An Optimization Formulation to Compute Nash Equilibrium in Finite Games

Bapi Chatterjee
Department of Mathematics
Indian Institute of Technology Delhi
Hauz Khas, New Delhi-110016, India
Email: bhaskerchatterjee@gmail.com

***Abstract*—In this paper, we consider a finite n-person non co-operative game. A non-linear optimization model is formulated in a space of dimension equal to sum of the total number of pure strategies in the game and the number of players. A Nash equilibrium of the given game is shown to be equivalent to an optimal solution of the optimization model with zero optimal value. The algorithm is coded in MATLAB using sequential quadratic programming based quasi Newton technique. To make the algorithm user-friendly, a GUI enabled software is developed. Performance of the software is analyzed. Our computational experience has been very satisfactory.**

***Index Terms*—Game theory, Nash equilibrium, n-person matrix game, nonlinear constrained optimization.**

I. Introduction

Beginning from the seminal work of John Nash, it has now been well recognized that a predictable solution in a conflicting situation is a Nash equilibrium. Since its inception and proof of its existence using the Brouwer's fixed point theorem, several researchers proposed different numerical schemes to compute Nash equilibrium. Most of these methods are for games involving two players only and there has been less emphasis on games involving more than two players. The trend is justified because as far as applications of non cooperative games are concerned, we hardly see modeling of games with more than two players in literature. The reason is not very hard to find, as whenever there are more than two persons involved in a conflict it is instinctive that two or more of them would start cooperating, even if not fully then at least partially to converge into two potential groups. Eventually the scenario turns out to be a two party game. Nevertheless, in recent time there has been a good progress in development of algorithms for n-person non cooperative games in normal form.

Starting from the work of Rosenmüller [10] and Wilson [12], who generalized the Lemke-Howson [8] algorithm for two person non co-operative matrix games to n-person non cooperative matrix games independently, there have been a number of approaches to find Nash equilibria in a game; Garcia, Lemke and Lüthi [2] gave a method of simplical approximation, Laan and Talman [11] gave an algorithm that operates directly in the strategy space that can be restarted at any point within that space. Recently Govindan and Wilson [3]–[5] proposed three different algorithms to compute Nash equilibrium in n-person games. All these cited works have their own distinct advantages and drawbacks from various points of views ranging from machine implementation to mathematical sophistication. For example, the methods given by Rosenmüller [10] and Wilson [12] merely prove the existence of a nonlinear path leading to a Nash equilibrium. These two methods have never been implemented as machine codings. The method given by Garcia *et al.* [2] is machine implementable but only provides an approximate solution. The algorithm proposed by Govindan and Wilson [3] is sophisticated but it uses heavy tools from homotopy and analytical dynamics. But here we shall be presenting an algorithm based on a nonlinear programming technique which has not only a simple mathematical formulation but also has a good machine implementation. We have shown that computation of a Nash equilibrium in an n-person non co-operative finite game is equivalent to a non-linear optimization problem in which the constraints and objective functions are polynomials, and the number of variables equals the sum of number of players and total number of available pure strategies in the game. To solve the optimization problem, the sequential quadratic programming based quasi Newton method has been used.

The paper is planned as follows. In section II, the preliminaries and formulation of an n-person matrix game are given. In section III, equivalent optimization formulation and related proofs are given. A software based on this equivalence is developed to compute a Nash equilibrium. In section IV, performance analysis of the software is presented. The last section includes conclusion.

II. Game Formulation

Let us first recall some definitions and related structures associated with a finite noncooperative matrix game. First we need to fix some notations.

A finite n-person non cooperative game in normal or strategic form is represented by a tuple

$$\Gamma = \left(N, \{S^i\}_{i\in N}, \{u^i\}_{i\in N}\right),$$

where N is a finite set of players, S^i is space of pure strategies of player i and u^i is the payoff function of player i.

If $N = \{1, 2, \ldots, n\}$, then $S = \prod_{i\in N} S^i$ is the space of possible pure strategies combinations in game Γ, and then $u^i : S \to \mathbb{R}$. Suppose player i has m^i pure strategies, then the number of pure strategies in the game is $\sum_{i=1}^{n} m^i$ and the number of pure strategies combinations in the game is

$\prod_{i=1}^{n} m^i$. We denote the two by m & M respectively.

For the purpose of representation of the game we arrange the pure strategies combinations by ranking them as follows.

$$\begin{aligned}(s_1^1, s_1^2, \ldots, s_1^{n-1}, s_1^n) &:= 1\\ (s_1^1, s_1^2, \ldots, s_1^{n-1}, s_2^n) &:= 2\\ &\vdots \quad\quad \vdots\\ (s_{m^1}^1, s_{m^2}^2, \ldots, s_{m^{n-1}}^{n-1}, s_{m^n-1}^n) &:= (M-1)\\ (s_{m^1}^1, s_{m^2}^2, \ldots, s_{m^{n-1}}^{n-1}, s_{m^n}^n) &:= M\end{aligned}$$

For instance, suppose a game has 3 players each having 2 pure strategies, that is, $n = 3$, $m^1 = m^2 = m^3 = 2$. Then number of pure strategies is $6(= m)$ and number of pure strategies combinations is $8(= M)$. The ranking of pure strategies combinations in this case is as follows:

$$\begin{aligned}(s_1^1, s_1^2, s_1^3) &:= 1\\ (s_1^1, s_1^2, s_2^3) &:= 2\\ (s_1^1, s_2^2, s_1^3) &:= 3\\ (s_1^1, s_2^2, s_2^3) &:= 4\\ (s_2^1, s_1^2, s_1^3) &:= 5\\ (s_2^1, s_1^2, s_2^3) &:= 6\\ (s_2^1, s_2^2, s_1^3) &:= 7\\ (s_2^1, s_2^2, s_2^3) &:= 8\end{aligned}$$

where s_j^i means j^{th} pure strategy of the i^{th} player.

With respect to each pure strategies combination a player has an associated payoff, so the payoff matrix of each player is realized as a vector of length M. For that matter, the input format of a non cooperative game with n players and each player having m^i, $i = 1, 2, \ldots, n$, pure strategies, so that the number of pure strategies combinations in the game is M, can be described as follows,

$$\begin{matrix} u_1^1 & u_1^2 & \ldots & u_1^n \\ u_2^1 & u_2^2 & \ldots & u_2^n \\ \vdots & \vdots & \vdots & \vdots \\ u_{M-1}^1 & u_{M-1}^2 & \ldots & u_{M-1}^n \\ u_M^1 & u_M^2 & \ldots & u_M^n \end{matrix}$$

A mixed strategy of player i is interpreted as a probability distribution over the space S^i and the space of all mixed strategies of player i is denoted by $\Sigma^i = \{\sigma^i \in \mathbb{R}^{m^i +} | \sum_{j=1}^{m^i} \sigma_j^i = 1\}$. For $\sigma^i \in \Sigma^i$, the probability assigned to pure strategy s_j^i is σ_j^i. The strategy space of the game is $\Sigma = \prod_{i \in N} \Sigma^i$.

If a mixed strategy combination σ is played then the probability that the pure strategies combination $s = (s_{j^1}^1, s_{j^2}^2, \ldots, s_{j^n}^n)$ occurs is given by $\sigma(s) = \prod_{i \in N} \sigma_{j^i}^i$. In such a situation the payoff assigned to player i is given by $u^i(\sigma) = \sum_{s \in S} \sigma(s) u^i(s)$, where $u^i(s)$ is the payoff to player i at the pure strategies combination s.

If σ^{-i} denotes the mixed strategy vector formed by all players except player i, then we can replace the mixed strategies combination σ by (σ^{-i}, σ^i). A central concern of a game is computation of Nash equilibrium, defined next.

Definition 1: **Nash equilibrium**: A mixed strategy profile σ^* is called a Nash equilibrium of the game Γ if

$$u^i(\sigma^*) \geq u^i(\sigma^{*-i}, \sigma^i), \ \forall i \in N, \ \forall \sigma^i \in \Sigma^i.$$

In other words it means that for each player i, she could not attain a better payoff than that at Nash equilibrium, by changing only her own mixed strategy leaving all others' strategies unchanged. Here the sense of optimization for each player is to maximize her payoff when others are playing by their Nash equilibrium strategies. Consequently, one attempts to minimize the gap between the optimal payoff and the payoff obtained by a possible mixed strategies combination. These ideas are explained in the section to follow.

III. Equivalent Optimization Formulation

Now if β^i is the optimal payoff to player i then the optimization problem of player i, $i \in N$ is

$$\begin{aligned}&(P^i)\\ &min \quad \beta^i - u^i(\sigma)\\ &s.t. \quad u^i(\sigma^{-i}, s_j^i) - \beta^i \leq 0 \quad \forall j = 1, \ldots, m^i\\ &\quad \sum_{j=1}^{m^i} \sigma_j^i = 1\\ &\quad \sigma_j^i \geq 0 \qquad \forall j = 1, \ldots, m^i\end{aligned}$$

where (σ^{-i}, s_j^i) denotes the mixed strategies combination in which player i plays with his j^{th} pure strategy, that is, a mixed strategy in which the j^{th} pure strategy of i^{th} player is assigned the probability 1.

Although convexities of problems (P^i), $i \in N$ are not assured, yet they definitely possess local optimal solutions. A local optimum satisfies KKT first order necessary conditions. Applying the KKT optimality conditions to the above problems we obtain following result which provides a simple characterization of Nash equilibrium of the game Γ as a solution to a system of equalities and inequalities.

Lemma 1: A necessary and sufficient condition for σ to be a Nash equilibrium of the game Γ is

$$\begin{aligned}\beta^i - u^i(\sigma) &= 0 \ \forall i \in N &(1)\\ u^i(\sigma^{-i}, s_j^i) - \beta^i &\leq 0 \ \forall j = 1, \ldots, m^i, \forall i \in N &(2)\\ \sum_{j=1}^{m^i} \sigma_j^i &= 1 \ \forall i \in N &(3)\\ \sigma_j^i &\geq 0 \ \forall j = 1, \ldots, m^i, \forall i \in N. &(4)\end{aligned}$$

It can easily be seen that if such a σ exists then it is nothing but an optimal solution of nonlinear programming problems (P^i), for $i \in N$, with global optimal value equals 0. We shall show that the Nash equilibrium strategy is an optimal solution of a single optimization problem.

Theorem 1: A necessary and sufficient condition for σ^* to be a Nash equilibrium of game Γ is that it is an optimal solution of the following minimization problem

$$
\begin{array}{ll}
(P) & \\
min & \sum_{i\in N}(\beta^i - u^i(\sigma)) \\
s.t. & u^i(\sigma^{-i}, s^i_j) - \beta^i \leq 0 \ \forall j = 1,\ldots,m^i, \forall i \in N \\
& \sum_{j=1}^{m^i} \sigma^i_j = 1 \qquad \forall i \in N \\
& \sigma^i_j \geq 0 \qquad \forall j = 1,\ldots,m^i, \quad \forall i \in N.
\end{array}
$$

The optimal value of this problem is 0. The value of β^i at the optimal point gives the expected payoff of the player i.

Proof: In light of Lemma 1, the feasible set of (P) is nonempty as for every finite non co-operative game a Nash equilibrium exists. Further let S be the feasible region for (P) then,

$$min_{(\sigma,\beta^1,\ldots,\beta^n)\in S} \sum_{i\in N}(\beta^i - \sum_{j=1}^{m^i} u^i(\sigma^{-i}, s^i_j)) \geq 0.$$

Thus, if σ^* is a Nash equilibrium it is feasible for (P), and from (1),

$$\sum_{i\in N}(\beta^{i*} - u^i(\sigma^*)) = 0,$$

yielding that σ^* is an optimal solution of (P).

Conversely, suppose $(\sigma^*, \beta^{1*}, \ldots, \beta^{n*})$ is an optimal solution of (P) then it satisfies (2) to (4).

By virtue of (2), $\sum_{i\in N}(u^i(\sigma^{-i*}, s^i_j) - \beta^{i*}) \leq 0$.

But by the existence theorem of Nash equilibrium, there must exist at least one $(\sigma, \beta^1, \ldots, \beta^n)$ feasible for (P) with $\sum_{i\in N}(\beta^i - u^i(\sigma)) = 0$. So for $(\sigma^*, \beta^{1*}, \ldots, \beta^{n*})$ to be a global minimum for (P),

$$\sum_{i\in N}(u^i(\sigma^*) - \beta^{i*}) = 0$$

Consequently σ^* is a Nash equilibrium of game Γ, on account of Lemma 1 the payoff β^{i*} is obviously the optimal expected payoff to player i. □

We see that the problem of computing a Nash equilibrium of Γ reduces to that of solving the optimization problem (P) with optimal value zero. In order to solve problem (P), we need to reformulate it in terms of a vector in $m + n$ dimensional Euclidean space. For that we do the necessary change of variables as follows.

Take $\mathbf{x}$ to be a vector of length $m + n$ described as follows. Arranging the strategies of players 1 to n in order, we have a total of m strategies and we take x_i's in order as: $x_1 = \sigma^1_1, x_2 = \sigma^1_2, \ldots, x_{m^1} = \sigma^1_{m^1}, \ldots, x_m = \sigma^n_{m^n}$, where subscripts in σ denote the strategies and superscripts stand for the players. Then take $x_{m+i} = \beta^i$, $i = 1, 2, \ldots, n$. Performing this transformation of variables in (P), the optimization problem (P) gets converted to the following form

$$
\begin{array}{ll}
(P^{new}) & \\
min & f(\mathbf{x}) \\
s.t. & g(\mathbf{x}) \leq 0 \\
& h(\mathbf{x}) = 0 \\
& x_i \geq 0 \quad \forall\ i = 1,\ldots,m \\
& x_i \ \ are \ \ unrestricted \\
& \forall\ i = m+1, m+2 \ldots, m+n
\end{array}
$$

where,

$$
\begin{array}{lcl}
f(\mathbf{x}) & = & \sum_{i\in N}(\beta^i - u^i(\sigma)) \\
g(\mathbf{x}) & = & u^i(\sigma^{-i}, s^i_j) - \beta^i \quad \forall j = 1, 2, \ldots, m^i, \forall i \in N \\
h(\mathbf{x}) & = & \sum_{j=1}^{m^i} \sigma^i_j - 1 \quad \forall i \in N
\end{array}
$$

To get a solution of this nonlinear minimization problem with nonlinear constraints we use the sequential quadratic programming (SQP) based quasi Newton method. For a detailed description of the method we refer to [1]. Here we present the steps for the algorithm.

1) Represent the game in the form described in section II.
2) Rank the possible pure strategies combinations as desired and explained in section II.
3) Take variables x_1 to x_{m+n} and form the optimization model (P^{new}).
4) Solve the problem (P^{new}) using SQP based quasi Newton method.

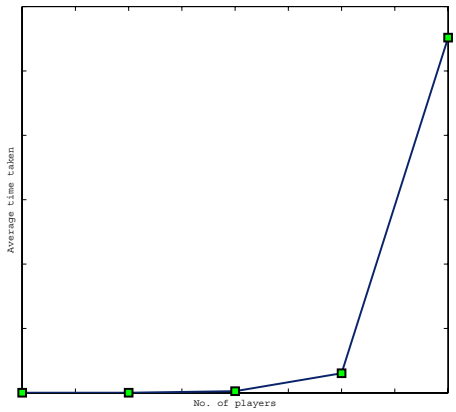
IV. Software: N-Person Game

The algorithm presented above has been coded in MATLAB and further to make it user-friendly a GUI has been designed in Visual Basic 2008 on Microsoft .NET framework 3.5. Emphasis has been put on the code optimization and presentation of the GUI. We have named the software N-Person Game.

We have tested the software on Pentium 2.66 GHz machine on Microsoft Windows XP platform. Performance of the software is as described below.

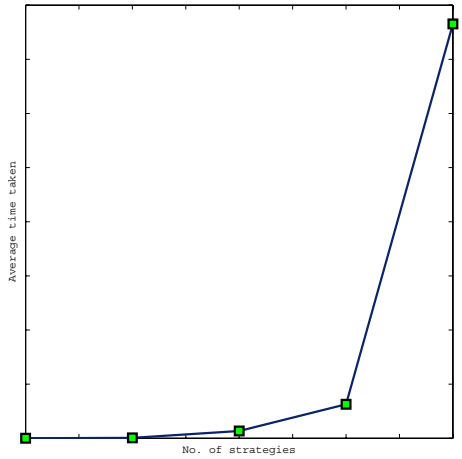
1) In the first case number of strategies are fixed and number of players vary. We have kept number of strategies = 5. Ten games with entries randomly generated in $(0, 1)$ are solved in each case and the average time taken are tabulated. Also the variation of *number of players* vs. *average time taken* has been shown graphically.

No. of players	Average time (Sec)
2	0.140625
3	2.003125
4	49.978125
5	606.875000
6	11034.809765



2) In the second case number of players are fixed and number of strategies vary. We have kept number of players = 5. Ten games with entries randomly generated in $(0, 1)$ are solved in each case and the average time taken are tabulated. Also the variation of *number of strategies* vs. *average time taken* has been shown graphically.

No. of strategies	Average time (Sec)
2	0.6625
3	7.29065
4	134.101563
5	628.234095
6	7654.234



We observe that with the increase in number of players, while keeping the number of strategies fixed (or vice versa), there is a sharp increase in average time taken by our scheme to compute a Nash equilibrium. This behavior essentially shows that the time complexity of computing a Nash equilibrium is not of polynomial order. This conclusion is based on only empirical evidences. We have also compared the performance of N-Person Game with that of the standard Gambit [9] software. The results are listed below.

No. of players	Strategies/player	Time by Gambit (Sec)	Time by N-Person Game (Sec)
2	5	0.093251	0.109375
	10	2.015625	2.005125
	20	22.203125	21.1875
3	3	0.203125	0.234375
	5	4.046875	3.840216
	10	62.578125	55.023065
4	3	1.375	1.000203
	5	48.921875	43.05103
5	2	0.6754	0.6375
	3	23.234375	23.3125
6	2	2.65625	2.640625
7	2	11.80236	12.21875
8	2	25.984375	26.125

From these analyses we can conclude that our software shows fairly satisfactory results and in fact in small dimensional setting of a game it is at par with Gambit for computing a Nash equilibrium.

V. Conclusion

In this paper a method is proposed to compute a Nash equilibrium of a finite non co-operative game via an optimal solution of a nonlinear optimization model rather than a path following method which has been a trend in last couple of decades. We definitely can not claim it to have an economic underpinning as given by Harsanyi-Selton in their famous linear tracing procedure [6], [7]. Our method does not talk about an equilibrium selection but attempts to find one equilibrium out of many which is best in the sense that all the players have optimized their payoffs rather than adjusted to their common beliefs about other players in the game.

Without any loss of generality we can take all payoffs in a game to be positive. In that case, with small manipulation, the optimization model developed here will have constraints and objective function as posynomials [1]. With that the optimization problem gets transformed into a geometric programming problem. As geometric programming problems containing large

[1]A posynomial is a function of the form $f(x_1, x_2, \ldots, x_n) = \sum_{k=1}^{K} c_k x_1^{a_{1k}} \cdots x_n^{a_{nk}}$, where all the coordinates x_i and coefficients c_k are positive real numbers, and the exponents a_{ik} are real numbers.

number of variables and constraints can be solved more efficiently, it is worthwhile to explore the application of geometric programming to solve the proposed optimization model.

REFERENCES

[1] R. Flethcher, *Practical Methods of Optimization*. John Wiley and Sons, 2004, ch. Nonlinear Programming, pp. 304–317.

[2] C. B. Garcia, C. E. Lemke, and H. J. Lüthi, *Mathematical Programming*. New York: Academic Press, 1973, ch. Simplical Approximation of an Equilibrium Point of Noncooperative N-Person Games, pp. 227–260.

[3] S. Govindan and R. Wilson, "A global Newton method to compute Nash equilibria," *Journal of Economic Theory*, vol. 110, pp. 65–86, 2003.

[4] S. Govindan and R. Wilson, "Computing Nash equilibria by iterated polymatrix approximation," *Journal of Economic Dynamics and Control*, vol. 28, pp. 1229–1241, 2004.

[5] S. Govindan and R. Wilson, "A decomposition algorithm for n-player games," *Stanford University Graduate School of Business*, no. 1967, August 2007.

[6] J. Harsanyi, "The tracing procedure: a Baysian approach to defining a solution for n-person non co-operative games," *International Journal of Game Theory*, vol. 4, pp. 61–94, 1975.

[7] J. Harsanyi and R. Selton, *A general theory of equilibrium selection in games*. Cambridge: MIT press, 1988.

[8] C. E. Lemke and J. T. Howson, "Equilibrium points in bimatrix games," *SIAM Journal on Applied Mathematics*, vol. 12, pp. 413–423, 1964.

[9] R. D. McKelvey, A. M. McLennan, and T. L. Turocy, "Gambit: Software tools for game theory, version 0.2007.01.30," *Available at: http://www.gambit.sourceforge.net/*, 2007.

[10] J. Rosenmüller, "On a generalization of the Lemke-Howson algorithm to non co-operative n-person games," *SIAM Journal on Applied Mathematics*, vol. 21, pp. 80–87, 1971.

[11] G. van der Laan and A. J. J. Talman, "On the computation of fixed points on the product space of unit simplices and an application to noncooperative n-person games," *Mathematics of Operations Research*, vol. 7, pp. 1–13, 1982.

[12] R. Wilson, "Computing equilibria of n-person games," *SIAM Journal on Applied Mathematics*, vol. 21, pp. 73–79, 1971.