

Generalized *Win-Stay, Lose-Shift* is Robust in the Repeated Prisoners' Dilemma with Noise Played by Multi-state Automata

Norman Siebrasse
Faculty of Law
University of New Brunswick
siebrass@unb.ca

Abstract

The evolution of strategies in the Repeated Prisoners' Dilemma with noise is examined using a computer simulation which uses a genetic algorithm to introduce new strategies. It is known from previous simulations that Win-Stay, Lose-Shift is robust in simulations with noise and memory one (that is, recalling at most the previous round of play): Nowak & Sigmund, "A strategy of win-stay, lose-shift that outperforms tit-for-tat in the Prisoner's Dilemma game" 364 Nature 56 (1993). The simulation described in this paper finds that a generalized form of Win-Stay-Lose-Shift is robust when memory is extended. The robust strategy has three features: a "cooperative loop", which cooperates with a cooperative opponent; a "sucker exploitation loop", which defects against an opponent who continues to cooperate; and a "punishment circuit", which defects against a defecting opponent for a set number of plays before returning to the cooperative loop. The strategy is robust in that it dominates the population for extended periods. The degree of dominance of the strategy, the number of plays in the punishment circuit, and the stability of the strategy are all functions of the payoff parameters, the noise level, the mutation rate, the number of individuals players in the population, and the number of plays per interaction. The relationships are discussed. Some possible implications for the evolution of concepts of justice are raised.

Introduction

The repeated Prisoner’s Dilemma is a central model for the evolution of cooperation between selfish agents. Interest in evolutionary models of the development of strategies was sparked by Axelrod’s famous computer tournaments, which indicated that the intuitively appealing strategy of Tit-for-Tat was robust. The conclusions drawn from that seminal tournament are undermined by the fact that the selection of strategies did not effectively search the strategy space. It is also more realistic to introduce noise into the model. Subsequent simulations have used genetic algorithms to generate new strategies. This paper describes such a simulation, which extends previous work in two main respects. First, it allows strategies to be conditional on a much longer history of play than previous simulations. This is a direct consequence of the second main difference, which is that the strategies are represented as explicit finite state automata, rather than as a vector of responses to previous play histories. This greatly facilitates interpretation of the resultant strategies, especially with long memory of previous play.

It is known that the strategy Win-Stay, Lose-Shift is robust in the repeated Prisoner’s Dilemma with noise when strategies are conditional only on the previous play of the players: Nowak & Sigmund (1993). We find that a generalized form of Win-Stay, Lose-Shift is highly robust. This generalized Win-Stay, Lose-Shift has three features: a “cooperation loop”, which cooperates with a cooperative opponent; a “sucker exploitation loop”, which defects against an opponent who continues to cooperate; and a “punishment circuit”, which defects against a defecting opponent for a set number of plays before returning to the cooperative loop. This class of strategies dominates the population over a wide range of payoff values and noise levels. The degree of dominance of the strategy and the number of plays in the punishment circuit is a function of the temptation to defect, the reward to cooperation, and the noise level. (In this paper the phrase “a strategy dominates the population” is used to indicate that a large proportion of the population plays the strategy. It is unrelated to the technical game theoretical use of a dominant strategy.) This result is also consistent with and generalizes the findings of Lindgren (1991) and Axelrod (1987).

The Simulation

In prior simulations a strategy has been represented by a vector of responses to play histories. For example, representing the play history by a pair (a,b) where ‘a’ and ‘b’ are the previous plays of players A and B, then Tit-for-tat is specified for player A by (1,0,1,0). In contrast the program used in the simulations discussed in this paper represents each strategy explicitly as a finite state automata with a maximum of S states (memory (S-1)): the play in each state (P(s), $s \leq S$) is specified, as well as the state which results when the opponent plays Defect (0) or Cooperate (1) (the ‘exit states’ E0(s), E1(s)). Without loss of generality, the initial state is always state 1. For example, with S=3, Tit-for-Tat is represented as shown below, where “-1” indicates an inactive state:

State 1			State 2			State 3		
Play	E(1)	E(0)	Play	E(1)	E(0)	Play	E(1)	E(0)
1	1	2	0	1	2	-1	-1	-1

The simulation also tracks the amount of time which the player spends in each state. An advantage to this representation is that it can make the resultant strategies easier to interpret, as state diagram can be drawn directly.

A population consists of a number of players, N. In each generation each player plays one round against every other player, where a round consists of a specified number of plays (“Pr” – plays/round). Noise is introduced so that on each play each player has a probability “n” of making an unintended play. Payoffs to both players are according to the noisy play, and not the intended play.

Further, a noise event does not change the state of the player making the play, but simply communicates the wrong play to the other player. In other words, a player’s next play is conditional on its intended play in the previous round and the actual play of the other player in

the previous round. In contrast, simulations which represent the strategy as a vector of responses to play history use the actual play of both players. For this reason the two representations are not completely equivalent. However, it is not clear which of these representations, if either, is generally a better modelling choice. The vector representation uses the actual play history because without an express state representation of the strategy, there is no way of knowing the 'intended' play. In contrast, when the strategies are represented by an explicit finite state automata, making play conditional on actual play is difficult. If a strategy was in a Cooperate state, and noise causes it to play Defect, the strategy must 'pretend' that it was actually in Defect state on that play. In other words, it must jump from the Cooperate state to a Defect state. In a multi-state strategy with several Defect states, the choice of which state to jump to is arbitrary. It would be possible to account for this by specifying the new state after an unintentional play. This was not done because it would have added complexity and because, as noted, it is not clear which option is generally a better modelling choice.

Payoffs are R to both players if both cooperate, P to both if the both defect, and S to the cooperator and T to the defector if one cooperates and the other defects. Without loss of generality we fix $S=0$ and $P=1$. Defining $T' \equiv T/R$, the Prisoner's Dilemma is then defined by $R>1$ and $1<T'<2$. (Hereafter S is used solely to denote the maximum number of states, and not the sucker's payoff.) After each generation strategies reproduce by the usual dynamic whereby the number of offspring is proportional to the success in the previous round. There is no cutoff below which strategies are automatically eliminated. Rather, when (as is usual) proportional shares result in some strategies having a fractional entitlements to a player in the next generation, this player is allocated to one of the strategies probabilistically, with the probability of claiming the player being equal to the size of the fraction. For example, if $N=30$ and there are three strategies, A, B, C, which are entitled to 20.33, 9.33 and 0.33 players in the next generation, strategies A and B will be allocated at least 20 and 9 players respectively, and each strategy will have a 1/3 probability of having the remaining player allocated to it.

After reproduction, each player has a probability M of dying and being replaced by a new strategy. New strategies are generated by three mechanisms: random mutation and two types of genetic operator: crossover of two existing strategies and point mutation of a single existing strategy. In random mutation, which is also used to generate initial strategies, 0 or 1 is randomly assigned to each Play variable, and a number from 1 to S is randomly assigned to each Exit 0/1 variable. The strategy is then simplified for clarity and to avoid duplicative strategies which differ only in their formal representation.

In crossover, two existing players are selected at random from the population to be “parents” and the new player is constructed by randomly combining states from the parents. When the offspring inherits a state from one parent, it inherits both the play variable and the two exit variables. If the parents have different numbers of active states, the offspring strategy may lead to inactive states. To avoid this, all inactive states in the offspring are randomized and then the offspring is then simplified in the same manner as a randomly generated strategy. Note that the form of crossover is

In point mutation a single existing player is selected at random to be the parent, and the new player is constructed by altering a single active locus, where each $P(s)$ and $E0/1(s)$ is considered a locus. There is a $1/3$ probability that the mutation will change a single Play variable and a $2/3$ probability that the mutation will change a single Exit variable. When the Play variable is to be changed, the point mutation algorithm selects an active Play variable at random and flips it. When the Exit variable is to be changed, a single Exit state variable is selected and replaced with a randomly selected active state. There is no requirement that a mutant strategy be novel either in the sense that it is not currently represented in the population, or in that it is different from the parent strategy.

There are of course a number of types of genetic operator which can be used, and there is as yet no consensus as to which is generally best. In the most common crossover operator in which a

random locus is selected and contiguous blocks of genotype code from either side of that locus are combined to form the offspring. Since the rationale for the crossover operator is to combine successful functional blocks from different parents, this type of crossover is appropriate when the strategy is represented as a vector of responses to play history. However, in the express state representation used in this simulation, contiguously numbered states are not necessarily functionally contiguous, and uniform crossover, as described above, is more appropriate.

The point-mutation operator used in this simulation changes a single locus with a 100% probability. The most obvious alternative would alter each locus with a much smaller probability. This alternative, using a range of point mutation probabilities, was tested on a selected set of parameters, with results which were qualitatively similar to those obtained using single point-mutation. The same class of strategies dominated, although the degree of dominance and the stability varied. This indicates that the results obtained are reasonably robust with respect to the genetic operators used.

The results of the simulation are reported in two ways. At specified intervals the state representation of the all the different strategies in the population and the number of player using that strategy is saved to file. Also a specified intervals, the proportion of the population using strategies with specified features, such as a Grim loop (ie a state which plays Defect and exits to itself on defection or cooperation by the other player), is calculated and saved to file. The features selected are those which appeared, from direct examination of the state representation, to be of interest.

The Basic Strategy

While it is traditional to bestow some descriptive name on strategies, for lack of imagination I

have named the strategy which dominates the repeated PD in this simulation “Basic”.¹ (Suggestions are welcome.) The Basic class of strategies has a cooperative loop, a sucker loop, and a punishment circuit. In the cooperative loop (“C loop”) the Basic strategy cooperates so long as the other player replies in kind. On a defection by the other strategy, the Basic strategy exits to the punishment circuit, where it defects a number of times before return to the C loop is possible. The punishment circuit contains a sucker exploitation loop (“S loop”), in which Basic exploits any excessive cooperativeness on the part of the opponent by playing defect so long as the opponent plays cooperate.

In the strict form, shown in Fig. Basic(5) Strict, the Basic strategy exits directly from the Cooperation loop to the Sucker exploitation loop when the opponent defects instead of cooperating.

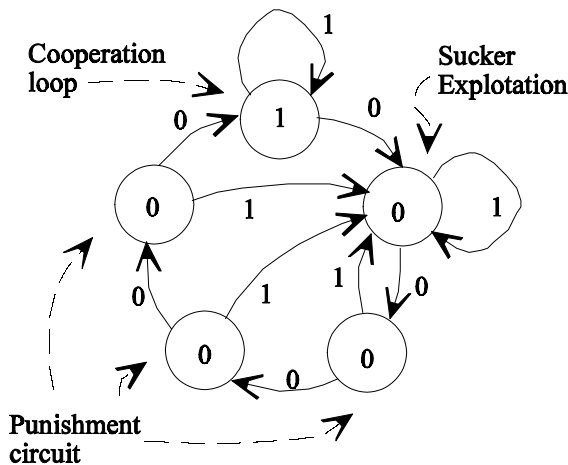


Figure Basic(5) Strict

cooperating. On any sign of weakness during the punishment circuit – if the opponent cooperates – the strategy returns directly to the S loop and the entire cycle must be repeated before the strategy is ready to return to mutual cooperation. The number of states in the punishment circuit varies with the payoff parameters. The example shown in Figure Strict Basic(5) has a four state punishment circuit.

The length of Basic, L , is defined as one (for the C loop) plus the length of the punishment circuit (algorithmically, the number of states which must be passed through to return to the C loop by playing 1). The Basic strategy of length L is represented as Basic(L).

¹While the strategy is a generalization of Win-Stay, Lose-Shift, “Generalized Win-Stay, Lose-Shift” is not appropriate because it is not accurately descriptive of the punishment phase of the strategy.

The Basic class of strategies consists of variants on this strict form. The algorithmic definition used to generate the statistics reported in this paper is as follows:

- There is a C loop state
- There is an S loop
- When the strategy is in the C loop and Defect is played against it, it exits to a state, “Initial Exit” which plays Defect.
- Starting from the Initial Exit the strategy will return to the C loop state when Defect is played against it a sufficient number of times, and there is no shorter path to any other Cooperate state
- Starting from the Initial Exit the strategy will enter the S loop state when Cooperate is played against it a sufficient number of times.

The simulations described in this paper provide for a maximum of 11 states. It is possible that a Basic strategy can be a rarely reached subset of a larger strategy. To allow for this, the statistics report the proportion of strategies playing Basic, weighted by the amount of time the strategy spends in the Basic loop.²

The various features of Basic serve different purposes. The C loop ensures the cooperative payoff when playing another cooperative strategy. The S loop ensures that an even higher payoff is obtained when playing an exploitable strategy. Note that once play has exited the C loop, Basic must pass through the S loop state before it will return to cooperation. Because of the presence of noise, an exploitable strategy will always be detected and exploited. The punishment circuit ensures that “crime doesn’t pay” when playing against Basic: if the length of the circuit is

²Algorithmically, an exit state is any state which can be reached from Exit 1 in fewer steps than it takes to return to the C Loop. The Loop Time is the percentage of plays which the strategy spends in the C Loop or an Exit State. The Basic strategy and all subsets are weighted by the Loop Time.

sufficient, the payoff to a strategy which attempts to exploit Basic’s cooperative state is less than that to a cooperative strategy. These conditions can be summarized in a few aphorisms:

- Never give a sucker an even break
- Respect (cooperate with) an opponent that stands up for itself
- Punish defection
- Forgive a defector who has paid for its transgression

Additionally, the C loop almost always exits directly to the S loop. This allows Basic to recover from a noise event when playing against itself. If noise causes an apparent defection by one player, the other player enters the sucker loop state and ‘waits’ there for a play until its defection causes the first player to also enter the sucker loop. The two players then move through the punishment loop together and enter the cooperative loop together. If a player with an S loop later in the punishment circuit receives an inadvertent defection, this coordination fails except under special conditions. The condition that the C loop exit to the S loop was not made part of the algorithmic definition because of a judgment that it is a technical requirement tied to the particular implementation of the RPD which is unlikely to have broader ramifications. However, statistics were collected on the proportion of Basic strategies with this characteristic. It was found that more than 98% of the Basic has this feature. The remaining 2% is probably the result of point mutations operating on strategy which do have this feature. In the discussion which follows I will consider only the form of Basic in which the C loop exits directly to the S loop, but it should be kept in mind that this is not a requirement in the statistics which are reported.

The main variants of the strict Basic are a soft variant, so called because once an opponent passes through the S loop cooperating will not always return it to the S loop, and a shortcut variant, where cooperating at one or more states in the punishment circuit after the S loop returns the strategy directly to the C loop: see Figures Basic(5) Soft and Basic(5) Short Cut for examples.

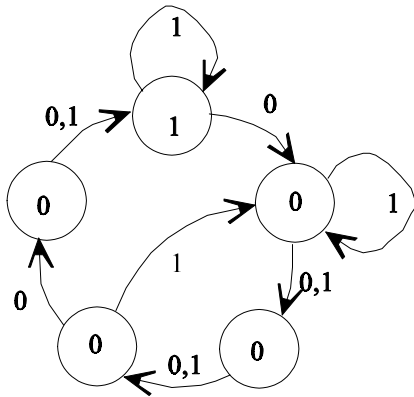


Figure Basic(5) Soft

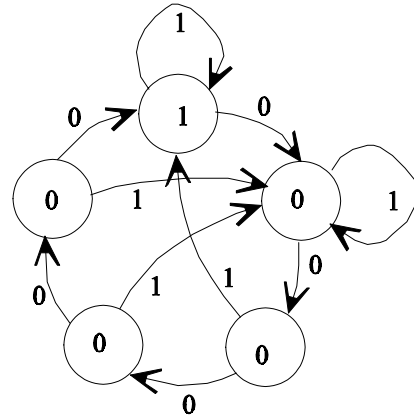


Figure Basic(5) Short Cut

Note that on these definitions the Basic(2) is Win-Stay, Lose-Shift. Grim is equivalent to Basic(∞).

Results: Two States, no noise

Some results involving 2 state systems without noise serve as benchmarks and as tests for the algorithm. Results for the 2 state system with N=1040, uniform distribution of initial strategies over all 26 two state strategies, R=3, T=5, no noise, and no mutation and mutation = 1%, we find the final distribution of states is:

	M=0	M=1%
Dove	0.1%	17%
Grim	67%	50%
Nice Win-Stay, Lose Shift	9.6%	12.5%
Tit-for-Tat	16%	12.5%
5	6.8%	7%

Where Strategy 5 is:

State 1			State 2		
Play	E 0	E 1	Play	E 0	E 1
0	1	2	1	1	1

These results for no mutation are consistent with those of Linster (as reported by Binmore p.202 n.26). With no noise and M=1% Grim initially rises to take over two-thirds of the population, while Dove almost dies out, but as Grim eliminates the nasty strategies and then keeps them in check, Dove recovers.

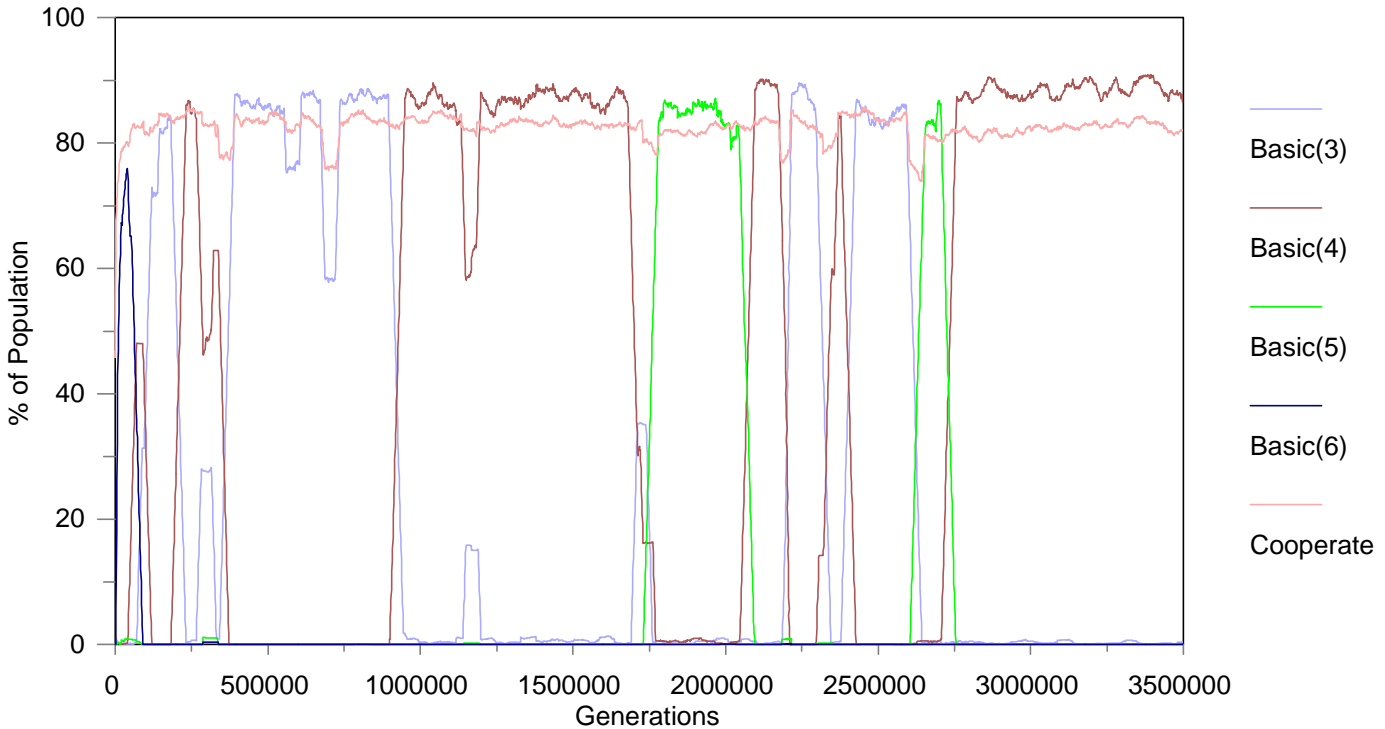
Results: Multiple States, with noise

Typical Run

A portion of a typical run is shown in Figure Typical Run. The Basic strategy dominates the results, with an overall average of just over 84% of the population playing Basic. Basic(3) and

(4) predominate and Basic (5) also makes occasional appearances. Basic(6) appears briefly at the beginning of the run. There are periodic abrupt transitions from one Basic strategy length to another, or to a period of reduced cooperation. The period during which a given length of Basic

Figure: Typical Run
 N40 S11 M5 n1 R2 T2.6 Pr120



dominates is variable.

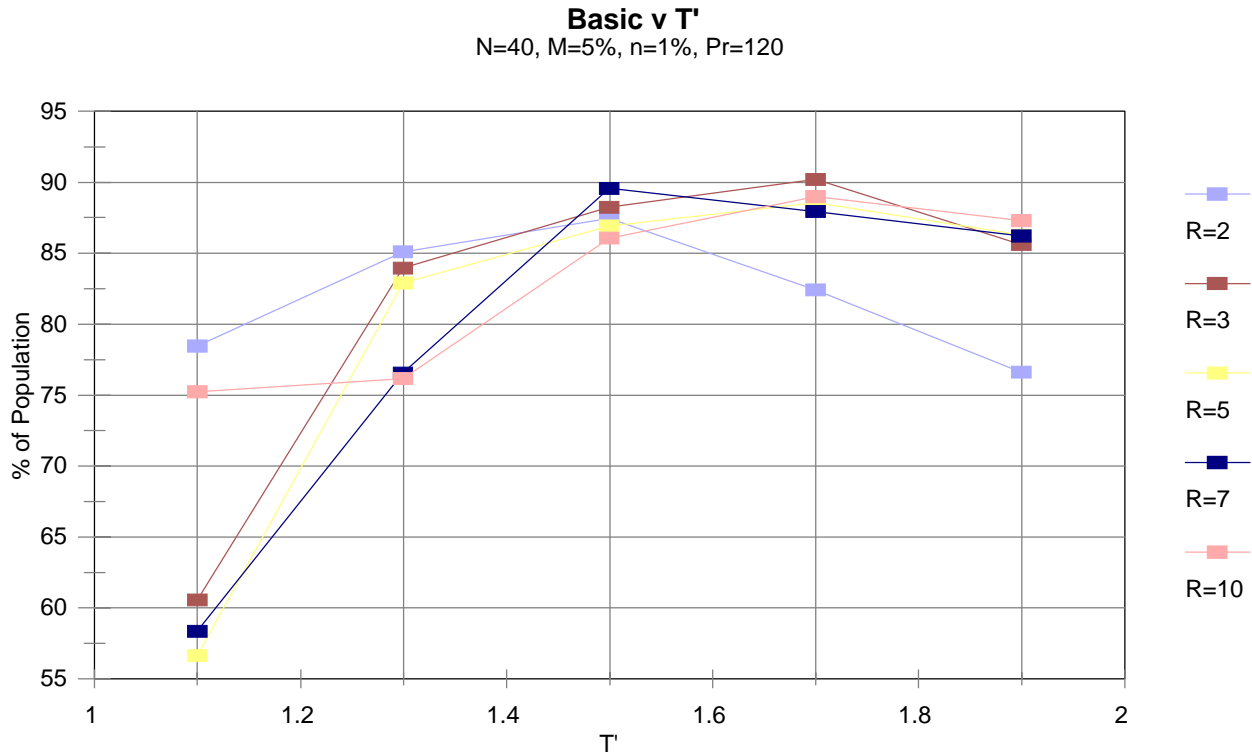
Figure Typical Run: N=40, R=2, T=2.6, M=5%, n=1%, Pr=120 (N40 S11 M5 n1 R20 T26 Pr120f cps 14-27-48). Sampled every 100 generations, smoothed with a moving average over 50 samples. The justification and implications of these specific parameter values are discussed below.

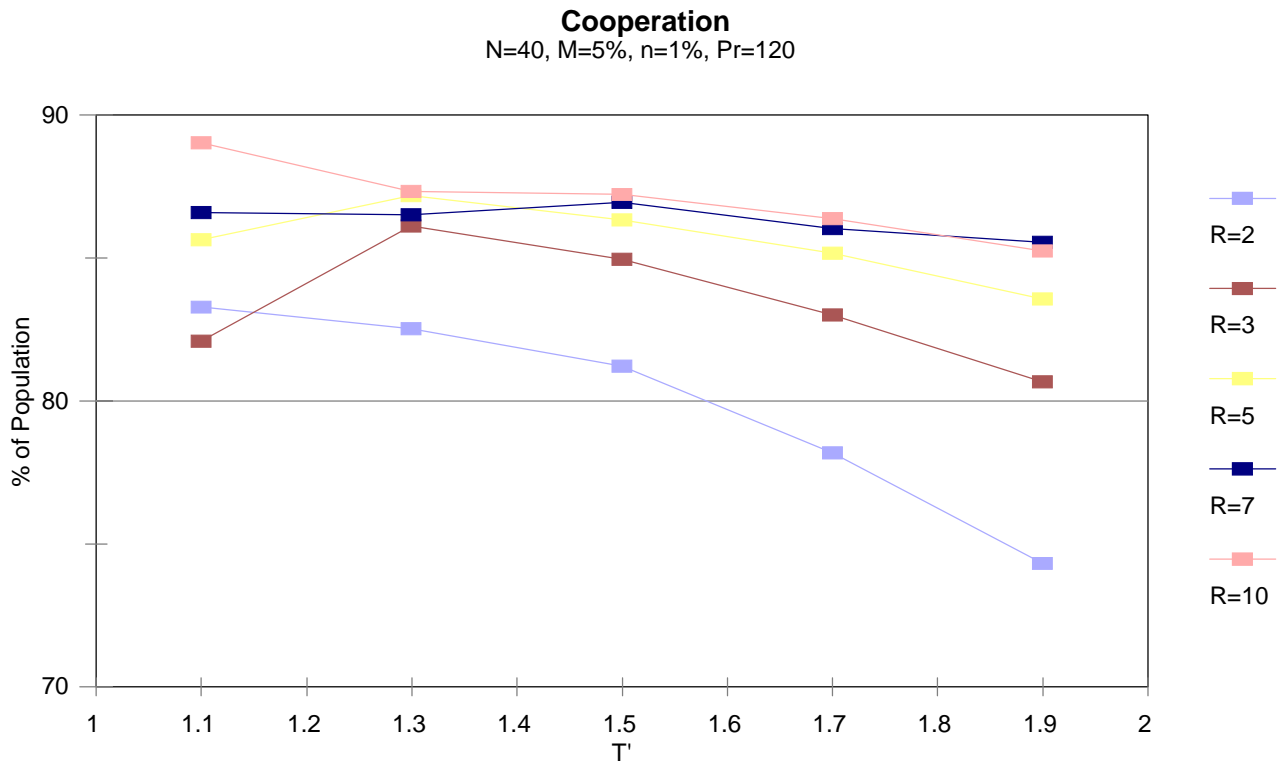
The dependent variables of interest are the proportion of the population playing Basic, η_B , the

distribution of L (the length of Basic), $\eta(L)$, or $\omega(L) \equiv \eta(L)/\eta_B$, and the stability of Basic(L), which will be considered in more detail below. These are functions of a number of variables: payoffs (R, T'), noise rate (n), number of players in the population (N), number of plays per round (Pr) and mutation rate (M).

Effect of Temptation to Defect

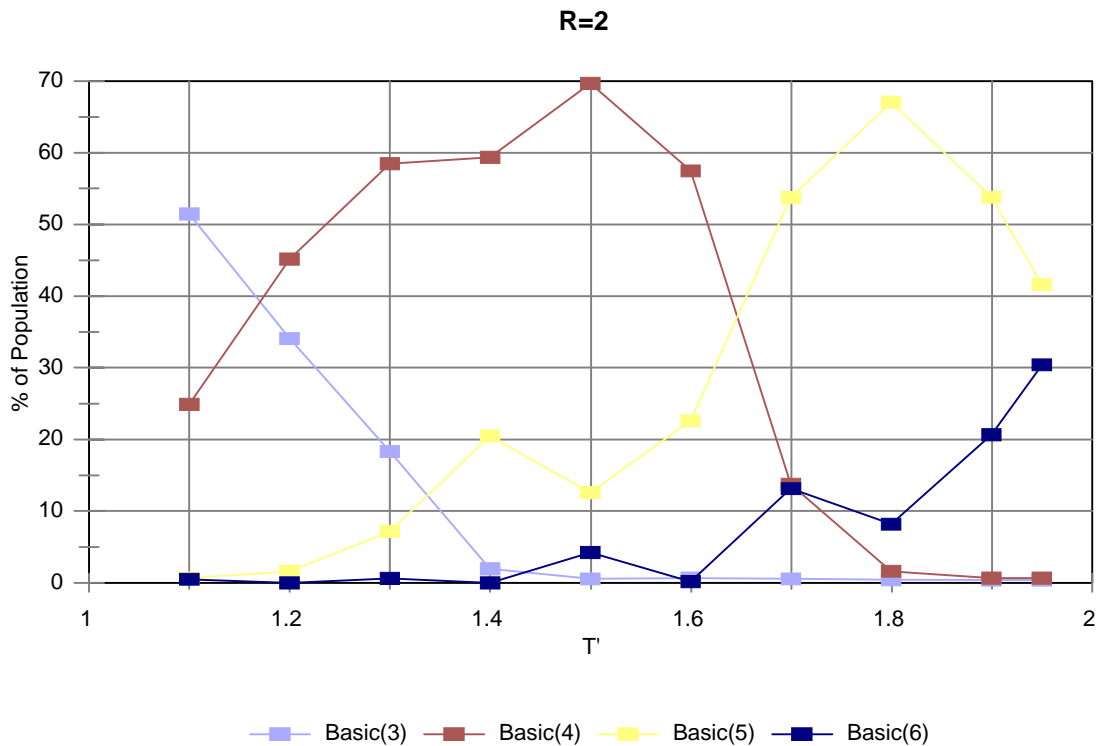
The proportion of the population playing Basic is substantial even at $T' = 1.1$, and it increases to a maximum at mid-ranges of T' , then decreases slightly as T' approaches 2: Figure Basic v T'. The degree of cooperation, η_c , defined as (Number of times 1 is played/Total number of plays), drops slightly as T' increases: Figure Cooperation. This implies that the proportion of cooperation attributable to Basic increases as the temptation to defect increases.



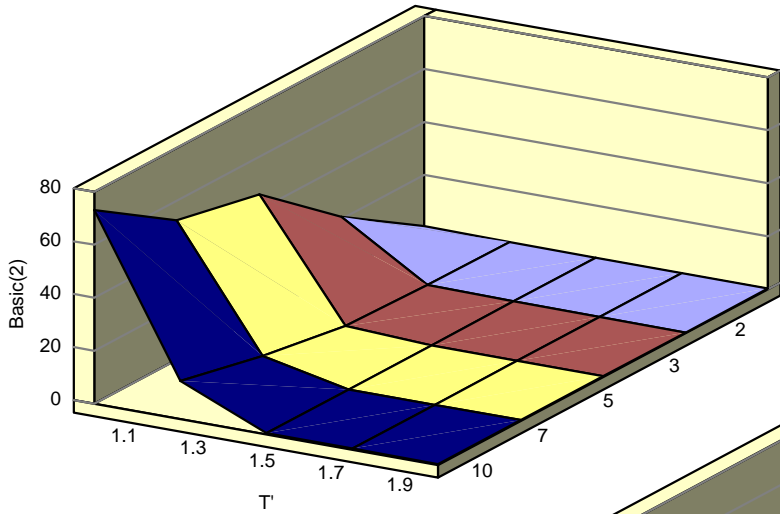


Distribution of Length of Punishment Circuit

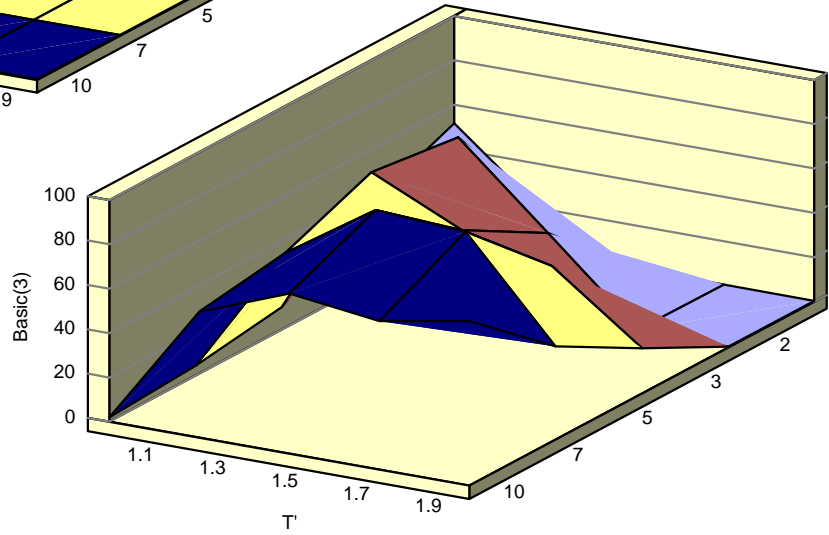
The general trend with respect to the distribution of the length of the punishment circuit (L) is clear in Figures Basic(2)-(6). As the relative advantage to repeated mutual cooperation is reduced, either because T increases, or because R decreases, the length of the punishment circuit increases. Figure R=2 shows this clearly for the case of R=2. The distribution of L can be explained by the need to resistance to invasion by Hawk, which sets a lower bound on the length of punishment, and the need to resist invasion by a more forgiving strategy, which sets an upper bound. This will be discussed in more detail below.



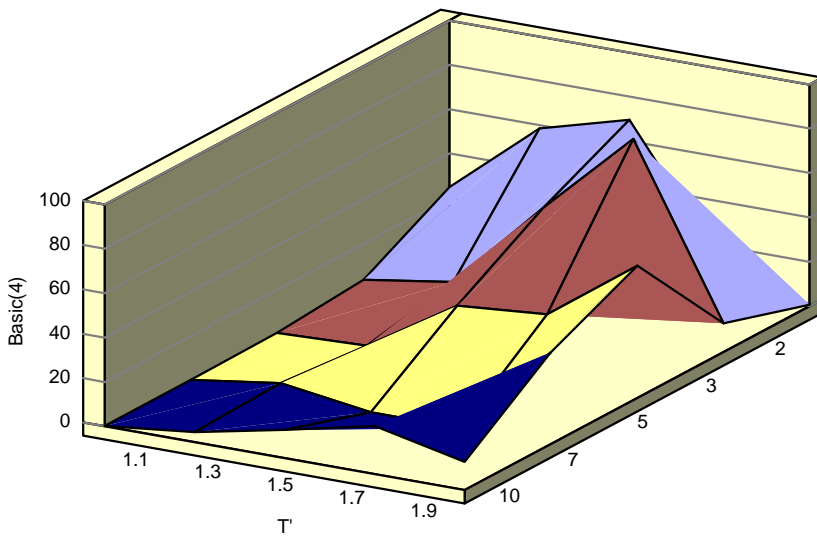
Basic(2)
R= 2 to 10

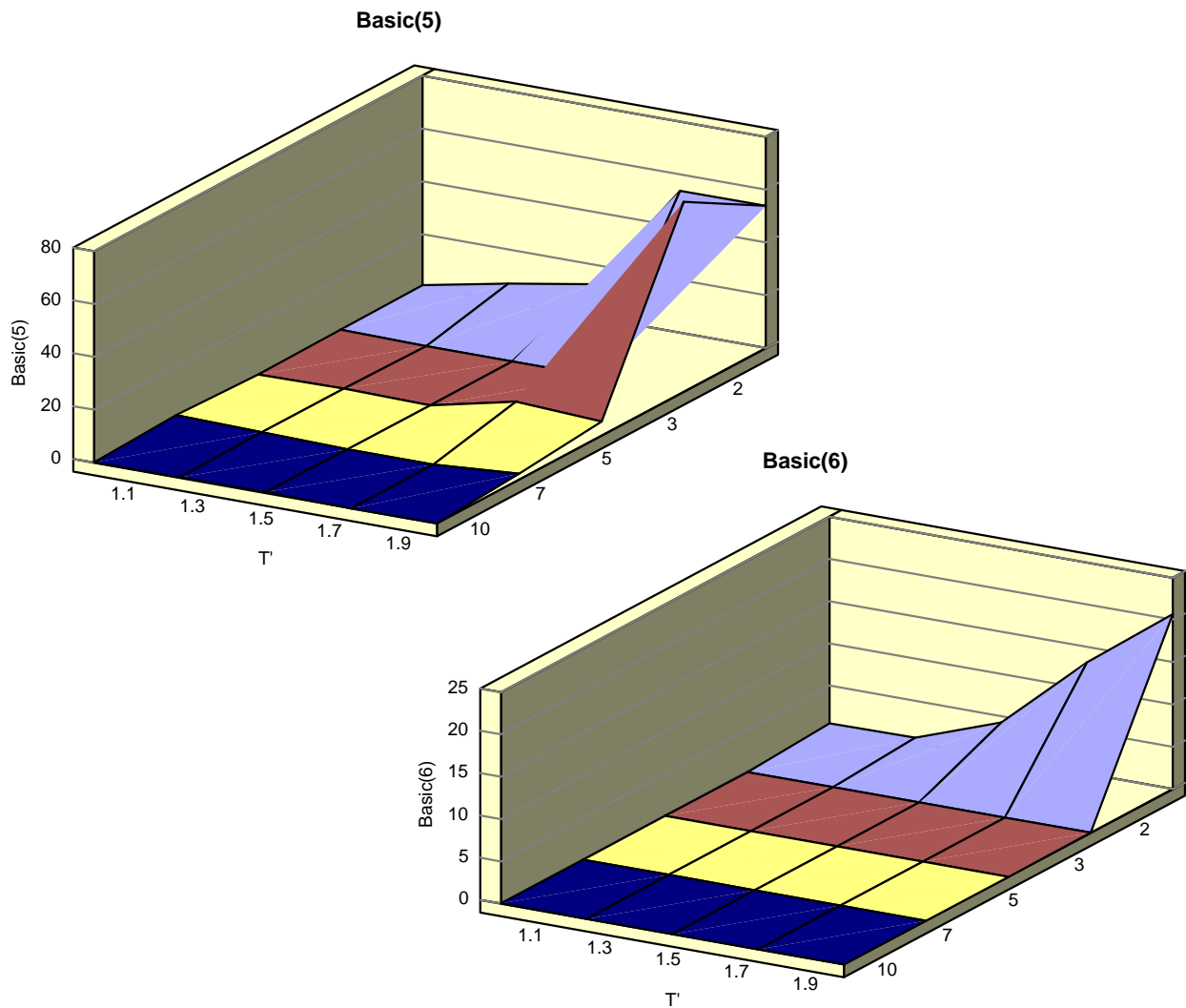


Basic(3)



Basic(4)





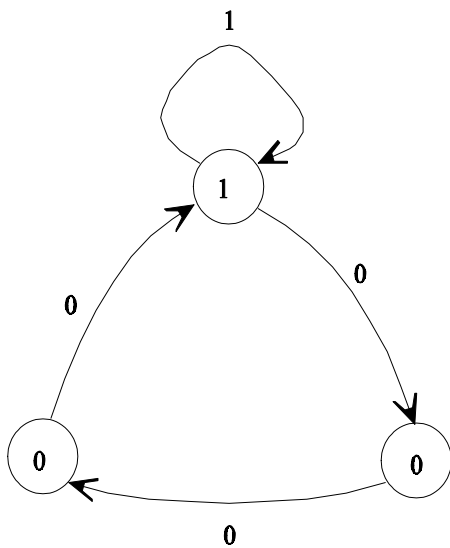
Figures Basic(2)-(6): Statistics were collected for a minimum of 10^6 generations, except for $R=2$ which has a minimum of $3 \cdot 10^6$ generation. This is insufficient to allow entirely stable distributions to emerge, particularly for higher R , as all strategies become more stable as R increases.

Comparison with Previous Simulations

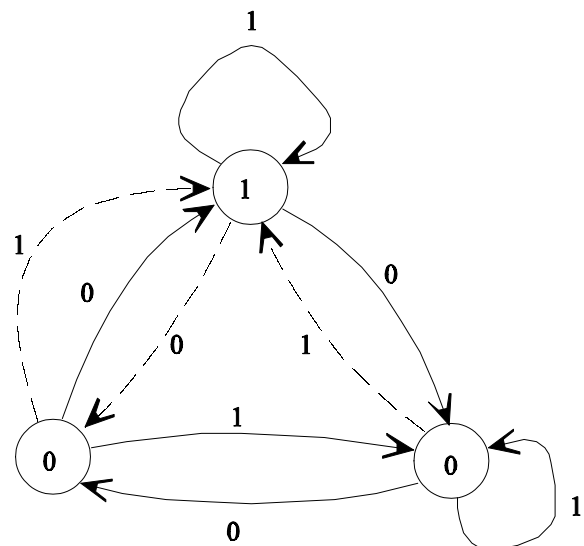
Since Basic(2) is the same as Win-Stay, Lose-Shift, the dominance of Basic is consistent with the findings of Nowak & Sigmund (1993) that WSLS dominates among memory-two strategies with noise. In fact, if we set the simulation is so that the maximum number of states is 2 and use Nowak & Sigmund's payoff parameters and 1% noise, we find that for the appropriate settings of the other parameters (mutation, and plays/round) Basic(2) dominates the population. The mutation and plays/round parameters must be set to produce fairly high stability (see Effect of Independent Variables, below).

Interestingly, when the maximum number of states is increased to 3 and all parameters are held constant, Basic(3) dominates, at least initially. However, the parameter values needed to produce stability for Basic(2) at the payoff values used by Nowak & Sigmund produce very high stability in Basic(3) and it is not clear whether Basic(2) or Basic(3) would dominate in the long run.

The dominance of Basic is also consistent with the Lindgren’s (1991) results for memory four, that is, $S=3$. Lindgren’s generic strategy [1xx10xxx0xxxx001] which he identifies as the dominant class of strategies, and his strategy [1001000100010001] which is the leading strategy in his simulation, are shown, translated into an explicit Markov representation, in Figure Lindgren. (See Appendix 2 for details of the translation.) His generic strategy is a form of Basic without the Sucker exploitation loop, and his leading strategy is essentially Basic(3). The strategies are not strictly equivalent because, as noted earlier, in my simulation a noise event does not cause a state transition in the strategy whose play is erroneous, while it does in Lindgren’s. While Lindgren’s leading strategy does exploit suckers, he does not emphasize this point. He



Lindgren Strategy (1xx10xxx0xxxx001)



Lindgren Strategy (1001000100010001): Dashed lines indicates a path which is a change of state caused by erroneous play by the strategy.. As discussed in the text, this is not a possible transition the simulation, discussed in this paper.

focuses on the generic strategy, which he notes is sufficiently punishing to avoid exploitation. This is true given Axelrod’s original parameters, which Lindgren uses, although it is not true for the entire range of prisoner’s dilemma parameters. (Note that Lindgren (1991) uses “memory one” to refer to a strategy is conditional on the previous play of the opponent only, “memory

two” is conditional on previous play of both players, and so on, while Nowak & Sigmund use “memory one” to refer to a strategy which is conditional on the previous *round* of play, ie the previous play of both players. In this simulation S=2 is equivalent to memory one in the sense of Nowak & Sigmund, S=3 is equivalent to memory three in their usage, and so one.)

The strategies discovered by Axelrod (1987) in his simulation of the repeated prisoner’s dilemma using a genetic algorithm with memory three (S=3) appear to be broadly similar to Basic. While Axelrod did not report the details of the dominant strategies, he found the following ‘behavioural alleles’:

Axelrod’s Description		Consistency with Basic
Verbal	Strategy	
Don’t rock the boat	C after RRR	Consistent
Be provocable	D after RRS	Consistent
Accept an apology	C after TSR	Consistent only when Basic(2) dominates
Forget	C after SRR	Consistent
Accept a rut	D after PPP	Consistent given that S=3

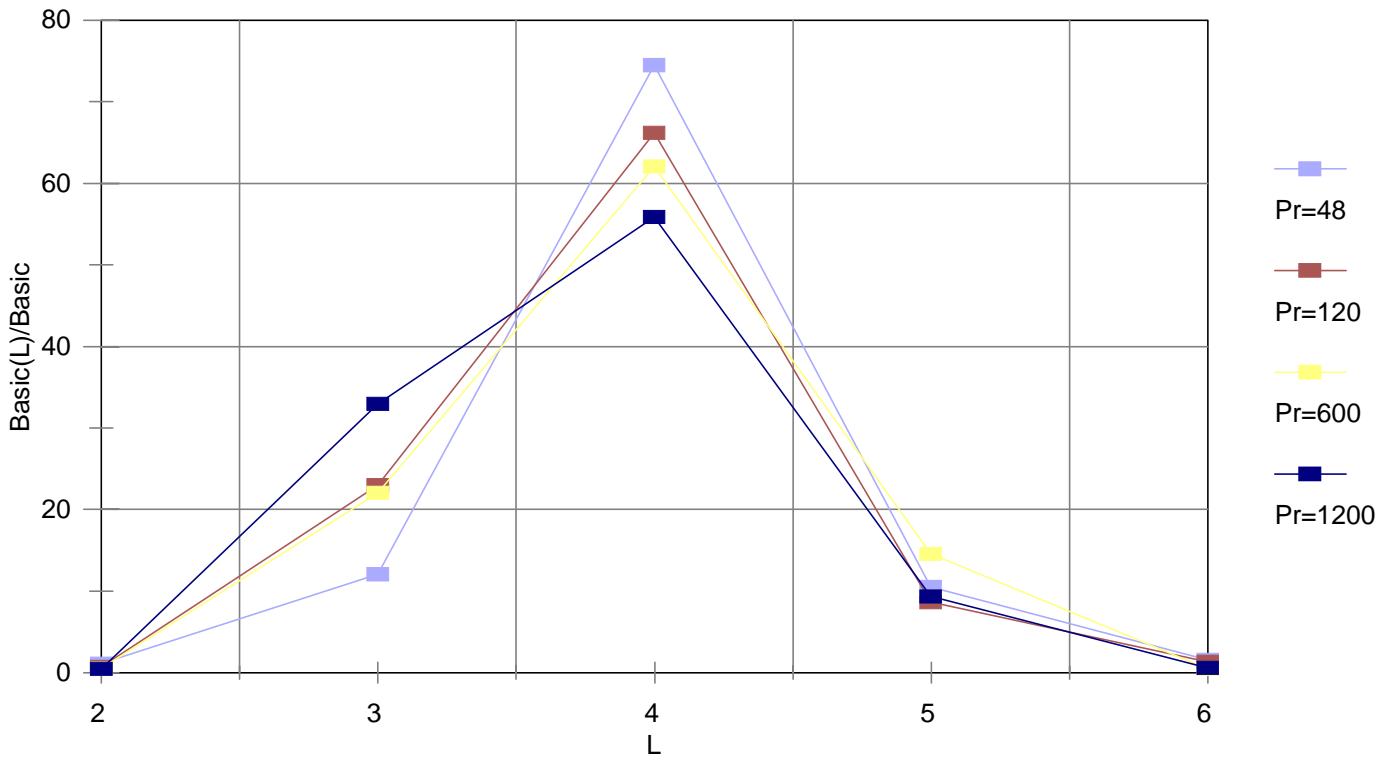
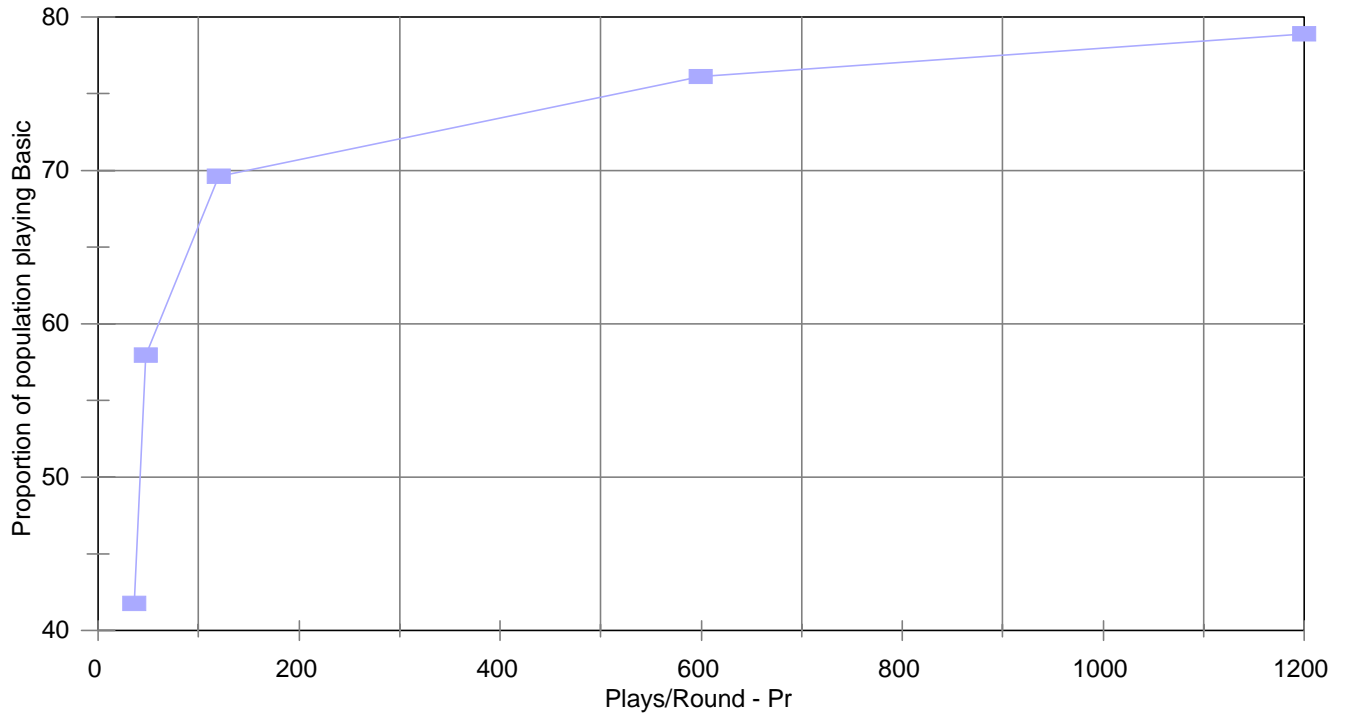
Effect of Independent Variables

Limiting cases of large numbers of players, an infinite number of interactions per round and low mutation rates, are of mathematical interest, but it is not clear that most cases in the real world would be best modelled by the limiting parameter values. At the same time, as a practical matter, approaching these limits dramatically increases the required computer time for the simulation. For this reason the parameters used were not good approximations to the limiting cases. The effect of varying the independent parameters in the simulation was tested by varying these parameters for one particular set of payoff parameters, namely $R=2$, $T=2.6$ ($T' = 1.3$). These payoff values were chosen because it gives rise to significant proportions of three different strategy lengths under most conditions, so that variations in the dominance of various strategy lengths are evident.

Variation of Plays per Round

Increasing the number of plays per round between two individuals (P_r) increases the proportion of the population playing Basic, and increases the proportion of Basic(3) at the expense of Basic(4): Figures Basic v Plays/Round and Variation of Plays/Round. This is likely because as P_r increases the distribution of the realized number of noise events in a given run converges on the expected value, reducing the variance in realized scores. Since the response to noise differentiates Basic from a number of invading strategies, with a high variance it is more likely that there will be a period in which there is insufficient noise for Basic to resist invasion. This will affect shorter lengths of Basic disproportionately because they maintain a smaller score differential with invading strategies.

Basic v Plays/Round

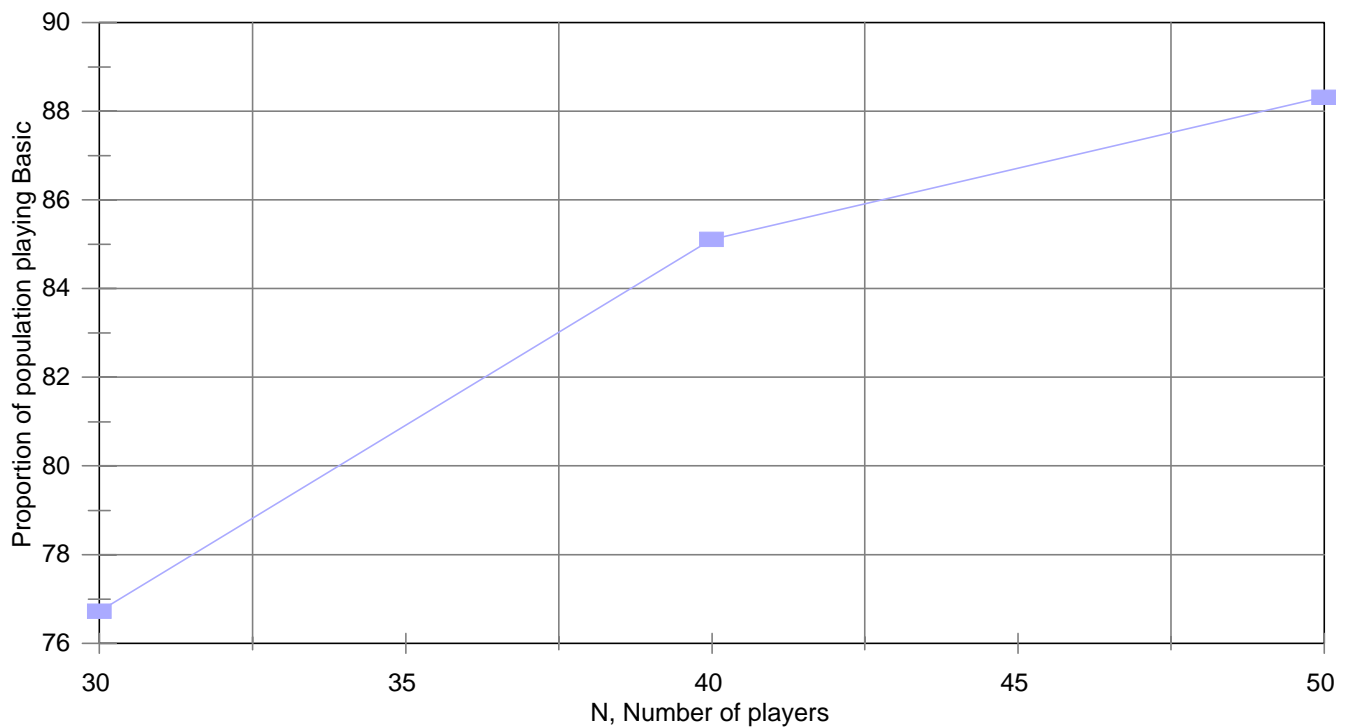


Variation of Number of Players

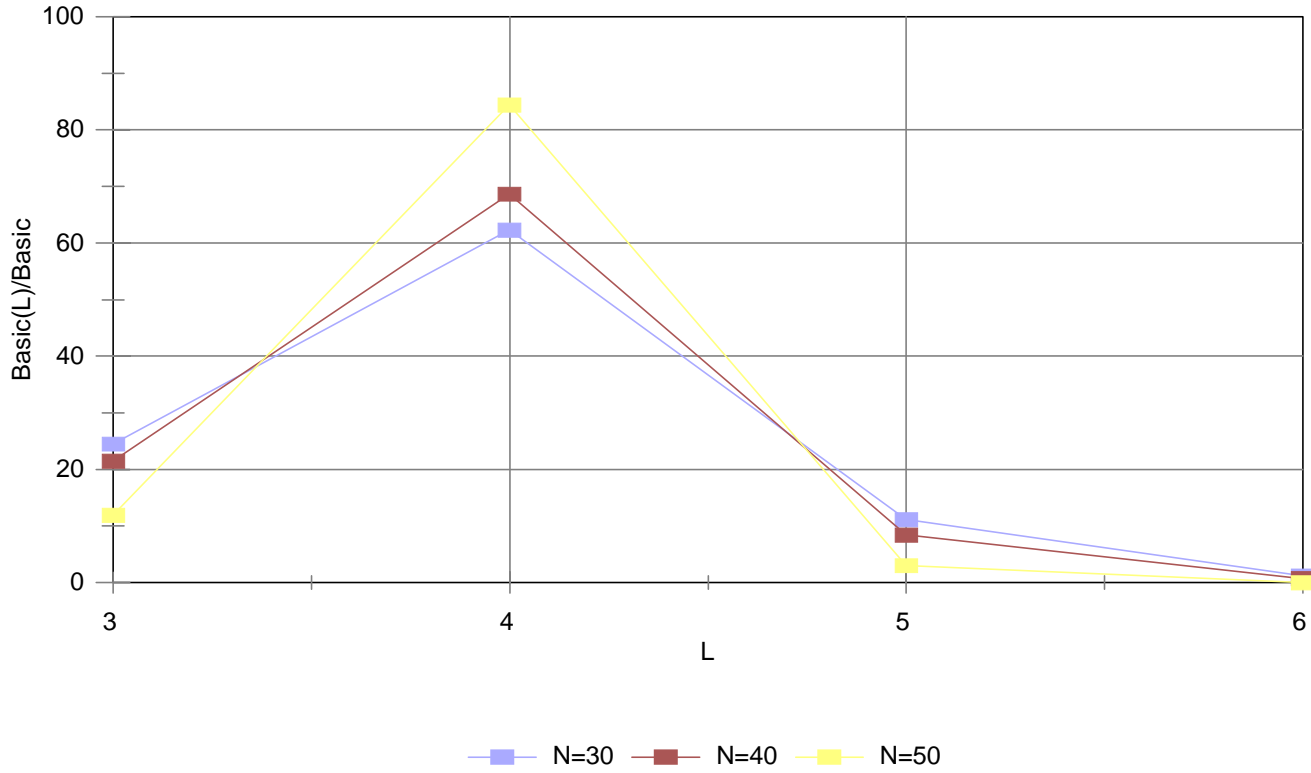
Increasing the N, the number of players, clearly increases the dominance of Basic, presumably because invasion by drift becomes more difficult: see Figure Basic v Number of Players.

Increasing N also appears to flatten the distribution, increasing the proportion of the shorter and longer forms at the expense of the dominant form: see Figure Distribution of L as N Varies.

Basic v Number of Players



Distribution of L as N Varies



Variation of Mutation Rate

Data not available.

Variation of Noise

Noise, n , was varied from 0.01% to 5%. We see from Figure Noise 1 that the proportion of the population playing Basic increases smoothly to a noise level of about 1.4%, then declines as cooperation declines and Hawk increases. (The Cooperation statistic is the number of Cooperate plays as a percentage of all plays.) This indicates that Basic is an effective way of maintaining cooperation in the presence of noise. Basic is less dominant at lower noise levels because maintaining cooperation is generally easier.

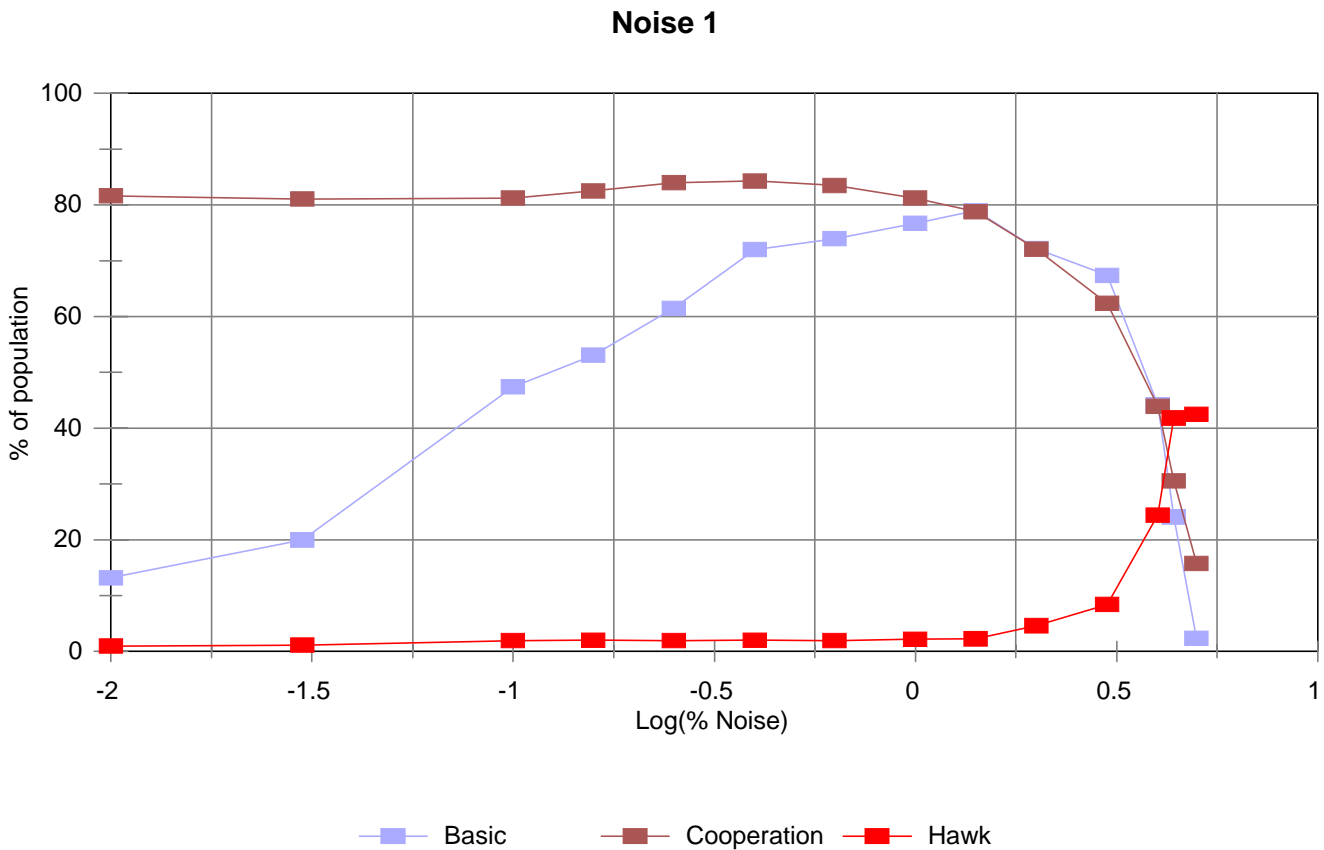
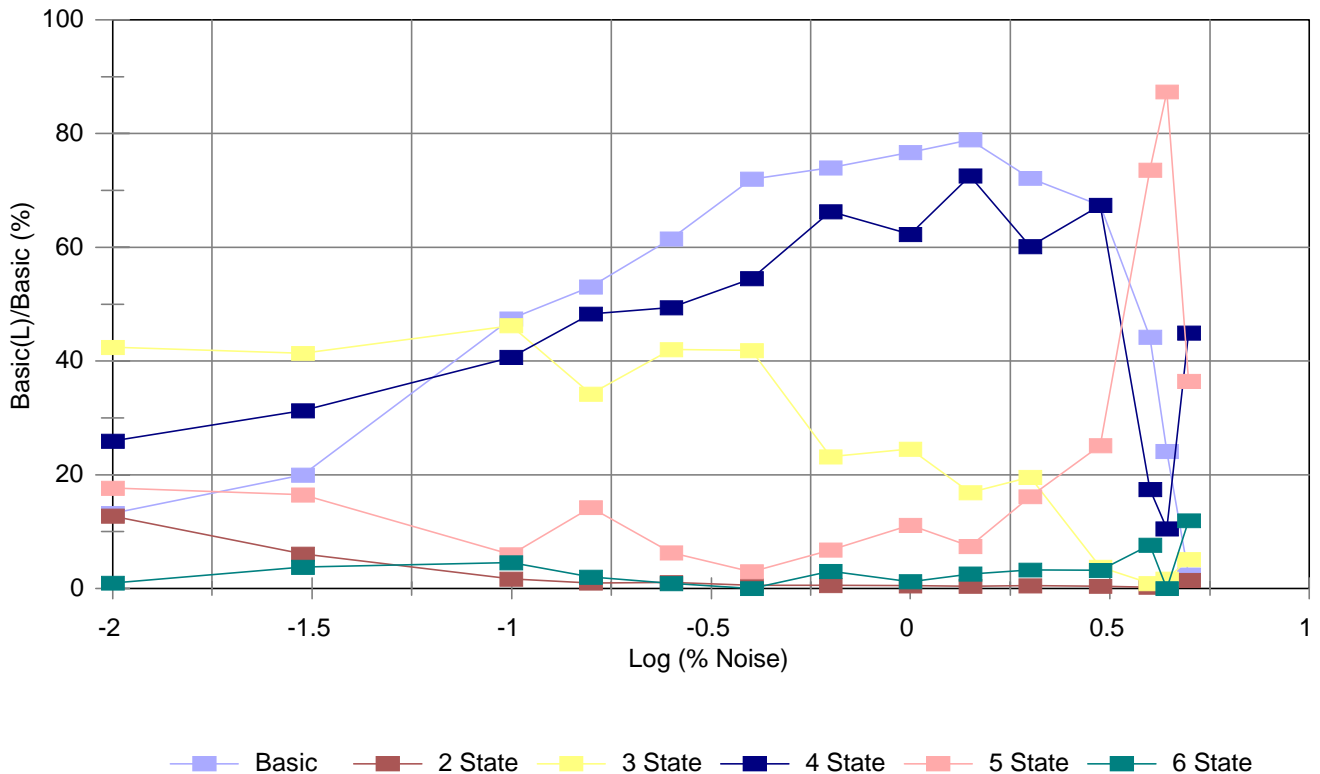


Figure Noise 2 shows how the distribution of the length of Basic changes as the noise varies. (The data point at the 5% noise level is not entirely reliable at this time as the amount of time spent playing Basic is very low.) We might expect that at high noise levels shorter lengths of Basic would be favoured since speed of recovery from noise becomes more important. Instead, relatively long punishment circuits become favoured. Note that Basic(5) begins to increase its share of Basic from about $n=1.4\%$ ($\text{Log}(N) = 0.47$), just as cooperation begins to decline. As cooperation becomes generally more difficult to sustain, uncooperative strategies will do relatively better and so can persist more easily. Shorter lengths of Basic are less resistant to invasion by uncooperative strategies (i.e. the basin of attraction of $\text{Basic}(L)$ with respect to Hawk becomes smaller as L decreases). This suggests that while shorter lengths of Basic are favoured

Noise 2



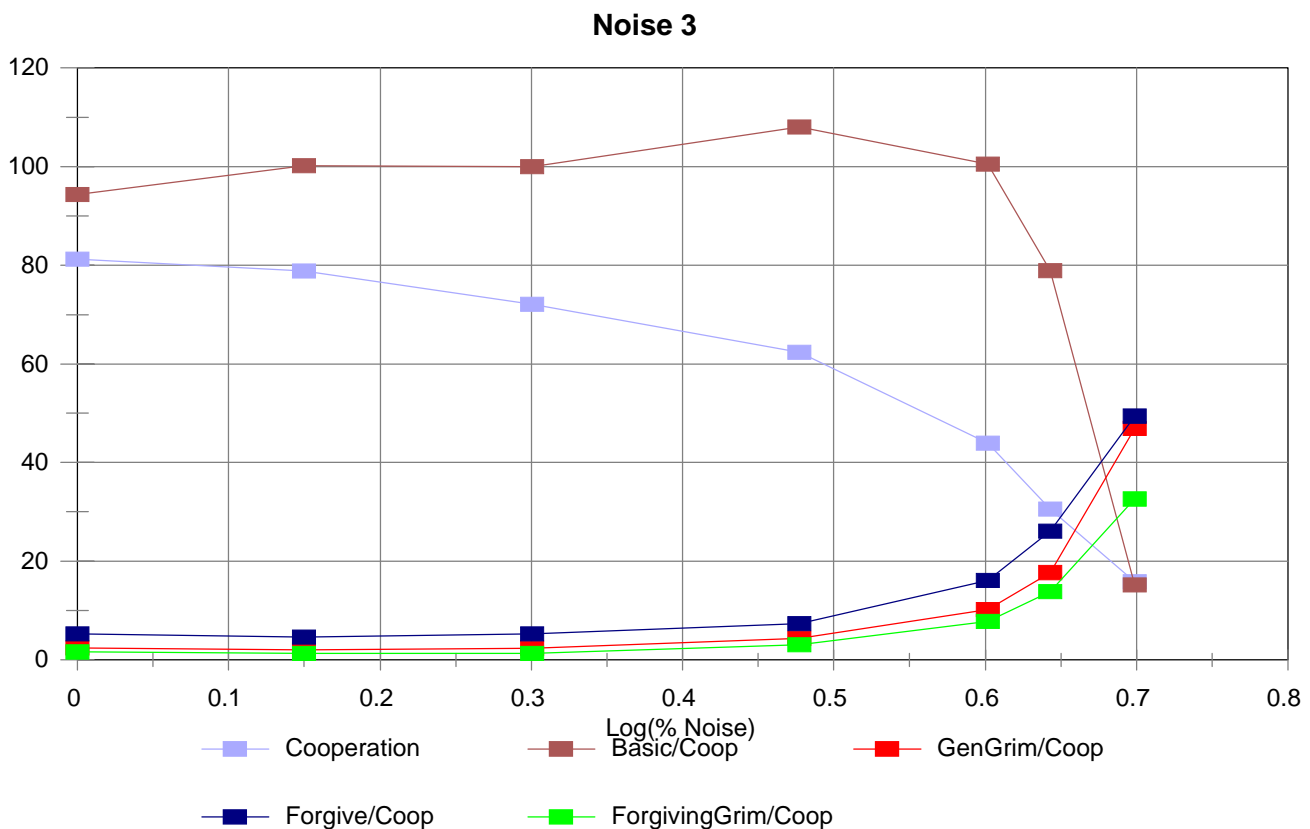
as noise increases because they recover more quickly from noise, their low resistance to invasion by uncooperative strategies becomes increasingly important, and this latter effect dominates.

Also somewhat counter-intuitively, as noise decreases shorter lengths of Basic gain, rather than losing, at the expense of longer lengths. This is almost certainly because strategies generally become less constrained. In the noise range where Basic dominates, 0.4% to 4%, there is an average of approximately seven different strategies in the population at any given time. The average number of strategies increases to almost twelve at the lowest noise levels. This indicates that as the disadvantage to slow recovery decreases, more variation and more circuitous recovery, as well as longer punishment circuits, are also tolerated. Longer lengths of Basic are disproportionately reduced because there are more loose variants on a longer circuit.

Figure Noise 3 shows the strategies which take an increasing share of the (declining) cooperation at higher noise levels. A Generalized Grim strategy is one in which there is a C loop and a Grim loop (where a Grim loop is a state which plays Defect and exits to itself on both Defect and Cooperate by the other player), and, starting from the C loop, playing Defect a sufficient number of times in a row will lead to the Grim loop. The statistic is weighted by the amount of time spent in the C and Grim loops and states reached in between when Defect is played. A Forgiving strategy is one which has a C loop and which does not immediately retaliate with Defect when it is in the C loop and Defect is played against it. That is, when it is in the C loop and Defect is played against it, its next play is Cooperate. A Forgiving Grim strategy is one which has both these features: it does not retaliate against a defection immediately, but sufficient defection against it will cause it to enter a Grim loop.

We see that at very high noise levels Forgiving Grim accounts for an increased proportion of cooperative play. This presumably reflect the premium on speedy recovery from noise. Forgiving Grim can recover quickly from noise when playing itself, but it cannot be exploited by Hawk. Despite these advantages, Forgiving Grim is not a robust strategy as it can be exploited

by a strategy which alternates Defect and Cooperate. Not surprisingly, the incidence of this Exploitation of Forgiveness strategy increases as the incidence of forgiving strategies increases (not shown). It appears that Forgiving strategies are not generally robust because of their susceptibility to this type of exploitation, but at high noise levels the relative advantage of quick recovery from noise outweighs the exploitability. However, the overall level of cooperation is low when this happens.



At low noise levels (not shown) Generalized Grim increases. This is to be expected, given a finite number of plays per round, since noise is less likely to trigger unrelenting defection. This is consistent with the observation that Grim dominates noiseless play.

Variation of Maximum Number of States, S

Some runs were performed with $S=21$ with qualitatively similar results. $S=11$ was chosen because the maximum length of Basic with more than a 1% share of the population is Basic(6). $S=11$ minimizes the state space while still allowing plenty of room for strategies which are long in comparison to the dominant Basic to evolve. Note the apparent independence of results on the number of allowed states is inconsistent with the suggestion that the “asocial interaction” modelled by the RPD favours the evolution of complex strategies, as argued by Seth (1997). Seth ran a simulated RPD using the standard vector representation of strategies and using a genetic algorithm with operators which allow strategies to increase in the length of their genome. Increasing the length of the genome corresponds to an increase in the memory of previous plays. Using genome length as an admittedly crude measure of complexity, Seth showed that “more complex strategies (memory 6) were more robust and led to greater overall levels of cooperation than simple strategies (of memory 1).” Seth did not report the actual strategies which dominated. The current simulation indicates that the dominant strategy for the payoff parameters used by Seth (Axelrod’s original parameters) would be Basic(4). Thus it is no surprise that strategies with memory 3 or longer will do better than strategies with memory 1. But no length of Basic is more ‘complex’ than any other, since each can be minimally described as having a Cooperative loop, a Sucker exploitation loop and a punishment circuit of length $(L-1)$. Seth’s conclusions regarding complexity are questionable because memory length is not a good proxy for complexity in this context.

Distribution of Length and Stability of Basic

Noise Conditional Stability

Why Basic is so successful? An obvious question is whether it is evolutionarily stable. Before turning to that question I would note that the traditional notion of evolutionary stability as an uninvadable strategy is not entirely appropriate in a complex environment. Despite this caveat, which will be discussed in more detail below, evolutionary stability remains a useful heuristic. It is known that no strategy is evolutionarily stable in a noiseless environment (Boyd and Richerson (1987)) and strict evolutionary stability is not possible in a noisy environment since the stochastic nature of the noise means at some point a period of play will occur which is noise-free.

However, noiseless period will be very rare if the noise level is high enough, and the standard definition of an evolutionarily stable strategy can usefully be extended to a noisy environment. We can define noise-conditional evolutionary stability by saying that a strategy is noise conditionally stable if the standard conditions for evolutionary stability are satisfied in a noisy game, for some sufficiently small noise level n , $0 < n < \varepsilon$.

If the upper limit on the noise, ε is sufficiently small, noise events will be isolated in the sense that the interaction pattern will almost always return to stability after one noise event before the next noise event occurs. Further, if ε is sufficiently small, the expected payoff will be dominated by payoffs in the noise free period between noise events. Note that the stable noise-free payoff of interest is that which arises after at least one noise event. Grim, for example, initially does well by cooperating with the nice version of Basic, but after a single noise event on either side, a stable pattern in which Grim defects and Basic cycles through cooperation and punishment emerges. When there are many plays per round, the initial period of cooperation can be ignored. Under these conditions, to a good approximation the expected payoff in an interaction between two strategies σ_A, σ_B is:

$$\pi(\sigma_A, \sigma_B) = (1-An)\pi'(\sigma_A, \sigma_B) + An \pi''(\sigma_A, \sigma_B)$$

where $\pi'(\sigma_A, \sigma_B)$ is the average payoff in the absence of noise (but subsequent to a noise event), $\pi''(\sigma_A, \sigma_B)$ is the average payoff during a noise event, that is, the average payoff from and including the noisy play, to the time that a stable play pattern is re-established, and A is the duration of the out of equilibrium play after a noise event.

Definition 1.1 We can then say that σ^* is noise conditionally stable if, for all $\sigma \neq \sigma^*$:

$$\pi'(\sigma^*, \sigma^*) \geq \pi'(\sigma, \sigma^*) \quad (1.1a)$$

$$\pi'(\sigma^*, \sigma^*) = \pi'(\sigma, \sigma^*) \Rightarrow \pi''(\sigma^*, \sigma^*) \geq \pi''(\sigma, \sigma^*) \quad (1.1b)$$

$$\pi'(\sigma^*, \sigma) = \pi'(\sigma, \sigma), \pi''(\sigma^*, \sigma^*) = \pi''(\sigma, \sigma^*) \Rightarrow \pi'(\sigma^*, \sigma) \geq \pi'(\sigma, \sigma) \quad (1.2a)$$

$$\pi'(\sigma^*, \sigma) = \pi'(\sigma, \sigma), \pi''(\sigma^*, \sigma^*) = \pi''(\sigma, \sigma^*), \pi'(\sigma^*, \sigma) = \pi'(\sigma, \sigma) \Rightarrow \pi''(\sigma^*, \sigma) > \pi''(\sigma, \sigma) \quad (1.2b)$$

Conditions 1.1 is the Nash equilibrium condition, in two parts: σ^* must do better than any invader during noiseless (but post-noise) play, or it must do equally well in noiseless play and recover better from noise when playing itself than the invader does when playing σ^* . If the invader does equally well under both these conditions, then we turn to 1.2, the stability conditions. More generally, we can say that a set of strategies, A, is noise-conditionally stable if strict equalities hold between any pair of strategies $\sigma_i, \sigma_j \in A$ and definition 1.1 is satisfied for any $\sigma^* \in A$ and any $\sigma \notin A$.

Lower Bound on L: Crime Doesn't Pay

For some parameter values Basic can be directly invaded by Hawk. The expected payoff in an interaction with Hawk is dominated by the noise free payoff. The relevant payoffs are:

$$\pi'(B(L), B(L)) = R \quad (2.1)$$

$$\pi'(H, B(L)) = (T+L-1)/L \quad (2.2)$$

from which it follows that the minimum L for Basic(L) to be stable invasion by Hawk is:

$$L > (T-1)/(R-1) \quad (2.3)$$

This condition, which we will call “crime doesn't pay”, sets a lower bound on L. Alternatively,

it can be used to specify a minimum value of R for a given L . In particular, for $L=2$, the condition is:

$$R > (T+1)/2 \quad (2.3a)$$

Note that when this condition is an equality, Basic(L) can be invaded by Hawk because conditions 1.1a and 1.1b are satisfied and Basic(L) fails the stability criterion of 1.2a (Hawk does better against itself than Basic(L) does against it).

If this condition is not met Basic is not stable because condition 1.1a fails for many strategies, notably Hawk. Because of the high mutation rate and small population, the minimum length of Basic which is actually observed in these simulations is longer than this strict minimum. It is therefore also useful to know the basin of attraction of Basic(L) with respect to Hawk.

The proportion of Basic(L) in the population, p_e , at which Hawk and Basic(L) do equally well is given by (see Appendix 1):

$$p_e = 1/\{L \pi(B(L), B(L)) - T - L - 2\} \quad (3.1)$$

$$\text{where } \pi(B(L), B(L)) = R - 2n[R(L+1) - (T+L-1)] \quad (3.1a)$$

The ability of Basic(L) to resist invasion by Hawk increases as p_e decreases, as this implies that a larger number of Hawk must be introduced by mutation. As is intuitively obvious, increasing T' decreases the resistance to invasion, as Hawk gains additional advantage from exploiting Basic's periodic cooperation. Equally obviously, reducing L reduces the resistance to invasion. Finally, increasing R , with T' constant, increases resistance to invasion because of the increased reward to cooperation over mutual defection. This is why, even though $\pi(\text{Basic}(2), \text{Basic}(3)) > \pi(\text{H}, \text{Basic}(2))$ for $T'=1.1$ at both $R=2$ and $R=10$, we only actually observe Basic(2) at the higher value of R . When $R=2$, $p_e = 0.59$ but when $R=10$, $p_e = 0.121$.

Upper Limit on L

If the condition that crime doesn't pay is satisfied, any strategy which intentionally defects will do worse than Basic in noise-free play. That is to say, inequality 1.1a will always be satisfied. In

this case, the main way in which Basic can be invaded is by a strategy which is more forgiving, i.e. which destabilizes Basic because inequality 1.1b is not satisfied. But Basic’s exploitation of suckers makes this route difficult as well.

$L=2$

First consider the case of Basic(2). When $S=2$ and the payoffs are such that conditions 2.3a (‘crime doesn’t pay’) is satisfied, we can see by inspection of the 26 possible strategies that Basic(2)/ Win-Stay, Lose-Shift (WSLS), is noise conditionally stable. This is consistent with the dominance of WSLS found by Nowak and Sigmund. While they do not make the point explicitly, in their simulation using a Markov process, the probability of transition from the defect state to the cooperate state after the opponent defects must be low enough that Hawk does worse than probabilistic Win-Stay, Lose-Shift: see Nowak & May 1993, esp Figure 3. (Interestingly, strategy 5, which we saw is one of the 5 surviving strategies in with no noise and $S=2$, does best against Basic(2): it recovers as well from noise when playing WSLS as does WSLS when WSLS receives the false noise event, but it recovers more slowly when 5 receives the false noise.)

Note that using Axelrod’s parameters, $S=0$, $P=1$, $R=3$, $T=5$, which were also used by Lindgren, the condition that crime doesn’t pay condition is an exact equality for $L=2$. We should be careful about generalizing from this to behaviour under parameters far from this equality point. For example, Lindgren (1991) says (p.305) that with memory two (in his sense, ie $S=2$), Grim can invade WSLS, which is true with his payoffs, but not when condition 2.3a is satisfied.

We can also provide a heuristic argument that Basic(2) is noise conditionally stable even when $S>2$, provided that inequality 2.3 is satisfied.

Since defection doesn’t pay, any strategy which intentionally defects will do worse than Basic in noise-free play. To do as well as Basic, a strategy must cooperate at least until a noise event,

which is to say it must have a C loop. An invader with a C loop will do as Basic in noise free play provided it can coordinate so that both strategies are in their C loop at the same time. To be successful, the invader must recover from noise as well or better than Basic does when playing Basic. When Basic(2) plays itself, a noise defection by one player results in to the following sequence.

A	B
1	1
1	0 ⁿ
0	1
0	0
1	1

An invader cannot do better than this when Basic(2) receives a noise defection (ie Basic(2) is player A). Basic(2) will defect on the next round and defect until the invader defects as well, before switching back to cooperation. If the invader delays defection, it will delay the resumption of cooperation and so forgo a payoff of R during the delay. An invader can do better than Basic(2) when the invader receives a noise defection by simply forgiving the defection. Cooperation will resume immediately, and the invader receives a payoff of $(S+2R)=2R$ where Basic(2) would receive $(S+T+P)=(T+1)$. By the condition that crime doesn't pay, $2R > T+1$, so the invader does better than Basic in this case. But if the invader is forgiving, so improving its payoff against Basic(2) when it receives a noise event, it will worsen its payoff when Basic(2) receives the noise event. In the latter case, the invader will forgive Basic(2)'s first defection, which means that it takes longer to return to mutual cooperation. The invader will then do worse than Basic(2) by $(R-S) = R$. The net effect is that the invader will only do better than Basic(2) if $2R+S > (T+1)+R \Rightarrow R > T+1$, which is impossible since $T > R$.

Nor can a stochastic Markov process do better.**

$L > 2$

When S is greater than 2, Basic can remain stable against Hawk and other intentionally defecting strategies, even at low R , by increasing L to maintain inequality 2.3. But increasing L increases susceptibility to invasion by a more forgiving strategy. This is obvious in the case of Grim, which is equivalent to $\text{Basic}(\infty)$, which does poorly because a single noise event triggers indefinite defection.

Strategies which are strictly able to invade Basic, that is, for which inequality 1.1b is not satisfied, are variants on a small number of themes. Consider the strategy Forgiving Basic(L),

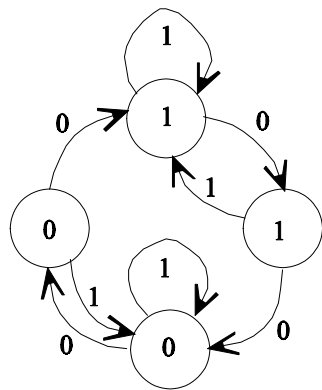


Figure **:Forgiving Basic(3)

adds a state which forgives a single defection between the C loop and the S loop (see Figure Forgiving Basic(3)). (A note on terminology: I will use “Forgiving” to indicate a strategy which, after cooperation has been established, plays Defect after a single defection by the other player, but a “more forgiving” strategy is simply one which retaliates less than the dominant strategy in response to a defection.) This strategy can invade $\text{Basic}(L)$, $L > 2$, provided R is sufficiently large. Invasion is possible if (see

Appendix 1 for derivation) if:

$$RL - R - T T' \rightarrow 2 \Rightarrow R_i > \frac{(L-1)}{(L-3)} \quad R T' \rightarrow 1 \Rightarrow R_i > \frac{(L-1)}{(L-2)} - 1 + 1 > 0$$

Which implies:

$$T' \rightarrow 2 \Rightarrow R_i > \frac{(L-1)}{(L-3)} \qquad T' \rightarrow 1 \Rightarrow R_i > \frac{(L-1)}{(L-2)}$$

A larger R makes invasion easier because that the reward to forgiveness is a quicker return to the reward of R from cooperation rather than P from mutual defection. For the same reason longer forms of Basic (increasing L) are also more susceptible to invasion.³ As T' increases invasion becomes harder because the Forgiving strategy foregoes temptation to defect when a noise event occurs against it and this is offset against the quicker return to cooperation.

Even when R is large enough that Basic is not strictly stable against Forgiving Basic (FB), invasion is nonetheless difficult. Forgiving Basic has only a slight advantage over Basic, so will invade slowly, and it is susceptible to exploitation by a strategy (“Exploit Forgiveness”) which alternates defection and cooperation. This exploitation is particularly effective at higher T', since at lower T' exploitation of forgiveness is not much more successful than cooperation. Since exploitation of forgiveness only requires two states in the exploitation loop, it arises quite commonly by random mutation. The result, particularly at higher T', is that at some point during Forgiving Basic's slow invasion of Basic, Exploit Forgiveness will arise and decimate Forgiving Basic. Even though Forgiving Basic is more effective against longer forms of Basic, it is unable to invade longer forms of Basic because the concomitant higher T' results in more effective exploitation of forgiveness, which keeps the forgiving strategies in check.

In a variant on Forgiving Basic (FB2) the second cooperative state leads to itself rather than back to the first C loop. This strategy is resistant to exploitation by Exploit Forgiveness, but it is liable to be invaded by Forgiving Basic, which can then be exploited. There is plenty of time for this to

³Note that the Forgiving Basic also does better against itself than Basic does against Basic, and even though Basic does better against Forgiving Basic than Basic does against Basic, the net effect is that the advantage to Forgiving Basic increases as its share of the population increases.

happen, because FB2's advantage over Basic is even less than that of FB. Overall though, FB2 appears to be more successful than FB in invading Basic. (Example Details 17-45-01 G1356500**Figure not included)

For $L > 3$ another type of invasion is possible. As noted, the strict form of Basic drifts relatively easily to a shortcut state in which a state in the punishment circuit to the S loop exits to the C loop, rather than returning to S loop, when the opponent plays Cooperate. (Note that while the various forms of Basic receive the same score when playing each other, for sufficiently small n , they do not necessarily receive equivalent scores against other strategies, so small number of other strategies introduced by mutation can affect the dominance of various forms of Basic). A strategy which cooperates at the right point the punishment circuit can thereby shorten the recovery period. This strategy is relatively effective at exploiting longer forms of Basic, particularly when R is high, and it is relatively easy to create through mutation of the dominant Basic, and for these reasons it appears to be a relatively common form of invasion at longer L . But the invasion results in a frequency dependent equilibrium at best, and Exploit Short Cut is liable to exploitation by Exploit Forgiveness, so it is difficult to displace Basic entirely with this strategy.

Typical Invasion Dynamic

So we see that the strategies which are strictly able to invade Basic rarely do so successfully. (The exception is that invasion by Forgiving Basic appears to occur at low T' and low L .) But with the high mutation rate and relatively small population used in these simulations, a strategy does not have to do strictly better than Basic when playing Basic in order to invade. A more forgiving strategy which does reasonably well against the dominant Basic can do better than average even in the early stages, and so get established, by doing better than the weak strategies which arise through mutation. Once sufficiently established, its advantage in playing itself will allow it to take over the population. The strict form of Basic is generally very difficult to invade in this manner because any strategy other than Basic will fail to coordinate on the C loop.

However, the strict form of Basic can easily drift to soft or shortcut forms, which do allow re-coordination with a shorter strategy. Often this strategy is very closely related to a shorter form of Basic, and mutates into Basic when the invasion is successful, so that the shorter form of Basic can, in effect, invade a longer form directly. Note also that shorter strategies, and in particular a shorter form of Basic, can arise relatively easily by point mutation of the dominant strategy. (See Details 3: N40 S11 M5 n1 R20 T26 Pr120f Details 12-10-20**Figure not included)

A particular length of Basic may be invaded by more forgiving strategies, but is apparently never invaded directly by a less forgiving strategy. When a less forgiving form of Basic takes over from a more forgiving form, this is always after a period of chaos after the previously dominant Basic has been undermined. A common general sequence is for the dominant Basic to be invaded by a more forgiving strategy, which is then invaded by Hawk. Hawk is then invaded by a Grim-like strategy and Grim in turn is invaded by a more forgiving strategy, which may ultimately settle to any stable length of Basic.

Thus while there are a variety of possible invading strategies, they all rely on being more forgiving than the dominant length of Basic. We should therefore expect that they will be affected roughly the same way as payoff parameters and length of the dominant strategy vary: holding other parameters constant, invasion becomes easier as L increases, easier as R increases, and harder as T' increases. (See Appendix 1 for a derivation for ForgivingBasic and ExploitShortCut.)

Consideration of invasion by Hawk on the one hand and invasion by a forgiving strategy on the other suffices to explain the broad trends in the distribution of L. To summarize:

		Invasion of Basic by:	
		Hawk	Forgiving
L increases	Harder	Easier	
R increases	Harder	Easier	
T' increases	Easier	Harder	

In all cases the susceptibility to invasion by Hawk changes much more as parameters change than does susceptibility to invasion by Forgiving, since Forgiving strategies differ from Basic only in noise recovery.

Explaining the trends in the distribution of L is straightforward. As T' increases, holding R constant, invasion by Hawk becomes easier and L increases to compensate. As R increases invasion by Hawk becomes more difficult at any given T', and the dominant L becomes shorter. L is not too much longer than is needed to resist Hawk because longer strategies are much more susceptible to invasion by forgiving strategies.

To anthropomorphize our conclusions, a successful strategy must be temperate in its punishment. A strategy which is harshly moralistic is more susceptible to invasion by a more forgiving strategy than is a strategy which punishes just enough to ensure that crime doesn't pay.

Limiting Distribution of Length of Punishment Circuit

It is tempting to conjecture that in the limit of high N, infinite plays/round, and low mutation rate the probability distribution of strategy lengths would place all its mass on the shortest length of Basic for which ‘crime doesn't pay’, since this strategy maintains the highest average score when

it dominates. But the simulations looking at the effect of variation in the independent parameters, while not conclusive, do support this conjecture. Increasing P_r does increase the dominance of shorter strategies, but at the expense of the mid-length strategy. The longest strategy maintains its share of the population. Increasing N appears to make the distribution more sharply peaked, decreasing the share of both short and long strategies.

Further, we should expect that the key factor in determining the distribution is not the relative scores in two equilibrium states, but the size of the basin of attraction of these states (see e.g. Kandori, Malaith & Rob (1991)). The term ‘basin of attraction’ needs qualification here. As discussed above, longer lengths of Basic are not strictly stable against invasion by Forgiving Basic, and so they do not have a basin of attraction in the strict sense. However, as we have seen, invasion by Forgiving Basic is very unlikely because Forgiving Basic is easily exploited by a strategy which exploits its forgiveness. In landscape language, while there is a fitness increasing path away from Basic, it is narrow and any slight deviation away from it leads onto a slope which returns the population to dominance by Basic. Given the algorithm used to generate new strategies, such a deviation from the narrow path is almost inevitable. It appears that most transitions are actually the result of invasions by strategies which do strictly worse than the dominant strategy, so that it may be reasonable to model the Basic strategy as being strictly stable. In this case, shorter lengths of Basic are more stable with respect to more forgiving strategies and less stable with respect to Hawk, as compared to longer versions of Basic. It is not clear which effect will dominate as the parameters approach limiting values.

Another point which this raises is that the stability of a particular strategy depends on the algorithm used to explore the surrounding strategy space. It is for this reason that evolutionary stability, which is not tied to any particular algorithm, is not necessarily a useful indication of stability. In general it seems likely that if the strategy space is sufficiently complex that a powerful search technique, such as a genetic algorithm, is required to search it, then stability measures defined independently of that algorithm are not guaranteed to be good predictors of

behaviour.

The algorithm used in this simulation is biased in two main ways.⁴ First, the random mutation algorithm for the generation of new strategies is biased towards shorter strategies because not all states in a randomly generated strategy will be reached. This is not necessarily a disadvantage, as it is plausible that in any realistic situation which is being modelled ‘simpler’ shorter strategies might be more common as mutations. However, the simplest Hawk and Dove are particularly disproportionately generated. When $S=11$ approximately 1.25% of new random strategies are Hawk. Perhaps counter-intuitively, this is likely to *increase* the stability of lengths of Basic which are long enough to resist Hawk, by keeping in check the more forgiving strategies which might otherwise invade. This suggests that stability with respect to Hawk may be important when the “crime doesn’t pay” condition is only just satisfied, but stability with respect to more forgiving strategies becomes more important once stability with respect to Hawk passes some threshold.

Secondly, it is quite easy to form a shorter related strategies through point mutation of an existing strategy, and much more difficult to create longer strategies in the same way. In particular a shorter version of Basic can be created by a single point mutation of a soft variant of a longer version, but Forgiving Basic, which is longer than the corresponding Basic, or a longer version of Basic, are difficult to form through point mutations. Recombination can only form a longer strategy when at least one of the parents is long, which is unlikely when the population is dominated by a given length of Basic. Random mutation is not a particularly effective way of generating new strategies. This may be why invasion by a shorter length of Basic is common, while invasion by Forgiving Basic and longer lengths of Basic are rare.

⁴There is no such thing as a “neutral” search, since the adequacy of a search strategy can only be defined with respect to the phenomenon being modelled.

Stability

Now consider the stability of Basic(L). Figure Stability shows the average number of generations over which a particular length of Basic dominates the population, weighted by dominance, as well as the average length of Basic.

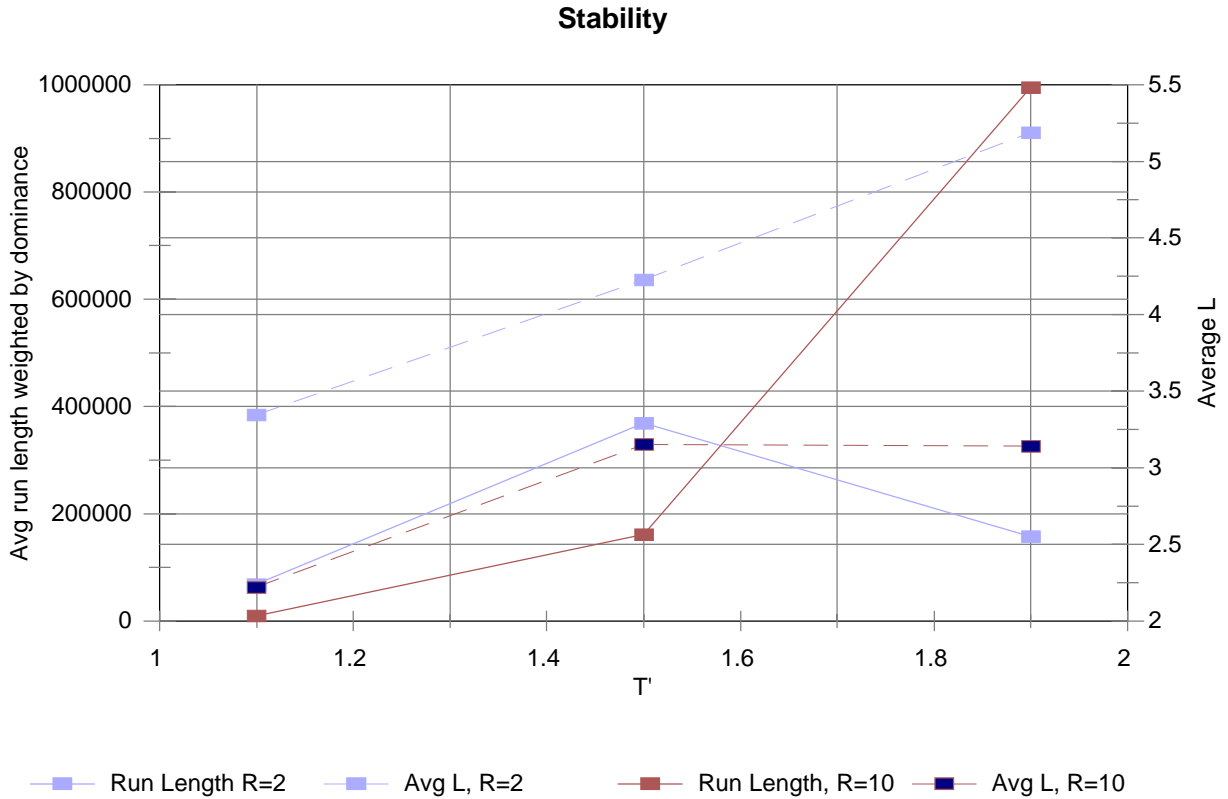


Figure Stability: ****Note: preliminary data: not reliable, particularly at higher R****. A “run” is considered to start when Basic(L) > 70% of the population, and it ends when Basic(L) < 10% of the population for more than 300 generations. With $\sigma(L)$ defined as the average length of a run of Basic(L), and $\eta(L)$ the average proportion of the population playing Basic(L), the dependent

variable, weighted run length is given by
$$\text{Weighted Run Length} = \sum_{L=2}^6 \eta(L)\sigma(L)$$

At $T' = 1.1$ the average run length is short, and it gets shorter as R increases. This is consistent with the relatively low proportion of the population dominated by Basic at low T' . Cooperation

is easier to sustain when T' is low and R high and, in particular, when T' is low, exploitation of Forgiveness is less effective, and Forgiving strategies can invade Basic relatively easily. With this in mind, the trend of increasing stability at $R=10$ can be readily explained. With high R , L remains low, with an average L of just over 3 even at $T' = 1.9$. Basic(3) is difficult for any of the strategies discussed to invade, and it becomes more difficult as T' increases. When T' increases to 1.9, none of the strategies discussed can strictly invade Basic(3).

We see a similar initial increase at $R=2$ as T' increases to 1.5, where the average L is 4.2. However, in contrast to $R=10$, as T' increases to 1.9, L also increases. While the increasing T' causes Basic to become more difficult to invade, hence the increased stability when $R=10$, the increased L more than offsets this, and with $R=2$ stability decreases as T' increases from 1.5 to 1.9.

Conclusions

While we must of course be careful in drawing direct conclusions about social norms from such simple models, this simulation provides some obvious parallels with common social norms. It reinforces the combination of “never give a sucker an even break” and “cooperate with someone who won’t be exploited” that are already familiar from the work of Nowak and May. It makes explicit the message which was implicit in their work, namely that retribution such that “crime doesn’t pay” is stable. We also see the notion that an offender must pay his dues to society, but should be reaccepted into society once that debt is paid. And we also see that punishment must be temperate – the punishment must fit the crime – if it is to be stable. The dynamics of the transition from an overly harsh punishment are interesting. A strategy with a very long punishment cycle – the strict moralist – gives way to the bleeding heart liberals, who are indeed too soft, and so can be taken advantage of by anti-social Hawks. Too strict a morality is ultimately destabilizing.

Axelrod, R, (1987) The Evolution of Strategies in the Iterated Prisoner's Dilemma, in L.D. Davis, ed., Genetic Algorithms and Simulated Annealing, Morgan Kaufmann.

Boyd, R, (1989) Mistakes Allow Evolutionary Stability in the Repeated Prisoner's Dilemma Game, 136 J. Theor. Biol., 47-56

Boyd, Robert & Peter J. Richerson (1987), No Pure Strategy is Evolutionarily Stable in the Repeated Prisoner's Dilemma Game, 327 Nature 58-59.

Kandori, M, G Mailath & R. Rob, (1993) Learning Mutation and Long Run Equilibria in Games, 61(1) Econometrica 29-56.

Lingren, K., (1991) Evolutionary Phenomena in Simple Dynamics, in Artificial Life II (D Farmer et al eds.) (Proc. Santa Fe Institute, Addison Wesley).

Nowak, M & K. Sigmund, (1993) A strategy of win-stay, lose-shift that outperforms tit-for-tat in the Prisoner's Dilemma game" 364 Nature 56.

Seth, Anil K, (1997) Interaction, Uncertainty and the Evolution of Complexity, 521-530 in Husbands and Harvey eds, Fourth European Conference on Artificial Life (MIT Press).

Appendix 1

Basic(L) v Basic(L)

B(L)	B(L)
1	1
1	0 ⁿ
0	1
0	0
(0,0) repeated a total of (L-1) times	
0	0
1	1

For notational convenience write $\pi(\text{Basic(L)}, \text{Basic(L)}) = BB$

$$BB = R - 2nR(L+1) + 2n(T+S+(L-1)P) = R - 2nR(L+1) + 2n(T+L-1)$$

The factor of 2 arises because noise occurs in respect of the play of each player independently.

Superscript “ⁿ” indicates a defection caused by noise.

Hawk invading Basic(L)

$$H(p) = HBp + HH(1 - p)$$

$$B(p) = BBp + BH(1 - p)$$

$$HB = \frac{T + L - 1}{L}$$

$$HH = 1$$

$$BB = R - 2n[R(L + 1) - T - L + 1]$$

$$BH = L - 1/L$$

$$B = H \Rightarrow p_e = \frac{1 - BH}{1 + BB - HB - BH}$$

$$p_e = \frac{1}{\{LR - T'R - L + 2 + 2n(LT'R + L^2 - L - L^2R - LR)\}}$$

$$\frac{dp_e}{dR} = \frac{(T' - L - 2n(LT' - L^2 - L))}{\{\}^2} \Rightarrow -ve$$

Since $L - T'$ is +ve for $L \geq 2$ and n is small and $(LT' - L^2 - L)$ is -ve

$$\frac{dp_e}{dL} = \frac{(1 - R - 2n(T'R + 2L - 1 - 2LR - R))}{\{\}^2}$$

-ve since $R > 1$ and n is small

$$\frac{dp_e}{dT'} = \frac{(R - 2nLR)}{\{\}^2}$$

+ve since n is small

Forgiving Basic(L) invading Basic(L)

Forgiving Basic(L) = "F", $\pi(\text{ForgivingBasic(L)}, \text{Basic(L)}) = \text{FB}$

B	F
1	1
0 ⁿ	1
1	1
1	1
1	0 ⁿ
0	1
0	1
0	0
(0,0) repeated (L-1) times	
0	0
1	1

$$\text{FB} = R - nR + nS - nR(L+2) + n(T + (L-1)P + 2S)$$

$$\text{BF} = R - nR + nT - nR(L+2) + n(S + 2T + (L-1)P)$$

$$F = FBf + FF(1 - f)$$

$$B = BBf + BF(1 - f)$$

$$BB = R - 2nR(L + 1) + 2n(T'R + L - 1)$$

$$BF = R - nR(L + 2) + n(2T + (L - 1)) - nR + nT$$

$$= R - nR(L + 3) + n(3T'R + L - 1)$$

$$FF = R - 2nR + nT$$

$$FB = R - nR - nR(L + 2) + n(T + (L - 1))$$

$$= R - nR(L + 3) + n(T'R + L - 1)$$

F invades if:

$$FB - BB > 0 \Rightarrow RL - R - T'R - L + 1 > 0$$

$$\text{If } L = 2, [RL - R - T'R - L + 1] \Rightarrow R(1 - T') - 1 \rightarrow -ve$$

Invasion is impossible for $L = 2$

$$\text{If } L = 3 \Rightarrow 2R - T'R - 2 > 0$$

$T' \rightarrow 2$, no invasion

$T' \rightarrow 1$ invasion iff $R > 2$

If $L=2$ then Basic(2) can resist if $R < 2$, otherwise any length can be invaded.

At proportion of Basic(L)=1:

$$\frac{d(FB - BB)}{dR} = n(L - T' - 1) \Rightarrow$$

+ ve for $L \geq 3$

$$\frac{d(FB - BB)}{dL} = n(R - 1) \Rightarrow +ve$$

$$\frac{d(FB - BB)}{dT'} = -nR$$

Hawk and ForgivingBasic

$$FF - HF = \frac{(R - nR(2 - T')(L + 1) - (2T + L - 1))}{(L + 1)}$$

$$= \frac{(RL + R - 2T'R - L + 1 - nR(2 - T')(L + 1))}{(L + 1)}$$

+ve, (FF can resist Hawk), ignoring noise, if $L > \frac{(R(2T'-1) - 1)}{(R-1)}$

For $R = 2 \quad T' \rightarrow 1 \Rightarrow L > 1 \quad T' \rightarrow 2 \Rightarrow L > 5$

$$FH = \frac{(L - 1)}{(L + 1)}$$

$$HF = \frac{(2T + L - 1)}{(L + 1)}$$

$$FF = R - 2nR + nT$$

$$\frac{d(FF - HF)}{dT'} = \frac{-2R}{(L + 1)^2}$$

$$\frac{d(FF - HF)}{dL} = \frac{(R - 1 - 2nR) - (RL + R - 2T'R - L + 1 - 2nR(L + 1))}{(L + 1)^2}$$

$$= \frac{-(L(R - 1) + 2 - 2T'R - 2nRL)}{(L + 1)^2} \Rightarrow \text{-ve if } L > \frac{2(T'R - 1)}{(R - 1)}$$

$$\frac{d(FF - HF)}{dR} \cong \frac{(L + 1 - 2T')}{(L + 1)^2}$$

ExploitShortCut invading Basic(L)

ExploitShortCut = "E"

B	E
1	1
0 ⁿ	1
1	1
1	1
1	0 ⁿ
0	1
0	1
0	0
0	1
1	1
1	1

$$EB = R - nR + nS - n5R + n(T + 3S + P)$$

$$BE = R - nR + nT - n5R + n(S + 3T + P)$$

When L2 = number of states in E,
 $3 \leq L2 \leq L$

$$EB = R - nR - nS - nR(L2 + 2) + n(T'R + 3S + L2 - 2)$$

$$= R - nR(L2 + 3) + n(T'R + L2 - 2)$$

$$BE = R - nR + nT'R - nR(L2 + 2) + n(S + 3T'R + L2 - 2)$$

$$= R - nR(L2 + 3) + n(4T'R + L2 - 2)$$

$$HE = (2T'R + L2 - 2) / L2$$

$$EH = (L2 - 2) / L2$$

The form of Exploit Basic shown here exploits a short cut in the dominant Basic which occurs immediately after the S loop. By introducing a suitable number of defection states, a shortcut at any position can be exploited, although the exploitation becomes less effective. The Basic(5) shown in Figure Shortcut can be invaded by the strategy shown in Figure ExploitShortCut. It is the same as Forgiving Basic when L=2, it is slightly less effective than Forgiving Basic at invading when L=3, and it becomes more effective at invading than Forgiving Basic for L>3 and for R sufficiently large (e.g for R>2 when L=4).

"E" indicates "Exploit Short Cut" strategy

This derivation is for the strategy shown in Figure ExploitShortCut, which exploits a shortcut in the first state after the S loop.

Shortcuts at later points may also be exploited, but less effectively.

$$EB = R - nR + nS - 5nR + n(T + P + 3S) = R - 6nR + n(T + 1)$$

Invasion is possible if:

$$EB - BB > 0 \Rightarrow$$

$$2LR - 4R - T'R - 2L + 3 > 0$$

$$\frac{d(EB - BB)}{dR} = 2L - 4 - T' \Rightarrow +ve \text{ for } L \geq 3$$

$$\frac{d(EB - BB)}{dT'} = -R \Rightarrow -ve$$

$$\frac{d(EB - BB)}{dL} = 2(R - 1) \Rightarrow +ve$$

$$HE = \frac{(2T'R + 1)}{3}$$

$$EH = \frac{1}{3}$$

$$BE = R - nR + nT'R - 5nR + n(3T'R + P + S) = R - 6nR + n(4T'R + 1)$$

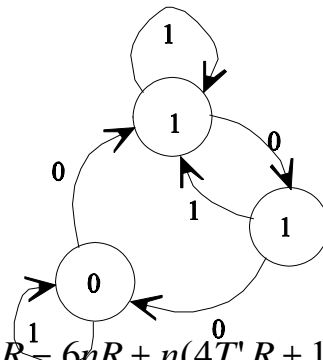


Figure **:ExploitShortcut

Relative advantage of ExploitShortCut to ForgiveBasic:

$$(EB - BB) - (FB - BB) = RL - 3R - L + 2$$

$$\frac{ddiff}{dL} = R - 1 \Rightarrow +ve$$

$$\frac{ddiff}{dR} = L - 3 \Rightarrow +ve \text{ for } L > 3$$

For $L=2$ Exploit Short Cut and ForgiveBasic are the same. For $L=3$ Forgiving Basic does better because it receives P as a final payoff before returning to mutual cooperation whereas ExploitShortCut receives S. Note also that ESC can only invade soft forms of Basic(3) whereas ForgiveBasic can invade the strict form. For $L>3$ Exploit Short Cut does better (against the Basic(L) with a short cut) provided R is sufficiently high.

Basic(L₂) invading Basic(L) Soft, L₂ < L.

$$BS = R - 2nR(L + 1) + 2n(2T'R + L - 2)$$

$$SB = R - 2nR(L + 1) + 2n(T'R + L - 2)$$

$$SS = R - 2nR(L_2 + 1) + 2n(T'R + L_2 - 1)$$

$$HS = \frac{(T'R + L_2 - 1)}{L_2}$$

$$SH = \frac{L_2 - 1}{L_2}$$

By adding dummy states, it is always possible to construct a strategy which receives a score

shown above, provided it invades a Basic which is soft at the right spot. Notice also that mutation is very easy from the soft Basic: the soft state is the one that has to make the switch: e.g. Basic(2) below can invade Basic(5) which is soft at state 3.

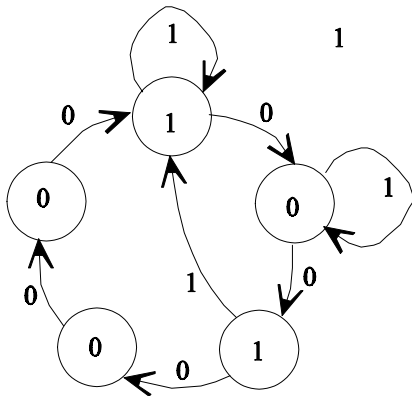
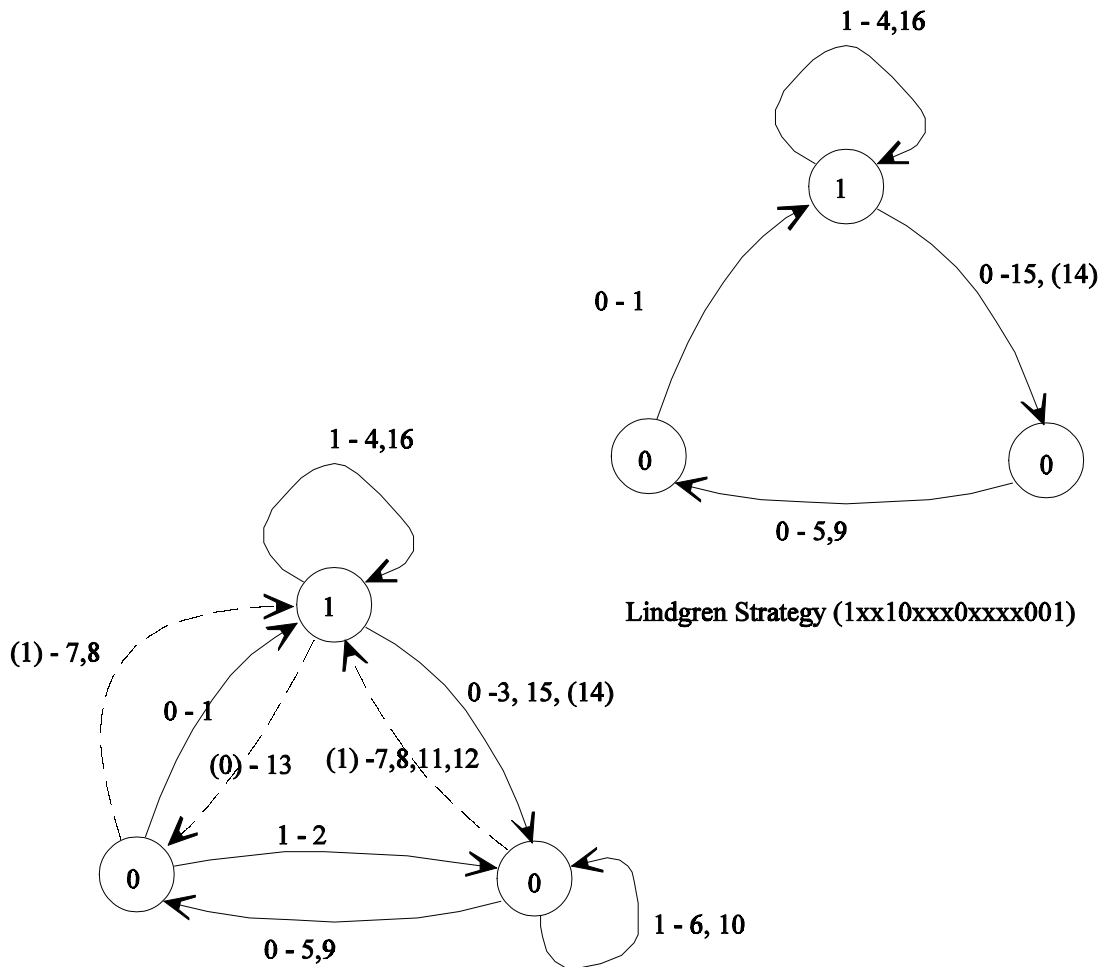


Figure **:Basic(2) invade Basic(5) soft

Appendix 2

Interpretation of Lindgren (1991) strategies.

Path	History				Play	
	A(t-2)	B(t-2)	A(t-1)	B(t-1)	A(t)	A(t)
1	0	0	0	0	1	1
2	0	0	0	1	x	0
3	0	0	1	0	x	0
4	0	0	1	1	1	1
5	0	1	0	0	0	0
6	0	1	0	1	x	0
7	0	1	1	0	x	0
8	0	1	1	1	x	1
9	1	0	0	0	0	0
10	1	0	0	1	x	0
11	1	0	1	0	x	0
12	1	0	1	1	x	1
13	1	1	0	0	x	0
14	1	1	0	1	0	0
15	1	1	1	0	0	0
16	1	1	1	1	1	1



Lingren Strategy (1001000100010001): The Markov process is constructed assuming the histories arose with the minimum number of noise events. Dashed lines indicates a path which is a change of state caused by erroneous play by the strategy. History 14 is a change of state when an erroneous play is made in state 1. These paths and this play cannot arise in my simulation, in which an erroneous play does not cause a change in strategy.