ОБОЗРЕНИЕ
ПРИКЛАДНОЙ И ПРОМЫШЛЕННОЙ
Том 23 МАТЕМАТИКИ Выпуск 2
2016

II INTERNATIONAL BALTIC SYMPOSIUM ON APPLIED AND INDUSTRIAL MATHEMATICS

E. Golshteyn, **U. Malkov**, **N. Sokolov** (Moscow, CEMI RAS). **On the experience of the numerical solution of finite three-person games.**

In [1], we presented an approximate algorithm solving finite non-cooperative three-person games in mixed strategies. The latter method is based on an iterative search for a global minimum of the Nash function, where each iteration consists of solving three linear programming problems. In this talk, we deal with the algorithm's efficiency and testing procedures (also, cf. [2]).

Consider a finite non-cooperative three-person game Γ of the following structure. Assume that player l, $l=1,2,3$, governs n_l strategies, and a table $(a^{(l)}_{ijk})$ determines its payoff whenever the players select strategies i $(1\leqslant i\leqslant n_1)$, j $(1\leqslant j\leqslant n_2)$, and k $(1\leqslant k\leqslant n_3)$, respectively. Since the game Γ is finite the set of Nash points X^* is non-empty even though not necessarily convex. We do not suppose [1] any convex structure for the game Γ.

The finite three-person game Γ in the mixed strategies is determined by the players' payoff functions

$$f_l(x)=f_l(x_1,x_2,x_3)=\sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\sum_{k=1}^{n_3}a^{(l)}_{ijk}x_{1i}x_{2j}x_{3k},\qquad l=1,2,3,$$

defined over the compact subset $X=X^{(1)}\times X^{(2)}\times X^{(3)}$ of the Euclidean space $\mathbf{E}^{n_1+n_2+n_3}$, where $X^{(l)}=\{x_l=(x_{l1},x_{l2}\ldots,x_{ln_l})\in\mathbf{E}^{n_l}:\ \sum_{r=1}^{n_l}x_{lr}=1,\ \ x_{lr}\geqslant 0,\ \ r=1,2,\ldots,n_l\}$, $l=1,2,3$.

It turns out that solving the game Γ is tantamount to the search of the global minimum (zero) of the standard Nash function $F(x_1,x_2,x_3)=\sum_{l=1}^{3}\delta_l(x)$, $x\in X$, where $\delta_l(x)=\max_{x_l\in X^{(l)}}f_l(x)-f_l(x)$, $l=1,2,3$.

Although the minimization of the function $F(x)$ by $x\in X$ is an extremely complicated task (mainly due to its numerous local minima distinct from the global one), the problem of minimization of this function along a strategy vector of only one player (with the strategies of the other two players being fixed) is easily reduced to a linear program. Indeed, having fixed an arbitrary pair of strategy vectors, say, $x'_2\in X^{(2)}$, $x'_3\in X^{(3)}$, we find an optimal solution x^*_1 of the following linear programming problem denoted as $\mathcal{P}_1(x'_2,x'_3)$:

$$\begin{aligned}
&\sum_{i=1}^{n_1}\Bigl(-\sum_{j=1}^{n_2}\sum_{k=1}^{n_3}\sum_{l=1}^{3}a^{(l)}_{ijk}x'_{2j}x'_{3k}\Bigr)x_{1i}+\alpha^{(1)}_1+\alpha^{(1)}_2\to\min,\\
&\sum_{i=1}^{n_1}\Bigl(\sum_{j=1}^{n_2}a^{(3)}_{ijk}x'_{2j}\Bigr)x_{1i}\leqslant\alpha^{(1)}_1,\qquad k=1,2,\ldots,n_3,\\
&\sum_{i=1}^{n_1}\Bigl(\sum_{k=1}^{n_3}a^{(2)}_{ijk}x'_{3k}\Bigr)x_{1i}\leqslant\alpha^{(1)}_2,\qquad j=1,2,\ldots,n_2,\\
&\sum_{i=1}^{n_1}x_{1i}=1,\quad x_{1i}\geqslant 0,\quad i=1,2,\ldots,n_1,\quad \alpha^{(1)}_1,\alpha^{(1)}_2\in\mathbf{E}^1.
\end{aligned}$$

As a consequence, the inequality $F(x_1^*, x_2', x_3') \leqslant F(x_1', x_2', x_3')$ holds for any strategy $x_1' \in X^{(1)}$. Similarly, having fixed the values of the other feasible pairs of strategy vectors $(x_1', x_3') \in X^{(1)} \times X^{(3)}$ and $(x_1', x_2') \in X^{(1)} \times X^{(2)}$, we detect the optimal solutions x_2^* and x_3^* for the corresponding linear programs $\mathcal{P}_2(x_1', x_3')$ and $\mathcal{P}_3(x_1', x_2')$. In the same manner, one has the inequalities $F(x_1', x_2^*, x_3') \leqslant F(x_1', x_2', x_3')$ holding for an arbitrary strategy $x_2' \in X^{(2)}$, and $F(x_1', x_2', x_3^*) \leqslant F(x_1', x_2', x_3')$ for any $x_3' \in X^{(3)}$, respectively.

Further, we are going to generate the sequences of the strategy pairs for which the Nash function is monotone non-increasing. In the developed algorithm, we start with the pure strategy pairs. For example, starting with an arbitrary pure strategy pair $h = (x_2^0, x_3^0) \in X^{(2)} \times X^{(3)}$, we generate the following sequence of the triples $x^t = (x_1^t, x_2^t, x_3^t),\ t \geqslant 1:$

$$\begin{array}{ll} x_1^{t+1} \text{ — solves program } \mathcal{P}_1(x_2^t, x_3^t), & \\ x_2^{t+1} \text{ — solves program } \mathcal{P}_2(x_1^{t+1}, x_3^t), & t = 0, 1, 2, \ldots \\ x_3^{t+1} \text{ — solves program } \mathcal{P}_3(x_1^{t+1}, x_2^{t+1}), & \end{array}$$

By the analogous procedures, one can construct the corresponding sequences starting from two remaining pure strategy pairs. It is obvious that $0 \leqslant F(x^{t+1}) \leqslant F(x^t)$ for all $t \geqslant 1$, therefore, there exists a limit $q(h) = \lim_{t\to\infty} F(x^t) \geqslant 0$. As an approximate value $\widetilde{q}(h)$, here we accept (just as in [1]) the value of $F(x^{t(h)})$, where $t(h) = \min\{t \geqslant 0 : F(x^t) - F(x^{t-1}) \leqslant \varepsilon_F\}$, and ε_F is a small fixed positive tolerance parameter (for example, 10^{-8}). In [2], we also introduced a maximum allowed number of iterations (triples of linear programs) T, that is, we set $\widetilde{q}(h) = F(x^T)$ whenever the corresponding estimate is broken for all $t \leqslant T$.

In [1], we define an approximate value of game as $\widetilde{q}(H) = \min_{h\in H} \widetilde{q}(h)$, where H is the set of all starting pure strategy pairs, hence $|H| = n_2 n_3 + n_1 n_3 + n_1 n_2$. Both in [2] and here, instead of the exhausting search over all starting pairs, we tested the algorithm by solving the following problem. Based on the (ordered) set of starting strategy pairs H we look for a strategy $x \in X$, for which the stopping rule $N(x) \leqslant \varepsilon$ holds, where ε is a small fixed tolerance parameter, say, 10^{-6}, while $N(x)$ is a modified (normed) Nash function (MNF) of the form $N(x) = \max_{1\leqslant l\leqslant 3}\{\delta_l(x)/f_l(x)\},\ x \in X$.

We have tested the algorithm with the aid of the code trimatrix.exe developed by the authors in the language FORTRAN. The code has been translated with the Compaq Visual Fortran compiler. The body of trimatrix.exe assembled/linked in the MS Visual Studio 6.0 environment. The procedure linprog (authored by U. Malkov) is exercised as the linear programming solver. To justify their comparison, all the results provided below have been obtained on the same personal computer with a processor Intel(R)Core(TM)i7CPU920/2,67 GHz and the 6,00 GB memory. All the data is stored in the operative memory.

The algorithm has been tested on a family of 3-person games comprising 6 series of different dimensions defined by the tabular parameters $n_1 = n_2 = n_3 = n \in \{10, 20, \ldots, 60\}$. For each series, $P = 10$ games were randomly generated. For every game, the 3-dimensional tables were generated in two steps. First, the numbers $\overline{a}_{ijk}^{(1)}$, $\overline{a}_{ijk}^{(2)}$, and $\overline{a}_{ijk}^{(3)}$ were randomly generated as independent random variables distributed uniformly in the interval [0, 1]. After that, the following transformation was applied:

$$a_{ijk}^{(l)} = 1 + \overline{a}_{ijk}^{(l)} - \frac{\varkappa}{2} \sum_{\substack{r=1 \\ r\neq l}}^{3} \overline{a}_{ijk}^{(r)}, \quad l = 1, 2, 3, \quad i, j, k \in \{1, 2, \ldots, n\}.$$

Here, $\varkappa$ is the *the mutual correlation coefficient* among the generated tables, $0 \leqslant \varkappa \leqslant 1$. Therefore, the tabular entries are running within the interval $(1 - \varkappa, 2)$.

First, the algorithm was tested on independent tables with $\varkappa = 0$, and its efficiency has proved to be rather high.

Table 1. Results of solving three-person games for the mutual correlation coefficient $\varkappa \in \{0.1,\ 0.3,\ 0.5,\ 0.9\}$

$\varkappa$	№	Size	Result			Start pairs		LP-iterations				Nash function			Mutual compl.		Time
		n	k_0	k^+	k^-	s^+_{av}	$s^+_{\max}$	t^+_{av}	$t^+_{\max}$	t^-_{av}	$t^-_{\max}$	$N^-_{\min}$	N^-_{av}	$N^-_{\max}$	$\Delta^-_{\min}$	$\Delta^-_{\max}$	T_{sum} sec
0.1	1	10	2	8	0	74.87	232	4.82	33								6.3
	2	20	0	10	0	138.10	229	6.54	62								46.9
	3	30	1	9	0	407.67	659	7.94	337								358.7
	4	40	0	10	0	273.20	1234	8.94	152								647.7
	5	50	0	10	0	371.70	762	9.71	102								1759.1
	6	60	0	10	0	850.60	3209	10.34	130								7666.1
0.3	1	10	0	7	3	175.86	266	5.47	63	5.57	38	106.57	130.85	153.39	1	1	27.0
	2	20	0	3	7	259.33	393	8.56	44	10.87	3625	17.612	119.74	258.29	1	2	553.2
	3	30	0	3	7	1426.33	1954	11.61	1814	11.74	2020	1.4406	127.93	641.84	1	2	4082.7
	4	40	0	1	9	2699.00	2699	13.29	290	13.72	2077	2.3934	54.640	225.08	1	3	20993.9
	5	50	0	1	9	4287.00	4287	15.47	173	15.69	1070	1.7801	80.235	174.43	1	3	71846.3
	6	60	0	0	10					17.51	808	12.573	89.710	190.92	2	3	218169.0
0.5	1	10	0	3	7	116.00	132	6.45	46‘	7.85	145	11.530	949.58	1687.8	1	2	44.4
	2	20	0	0	10					10.66	1731	4.7492	773.49	1279.3	1	3	785.7
	3	30	0	0	10					14.04	1619	160.99	595.63	981.93	1	7	6602.1
	4	40	0	0	10					16.84	1512	193.92	581.96	819.37	3	6	33435.2
	5	50	0	0	10					19.53	567	196.29	555.19	688.86	4	11	118605.9
	6	60	0	0	10					22.23	482	307.15	494.56	612.30	6	11	357086.0
0.9	1	10	0	1	9	42.00	42	5.19	15	6.28	163	774.16	2047.1	3024.4	1	3	15.1
	2	20	0	1	9	588.00	588	9.73	85	10.55	128	602.13	1296.5	1846.3	3	5	737.6
	3	30	0	0	10					14.63	186	712.85	1128.3	1430.8	4	9	7020.5
	4	40	0	0	10					18.00	195	866.03	1001.4	1090.9	6	13	36865.2
	5	50	0	0	10					20.89	601	697.80	752.03	833.01	11	15	131216.4
	6	60	0	0	10					24.04	1338	589.92	705.21	847.79	13	16	407730.8

Table 1 shows the testing results performed by the algorithm for the four nonzero values of the correlation parameter $\varkappa$. Here, for each series of games, the following notation is used: k_0 is the number of games boasting at least one pure strategy solution (such a solution can be detected easily enough even without making use of the tested method); Π^+ denotes the set of games solved under the given tolerance, k^+ is the number of such games, i. e., $k^+ = |\Pi^+|$; Π^- is the set of games on which the algorithm failed having started with all $3n^2$ initial pure strategy pairs, k^- denotes the number of those failures, that is, $k^- = |\Pi^-|$; hence $k_0 + k^+ + k^- = P = 10$.

For the game series with $k^+ > 0$, the following additional notation is used: s_p equals the number of starting strategy pairs, and τ_p is the running time (in seconds) needed to solve game p, with t_{pg} denoting the number of iterations conducted to solve the game with the starting pair $g \in \{1, 2, \ldots, s_p\}$. Therefore, the summary number and the maximal numbers of iterations (linear programming triples $\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$) constitute $t_p = \sum_{g=1}^{s_p} t_{pg}$ and $\bar{t}_p = \max_{1\leqslant g\leqslant s_p} t_{pg}$, respectively. Calculate $S^+ = \sum_{p\in\Pi^+} s_p$ and the following quantities for Table 1:

$$s^+_{\max} = \max_{p\in\Pi^+} s_p, \quad s^+_{\text{av}} = \frac{S^+}{k^+}, \quad t^+_{\max} = \max_{p\in\Pi^+} \bar{t}_p, \quad t^+_{\text{av}} = \frac{1}{S^+} \sum_{p\in\Pi^+} t_p, \quad T^+_{\text{sum}} = \sum_{p\in\Pi^+} \tau_p.$$

For the series with $k^- > 0$, the analogous notation was introduced: here, s_p is the number of starting strategy pairs (equaling $S = 3n^2$) used by the algorithm in order to conclude that game p hasn't been solved; the total number of linear programming triples $\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$ solved equals t_p; the total running time needed for that is τ_p seconds; the record (minimum) Nash function $N(x)$ value found is n_p, and the record value of the sum of the violated mutual complementarity conditions [2] is denoted by Δ. In the same manner, we find $t_p = \sum_{g=1}^{s_p} t_{pg}$ and hence determine $\bar{t}_p = \max_{1\leqslant g\leqslant s_p} t_{pg}$ in order to calculate the following quantities for Table 1:

$$\begin{gathered}
t^-_{\max} = \max_{p\in\Pi^-} \bar{t}_p, \quad t^-_{\text{av}} = \frac{1}{k^- S} \sum_{p\in\Pi^-} t_p, \quad T^-_{\text{sum}} = \sum_{p\in\Pi^-} \tau_p, \\
N^-_{\min} = \mu \min_{p\in\Pi^-} n_p, \quad N^-_{\max} = \mu \max_{p\in\Pi^-} n_p, \quad N^-_{\text{av}} = \frac{\mu}{k^-} \sum_{p\in\Pi^-} n_p, \\
\Delta^-_{\min} = \min_{p\in\Pi^-} \Delta_p, \quad \Delta^-_{\max} = \max_{p\in\Pi^-} \Delta_p, \quad T_{\text{sum}} = T^+_{\text{sum}} + T^-_{\text{sum}}.
\end{gathered}$$

Here, $\mu = 1/\varepsilon = 10^6$ is the norming factor: $\mu\, n_p \leqslant 1$ means that game p has been solved with the given tolerance. Otherwise, the value of $\mu\, n_p$ demonstrates how much the record value N_p of the modified Nash function (MNF) is higher than the desired tolerance value of 10^{-6}.

The main result of our testing procedures is the following conclusion: the proposed algorithm is quite efficient for independent or slightly dependent payoff tables. However, the higher the mutual correlation coefficient value for the tables, the lower the proposed method's reliability.

REFERENCES

1. *Golshtein E. G.* An approximate method for solving finite three-person games. — Ekonomika i Matematicheskie Metody (Economics and Mathematical Methods), 2014, v. 50, № 1, p. 110–116.
2. *Golshtein E. G.*, *Malkov U. H.*, *Sokolov N. A.* Efficiency of an approximate algorithm to solve finite three-person games (a computational experience). — Ekonomika i Matematicheskie Metody (Economics and Mathematical Methods), 2016, v. 52 (in press).