure

The selection pressure was varied —as explained in section 2.1.5— to see the effect of plasticity on different environmental circumstances.

The minimum number of encounters required for agents to be able to reproduce took values of [1, 5, and 15]. These 3 values as explained before were introduced to make it more difficult for the agents to reproduce, therefore plastic individuals could exploit more their exploration period and rearrange themselves properly.

Most of the experiments described below were repeated for all the values of selection pressure mentioned above.

¹Werner & Dyer don't mention it explicitly but it can be deduce it from their results table.

Parameter	Value
Grid size	200 x 200
Number of males agents	1600
Number of female agents	1600
Crossover	0.5%
Mutation Chance	0.01%
Learning rate α	0.1
γ	0.98
T_{max}	10
ΔT	0.1

Table 3.1: Table showing the parameters which remained constant in every experiment.

3.2 Evolution without Learning

In order to compare the effect of plasticity on an evolutionary run two different experiments were made. On the first experiment agents were allowed to explore the world as described in section 2.2.2, but were not able to adapt their Q-values. In the following sections this type of agents are going to be referred as ‘exploring’ agents even though plastic agents also explore the environment. A second experiment was made where agents were absolutely rigid and followed the action dictated by the highest Q-value. Since they don’t explore the environment they are going to be referred as ‘non-exploring agents’. This last experiment was made to investigate the evolutionary cost of exploration. It is important to note that exploration is one of the main costs involved in learning.

All this experiments were done for the 3 different values of selection pressure and non-exploring females.

3.3 Learning and Evolution

On another set of experiments agents were subject to a genetic algorithm and learning in the following ways:

- Every single agent —male and female— was allowed to learn. This is the only experiment where females adapt to the environment. In all other experiments the females are rigid and don’t explore the environment.
- Only male agents were allowed to learn. Female agents were rigid and didn’t explore the environment either. These agents are going to be referred as ‘plastic’ agents.
- Learning was genetically specified as it was explained in the previous chapter. Although both male and female agents could carry an ‘ON’ learning gene it was only active on males i.e. only males adapted.

Chapter 4

Results

Some interesting results are going to be described along this chapter and analyzed in the next.

The simplest search strategy that male agents can exercise to find a mate is to go straight and eventually run into a female. More elaborate strategies will involve taking advantage of the signals that female agents emit simply because they have some information about the world which males don't. In order for them to cooperate they first have to agree on what each signal mean. Males would have to interpret the signal in the correct way to avoid being steered away, but females also have to know which of the signals would make a male behave the way she pleases.

To be able to visualize the behavior of the population, information about the behavior of each male was encoded into a four digit 'communication' protocol which specifies the male agent reaction toward each of the possible signals it can receive. It is done in the same way W&D (1991) did, so as to complement and confirm their results, but also because it is an appropriate way of condensing behavior into a single manageable number.

By doing this we are interpreting that female agents direct male agents toward themselves, and makes sense since it is the only way a female can aid a male get to her with the current model constrains.

We have to remember that what females are doing is difficult to visualize since they have 33 inputs, which in terms of protocols would be around $4^{33} \sim 7 \times 10^{19}$ different protocols, impossible to analyze or compress into a manageable size. However, by studying the overall male behavior we can get a glance of what the female population is doing. Further, it is the male agents who are learning, making it appropriate for us to look closely at their behavior and not at the females'.

The first section of this chapter is an explanation of what is exactly meant by a communication protocol and how are they classified into different classes or categories. Subsequent sections would point out the behavior observed in the actual experiments.

4.1 Communication Protocols

Each male agent can be characterized by a 4 digit protocol which describes the action that the agent will take given he is in a certain state. Every protocol consists of four digits, one per each of the signals that a male agent can receive. The number on each digit is an integer from 0 to 3 which accord to the action—go forward (0), turn right (1), stand still (2), or turn left (3)—that the male will output upon receiving the signal corresponding to that particular digit. The digits are ordered from back to front, i.e. the first digit stands for signal ‘3’ and the last digit stands for signal ‘0’ (or silence)¹.

An agent described with the protocol 0000 will behave such that it will ‘go forward’ no matter what state it is in. For this particular case the male agent doesn’t pay attention to the signal that nearby females are feeding him. This class of protocols where agents interpret every signal in the same way is going to be referred as a ‘one signal protocol’ (1SP).

If an agent is described by the protocol 0100 it will ‘go forward’ upon receiving inputs [0, 1, and 3] and will ‘turn right’ upon receiving a signal ‘2’. Another agent can follow a ‘3003’ protocol, which means that he turns left when he hears signals ‘0’ and ‘3’, and will ‘go forward’ when hearing signal ‘1’ or signal ‘2’. On the two examples above there are 2 different actions (or ‘meanings’) that agents interpret among the 4 signals, and therefore they belong to the ‘two signal protocol’ (2SP) class.

A ‘three signal protocol’ (3SP) class is one that consists of three different meaningful signals, e.g. ‘0102’, ‘1301’ or ‘0320’. In other words the agents have a different interpretation—or make a different action—for 3 out of the 4 signals.

Finally there can be a ‘four signal protocol’ where agents have a different interpretation for each of the signals.

A 3SP protocol which includes meanings for both ‘turn right’ and ‘turn left’ signals is a more efficient reproductive strategy than a 2SP protocol with a meaning for one of them only. The 2SP can still solve the task since three left wise rotations are equivalent to one right wise, however, the time spent solving the task gets tripled. A 4SP includes a meaning for both ‘turn right’ and ‘turn left’, but also a for ‘stand still’ which is not very clever when you have to move in order to reproduce. Since agents are exposed to exploration and hence, there is a probability that an agent follows an action different from the one which—it believes— would bring him the most benefit, each digit on a protocol is determined by the maximum Q-value for the respective state.

This model differs from W&D’s in that agents are plastic i.e. they change during their lifetimes. Agents are born with an *innate* protocol and because of their plasticity and experiences acquired during their lifetime it will change to that of a *learned* behavior. Therefore, it is appropriate to talk of a *learned* protocol and an *innate* protocol. Mayley (1996) identifies 3 different fitness measures to calculate the costs and benefit that plasticity incurs on the individual over

¹It was computationally implemented this way, and kept like this to avoid confusion or having to turn every protocol upside down.

the evolutionary process. He talks of an *innate* fitness, *lifetime* fitness and a *learned* fitness and from these he is able to assign a value for the cost and benefit of being plastic.

In a W&D typical simulation run male agents that tended towards going forward reproduced better and became popular among the population, and those that remained still most of the time became extinct. At early stages of evolution paying attention to females can be dangerous since the protocols used by the female population are dispersed and therefore they can steer you away. If a large percentage of the female population is using the same signal for the same inputs, then males that interpret those signals in the right way can be steered by the females into them and reproduce. Further, male agents following similar protocols such as ‘0100’ or ‘0110’ would have a similar behavior and can interpret correctly more of the signals coming from the same female than say protocols ‘3000’ and ‘0300’ because on the first one they share more than one meaning—‘go forward’ and ‘turn right’—while on the second one they just share a ‘go forward’.

Note on Graphs

Graphs on this section show either the percentage of individuals that follow particular protocols or the accumulated percentage for the whole protocol class, against the average number of mates—meaning reproductions—per individual. This average is computed by accumulating all the reproductions that have taken place since the simulation started and dividing it by the number of agents. It was decided it is an appropriate scale since the number of reproductions is more significant than the number of time steps on an evolutionary process. The pressure on the agents and even the agents themselves change from one experiment to the other making it better to have a time scale with evolutionary significance rather than the individual’s lifetime time scale. Further, protocols that were used by less than 5 % of the population are not shown on graphs to avoid overcrowding—there are 256 different protocols in total—and for this same reason the population percentages at a given instant in time don’t always add up to one.

The sampling rate used while constructing the graphs changed according to the x-axis scale and also between graphs corresponding to each value of selection pressure i.e. reproducing every 1, 5, and 15 encounters. Graphs that complement each other were kept with a same sampling resolution.

Graphs were constructed with learned protocols unless stated otherwise.

4.2 Agents Reproduce Every Encounter

This section describes the results obtained for the experiments in which agents were subject of ‘high’ selection pressure i.e. reproduced every time a male ran into a female.

In most of the experiments there is a clear transition from a population

No Exploration - Reproduce Every Encounter

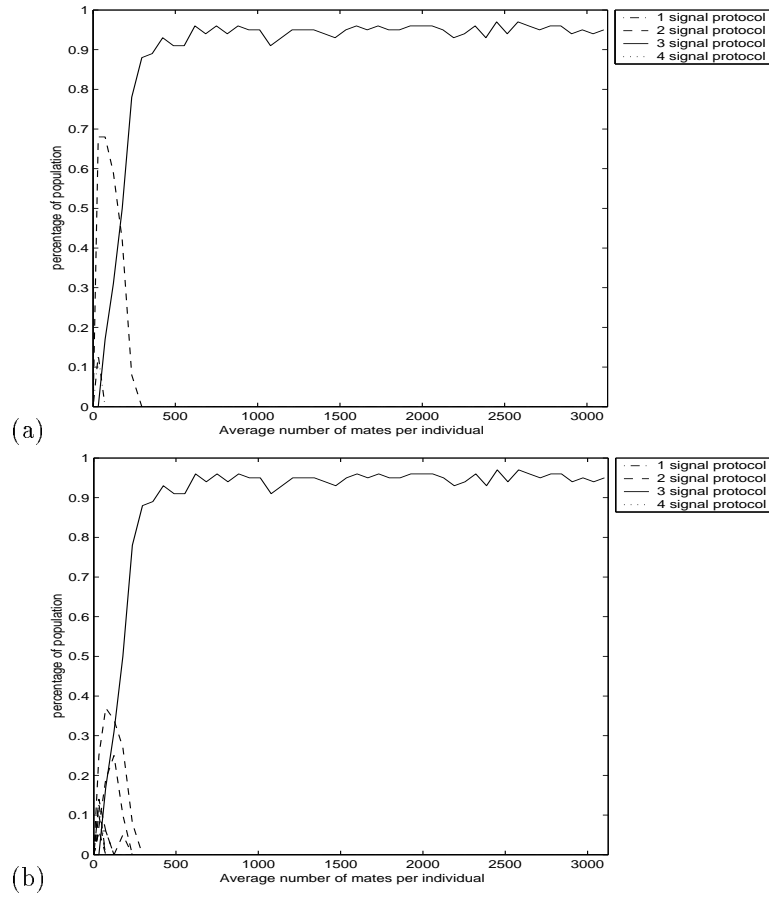


Figure 4.1: (a) Percentage of population belonging to each of the protocol classes and (b) percentage of male agents following one same protocol against the average number of reproductions per individual. On (a) each line is an accumulated sum of all the protocols belonging to that particular protocol class (see text for an explanation).

Exploration (no learning) - Reproduce Every Encounter

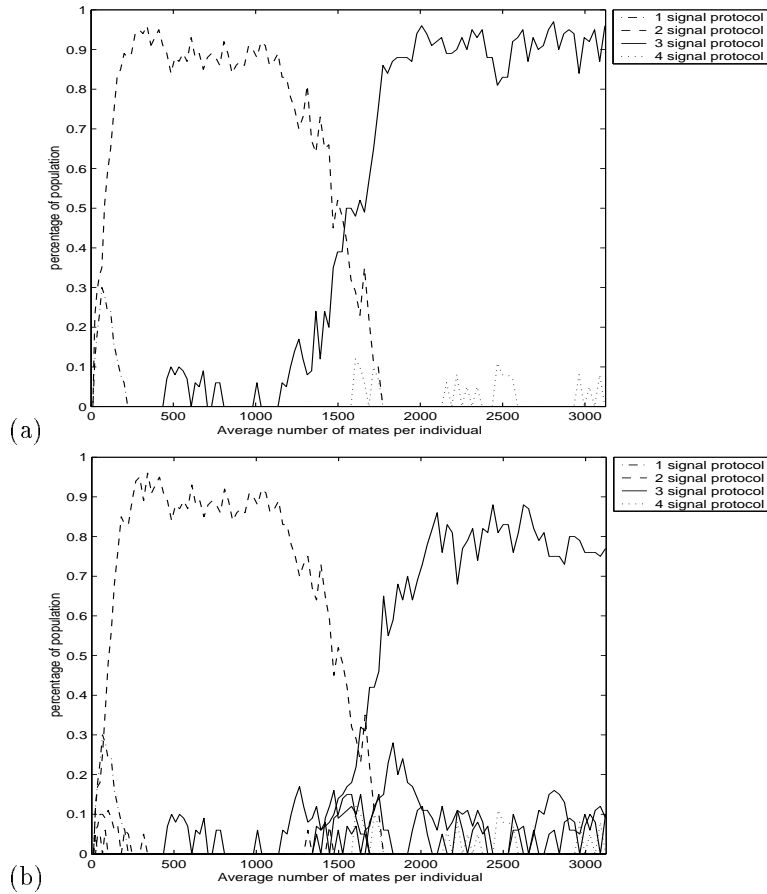


Figure 4.2: (a) Percentage of population belonging to each of the protocol classes and (b) percentage of male agents following one same protocol against the average number of reproductions per individual. On (a) each line is an accumulated sum of all the protocols belonging to that particular protocol class (see text for an explanation).

Q-Learning - Reproduce Every Encounter

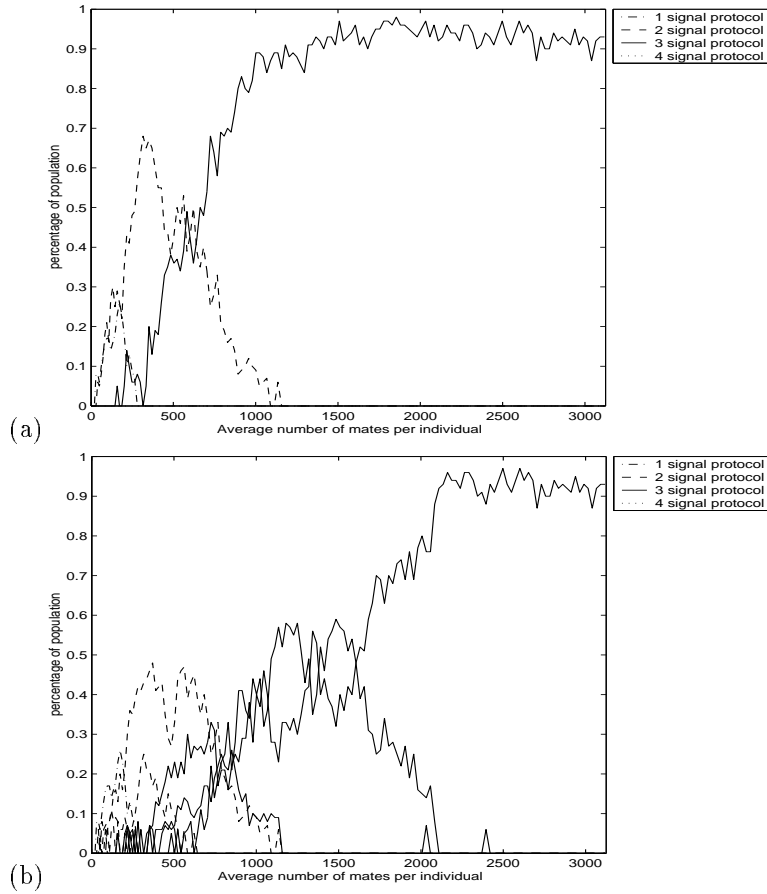


Figure 4.3: (a) Percentage of population belonging to each of the protocol classes and (b) percentage of male agents following one same protocol against the average number of reproductions per individual. On (a) each line is an accumulated sum of all the protocols belonging to that particular protocol class (see text for an explanation).

Genetically Specified Plasticity - Reproduce Every Encounter

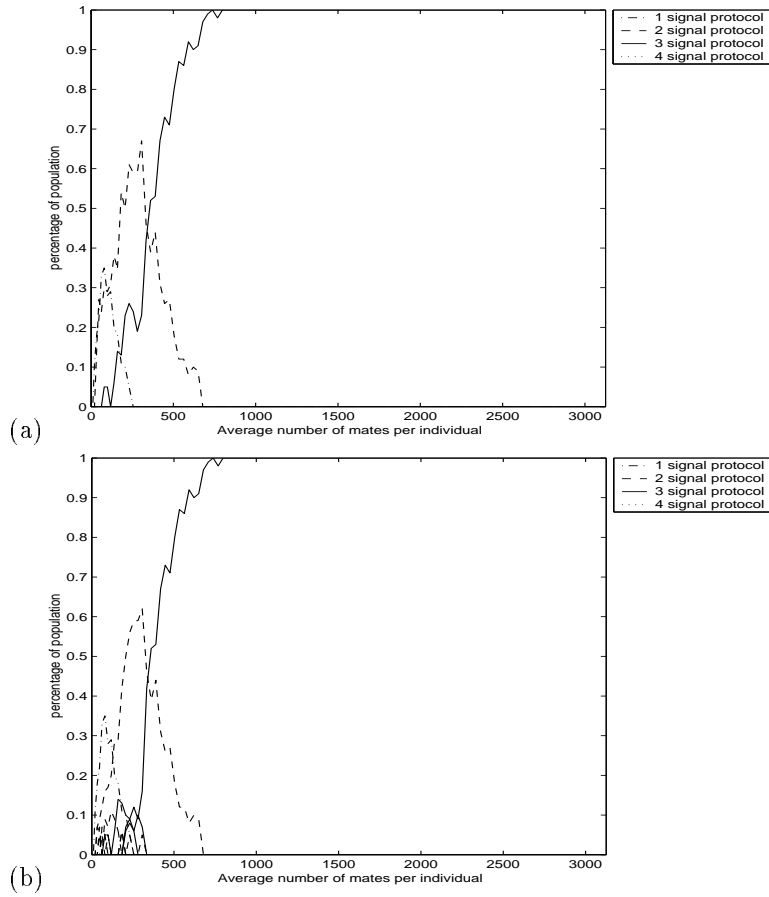


Figure 4.4: (a) Percentage of population belonging to each of the protocol classes and (b) percentage of male agents following one same protocol against the average number of reproductions per individual. On (a) each line is an accumulated sum of all the protocols belonging to that particular protocol class (see text for an explanation).

Male and Female Learn - Reproduce Every Encounter

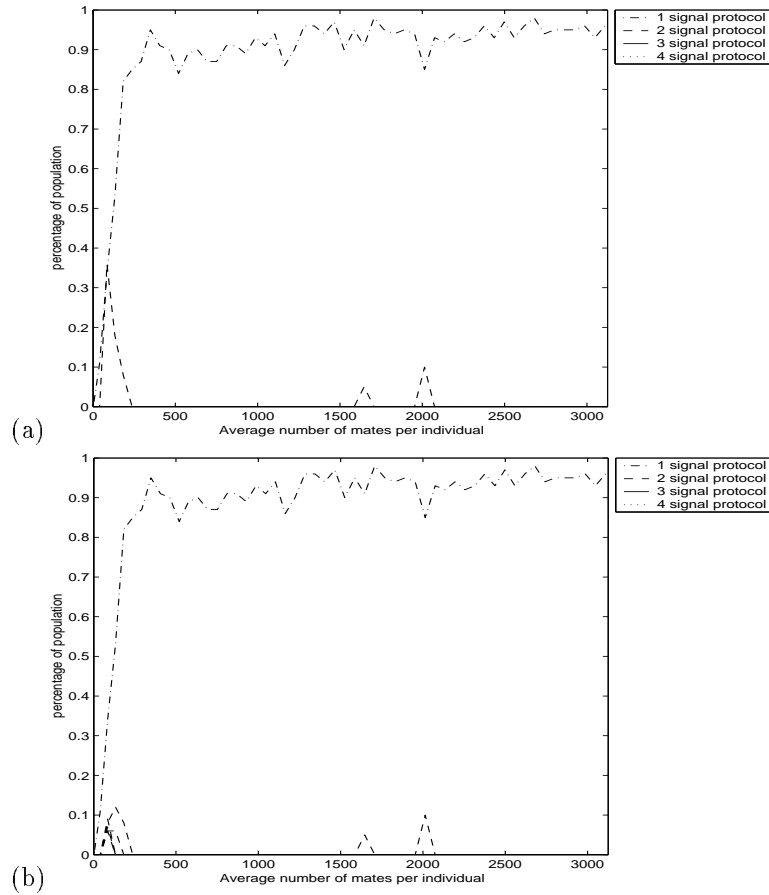


Figure 4.5: (a) Percentage of population belonging to each of the protocol classes and (b) percentage of male agents following one same protocol against the average number of reproductions per individual. On (a) each line is an accumulated sum of all the protocols belonging to that particular protocol class (see text for an explanation).

Reproduce Every Encounter

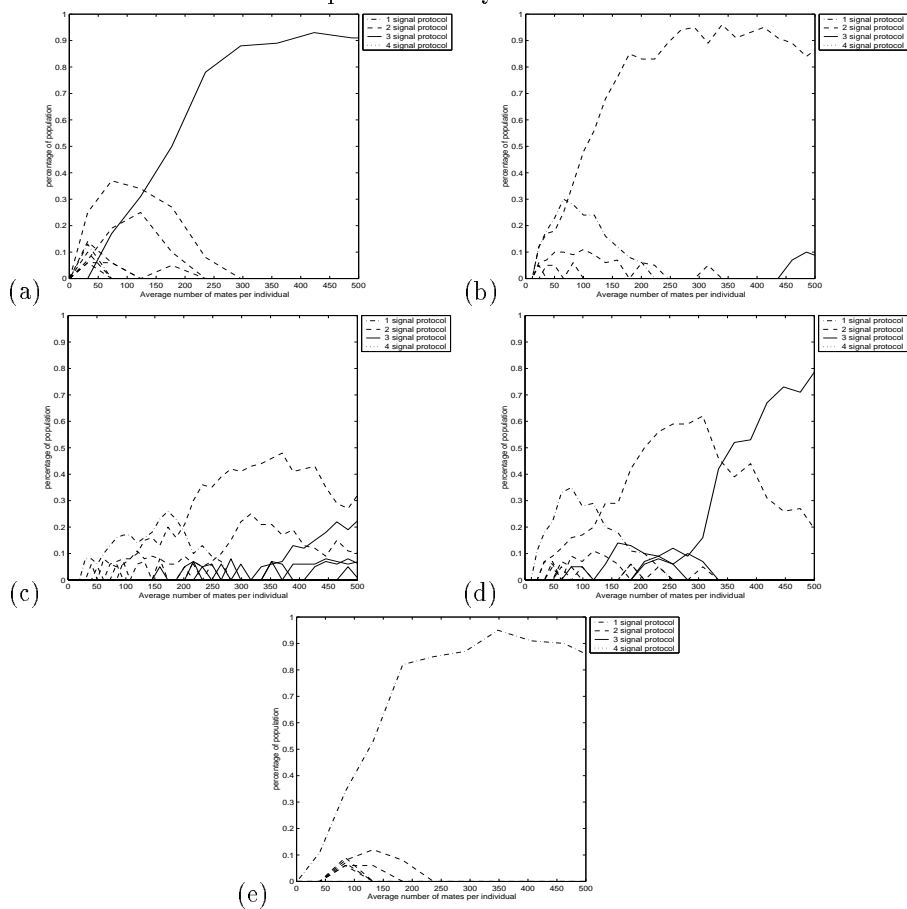


Figure 4.6: (a) Zoom in of fig.4.1 non-exploring agents, (b) Zoom in of fig.4.2 Exploring without learning, (c) Zoom in of fig.4.3 Plastic Agents, (d) Zoom in of fig.4.4 Genetically Specified Plasticity and (e) Zoom in of fig.4.5 Male and Female Agents Learn.

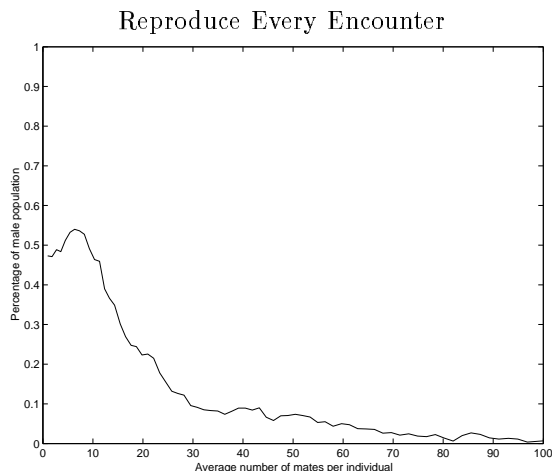


Figure 4.7: Percentage of population phenotypically plastic for a population in which learning was genetically specified. Note the different time scale.

following a 2SP to one following a 3SP. The transition between a 1SP and a 2SP population simply doesn't exist or is insignificant since both of this protocol classes appear almost simultaneously in time.

A more detailed description for each of the experiments will be given below.

4.2.1 No Exploration

The result of this experiment (figure 4.1) show that a population consisting of 'no-exploring' agents subject to a genetic algorithm converged into using a 3SP. The convergence rate of this type of agents turned out to be higher than the one obtained for any of the other experiments described on this section.

Only one 3SP —0310— appeared during the whole simulation run and very rapidly took over the population. By the time it first appeared —when the average number of mates per individual went just over 30— several 2SP were already being used by a significant percentage of the population (70%). However, none of them had more than 40% popularity.

From all the experiments in this project this is the one with closest conditions to W&D's second experiment and hence was used as a control to make sure that we were obtaining what was expected. W&D report that on their experiment the whole population converged to a single protocol after 40000 time steps. On our experiment 90% of the whole population converged to a 3SP before 10000 time steps, however, it was never assumed by the whole population as it was reported to be on W&D's. The behavior observed in both of the experiments is more or less the same.

The experiment made here differs from W&D's second experiment in the size of the search space. On theirs the agents were controlled by a lookup table

of integer numbers of size equal to the number of states that an agent could assume. The search space on this experiment consists of float numbers on a lookup table of size equal to the number of states multiplied by the number of actions that agents can have. Some other minor differences between the two models were pointed out implicitly chapter 2, but in the end they are more or less the same. The 2 experiments were expected to obtain similar results and they did.

4.2.2 Exploration Without Learning

This is the only case in which the population doesn't agree upon a single communication protocol but several manage to remain occasionally active over 5% of the population on the late stage of the simulation. On the experiment shown in figure 4.2 four main different protocols were constantly making appearance among the agents. The most popular one being followed by around 80% of the population, going up or down depending on how the other 3 protocols were doing, which as well oscillated in popularity and added up to the missing 20%. Three of them belong to the 3SP type (3100, 3110, and 3130) and one to the 4SP type (3120).

On the experiment shown on figure 4.2 where agents didn't learn but were allowed to explore the environment, the system doesn't converge to a single communication protocol. Four different protocols manage to survive. Three of them belong to the 3SP type (3100, 3110, and 3130) and one to the 4SP type (3120). These 4 protocols have an exact same effect if we assume that female agents evolved so that they don't need signal '1' on their strategies to guide the males, which is a reasonable assumption given that only three signals are really needed efficiently guide agents from any incoming direction. Further, a signal meaning 'stand still' is not very practical and gives no evolutionary advantage. However, for probabilistic reasons in the scenario described above you will expect the four protocols to end up covering equal amounts of the population which doesn't seem to be happening. Unfortunately the simulation was stopped and we can only try to guess what will happen next.

A 2SP '0100' becomes very popular early in the simulation taking over more than 90% of the population before the first 3SP '3100' showed up. Although the only difference between these 2 protocols is on the agreed meaning of the fourth signal, the great popularity of the 2SP puts the system on top of a local maximum making it difficult for the 3SP to take over. However, the 3SP finally does take over.

A main difference between this experiment and the one without exploration is that on this one the 3SP starts to develop when there is already a protocol with a very high percentage of followers. While on the previous one none of the protocols was very popular when the 3SPs started to appear, there was more divergence of protocols and less converged females hence evolution.

4.2.3 Plastic Agents

The plastic population assumed a 3SP strategy faster than what the exploring population did. For a while this 3SP population was divided into two different groups but finally converged into one. On the very first stage of evolution plasticity had the effect of increasing the diversity of popular protocols among the population, hence smoothing the fitness landscape and therefore retaining any 2SP from becoming too popular and taking the system into a local maxima (figure 4.6).

Protocols ‘0130’ and ‘0131’ became very popular at a middle stage of the simulation run, until finally this last one went extinct because their interpretation of signal ‘0’ —the silence signal— was to turn right making them spin around when no female was nearby.

4.2.4 Genetically Specified Plasticity

Figure 4.4 show that starting half of the population plastic and half of the population rigid does speed up the evolutionary process. Learning brings diversity on early stages of evolution helping more complex protocols appear earlier when it is easier to get popularity.

It is the competition among the variety of protocols that stops a protocol from becoming too popular and therefore avoiding evolution from getting stuck in a local maxima.

However, figure 4.7 show that the ability to learn is lost amazingly fast. When it does so, the population behaves like the ‘exploring without learning’ population. Rigidity helps the population converge into one of the protocols of greater complexity available. Further, by removing plasticity from the population evolution is removing the divergence and favouring the most reliable protocols.

After the average number of mates per individual has reached 100, the learning population is almost extinct. There is a selection pressure for the evolution of instinctual behaviours.

Thus after the population has lost the ability to adapt the dynamics are the same as in the plain GA with exploring rigid individuals. Although starting from a condition where some good 3SPs were already notorious among the population.

4.2.5 Male and Female Agents Learning

On figure 4.5 we can observe that male and female were only able to ‘agree’ on a 1SP which is the trivial solution to the problem and the only one that doesn’t involve mutual understanding between the sexes. With this strategy a male would ‘go forward’ for every single signal. Ten 2SPs became popular enough to pass the 5% threshold and appear on the graph but none of them became

popular enough to assume more than 10% of the population ².

When the average number of mates per individual had reached around 50 the ‘go forward’ protocol had already gained 10% popularity and the 2SP were starting to make appearance. By the time the best 2SP had reached around 10% the 1SP was already being used by more than 50% of the population. 2SPs go extinct and males evolve to go forward.

4.3 Agents reproduce after 5 encounters

The results shown on this section were obtained from evolving populations of agents that reproduced every 5 times they encountered a mate.

4.3.1 Non-Exploring Rigid Population

The population converges very rapidly to one following the same 2SP. The protocol ‘always go forward’ gets some popularity at the very beginning but falls rapidly giving some space to a 2SP to emerge. While the simulation lasted no 3SP appeared.

The population seem to be very stable at this point. Not many changes seem to be going on suggesting that the situation might change. Once in a while a different 2SP might show up but very rapidly disappears without affecting the behavior of the population. If you leave the simulation running enough time a 3SP might eventually appear and thus take over the population but the amount of time might be immense.

The system seems to be trapped at a local maximum.

4.3.2 Exploring agents without Learning

There is no significant difference from the case described above. The amount of protocols at the very beginning is a bit higher i.e. there is greater diversity, but in the long run converges to a 2SP just as above. It takes a bit more time to converge due to the exploration of the environment. A 3SP showed up without gaining sufficient popularity to get the system out of this situation.

4.3.3 Plastic Population

A plastic population of individuals converge at a much slower rate towards adopting a common protocol than the population of rigid agents. However, the protocol reached by the plastic population is one of a greater complexity (3SP).

On figure 4.10 we can see that several 3SPs do appear and one of them eventually takes over the population. The first 3SP appears when the population reached 350 average reproductions per individual. At this time no 2SP protocol

²The 10 protocols are not shown because of the sampling used when constructing the graphs.

No Exploration - Reproduce Every 5 Encounters

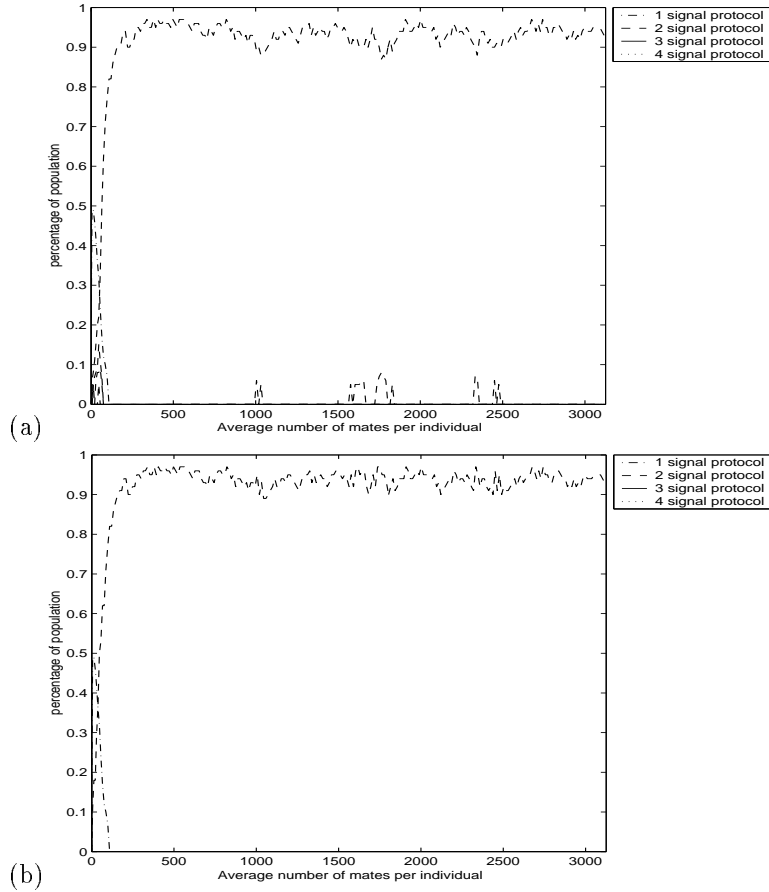


Figure 4.8: (a) Percentage of population belonging to each of the protocol classes and (b) percentage of male agents following one same protocol against the average number of reproductions per individual. On (a) each line is an accumulated sum of all the protocols belonging to that particular protocol class.

Exploration (no learning) - Reproduce Every 5 Encounters

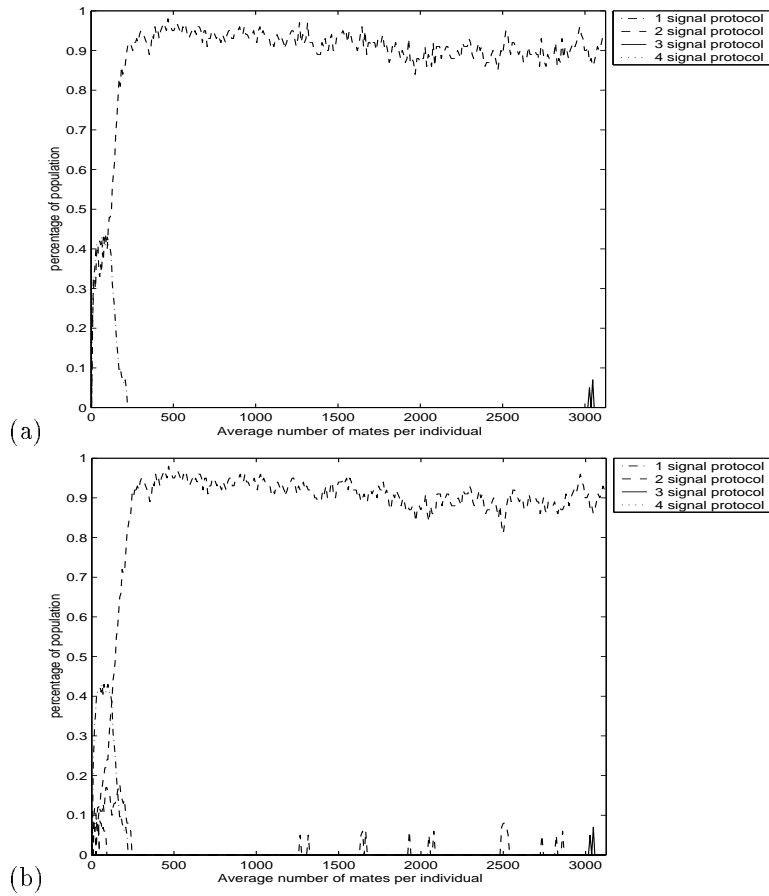


Figure 4.9: (a) Percentage of population belonging to each of the protocol classes and (b) percentage of male agents following one same protocol against the average number of reproductions per individual. On (a) each line is an accumulated sum of all the protocols belonging to that particular protocol class.

Q-Learning - Reproduce Every 5 Encounters

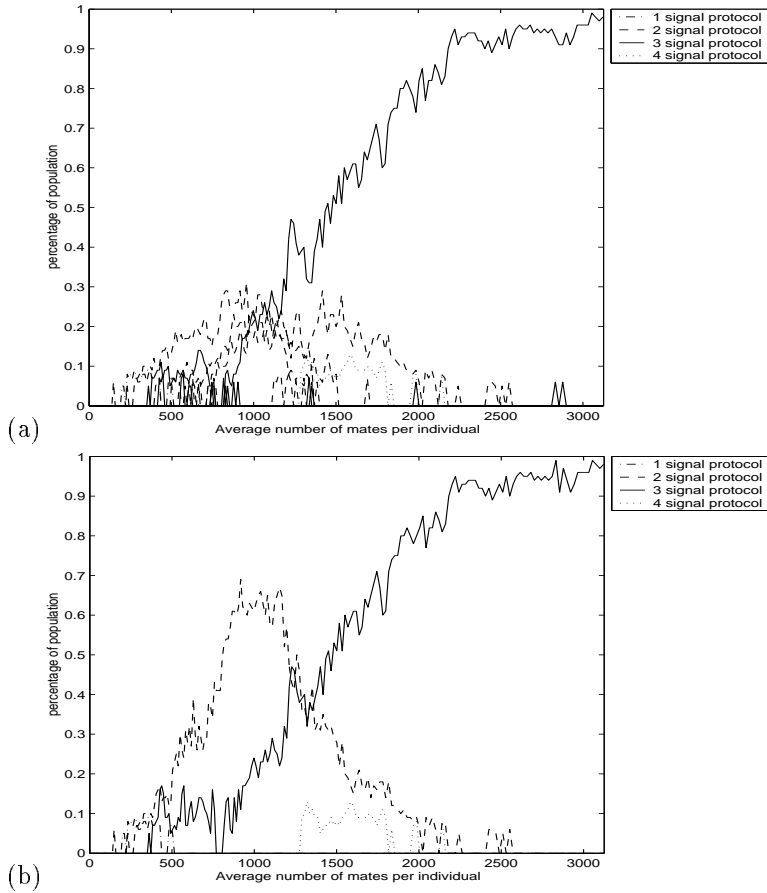


Figure 4.10: (a) Percentage of population belonging to each of the protocol classes and (b) percentage of male agents following one same protocol against the average number of reproductions per individual. On (a) each line is an accumulated sum of all the protocols belonging to that particular protocol class.

Genetically Specified Plasticity - Reproduce Every 5 Encounters

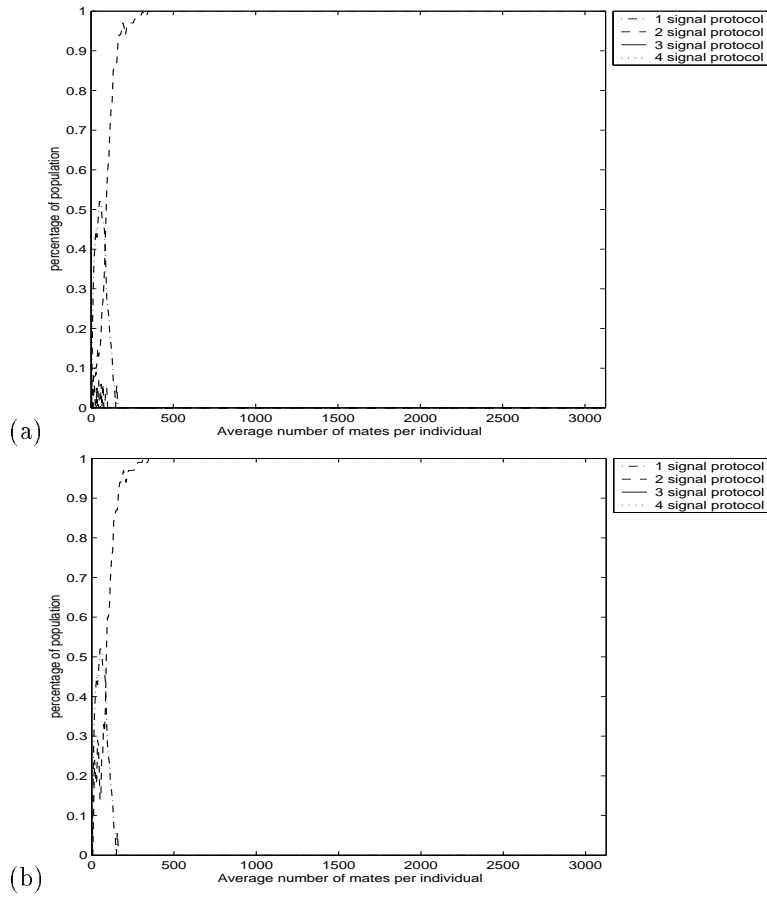


Figure 4.11: (a) Percentage of population belonging to each of the protocol classes and (b) percentage of male agents following one same protocol against the average number of reproductions per individual. On (a) each line is an accumulated sum of all the protocols belonging to that particular protocol class.

Male and Female Learn - Reproduce Every 5 Encounters

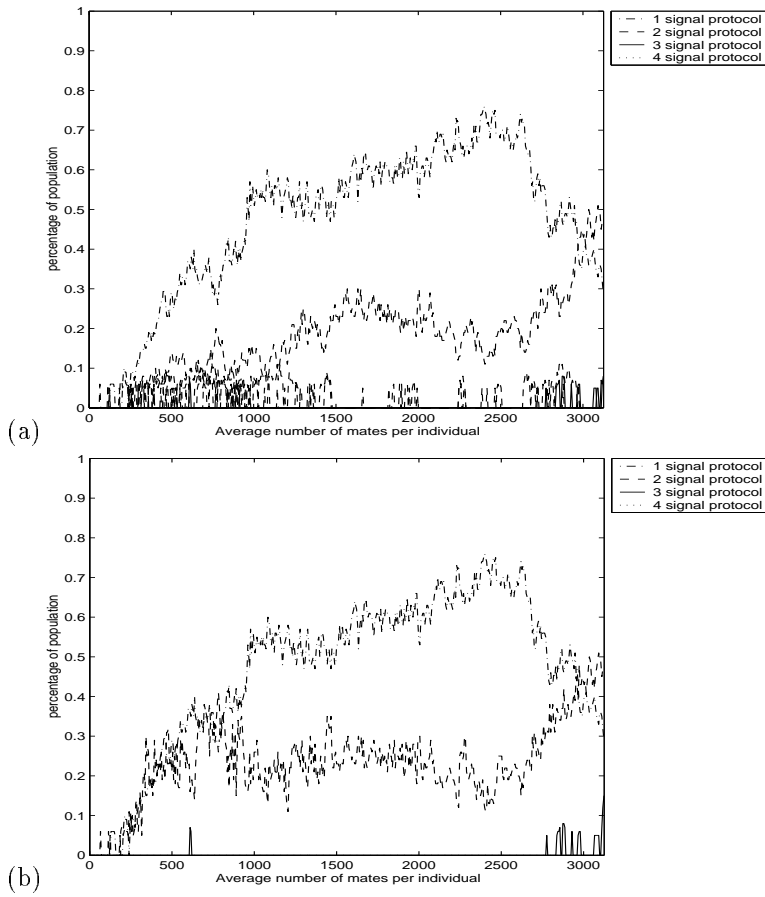


Figure 4.12: (a) Percentage of population belonging to each of the protocol classes and (b) percentage of male agents following one same protocol against the average number of reproductions per individual. On (a) each line is an accumulated sum of all the protocols belonging to that particular protocol class.

Reproduce Every 5 Encounters

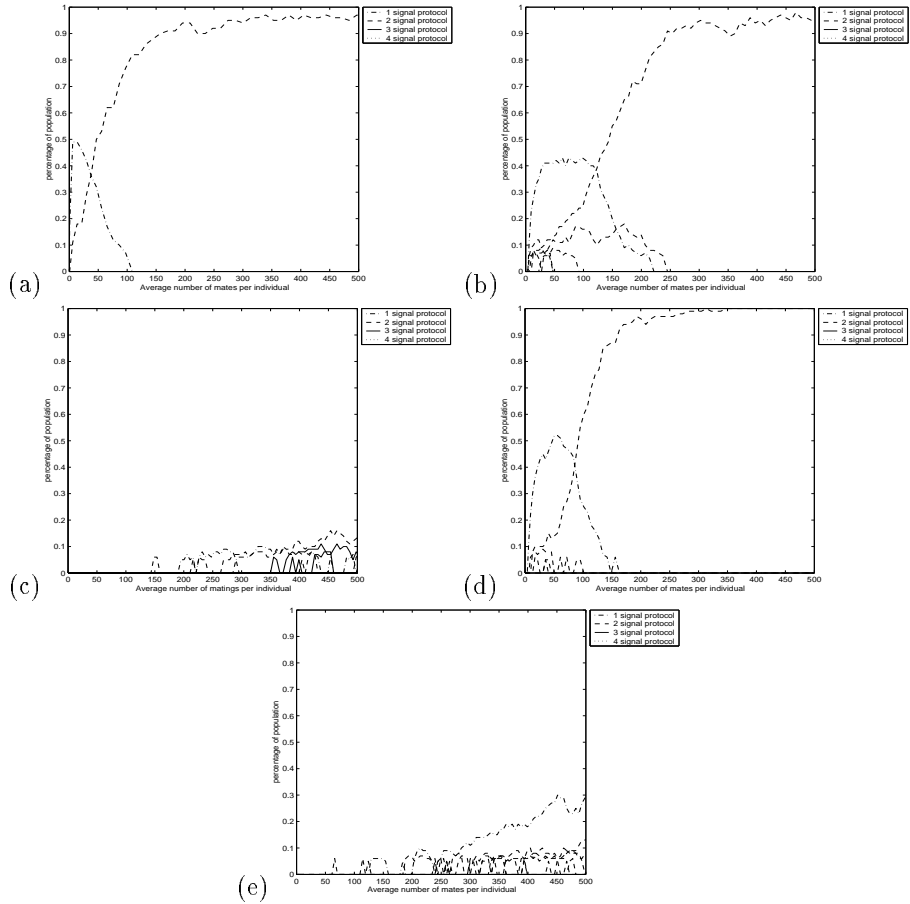


Figure 4.13: (a) Zoom in of fig.4.8 non-exploring agents, (b) Zoom in of fig.4.9 Exploring without learning, (c) Zoom in of fig.4.10 Plastic Agents, (d) Zoom in of fig.4.11 Genetically Specified Plasticity and (e) Zoom in of fig.4.12 Male and Female Agents Learn.

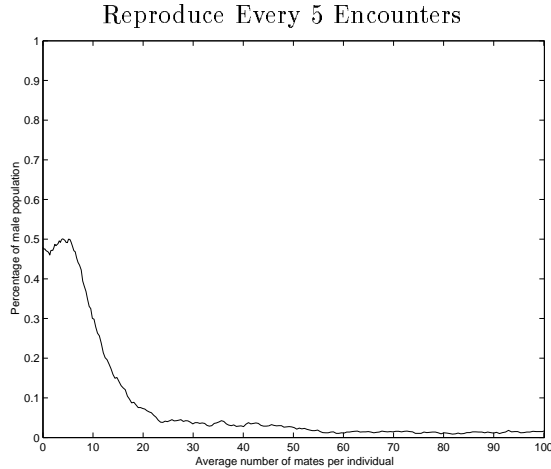


Figure 4.14: Percentage of population with the learning gene. Note the different time scale.

had exceeded 20% popularity. Contrary to the 90% achieved for both of the experiments described above.

Looking closely (figure 4.13c) at the early stage of evolution there is a gap where the population is so diverge that no protocol is popular enough to exceed 5% of the population. By the time the first protocol appeared the populations in all the other experiments of this section were already on there way to adopting a single 2SP.

There is a lot of diversity and several 2SPs and 3SPs can be observed.

4.3.4 Genetically specified plasticity

A main difference between the results obtained here and the ones described in section 4.2.4 is that in this experiment the population converged to a 2SP instead of a 3SP. The system is again stuck in local maxima.

Figure 4.14 shows how evolution selects for rigidity and the learning population decreases incredibly fast. The population becomes rigid before a 3SP appeared.

The dynamics of the population once the ability to learn has been lost is almost the same as the one described in section 4.3.2 involving exploring rigid agents. Comparing figures 4.13a and 4.13d there is a greater variety of protocols on the second one before the average number of mates per individual had reach 100, period of time in which there are still some learning individuals among the population.

After that, the decreasing amount of learning agents help the population agree more on a single protocols accelerating the convergence.

4.3.5 Male and Female agents Learn

Upon decreasing the selection pressure and hence increasing the time agents had to learn a great variety of protocols start to appear. The most popular is still a ‘0000’ protocol, but one 2SP slowly starts to get attention. Some 3SP have even started to appear. By the end of the simulation the 1SP was quickly losing popularity. The 2SP seems to take advantage of this fall and starts to go up. Unfortunately running a longer simulation to see if a 3SP does attract the population was impossible due to time constraints.

4.4 Agents reproduced after 15 encounters

The results are very similar to the ones obtained with agents that reproduced every 5 encounters. Only the main differences are going to be pointed out.

4.4.1 Rigid Non-exploring Population

This run is different from the case in which agents reproduced every 5 encounters because the final 2SP population was split into 2 groups. Two different 2SP groups manage to compete. One ‘0100’ oscillating around 70% of the population while the other ‘0110’ around 30%. The system seems very stable in this configuration. No 3SP popped out during the simulation.

4.4.2 Exploring Population

On the population of exploring individuals that reproduced every 15 encounters the convergence towards a 2SP population takes a bit longer than the population of rigid non-exploring individuals. However, the entire population assumes the same 2SP. Some protocol diversity was observed at the very early stage of the simulation due to the exploration of the agents.

4.5 Plastic Population

There is a huge gap at the early stage of evolution where no protocol gets more than 5% popularity. The convergence is very slow towards a single protocol. By the end there are still several 3SP fighting and more than 80 % of the population is still undecided about which protocol to follow. Evolution is very slow for this level of selection pressure, the simulation ended in a transient state is difficult to deduce the stable state.

4.5.1 Plasticity is genetically specified

When agents reproduced after having found a mate 15 times the population of plastic individuals —those with an active learning gene— increase at the beginning, reaches a maximum and goes down again towards extinction. A 3SP

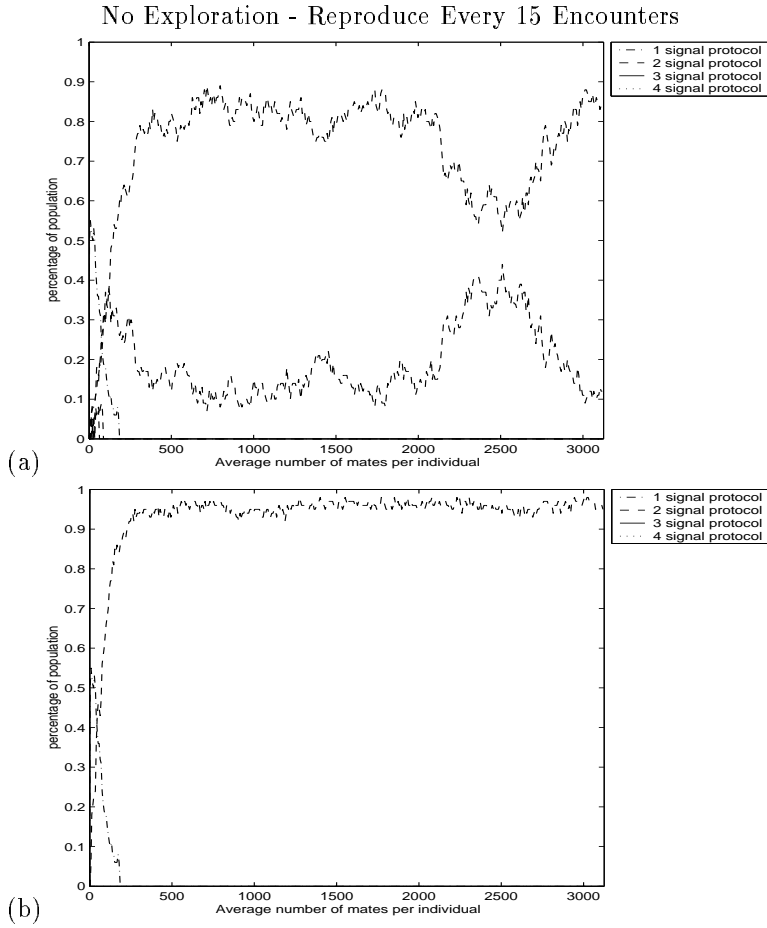


Figure 4.15: (a) Percentage of population belonging to each of the protocol classes and (b) percentage of male agents following one same protocol against the average number of reproductions per individual. On (a) each line is an accumulated sum of all the protocols belonging to that particular protocol class.

Exploration (no learning) - Reproduce Every 15 Encounters

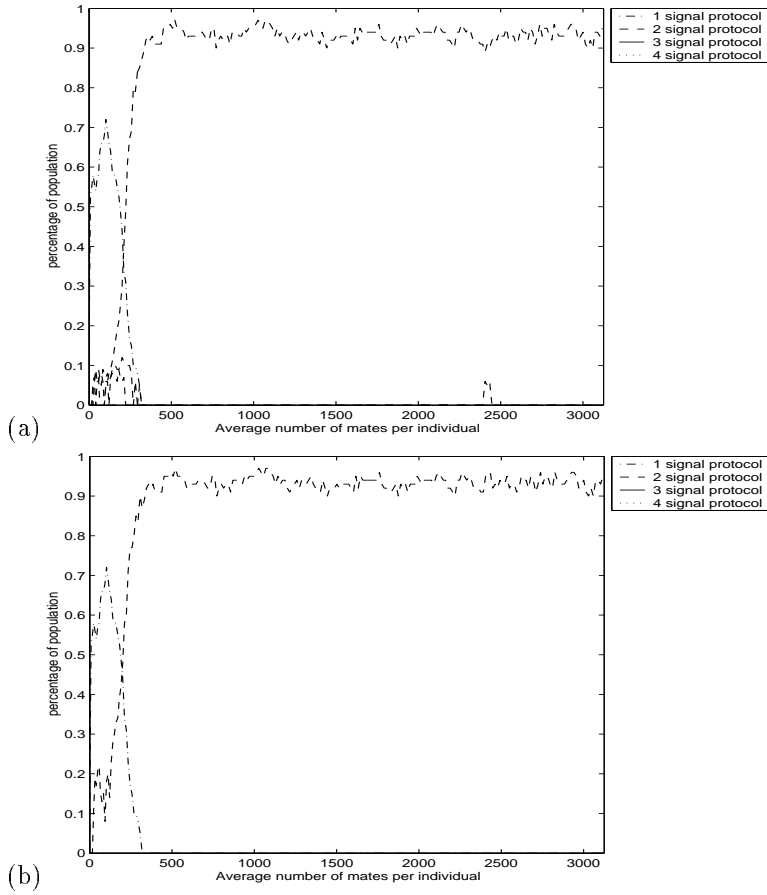


Figure 4.16: (a) Percentage of population belonging to each of the protocol classes and (b) percentage of male agents following one same protocol against the average number of reproductions per individual. On (a) each line is an accumulated sum of all the protocols belonging to that particular protocol class.

Q-Learning - Reproduce Every 15 Encounters

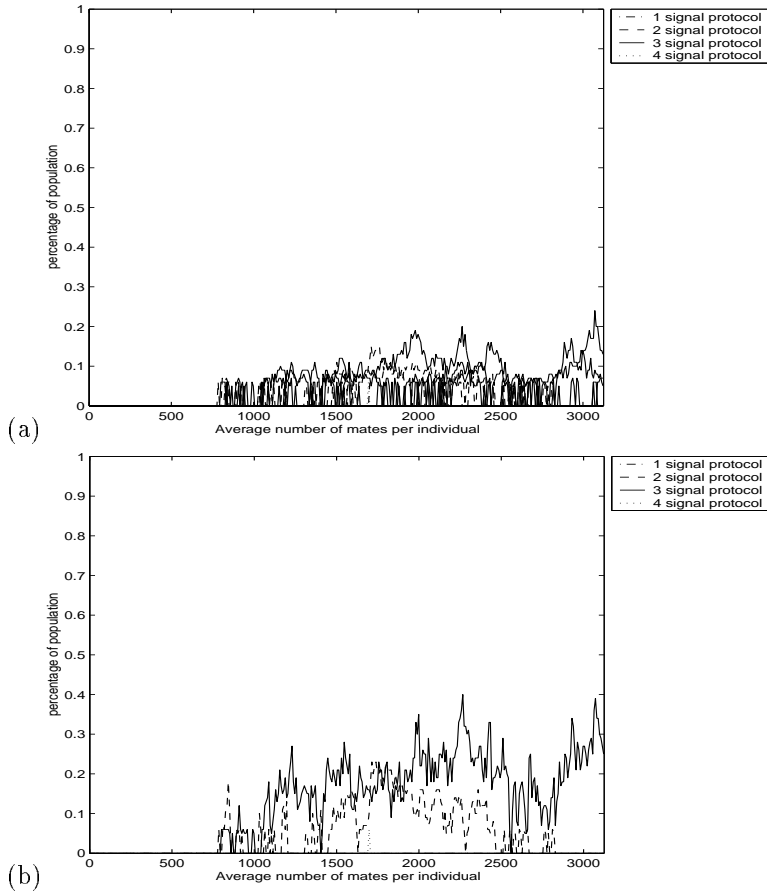


Figure 4.17: (a) Percentage of population belonging to each of the protocol classes and (b) percentage of male agents following one same protocol against the average number of reproductions per individual. On (a) each line is an accumulated sum of all the protocols belonging to that particular protocol class.

Genetically Specified Plasticity - Reproduce Every 15 Encounters

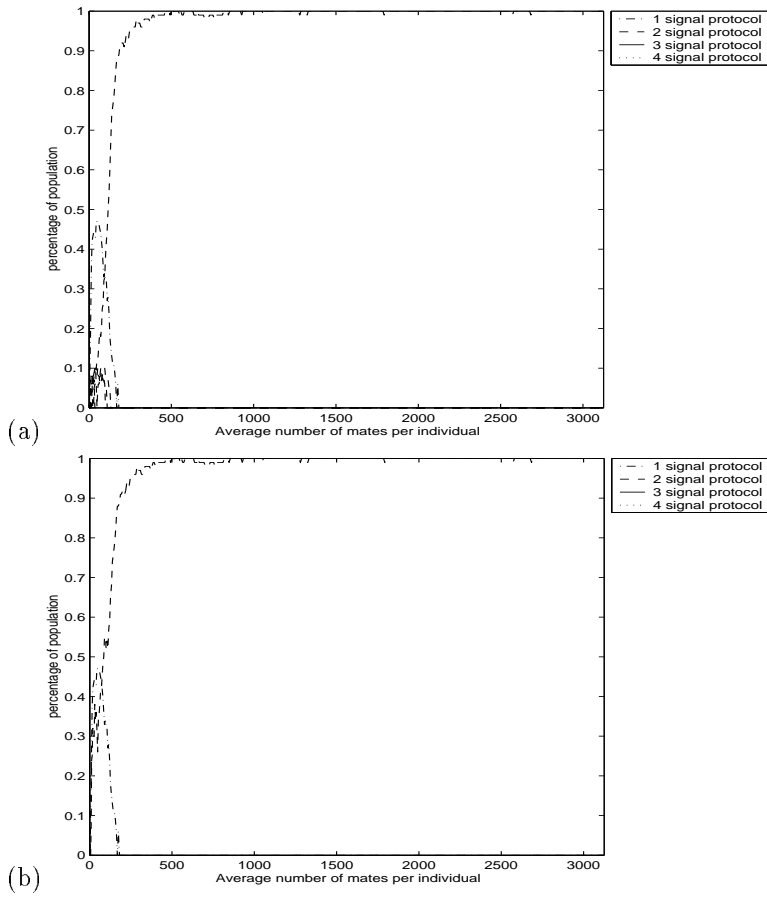


Figure 4.18: (a) Percentage of population belonging to each of the protocol classes and (b) percentage of male agents following one same protocol against the average number of reproductions per individual. On (a) each line is an accumulated sum of all the protocols belonging to that particular protocol class.

Reproduce Every 15 Encounters

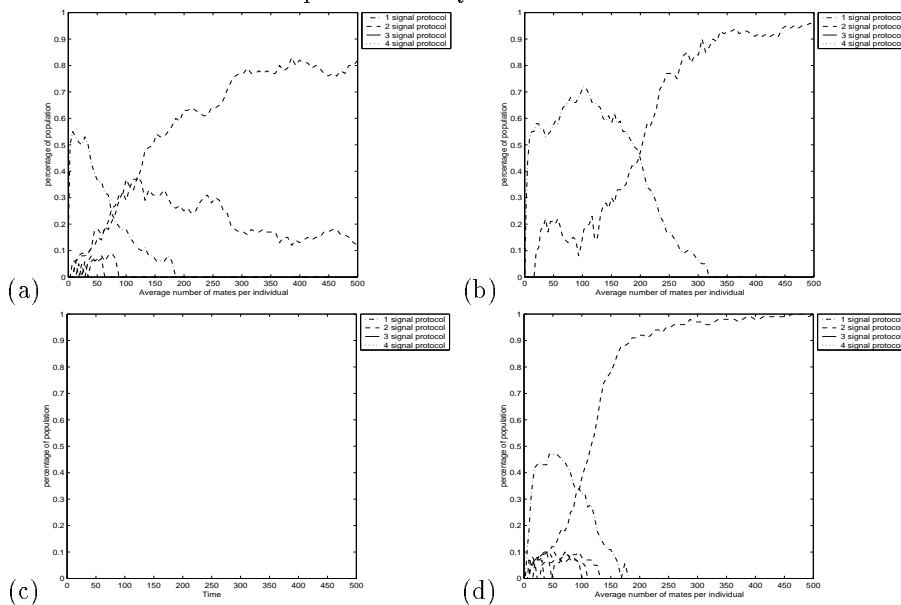


Figure 4.19: (a) Zoom in of fig.4.15 non-exploring agents, (b) Zoom in of fig.4.16 Exploring without learning, (c) Zoom in of fig.4.17 Plastic Agents, and (d) Zoom in of fig.4.18 Genetically Specified Plasticity.

Reproduce Every 15 Encounters

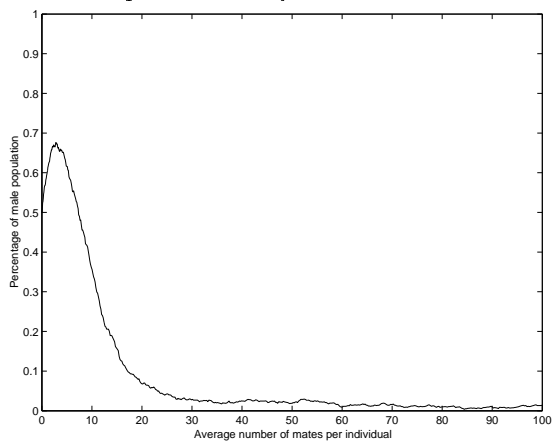


Figure 4.20: Percentage of population with the learning gene. Note the smaller time scale.

doesn't appear through out the run, and as the case described in section 4.3.4 the population rapidly converges into a single 2SP without leaving time for more complex protocols to evolve.

Chapter 5

Analysis

Unfortunately experimental runs were computationally expensive sometimes taking several days to finish, and thus making it difficult to include a deep statistical analysis or even to obtain variety on the results.

Three different cases of selection pressure were studied each having different consequences over the evolutionary process. Learning only speeded up the evolutionary process when the individuals were subject to a ‘high’ selection pressure—agents reproduced every time they mate—if we compare it with a exploring population, but not with the non-exploring population. For a ‘medium’ selection pressure—when agents reproduced every 5 encounters—learning helped the population reach a more efficient communication protocol than a non-learning population, however the convergence towards a single protocol took a longer period of time. This time period became even longer for the case in which agents reproduced after 15 encounters, in fact it never converged. By the end of the simulation the 2SP had already gone extinct. There were still three 3SPs competing for popularity but none of them had more than 30% popularity.

The scatter plots shown in this chapter illustrate the number of mates that each individual has accumulated so far against its age at 2 different time instants, very early in evolution when the average number of mates per individual was 300 and at a late stage of evolution when the average number of mates per individual had reached 3000. Lines illustrate a linear fit of the data. More efficient populations are represented on the scatter plots with a bigger slope since agents mate more when they are younger. As agents become more efficient the selection pressure increases making it harder for agents to survive. This is mainly because the increase in the number of reproductions will imply as well an increase in the number of deaths. In other words, individuals in efficient populations don’t live as long as individuals in inefficient populations. An efficient population will be spread towards the upper left side of the graphs while a less efficient towards the lower right. By looking at the scatter plots we get an idea of the performance of a whole population.

The cost involved in exploring the environment can be observed by comparing figures 5.4a and 5.4b. On the first one, the population involved doesn’t

Learned vs Innate Phenotypes - Reproduce Every Encounter

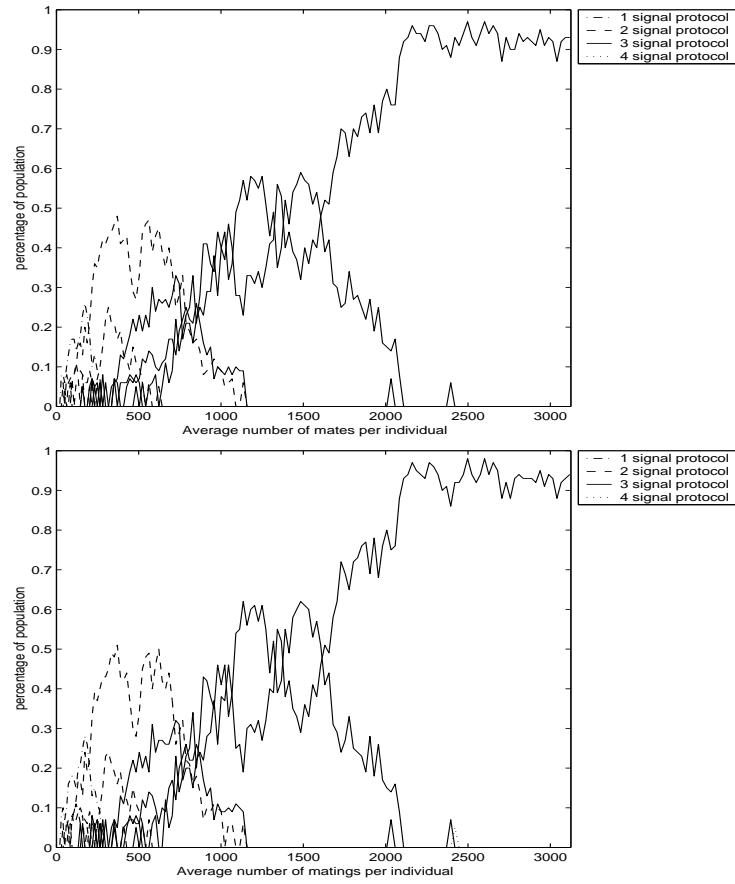


Figure 5.1: Proportion of learned (top) and innate (bottom) phenotypes of a population of plastic agents that reproduce every time they find a mate throughout an evolutionary simulation run.

explore the environment, while on second one it does. On neither of the graphs many agents reproduce more than 10 times during their lifetimes, which mean that the ‘exploring’ agents don’t reach their mature state and therefore remain exploring the environment throughout their life rather than exploiting what they already know about it.

Learned vs Innate Phenotypes - Reproduce Every 5 Encounters

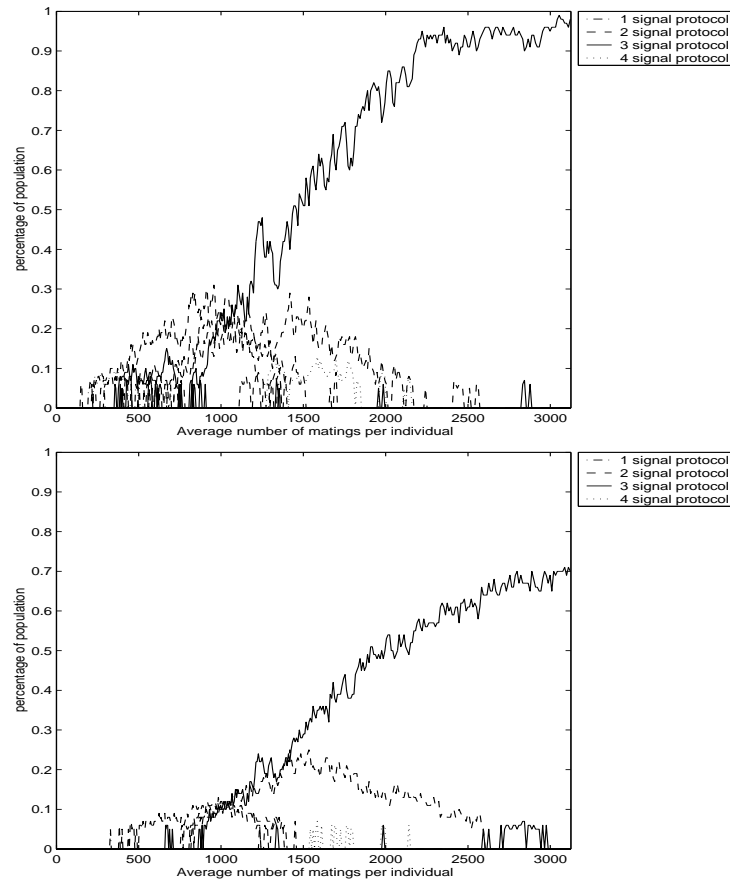


Figure 5.2: Proportion of learned (top) and innate (bottom) phenotypes of a population of plastic agents that reproduce every 5 times they find a mate throughout an evolutionary simulation run.

There is some cost involved in exploring the environment, and although exploring brings some diversity to the population by smoothing the fitness landscape, it slows down the convergence towards a single protocol. The stochastic factor involved in exploring the environment can aid individuals that would had otherwise died if they hadn’t been able to explore new behaviours within the environment. The evolutionary cost that arises from exploration is that young

agents might follow the correct action by chance and reproduce, passing ‘bad’ genes into a subsequent generation. Or the other way around where a ‘fit’ individual follows an inappropriate behaviour so as to explore new possibilities and hence decreases its number of reproductions. In other words, non-exploring agents are more reliable than the exploring ones. This cost is less notable when agents reproduced every 5 encounters (see figures 5.5a and 5.5b) and almost disappears when agents reproduced every 15 encounters (see figures 5.6a and 5.6b). For this first case, agents reached up to 80 mates during their lifetimes and more than half of the population is on their exploitation period. Agents are halfway mature when they reproduce for the first time and fully mature by the time they reproduce for the second time, which is not the case for agents that reproduced every encounter and had to reproduce 10 times before achieving the mature —exploitation— state, that is, if they were lucky enough to get there. For the other case in which agents reproduced every 15 encounters agents are fully mature when they mate for the first time. They do live a little bit longer due to the fact that exploration decreases the amount of encounters and hence the amount of deaths. This effect is observed in that the cluster on the scatter plot for the exploring agents is a bit more stretched.

In the experiments involving plastic agents there is an initial period of time where no protocol becomes popular enough to appear on the graphs. This protocol gap increases in size as the selection pressure decreases. During this time the population is so diverse that no protocol exceeds 5% of the population and hence appears on the graph.

The scatter plots shown in figure 5.4 and 5.5 help illustrate why evolution selects for rigidity even though results show that plasticity helps the evolving population acquire more ‘efficient’ communication protocols.

The graphs show very clearly that the plastic agents are less efficient than all the other agents because the slope is much slower. There is obviously some cost involved in learning.

When agents reproduced every 5 encounters plastic individuals were not able to reproduce as fast as the rigid ones did. On figure 5.5c plastic agents employing a 3SP are getting fewer reproductions per age of the individual —or less slope on the graph— than rigid ones employing a 2SP (figure 5.5b) even though their protocol is more complex and efficient.

On figure 5.5c we can see a cluster of agents that have mated fewer than ten times, which means that the great majority of agents are still on an exploration state. The few agents that reach a mature state are very good at reproducing, since they don’t explore the environment anymore and have a more reliable behaviour, but the other ones are not.

The benefits of learning are greater than the costs at the very beginning, but as the population starts to become more efficient in reproduction the selection pressure increases and learning individuals are not able to compete anymore with the rigid ones. It is shown in graphs 4.7, 4.14 and 4.20 that evolution selects for rigidity even though plasticity helps the population reach more complex behaviours. This is because rigidity is a more reliable mechanism. However for 3 levels of selection pressure the amount of learning agents does go up at the very

Learned vs Innate Phenotypes - Reproduce Every 15 Encounters

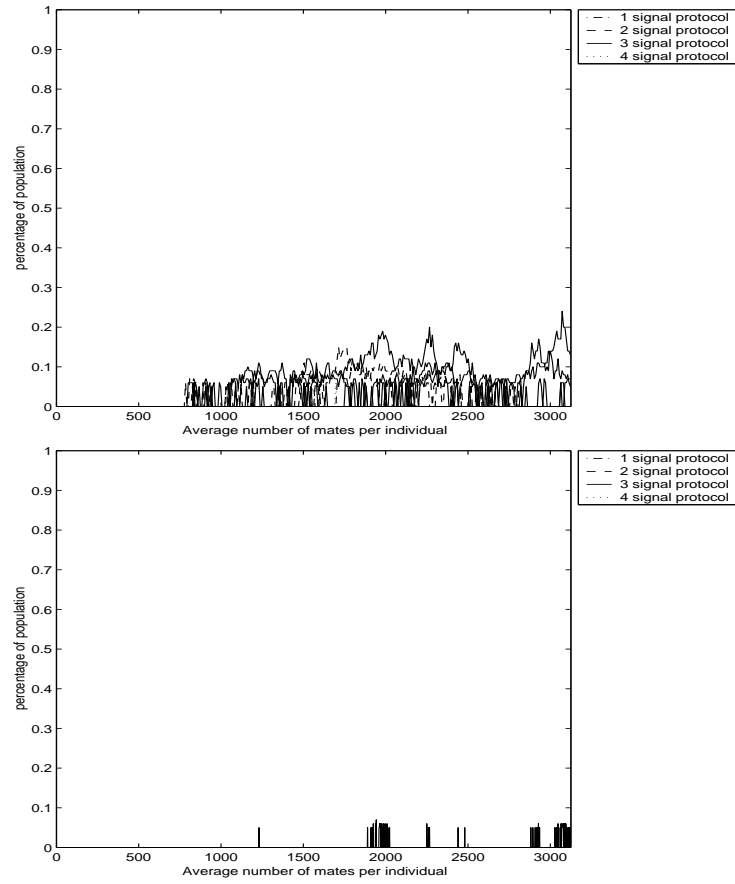


Figure 5.3: Proportion of learned (top) and innate (bottom) phenotypes of a population of plastic agents that reproduce every 15 times they find a mate throughout an evolutionary simulation run.

Reproduce Every Encounter

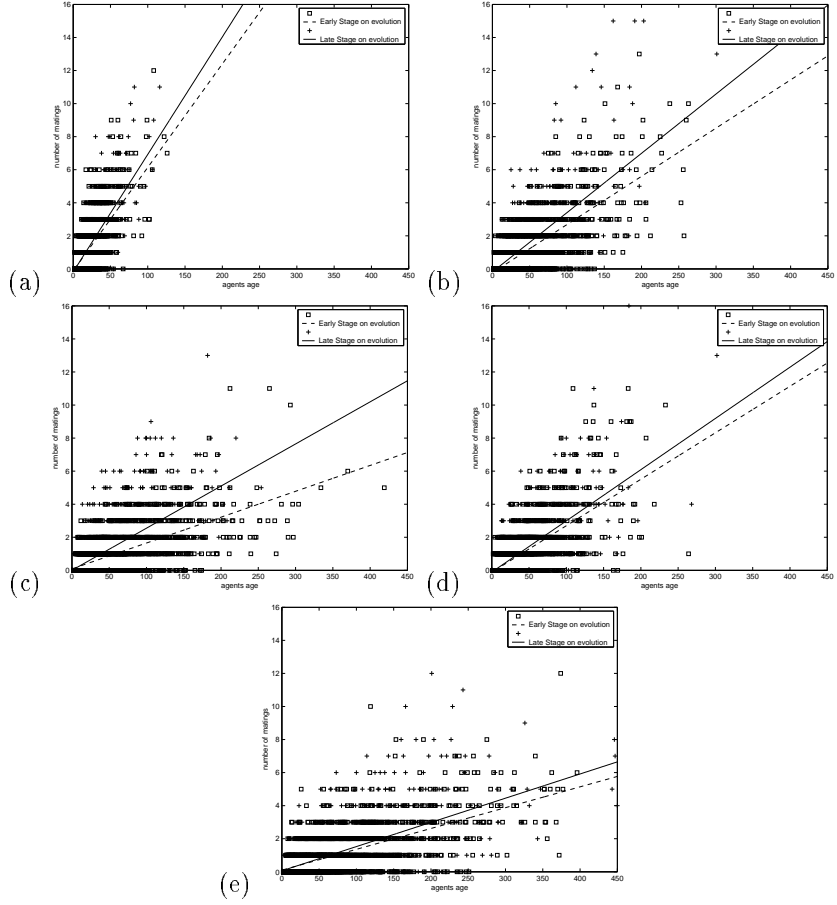


Figure 5.4: Scatter plot for the number of mates and age of the entire population at 2 different instants in time for (a) non-exploring rigid agents, (b) exploring rigid agents, (c) plastic agents, (d) when learning is genetically specified among the agents, (e) plastic male and female agents.

beginning of the run just before the ‘0000’ agents start to gain popularity and then decreases exponentially. Rigid agents are very fast at reproducing if they have the ‘right’ —meaning popular— protocol because they always behave in the same way without making any mistakes. However, if they have the ‘wrong’ protocol they won’t do the right thing and hence won’t reproduce.

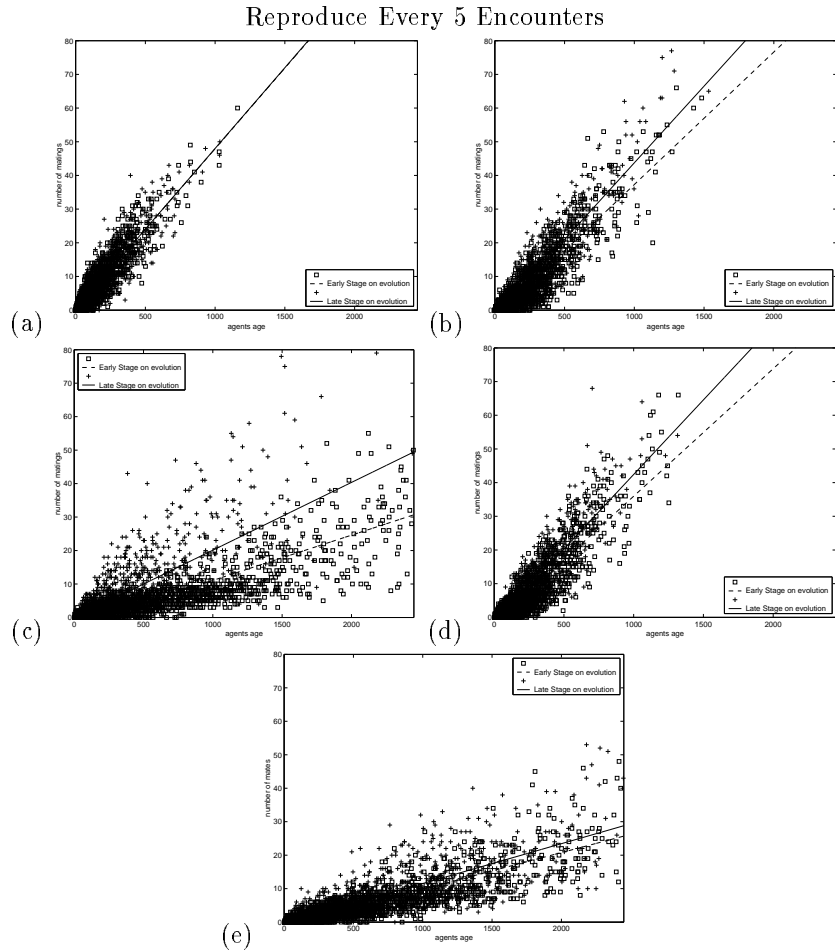


Figure 5.5: Scatter plot for the number of mates and age of the entire population at 2 different instants in time for (a) non-exploring rigid agents, (b) exploring rigid agents, (c) plastic agents, (d) when learning is genetically specified among the agents, (e) plastic male and female agents.

The learning gene stays active in the populations before the average number of reproductions per individual had reached 100. During this period for the case of ‘high’ selection pressure several 3SPs had already shown up due to the plasticity that still remained on the population. For the rigid agents the first 3SP

appears when the average number of reproductions per individual was around 450 i.e. much later in time.. On figure 4.20 we can see that the population of learning individuals increased almost to 70% at the beginning of the run. This tells us that the costs and benefits of learning change throughout evolution and supports Baldwin's thought that phenotypic plasticity is good on the early stages of evolution when the world is highly dynamic, and instinctual behaviours become better as the world becomes more static. After the agents have lost their ability to learn they perform exactly as the rigid ones (compare figures 5.6b and 5.6d).

For any particular population evolution would favour the 3SP protocols simply because they are more efficient. However, the more popular a protocol gets the harder it is for a more complex one to take over a population. If the whole population converges to a single protocol it becomes difficult for evolution to change this. Protocols evolve because there is some diversity among the individuals and some protocols turn out to be better than others. But if all females are behaving in exactly the same way, a male with a different protocol is going to be less fit because it simply doesn't understand the signals that female are emitting. Plasticity showed to provide variety among the population keeping protocols from becoming too famous early in time. As a protocol becomes popular among the population the density of individuals following it increases, thus increasing as well the chance of running into an individual that belongs to it. On the one hand, if you belong to it the probability of passing your genes to a subsequent generation increases with the popularity of the protocol you are using. On the other hand, if you are different from the majority of the population your amount of reproductions would decrease for two main reasons. First, there is a chance that you get killed because the number of reproductions per time step has augmented due to popularity, and second you might not be reproducing very much since you misinterpret an increasing amount of individuals within the population.

By the time the 3SPs started to appear in the plastic population subject to a 'high' selection pressure 2SPs weren't still popular, which is not the case of the experiment described involving exploring individuals under the same selection pressure where more than 90% of the population was already following one same 2SP when the first 3SP appeared (figures 4.6c and 4.6b respectively). On this experiment the first 3SP appears when the average number of mates per individual is around 150. At this time instant the most popular protocol was still '0000' followed by a couple of 2SP, one of which was '0030'. But none of them exceeded 20% of the population. Having a more dispersed population of 2SPs makes it easier for a 3SP to take over the population since the female agents are also dispersed, i.e. fitness landscape is smooth.

Figure 5.1 show that the innate phenotypes are very similar to the learned ones when the pressure was 'high', that is, because agents don't have time to learn due to the 'aggressiveness' of the environment. As we decrease the pressure (figure 5.2) the amount of agents changing from an innate protocol to a learned becomes significant. When the simulation stopped almost 20% of the population was switching their behaviour towards the most popular one, which is good in

Reproduce Every 15 Encounters

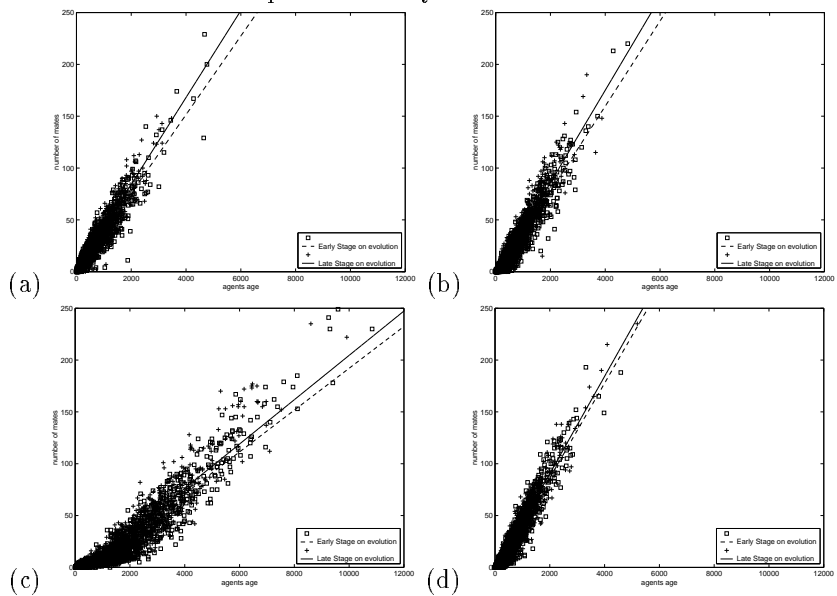


Figure 5.6: Scatter plot for the number of mates and age of the entire population at 2 different instants in time for (a) non-exploring rigid agents, (b) exploring rigid agents, (c) plastic agents, (d) when learning is genetically specified among the agents, (e) plastic male and female agents.

the sense that the genetic information for other protocols still remains within the population and maintains the divergence which facilitates the evolution of more efficient solutions. When the selection pressure was reduced even further and agents reproduced every 15 encounters not many innate protocols were popular. On figure 5.3 we can see that the popular protocols are being learned by the population, and there are almost none innate popular protocols. Individuals have enough time to learn a protocol and adapt to their local environment. Evolution works at a slower rate because it is now selecting for the ability to learn. Those individuals that have to adapt less to reach an appropriate protocol have a slight advantage.

Although there are problems when two agents are learning at the same time from examples coming from each other, agents seem to be agreeing on a 2SP in the experiment where both male and female learned subject to the constrain that they reproduced every 5 encounters. Some 3SPs have even appeared on the population. Unfortunately the experiment with both male and female learning and agents reproducing after 15 encounters crashed and it was impossible to repeat it in time due to the computational expenses involved. It will be interesting to see if this population would agree faster on a particular protocol. Two learning species make the convergence of the population much slower.

Chapter 6

Discussion

This study illustrates some interesting issues about the costs and benefits of learning. The model implemented on this dissertation is an example that learning is not always good and sometimes instinctual behaviours can have a better net evolutionary benefit.

Our results show that for the model implemented plastic individuals do developed better and more efficient mating strategies i.e. communication protocols. However, these plastic agents performed much worst than the rigid ones subject to identical conditions. On the experiments where we assumed that learning was genetically specified there was a selective pressure against plastic mechanisms.

Two aspects of the Baldwin effect mentioned by Turney (1996) are clearly observed from the results. 1. Learning in some situations accelerates evolution and 2. Learning is expensive so there is a selective pressure for the evolution of instinctive behaviours which tend to be less expensive than learned mechanisms.

On the very early stage of evolution when the world is highly dynamic learning is selected for, but because of the nature of the environment which imposes a high cost and a low benefit for plasticity, the situation rapidly change and plasticity is selected against. Learning is useful to adapt to changes in the environment that happen on the individual's lifetime scale. Therefore we might expect that evolution selects for plasticity more likely in a dynamic environment, and against it if otherwise. In our simulation as time progresses and a high percentage of the population converges to one protocol the world becomes very static bringing no benefit to being plastic.

At the beginning of a simulation run local search (learning) can be useful, since the population is diverse and the individual's performance depends on what every other individual is doing. Learning brought diversity to the population at the early stages of evolution having the effect of smoothing the fitness landscape. As time progresses and the population adopts a protocol a global search strategy (rigidity) becomes a better strategy. In some of the experiments —when agents reproduced every 5 and 15 encounters— rigid agents got trapped in a local maximum (a 2SP protocol) that plastic agents were able to overcome

by smoothing the search space. Turney summarizes this process very well. “If it is possible for the behaviour to be performed by an instinctive mechanism, it will usually be advantageous for such a mechanism to evolve, since instinctive mechanisms tend to be less expensive than learned mechanisms. However, when a new behaviour is first evolving, an instinctive mechanism may require the population to make a large evolutionary leap, while a learned mechanism may be able to arise in smaller evolutionary increments. Learning may allow the behaviour to eventually become common and robust in the population, which then gives evolution the time required to find an instinctive mechanism to replace the learned mechanism. In summary, at first learning is advantageous, but later it is not” (Turney, 1996).

By comparing the results obtained from experiments involving ‘exploring’ agents with the ones involving ‘non-exploring’ agents we are able to appreciate the cost involved in having the ability to explore the environment is significant in this model. Exploring keeps you away from exploiting, and although you are acquiring knowledge from the environment by exploring you are not taking advantage of it.

Learning has other sources of cost. The fact that plastic individuals performed worst than the exploring agents suggests that there are other costs involved in learning.

On this model there is not really a need for plasticity to search for new innovative advantageous solutions. Werner & Dyer (1991) describing some of the implications about their model said that their “... approach, however, does not provide the population with the flexibility to evolve a communication system that the experimenter does not expect”. The initial distribution of protocols is so that there are several individuals belonging to each one of the protocols. Also the domain of possible solutions to the problem is not very big because it was designed that way. There is no innovation to be found, everything is already there.

A learning mechanism can be useful in exploiting environmental regularities and hence reduce the genome’s size, however on this model there isn’t any regularity to exploit and the genome size is constant.

The greatest cost individuals have is that they spend some period of their lives learning this very ‘aggressive’ environment. During the learning period the agent is not reproducing as much as it would have given that its behaviour was instinctual.

Agents during their lifetimes can get bad examples as well, which can make them perform less efficiently. Learning has a stochastic factor not found on instinctual behaviours, which makes instinct a more reliable mechanism.

W&D artificial world isn’t suitable for adaptive individuals. For it to be suitable the net benefit of learning would have to be positive, that is, low cost and high benefit. When plasticity works on its own it finds better solution, but when it competes against rigidity it fails. Instinctual mechanisms are better on this world where you have to reproduce as fast as you can before you go dead. You are not allowed to make mistakes.

In summary the benefits of learning for the model implemented on this paper

are only applicable on very early stages of evolution. The cost of learning is high and although learning is helping the system reach a more efficient solution the instinctual behaviour is more reliable and performs better.

Conclusion

Learning is not always good. By introducing learning into a simple communication system (the W&D (1991) model) it has been shown that although solutions achieved by plastic individuals are more efficient, the agents themselves are not. For the parameters used in this model there is an evolutionary selection pressure that selects for rigidity, even though plasticity proved to achieve faster and better results. The cost of learning turned out to be greater than the benefit and hence learning was selected against.

Further Work

Due to time constraints some simulations were left unfinished. It can be interesting to run more simulations with low selection pressure for much longer periods of time and compare the innate with the learned phenotypes. On the experiment with low selection pressure —where agents reproduced every 15 mates— things were starting to happen, it would be interesting to finish it and compare innate and learned phenotypes looking for the phenomena of assimilation. Also most of the runs got stuck in local maxima. It can be interesting to see if time can get them out of that situation to finally achieve a 3SP population.

There are lots of parameters to play around, including density of individuals, selection pressure, exploration period, learning rate, etc. to vary the cost and benefit of plasticity and rigidity. However, to be able to appreciate the benefits of learning the selection pressure has to go down, and hence CPU time goes up.

It can be interesting to genetically encode for learning rates and/or exploration periods instead of having plasticity as an ON-OFF switch that evolution can turn off very easily.

The world studied here is not very suitable for plastic individuals. For a further study of the cost and benefit of learning a new scenario has to be developed.

Bibliography

- [1] Baldwin, J.M. (1896). A new factor in evolution. *American Naturalist*, 30, 441-451.
- [2] Belew, R.K. and Mitchell, M. (1996). Editors, *Adaptive Individuals in Evolving Populations: Models and Algorithms*, Massachusetts: Addison-Wesley.
- [3] Bullock, S. (1997). Evolutionary Simulation Models: On their Character, and Application to Problems Concerning the Evolution of Natural Signaling Systems. Ph.D. Thesis, School of Cognitive and Computing Sciences, University of Sussex, Brighton, UK.
- [4] Fagen, R. (1981). *Animal Play behavior*. Oxford University Press.
- [5] Hinton, G.E., and Nowlan, S.J. (1987). *Complex Systems*, 1, 495-502.
- [6] Mayley, G. (1996a). Landscapes, Learning Costs and Genetic Assimilation. In special edition of *Evolutionary Computation, Evolution, Learning, and Instinct: 100 Years of the Baldwin Effect*. Turney, P., Whitley, D., & Anderson, R. (Eds.),
- [7] Mayley, G. (1996b). The evolutionary cost of learning. In Maes, P. Mataric, M., Meyer, J., Pollack, J. & Wilson, S. (Eds.), *From Animals to Animats 4: Proceedings of the Fourth International Conference on Simulation of Adaptive behavior*. MIT Press.
- [8] Maynard Smith, J. (1993). *The Theory of Evolution*. (3rd edition). Cambridge University Press.
- [9] Menczer, F., & Belew, R. (1994). Evolving sensors in environments of controlled complexity. In Brooks, R., & Maes, P. (Eds.). *Artificial Life IV*.
- [10] Morgan, C.L. (1896). On Modification and Variation. *Science*, 4, 733-740.
- [11] Osborn, H.F. (1896). Ontogenic and phylogenic variation. *Science*, 4, 786-789.

- [12] Noble, J.(1998) The Evolution of Animal Communication Systems: Questions of Function Examined through Simulation. Ph.D. Thesis, School of Cognitive and Computing Sciences, University of Sussex, Brighton, UK.
- [13] Sutton, R. S., & Barto, A. G. (1998). *Reinforcement Learning*. MIT Press.
- [14] Turney, P. (1996) Myths and legends of the Baldwin Effect. In *Proceedings of the Workshop on Evolutionary Computation and Machine Learning at the 13th International Conference on Machine Learning (ICML-96)*, pp. 135-142 Bary, Italy.
- [15] Turney, P., Whitley, D., & Anderson, R.(Eds.), (1996).*Special Edition of Evolutionary Computation, Evolution, Learning, and Instinct: 100 Years of the Baldwin Effect*.
- [16] Waddington, C., H., (1942). Canalization of development and the inheritance of acquired characteristics. *Nature*, 150, 563-565.
- [17] Watkins, C., J., C., (1989). *Learning From Delayed Rewards*. PhD thesis, King's College, Cambridge, UK.
- [18] Weismann, A. (1893). *The Germ-Plasm: A Theory of Heredity*. New York: Scribners.
- [19] Werner, G. M., & Dyer, M. G. (1991). Evolution of Communication in Artificial Organisms. In Langton, C, G., Taylor, C., Farmer, J. D., & Rasmussen, S. (Eds.), *Artificial Life II-SFI Studies in the Sciences of Complexity*, Vol. X, pp. 659-687 Redwood City, California, Addison-Wesley.

Appendix A

PROGRAM CODE

```
/*
Main Program
MSc Dissertation
Juan Pablo Calderon
*/

LIBRARIES
#include <stdio.h>
#include <time.h>
#include <stdlib.h>
#include <math.h>

CONSTANTS
#define WORLD_SIZE 200
#define NEIGHBOURHOOD 5 //visual and hearing field
#define FEMALES 1600
#define MALES 1600

//sex
#define MALE 1
#define FEMALE 0

#define INPUTS 33
#define SIGNALS 4

#define INPUT_M 4
#define OUTPUT_M 4
#define MSIZE 4*4

//#define INPUT_F 33
```

```

//#define OUTPUT_F 4
#define FSIZE INPUTS*4 //size of females Q table

//#define BIAS 1

//ACTIONS
#define STILL 2
#define FORWARD 0
#define LEFT 3
#define RIGHT 1

//DIRECTIONS
#define UP 0
#define DOWN 2

#define MAXQ          12 //range of Qvalues

//#define BETA 1

#define MUTATION 1 // over 1000
#define CROSSOVER 5 // over 100
#define SWITCH 0

#define TMAX 100000 //not use, was maximum time of simulation in timesteps
#define MAX_MATINGS          5000000 //stopping criteria

#define SAMPLET 100 //sampling rate to store info in files
#define T_SCATTER 10000 //sampling time for scatter plots
#define M_SCATTER 100000 //same as above but in terms of reproductions

//Qlearning constants
#define ALFA 0.15f //how much Q changes
#define GAMMA 0.98f //how much value has the maximum next Q
#define R 1.0f // reward

#define ON 1
#define OFF 0

#define T_LEARNING -1 //time step when males become plastic (-ve) for no learning
#define T_LEARNING_F          -1 //time step when females become plastic

#define GA_ON ON //turns on- off the GA
#define GA_F_ONLY OFF //if on only females are subject to GA
#define EXPLORATION ON //turns on off exploration on males
#define EXPLORATION_F          OFF //turns on off exploration on females
#define MIN_MATE          1 // Agents reproduce every MIN_MATE encounters

#define MIN_PROP 0.05f //protocols with less than this percentage are not recorded

```

```

//EXPLORATION CONSTANTS
#define MAXTEMP 10.0f
#define MINTEMP 0.5f
#define EXPSTEP 1.0f

/*****
AGENTS STRUCTURE
*****/

typedef struct{
    int input;
    int previousI; //previous input used for learning
    int output;
    int previousO;

    int x;
    int y;
    // int learn; //learning gene
    float genomeM[MSIZE];
    float genomeF[FSIZE];
    float Qvalue[INPUTS][4];
    int orientation;
    int age;
    int matings; //total number of encounters
    int exploration; //1 if exploration 0 if exploitation
    int mating_counter; //number of encounters since last reproduction
    float reward;
}typeagent;

/*****
GLOBAL VARIABLES
*****/

typeagent male[MALES];
typeagent female[FEMALES];

int mates; // total number of reproductions of the whole population
int LEARNING; // variable specifying if the population is learning
int LEARNING_F;
//int exploration;

/*****
DECLARATION OF FUNCTIONS
*****/

void createWorld();
void placeAgent(typeagent*);
//int checkneighbourhood(typeagent*);
void femaleLoop();
void getInput(typeagent*,typeagent,int,int,int);
void maleLoop();

```

```

void moveAgent(typeagent*);
void moveForward(typeagent*);
//void network(typeagent*,int);
//void resetMale();
//void readInput(typeagent*,float*,int);
//void readWeightsOutput(typeagent*,float*,int ,int);
//void readWeightsInner(typeagent*,float*,int ,int);
//void activationFunction(float* ,int );
void giveGenotype(typeagent*);
//void writeOutput(typeagent*,float*,int,int);
int distance(int,int);
int checkAgentNeighbourhood(typeagent * ,typeagent*,int);
//void updateWeightsInner(typeagent*,float,int,int);
//void updateWeightsOutput(typeagent*,int,int,int);
void updateQ(float r,float Q[INPUTS][4],int previous,int newInput,int output);
//void test();
void printGenome(typeagent);
void printAgent(typeagent);
//void testnet();
void statistics(int t);
void table(int t,int);
//int output(int out[OUTPUT_M]);
void printTable(int data[MALES][OUTPUT_M],int,FILE*);
//int mapping(int data[OUTPUT_M]);
float ageToTemp(typeagent*);//,int);
float matingsToTemp(typeagent*);//,int);
void boltzman(float*,float,float*);
void assignGenome(typeagent* ,int );
int maxQoutput(float [INPUTS][4],int);
void GA(typeagent*,typeagent*);
//void onePointCrossover(typeagent*,typeagent*,typeagent*,typeagent*);
void crossover(typeagent*,typeagent*,typeagent*,typeagent*);
void mutate(typeagent*);
void eraseFiles();
//void maleMapping(typeagent*);
void inOutMapping(typeagent*);//,int);
int inverseMapping(int [4]);
//void femaleMapping(typeagent*);
void scatter1();
void scatter2();
void newAgent(typeagent*,int);
float myrand();
void debug(int );

```

```

/*****

```

```

Main program

```

```

It creates the world, runs the main Loop where it
calls the male and female loop.
It also calls functions for saving the information

```

```

into files

*****/
void main(void)
{
    int t, counter = 0;

    srand(time(NULL));

    // male = (typeagent *) malloc (MALES * sizeof(typeagent));
    // female = (typeagent *) malloc (FEMALES * sizeof(typeagent));

    eraseFiles(); //opens and closes files to erase contents
    createWorld(); //initializes everything

    mates = 0; //total number of matings so far

    LEARNING = OFF; //learning initialized in OFF
    LEARNING_F = OFF;

    // printf("THE END!!!");

    for(t=0; mates<MAX_MATINGS ;t++)
    {
        if(t==T_LEARNING)
            LEARNING=ON; //activates learning males

        if(t==T_LEARNING_F)
            LEARNING_F=ON; //activates learning males

        femaleLoop(); //runs through females
        maleLoop(); //runs through males

        /* every T_SCATTER stores information about
           age and matings of population*/
        if( (t % T_SCATTER) == 0 && t > 0)
            scatter1(); //stores info to make scatter plots

        /*counter is increased by M_SCATTER every time a
           it stores info about matings and age of population*/
        if( mates > counter )
        {
            counter += M_SCATTER;
            scatter2(); //stores info to make scatter plots
        }

        /*Store information about the percentage of protocols*/

        if(t%SAMPLET == 0 && t>0)
        {

```

```

table(t,0); // makes tables of learned protocols

if(LEARNING)
    table(t,1); // makes tables of innate protocols.
}
}
}
/*****
debug function
prints a number in a file to help debugging
*****/
void debug(int function)
{
    FILE *f;

    f = fopen("debug","w");
    fprintf(f,"%d",function);
    fclose(f);
}
/*****
erases the contents of the files
*****/
void eraseFiles()
{
    FILE* file1;

    // file1=fopen("small","w");
    // fclose(file1);
    file1=fopen("table","w");
    fclose(file1);
    file1=fopen("proportion","w");
    fclose(file1);
    file1=fopen("graph","w");
    fclose(file1);
    file1=fopen("graph_gen","w");
    fclose(file1);
    file1=fopen("scatter_mates","w");
    fclose(file1);
    file1=fopen("scatter_time","w");
    fclose(file1);
}
/*****/
void resetMale() //resets inputs (it is never used)
{
    int i;
    for(i=0;i<MALES;i++)
        male[i].input=0;
}
/*****/
This function initializes a new agent

```

```

*****/
void newAgent(typeagent* agent,int sex)
{
    placeAgent(agent); //gives position

    /*assigns initial exploration if user specifies*/
    if(sex==MALE)
    {
        if(EXPLORATION)
            agent->exploration= ON;
        else
            agent->exploration=OFF;
    }
    if(sex==FEMALE)
    {
        if(EXPLORATION_F)
            agent->exploration= ON;
        else
            agent->exploration=OFF;
    }

    agent->input=0;
    agent->previousI =0;
    agent->output=0;
    agent->previousO = 0;
    agent->age=1;
    agent->matings=0;
    agent->mating_counter=0;
    agent->reward=0;
}

/*****
initializes agents with genomes, positions and Qtables
*****/
void createWorld()
{
    int i;

    for(i=0;i<(FEMALES);i++)
    {
        newAgent(&female[i],FEMALE); //initializes agent
        giveGenotype(&female[i]); //assigns random genome
        assignGenome(&female[i],FEMALE); //maps genome to Qtable
    }
    for(i=0;i<(MALES);i++)
    {
        newAgent(&male[i],MALE);
        giveGenotype(&male[i]);
        assignGenome(&male[i],MALE);
    }
}

```

```

}
/*****
Gives a position and orientation to the agent
*****/
void placeAgent(typeagent* agent)
{
agent->x=rand()%WORLD_SIZE;
agent->y=rand()%WORLD_SIZE;
agent->orientation=rand()%4; //not used for females

}
/*****
Main female loop
Runs thru all the females update weights if learning is on.
checks the for nearby males and output a signal
The input is received inside the checkAgentNeighbourhood function
*****/
void femaleLoop()
{
int i,m;

for(i=0;i<(FEMALES);i++)
{
/*if learning */
if(female[i].age>2 && LEARNING_F)
updateQ(female[i].reward ,female[i].Qvalue,female[i].previousI,
female[i].input,female[i].previous0);

/*store information for Qlearning algorithm*/
female[i].previousI = female[i].input;
female[i].previous0 = female[i].output;

female[i].reward =0; //initialize in 0

/*check neighbourhood for males, returns (closest male index +1)
if there is no male the input is 0.
inside this function it gets an input and runs the ga if necessary.*/

if(!(m = checkAgentNeighbourhood(&female[i],male,FEMALE)))
{
female[i].input = 0;
}

inOutMapping(&female[i]); //computes output, hence exploration.
female[i].age++;
}

}
/*****/

```

```

int distance(int X1,int X2)
// distance from agent 1 to agent 2
//simple cartesian distance in a toroidal world
{
int dist,distTemp=0;

//calculate distance from male i
dist = X2 - X1;

//check boundaries
if(X1 < 2)
{
distTemp = (X2 - (WORLD_SIZE)) - X1;
if (abs(distTemp) < abs(dist))
dist = distTemp;
}

if(X1 >= (WORLD_SIZE - 2) )
{
distTemp = X2 - (X1 - WORLD_SIZE);
if (abs(distTemp) < abs(dist))
dist = distTemp;
}

return dist;
}
/*****
// Runs thru all agents of the opposite sex,
// if one has the same position runs a ga and rewards,
// if not calls getinput with the closest mate index
*****/
int checkAgentNeighbourhood(typeagent* agent1,typeagent *agent2,int sex)
{

int i,distX,distY,m,mindistX,mindistY,OTHERS, found_one=0, that_one;

// which sex is it going to run through
if (sex == FEMALE)
OTHERS = MALES;
else
OTHERS = FEMALES;

//run thru all males

for(i=0;i<OTHERS;i++)
{
distX = distance(agent1->x,agent2[i].x); //distance in x
distY = distance(agent1->y,agent2[i].y); //distance in y

//calculate distance from male i

```

```

        //if male is within site
        if( abs (distX ) <= 2 && abs (distY ) <= 2)
    {
        //male is over female
        if(distX == 0 && distY == 0)
        {
            //run a GA only if you are female, see document for explanation.
            if(sex==FEMALE)
        {
            GA(agent1,&agent2[i]);

            agent1->reward = R; //reward female
            male[i].reward = R; //reward male
        }

            return 0;
        }
        //check the closer neighbourhood (cross)
        if(distX == 0 || distY == 0 && (!found_one))
        {
            //if it is right next to the babe stores its information to get input later.
            if( (abs(distX ) == 1 || abs (distY ) == 1) && (!found_one) )
        {
            found_one=1;
            that_one=i+1;
        }
            else //also stores info because its the closest so far
        {
            m=i;
            mindistX=distX;
            mindistY=distY;
        }
        }
        else //checks corners of visual field
        {
            if( !(mindistX == 0 || mindistY == 0) && (!found_one) )
        {
            m=i;
            mindistX=distX;
            mindistY=distY;
        }
        }
        }
        }
        /* Call get input with the closest mate found*/

        if(found_one)
        {
            getInput(agent1,agent2[that_one - 1],distX,distY,sex);
            return that_one;
        }
    }
}

```

```

    }

    if( abs(mindistX) <=2 && abs(mindistY)<=2 )
    {
        getInput(agent1,agent2[m],mindistX,mindistY,sex);
        return m+1;
    }

    return 0; //found nothing
}
/*****
Gets the input according to the closest mate
see document for explanation of the mapping
*****/
void getInput(typeagent* agent1,typeagent agent2,int distX,int distY,int sex)
{

    if (sex == FEMALE)
    {

        agent1->input = 0;

        if(distY > 0)
        {
            if( distX > 0 )
                agent1->input=1;
            else
            {
                if(distX == 0)
                    agent1->input=2;
                else
                    agent1->input=3;
            }
        }

        if(distY == 0)
        {
            if( distX > 0 )
                agent1->input=4;
            else
                agent1->input=5;
        }

        if(distY < 0)
        {
            if( distX > 0 )
                agent1->input=6;
            else
            {
                if(distX == 0)
                    agent1->input=7;
            }
        }
    }
}

```

```

        else
agent1->input=8;
    }
}

    agent1->input += 8 * (agent2.orientation);
    }
/*males get the input straight from the female*/
if(sex == MALE)
    {
        if(agent2.output >= 0 && agent2.output < 4) //just in case
agent1->input = agent2.output;
        else
agent1->input = 0;
    }
}
/*****
Randomly initializes a genome
*****/
void giveGenotype(typeagent* agent)
{
    int i;

    for(i=0;i<MSIZE;i++)
    {
        agent->genomeM[i]=myrand();
    }

    for(i=0;i<FSIZE;i++)
    {
        agent->genomeF[i]=myrand();
    }

    /*uncomment line below to have genetically specified plasticity*/
    // agent->learn=rand()%2;
}

/*****/
void maleLoop()
{
    int i,m;

    for(i=0;i<MALES;i++)
    {
        /*if learning */
        if(male[i].age>2 && LEARNING)
// if(male[i].age>2 && male[i].learn) for genetically specified plasticity
updateQ(male[i].reward ,male[i].Qvalue,male[i].previousI
,male[i].input ,male[i].previous0);

```

```

/*store information for Qlearning algorithm*/
male[i].previousI = male[i].input;
male[i].previousO = male[i].output;

male[i].reward=0;//initialize in 0

//checkneig.. returns 0 if no mate nearby then input ={0}
/*check neighbourhood for males, returns (closest male index +1)
if there is no male the input is 0.
inside this function it gets an input and runs the ga if necessary.*/

    if(!(m=checkAgentNeighbourhood(&male[i],female,MALE)))
{
    male[i].input=0;
}

    inOutMapping(&male[i]); //computes output, hence exploration.
    moveAgent(&male[i]);
    male[i].age++;
}
}
/*****
Moves an agent or rotates him

*****/
void moveAgent(typeagent* mal)
{
    if( mal->output==FORWARD )
    {
        moveForward(mal);
        return;
    }
    else
        if( mal->output==LEFT )
        {
            if(mal->orientation == 0)
                mal->orientation = 3;
            else
                mal->orientation --;
            return;
        }

        else
        {
            if( mal->output==RIGHT )
            {
                if(mal->orientation == 3)
                    mal->orientation = 0;
                else

```

```

        mal->orientation ++;
    return;
}
}
}
/*****
moves a male forward, with wrap around
*****/
void moveForward(typeagent* mal)
{
    if( mal->orientation == UP)
    {
        if( mal->y == 0)
mal->y = (WORLD_SIZE - 1);
        else
mal->y--;
    }
    if( mal->orientation == DOWN )
    {
        if( mal->y == (WORLD_SIZE - 1))
mal->y = 0;
        else
mal->y++;
    }
    if( mal->orientation ==LEFT)
    {
        if( mal->x == 0)
mal->x = (WORLD_SIZE - 1);
        else
mal->x--;
    }

    if( mal->orientation == RIGHT)
    {
        if( mal->x == (WORLD_SIZE - 1))
mal->x = 0;
        else
mal->x++;
    }
}
/*****
Prints agent information into the screen
for debugging purposes
*****/
void printAgent(typeagent agent)
{

int i,j;
FILE *apt;

```

```

apt=fopen("agente86","w");

fprintf(apt,"input %d\noutput %d\nx %d\ny %d\n",
agent.input ,agent.output ,agent.x,agent.y);

fprintf(apt,"\ngenomeM\t");
for(i=0;i<MSIZE;i++)
fprintf(apt,"%f\t",agent.genomeM[i]);

fprintf(apt,"\ngenomeF\t");
for(i=0;i<FSIZE;i++)
fprintf(apt,"%f\t",agent.genomeF[i]);

fprintf(apt,"\nQ values\t");
for(i=0;i<INPUTS;i++)
for(j=0;j<4;j++)
fprintf(apt,"%f\t",agent.Qvalue[i][j]);

fprintf(apt,"\norientation %d\tage %d\texp %d\nmatings %d\tcounter %d",
agent.orientation,agent.age,agent.exploration,agent.matings,agent.mating_counter);

fclose(apt);
}
/*****
For debugging
prints genome
*****/
void printGenome(typeagent agent)
{
int i;

for(i=0;i<FSIZE;i++)
printf("%d ",agent.genomeF[i]);
}
/*****
Genetic Algorithm
*****/
void GA(typeagent* fem,typeagent* agent2)
{
int index_male,index_female,flag1=0,flag2=0;
typeagent cM,cF;

// mates++;

//increases counters
agent2->matings++;
agent2->mating_counter++;

fem->matings++;
fem->mating_counter++;

```

```

//new position for parents
placeAgent(agent2);
placeAgent(fem);

// if the conditions for a GA to take place are true
if( ( GA_ON || GA_F_ONLY) &&
    (fem->mating_counter >= MIN_MATE || agent2->mating_counter >= MIN_MATE ))
{
    //increase reproduction counter
    mates++;

    index_male=rand()%MALES; //random death
    index_female=rand()%FEMALES;

    // if females reproduce
    if( flag1 = (fem->mating_counter < MIN_MATE) ) //GA_F_ONLY)
        cF=female[index_female]; //store a temp copy of the male

    // if males reproduce
    if(flag2 = ( agent2->mating_counter < MIN_MATE || GA_F_ONLY))
        cM=male[index_male]; //store a temp copy of the female

    //initialize new agents
    newAgent(&male[index_male],MALE);
    newAgent(&female[index_female],FEMALE);

    //crossover gives a genome to the agents
    crossover(agent2,fem,&male[index_male],&female[index_female]);

    mutate(&male[index_male]);
    assignGenome(&male[index_male],MALE); //creates qTable

    mutate(&female[index_female]);
    assignGenome(&female[index_female],FEMALE);

    //if female didn't reproduced
    if(flag1)//fem->mating_counter < MIN_MATE )//GA_F_ONLY)
        female[index_female]=cF;

    //if male didn't reproduced
    if(flag2)//agent2->mating_counter < MIN_MATE || GA_F_ONLY)
        male[index_male]=cM;

    //reset counter for number of matings before reproduction
    if(agent2->mating_counter >= MIN_MATE)
        agent2->mating_counter=0;

    if(fem->mating_counter >= MIN_MATE)
        fem->mating_counter=0;
}

```

```

}
}
/*****
Every gene has a probability of being crossover between parents
*****/
void crossover(typeagent* parent1,typeagent* parent2,typeagent* child1,typeagent* child2)
{
int i;

//Male part of the genome
for(i=0;i<MSIZE;i++)
{
if(rand()%100<CROSSOVER)
{
child1->genomeM[i]=parent2->genomeM[i];
child2->genomeM[i]=parent1->genomeM[i];
}
else
{
child1->genomeM[i]=parent1->genomeM[i];
child2->genomeM[i]=parent2->genomeM[i];
}
}
//female part of the genome
for(i=0;i<FSIZE;i++)
{
if(rand()%100<CROSSOVER)
{
child1->genomeF[i]=parent2->genomeF[i];
child2->genomeF[i]=parent1->genomeF[i];
}
else
{
child1->genomeF[i]=parent1->genomeF[i];
child2->genomeF[i]=parent2->genomeF[i];
}
}
}
/*****
Mutations consist of tossing a random gene
*****/
void mutate(typeagent* child1)
{
int i;

for(i=0;i<MSIZE;i++)
{
if(rand()%1000 <= MUTATION)
child1->genomeM[i]= myrand();
}
}

```

```

    }
    for(i=0;i<FSIZE;i++)
    {
        if(rand()%1000 <= MUTATION)
child1->genomeF[i]=myrand();
    }
    //uncomment for genetically specified plasticity
    //if(rand()%1000 <= MUTATION)
    // child1->learn=rand()%2;

}
/*****
Gives random numbers between 1 and MAXQ
*****/
float myrand()
{
return MAXQ * ( (float) (rand() % 100) ) / 100;
}
/*****
This function is not used any more, it stored data in a file
*****/
void statistics(int t)
{
    int i,j,k,count[4]={0};
    typeagent pepe;
    float average[4]={0};
    FILE* pointer;

pointer=fopen("small","a");
fprintf(pointer,"\n TIME %d\n",t);

for(j=0;j<SIGNALS;j++)
{
for(k=0;k<4;k++)
count[k]=0;

for(i=0;i<MALES;i++)
{
pepe=male[i];
pepe.input=j;

inOutMapping(&pepe);
count[pepe.output]++;
}

for(k=0;k<4;k++)
average[k]+=count[k];

fprintf(pointer,"%d\t%d\t%d\t%d\t%d\n",j,count[0],count[1],count[2],count[3]);
}

```

```

for(k=0;k<4;k++)
{
average[k]/= SIGNALS*(MALES);
average[k]*= 100;
}
fprintf(pointer,"\n-----\npercentage\t%.2f\t%.2f\t%.2f\t%.2f\n",
average[0],average[1],average[2],average[3]);

fprintf(pointer,"\n\n*****\n\n");
fclose(pointer);
}

/*****

/*****
counter stores the response each male have upon receiving
each of the four possible signals
gen specifies if the innate protocols are going to be stored
instead of the learned ones.
*****/
void table(int t,int gen)
{
int m,s,counter[MALES][SIGNALS];
typeagent pepe; //temp agent
FILE* pointer;

if(gen)
pointer=fopen("graph_gen","a");

else
pointer = fopen ("graph","a");
//run thru males
for(m=0;m<MALES;m++)
{
pepe=male[m];
if(gen) assignGenome(&pepe,MALE); //innate behavior

//for each signal stores output
for(s=0;s<SIGNALS;s++)
{
counter[m][s]=maxQoutput(pepe.Qvalue,s);
}
}

printTable(counter,t,pointer); //store info in files
fclose(pointer);
}
/*****
This function add the number of males that follow

```

```

each of the 256 different protocols and stores it in
a file
table is used to store the information
data comes from unction table
*****/
void printTable(int data[MALES][SIGNALS],int t,FILE* pointer3)
{
    int i,j,k,l,table[256][2]={0},index,data2[4];
    float ratio;
    FILE* pointer,*pointer2;//,*pointer3,*pointer4;

    pointer=fopen("table","a"); //stores werner and dyers tables
    pointer2=fopen("proportion","a"); //stores the percentage of population
    // pointer3=fopen("graph","a");
    // pointer4=fopen("graph_gen","a");

    fprintf(pointer,"\n TIME %d\n",t);

    if(t==0)
        fprintf(pointer2,"\t\tprotocol\tpercentage\n");

    //creates all possible protocol names
    for(i=0;i<4;i++)
    {
        for(j=0;j<4;j++)
    {
        for(k=0;k<4;k++)
        {
            for(l=0;l<4;l++)
    {
        data2[0]=1;
        data2[1]=k;
        data2[2]=j;
        data2[3]=i;
        index=mapping(data2); //turns the array into a int number
        table[index][1]=(data2[0]+10*data2[1]+100*data2[2]+1000*data2[3]);
        //store names
    }
        }
    }
    }
    //only the first time stores names
    if(t==SAMPLET)
    {
        fprintf(pointer3,"0\t0\t");
        for(i=0;i<256;i++)
    fprintf(pointer3,"%d\t",table[i][1]);
    }
    fprintf(pointer3,"\n");
}

```

```

for(i=0;i<MALES;i++)
{
    index=mapping(data[i]); //index of the particular protocol.
    table[index][0]++;

}
//the table that is going to be printed has dimensions 32x8
//prints werner and dyer tables see Werner and Dyer (1991).
for(i=0;i<32;i++)
{
    for(j=0;j<8;j++)
{
    index=j+8*i;

    if(table[index][0])
        fprintf(pointer,"%d:%04d\t",table[index][0],table[index][1]);
    else
        fprintf(pointer,"0\t");
}
    fprintf(pointer,"\n");
}

fprintf(pointer2,"T = %d\tM = %d\n",t,mates);
fprintf(pointer3,"%d\t%d\t",t,mates);

//for all protocols
//stores a table of number of protocols against popularity
//adds a new line every time the function is called
for(i=0;i<256;i++)
{
    ratio=((float)table[i][0])/MALES;

    if(ratio>MIN_PROP)
{
    fprintf(pointer2,"\t\t%04d\t%.2f\n",table[i][1],ratio);
    fprintf(pointer3,"%.2f\t",ratio);
}
    else
{
    fprintf(pointer3,"0\t");
}
}

fprintf(pointer3,"\n");
fprintf(pointer2,"\n");

fclose(pointer);
fclose(pointer2);
// fclose(pointer3);
}

```

```

/*****
int inverseMapping(int table[4])
{
int same,j,protocol[4]={0},sum_protocol[4]={0};

same=0;

for(j=0;j<4;j++)
{
protocol[j]=((int)(table[1]/(pow(10,j))))%10;
sum_protocol[protocol[j]]++;
}

for(j=0;j<4;j++)
{
if(sum_protocol[j]==4)
return 1;
if(sum_protocol[j]==3)
return 2;
if(sum_protocol[j]==2 || sum_protocol[j]==1)
same++;
}
return same;
}
/*****
This function returns a number describing the
protocol. maps the data into a number from 0 to 255
each data is between 0 and three
*****/
int mapping(int data[OUTPUT_M])
{
return data[0]+4*data[1]+16*data[2]+64*data[3];
}
/*****
Computes the output, and temperature if exploration is ON
to calculate the probabiltiy of taking an action from a
boltzman distribution of the Qvalues
*****/
void inOutMapping(typeagent* agent)
{

int i;
float dice,T,p[4];

//if exploration
if(agent->exploration == ON)
{

dice = ((float)(rand()%100))/100; //random number

```

```

    T = matingsToTemp(agent); //compute temp.
    // T = ageToTemp(agent);
    boltzman((agent->Qvalue[agent->input]),T,p); //returns probabilities using a BDF

    //calculate an outputs
    for(i=0;i<4;i++)
{
    if(p[i] > dice || i==3)
    {
        agent->output = i;
        return;
    }
    else
    {
        dice -= p[i];
    }
}
//when exploitation
else
{
    agent->output = maxQoutput(agent->Qvalue,agent->input);
}
}
/*****
returns the index of the maximum Qvalue
*****/
int maxQoutput(float Q[INPUTS][4],int input)
{
    int i,max=0;
    for(i=1;i<4;i++)
    {
        if( Q[input][i] > Q[input][max] )
max=i;
    }
    return max;
}
/*****
void femaleMapping(typeagent* agent)
{
agent->output=agent->genomeF[agent->input];
}
*****/
Copies genome into Q-value table
*****/
void assignGenome(typeagent* agent,int sex)
{
int i,j;

```

```

if (sex==MALE)
{
for(i=0;i<(4);i++)
{
for(j=0;j<4;j++)
{
agent->Qvalue[i][j]=agent->genomeM[4*i+j];
}
}
}
else
{
for(i=0;i< INPUTS ;i++)
{
for(j=0;j<4;j++)
agent->Qvalue[i][j]=agent->genomeF[4*i+j];
}
}
}
/*****
Boltzman distribution function
*****/
void boltzman(float *Q,float T,float *P)
{
int i;
float sum=0,E[4];

for(i=0;i<4;i++)
{
E[i]=(float) exp(Q[i]/ T);
sum+=E[i];
}

for(i=0;i<4;i++)
{
P[i]=E[i]/sum;
}

}
/*****
mapping from age to temp, this is not used anymore
*****/
float ageToTemp(typeagent* agent)//,int LEARN)
{
float temp;
//return (10/((float)age)) + MINTEMP;

if((temp=(MAXTEMP - EXPSTEP * (agent->age -1) )) >= MINTEMP)// && LEARN)
return temp;
}

```

```

else
{
    agent->exploration = OFF;
    return MINTEMP;
}
}

/*****
linear mapping from number of matings to temperature
*****/
float matingsToTemp(typeagent* agent)
{
    float temp;
    //return (10/((float)age)) + MINTEMP;

    if((temp=(MAXTEMP - EXPSTEP * (agent->matings) )) >= MINTEMP)// && LEARN)
        return temp;
    else
    {
        agent->exploration = OFF;
        return MINTEMP;
    }
}

/*****
Q learning algorithm
*****/
void updateQ(float r,float Q[INPUTS][4],int previous,int newInput,int output)
{
    int i;
    float max=0;

    for(i=0;i<4;i++)
    {
        if(max<Q[newInput][i])
            max=Q[newInput][i];
    }
    Q[previous][output] += ALFA * ( ( r + GAMMA * max ) - Q[previous][output] );
}

/*****
Store information for constructing scatter plots
based on a time sampling
*****/
void scatter1()
{
    FILE* pointer;
    int i;

    pointer = fopen("scatter_time","a");

    for(i=0;i<MALES;i++)

```

```

        fprintf(pointer,"%d\t%d\n",male[i].age,male[i].matings);

fprintf(pointer,"\n");

fclose(pointer);
}
/*****
Store information for constructing scatter plots
based on a matings sampling
*****/
void scatter2()
{
    FILE* pointer;
    int i;

    pointer = fopen("scatter_mates","a");

    for(i=0;i<MALES;i++)
        fprintf(pointer,"%d\t%d\n",male[i].age,male[i].matings);

    fprintf(pointer,"\n");

    fclose(pointer);
}
/*****/

```

Appendix B

MATLAB CODE

B.1 plot_protocols.m

```
%Plots accumulated proportions of protocols,  
%and protocol types, ageinst average number of matings  
%and time  
%reads information from file 'graph'  
  
close all  
clear data  
  
data=load('graph'); %load data  
time =data(2:end,1); %array of time  
matings=data(2:end,2)/1600; %array of average number of matings  
%stop=100;  
step=60; %sampling  
pos=-1; %legend position  
  
i=1:length(data(2,3:end));  
  
kind=sameKind(data(1,3:end)); %gets array of protocol classes  
sum=zeros(length(data(2:end,1)),4); %for accumulating proportions  
  
figure(1);  
% Plots protocol types  
  
for i=3:length(data(2,:)) %run thru protocols  
    if max(data(2:end,i))>0 %if not empty  
        if kind(i-2)==0 % 1SP  
            a=plot(matings(1:step:end),data(2:step:end,i),'k-.');  
            sum(:,1)=sum(:,1)+data(2:end,i);  
        elseif kind(i-2)==1 %2SP  
            b=plot(matings(1:step:end),data(2:step:end,i),'k--');
```

```

        sum(:,2)=sum(:,2)+data(2:end,i);
    elseif kind(i-2)==2    %3SP
        c=plot(matings(1:step:end),data(2:step:end,i),'k-');
        sum(:,3)=sum(:,3)+data(2:end,i);
    elseif kind(i-2)==3    %4sp
        d=plot(matings(1:step:end),data(2:step:end,i),'k:');
        sum(:,4)=sum(:,4)+data(2:end,i);
    end
    hold on;
end
end

%prints all protocols
for i=3:length(data(2,:))
    if max(data(2:end,i))>0
        data(1,i)
    end
end

%to make legend
a=plot(0,0,'k-.');
hold on
b=plot(0.0,'k--');
hold on
c=plot(0,0,'k-');
hold on
d=plot(0,0,'k:');

%title('Phenotype');
legend([a,b,c,d],'1 signal protocol','2 signal protocol',...
    '3 signal protocol','4 signal protocol',pos);
xlabel('Average number of mates per individual');
ylabel('percentage of population');
axis([0 5000000/1600 0 1])

figure(2);
%subplot(3,1,2);

%plot accumulated sum
a=plot(matings(1:step:end),sum(1:step:end,1),'k-.');
hold on
b=plot(matings(1:step:end),sum(1:step:end,2),'k--');
hold on
c=plot(matings(1:step:end),sum(1:step:end,3),'k-');
hold on
d=plot(matings(1:step:end),sum(1:step:end,4),'k:');
hold on

%title('sum of above');
legend([a,b,c,d],'1 signal protocol','2 signal protocol',...

```

```

    '3 signal protocol', '4 signal protocol', pos);
xlabel('Average number of mates per individual');
ylabel('percentage of population');
axis([0 5000000/1600 0 1])

figure(3);

finalT=max(time);
%subplot(3,1,3);

%plot accumulated sum different time scale
a=plot(time(1:step:end),sum(1:step:end,1),'k-');
hold on
b=plot(time(1:step:end),sum(1:step:end,2),'k--');
hold on
c=plot(time(1:step:end),sum(1:step:end,3),'k-');
hold on
d=plot(time(1:step:end),sum(1:step:end,4),'k:');
hold on

%title('same above different time scale')
legend([a,b,c,d], '1 signal protocol', '2 signal protocol', ...
    '3 signal protocol', '4 signal protocol', pos);
xlabel('Time');
ylabel('percentage of population');
axis([0 finalT 0 1])

```

B.2 sameKind.m

```

% returns the compexity of the protocol
function [sum]=sameKind(data)

data;
a(1,:)=mod(data,10);
a(2,:)=mod((data - a(1,:))/10,10);
a(3,:)=mod((data -10*a(2,:) -a(1,:))/100,10);
a(4,:)=mod((data - 100*a(3,:) -10*a(2,:) - a(1,:))/1000,10);

s = sort(a,1);

sum = (s(1,:)~s(2,:)) + (s(2,:)~s(3,:)) + (s(3,:)~s(4,:));

```

B.3 scatter_plot.m

```

% This program creates the scatter plots

clear all

```

```

close all

points=1600;
agents=1600;

dim=1;
maxX=12000%2450;
maxY=250;

d1=5;
d2=10;
d3=49;

time_1 = d1 * 100000 / agents
time_2 = d2 * 100000 / agents
time_3 = d3 * 100000 / agents

data = load('scatter_mates');

data1=data(d1*agents:d1*agents + points,:);
data2=data(d2*agents+1:d2*agents + points,:);
data3=data(d3*agents+1:d3*agents + points,:);

p = polyfit(data1(:,1),data1(:,2),dim);
p2 = polyfit(data2(:,1),data2(:,2),dim);
p3 = polyfit(data3(:,1),data3(:,2),dim);

x=1:maxX;
a=plot( x ,p(1)*x + p(2) , 'k--' )%p(1)*x.^3+p(2)*x.^2+p(3)*x + p(4), 'k:');
hold on;
%b=plot( x , p2(1)*x.^3+p2(2)*x.^2+p2(3)*x + p2(4), 'k--');
%hold on
c=plot( x , p3(1)*x + p3(2) , 'k-' )%p3(1)*x.^3+p3(2)*x.^2+p3(3)*x + p3(4), 'k-');

hold on
e = plot( data1(:,1), data1(:,2), 'ks')%x , p(1)*x.^3+p(2)*x.^2+p(3)*x + p(4), 'k:');
hold on
%plot( data2(:,1), data2(:,2), 'ks' )
%hold on
f = plot( data3(:,1), data3(:,2), 'k+' )

%hold on
%figure
%plot(data(d1*agents:d1*agents + points,1),data(d1*agents:d1*agents + points,2), 'b. ');
%hold on
%plot(data(d2*agents+1:d2*agents + points,1),data(d2*agents+1:d2*agents + points,2), 'rs');
legend([e,a,f,c],'', 'Early Stage on evolution', '', 'Late Stage on evolution', 4)
xlabel('agents age');
ylabel('number of mates');

```

```
axis([0 maxX 0 maxY])
```