# A Genetic Algorithm for the Structural Estimation of Games with Multiple Equilibria

Victor Aguirregabiria[1] and Pedro Mira
Boston University CEMFI

January 2005

**Abstract:** This paper proposes an algorithm to obtain maximum likelihood estimates of structural parameters in discrete games with multiple equilibria. The method combines a genetic algorithm (GA) with a pseudo maximum likelihood (PML) procedure. The GA searches efficiently over the huge space of possible combinations of equilibria in the data. The PML procedure avoids the repeated computation of equilibria for each trial value of the parameters of interest. To test the ability of this method to get maximum likelihood estimates, we present a Monte Carlo experiment in the context of a game of price competition and collusion.

**Keywords:** Genetic algorithms; Maximum likelihood estimation; Multiple equilibria.

**JEL classification:** C13, C35.

**Contact address:** Victor Aguirregabiria. Boston University. Department of Economics. 270 Bay State Road. Boston, MA 02215. E-mail: vaguirre@bu.edu

# 1 Introduction

Multiplicity of equilibria is a prevalent feature in empirical discrete games. Models with multiple equilibria do not have a unique reduced form and this indeterminacy poses practical estimation problems. An important issue associated with maximum likelihood (ML) estimation of models with multiple equilibria is that one should maximize the likelihood not only with respect to the structural parameters but also with respect to the *equilibrium types* that generate the observations in the data. There are two main reasons why optimization with respect to *equilibrium types* can be a very complicated task. First, computing all the equilibria associated with each trial value of the parameters can be computationally very demanding. And second, the number of possible combinations of equilibria in the data increases exponentially with sample size and it is a huge number even for the simplest problems with three equilibria and one hundred observations.

---

[1] This material is based upon work supported by the National Science Foundation under Grant No. 0241943.

In the context of empirical games, several papers have proposed pseudo maximum likelihood (PML) estimators that avoid the problems associated with ML estimation (see Tamer, 2003, Aguirregabiria and Mira, 2004, Bajari, Benkard and Levin, 2004, Pakes, Ostrowsky and Berry, 2004, and Pesendorfer and Smichdt-Dengler, 2004).These PML methods proceed in two steps. The first step identifies nonparametrically the equilibrium, or equilibria, played in the data. In the second step, estimates of structural parameters are obtained by maximizing a (pseudo) likelihood based on best response functions evaluated at the equilibrium estimated in the first step. The main advantage of PML estimation is its computational simplicity. However, these two-step methods are not statistically efficient, and they typically perform poorly in small samples. To deal with this limitation, Aguirregabiria and Mira (2002, 2004) propose a recursive extension of the pseudo likelihood method, the so called nested pseudo likelihood (NPL) algorithm. The NPL provides estimates which are statistically more efficient than the two-step PML, both asymptotically and in finite samples. However, in the context of empirical games, NPL does not necessarily returns the ML estimator.[2] We show in this paper that the ML estimator should be a fixed point of the NPL algorithm, but the algorithm can have more than one fixed point. This issue motivates the combination of the NPL with an algorithm that searches efficiently for the ML estimator in the set of NPL fixed points.

This paper proposes a relatively simple algorithm to obtain the ML estimator. The algorithm combines the NPL procedure in Aguirregabiria and Mira (2002, 2004) with a Genetic Algorithm (GA). The GA searches efficiently over the huge space of possible combinations of equilibria in the data. The PML procedure avoids the repeated computation of equilibria for each trial value of the structural parameters. GAs were first proposed by Holland (1975).[3] Among their many applications, GAs have been successfully used to search for the global optimum of discrete and step functions with very large search spaces (see chapter 4 in Mitchell, 1996, Mitchell, Holland and Forrest, 1994, and section 8.3 in Judd, 1998). The problem of maximum likelihood estimation of models with multiple equilibria belongs to this class. Although GAs have been extensively used in experimental and evolutionary economics, the application of GAs in econometrics has been rare. Some important exceptions are Dorsey and Mayer (1995) and Beenstock and Szpirob (2002).

The rest of the paper is organized as follows. Section 2 presents the model. Here we consider static games of incomplete information, but the method can be extended to dynamic games. Section 3 discusses the estimation problem and the NPL algorithm. We describe

---

[2] In single-agent dynamic programming models, the NPL always provides the MLE. This is because in these models the NPL has a unique fixed-point that is the MLE. See Aguirregabiria and Mira (2002).

[3] For an excellent survey on the theory and application of GAs see Mitchell (1996).

the genetic algorithm in section 4. We test the ability of this procedure to obtain maximum likelihood estimates using a Monte Carlo experiment in the context of a game of price competition and collusion. Section 5 presents this experiment. We summarize and conclude in section 6.

## 2 Model

There are $N$ players which are indexed by $i \in I = \{1, 2, ..., N\}$. Each player chooses an action from a set of choice alternatives $A = \{0, 1, ..., J\}$. We represent the decision of player $i$ by the variable $a_i \in A$. The utility function of player $i$ is:

$$U_i = u_i(a_i, a_{-i}, x) + \varepsilon_i(a_i) \tag{1}$$

where $a_{-i}$ is the vector with the decisions of players other than $i$; $x$ is a vector of players' characteristics which are exogenous and common knowledge for all players; and $\varepsilon_i \equiv (\varepsilon_i(0), \varepsilon_i(1), ..., \varepsilon_i(J))$ represents characteristics that are private information of player $i$.

*ASSUMPTION 1: For any $i \in I$ the vector $\varepsilon_i$ is: (1) a vector of real valued random variables, $\varepsilon_i \in R^{J+1}$; (2) independent of common knowledge variables $x$; and (3) independently distributed across players with distribution function $G_i(.)$ that is absolutely continuous with respect to the Lebesgue measure.*

Let $\sigma = \{\sigma_i(x, \varepsilon_i) : i \in I\}$ be a set of strategy functions where $\sigma_i : X \times R^{J+1} \rightarrow A$. Associated with a set of strategy functions we can define a set of *choice probabilities* $P^\sigma(x) = \{P_i^\sigma(a_i|x) : (a_i, i) \in A \times I\}$ such that:

$$P_i^\sigma(a_i|x) \equiv \int I\left\{\sigma_i(x, \varepsilon_i) = a_i\right\} \; dG_i(\varepsilon_i) \tag{2}$$

where $I\{.\}$ is the indicator function. These probabilities represent the expected behavior of player $i$ from the point of view of the other players (who do not know $\varepsilon_i$) when he follows his strategy in $\sigma$.

Let $u_i^\sigma(a_i, x)$ be player $i$'s expected utility if he chooses alternative $a_i$ and the other players behave according to their respective strategies in $\sigma$. By the independence of private information in Assumption 1,

$$u_i^\sigma(a_i, x) = \textstyle\sum_{a_{-i}} \left(\prod_{j \neq i} P_j^\sigma(a_j|x)\right) u(a_i, a_{-i}, x) \tag{3}$$

*DEFINITION: A Bayesian Nash equilibrium (BNE) in this game is a set of strategy functions $\sigma^*$ such that for any player $i$ and for any $(x, \varepsilon_i) \in X \times R^{J+1}$,*

$$\sigma_i^*(x, \varepsilon_i) = \arg\max_{a_i \in A} \left\{ \; u_i^{\sigma^*}(a_i, x) + \varepsilon_i(a_i) \; \right\} \tag{4}$$

We can represent this BNE in probability space. Let $\sigma^*$ be a set of BNE strategies, and let $P^*$ be the choice probabilities associated with these strategies. By definition, $P_i^*(a_i|x) = \int I\left\{a_i = \sigma_i^*(x, \varepsilon_i)\right\} \, dG_i(\varepsilon_i)$. Solving the equilibrium condition (4) in this expression we get that for any $(a_i, i) \in A \times I$:

$$P_i^*(a_i|x) = \int I\left(a_i = \arg\max_{a \in A} \left\{u_i^*(a, x) + \varepsilon_i(a)\right\}\right) \, dG_i(\varepsilon_i) \tag{5}$$

Notice that the function $u_i^\sigma$ depends on players' strategies only through the choice probabilities $P^\sigma$ associated with $\sigma$. Therefore, the right hand side in equation (5) is a function that we define as $\Lambda_i(a_i|x; P)$. We call the functions $\Lambda_i$ *best response probability functions.* The vector of equilibrium probabilities $P^*(x) \equiv \{P_i^*(a_i|x) : (a_i, i) \in A \times I\}$ is a fixed point of the best response mapping $\Lambda(x, P) \equiv \{\Lambda_i(a_i|x, P) : (a_i, i) \in A \times I\}$. Given Assumption 1, best response probability functions are continuous in the compact set of players' choice probabilities. By Brower's theorem, there exists at least one equilibrium. In general, the equilibrium is not unique.

The primitives of the model $\{u_i, G_i : i \in I\}$ can be described in terms of a vector of parameters $\theta \in \Theta \subseteq R^K$. Primitives are assumed to be continuously differentiable in $\theta$. We use $\Lambda(x, \theta, P)$ to denote the equilibrium mapping associated with $(x, \theta)$. And $P(x, \theta) = \{P_i(a|x, \theta) : (a, i) \in A \times I\}$ represents an equilibrium associated with $(x, \theta)$ such that $P(x, \theta) = \Lambda(x, \theta, P(x, \theta))$. For some values of $(x, \theta)$ the model has multiple equilibria. Let $\{P^\tau(x, \theta) : \tau = 1, 2, ...\}$ be the set of equilibria associated with $(x, \theta)$. The equilibria are indexed by the variable $\tau \in \{1, 2, ...\}$ that is called the *equilibrium type*.

**EXAMPLE: Price Competition and Collusion.** Consider a market with $N = 2$ firms selling a differentiated product and competing in prices. The profit of firm $i$ is $U_i = (p_i - c)$ $D_i(p_i, p_j, x)$ where: $p_i$ and $p_j$ are the prices of the two firms; $c$ is the cost per unit; $D_i$ is the demand function of firm $i$ that depends on prices and on some $x$ variables that represent the state of the market demand. Each of these firms chooses between two alternatives: charge the monopoly price (i.e., $a_i = M$) or charge the Bertrand competition price (i.e., $a_i = B$). The payoff matrix of this game is:

| | | **Firm j** | |
|---|---|---|---|
| | | **Monopoly** | **Bertrand** |
| **Firm i** | **Monopoly** | $U(M,M)$<br>$U(M,M)$ | $U(M,M) - \Delta(M,B)$<br>$U(M,M) + \Delta(B,M)$ |
| | **Bertrand** | $U(M,M) + \Delta(B,M)$<br>$U(M,B) - \Delta(M,B)$ | $U(M,M) - \Delta(B,B)$<br>$U(M,M) - \Delta(B,B)$ |

$U(M, M)$ is a firm's profits when both firms choose monopoly prices (i.e., collusive outcome). $\Delta(M, B) > 0$ represents the reduction in own profits when the firm chooses the monopoly price but the competitor plays Bertrand. $\Delta(B,M) > 0$ is the increase in own profits when the firm chooses the Bertrand pice and the competitor charges the monopoly price. Finally, $2\Delta(B,B) > 0$ represents the reduction in the total profits of the two firms when we go from the monopoly outcome to the Bertrand outcome. For notational simplicity I have omitted $x$ as an argument in these payoff functions. Under very general conditions, we have that:

$$\Delta(M,B) - \Delta(B,M) > 2\ \Delta(B,B) > 0 \tag{6}$$

Suppose that there is also an additive private information component $\varepsilon_i(a_i)$ in the profits function of these firms. Then, firm $i$'s expected profit is:

$$u_i^\sigma(a_i,x) = \begin{cases} U(M,M) + P_j(B|x)\ \ [-\Delta(M,B)\ ] + \varepsilon_i(M) & if \quad a_i = M \\ U(M,M) + \Delta(B,M) + P_j(B|x)\ \ [-\Delta(B,B) - \Delta(B,M)\ ] + \varepsilon_i(B) & if \quad a_i = B \end{cases} \tag{7}$$

Given these expected profits, the best response function for firm $i$ is:

$$a_i = B \ \ \Leftrightarrow \ \ \varepsilon_i(M) - \varepsilon_i(B) \leq \Delta(B,M) + P_j(B|x)\ \ [\Delta(M,B) - \Delta(B,M) - \Delta(B,B)\ ] \tag{8}$$

Suppose that the private information variables are such that $\varepsilon_i(M) - \varepsilon_i(B)$ is $iid$ $N(\mu,\sigma^2)$. Following our description of a BNE in probability space, it is straightforward to show that the equilibrium probabilities are such that $P_i^*(B|x) = \Lambda_i(B|x,P^*)$ where:

$$\Lambda_i(B|x;P) = \Phi\ (\ \theta_0(x) + \theta_1(x)\ P_j(B|x)\ ) \tag{9}$$

where $\Phi(.)$ is the CDF of the standard normal; $\theta_0(x) \equiv [\Delta(B,M) - \mu]/\sigma$ and $\theta_1(x) \equiv [\Delta(M,B) - \Delta(B,M) - \Delta(B,B)]/\sigma$. Notice that condition (6) implies that $\theta_1(x)$ is strictly positive. Therefore, there is strategic complementarity between the pricing decisions of these firms.

This model can have multiple equilibria. In fact, it can have multiple symmetric equilibria. Figure 1 presents an example of multiple symmetric equilibria. The value of the payoffs in this example are such that $\theta_0(x) = -1.7$ and $\theta_1(x) = 3.6$.

# 3 Estimation

## 3.1 Data generating process

Suppose that the game is played at different moments in time or at different locations or markets. We have a random sample of $T$ realizations of the game where we observe players' actions and common knowledge state variables $\{a_t, x_t : t = 1, 2, ..., T\}$ with $a_t =$

$(a_{1t}, a_{2t}, ..., a_{Nt})$. Let $\theta^0 \in \Theta$ be the true value of $\theta$ in the population under study. We are interested in the estimation of $\theta^0$.

Let $\tau_t$ be the equilibrium type of observation $t$. The equilibrium types of the different sample observations are unknown to the researcher. Let $P_t^0(x_t) \equiv \{\Pr(a_{it} = a|x_t) : (a, i) \in A \times I\}$ be the distribution of $a_t$ conditional on $x_t$ in the population that generates observation $t$. Since $a_t$ comes from an equilibrium of the game associated with $(x_t, \theta^0)$, we know that there is an equilibrium type $\tau_t$ such that $P_t^0(x_t) = P^{\tau_t}(x_t, \theta^0)$. The following assumption establishes some conditions on the data generating process that guarantee the identification of $\theta^0$.

*ASSUMPTION 2: (A) For every observation t, the equilibrium type $\tau_t$ is determined by a function of the common knowledge state variables, $\tau^0(.) \in \Upsilon$, such that $\tau_t = \tau^0(x_t)$. And (B) there is a unique pair $(\theta^0, \tau^0) \in \Theta \times \Upsilon$ such that $P^0(x_t) = P^{\tau^0(x_t)}(x_t, \theta^0)$ for every $x_t \in X$.*

Under Assumption 2(A), two games with the same common knowledge variables have the same equilibrium probabilities. The function $\tau^0$ is called the *equilibrium selection mechanism.* Under this assumption we can identify nonparametrically the equilibrium probabilities in the population associated with possible value of $x_t$. For instance, if $x$ is discrete, the frequency estimator $\hat{P}_i^0(a_i|x) = \sum_{t=1}^T I\{a_{it} = a_i; x_t = x\} / \sum_{t=1}^T I\{x_t = x\}$ is a consistent estimator of $P_i^0(a_i|x)$. Assumption 2(B) establishes the joint identification of the structural parameters and the equilibrium selection mechanism.

## 3.2 Maximum likelihood estimation

The maximum likelihood estimator (MLE) of $(\theta^0, \tau^0)$ maximizes the likelihood with respect to $\theta$ and with respect to the equilibrium types in the sample. Thus, the MLE is:

$$\hat{\theta} = \arg\max_{\theta \in \Theta} \left\{ \begin{array}{ll} \sup_{\tau_1, ..., \tau_T} & \sum_{t=1}^T \sum_{i=1}^N \log P_i^{\tau_t}(a_{it}|x_t; \theta) \\ \textit{subject to}: & \tau_t = \tau_s \text{ for } x_t = x_s \end{array} \right\} \tag{10}$$

When the support of the variable $x$ is discrete the number of equilibrium types is finite and all the regularity conditions of maximum likelihood estimation hold. This estimator is consistent and asymptotically efficient. However, its implementation can be computationally very costly. The problem is in the maximization with respect to the equilibrium types. First, we need to know all the equilibrium types that the model has for every trial value of $\theta$ and for each sample value of $x$. This is impractical in most applications. And second, the number of of possible values of $\{\tau_1, ..., \tau_T\}$ is typically huge. For instance, if the number of equilibrium types is 3 and the number of possible values of $x_t$ is 20, we have $3^{20} \simeq 10^{10}$ possible values for the vector $\{\tau_1, ..., \tau_T\}$. It is clear that optimization with respect to the equilibrium types can be extremely costly.

This problem has motivated the development of alternative methods, like pseudo maximum likelihood (PML) estimation, that avoid the search for the MLE of $\tau^0$. Here we propose an algorithm to compute the MLE of $(\theta^0, \tau^0)$. This algorithm combines the PML procedure in Aguirregabiria and Mira (2003) with a Genetic Algorithm (GA). The GA searches efficiently over the huge space of possible combinations of equilibria in the data. In principle, we could use the GA to solve the $\sup_{\tau_1,...,\tau_T}\{.\}$ problem in the MLE for each trial value of $\theta$. However, that type of algorithm would have to solve the $\sup_{\tau_1,...,\tau_T}\{.\}$ problem many times, i.e., as many as the number of values of $\theta$ that we consider in the search for the MLE. This approach requires also to compute all the equilibria that the model has for each trial value of $\theta$. These tasks are computationally very demanding for most applications of these models. Our algorithm avoids these problems by combining the GA with a PML procedure.

Before we describe our algorithm, it is convenient to characterize the MLE in a way that will be useful for the implementation of the algorithm. The MLE in (10) can be also described as:

$$\hat{\theta} = \arg\max_{\theta\in\Theta} \left\{ \begin{array}{rl} \sup_{P_1,...,P_T} & Q\left(\theta, P_1, ..., P_T\right) \\ \textit{subject to}: & P_t = \Lambda(x_t, \theta, P_t) \text{ for each } t \\ & P_t = P_s \text{ for } x_t = x_s \end{array} \right\} \tag{11}$$

where

$$Q\left(\theta, P_1, ..., P_T\right) = \sum_{t=1}^{T}\sum_{i=1}^{N} \log \Lambda_i(a_{it}|x_t, \theta, P_t) \tag{12}$$

We call $Q\left(\theta, P_1, ..., P_T\right)$ the *pseudo likelihood function.* This function is defined for arbitrary choice probabilities, not necessarily in equilibrium. In the characterization of the MLE in (11) we have replaced the maximization with respect to the equilibrium types in (10) by a maximization with respect to the vector of equilibrium probabilities. This is a very simple modification but it has important implications for our procedure to search for the MLE. The evaluation of the equilibrium probabilities $P_i^{\tau_t}(a_{it}|x_t;\theta)$ requires one to solve for the equilibria associated with $(x_t;\theta)$. In contrast, the evaluation of the best response probabilities $\Lambda_i(a_{it}|x_t, \theta, P_t)$ is very simple and it does not require one to compute an equilibrium. In our algorithm, the restriction $P_t = \Lambda(x_t, \theta, P_t)$ will not be imposed for every trial value of $\theta$ but only for the final estimator $\hat{\theta}$.

## 3.3 Pseudo maximum likelihood estimation

This section follows Aguirregabiria and Mira (2004). The two-step PML estimator is the value of $\theta$ that maximizes the pseudo likelihood function $Q\left(\theta, \hat{P}_1, ..., \hat{P}_T\right)$, where $\{\hat{P}_1, ..., \hat{P}_T\}$ are nonparametric estimates of players' choice probabilities conditional on $x$. Notice that the nonparametric estimates of choice probabilities can be interpreted as estimates of the equilibrium selection mechanism $\tau^0$. Therefore, this estimator avoids the search for the MLE of $\tau^0$ by estimating it nonparametrically.

The nested pseudo likelihood (NPL) estimator is a recursive extension of the two-step PML. Given the initial nonparametric estimates of choice probabilities, the NPL generates a sequence of estimators $\{\hat{\theta}^K : K \geq 1\}$ where the $K - stage$ estimator is defined as:

$$\hat{\theta}^K = \arg\max_{\theta \in \Theta} Q\left(\theta, \hat{P}_1^K, ..., \hat{P}_T^K\right) \tag{13}$$

and the probabilities $\{\hat{P}_1^K, ..., \hat{P}_T^K : K \geq 2\}$ are obtained recursively as

$$\hat{P}_t^{K+1} = \Lambda(x_t, \hat{\theta}^K, \hat{P}_t^K) \tag{14}$$

The algorithm iterates until convergence. The NPL estimator of the probabilities can be described as a fixed point of a composed mapping that combines the first order conditions of (13) with the iteration in (14). If $\Lambda$ is continuously differentiable with respect to $\theta$ and with respect to the probabilities, then the NPL mapping is continuous and differentiable in the compact space of probabilities. Therefore, Brower's fixed-point theorem guarantees that for a given sample there exists at least one NPL fixed-point. The NPL estimator is the NPL fixed-point with the largest value of the pseudo likelihood. function.

*PROPOSITION: The NPL estimator is the MLE estimator.*

*Proof:* According to the definition in (11), the MLE can be characterized as a pair $(\hat{\theta}, \hat{P})$ that maximizes $Q(\theta, P)$ subject to the constraints $\hat{P}_t = \Lambda(x_t, \hat{\theta}, \hat{P}_t)$. An NPL fixed point, say $(\hat{\theta}_{NPL}, \hat{P}_{NPL})$ is such that $\hat{\theta}_{NPL}$ maximizes $Q\left(\theta, \hat{P}_{NPL}\right)$ and $\hat{P}_{t,NPL} = \Lambda(x_t, \hat{\theta}_{NPL}, \hat{P}_{t,NPL})$. It is clear that the MLE satisfies these conditions and therefore it belongs to the set of NPL fixed points. It is also clear that the MLE is the NPL fixed point with the highest value of $Q(\theta, P)$. But this is also the definition of the NPL estimator. Therefore, the NPL estimator is the MLE estimator. $\blacksquare$

# 4 Genetic algorithm

Computing a NPL fixed-point is a computationally simple task even in relatively complex models like dynamic games (see Aguirregabiria and Mira, 2004). However, when there is not a unique NPL fixed-point, computing the NPL estimator or MLE can be a much more demanding problem. A possible procedure to obtain the MLE is a *parallel NPL* method. That is, we can obtain $M$ fixed points of the NPL procedure by applying this method with $M$ different initial values for the choice probabilities. These $M$ initial values can be obtained by drawing $M$ bootstrap samples from the original sample and then calculating a nonparametric estimator of the probabilities for each bootstrap sample. Given these NPL fixed-points, we choose as estimator of $\theta^0$ the fixed point with the highest value of the pseudo likelihood. A limitation of this approach is that we may need a large number of initial values to guarantee that this estimator is the MLE. To deal with this problem we combine the *parallel NPL* method with a GA method.

We initialized the algorithm with $M$ different values of the choice probabilities. At every step of the algorithm we perform three operations on the $M$ vectors of probabilities: crossover, mutation, and selection. These operations, which are characteristic of GAs, make the $M$ processes not independent of each other and this allows for a much more global search over the space of probabilities. Here we describe the algorithm in more detail.

*(0) Initial population.* The initial "population" of probability vectors is $\Pi^1 = \{\hat{P}^1_{mt} : t = 1, .., T; m = 1, ..., M\}$, where $M$ is the size of the population. This initial population may be arbitrarily chosen, or it may come from a nonparametric estimates. For instance, the probabilities could be obtained as $M$ bootstrap nonparametric estimates of players' choice probabilities.

The GA generates a sequence of populations of size $M$ that we denote by $\{\Pi^K : K \geq 1\}$. Associated with this sequence of probabilities the GA also generates a sequence of sets of parameter estimates $\{\Theta^K : K \geq 1\}$.with $\Theta^K = \{\hat{\theta}^K_m : m = 1, ..., M\}$. An iteration of the algorithm consists in the creation of a new generation with the offsprings of the existing generation. An iteration can be described in terms of four processes or steps that are followed sequentially: (1) mating or selection of parents; (2) crossover and mutation; (3) NPL iteration; (4) selection of offsprings.

*(1) Selection of parents.* We draw, with replacement, $O > M$ pairs of probability vectors from the population $\Pi^K$. The probability that a vector is chosen depends on its relative *fitness*. Fitness is a term from evolution theory. In our problem, the fitness of a probability vector is the Lagrangian function:

$$l(\hat{P}_1, ..., \hat{P}_T) \quad = \quad Q\left(\hat{\theta}, \hat{P}_1, ..., \hat{P}_T\right) - \lambda \textstyle\sum_{t=1}^{T} \left\| \hat{P}_t - \Lambda(x_t, \hat{\theta}, \hat{P}_t) \right\| \tag{15}$$

where $\hat{\theta}$ is the PML estimate associated with $\left(\hat{P}_1, ..., \hat{P}_T\right)$, and $\lambda$ is a small and positive constant. Given the measures of fitness of the $M$ elements of $\Pi^K$, the probability that the $m-th$ element is selected is:

$$S^K_m = \frac{\exp\left\{\varphi\ l(\hat{P}^K_{m1}, ..., \hat{P}^K_{mT})\right\}}{\sum_{j=1}^{M} \exp\left\{\varphi\ l(\hat{P}^K_{j1}, ..., \hat{P}^K_{jT})\right\}} \tag{16}$$

where $\varphi \geq 0$ is a parameter that measures the strength of the dependence of selection on fitness. If $\varphi = 0$, every individual has the same probability $1/M$ of being selected. If $\varphi = \infty$, only the fittest individual is selected in the $O$ draws.

*(2) Crossover and mutation.* Each couple generates one offspring. An offspring inherits "chromosomes" from its parents, but there maybe mutation as well. We represent this with two sets of binary variables: $\{z_t, d_t : t = 1, 2, ..., T\}$. $z_t$ is the indicator of a mutation for chromosome $t$ and it is i.i.d. over $t$ with mutation probability $\Pr(z_t = 1) = \gamma$. The variable $d_t$ is the indicator for the identity of the parent who transmits the $t - th$ chromosome and it is i.i.d. over $t$ with $\Pr(d_t = 1) = 1/2$. Let $\{\hat{P}_{m1}, ..., \hat{P}_{mT}\}$ and $\{\hat{P}_{m'1}, ..., \hat{P}_{m'T}\}$ be a couple. Then, the offspring from this couple is $\{\hat{P}'_1, ..., \hat{P}'_T\}$ where for any $t$:

$$\begin{aligned} \hat{P}'_t &= d_t \left\{ \hat{P}_{mt} + z_t \; \delta \; (\hat{P}_{mt} - U_t) \right\} \\ &+ (1 - d_t) \; \left\{ \hat{P}_{m't} + z_t \; \delta \; (\hat{P}_{m't} - U_t) \right\} \end{aligned} \tag{17}$$

where $U_t$ is a vector of $N$ independent random draws from a $U(0, 1)$; and $\delta$ is a parameter that represents the magnitude of the mutation. The mutation parameters $\gamma$ and $\delta$ can change over the iterations in such a way that the go to zero.

*(3) NPL iteration.* For each offspring we obtain its associated PML estimator of $\theta^0$, i.e., the value of $\theta$ that maximizes the pseudo likelihood $Q\left(\theta, \hat{P}'_1, ..., \hat{P}'_T\right)$. Then, for each offspring $\{\hat{P}'_1, ..., \hat{P}'_T\}$ and its associated PML estimator $\hat{\theta}$, we obtain a new offspring $\{\hat{P}''_1, ..., \hat{P}''_T\}$ as $\hat{P}''_t = \Lambda(x_t, \hat{\theta}, \hat{P}'_t)$.

*(4) Selection of offsprings.* We calculate the fitness of each new $O$ offsprings and select the $M$ ones with highest fitness. This is the new population $\Pi^{K+1}$.

The algorithm iterates until convergence of the sequence of populations $\{\Pi^K\}$.

# 5 Monte Carlo experiment

## 5.1 Specification of the experiment

In this section we test the performance of the previous GA using a Monte Carlo experiment. The experiment is based on the model of collusion that we presented at the end of section 2. Let $x_t$ be a common knowledge variable that represents the state of the demand, e.g., market size. The specification of the model is such that:

$$\begin{aligned} \theta_0(x_t) \equiv \frac{\Delta(B, M; x_t) - \mu}{\sigma} &= -\theta_{00} + \theta_{01} \; x_t \\ \theta_1(x_t) \equiv \frac{\Delta(M, B; x_t) - \Delta(B, M; x_t) - \Delta(B, B; x_t)}{\sigma} &= \theta_{10} - \theta_{11} \; x_t \end{aligned} \tag{18}$$

where all the $\theta$ parameters are positives. Therefore, $\theta_0(x_t)$, that represents the benefits of deviating from the collusive outcome, increases with market size. However, $\theta_1(x_t)$, that represents the degree of strategic complementarity, decreases with market size. We consider the following values for the parameters:

$$\theta_{00} = 2.25 \quad ; \quad \theta_{01} = 0.75 \quad ; \quad \theta_{10} = 4.50 \quad ; \quad \theta_{11} = 1.50$$

Market size $x_t$ is drawn from a uniform distribution on the interval $[0, 1]$. For these parameter values and for any $x_t \in [0, 1]$ the model has always three equilibria. Figure 2 presents the best response mapping and the corresponding equilibria for three different market sizes.

To complete the specification of the model we should define an equilibrium selection mechanism. Define the three equilibrium types as "Low-P", "Medium-P", and "High-P". It is clear that "Medium-P" is always an unstable equilibrium while "Low-P" and "High-P" are stable equilibria. The equilibrium selection mechanism is such that: (1) the unstable equilibrium is never selected; and (2) when market size is relatively small the Low-P equilibrium is chosen, while the High-P equilibrium is selected when market size is large. More specifically,

$$\tau^0(x_t) = \begin{cases} \text{Low-P} & if \quad x_t \leq 0.5 \\ \text{High-P} & if \quad x_t > 0.5 \end{cases} \tag{19}$$

The idea behind this selection mechanism is that it is easier to collude (i.e., Low-P equilibrium) in small markets than in large markets. Figure 3 presents the population probability function $P^0(B|x_t)$ that results from this model.

The number of sample observations is $T = 500$ and the number of Monte Carlo samples is $1,000$. For each sample we obtain five estimators of structural parameters.

> *(a) MLE.* Taking into account the form of $\tau^0(.)$ we can implement a nested fixed-point algorithm in the spirit of Rust (1987). For each trial value of $\theta$ and each sample observation $x_t$ we calculate an equilibrium by iterating in the best response probability mapping. If $x_t \leq 0.5$, we initialize the iterations with a probability equal to zero and therefore we converge to the Low-P equilibrium associated with $(x_t, \theta)$. If $x_t > 0.5$, we initialize the iterations with a probability equal to one and therefore we converge to the High-P equilibrium associated with $(x_t, \theta)$. For some values of $(x_t, \theta)$ the equilibrium is unique and therefore the Low-P and the High-P equilibria are the same. Notice that we can implement this estimator only because we know $\tau^0(.)$ and we know the characteristics of the equilibria associated with every value $(x_t, \theta)$.

> *(b) Two-Stage PML:* We use a Nadaraya-Watson estimator to obtain the initial estimates of the choice probabilities.

*(c) Single NPL:* We also initialize it with the Nadaraya-Watson estimator of choice probabilities.

*(d) Best of Five NPLs:* We generate five bootstrap samples of size $T = 500$ and use each of these samples to obtain a NPL fixed point. Then, we choose the NPL fixed-point with the highest value of the pseudo likelihood function.

*(e) NPL-Genetic Algorithm:* We initialize this algorithm using the same five bootstrap samples as for the estimator in (d). Therefore, we choose $M = 5$. The number of offsprings is $O = 10$. The parameter that measures the dependence of selection on fitness is $\varphi = 1$. The probability of mutation is $\gamma = 0.1$. And the parameter that represents the magnitude of the mutation, $\delta$, is equal to $(0.8)^K$ where $K$ is the number of iterations. Therefore, $\delta$ goes to zero as the number of iterations increase.

## 5.2 Results

Tables 1 and 2.present the results of the Monte Carlo experiment. First, it is important to notice that for all 7,000 applications of the NPL method in this experiment, as well as for others not reported here, the NPL always converged. However, in contrast with results reported by Aguirregabiria and Mira (2002 and 2004) for dynamic models, we converge to different NPL fixed points if we initialize the algorithm with different probabilities.

Table 1 compares the ability of the three NPL methods to obtain the MLE. The Single-NPL reaches the MLE in 764 of the 1,000 Monte Carlo replications. The Best-of-Five-NPL performs only slightly better than the Single-NPL. It reaches the MLE only 23 times more than the Single-NPL. However, the NPL-GA performs much better and provides the MLE for almost every sample, i.e., 97.1% of the times. This result confirms that the GA is an efficient method to obtain the global optimum over the NPL fixed-points.

Table 2 reports finite sample biases, standard deviations and mean square errors of the different estimators. These statistics have been calculated using the $1,000$ Monte Carlo replications. These statistics show that the ability of the NPL-GA to reach the MLE has clear and important implications on the finite sample performance of this estimator. According to these results, the finite sample distribution of the NPL-GA estimator is very close to the of the MLE and much closer to the true parameters than the one the Single-NPL or the Best-of-Five-NPL. Notice also that the Best-of-Five-NPL estimator is very similar in terms of bias and variance to the Single-NPL. The experiment also confirms the very poor finite sample properties of the two-stage PML, a result that has been reported in previous Monte

Carlo studies by Hotz, Miller, Sanders and Smith (1994), and Aguirregabiria and Mira (2002 and 2004).

# 6 Conclusions

This paper proposes a new algorithm to obtain maximum likelihood estimates of structural parameters in empirical games with multiple equilibria. The algorithm is based on the nested pseudo likelihood (NPL) method proposed by Aguirregabiria and Mira (2002 and 2004). The NPL avoids the repeated computation of all the equilibria of the game for each trial value of the structural parameters. In general, the maximum likelihood estimator is a fixed-point of the NPL algorithm. However, the NPL can have other fixed-points which are not the maximum likelihood estimator. A possible approach to obtain the MLE is to initialize the NPL algorithm with a number $M > 1$ of different vectors of probabilities, obtain $M$ fixed-points of the NPL, and then choose the one with the highest value of the likelihood function. The main limitation of this approach is that we may need a large number of initial values to guarantee that this estimator is the MLE. Instead, we propose here to combine the NPL with a genetic algorithm (GA). This hybrid algorithm starts also with a set of $M$ different probabilities but it does not apply the NPL separately to each of these $M$ probabilities. Instead, at each iteration the GA combines the information of $M$ candidates and applies the concept of *selection of the fittest* to generate a new set of $M$ candidates. This feature of the algorithm implies a more global search for the MLE.

We have illustrated the application of this algorithm in the context of a game of price competition and collusion. Though the model is relatively simple, with only two players, one exogenous explanatory variables and three equilibria, the standard approach to compute the MLE is practically unfeasible even in this model. We show that NPL, without GA, does reach the MLE when we initialize it with probabilities which are not too far away from the actual MLE estimates. However, when the initial probabilities are not good enough, the NPL converges to a fixed-point that is not the MLE. In contrast, the NPL-GA works extremely well in this example and reaches the MLE more than 97% of the times.

For the sake of simplicity, we have presented our algorithm in the context of a relatively simple model: a static game where all common knowledge variables are observable to the researcher. However, it is possible to use the results in Aguirregabiria and Mira (2004) to apply the NPL-GA to dynamic games with state variables that are unobservable to the econometrician.

## REFERENCES

Aguirregabiria, V., and P. Mira (2002): "Swapping the Nested Fixed Point Algorithm: A Class of Estimators for Markov Decision Models," *Econometrica,* 70, 1519-43.

Aguirregabiria, V., and P. Mira (2004): "Sequential Estimation of Dynamic Discrete Games," manuscript, Department of Economics, Boston University.

Bajari, P., L. Benkard and J. Levin (2004): "Estimating dynamic models of imperfect competition," manuscript, Stanford University.

Beenstock M. and G. Szpirob (2002): "Specification Search in Nonlinear Time-Series Models Using the Genetic Algorithm," *Journal of Economic Dynamics and Control,* 26, 811-835

Dorsey, R. E. and W. J. Mayer (1995): "Genetic Algorithms for Estimation Problems with Multiple Optima, Nondifferentiability, and Other Irregular Features," *Journal of Business and Economic Statistics,* 13, 53-66.

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

Hotz, J., R.A. Miller, S. Sanders and J. Smith (1994): "A Simulation Estimator for Dynamic Models of Discrete Choice," *Review of Economic Studies,* 61, 265-289.

Judd, K. (1998): "Numerical Methods in Economics," The MIT Press. Cambridge, Massachusetts.

Mitchell, M. (1996): "An Introduction to Genetic Algorithms," MIT Press. Cambridge.

Mitchell, M., J. H. Holland and S. Forrest (1994): "When Will a Genetic Algorithm Outperform Hill Climbing?" In J. D. Dowan, G. Tesauro, and J. Alspector, eds., *Advances in Neural Information Processing Systems 6*, Morgan Kauffman.

Pakes, A., M. Ostrowsky, and S. Berry (2004): "Simple Estimators for the Parameters of Discrete Dynamic Games (with Entry/Exit Examples)," manuscript, Harvard University.

Pesendorfer, M. and Schmidt-Dengler (2004): "Identification and Estimation of Dynamic Games," London School of Economics Working Paper.

Rust, J. (1987): "Optimal replacement of GMC bus engines: An empirical model of Harold Zurcher," *Econometrica,* 55, 999-1033.

Tamer, E. (2003): "Incomplete Simultaneous Discrete Response Model with Multiple Equilibria ," *Review of Economic Studies*, 70, 147-166.

**Table 1**
**Proportion of Monte Carlo Simulations in which NPL algorithms reach the MLE**

| *Estimator / Algorithm* | *Percentage* |
|---|---|
| *Single NPL* | 76.4 % |
| *Best of 5 NPLs* | 78.7 % |
| *NPL-Genetic Algorithm* | 97.1 % |

**Table 2**
**Bias, Std. Deviation, and Mean Square Error**

**BIAS**

| *Estimator / Algorithm* | *Parameters* | | | |
|---|---|---|---|---|
| | $\theta_{00}$ | $\theta_{01}$ | $\theta_{10}$ | $\theta_{11}$ |
| *Two-stage PML* | 0.190 | -6.110 | 5.736 | 0.319 |
| *Single NPL* | -0.107 | 0.424 | -0.002 | -0.316 |
| *Best of 5 NPLs* | -0.085 | 0.366 | -0.002 | -0.311 |
| *NPL-Genetic Algorithm* | -0.046 | 0.107 | 0.103 | -0.169 |
| *MLE* | -0.044 | 0.102 | 0.104 | -0.156 |

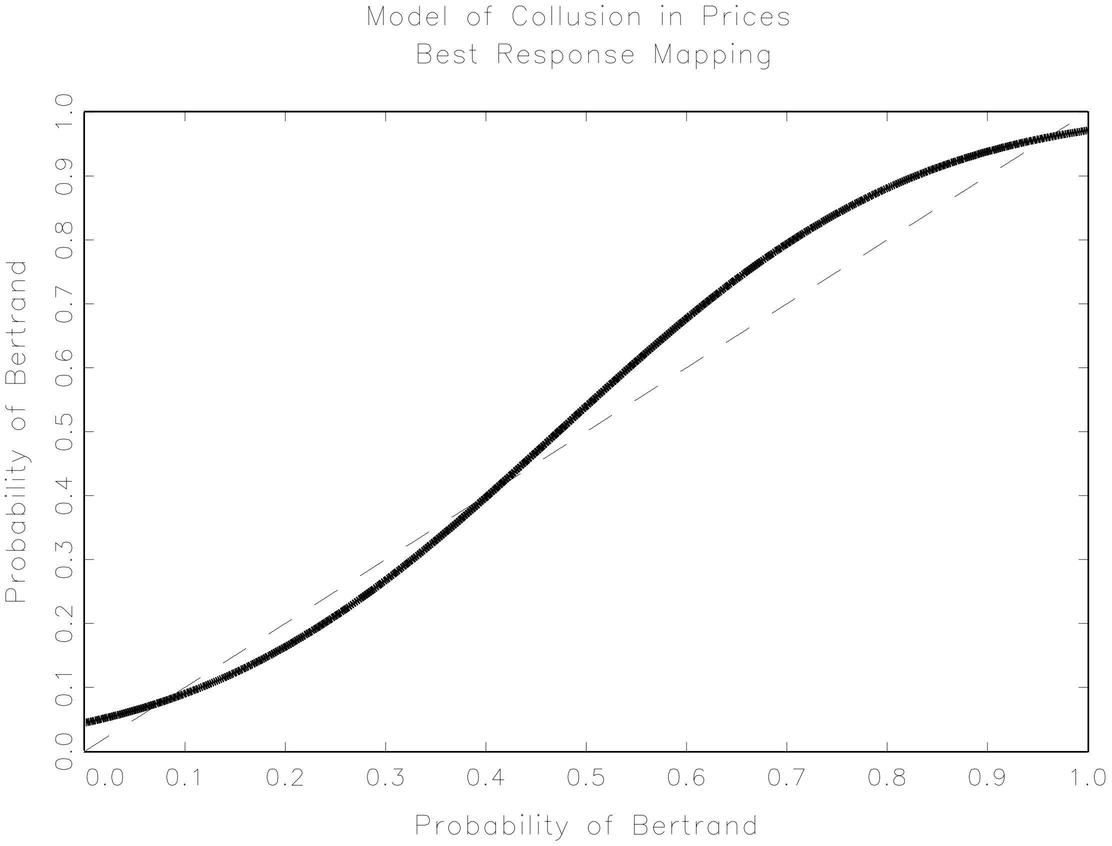
**STANDARD DEVIATION**

| *Estimator / Algorithm* | *Parameters* | | | |
|---|---|---|---|---|
| | $\theta_{00}$ | $\theta_{01}$ | $\theta_{10}$ | $\theta_{11}$ |
| *Two-stage PML* | 0.234 | 0.981 | 0.638 | 0.578 |
| *Single NPL* | 0.240 | 0.579 | 0.319 | 0.592 |
| *Best of 5 NPLs* | 0.238 | 0.576 | 0.319 | 0.590 |
| *NPL-Genetic Algorithm* | 0.201 | 0.506 | 0.301 | 0.536 |
| *MLE* | 0.201 | 0.506 | 0.301 | 0.536 |

**SQUARE ROOT OF MSE**

| *Estimator / Algorithm* | *Parameters* | | | |
|---|---|---|---|---|
| | $\theta_{00}$ | $\theta_{01}$ | $\theta_{10}$ | $\theta_{11}$ |
| *Two-stage PML* | 0.301 | 6.188 | 5.771 | 0.660 |
| *Single NPL* | 0.263 | 0.718 | 0.319 | 0.671 |
| *Best of 5 NPLs* | 0.253 | 0.682 | 0.319 | 0.676 |
| *NPL-Genetic Algorithm* | 0.206 | 0.517 | 0.318 | 0.562 |
| *MLE* | 0.206 | 0.516 | 0.318 | 0.558 |

## FIGURE 1

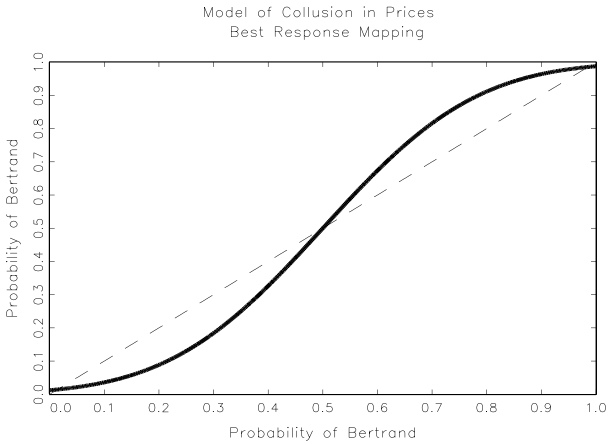


Best response mapping: $P = \Phi\left(-1.7 + 3.6\ P\right)$

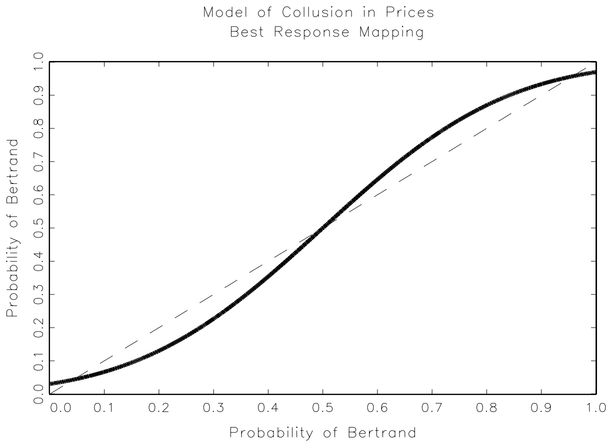
| **Equilibrium type** | **Equilibrium Probability** |
|---|---|
| Low-P | 0.078 |
| Medium-P | 0.407 |
| High-P | 0.961 |

**FIGURE 2.** Parameter Values: $\theta_{00} = 2.25$ ; $\theta_{01} = 0.75$ ; $\theta_{10} = 4.50$ ; $\theta_{11} = 1.50$

**(A) Market Size $x_t = 0.0$. Equilibrium Probabilities: 0.014; 0.500; 0.986**



**(B) Market Size $x_t = 0.5$. Equilibrium Probabilities: 0.044; 0.500; 0.957**



**(C) Market Size $x_t = 1.0$. Equilibrium Probabilities: 0.140; 0.500; 0.860**

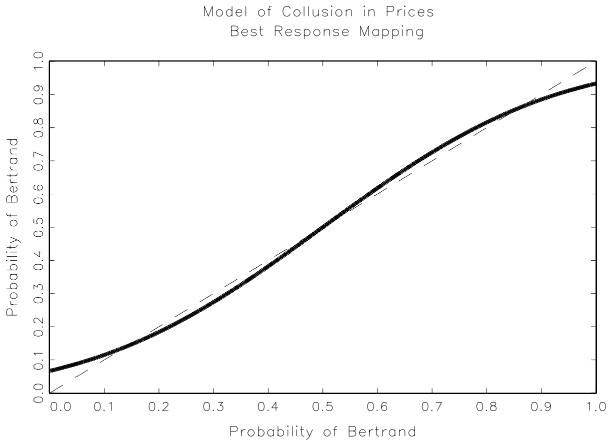
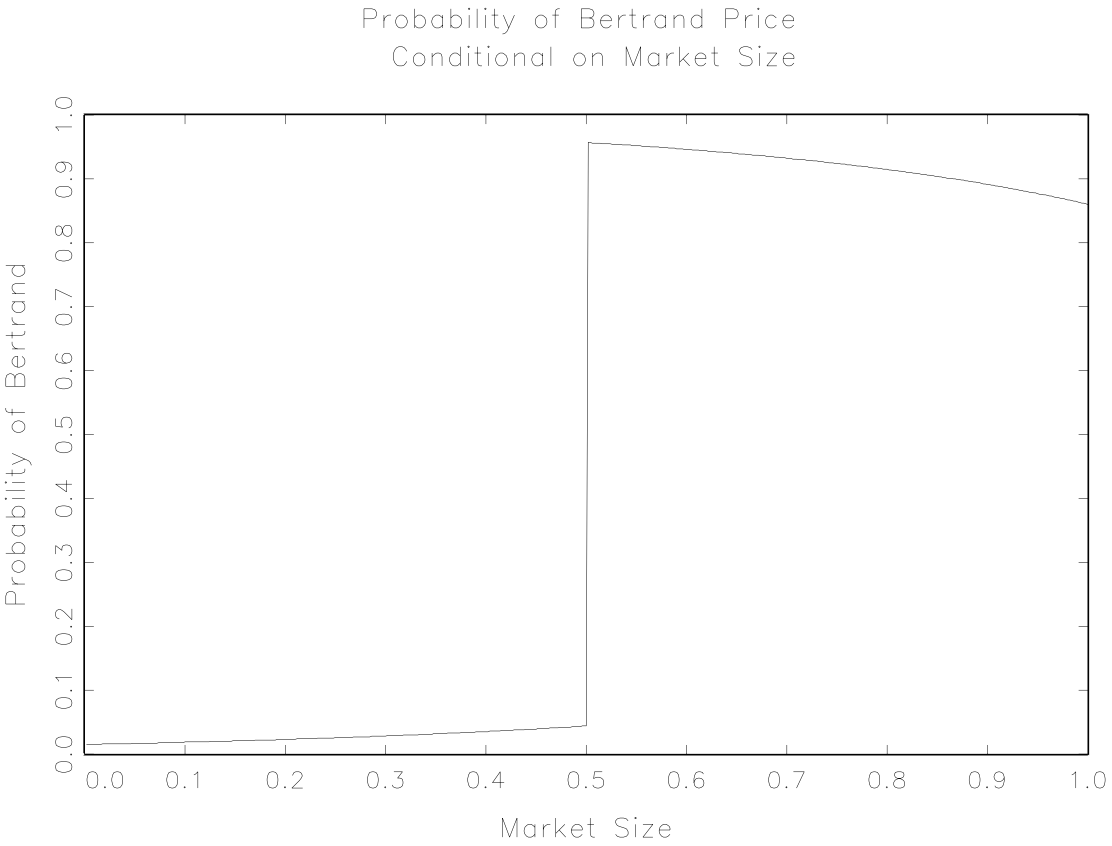


**FIGURE 3**


Probability of Bertrand Price
Conditional on Market Size
Probability of Bertrand
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
Market Size
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0