ENFORCEMENT LEARNING:
A GENETIC ALGORITHM APPROACH TO THE ECONOMICS OF TAX EVASION

by

Bryan Vincent Yu
Bachelors of Arts (Honours)
University of Manitoba 2003

PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF ARTS

In the
Department
of
Economics

SIMON FRASER UNIVERSITY

August 2004

APPROVAL

Name: **Bryan Vincent Yu**

Degree: **Master of Arts**

Title of Project: **Enforcement Learning: A Genetic Algorithm Approach to the Economics of Tax Evasion**

Examining Committee:

Chair: **Professor Terry Heaps**
Associate Professor, Department of Economics

Professor Jasmina Arifovic
Senior Supervisor
Professor, Department of Economics

Professor Steeve Mongrain
Supervisor
Assistant Professor, Department of Economics

Professor Gordon Myers
Internal Examiner
Professor, Department of Economics

Date Defended: August 4th, 2004

SIMON FRASER UNIVERSITY

Partial Copyright Licence

ABSTRACT

A traditional assumption that underlies most economic theory is the postulate of agent rationality. In an environment where agents are fully informed about their respective environments, agents are assumed to instantaneously optimize their objective functions. This assumption is arguably extreme since it overestimates the computational ability of agents to solve complex optimization problems. This paper relaxes this assumption in a tax evasion setting. Through the use of Genetic Algorithm learning, it is shown that even in an environment where tax authorities lack the explicit ability to optimize, theoretically optimal enforcement strategies can be learned through a process resembling natural selection. This approach not only leads to theoretically valid results, but it does so in a manner that more accurately reflects the adaptive behaviour and innovative flair of humanity.

DEDICATION

To my family and the memory of a friend and mentor Ajmal Mohamed.

ACKNOWLEDGEMENTS

I would like to acknowledge my supervisors Dr. Jasmina Arifovic and Dr. Steeve Mongrain for their insightful comments and help in this research project. Further, I am grateful to Scott Skjei for his time and comments on this project as well as his words of encouragement and interesting debate during my time here. And of course, to all the friends I have made in the program, I thank you all, with special recognition in no particular order to Kemi Afolabi, Michael Gager, Edward Sit and Joyce Wan.

TABLE OF CONTENTS

LIST OF FIGURES

Environment 1

Environment 2

Differential Tax Rates

LIST OF TABLES

1 INTRODUCTION

A traditional assumption that underlies most economic theory is the postulate of agent rationality. Part of this assumption entails that fully informed agents instantaneously optimize their objective functions through the selection of optimal choice variables. However, this assumption is strong and gives too much credit to the computational ability of agents when making decisions. As result, it disregards the dynamic learning process that potentially occurs even in what appear to be simple decisions. In contrast to full rationality, it seems that decisions are made with a general recognizance of our surrounding environment. In particular, we learn from the actions of others, adopting the choices that worked well for them, while at the same time experimenting with our choices in hope that better results emerge. Sometimes, incorrect choices are adopted, but given enough time, through trial and error the correct decisions tend to be made. In a sense this brings static optimization into a dynamic framework.

This paper relaxes this assumption of full rationality in a tax evasion setting. It shows that in an environment where tax enforcement authorities lack the explicit ability to optimize, theoretically optimal enforcement strategies can be learned in process that captures the adaptive and innovative flair of human behaviour.

The paper first outlines the justification of using tax evasion as a basis of study, and follows with a survey of traditional tax evasion and enforcement models. The genetic algorithm learning mechanism is introduced and applied to the decision making process of the tax enforcement agency, simulations and interpretation conclude.

2 WHY SHOULD WE STUDY TAX EVASION?

Tax evasion is often viewed as an insignificant subtopic within the realm of the economics of crime. However, certain implications that result from tax evasion spill over in to public economics that make it worth studying. By definition, *tax evasion* is the intentional failure of individuals to declare taxable economic activity to the authorities. This is an illegal practice as opposed to *tax avoidance*, where individuals reduce their own tax payments in ways unintended by the taxing authority but permissible under law (Franzoni, 1998).

Tax evasion itself stems from the existence of imperfect information. The tax base (taxable activity) is often unobservable by the tax authority, and thus the true tax liability is uncertain without a costly auditing system. In contrast, individual taxpayers are informed of their true taxable income, abstracting from inadvertent measurement errors, and can take advantage of this to elude taxes for a potential financial gain.

The negative implications of evasion are not immediately apparent since individual agents arguably increase their well being through the failure to report income. However, the traditional role of the government is to maximize the level of social welfare. In order to attain this goal, often through the provision of public goods and a redistribution of income among individuals, taxes must be collected to finance these activities. Under the assumption that taxes are set optimally in order to just satisfy the government's budget constraint, evasion activities result in the under provision of public goods, which in turn harms the well being of the general public. However this implication

hinges on the assumption that the extra income garnered from evasion is not reinvested in taxable activities that would increase the tax base (Myles, 1995).

2.1 The Extent of Tax Evasion

By nature, the amount of tax evaded in an economy is a concealed value. It is difficult to measure the true amount of evasion, since true income is never fully revealed. Nonetheless, estimates of the magnitudes of evasion have been made throughout the world, and the numbers are quite significant. A leading indicator of evasion is the tax gap, or the differential between what is actually owed and what is paid voluntarily. In the US, from the years 1973 - 1992, the nominal tax gap increased by a multiple of five, from $22.7 B to $95.3 B. Using the Taxpayer Compliance Measurement Program (TCMP) of the Internal Revenue Service (IRS), this significant level of evasion can be attributed to the estimate that 40 percent of U.S. households underpaid their taxes for the year, and a quarter underpaid by a significant amount (over $1,500) (Andreoni et al, 1998)

The TCMP is not the only measure of evasion, nor is evasion unique to the American economy. For example, Rey (1965), in an analysis of the Italian General Sales Tax found evasion of over 50 percent of the actual tax yield. Feige (1979) found that undeclared economic activity was equivalent to 15 percent of GNP. In Pissarides and Weber (1989) study, unreported economic activity amounted to roughly 5.5 percent of the GNP. Although these measurements may be imprecise, they do highlight the amount of unreported activity, and hence the tax evasion in the economy, and therefore justifies further study in the subject.

3 LITERATURE REVIEW AND THE BASIC MODEL

3.1 Basic Model

The application of a genetic algorithm learning mechanism to the economics of tax evasion necessitates first a review of the theoretical models of compliance and enforcement from which this paper is based. In particular, primary interest rests in an analysis of the standard expected utility model of tax compliance, and the determination of the corresponding enforcement rules.

Tax evasion can be viewed in the form of a principal-agent framework. The agent (taxpayer) decides whether or to what degree he commits a crime. This decision is made subject to the principal's (enforcement agency's) imposition of disincentives used in order to curb the undesired behaviour. One of the early models in this field of tax compliance is attributed to the pioneering work of Allingham and Sandmo (1972). In their model, the extent to which taxes are evaded is modelled as a portfolio allocation decision or a choice under risk.

The construction of Allingham and Sandmo's model relied heavily on Becker's (1968) theoretical analysis of the economics of crime and punishment, and was inspired from his assertion that his model was applicable to `white collar' crimes such as tax evasion. From this basis, Allingham and Sandmo modelled the taxpayer's decision with the following functional form.

$$EU(x) = (1-p)U(c) + pU(c - Se)$$

where disposable income is $c = y - \tau x$

and income evaded is $e = y - x$

As a portfolio allocation decision, a risk adverse agent decides how much of his exogenously determined taxable income y to report to the government, or equivalently how much of their tax burden they are willing to evade e. In essence, this is equivalent to the decision to invest in a risky asset. There is always a chance that the agent is the victim of an audit and therefore subject to a sanction or a fine above and beyond the tax value of their unreported income (Franzoni, 1998). If the agent is audited, it is assumed that the enforcement agency accurately uncovers their true tax liability and that they are convicted with 100 percent probability. If the taxpayer is very risk adverse, he reports his full income, and pays his taxes in full ($e=0$); otherwise he reports a fraction of his true income and risks audit. It is clear that this model directly integrates what Becker deemed to be the two most important variables that alter the agent's criminal behaviour- detection probabilities and punishments. In particular, it is the probability of audit that determines the degree of risk the agent faces at any given period.

The agent maximizes the above von Neumann-Morgenstern utility function through the choice of income, where expected utility is assumed to increase solely in the level of net income. Furthermore, government imposed tax rates τ, the probability of detection or audit p and sanction levels S are publicly announced and known to the taxpayers prior to their decision-making.

3.2 Theoretical Results

Allingham and Sandmo's primary motivation in undertaking their study was to examine the relationship between the level of tax compliance and the government tax rate. On the basis of the agent's choice function, the effect is ambiguous. An increase in

the tax rate has two offsetting effects on compliance levels. First, increased tax rates lower the disposable income garnered from full compliance. Under the assumption of decreasing absolute risk aversion, the agents are more risk averse to evasion with any decrease in disposable income, hence would likely cheat less. However, increased tax rates with a constant sanction level imply that the returns from cheating have increased, which in turn encourages further cheating (Andreoni et al, 1998). However, Yitzhaki (1974) has shown that under the assumption that the sanction is proportional to the amount of tax evaded, the effect on the evasion level is unambiguously negative.

Finally, they show that the level of tax evasion is inversely related to both the audit rates and the sanction levels. Although, this result was of secondary importance to Allingham and Sandmo, it plays a primary role in this paper and the implementation of the genetic algorithm. It should also be noted that the unambiguous disincentive effect of the enforcement policy conforms to Becker's analysis.

In order to `close' this model, the role of the tax enforcement agency and how optimal enforcement policies are chosen should be reviewed. It was stated that the government needed to satisfy certain revenue constraints to attain their social goals. Therefore, taxes must be enforced in a way that taxpayers are induced to limit or cease altogether their practice of evasion. It is sometimes assumed that a single taxing authority jointly determines the tax rates and enforcement of taxes. However, in practice the taxing authority appoints an independent enforcement agency, for example the IRS, to collect taxes (Myles, 1995). In turn, this agency chooses an enforcement policy that consists of independently determined audit and sanction rates to maximize their objective function that is assumed here to be a revenue function.

Under the assumption that the tax authority and the enforcement agency are independent entities, the government can choose a tax rate that they believe maximizes social welfare. This tax rate is assumed chosen in order to just finance their government spending levels and becomes an exogenous variable imposed on both the taxpayer and the enforcement agency. Hence, through the optimization of the revenue function, the enforcement agency indirectly maximizes the level of social welfare.

The enforcement agency thus optimizes the following revenue function, solving for optimal levels of audit rates and sanctions.

$$R = \tau x + pSe - C(p)$$

The total government revenue in this case is a function of the taxes collected on reported income x. In addition, the government collects a fraction of the unreported income in the form of sanctions, where this fraction p is determined by the vigilance of enforcement agency, and corresponds to the probability of audit. Finally, revenues take into account the cost of detection C, which itself is a function of the audit rate p.

The specification of this enforcement model engrains the traditional assumption that detection is a costly venture while moving from one sanction level to another is costless (Myles, 1995). The auditing procedure itself is assumed to require higher levels of expenditure as the agency chooses to be more vigilant in their enforcement to catch the marginal evaders. This follows from the argument that increasing detection rates necessitates both increased policing efforts and more expensive technologies. Hence, following Becker's assumption, the marginal cost of apprehension is an increasing function of the audit rate, hence $C'(p)>0$.

On the basis of this specified environment, in order to minimize evasion, or to maximize revenues, it follows that the optimal enforcement strategy is to increase both the audit rates and the corresponding sanctions. However, since increases in the audit rate requires public expenditures, while sanctions do not, the optimal strategy is to set maximum sanctions with detection rates just sufficient to induce agents to follow the law.

3.3 Extensions of the basic model

While the expected utility framework of Allingham and Sandmo seems simplistic in nature, it has nevertheless remained the primary building block for future studies. In particular, the level of income y has been endogenized through the incorporation of the labour supply. However, the introduction of labour supply, has led to ambiguous effects of the enforcement variables. For example, increased enforcement levels decrease the effective wage rate faced by the taxpayer, inducing them to lower their supply. However, given a backward bending supply curve, increased enforcement may lead to increase labour supply and thus higher levels of evasion (Andreoni et al, 1998).

Furthermore, with an extension to dynamic modelling, Engel and Hines (1998) endogenized the probability of detection (or enforcement audit) that they modelled to be dependent on past evasion behaviour. Under this model, current evasion is a decreasing function of past evasion, where if caught in one year they may be penalized for the prior years behaviour. This allows for a more complex structure that further enriches the original model.

3.4 Criticisms

Although Allingham and Sandmo present an attractive model to depict agent behaviour in an economy with tax evasion, it has been duly criticized on the grounds that it is not representative of the empirics of evasion. Under the portfolio allocation model, the agent determines the magnitude of evasion rather than a binary choice of whether or not to evade. In order to maintain high levels of tax compliance, it has been shown that optimal policies consist of both high sanctions and relatively low audit rates. However, empirical evidence suggests that penalties are never set this high, and are usually in the range of 25 percent of the unpaid liabilities (Engel and Hines, 1998). Furthermore, it can be inferred from the fact that only 1 percent of tax returns are typically audited, the probability that an agent is caught is extremely slim. It should therefore be expected that given these high incentives, tax compliance should be low. Previously, it has been shown that evasion in general is an increasing problem within the global economy. However, the actual magnitude is much lower than what is implied by the above model since a majority of all agents do truthfully report their income, and pay their taxes.

As a result, the specification of this model that assumes decision making based purely on financial self-interest and perfectly amoral agents is overtly limited. In order to realistically model the tax compliance environment, there necessitates the integration of non-economic factors and variables that influence the agent behaviour in a moral sense. In particular, critics would argue that of great importance is the idea of social norms and role of the stigma that accompanies disobedience as a primary explanation.

The addition of social norms necessitates introduction of social interaction among agents (Cowell, 1990). In particular, the idea that tax evasion creates an externality effect

in terms of social stigma that is attributed to the violation of social norms. In an economy where the majority of agents pay their taxes, compliance becomes the social norm. The act of tax evasion carries with it a large stigma effect that discourages evasion. Evasion in this case lowers the corresponding utility levels. However, the integration of social norms also introduces multiple equilibria; high and low levels of evasion. If agents evade in larger numbers, the stigma effect is low, hence, evasion becomes a new social norm and evasion is not discouraged. The stigma attached to tax evasion decreases, and non-compliance actually spreads to a larger fraction of the population. Thus, it is implied that the level of tax evasion inferred is attributed to an economy that follows the former environment. Furthermore the importance of norms creates a third tool for the enforcement agency or the government. It can implement new policies to promote these types of tax norms that increase this stigma effect (Edlund and Aberg, 2002).

However, empirical analysis of the effect of social norms is difficult given that it is not an easily measurable variable, and necessitates inefficient proxies. In order to test this theory, a macroeconomic framework is commonly used, where social norms are measured as a country specific attribute. Edlund and Aberg (2002) use OECD data to show that social tax norms are insignificant in the determination of tax evasion. Furthermore, they state that any tax norm is seldom strong enough in the minds of individuals to counteract their self-interested tendencies. However, adopting an experimental methodology, Sanchez and Juan (1995) argue that a substantial difference in tax evasion behaviour exists between individuals in Spain and the United States. Given the similarity in results attributable of their other incentives, for example the implementation of public goods, sanctions and audit rates, they attribute the residual to

the social norms prevalent in each country. They further find as expected that agents clearly respond to the financial disincentives of audit rates and sanctions.

Although Allingham and Sandmo's model may be oversimplified, given the measurability problems and the ambiguous effects of other social determinants, it nonetheless is a valid platform to analyze the topic at hand.

Criticisms have also arisen from the simple enforcement model. Although Engel and Hines (1998) model tax evasion as determined by past evasion behaviour, there remains the problem with assuming a random audit structure. Each agent faces the same probability of detection; however, it is probable that detection rates are determined by economic status. The authorities likely target some occupations more so than others, hence detection rates are not uniform in nature, and this should be reflected in the model (Andreoni, 1991).

Just as the audit rates may be non-uniform, the case may be true for sanction levels. The above enforcement-compliance model assumes a constant sanction rate; however, it may be more realistic to have it dependent on the level of evasion e. This follows from the relaxation of the strong assumption made about enforcement policy decisions. Although the specification of a government appointed agency enforcing taxes through independently chosen sanction and audit rates is more accurate than placing this responsibility into the hands of the taxing authority, it may still be too simple. Rather, it may be more realistic to incorporate a judicial system that imposes the sanction levels on the enforcement agency, which then needs only to choose optimal detection probabilities.

Using this specification of the enforcement-compliance framework, Andreoni (1991) could partly explain why sanction and audit levels would not be as high as the

traditional model would predict. In particular, he argues that the probability of audit and the sanction levels are not independent variables. Rather, as the sanction levels increase the audit or conviction rates would fall. This is due to the aversion of the jury to Type II errors or the conviction of agents for crimes which they may be innocent of. Hence, for small crimes they are unwilling to impose stiff penalties. In short this model predicts that penalties fit the severity of the crimes, which is more representative of the real world.

Thus far, the importance of studying tax evasion and the traditional models used to explain the problem have been discussed. Clearly, the above extensions to both Allingham and Sandmo's (1972) tax compliance model and the enforcement model create a richer environment which to work. However, the simple environment outlined above is an adequate platform to show whether an enforcement agency can learn to implement optimal policies in a traditional albeit simplified framework. If optimal policies can be learned, future research can integrate intricacies such as the presence of judicial systems or social norms discussed above.

4 FUNCTIONAL FORM OF THE MODEL

Following the above discussion, the model used in this paper is comprised of two economic agents; the risk adverse individual taxpayer and a government appointed tax enforcement agency. This is a two-stage decision process where the taxpaying agent initially reports some level of their taxable income to the government. This follows with the response by enforcement agency that chooses and optimal enforcement policy.

Individual taxpayer

The agent chooses the level of income to report in order to maximize their expected utility, which is assumed to increase solely as a function of net income. It is also assumed that government imposed tax rates τ, probability of detection or audit p and sanction levels S are publicly announced and hence known to the taxpayers prior to their decision-making. The objective of the taxpayer is therefore to maximize the following expected utility function:

$$MaxEU(x) = (1-p)\ln(y-\tau x) + p\ln(y-\tau x - S(y-x)) \qquad (1)$$

which yields an optimal value of reported income x, for the agent of:

$$x = \frac{\left[\left(1-\frac{p(S-\tau)}{(1-p)\tau}\right)y - Sy\right]}{\left[\left(1-\frac{p(S-\tau)}{(1-p)\tau}\right)\tau - S\right]} \qquad (2)$$

Tax enforcement agency

The enforcement agency thus maximizes the following revenue function, solving for optimal levels of detection and sanctions. The reported income of agents is assumed to be contemporaneously exogenous.

$$\underset{p,S}{Max}\, R = \tau x + pS(y - x) - p^2 \qquad (3)$$

Furthermore, it is assumed that it becomes increasingly difficult to catch marginal evaders; hence the cost of increasing the probability of detection increases at an increasing rate of *2p*.

From the specification of this revenue function and solving for first order conditions, the optimal choice for the enforcement agency is to set sanctions at maximal levels, with relatively low detection rates (see Myles, 1995 and Becker, 1968). The question that follows is if the postulate of full agent rationality is relaxed, can optimal enforcement policies be *learned over time*? To answer this, a genetic algorithm learning mechanism is employed.

5 THE GENETIC ALGORITHM

5.1 Introduction of the Genetic Algorithm

The genetic algorithm is an adaptive learning mechanism developed by Holland (1975) where optimization of an objective function occurs through processes that imitate natural selection and genetics. Optimization occurs through an adaptation of behaviour that did well in the past with the occasional process of experimentation. Often, this process converges to values comparable to those yielded by first-order conditions when solving optimization problems provided that they exist (Brooks, 2000).

In general, there are 3 applications of the genetic algorithm in the literature. In applications of *estimation*, the algorithm is used to solve high dimension optimization problems that do not necessarily have closed forms. Furthermore, where multiple equilibria exist, the genetic algorithm provides a method to decide between different maxima, meaning the global as opposed to local maxima is converged upon. For the purpose of this paper, the genetic algorithm as a behavioural metaphor of learning is of importance. The genetic algorithm is arguably more realistic in portraying the adaptive reasoning and learning process that characterizes human behaviour while necessitating minimal levels of mathematical competence.

As opposed to solving complicated mathematical formulas for optimal values, agents in the genetic algorithm framework act on the basis of imitation and experimentation. In a behavioural sense, this means that agents are aware of the choices made by others around them. They attempt to mimic ideas (or *decision rules*) that

performed well in the past, while disregarding those that were the least fruitful. Furthermore, the elements of ambition and innovation that characterize human behaviour are well represented in this learning model. Agents can further experiment with these adopted ideas with the intention of creating better ideas. However it is clear that innovation does not always lead to better results, and may entail a regression in performance. These aspects of behavioural learning are well represented using the genetic algorithm methodology; in particular, using the *replication, crossover* and *mutation* operators.

However, it is unclear why agents trying to optimize would adopt strategies resulting from innovation that have higher probabilities of being inferior given their present information. Hence, through the introduction of Arifovic's (1994) *election* operator it is assumed that agents have the intelligence to compare the outcomes of the original idea with the potential outcomes of the newly created ones. They then adopt those that yield better performance. The implementation of this election operator yields the *Augmented Genetic Algorithm* (Arifovic, 1994).

5.2 Application of the Genetic Algorithm

In this application of the genetic algorithm to the tax evasion and enforcement behaviour, the assumption of taxpayer rationality is maintained. Taxpayers optimally choose levels of taxable income to report to the government that maximize their expected utility. This assumption is relaxed for the enforcement authority, which must learn over time what constitute optimal detection probabilities and sanction levels to implement. It seems that the structure of the enforcement agency would lend itself to decision making by trial and error. As in many firm based scenarios, it is probable that there exist multiple

sub-departments that essentially work on the same issues, for example, how to counteract tax evasion. It is realistic to assume that these departments are in continuous contact with each other, adopting each other's strategies that worked well while maintaining their own level of research and development. Furthermore, the structure of the department is not necessarily static. If employees perform poorly, those that populate the high-performance department potentially replace them. This movement of agents in and out of the population also means fresh thinking and innovation. As a result, learning at the agency level seems an appropriate stage to showcase the genetic algorithm. Although there is little reason not to have both agents learning, the purpose of this research is to model the enforcement agencies learning process. Furthermore, since the taxpayer needs only to optimize with respect to a single variable while the agency is optimizing a single equation with two unknown variables, it seems more plausible that taxpayer rationality can be maintained.

The GA discussed above is slightly modified in order to be a more realistic representation of the economy. Both a single tax enforcement agency and a representative taxpayer populate the economy. The agency has mutually competing ideas about what enforcement policies to implement in a given environment. For practical purposes, these ideas may be thought of as a collection of rules proposed by the various sub-departments and those used by other enforcement agencies in the world. From this collection, the director of the agency can implement only one enforcement policy at any given time. It is this set of competing ideas that is subject to the genetic algorithm learning operations of reproduction, mutation and election.

Two n-sized initial populations of chromosomes represent the competing ideas or decision rules at a given time period *t*, and denoted A_t and B_t. Each of these individual chromosome ($A_{i,t}, B_{i,t}$) are a binary string of a finite length *l*, where each value ($a_{i,j}, b_{i,j}$) in the string is written over the *{0, 1}* alphabet. The decoding and normalization of these strings yield potential values for the choice variables, where $A_{i,t}$ decode to detection probabilities $p_{i,t}$ and $B_{i,t}$ represent different sanction levels $S_{i,t}$ For example, in the following set of populations of arbitrary string length *l*,

$A_{1,t}$: *10010101* $B_{1,t}$: *00001010*

$A_{2,t}$: *01001100* $B_{2,t}$: *10100100*

there exist two potential enforcement policies (detection probability and sanction combinations). The first enforcement policy at time period t is represented by the set $\{A_{1,t}, B_{1,t}\}$ while the second policy set is $\{A_{2,t}, B_{2,t}\}$

For a string *i* of exogenously determined length *l*, the decoding works in the following way:

$$z_{i,t} = \sum_{k=1}^{l} a_{i,j}^{k} 2^{k-1} \qquad z_{i,t} \in (0, \overline{K}) \qquad (4)$$

$$i=1,.., n$$

$$j=1,.., l$$

where $a_{i,t}^{k}(b_{i,t}^{k})$ is the bit value *{0, 1}* taken at the k^{th} position in the string. After the string is decoded to a set of values $z_{i,t}$, it is normalized in order to obtain a set of real numbers that represent the potential detection probabilities and sanctions that the

enforcement agency can impose on taxpayers. The normalization takes the following form:

$$q_{i,t} = z_{i,t} / \overline{K} \qquad (5)$$

where $\overline{K}$ is the maximum value that the bit string can decode to. For example, a string of length $l=5$, has a maximum string value defined by:

A= 11111

the decoding equation (4) yields a value of $\overline{K}$ =31

From the normalization of these decoded strings, the potential revenue associated with each combination of the decision rules can be determined. These potential revenue levels solved at time period t using the taxpayers previous reported income levels x_{t-1} are known as the fitness $\mu_{i,t}$ where $i=1,...,n$.

$$\mu_{i,t} = R_{i,t} = \tau x_{t-1} + p_{i,t} S_{i,t}(y - x_{t-1}) - p_{i,t}^2 \qquad (6)$$

The enforcement agencies decision rules are then updated using the replication, mutation and election operators. The crossover operator is omitted from this GA application for reasons explained later in the paper.

5.2.1 Replication

The replication operator makes copies of the individual sets of chromosomes on the basis of the *potential fitness* of the decision rule sets. Replication occurs through the implementation of a biased lottery, which uses the fitness values as the criterion for chromosomes to be introduced into the new set of potential decision rules. The higher the fitness value that an enforcement policy decodes to yield a higher probability of being

chosen. The probability that an enforcement policy ($A_{i,t}, B_{i,t}$) is copied into the population set ($A'_{i,t}, B'_{i,t}$), is given by

$$(A'_{i,t}, B'_{i,t}) = \mu_{i,t} / \sum_{i=1}^{n} \mu_{i,t} \qquad i=1,\ldots, n$$

This lottery is repeated n times, until n strings from the old populations of audit and sanction rates are chosen. This creates the new pool of strings that undergoes the mutation and election operators. It may be questioned why the best performing strings are not automatically copied into the new pool, and why the poorly performing strings have any chance at all. However, this is a good metaphor to model the risk aversion of agents. Arguably, they are taking into consideration the small probability those last periods best performing rules may not do as well in the following period. Hence, they do not wish to rid themselves of all rules that may be preferable in a different environment.

5.2.2 Mutation

The mutation operator, which is used as a metaphor to incorporate agent innovation, is performed by random modifications to the individual bit values of a chromosome. The potential that any single bit in a string will be altered is small, and occurs at an exogenously determined rate of *pmut*. The change in any bit occurs independently of changes to any other bit value in the string. This mutation operator affects both the strings representing detection probabilities and sanction levels.

As opposed to the replication operator that reduces the variation of the rules, the mutation operator introduces new diversity into the population; the degree of which determined by the probability of mutation (Arifovic, 1994). This mutation operator allows the agent to search for optimal rules within the entirety of the potential rule space.

This is necessary since the assumption that the optimal policy exists within the initial population and therefore can be learned purely through replication is unrealistic.

5.2.3 Election

After replication and mutation, the final process is to implement the election operator. The role of this operator is to reduce the deviation of decision rules from their optimal quantities. This occurs by comparing the fitness values of the set of decision rules in the post-replication pool denoted the *parents* and those that could result if the mutated strings, the *offspring*, existed in the same environment as the parents. If the offspring yield a better potential fitness than the parents, they in turn take the parents place in the mating pool otherwise they are discarded.

To exemplify the election operator used in this study, assume that an enforcement policy from the replicated population consists of:

$$A_{i,t}: \ 10010101 \qquad B_{i,t}: \ 00001010$$

Mutation of these strings may yield:

$$A'_{i,t}: \ 11011011 \qquad B'_{i,t}: \ 11000111$$

It is assumed that agents are equipped with enough intelligence to realize that old (new) audit rates may work well with new (old) sanction rates. This yields *three* potential enforcement policies or *offspring* to be compared with the *parent* string.

1. $(A_{i,t}, B_{i,t})$
2. $(A'_{i,t}, B_{i,t})$
3. $(A_{i,t}, B'_{i,t})$
4. $(A'_{i,t}, B'_{i,t})$

Enforcement policies 2-4 are decoded and used to solve their potential fitness values using equation (6) that would prevail if they existed in their parent's $(A_{i,t}, B_{i,t})$ environment. These enforcement policies are then ranked according to their potential fitness values. The offspring with the highest potential fitness value replaces the parent string, providing that its potential fitness is higher than the parent's actual fitness. Otherwise, all of the offspring are discarded, and the parent rule remains in the population. This is repeated n times for each of the parent-offspring combinations. This new population of strings that emerges becomes the initial population in the following time period.

Once this GA learning has been completed for an individual time period, a single decision rule set (p_t, S_t) is selected from the new string population on the basis of its relative fitness levels. These detection probabilities and sanction are announced to the taxpayer who then optimizes their level of reported income according to their following first order condition derived from (1):

$$x_t = \frac{\left[\left(1 - \frac{p_t(S_t - \tau)}{(1-p_t)\tau}\right)y - S_t y\right]}{\left[\left(1 - \frac{p(S_t - \tau)}{(1-p_t)\tau}\right)\tau - S_t\right]}$$

This reported income becomes known information to the enforcement agency and used in the following period of enforcement learning which a repeat of the above GA application. As these decision rules approach optimal values, due to the inclusion of the election operator, the variation of the population rules approach zero since any mutation

of the strings yield potential fitness values inferior to those of the parent population of strings (Arifovic, 1994).

5.3 The Crossover

As stated prior, the crossover operator has purposely been omitted from this study. However, it usually plays a prominent role in the attempt to explain the behavioural learning process as agents reach optimal solutions. Hence, its application should be explained briefly. The crossover operator is a metaphor for the interactions that take place between different rules. In particular, it attempts to portray how agents learn to mix and match different aspects of the potential rules in their population set in order to create "better" rules.

This operator is implemented on the new set of chromosomes after replication has occurred. The sets of chromosomes (the different enforcement policies in the case of this study) are then mated through a random exchange of their parts. However, this exchange is probabilistic in nature, and occurs at an exogenously determined rate of *pcross*. If a randomly paired set of the chromosomes is not chosen for crossover, they remain in the new set as is. However, if they are chosen for crossover, they enter this new set as a combination of different aspects of the successful past decisions. This occurs by choosing a random cut point g, which is an integer between *1* and string length l. The elements on one side of this cut point in a chromosome is exchanged with those of another. In the following example, with a random cut point $g=2$, and string length $l=7$ the two sets of chromosomes are

10:01101

01:11011

where the : represents the cut point. This yields the post-crossover chromosomes

10:11011

01:01101

which enters the new pool and is subject to the mutation operator.

The use of a crossover operator to model this part of human behaviour is attractive. However, it is omitted from this study on the basis that it may take the metaphor of learning too far. In particular, it does not give any merit to under what criteria the agent would choose to pair up part of old decisions to make new ones. In particular, why is the cut point a random integer rather than a choice for the agent? This crossover operator gives no consideration of how these rules may actually be matched up. As Brooks (2000) argues, it is the equivalent to the claim that tearing two old shopping lists in half, and taping them together would yield a better list of groceries than previously. It is thus a discretionary choice to omit this operator. This omission has little effect on the final results with the exception of omitting a behavioural metaphor. Although the crossover operator brings diversity to the model, the mutation operator substantiates the loss, with the exception of necessitating more iteration for convergence.

Of course, the use of the genetic algorithm is not without its criticisms. In particular, the way that the genetic algorithm has been modelled places strict bounds on the amount of information that an agent can store. The net entry or exit of rules or ideas in an agent's population of potential rules must always be zero. Furthermore, and particularly in the discussion of the crossover operator, the algorithm may overstep the accepted bounds of the learning metaphor for economic behaviour. In spite of these flaws

however, the genetic algorithm seems to represent human behaviour more realistically than traditional models that assume instantaneous optimization.

6 SIMULATIONS, RESULTS & INTERPRETATIONS

The simulations in this study were conducted for two reasons; to observe the convergence ability of the augmented genetic algorithm and to interpret the process by which it approaches optimal values. Hence, this section examines the technical aspects of the genetic algorithm results, the interpretation of the path, and any policy implications that it may entail.

Simulations in this study were conducted under the augmented genetic algorithm framework over a time period of 2000 iterations under various mutation rates. The exogenous tax rate was set at 0.3 for all iterations while the cost of the audit was maintained at p^2. In each case, the enforcement agency converged to a distinct enforcement policy with sanctions $S = 1$ and audit rates $p = 0.2617$, after an initially volatile period of learning. These enforcement policies conformed relatively well to Becker's theoretically implied results. In particular, the sanction level converged to its maximum attainable values. This implies that the authorities confiscate the entire evaded wealth of the agent. However, if this restriction in the simulation were relaxed in order to have greater sanctions, it would not change the qualitative implication of this result. It is still the maximum to impose on an evading agent. In contrast, the audit rates were substantially less in relative terms, which are attributed to the cost of increasing detection rates.

Two sets of simulations were conducted under different environments. First, sanctions were maintained at their maximum attainable levels, simplistically modelling

an environment where the decisions regarding punishments and detections were made by independent entities. The judicial system would impose the sanctions, while the enforcement agency's job would be only to detect those that broke the law. The second environment follows that which is described in the majority of this paper; a single agency whose capacities including enforcing the law through detection and administering sanctions to those who break it. The results of these simulations are as follows:

Table 1: Result of the Simulations conducted in *Environment 1*

time period	*mutation rate*	*0.003*	*0.01*	*0.04*
	audit	0.2617	0.2617	0.2617
t=1-2000	σ_a	0.0158	0.009	0.0108
t=1-250	σ_a	0.0416	0.0254	0.0305
	revenue	0.2114	0.2114	0.2114
t=1-2000	σ_r	0.0046	0.0043	0.0044
t=1-250	σ_r	0.0126	0.012	0.0124

Table 2: Result of the Simulations conducted in *Environment 2*

time period	*mutation rate*	*0.003*	*0.01*	*0.04*
	sanction	1	1	1
t=1-2000	σ_S	0.0208	0.0191	0.0156
t=1-250	σ_S	0.0517	0.05	0.0428
	audit	0.2617	0.2617	0.2617
t=1-2000	σ_a	0.0171	0.0169	0.0098
t=1-250	σ_a	0.0462	0.0458	0.0275
	revenue	0.2114	0.2114	0.2114
t=1-2000	σ_r	0.0073	0.0069	0.0054
t=1-250	σ_r	0.0192	0.0185	0.0151

First, on the basis of the time series plots (refer to figure sets 1 and 2 in Appendix 1), it should be apparent that convergence towards an optimal policy is not an immediate phenomenon. This is clear in any of the revenue (or fitness) plots. The tax revenue continuously fluctuates as it reacts to the policies learned by the enforcement agency during each iteration. However, there is a positive trend in the movement of the revenue levels which reflects the continual search by the enforcement agency for better performing policies. The path to optimality is not smooth though, which reflects the fact that mistakes are made, but reactions to reverse the problem are relatively quick The genetic algorithm moves gradually towards the convergence levels, and in cases such as the audit rates, it over and undershoots its final target. This captures the behavioural learning process that is central in this paper and seems realistic given that audit rates in one period may not be optimal in the next period if sanctions are also changing. This causes the behaviour of the evading taxpayer to change, which entails a further adjustment of the audit rates. If the enforcement agency was explicitly optimizing, there would be no over or undershooting. The genetic algorithm therefore resembles the mistakes being made in the process of policy implementation before full information exists. Furthermore, it seems realistic to believe that through trial and error, the differences between the implemented policies and their optimal values will become minimized over time (Brooks, 2000). This is clearly represented in the simulations, and is directly attributed to implementation of the election operator.

Furthermore, the learning agents decipher quite quickly that optimal levels of the costless sanctions are in fact maximal. However, there is a greater learning curve in the time it takes to learn the optimal audit rates. This of course is due to trade-off between the

amounts of additional revenue that can be generated by increases in the audit as opposed to how much of a cost increase it entails; this is very much like balancing the marginal revenues to the marginal costs of enforcement. One should not overemphasize the magnitudes of the standard deviations of the above table. In all cases it seems that higher levels of mutation entail lower standard deviations. However, this is in part due to the election operator that controls the deviation away from the optimal value as well as the quicker convergence rate of high mutation strings.

In all of the simulations there were no differences in the levels of convergence between the two environments. However, differences did manifest themselves in the speed of convergence. In the environment with an exogenously set sanction levels, convergence occurred quicker, and with much less deviation in the early simulations. However, the difference was negligible since the learning agent was able to decipher quickly that maximum levels of sanctions were optimal in the second environment. After reaching this level, the role of the genetic algorithm was to search for the best potential audit rates. However, since there was simultaneous learning occurring, the genetic algorithm was already on its way to finding optimal solutions, on the basis of the increasing values of the sanction levels. In some sense, simultaneous learning seems preferable. Otherwise, if more than a single learning agent was imposed, we would have sequential learning, where the enforcement agency would need to wait for instruction by the sanctioning committee who would also need to learn.

With respect to the convergence speed of the genetic algorithm, higher mutation rates were generally associated with quicker convergence. The reason being that by increasing the diversity in the string at a higher rate, the genetic algorithm can search for

substantially better rules with the mutation operator. This occurs without fear that rules that are substantially sub optimal will enter the population, which is a product of the election operator. If the election operator were not in place, the algorithm would still converge quicker with higher mutation rates. However, it would be characterized by large deviations away from the optimal position or `noisy convergence' since all strings whether beneficial or detrimental would be allowed into the population. The unattractive strings in this case could only be disposed of through the replication process.

In essence, what this implies is that the enforcement agency should grant substantial freedom of innovation to its departments. With greater emphasis on research and development, convergence upon optimal levels are quicker, hence this decreases much of the losses attributed from operating away from the optimal levels prior to convergence. Under a strict, and slow evolving regime, the agency can still reach the same convergence levels, however, they also must incur the extra cost of inefficient operation during the learning path.

It should be noted that the ability to converge was dependent on the minimal intelligence levels granted to the enforcement agency. In preliminary simulations, the enforcement agency lacked the ability to innovate only one of its tools, either sanctions or audit rates, while holding the other constant. The implications of this were clear; under different mutation rates, the enforcement agency settled or was getting "stuck" in different policies. For example, if sanctions reached optimal levels in one period, it was improbable that the detection rates in the next period could be improved through innovation without there also being a negative effect on sanctions. Hence, new rules would never be accepted, and through the replication process, a single rule would

emerge, which would in turn be self-fulfilling optimal values given that the taxpayer would in turn optimize to these choice values. In order to test this, simulations were conducted using less intelligent agents with shorter chromosomes of string length $l=7$, since there was a higher probability in this case that one string would be untouched by mutation while slight changes in the other would be possible. This did indeed improve the convergence ability of the simulations; however, to decrease string length entailed a trade-off of precision in results. Although the actual converged values in this paper are not of primary importance since the model has not been calibrated to real data, in other applications of the genetic algorithm it would be, hence a logical rethinking of the agency intelligence was appropriate.

It should be noted that in some instances, a genetic algorithm would not converge even under the intelligence specified in this paper due to its explicit design. Examining equation (3) in the application of the genetic algorithm, in any given chromosome, greater importance is placed on the latter bits in the string as opposed to the initial values. If the structure of these latter bits enables the chromosome to decode to values that are close to optimal values, this chromosome may be "stuck" in a search space. Although the mutation operator would work to increase the diversity in the string, if bit values at the beginning of chromosome switch to values that yield better performance *ceteris paribus*, it means little if the latter bits mutate to sub-optimal levels.

This problem would be more apparent with the use of extremely low or extremely high mutation rates; extremely high mutation means that almost all of the bits in the strings would switch, while too low of a mutation rate may entail convergence over too long of a period. The election operator would ensure that these mutated chromosomes are

not allowed into the general population. Hence, given a population of chromosomes of a long string length sub-optimal convergence is possible.

6.1 Transitions from different Tax Rates

Given the above analysis, the final set of simulations was to test how well the enforcement agency could adjust its policies in light of a change in the tax rate. The purpose of this experiment was to show the necessity of the mutation or innovation operator in the learning process. As expected, in light of the literature on mutation rates in the genetic algorithm framework, the highly innovative agents can search out a new equilibrium much quicker than the slow learning agents. This is clearly depicted in Figures 3.1-3.2 with mutation rates of 0.1 in contrast to Figures 4.1- 4.2 with lower innovation rates of 0.003. In figure 3.2, it takes approximately 300 iterations to converge to the new equilibrium, however, under the low mutation rate this takes nearly 700 iterations. Furthermore, it should be noted that under high mutation rates, the genetic algorithm reaches the general area of optimality sooner, and due to the election operator and replication sees much smaller variation. Sanctions were not an issue of importance in these simulations, since maximum sanctions learned under the first tax rate are maintained at the maximum levels under the other. In essence, the enforcement agency learns in a quicker time interval that taxpaying agents do in fact change their behaviour when exogenous variables such as the tax rates change.

Once the tax rate has changed, due to their high levels of experimentation with different rule sets, the highly innovative enforcement agency can quickly find better rules that work in this new environment. However, the agency that is stuck in the same mode of thinking will still move towards a new policy regime, albeit at a slower and more

inefficient rate. What occurs is that when the tax rate changes, these slow learning agents experiment with small changes to their rule set. These slightly better performing rules are replicated throughout the population, subject to future experimentation. Slowly but surely, the agency will learn policies that give maximum attainable tax revenues.

Again, this sheds more light on how agents may actually behave when changes occur to their environment, rather than taking for granted the basic assumption of full rationality. Furthermore, what can be taken from this is that in order for the enforcement agency to react in a more efficient manner to exogenous changes in its environment, it should be granted a high level of innovative freedom. Without innovation, if agents have purely adaptive reasoning (learning purely by replication), they will never be able reach their true potential in an ever-changing environment.

7 CONCLUSION

The primary emphasis of this paper was to showcase the idea that agent learning might be a more realistic representation of agent behaviour than the assumption of full rationality. In order to model such behaviour in an economic framework, the genetic algorithm has been employed as the tool of analysis. It has been shown that under this method of learning and assuming a minimal level of intelligence, the enforcement agency can learn policies that will optimize their revenue functions through a process of imitation and experimentation. Although many of the drawbacks of the genetic algorithm framework have been outlined in the body of this paper, there should be little doubt that it depicts some of the flair of humanity that the traditional analysis sorely misses.

If optimization is a process rather than an instantaneous reaction, the genetic algorithm gives us insight on how we can minimize the revenue losses incurred in the searching for optimal values in an economy with tax evasion. In particular, in the environment depicted in this paper, the implementation of higher rates innovation on the part of the enforcement agency arguably yields better performance. This lessens the amount by which there is an under-provision of public goods, which entails higher social utility. Finally, much of the power of the genetic algorithm can be seen in how agents can react to changes in their exogenous environment. Under full rationality, there is seldom an explanation for how we reach new equilibriums. In contrast, in a genetic algorithm environment, the question pertaining to how we get to this point is of utmost importance. The answer is through learning.

Admittedly, there are quite a few limitations in this paper that should be dealt with in future research. Clearly, the model used in this paper is the most basic in the evasion literature, and does not incorporate many of the intricacies explained in the literature survey. As a result, it may not capture the empirical properties of the economics of evasion. To correct for this entail a much richer model that can incorporate the stigma effects of evasion accurately. Furthermore, the genetic algorithm itself may be too simple of a basis to model learning, when many other sophisticated forms of artificial intelligence are available. Finally, in order to capture further realism, the next step is to implement a learning taxpayer as opposed to our assumed optimizing agent. This allows modelling the full learning environment, and is a path for future research.

APPENDIX 1

ENVIRONMENT 1

Figure 1.1. 1 Time series plot for audit t = 1:2000 *pmut*=0.003

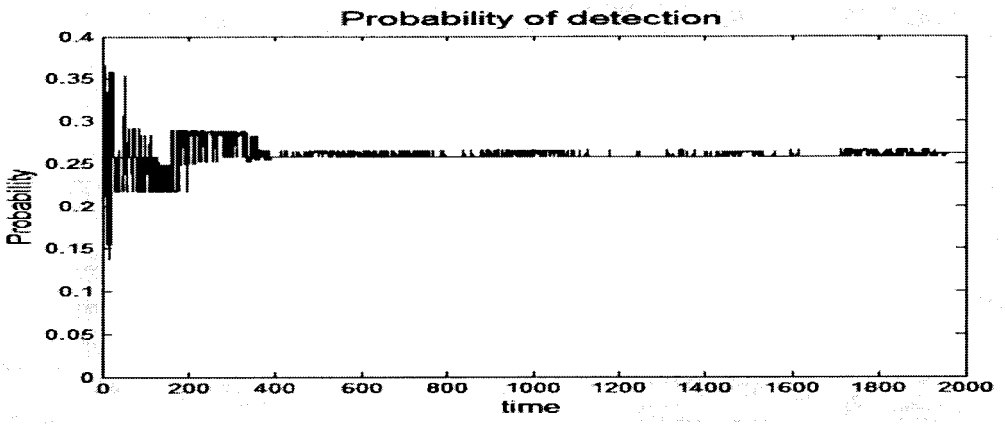


Figure 1.1. 2 Time series plot for revenue t = 1:2000 *pmut*=0.003

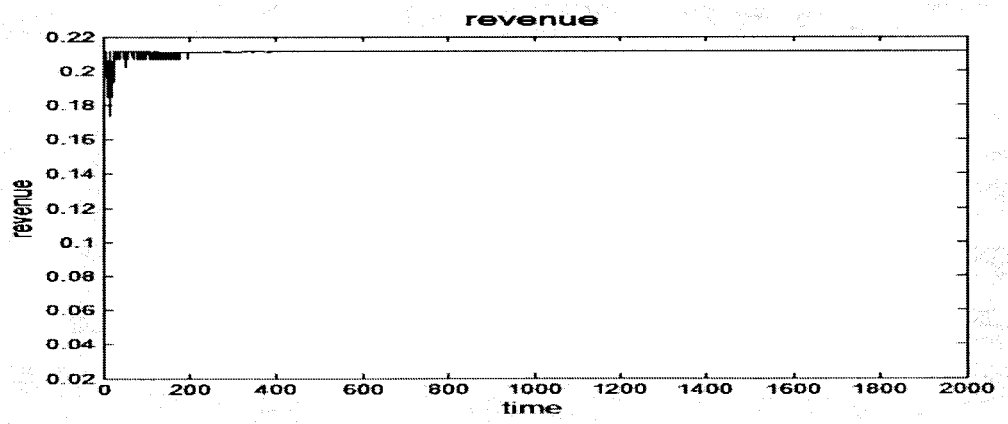


Figure 1.1. 3 **Time series plot for audit t = 1:250** ***pmut*=0.003**

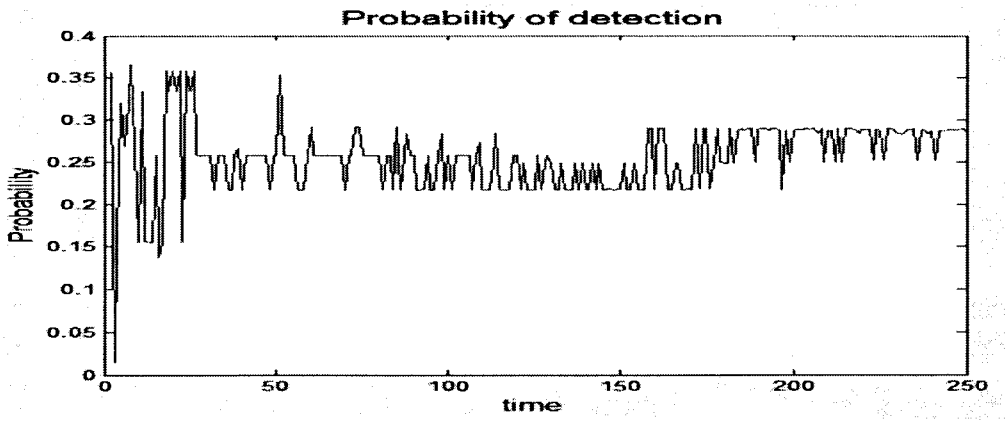


Figure 1.1. 4 **Time series plot for revenue** **t = 1:250** ***pmut*=0.003**

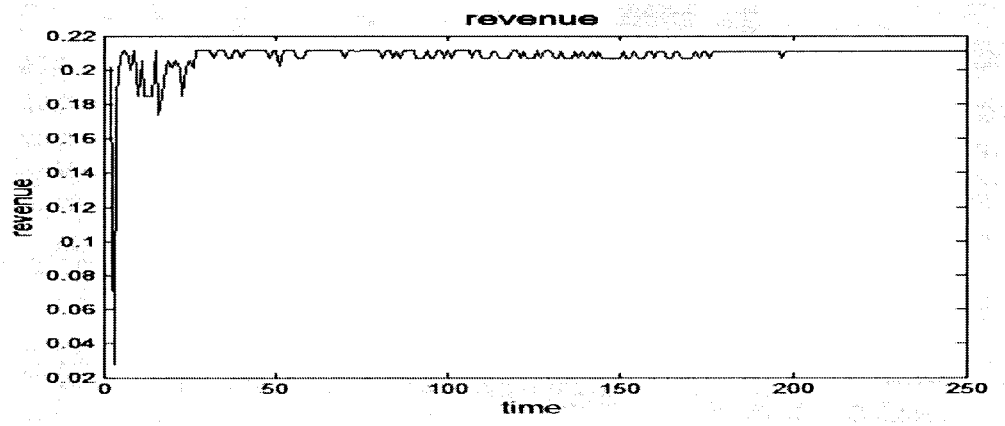


Figure 1.2. 1 **Time series plot for audit t = 1:2000** ***pmut*=0.01**

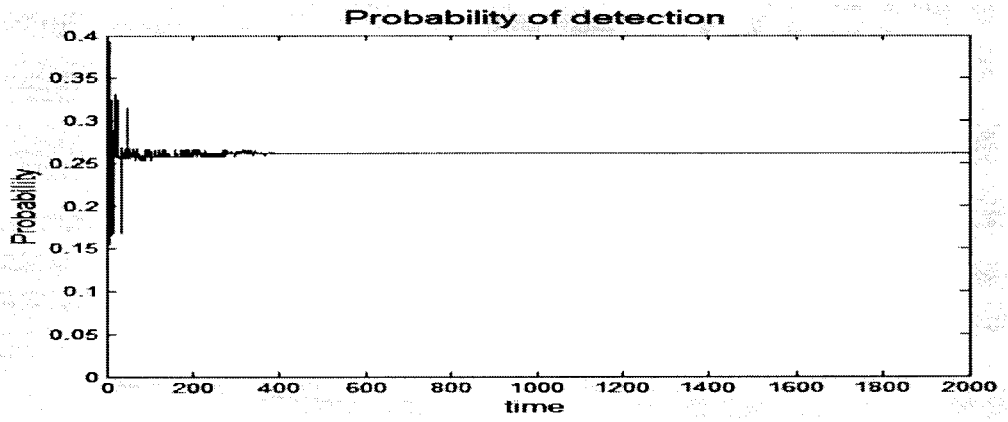


Figure 1.2. 2 Time series plot for revenue t = 1:2000 *pmut*=0.01

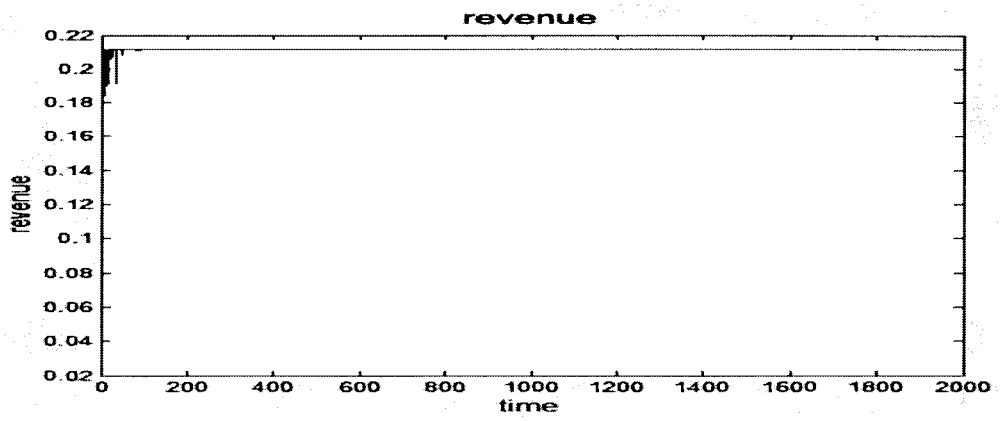


Figure 1.2. 3 Time series plot for audit t = 1:250 *pmut*=0.01

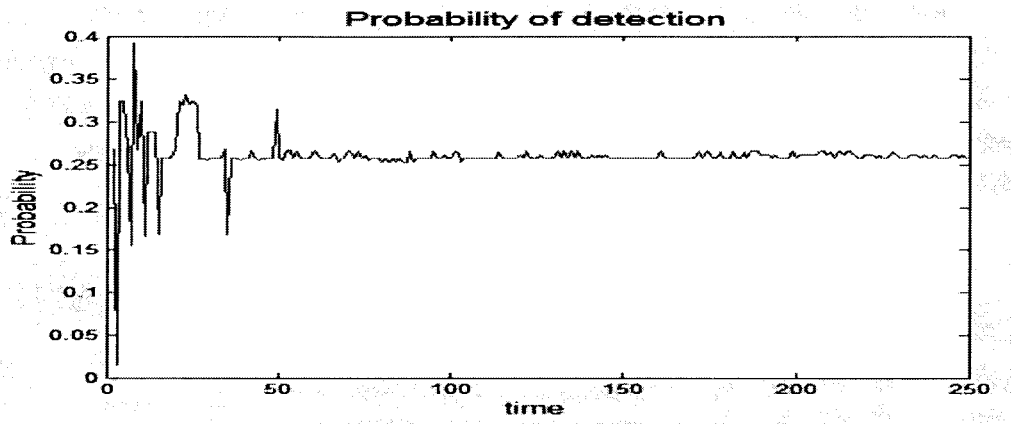


Figure 1.2. 4 Time series plot for revenue t = 1:250 *pmut*=0.01

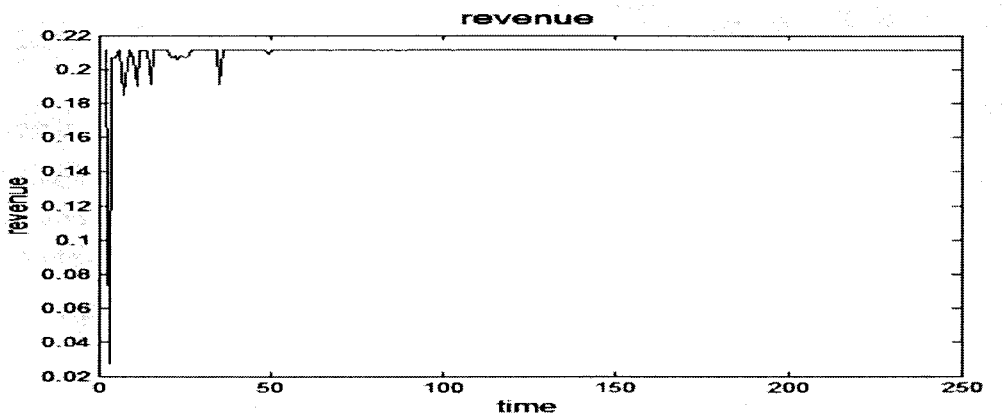


Figure 1.3. 1 Time series plot for audit = 1:2000 *pmut*=0.04

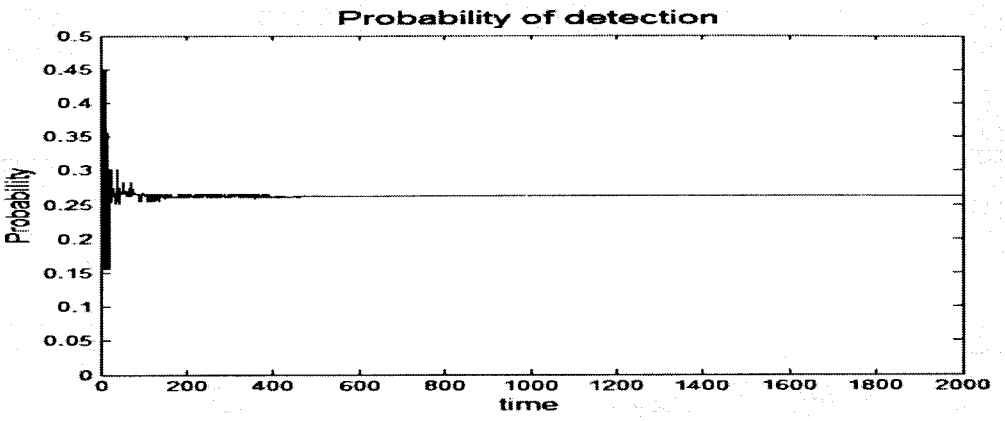


Figure 1.3. 2 Time series plot for audit t = 1:2000 *pmut*=0.04

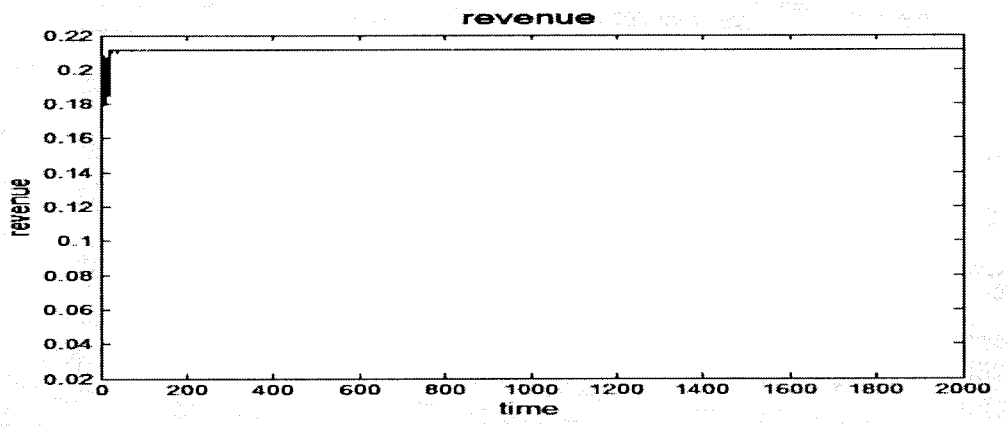


Figure 1.3. 3 Time series plot for audit t = 1:250 *pmut*=0.04

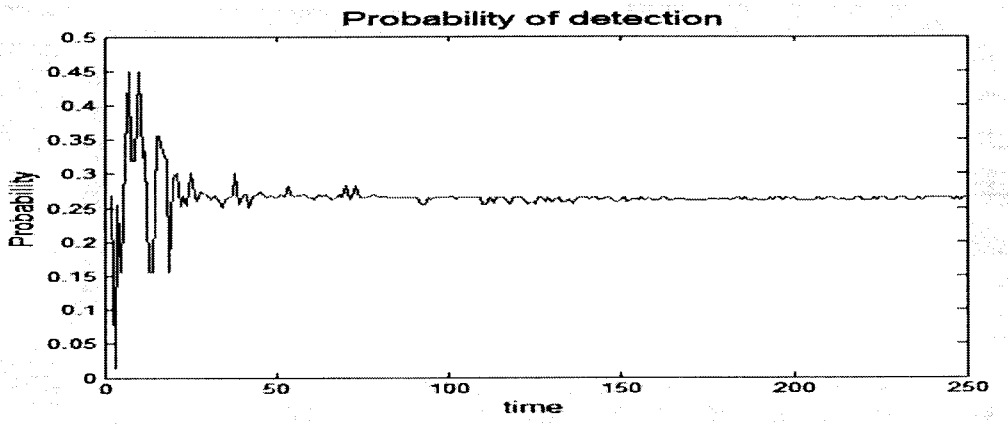
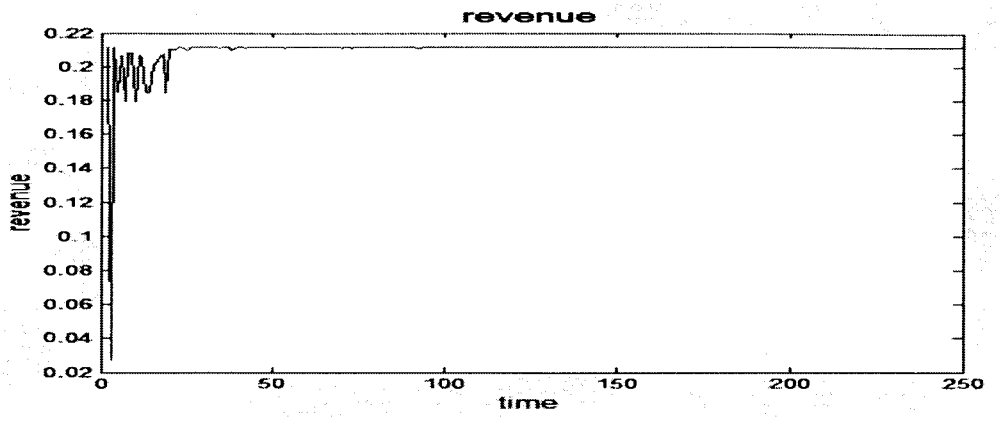


Figure 1.3. 4 **Time series plot for revenue** **t = 1:250** ***pmut*=0.04**



ENVIRONMENT 2

Figure 2.1. 1 **Time series plot for sanction levels** **t=1:2000** ***pmut* = 0.003**

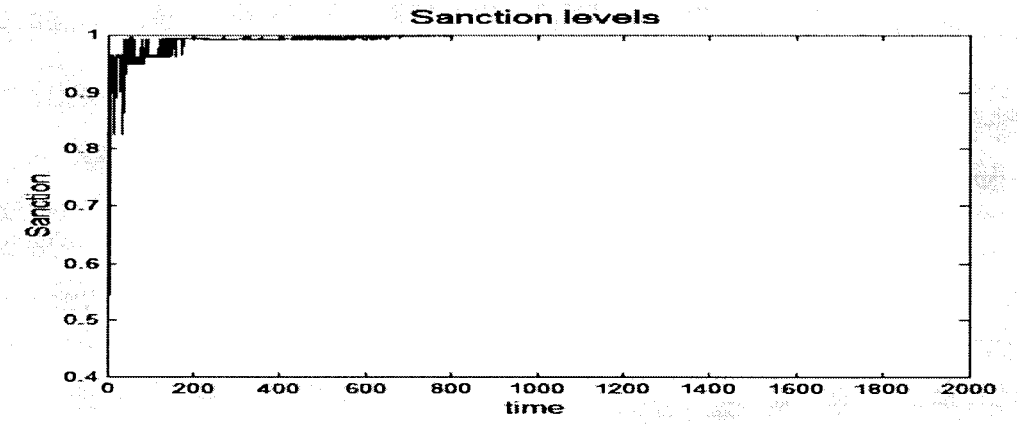


Figure 2.1. 2 **Time series plot for audit** **t=1:2000** ***pmut* = 0.003**

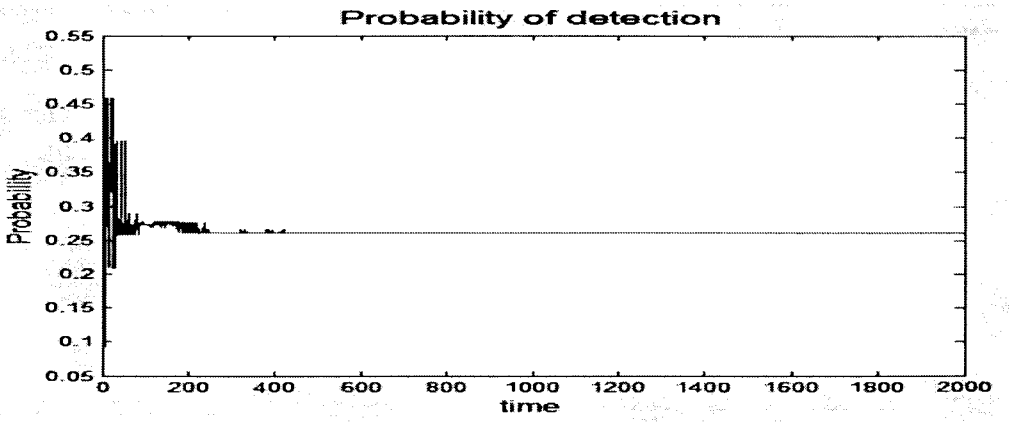


Figure 2.1. 3 **Time series plot for sanction levels** **t=1:250** ***pmut* = 0.003**

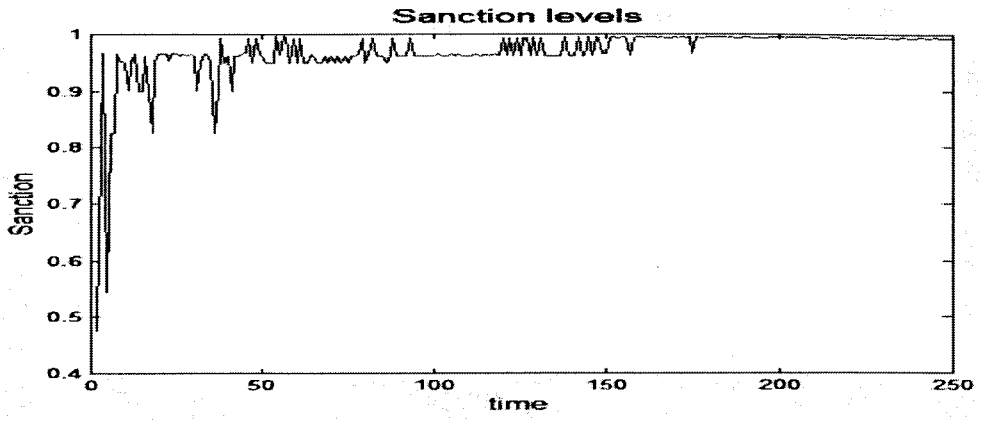


Figure 2.1. 4 **Time series plot for audit** **t=1:250** ***pmut* = 0.003**

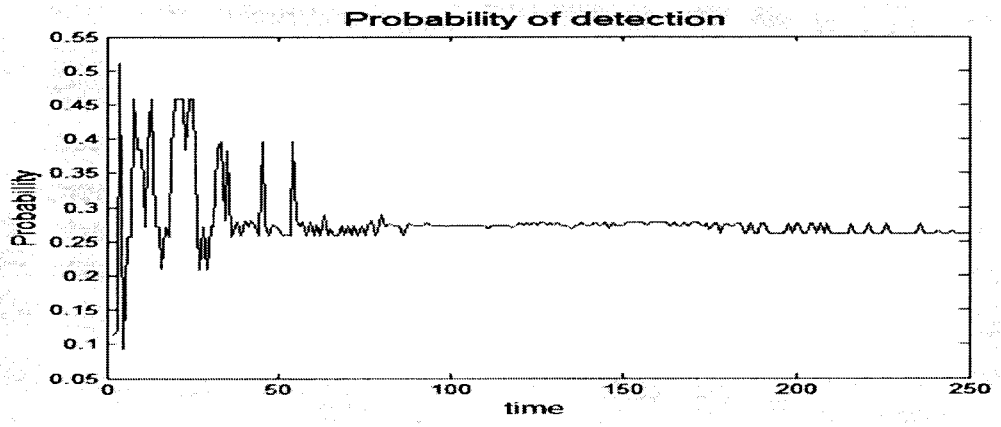


Figure 2.1. 5 **Time series plot for revenue** **t=1:250** ***pmut* = 0.003**

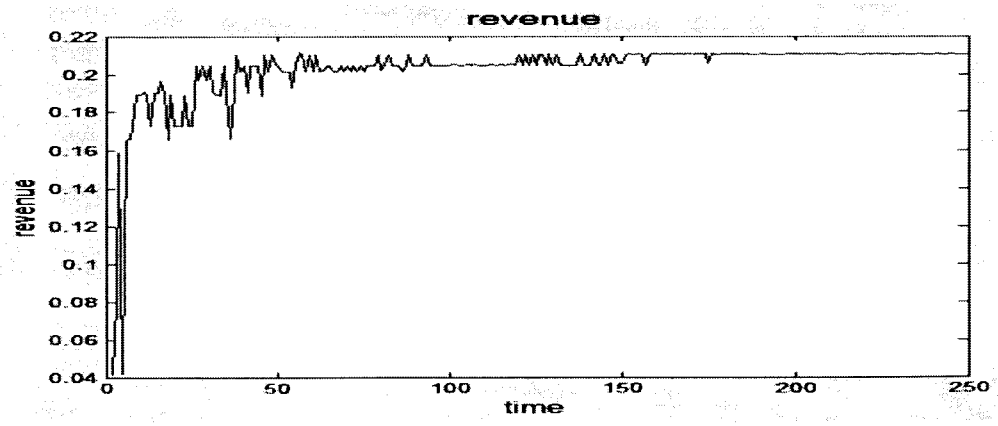


Figure 2.2. 1 **Time series plot for sanction levels** **t=1:2000** ***pmut* = 0.01**

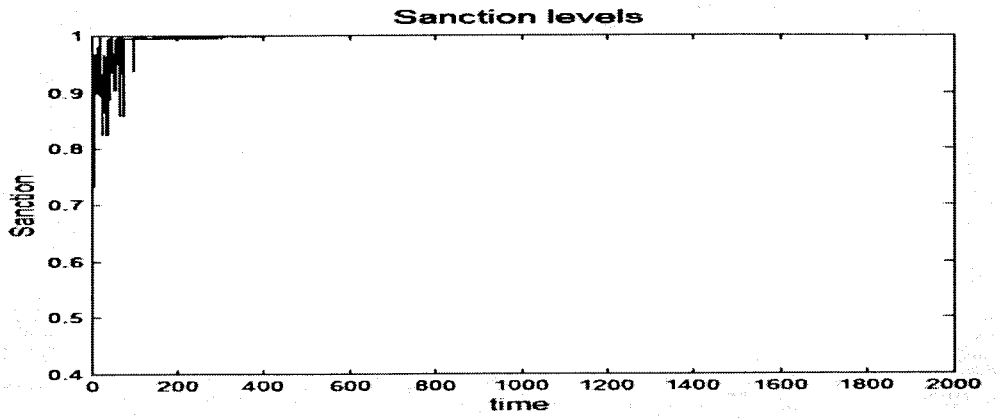


Figure 2.2. 2 **Time series plot for audit** **t=1:2000** ***pmut* = 0.01**

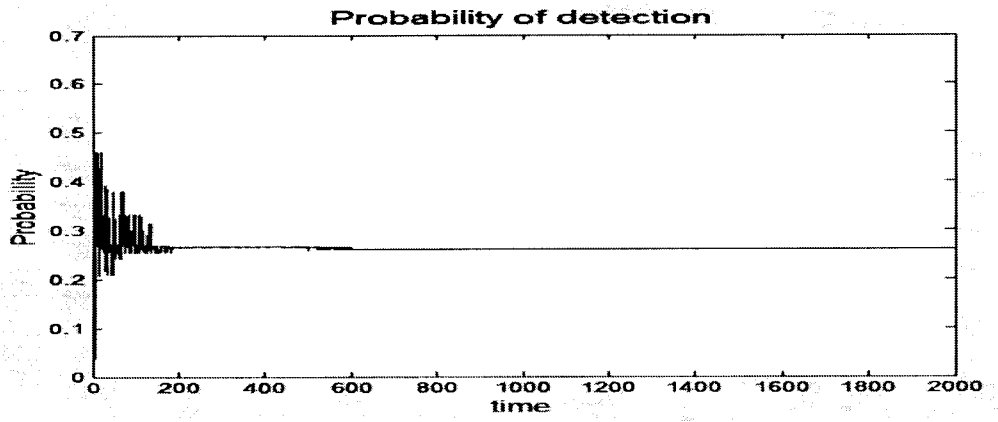


Figure 2.2. 3 **Time series plot for revenue** **t=1:2000** ***pmut* = 0.01**

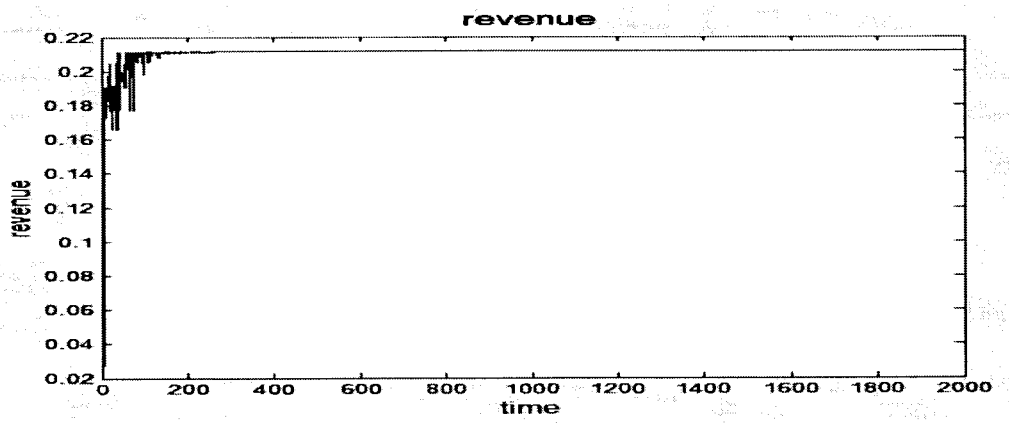


Figure 2.2. 4 **Time series plot for sanction levels** **t=1:250** ***pmut* = 0.01**

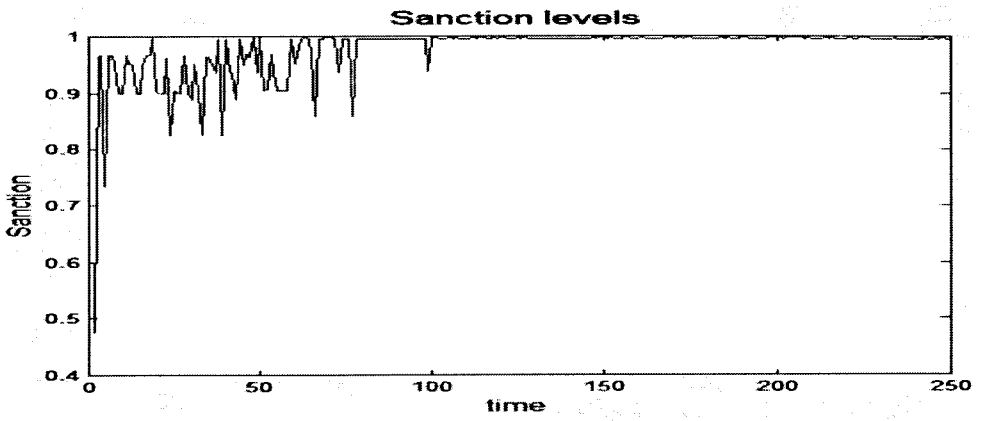


Figure 2.2. 5 **Time series plot for audit** **t=1:250** ***pmut* = 0.01**

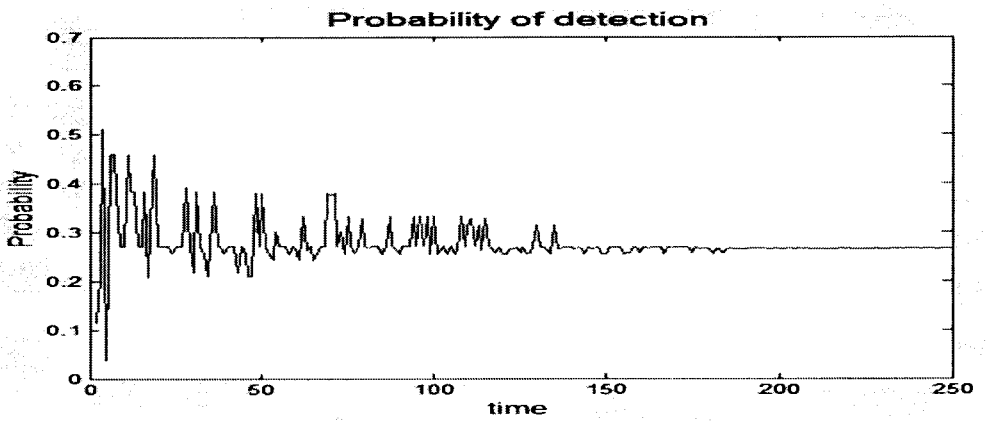


Figure 2.2. 6 **Time series plot for revenue** **t=1:250** ***pmut* = 0.01**

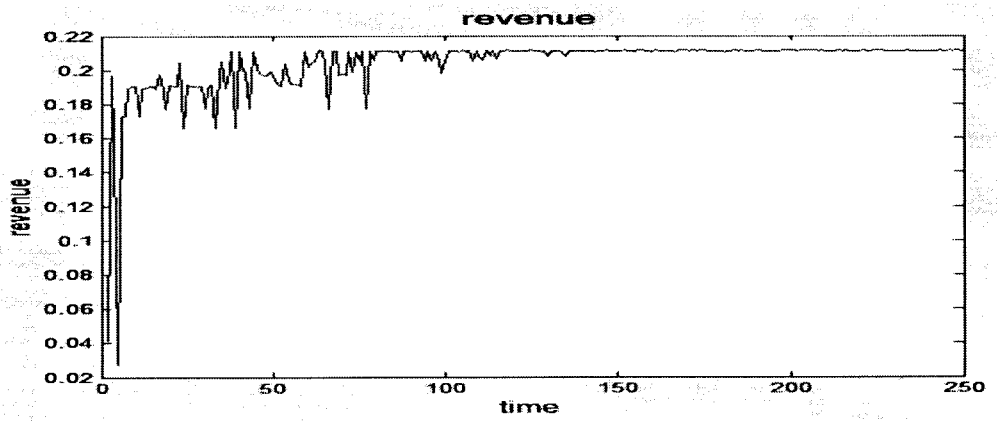


Figure 2.3. 1 Time series plot for sanction levels t=1:2000 *pmut* = 0.04

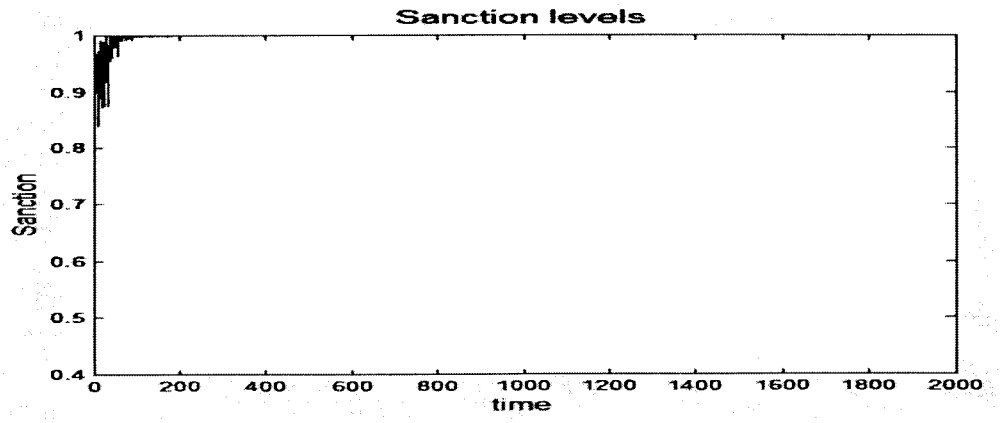


Figure 2.3. 2 Time series plot for audit t=1:2000 *pmut* = 0.04

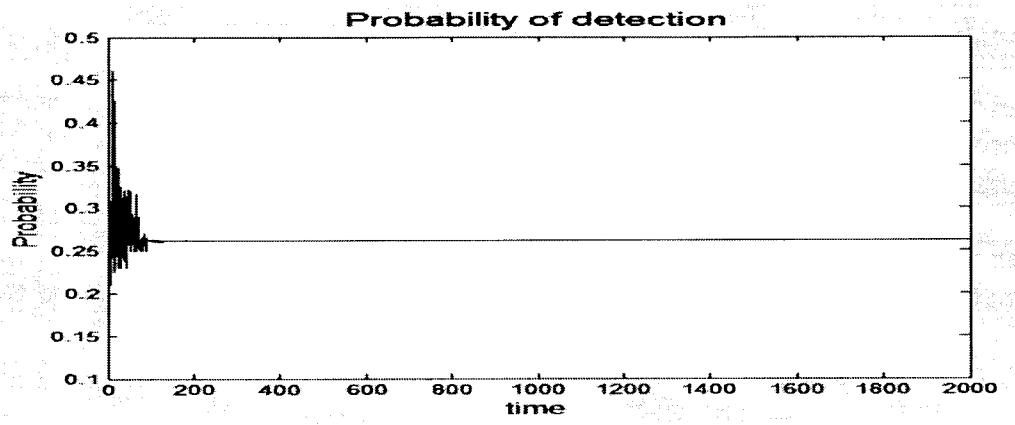


Figure 2.3. 3 Time series plot for revenue t=1:2000 *pmut* = 0.04

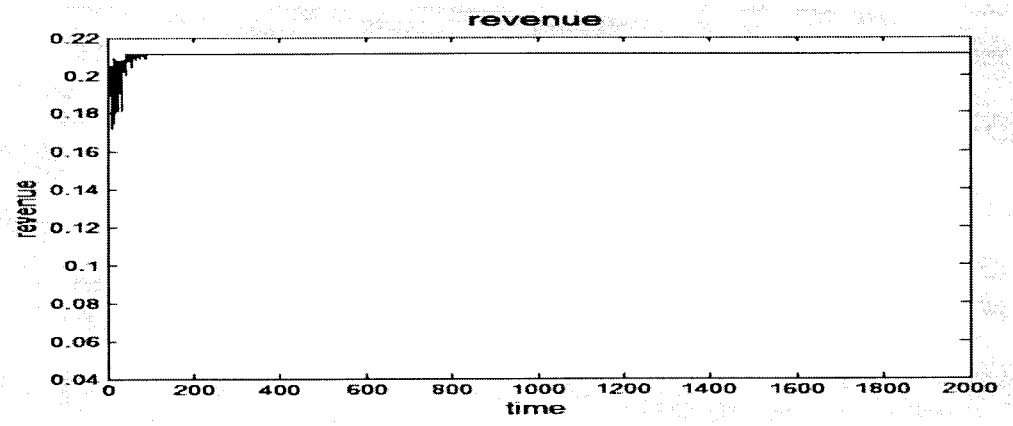


Figure 2.3. 4 **Time series plot for sanction levels** **t=1:250** ***pmut* = 0.04**

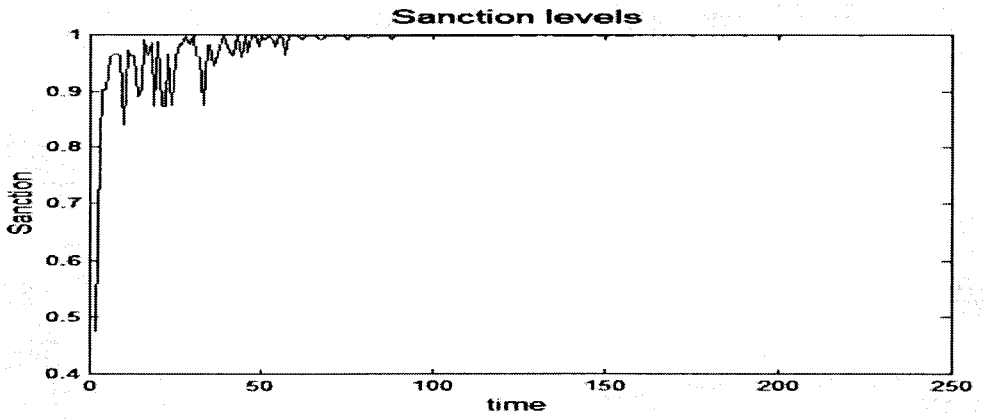


Figure 2.3. 5 **Time series plot for audit** **t=1:250** ***pmut* = 0.04**

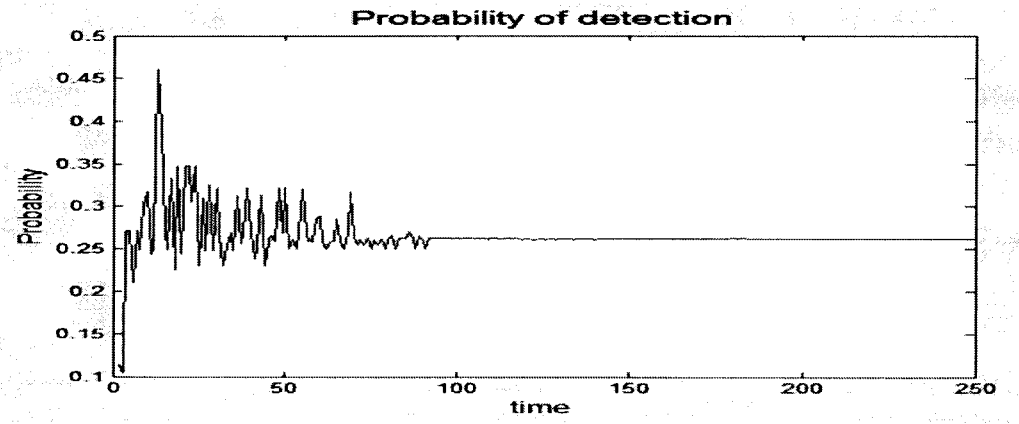


Figure 2.3. 6 **Time series plot for revenue** **t=1:250** ***pmut* = 0.04**

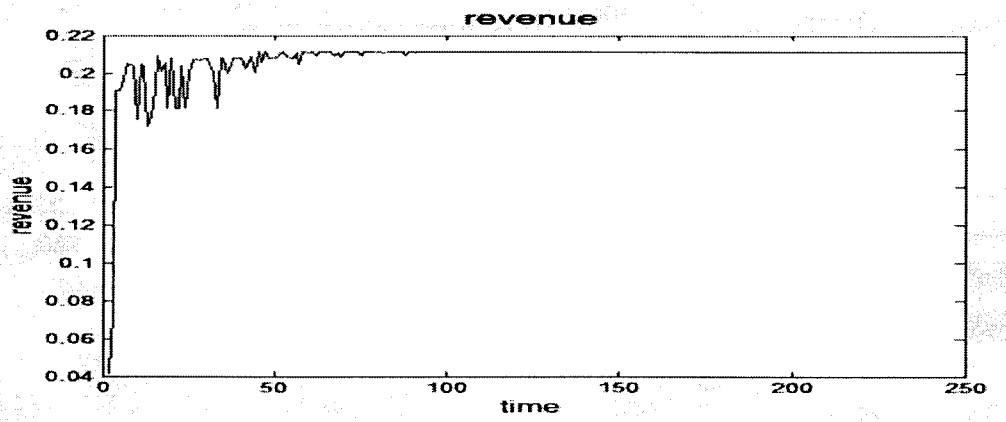
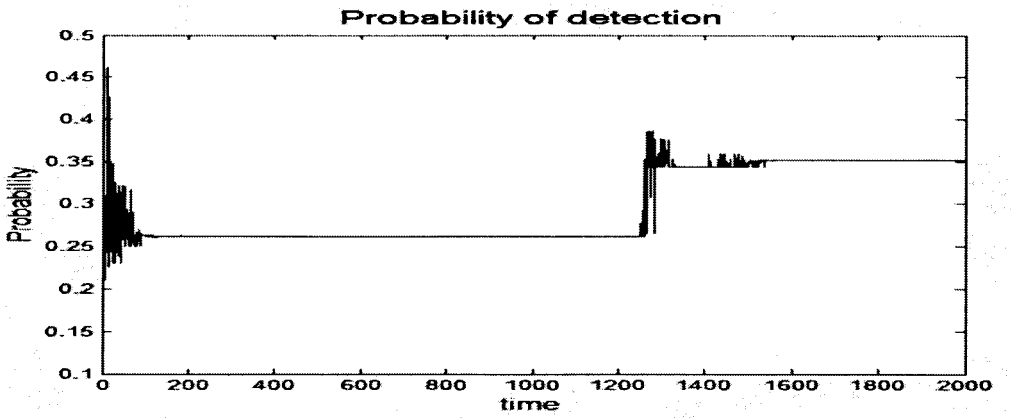
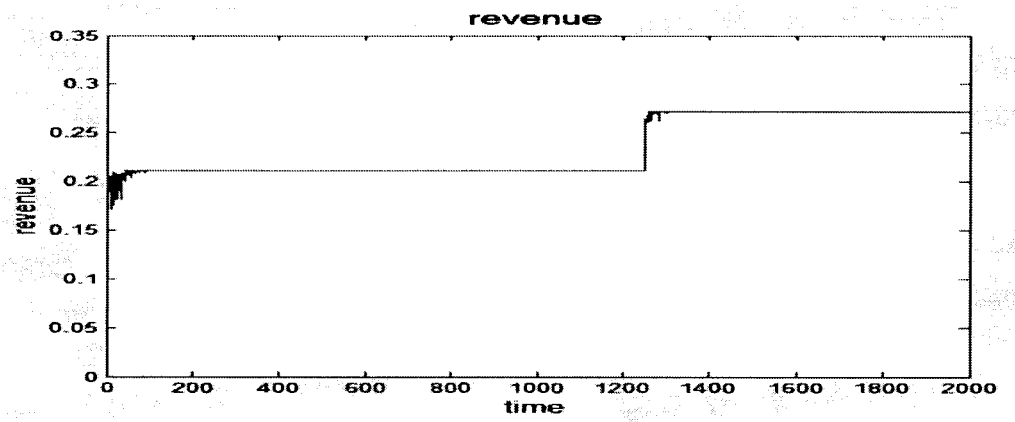


Figure 3.1. 1 **Time series plot for audit** **t=1:2000** ***pmut*** **= 0.1**



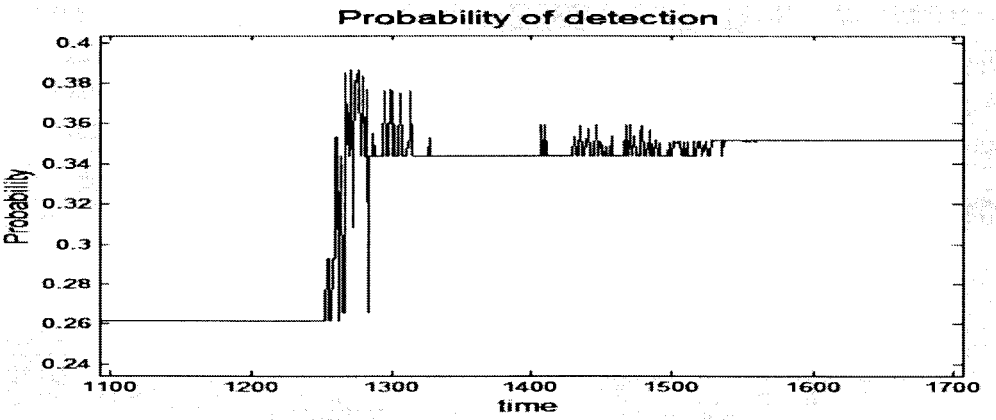
t<1250 τ =0.3 t<1250 τ =0.6

Figure 3.1. 2 **Time series plot for revenue** **t=1:2000** ***pmut*** **= 0.1**



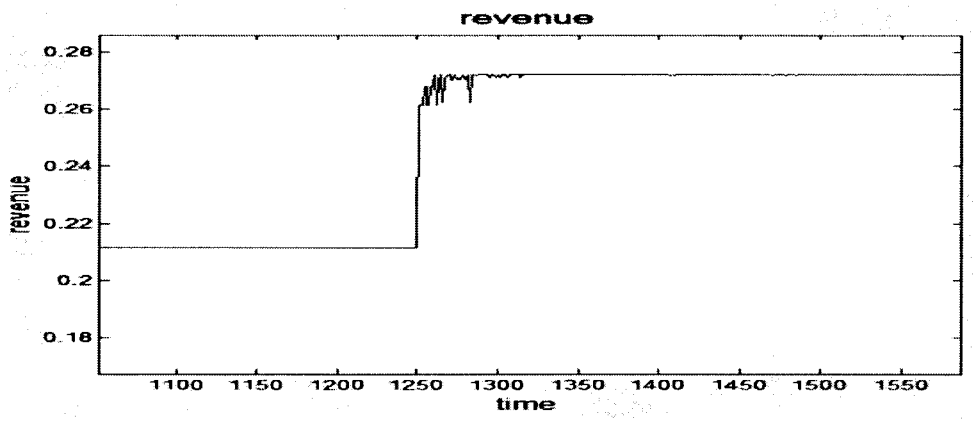
t<1250 τ =0.3 t<1250 τ =0.6

Figure 3.1. 3 **Time series plot for audit** **t=1080:1720** ***pmut*** **= 0.1**



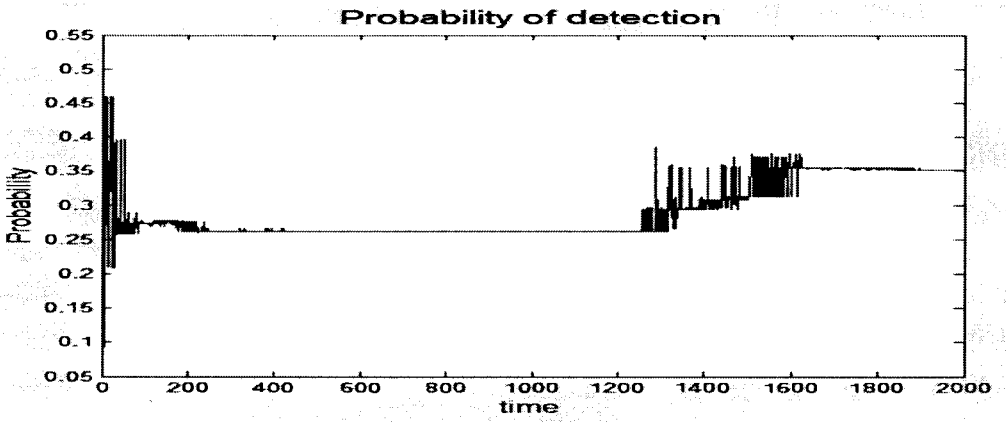
t<1250 τ =0.3 t<1250 τ =0.6

Figure 3.1. 4 Time series plot for revenue t= 1050:1600 *pmut* = 0.1



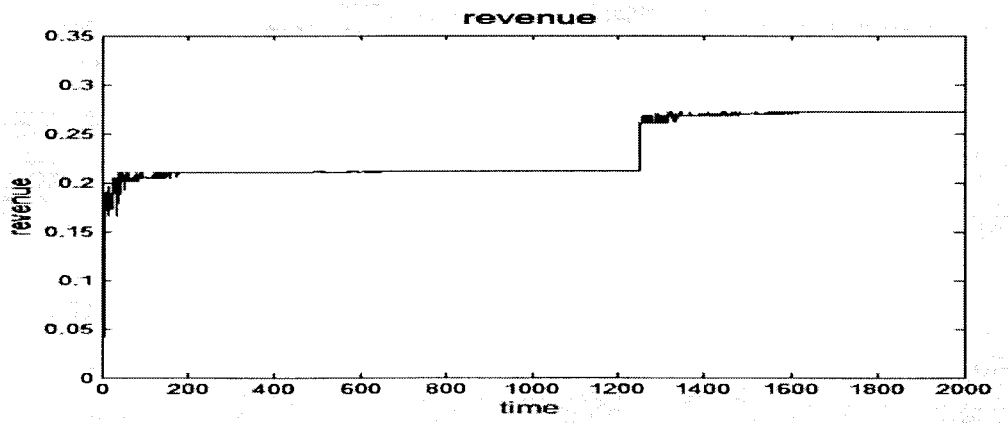
t<1250 τ =0.3t<1250 τ =0.6

Figure 4.1. 1 Time series plot for audit t=1:2000 *pmut* = 0.003



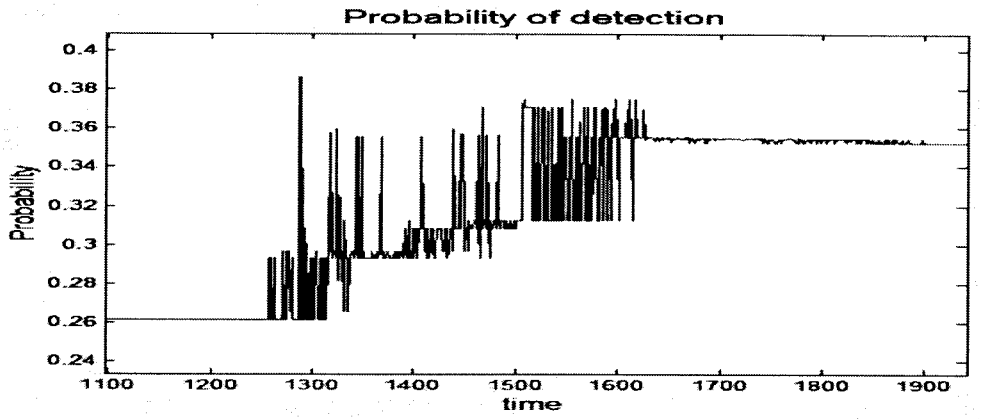
t<1250 τ =0.3t<1250 τ =0.6

Figure 4.1. 2 Time series plot for revenue t=1:2000 *pmut* = 0.003



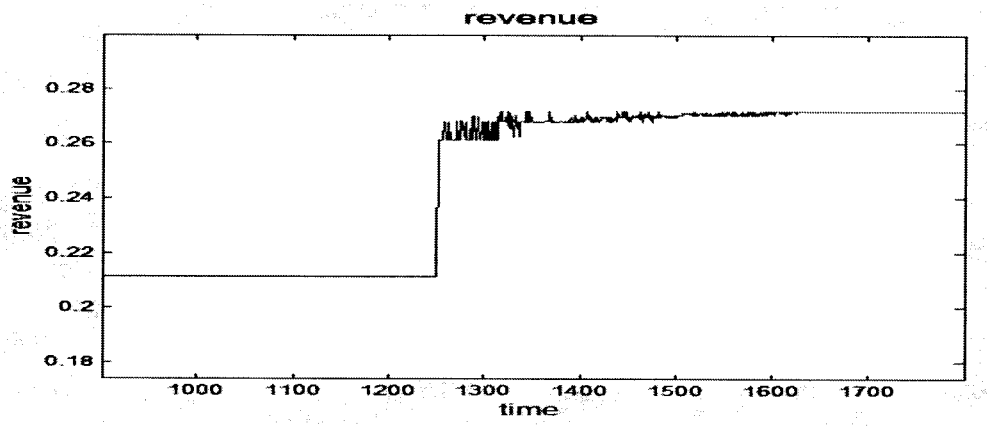
t<1250 τ =0.3t<1250 τ =0.6

Figure 4.1. 3 Time series plot for auditt=1100:1980 *pmut* = 0.003



t<1250 τ =0.3t<1250 τ =0.6

Figure 4.1. 4 Time series plot for revenue t=900:1810 *pmut* = 0.003



t<1250 τ =0.3t<1250 τ =0.6

APPENDIX 2

Matlab Code for the Simulations in environments 1 & 2. Section 6 in the paper.

Environment 1

```
% PREPARED BY BRYAN VINCENT YU
% SIMON FRASER UNIVERSITY

% FOC that the agents will abide by follow the form
        % X = (1-(P*(S-t)/t*(1-P))-S)*M/((1-P*(S-t)/t*(1-P))*t-S)
% The objective function for the government
        % R = t*X +P*S*(M-X)-[P*C]^2\

% parameter values
x = 1;          % This is the initial random reported income
tax = 0.3;      % Setting the tax rate constant across all periods (assumed dictated by govt)
d = 2;          % this is the place for the decoded string (normalized)
M = 1;          % Total income is normalized to one

% GA parameter values

pmut = 0.003;       % rate of mutation.
n = 100;            % number of strings/divide by two to find the number of potential rules that the government has in its repertoire.
k = 30;             % string length.
T = 2000;           % number of simulation periods.
seed = 0;           % seed value for random number generator.
election = 1;       % FLOW CONTROL. SET TO "1" FOR ELECTION, "0" FOR BASIC.

% Initialize random number generators

%rand('state', seed);

% create the vectors
population_p= zeros(n,k+d);    % population for the audit
population_s = zeros(n,k+d);   % population for the sanction
p_audit = zeros(1,n);
sanction = zeros(1,n);
revenue =  zeros(1,n);
pot_revenue = zeros(1,n);

replicate = zeros(n,k+d);                   % replicates hold cell
matrix_hold = zeros(4,(2*k+1));             % used in tournament and election
ww = zeros(4,(2*k+1));                      % used in tournament and election


% time vectors
rep_inc = zeros (1,T);
prob_audit = zeros(1,T);
Sanc = zeros(1,T);
revenue_time = zeros(1,T);

% (1e). Initialization of rules.

for i = 1:n
   for j = 1:k
      if (rand > 0.5)
         population_p(i,j) = 1;
      else
         population_p(i,j) = 0;
      end
```

```
        population_s(i,j) = 1;
    end
end

% (1f). Calculate normalization parameter, 'K'.

K_bar = 0;
for j = 1:k
    K_bar = K_bar + 2^(j-1);
end
K_bar = K_bar;

for t=1
    rep_inc(1,t) = x;
end
for t = 2:T
Time = t

% decode initially for the probability of audit (note: this is less than 1)
  for i = 1:n
      sum = 0;
      for j = 1:k
          sum = sum + population_p(i,j)*2^(j-1);
      end
      p_audit(1,i)= sum/K_bar;
      population_p(i,k+1) = sum/K_bar;
  end
 % sanction
  for i = 1:n
      sum = 0;
      for j = 1:k
          sum = sum + population_s(i,j)*2^(j-1);
      end
      sanction(1,i)= sum/K_bar;
      population_s(i,k+1) = sum/K_bar;
  end
  % from this the fitness of the strings can be deciphered.
  for i=1:n
      revenue(1,i) = tax*rep_inc(1,t-1)+p_audit(1,i)*sanction(1,i)*(M-rep_inc(1,t-1))-p_audit(1,i)^2;
      if revenue(1,i) < 0
          revenue(1,i)=0;
      end
  end
% put in the fitness values
  for i=1:n
      population_p(i,k+d) = revenue(1,i);
      population_s(i,k+d) = revenue(1,i);
  end

  % need to do the tournament selection for the strings
  first = zeros(n,k+d);
  second = zeros(n,k+d);


  for i=1:n
      for j = 1:k+d
          first(i,j) = population_p(i,j);
          second(i,j) = population_s(i,j);
      end
  end

  sum=0;
  for i=1:n
      sum = sum + first(i,k+d);
  end

  % find the relative fitness for this tournament cell
  relative = zeros(n,1);

  for i=1:n
```

```
   relative(i,1) = first(i,k+d)/sum;
end

for i = 1:n
   count = 0;
   click = 0;
   stop = rand;
   while (count<=stop)
      click = click + 1;
      count = count + relative(click,1);
   end
   for j = 1:k+d
      replicate_p(i,j) = first(click,j);
      replicate_s(i,j) = second(click,j);
   end
end

% this is the replicate cell based on the roulette wheel, now we need to change it into long column

% after having this tournament_cell can now make the replicates
% ok the replicates are made...

% mutation operators
if election == 0
   for i= 1:n
      for j=1:k
         if (rand < pmut);
            if  replicate_p(i,j) == 1;
               replicate_p(i,j) = 0;
            else
               replicate_p(i,j) = 1;
            end
         end
      end
   end
end % THIS END THE ELECTION == 0
if election == 1
   election_hold_p = zeros(n,k+d);
   election_hold_s = zeros(n,k+d);
   for i=1:n
      for j=1:k
         election_hold_p(i,j) = replicate_p(i,j);
         election_hold_s(i,j) = replicate_s(i,j);
      end
   end
% mutation operator with the election operator

   for i = 1:n
      for j=1:k
         if (rand < pmut);
            if election_hold_p(i,j) == 1;
               election_hold_p(i,j) = 0;
            else
               election_hold_p(i,j) = 1;
            end
         end
      end
   end
   % given this election hold cell, the potential fitness of the strings can be looked at
   % going to need to decode these strings as well and use the past values of x observed in market as starting poin
   for i = 1:n
      sum = 0;
      for j = 1:k
         sum = sum + election_hold_p(i,j)*2^(j-1);
      end
      election_hold_p(i,k+1)= sum/K_bar;

      sum = 0;
      for j = 1:k
         sum = sum + election_hold_s(i,j)*2^(j-1);
```

```
        end
        election_hold_s(i,k+1)= (sum/K_bar);
      end
      % now need to check for the potential fitness of these new strings
      for i=1:n
        for j= 1:k
          matrix_hold(1,j)=replicate_p(i,j);
          matrix_hold(1,j+k)=replicate_s(i,j);
          matrix_hold(2,j)=replicate_p(i,j);
          matrix_hold(2,j+k)=election_hold_s(i,j);
          matrix_hold(3,j)=election_hold_p(i,j);
          matrix_hold(3,j+k)=replicate_s(i,j);
          matrix_hold(4,j)=election_hold_p(i,j);
          matrix_hold(4,j+k)=election_hold_s(i,j);
        end

        % this finds the relative fitness and imposes the election operator
        matrix_hold(1,2*k+1) = replicate_p(i,k+d);
        matrix_hold(2,2*k+1) = tax*rep_inc(1,t-1)+replicate_p(i,k+1)*election_hold_s(i,k+1)*(M-rep_inc(1,t-1))-
replicate_p(i,k+1)^2;

        matrix_hold(3,2*k+1) = tax*rep_inc(1,t-1)+election_hold_p(i,k+1)*replicate_s(i,k+1)*(M-rep_inc(1,t-1))-
election_hold_p(i,k+1)^2;

        matrix_hold(4,2*k+1) = tax*rep_inc(1,t-1)+election_hold_p(i,k+1)*election_hold_s(i,k+1)*(M-rep_inc(1,t-1))-
election_hold_p(i,k+1)^2;

        ww = sortrows(matrix_hold,2*k+1);
        for j=1:k
          replicate_p(i,j) = ww(4,j);
          replicate_s(i,j) = ww(4,j+k);
        end
        replicate_p(i,k+d) = ww(4,2*k+1);
        replicate_s(i,k+d) = ww(4,2*k+1);
        if replicate_p(i,k+d)<0;
          replicate_p(i,k+d) = 0;
        end
        if replicate_s(i,k+d)<0;
          replicate_s(i,k+d) = 0;
        end
      end
    end

    % this in theory should have created the new set of binary strings that we will base stuff on now use the roulette wheel in order to
choose one single string to use
    fitness = zeros(n,1);
    for i=1:n
      fitness(i,1) = replicate_p(i,k+d);
    end
    sum_fitness=0;
    for i=1:n
      sum_fitness= sum_fitness +fitness(i,1);
    end
    relative_fitness = zeros(n,1);
    % find the relative fitness
    for i=1:n
      relative_fitness(i,1) = fitness(i,1)/sum_fitness;
    end
    % at the end make sure to convert back

    % using this relative fitness try to implement the roulette wheel

    count=0;
    click=0;
    stop=rand;
      while (count<=stop)
        click=click+1;
        count= count + relative_fitness(click,1);
      end
    % We can now use this counter to pick the appropriate parameters for use in the next period
```

```
% first convert these replicates back into to actual population for use in next time period

 for i=1:n
    for j=1:k+d
       population_p(i,j) = replicate_p(i,j);
       population_s(i,j) = replicate_s(i,j);
    end
 end
 % create a new holding cell for the rule set that the agents will choose this time around
 pick_cell=zeros(2,k);
 for j=1:k+d
    pick_cell(1,j)= replicate_p(click,j);
    pick_cell(2,j)= replicate_s(click,j);
 end

 % given this, decode for actual probability of audit and the sanction that is learned from the
 % constant tax rates (note that the odd number here are the probability of audit while the sanction are even numbered rows

 % need to decode these to find the final government choices
 %first decode for the tax levels
prob_audit(1,t) = pick_cell(1,k+1);
P = prob_audit(1,t);


        Sanc(1,t)= pick_cell(2,k+1);
S = Sanc(1,t);

% decode for the actual reported income that maximizing agents would choose in this case
rep = (1-(P*(S-tax)/tax*(1-P))-S)*M/((tax-P*(S-tax)/(1-P))-S);
if rep > 1
   rep = 1;
end
if rep < 0
   rep = 0;
end
rep_inc(1,t) = rep;
% calculate the actual revenue that will prevail
revenue_time(1,t) = tax*rep_inc(1,t)+prob_audit(1,t)*Sanc(1,t)*(M-rep_inc(1,t))-prob_audit(1,t)^2;
end

figure(1)
plot(2:T, Sanc(2:T));
xlabel('time', 'fontsize',12);
ylabel('Sanction','fontsize',12);
title('Sanction levels','fontsize',14);

figure(2)
plot(2:T, prob_audit(2:T))
xlabel('time', 'fontsize',12);
ylabel('Probability','fontsize',12);
title('Probability of detection','fontsize',14);

figure(3)
plot(2:T, rep_inc(2:T))
xlabel('time', 'fontsize',12);
ylabel('reported income','fontsize',12);
title('reported income','fontsize',14);

figure(4)
plot(2:T, revenue_time(2:T))
xlabel('time', 'fontsize',12);
ylabel('revenue','fontsize',12);
title('revenue','fontsize',14);

figure(5)
plot(2:250, Sanc(2:250));
xlabel('time', 'fontsize',12);
ylabel('Sanction','fontsize',12);
title('Sanction levels','fontsize',14);
```

```
figure(6)
plot(2:250, prob_audit(2:250))
xlabel('time', 'fontsize',12);
ylabel('Probability','fontsize',12);
title('Probability of detection','fontsize',14);

figure(7)
plot(2:250, rep_inc(2:250))
xlabel('time', 'fontsize',12);
ylabel('reported income','fontsize',12);
title('reported income','fontsize',14);

figure(8)
plot(2:250, revenue_time(2:250))
xlabel('time', 'fontsize',12);
ylabel('revenue','fontsize',12);
title('revenue','fontsize',14);

sanc = Sanc(1,T)
prob = prob_audit(1,T)
revenue = revenue_time(1,T)

sanc_std = std(Sanc(2:T))
prob_std = std(prob_audit(2:T))
revenue_std = std(revenue_time(2:T))

sanc_std_short = std(Sanc(2:250))
prob_std_short = std(prob_audit(2:250))
revenue_std_short = std(revenue_time(2:250))
```

Environment 2

```
% parameter values
x = 1;%rand;        % This is the initial random reported income
tax = 0.3;      % Setting the tax rate constant across all periods (assumed dictated by govt)
d = 2;          % this is the place for the decoded string (normalized)
M = 1;          % Total income is normalized to one

% GA parameter values

pmut = 0.01;     % rate of mutation.
n = 100;         % number of strings
k = 10;          % string length.
T = 2000;        % number of simulation periods.
seed = 0;        % seed value for random number generator.
election = 1;    % FLOW CONTROL. SET TO "1" FOR ELECTION, "0" FOR BASIC.
c=1;

% Initialize random number generators

rand('state', seed);

% create the vectors
population_p= zeros(n,k+d);   % population for the audit
population_s = zeros(n,k+d);  % population for the sanction
p_audit = zeros(1,n);
sanction = zeros(1,n);
revenue =  zeros(1,n);
pot_revenue = zeros(1,n);

replicate = zeros(n,k+d);                  % replicates hold cell
matrix_hold = zeros(4,(2*k+1));            % used in tournament and election
ww = zeros(4,(2*k+1));                     % used in tournament and election


% time vectors
rep_inc = zeros (1,T);
prob_audit = zeros(1,T);
Sanc = zeros(1,T);
revenue_time = zeros(1,T);
```

```
% (1e). Initialization of rules.

for i = 1:n
   for j = 1:k
      if (rand > 0.5)
         population_p(i,j) = 1;
      else
         population_p(i,j) = 0;
      end
      if (rand > 0.5)
         population_s(i,j) = 1;
      else
         population_s(i,j) = 0;
      end
   end
end

% (1f). Calculate normalization parameter, 'K'.

K_bar = 0;
for j = 1:k
   K_bar = K_bar + 2^(j-1);
end
K_bar = K_bar;

for t=1
   rep_inc(1,t) = x;
end
for t = 2:T

Time = t

% decode initially for the probability of audit (note: this is less than 1)
  for i = 1:n
      sum = 0;
      for j = 1:k
         sum = sum + population_p(i,j)*2^(j-1);
      end
      p_audit(1,i)= sum/K_bar;
      population_p(i,k+1) = sum/K_bar;
   end
 % sanction
  for i = 1:n
      sum = 0;
      for j = 1:k
         sum = sum + population_s(i,j)*2^(j-1);
      end
      sanction(1,i)= sum/K_bar;
      population_s(i,k+1) = sum/K_bar;
   end
   % from this the fitness of the strings can be deciphered.
   for i=1:n
      revenue(1,i) = tax*rep_inc(1,t-1)+p_audit(1,i)*sanction(1,i)*(M-rep_inc(1,t-1))-(c*p_audit(1,i))^2;
      if revenue(1,i) < 0
         revenue(1,i)=0;
      end
   end
% put in the fitness values
   for i=1:n
      population_p(i,k+d) = revenue(1,i);
      population_s(i,k+d) = revenue(1,i);
   end

   % need to do the tournament selection for the strings
   first = zeros(n,k+d);
   second = zeros(n,k+d);


   for i=1:n
```

```
   for j = 1:k+d
      first(i,j) = population_p(i,j);
      second(i,j) = population_s(i,j);
   end
end

sum=0;
for i=1:n
   sum = sum + first(i,k+d);
end

% find the relative fitness for this tournament cell
relative = zeros(n,1);

for i=1:n
   relative(i,1) = first(i,k+d)/sum;
end

for i = 1:n
   count = 0;
   click = 0;
   stop = rand;
   while (count<=stop)
      click = click + 1;
      count = count + relative(click,1);
   end
   for j = 1:k+d
      replicate_p(i,j) = first(click,j);
      replicate_s(i,j) = second(click,j);
   end
end

% this is the replicate cell based on the roulette wheel, now we need to change it into long column

% after having this tournament_cell can now make the replicates
% ok the replicates are made...

% mutation operators
if election == 0
   for i= 1:n
      for j=1:k
         if (rand < pmut);
            if  replicate_p(i,j) == 1;
               replicate_p(i,j) = 0;
            else
               replicate_p(i,j) = 1;
            end
         end
         if (rand < pmut);
            if  replicate_s(i,j) == 1;
               replicate_s(i,j) = 0;
            else
               replicate_s(i,j) = 1;
            end
         end
      end
   end
end % THIS END THE ELECTION == 0
if election == 1
   election_hold_p = zeros(n,k+d);
   election_hold_s = zeros(n,k+d);
   for i=1:n
      for j=1:k
         election_hold_p(i,j) = replicate_p(i,j);
         election_hold_s(i,j) = replicate_s(i,j);
      end
   end
% mutation operator with the election operator

   for i = 1:n
```

```
        for j=1:k
          if (rand < pmut);
            if election_hold_p(i,j) == 1;
              election_hold_p(i,j) = 0;
            else
              election_hold_p(i,j) = 1;
            end
          end
          if (rand < pmut);
            if election_hold_s(i,j) == 1;
              election_hold_s(i,j) = 0;
            else
              election_hold_s(i,j) = 1;
            end
          end
        end
      end
      % given this election hold cell, the potential fitness of the strings can be looked at
      % going to need to decode these strings as well and use the past values of x observed in market as starting poin
      for i = 1:n
        sum = 0;
        for j = 1:k
          sum = sum + election_hold_p(i,j)*2^(j-1);
        end
        election_hold_p(i,k+1)= sum/K_bar;

        sum = 0;
        for j = 1:k
          sum = sum + election_hold_s(i,j)*2^(j-1);
        end
        election_hold_s(i,k+1)= (sum/K_bar);
      end
      % now need to check for the potential fitness of these new strings
      for i=1:n
        for j= 1:k
          matrix_hold(1,j)=replicate_p(i,j);
          matrix_hold(1,j+k)=replicate_s(i,j);
          matrix_hold(2,j)=replicate_p(i,j);
          matrix_hold(2,j+k)=election_hold_s(i,j);
          matrix_hold(3,j)=election_hold_p(i,j);
          matrix_hold(3,j+k)=replicate_s(i,j);
          matrix_hold(4,j)=election_hold_p(i,j);
          matrix_hold(4,j+k)=election_hold_s(i,j);
        end

        % this finds the relative fitness and imposes the election operator
        matrix_hold(1,2*k+1) = replicate_p(i,k+d);
        matrix_hold(2,2*k+1) = tax*rep_inc(1,t-1)+replicate_p(i,k+1)*election_hold_s(i,k+1)*(M-rep_inc(1,t-1))-
(c*replicate_p(i,k+1))^2;

        matrix_hold(3,2*k+1) = tax*rep_inc(1,t-1)+election_hold_p(i,k+1)*replicate_s(i,k+1)*(M-rep_inc(1,t-1))-
(c*election_hold_p(i,k+1))^2;

        matrix_hold(4,2*k+1) = tax*rep_inc(1,t-1)+election_hold_p(i,k+1)*election_hold_s(i,k+1)*(M-rep_inc(1,t-1))-
(c*election_hold_p(i,k+1))^2;

        ww = sortrows(matrix_hold,2*k+1);
        for j=1:k
          replicate_p(i,j) = ww(4,j);
          replicate_s(i,j) = ww(4,j+k);
        end
        replicate_p(i,k+d) = ww(4,2*k+1);
        replicate_s(i,k+d) = ww(4,2*k+1);
        if replicate_p(i,k+d)<0;
          replicate_p(i,k+d) = 0;
        end
        if replicate_s(i,k+d)<0;
          replicate_s(i,k+d) = 0;
        end
      end
```

```
end

% this in theory should have created the new set of binary strings that we will base stuff on now use the roulette wheel in order to
choose one single string to use
fitness = zeros(n,1);
for i=1:n
   fitness(i,1) = replicate_p(i,k+d);
end
sum_fitness=0;
for i=1:n
   sum_fitness= sum_fitness +fitness(i,1);
end
relative_fitness = zeros(n,1);
% find the relative fitness
for i=1:n
   relative_fitness(i,1) = fitness(i,1)/sum_fitness;
end
% at the end make sure to convert back

% using this relative fitness try to implement the roulette wheel

count=0;
click=0;
stop=rand;
   while (count<=stop)
     click=click+1;
     count= count + relative_fitness(click,1);
   end
% We can now use this counter to pick the appropriate parameters for use in the next period
% first convert these replicates back into to actual population for use in next time period

for i=1:n
   for j=1:k+d
      population_p(i,j) = replicate_p(i,j);
      population_s(i,j) = replicate_s(i,j);
   end
end
% create a new holding cell for the rule set that the agents will choose this time around
pick_cell=zeros(2,k);
for j=1:k+d
   pick_cell(1,j)= replicate_p(click,j);
   pick_cell(2,j)= replicate_s(click,j);
end

% given this, decode for actual probability of audit and the sanction that is learned from the
% constant tax rates (note that the odd number here are the probability of audit while the sanction are even numbered rows

% need to decode these to find the final government choices
%first decode for the tax levels
prob_audit(1,t) = pick_cell(1,k+1);
P = prob_audit(1,t);


        Sanc(1,t)= pick_cell(2,k+1);
S = Sanc(1,t);
if t > 1250
  tax = 0.5;
end
% decode for the actual reported income that maximizing agents would choose in this case
rep = (1-(P*(S-tax)/tax*(1-P))-S)*M/((tax-P*(S-tax)/(1-P))-S);
if rep > 1
   rep = 1;
end
if rep < 0
   rep = 0;
end
rep_inc(1,t) = rep;
% calculate the actual revenue that will prevail
revenue_time(1,t) = tax*rep_inc(1,t)+prob_audit(1,t)*Sanc(1,t)*(M-rep_inc(1,t))-(c*prob_audit(1,t))^2;
```

```
end

figure(1)
plot(2:T, Sanc(2:T));
xlabel('time', 'fontsize',12);
ylabel('Sanction','fontsize',12);
title('Sanction levels','fontsize',14);

figure(2)
plot(2:T, prob_audit(2:T))
xlabel('time', 'fontsize',12);
ylabel('Probability','fontsize',12);
title('Probability of detection','fontsize',14);

figure(3)
plot(2:T, rep_inc(2:T))
xlabel('time', 'fontsize',12);
ylabel('reported income','fontsize',12);
title('reported income','fontsize',14);

figure(4)
plot(2:T, revenue_time(2:T))
xlabel('time', 'fontsize',12);
ylabel('revenue','fontsize',12);
title('revenue','fontsize',14);

figure(5)
plot(2:250, Sanc(2:250));
xlabel('time', 'fontsize',12);
ylabel('Sanction','fontsize',12);
title('Sanction levels','fontsize',14);

figure(6)
plot(2:250, prob_audit(2:250))
xlabel('time', 'fontsize',12);
ylabel('Probability','fontsize',12);
title('Probability of detection','fontsize',14);

figure(7)
plot(2:250, rep_inc(2:250))
xlabel('time', 'fontsize',12);
ylabel('reported income','fontsize',12);
title('reported income','fontsize',14);

figure(8)
plot(2:250, revenue_time(2:250))
xlabel('time', 'fontsize',12);
ylabel('revenue','fontsize',12);
title('revenue','fontsize',14);

sanc = Sanc(1,T)
prob = prob_audit(1,T)
revenue = revenue_time(1,T)

sanc_std = std(Sanc(2:T))
prob_std = std(prob_audit(2:T))
revenue_std = std(revenue_time(2:T))

sanc_std_short = std(Sanc(2:250))
prob_std_short = std(prob_audit(2:250))
revenue_std_short = std(revenue_time(2:250))
```

REFERENCE LIST

Allingham, M.G. and A. Sandmo (1972) Income tax evasion: A theoretical analysis, *Journal of Public Economics* 1, 323-338

Andreoni, J., B. Erard and J. Feinstein (1998) Tax Compliance, *Journal of Economic Literature* 36 (2), 818-860

Andreoni, J. (1991) Reasonable Doubt and the Optimal Magnitude of Fines: Should the Penalty fit the Crime?, *RAND Journal of Economics* 22 (3), 385-395

Arifovic, J. (1994) Genetic algorithm learning and the cobweb model, *Journal of Economic Dynamics and Control* 18, 3-28

Becker, G.S. (1968) Crime and Punishment: An Economic Approach, *Journal of Political Economy*, 78, 169-217

Brooks, A.C. (2000) Genetic Algorithms and Public Economics, *Journal of Public Economic Theory* 2(4), 493-513

Cowell, F.A. (1990) *Cheating the Government.* Cambridge, Massachusetts: The MIT Press

Edlund, J. and R. Aberg (2002) Social norms and tax compliance, *Swedish Economic Policy Review* 9, 201-228

Feige, E.L. (1979), How big is the irregular economy?, *Challenge*, 5-13

Franzoni, L.A. (1998) Tax Evasion and tax compliance, *Encyclopedia of Law and Economics*

Goldberg, D.E. (1989) *Genetic algorithms in search, optimization and machine learning.* New York, NY: Addison Wesley

Holland, J.H. (1975) *Adaptation in Natural and Artificial Systems.* Ann Arbor: University of Michigan Press

Myles, G..D. (1995) *Public Economics*. Cambridge, United Kingdom: Cambridge University Press

Rey, M. (1965) Estimating tax evasions: the example of the Italian General Sales Tax;, *Public Finance* 20, 366-392

Roth, J., J. Scholz, and A. Witte, 1989, *Taxpayer Compliance Volume I: An Agenda for Research*. Philadelphia: University of Pennsylvania Press

Sanchez, J. and A. Juan (1995) Economic and Noneconomic Factors in Tax Compliance, *Kyklos* 48, 3-18

Yitzhaki, S. (1974) A note on income tax evasion: A theoretical analysis, *Journal of Public Economics* 3, 201-202