ЭКОНОМИКА И МАТЕМАТИЧЕСКИЕ МЕТОДЫ, 2013, том 49, № 4, с. 94–104

МАТЕМАТИЧЕСКИЙ АНАЛИЗ ЭКОНОМИЧЕСКИХ МОДЕЛЕЙ

ОБ ОДНОМ ЧИСЛЕННОМ МЕТОДЕ РЕШЕНИЯ БИМАТРИЧНЫХ ИГР

© 2013 г. Е.Г. Гольштейн, У.Х. Малков, Н.А. Соколов

(Москва)

Предложен метод решения биматричных игр, основанный на поиске глобального минимума функции Нэша. Осуществляя перебор начальных чистых стратегий, метод отыскивает точное решение игры, если выполнено условие дополнительности, либо приемлемое приближение к множеству точек Нэша. Проведено численное тестирование метода, выявившее его достоинства и недостатки.

Ключевые слова: выпуклая игра, функция Нэша, точка Нэша, биматричная игра, чистая стратегия, смешанная стратегия, дополнительность.
Классификация JEL: C02, C72.

Пусть X_1 и X_2 – подмножества евклидовых пространств E_1 и E_2 соответственно; f_1 и f_2 – функции, определенные конечными значениями на множестве $X = X_1 \times X_2 \subset E = E_1 \times E_2$. Рассмотрим бескоалиционную игру Γ двух лиц, задаваемую множествами стратегий игроков X_1, X_2 и функциями их выигрышей f_1, f_2. Обозначим через X^* множество точек Нэша игры Γ. Предполагая супердифференцируемость функций f_1 и f_2 по $x_1 \in X_1$ и $x_2 \in X_2$ соответственно, свяжем с игрой Γ точечно-множественное отображение T_Γ, определяемое соотношением

$$T_\Gamma(x) = T_\Gamma(x_1, x_2) = (t = (t_1, t_2): -t_i \in \partial_{x_i} f_i(x_1, x_2), i = 1, 2\}, x = (x_1, x_2) \in X. \quad (1)$$

Отображение T_Γ порождает вариационное неравенство

$$t \in T_\Gamma, \ \langle t, x' - x \rangle \geq 0 \quad \forall x' \in X, \quad (2)$$

множество решений которого совпадает с множеством X^* точек Нэша игры Γ. Поэтому любой сходящийся численный метод решения вариационного неравенства (2) позволяет решить игру Γ (найти с любой степенью точности одну из ее точек Нэша).

Игра называется *выпуклой*, если определяющие ее функции f_1, f_2 и множество X удовлетворяют следующим требованиям: X – непустой выпуклый компакт; функция f_1 определена на множестве $\widetilde{X}_1 \times X_2$, вогнута по $x_1 \in \widetilde{X}_1$ при любом фиксированном $x_2 \in X_2$ и непрерывна на $\widetilde{X}_1 \times X_2$, а функция f_2 – на множестве $X_1 \times \widetilde{X}_2$, вогнута по $x_2 \in \widetilde{X}_2$ при любом фиксированном $x_1 \in X_1$ и непрерывна на $X_1 \times \widetilde{X}_2$; $X_i \subset \widetilde{X}_i$, $\widetilde{X}_i$ – открытое выпуклое множество, $i = 1, 2$.

Любая выпуклая игра имеет непустое множество X^* точек Нэша. Однако множество X^* не обязано быть выпуклым (например, оно может состоять из нескольких изолированных точек).

Условимся, что выпуклая игра Γ *имеет выпуклую структуру*, если связанное с ней согласно (1) отображение T_Γ является монотонным. У игры, имеющей выпуклую структуру, множество точек Нэша – непустой выпуклый компакт. В работе (Гольштейн, 2002) описан эффективный численный метод решения вариационных неравенств, порожденных монотонными отображениями. В частности, если игра Γ имеет выпуклую структуру, этот метод, примененный к вариационному неравенству (2), порождает последовательность, сходящуюся к множеству точек Нэша игры Γ.

Далее мы будем иметь дело с *конечными* бескоалиционными играми двух лиц, в которых каждый игрок имеет конечное множество стратегий.

Рассмотрим конечную игру Γ, предполагая, что первый игрок располагает m стратегиями, а второй – n стратегиями. Пусть $a_{ij}^{(r)}$ – выигрыш игрока r, когда игрок 1 использует свою стратегию

i, а игрок 2 – стратегию j, $r = 1, 2..$ Таким образом, конечная игра Γ задается двумя матрицами A_1 и A_2, имеющими m строк и n столбцов. Конечные игры двух игроков в смешанных стратегиях принято называть *биматричными*. Если Γ – биматричная игра, определяемая матрицами A_1 и A_2, она характеризуется множествами стратегий

$$X_1 = \{x_1 = (x_{11}, ..., x_{m1}): \sum_{i=1}^{m} x_{i1} = 1, \quad x_{i1} \geq 0, \quad 1 \leq i \leq m\},$$
$$X_2 = \{x_2 = (x_{12}, ..., x_{n_2}): \sum_{j=1}^{n} x_{j2} = 1, \quad x_{j2} \geq 0, \quad 1 \leq j \leq n\} \tag{3}$$

и функциями выигрышей игроков

$$f_1(x_1, x_2) = x_1 A_1 x_2^{\tau}, \; f_2(x_1, x_2) = x_1 A_2 x_2^{\tau}, \tag{4}$$

где τ обозначает операцию транспонирования.

Каждая биматричная игра является выпуклой игрой. Как установлено в (Гольштейн, 2008), биматричная игра, определяемая матрицами A_1, A_2, имеет выпуклую структуру в том и только в том случае, если

$$A = (a_{ij})_{mn} = A_1 + A_2 = (\alpha_i + \beta_j)_{mn}. \tag{5}$$

Если биматричная игра Γ, определяемая матрицами A_1, A_2, имеет выпуклую структуру, т.е. для матрицы $A = A_1 + A_2$ справедливо представление (5), множество точек Нэша игры Γ совпадает с множеством седловых точек матричной игры Γ' с матрицей $A'_1 = (a_{ij}^{(1)} - \beta_j)_{mn}$. Таким образом, класс биматричных игр с выпуклой структурой фактически совпадает с классом матричных игр. Поскольку решение матричной игры сводится к анализу пары взаимодвойственных задач линейного программирования, класс биматричных игр с выпуклой структурой обладает полиномиальной сложностью. Что касается *всех* биматричных игр, то этот класс задач имеет, по-видимому, сверхполиномиальную сложность. Во всяком случае, к настоящему моменту не существует численного алгоритма, который решает любую биматричную игру с наперед заданной точностью при числе операций, полиномиально зависящем от исходных данных игры.

Пусть Γ – произвольная бескоалиционная игра двух лиц, задаваемая множествами X_1 и X_2 стратегий игроков и функциями f_1 и f_2 их выигрышей, $f = f_1 + f_2$– функция суммарных выигрышей обоих игроков. Введем функцию

$$N(x) = N(x_1, x_2) = \max_{x'_1 \in X_1} f_1(x'_1, x_2) + \max_{x'_2 \in X_2} f_2(x_1, x'_2) - f(x_1, x_2), \tag{6}$$

определенную на множестве $X = X_1 \times X_2$.

Из (6) и определения точки Нэша игры Γ следует, что, во-первых, $N(x) \geq 0$ для всех $x \in X$, а, во-вторых, $N(x) = 0$ в том и только в том случае, если x является точкой Нэша игры Γ. Следовательно, множество точек минимума функции $N(x)$ на X совпадает с множеством X^* точек Нэша игры Γ, причем значение минимума равно нулю. Таким образом, решение игры Γ оказывается эквивалентным поиску точек глобального минимума функции N на множестве X, причем $N(x)$ – это естественная мера отклонения точки $x \in X$ от множества X^*.

В случае биматричных игр вычисление функции $N(x)$ в любой точке $x \in X$ может быть упрощено. Действительно, если f_r, X_r, $r = 1, 2$, определяются соотношениями (3), (4), то

$$\max_{x_1 \in X_1} f_1(x_1, x_2) = \max_{1 \leq i \leq m} \sum_{j=1}^{n} a_{ij}^{(1)} x_{j2}, \quad \max_{x_2 \in X_2} f_2(x_1, x_2) = \max_{1 \leq i \leq n} \sum_{i=1}^{m} a_{ij}^{(2)} x_{i1}.$$

Следовательно,

$$N(x) = \max_{1 \leq i \leq m} \sum_{j=1}^{n} a_{ij}^{(1)} x_{j2} + \max_{1 \leq j \leq n} \sum_{i=1}^{m} a_{ij}^{(2)} x_{i1} - x_1 A x_2^{\tau}, \tag{7}$$

где $A = A_1 + A_2$. Функция $N(x)$ для биматричных игр была введена в работе (Mills, 1960).

Из представления (7) вытекает, что задача минимизации функции $N(x)$ на X в случае биматричных игр эквивалентна задаче минимизации *функций Нэша*

$$\tilde{N}(x_1, x_2, v_1, v_2) = (v_1 - x_1 A_2 x_2^\tau) + (v_2 - x_1 A_1 x_2^\tau), \tag{8}$$

$$\bar{N}(x_1, x_2, v_1, v_2) = \max\{v_1 - x_1 A_2 x_2^\tau, v_2 - x_1 A_1 x_2^\tau\}, \tag{9}$$

где v_1 и v_2 – скалярные переменные, связанные с векторными переменными $x_1 \in X_1$ и $x_2 \in X_2$ условиями

$$\sum_{j=1}^{n} a_{ij}^{(1)} x_{j2} \le v_2, \quad 1 \le i \le m, \quad \sum_{i=1}^{m} a_{ij}^{(2)} x_{i1} \le v_1, \quad 1 \le j \le n.$$

Как и функция $N(x)$, функции Нэша и их различные нормированные варианты служат мерой близости точки $x \in X$ к множеству X^*.

При любом фиксированном $x_2 \in X_2$ введем задачу линейного программирования $P_1(x_2)$ относительно переменных x_2, v_1, определенную соотношениями

$$v_1 - x_1 A x_2^\tau \to \min, \quad \sum_{i=1}^{m} a_{ij}^{(2)} x_{i1} \le v_1, \quad 1 \le j \le n, \quad x_1 = (x_{11}, ..., x_{m1}) \in X_1. \tag{10}$$

Аналогично, при любом $x_1 \in X_1$ введем задачу линейного программирования $P_2(x_1)$ относительно переменных x_2, v_2:

$$v_2 - x_1 A x_2^\tau \to \min, \quad \sum_{j=1}^{n} a_{ij}^{(1)} x_{j2} \le v_2, 1 \le i \le m, x_2 = (x_{12}, ..., x_{n_2}) \in X_2. \tag{11}$$

Определим дополнительные переменные $u_1 = (u_{11},..., u_{n1})$ задачи (10) и $u_2 = (u_{12},..., u_{m2})$ задачи (11) при помощи формул

$$u_{j1} = u_{j1}(x_1, v_1) = v_1 - \sum_{i=1}^{m} a_{ij}^{(2)} x_{i1}, \quad 1 \le j \le n,$$

$$u_{i2} = u_{i2}(x_2, v_2) = v_2 - \sum_{j=1}^{n} a_{ij}^{(1)} x_{i2}, \quad 1 \le i \le m,$$

в этих обозначениях функция Нэша (8) имеет вид

$$\tilde{N} = \sum_{i=1}^{m} u_{i2} x_{i1} + \sum_{j=1}^{n} u_{j1} x_{j2}.$$

Все величины в этой формуле неотрицательные, поэтому, для того чтобы функция $\tilde{N}$ обратилась в нуль, необходимо и достаточно, чтобы в каждой паре хотя бы один сомножитель был равен нулю, т.е. выполнялось *условие дополнительности*. Пусть $I\{A\}$ – индикатор события A, принимающий значение 1, если событие A истинно, и 0, если событие A ложно; функция

$$\Delta = \Delta(x_1, x_2, u_1, u_2) = \sum_{i=1}^{m} I\{x_{i1} > 0, u_{i2} > 0\} + \sum_{j=1}^{n} I\{x_{j2} > 0, u_{j1} > 0\} \tag{12}$$

есть число нарушенных в точке $x \in X$ условий дополнительности. Итак, $x \in X^*$ тогда и только тогда, когда $\Delta(x_1, x_2, u_1, u_2) = 0$.

Введем множества X' точек $x \in X$, являющихся точками локального минимума функции $N(x)$ (см. (6)) на X, и X'', состоящего из точек $x = (x_1, x_2) \in X$ таких, что x_1 – решение задачи $P_1(x_2)$, а x_2 – решение задачи $P_2(x_1)$. Очевидно, $X'' \supset X' \supset X^*$, где X^* – введенное ранее множество точек Нэша игры Γ, совпадающее с множеством точек глобального минимума функции $N(x)$ на X. Нетрудно проверить, что любая точка $x = (x_1, x_2) \in X''$ содержится в X', если хотя бы одна из задач линейного программирования $P_1(x_2)$, $P_2(x_1)$ имеет единственное решение. Поэтому выход точки $x \in X''$ за пределы множества X' скорее исключение, чем правило.

Отправляясь от произвольной точки $x_1 \in X_1$, определим последовательность точек $x^{(k)} = (x_1^{(k)}, x_2^{(k)})$, руководствуясь правилом: $x_1^{(1)} = x_1$, $x_2^{(1)}$ – решение задачи $P_2(x_1)$, $x_1^{(k+1)}$ – решение задачи $P_1(x_2^{(k)})$, $x_2^{(k+1)}$ – решение задачи $P_2(x_1^{(k+1)})$, $k = 1, 2,...$ Поскольку $N(x^{(k+1)}) \le N(x^{(k)})$, $k \ge 1$, существует $\lim\limits_{k\to\infty} N(x^{(k)}) = q_1(x_1)$.

Очевидно, что любая предельная точка последовательности $\{x^{(k)}\}$ содержится в X'', причем если $\bar{x}$ – предельная точка $\{x^{(k)}\}$, то $N(\bar{x}) = q_1(x_1)$.

По аналогичным правилам можно определить последовательность $\{x^{(k)}\}$, отправляясь от произвольной точки $x_2 \in X_2$, и ввести функцию $q_2(x_2)$, $x_2 \in X_2$. В таком случае все предельные точки этой последовательности также содержатся в X'' и значение функции N на каждой из них равно $q_2(x_2)$.

Предложенный в данной работе приближенный метод решения биматричных игр основан на последовательном вычислении значений функций $q_1(x_1)$ и $q_2(x_2)$ при различных стратегиях $x_1 \in X_1$ и $x_2 \in X_2$.

Мы не будем останавливаться на существующих численных методах решения биматричных игр, а отошлем читателя к монографии (Стрекаловский, Орлов, 2007), в которой дан достаточно подробный анализ таких методов, а также детально описан и обоснован оригинальный метод авторов монографии.

Приведем блок-схему предлагаемого метода решения биматричной игры Г.

Шаг 0. Зададим константы: $\bar{S}$ – максимально разрешенное число выборов начальной стратегии; $\bar{K}$ – максимально разрешенное число решаемых пар задач линейного программирования P_1, P_2; $\hat{N}$ – большое положительное число, превышающее величину $N(x)$ для всех $x \in X$; ε_x – небольшое положительное число (компоненты векторов x_1, x_2, u_1, u_2, меньшие этого числа, при вычислении функции $\Delta(x_1, x_2, u_1, u_2)$ в (12) считаются нулями); ε_N – небольшое положительное число (оценка близости значений функции Нэша (7) в соседних точках); положим $s = 0$ (s – счетчик начальных стратегий).

Шаг 1. $s := s + 1$, в качестве очередной начальной точки по некоторым правилам выбираем либо вектор $x_1^{(0)}$, либо вектор $x_2^{(0)}$; полагаем $k = 0$, $N(x^{(k)}) = \hat{N}$ (здесь $2k$ – число обращений к решению задач линейного программирования P_1, P_2).

Шаг 2. $k := k + 1$, отыскиваем новую точку $x^{(k)} = (x_1^{(k)}, x_2^{(k)})$, где в зависимости от первоначального выбора либо $x_2^{(k)} \in \operatorname{Arg\,min} P_2(x_1^{(k-1)})$, $x_1^{(k)} \in \operatorname{Arg\,min} P_1(x_2^{(k)})$, либо $x_1^{(k)} \in \operatorname{Arg\,min} P_1(x_2^{(k-1)})$, $x_2^{(k)} \in \operatorname{Arg\,min} P_2(x_1^{(k)})$, вычисляем $N_k = N(x^{(k)})$, $\bar{N}_k = \bar{N}(x^{(k)}, \bullet)$ и $\Delta_k = \Delta(x^{(k)}, \bullet)$ по формулам (7), (9) и (12).

Шаг 3. Проверяем критерий остановки: если $\Delta_k = 0$, то $x^{(k)}$ – решение биматричной игры, переходим к шагу 6.

Шаг 4. Проверяем убывание функции Нэша: если $N_{k-1} \ge N_k + \varepsilon_N$ и $k < \bar{K}$, то поиск минимума функции Нэша еще не завершен, переходим к шагу 2.

Шаг 5. Поиск минимума функции Нэша закончен, но $\Delta_k > 0$. Если $s < \bar{S}$, то переходим к шагу 1, в противном случае для всех S начальных стратегий точное решение биматричной игры не найдено.

Шаг 6. Стоп.

Тестирование метода осуществлено при помощи программы bimatrix.exe, написанной авторами статьи на языке Фортран. Текст программы оттранслирован Compaq Visual Fortran компилятором. Сама программа bimatrix.exe собрана/линкована в среде MS Visual Studio 6.0. В качестве решателя задач линейного программирования использована процедура linprog (автор У.Х. Малков). Все приводимые ниже результаты для удобства сравнения получены на одном персональном компьютере: процессор Intel(R)Core(TM)i7CPU920/2,67GHz, память 6,00 Гб. Все данные хранились в оперативной памяти. Матрицы A_1 и A_2 генерировались по строкам при помощи нескольких однопараметрических датчиков случайных чисел, равномерно распределенных на отрезке [0, 1] (*rnd*, написан на C++, найден в Интернете; *ur* – старинная простейшая фортранная

функция; выбранные датчики позволяют обеспечить повторяемость процесса поиска решения задачи), все элементы матриц выбирались также из диапазона [0, 1].

Для тестирования использовалось два семейства биматричных задач: семейство малых задач размера $m \times n$, где $m, n \in \{20, 40, 60, 80, 100\}$; семейство больших задач размера $m \times n$, где $m, n \in \{200, 400, 600, 800,1000\}$. Для каждого варианта (из 50 пар сочетаний m, n) генерировалась серия либо из $P = 100$ задач (при $m, n \le 100$), либо из $P = 10$ задач (при $m, n > 100$). Если не оговорено особо, то основные параметры метода таковы: $\bar{S} = m + n$, $\bar{K} = 60$ (это число выбрано достаточно большим и не было достигнуто при тестировании), $\hat{N} = 1{,}23 \times 10^{20}$, $\varepsilon_x = 10^{-8}$, $\varepsilon_N = 10^{-11}$. В качестве начальных стратегий выбирались все чистые стратегии обоих игроков, упорядоченные следующим образом: $x_2^{(1)} = (1, 0, ..., 0)$, $x_2^{(2)} = (0, 1, 0, ..., 0)$, ..., $x_2^{(n)} = (0, ..., 1)$, $x_1^{(n+1)} = (1, 0, ..., 0)$, $x_1^{(n+2)} = (0, 1, 0, ..., 0)$,..., $x_1^{(n+m)} = (0, ..., 1)$. Здесь первые n точек взяты из множества X_2, остальные m точек – из множества X_1 (см. (3)).

Вначале были протестированы задачи со 100%-ной заполненностью матриц A_1 и A_2, результаты этого тестирования приведены в табл. 1–2. Для 2500 малых задач и 250 больших задач метод

Таблица 1. Решение малых задач с заполненностью матриц 100%

Размеры		Начальные точки		*LP*-итерации		Время T_{av}
m	n	S_{av}	$S_{\max}$	K_{av}	$K_{\max}$	
20	20	4,32	21	3,02	9	0,05
40	20	5,56	26	3,24	8	0,07
60	20	5,77	22	3,38	12	0,08
80	20	7,56	32	3,73	10	0,10
100	20	8,14	34	3,63	9	0,12
20	40	4,33	42	2,97	8	0,05
40	40	6,58	24	3,34	10	0,09
60	40	6,15	27	3,48	8	0,10
80	40	8,42	28	3,77	13	0,15
100	40	7,17	23	3,74	9	0,15
20	60	4,37	21	2,83	10	0,06
40	60	6,16	36	3,24	13	0,10
60	60	7,65	38	3,44	14	0,14
80	60	8,06	31	3,63	13	0,17
100	60	7,07	30	3,70	13	0,19
20	80	4,26	15	2,79	9	0,06
40	80	6,23	26	3,30	11	0,11
60	80	6,73	31	3,40	11	0,14
80	80	8,29	36	3,57	10	0,20
100	80	8,16	41	3,66	9	0,24
20	100	5,81	104	3,33	9	0,09
40	100	6,48	26	3,27	10	0,13
60	100	6,20	32	3,29	12	0,16
80	100	8,37	39	3,55	14	0,23
100	100	6,73	31	3,69	9	0,26

Таблица 2. Решение больших задач с заполненностью матриц 100%

Размеры		Начальные точки			LP-итерации			Время
m	n	$S_{\min}$	S_{av}	$S_{\max}$	$K_{\min}$	K_{av}	$K_{\max}$	T_{av}
200	200	2	10,0	27	1	4,01	9	1,65
400	200	1	19,7	51	1	4,24	11	7,58
600	200	3	17,8	36	2	4,39	17	11,34
800	200	2	24,9	49	1	4,73	12	24,24
1000	200	3	20,8	52	1	4,74	11	28,90
200	400	1	8,0	19	1	4,01	8	3,85
400	400	2	16,4	45	1	4,31	12	25,37
600	400	3	15,3	42	1	4,56	9	46,48
800	400	1	9,8	30	1	4,47	11	47,07
1000	400	8	25,9	81	1	4,80	10	155,99
200	600	1	10,5	32	1	4,15	8	7,70
400	600	2	13,1	43	2	4,27	10	40,11
600	600	2	7,5	22	1	4,43	10	53,89
800	600	1	18,2	52	1	4,39	13	220,14
1000	600	2	19,9	69	2	4,63	11	316,15
200	800	1	12,6	26	1	3,87	9	13,75
400	800	2	9,4	17	1	4,12	11	50,68
600	800	2	17,4	75	2	4,53	13	241,76
800	800	1	20,6	43	1	4,47	12	478,92
1000	800	4	18,3	66	1	4,74	14	724,88
200	1000	3	14,3	34	1	3,86	9	24,04
400	1000	1	23,4	64	1	4,47	10	179,25
600	1000	2	9,3	17	1	4,19	12	179,23
800	1000	2	20,7	69	1	4,54	11	833,60
1000	1000	1	21,8	53	1	4,63	11	1308,54

нашел точное решение. Пусть s_p – число начальных точек, понадобившихся методу для получения точного решения биматричной задачи p, причем для этого потребовалось суммарно решить k_p пар линейных задач P_1, P_2, затратив на все t_p секунд, $p = 1, ..., P$. Тогда $S_{\min} = \min\limits_{1 \le p \le P} s_p$, $S_{\max} = \max\limits_{1 \le p \le P} s_p$, $S_{av} = P^{-1} \sum\limits_{p=1}^{P} s_p$, $K_{min} = \min\limits_{1 \le p \le P} (k_p / s_p)$, $K_{max} = \max\limits_{1 \le p \le P} (k_p / s_p)$, $K_{av} = P^{-1} \sum\limits_{p=1}^{P} (k_p / s_p)$, $T_{av} = P^{-1} \sum\limits_{p=1}^{P} t_p$. Как оказалось, для всех малых задач $K_{min} = S_{min} = 1$, поэтому соответствующие им столбцы в табл. 1 отсутствуют.

Анализ табл. 1–2 позволяет сделать следующие выводы.

1. В отличие от S_{max}, принимающего значения из диапазона [15, 104] в табл. 1 и из диапазона [17, 81] в табл. 2, изменение величины S_{av} более гладкое: отношение наибольшего значения

(S_{av} = 8,42) к наименьшему (S_{av} = 4,26) в табл. 1 приблизительно равно 2; это же отношение в табл. 2 равно 25,9/7,5 ≈ 3,45.

2. Величины K_{max} и K_{av} изменяются еще медленнее. Отношение максимального значения K_{av} к его минимальному значению для обеих таблиц оказалось одинаковым (≈1,3). С некоторой натяжкой можно считать K_{av} = const. Диапазоны изменения величины K_{max} ([8, 14] и [8, 17] соответственно) также оказались практически одинаковыми.

3. Величина T_{av} среднего времени решения всех P задач для обеих таблиц монотонно возрастает с ростом m и n.

4. Сравнивая аналогичные столбцы табл. 1 и 2, можно заметить, что с увеличением размеров игр наблюдаемый в табл. 2 рост величин K_{av} незначителен, в то время как рост S_{av} и S_{max} более ощутим.

Итак, судя по результатам, приведенным в табл. 1–2, метод работает стабильно, решив все предложенные биматричные игры с полностью заполненными матрицами A_1 и A_2. Очень хотелось надеяться, что метод будет так же успешен и при решении биматричных игр с разреженными матрицами. Увы! Как оказалось, с уменьшением заполненности матриц повышается риск не получить точное решение игры.

Было проведено тестирование метода для нескольких значений процента к заполненности матриц A_1 и A_2. Генерация производилась по строкам. Для каждой из матриц A_1, A_2 mn раз производится обращение к датчику *ur*, причем если его вычисленное значение меньше к/100, то очередной элемент матрицы определялся посредством обращения к датчику *rnd*, в противном случае элементу матрицы присваивается значение ноль. Для слабо заполненных матриц характерно наличие пустых строк и/или столбцов. Для каждого из них датчик *ur* задает случайный номер одного элемента, а *rnd* – его значение.

Табл. 3–4 иллюстрируют поведение метода при поиске точного решения малых и больших задач соответственно, для к = 5%-ной заполненности матриц. Как видно из таблиц, получение точного решения для всех P игр является скорее исключением, чем правилом (5 случаев из 25 для малых задач и 8 случаев из 25 для больших задач; смысл величин S_{min}, S_{av}, S_{max}, K_{min}, K_{av}, K_{max}, T_{av} для строк, у которых число нерешенных задач K_{no} = 0, такой же, как в табл. 1–2).

Пусть $P^+ = \{p\}$ – множество номеров решенных задач, для каждого $p \in P^+$ известно s_P – число начальных стратегий, понадобившихся методу для нахождения точного решения игры p, при этом решено k_p пар задач P_1, P_2 и затрачено t_p секунд. Для остальных $p \in P^- = \{1,.., P\}\backslash P^+$, $|P^-| = K_{no}$, выбрано $s_p = \bar{S} = n + m$ начальных стратегий, решено k_p пар задач P_1, P_2 за t_p секунд, но точное решение игры p не получено. Тогда при $P^+ \neq \varnothing$ имеем $S_{min} = \min\limits_{p \in P^+} s_p$, $S_{max} = \max\limits_{p \in P^+} s_p$,

$$S_{av} = |P^+|^{-1} \sum_{p \in P^+} s_p; \quad K_{min} = \min_{1 \le p \le P} (k_p / s_p), \quad K_{max} = \max_{1 \le p \le P} (k_p / s_p), \; K_{av} = P^{-1} \sum_{p=1}^{p} (k_p / s_p), \quad T_{av} = P^{-1} \sum_{p=1}^{P} t_p.$$

Если $K_{no} > 0$, то для каждого $p \in P^-$ сначала находятся наилучшее (минимальное) значение функции Нэша $\overline{\overline{N}}_p = \min\limits_{1 \le s \le \bar{s}} \bar{N}_{ps} = \bar{N}_{p\bar{s}}$ и соответствующее $\bar{s}$ значение функции Δ (см. (12)) $\bar{\Delta}_p = \Delta_{p\bar{s}}$, а затем среди отобранных значений выбираются наименьшее, наибольшее и вычисляется усредненное значение: $\bar{N}_{\min} = \min\limits_{p \in P^-} \overline{\overline{N}}_p$, $\bar{N}_{max} = \max\limits_{p \in P^-} \overline{\overline{N}}_p$, $\bar{N}_{av} = K_{no}^{-1} \sum\limits_{p \in P^-} \overline{\overline{N}}_p$, $\Delta_{min} = \min\limits_{p \in P^-} \bar{\Delta}_p$, $\Delta_{\max} = \max\limits_{p \in P^-} \bar{\Delta}_p$,

$\Delta_{av} = K_{no}^{-1} \sum\limits_{p \in P^-} \bar{\Delta}_p$. Для малых задач $S_{min} = K_{min} = \Delta_{min} = 1$, поэтому соответствующие столбцы в табл. 3 отсутствуют. По этой же причине в табл. 4 отсутствует столбец, соответствующий значению $K_{min} = 1$. Кроме того, чтобы не перегружать таблицы, приводится лишь один столбец, содержащий среднее значение функции Нэша $\bar{N}_{av}$.

Проанализировав табл. 3–4, можно сделать следующие выводы.

Таблица 3. Решение малых задач с заполненностью матриц к = 5%

Размеры задач		Не решено K_{no} задач	Начальные точки		*LP*-итерации		Функция Нэша	Дополнительность		Время счета
m	n		S_{av}	S_{max}	K_{av}	K_{max}	$\bar{N}_{av}$	Δ_{av}	Δ_{max}	T_{av}
20	20	3	4,4	20	4,4	11	$5{,}2 \times 10^{-4}$	1,3	2	0,08
40	20	11	10,4	39	5,0	12	$2{,}4 \times 10^{-4}$	1,3	3	0,24
60	20	6	11,7	39	4,7	13	$7{,}9 \times 10^{-4}$	1,7	4	0,23
80	20	7	14,0	40	4,6	13	$1{,}3 \times 10^{-4}$	1,1	2	0,29
100	20	7	15,7	40	4,6	12	$2{,}1 \times 10^{-4}$	1,3	3	0,34
20	40	9	7,3	59	4,9	13	$5{,}4 \times 10^{-4}$	2,1	8	0,19
40	40	0	10,5	50	5,8	14				0,20
60	40	3	21,3	78	6,2	18	$8{,}1 \times 10^{-5}$	1,0	1	0,48
80	40	8	20,2	78	6,0	14	$1{,}1 \times 10^{-4}$	2,2	8	0,57
100	40	8	26,0	75	6,3	16	$3{,}9 \times 10^{-5}$	1,6	3	0,73
20	60	5	8,6	77	4,4	11	$1{,}2 \times 10^{-4}$	1,4	2	0,17
40	60	10	16,2	96	6,0	16	$4{,}0 \times 10^{-5}$	1,4	3	0,48
60	60	14	26,4	108	6,5	16	$3{,}9 \times 10^{-5}$	2,9	12	0,85
80	60	0	12,3	79	6,0	14				0,26
100	60	0	9,0	51	4,8	11				0,17
20	80	3	9,7	100	4,8	12	$2{,}2 \times 10^{-4}$	1,7	3	0,20
40	80	11	15,6	108	5,9	15	$5{,}0 \times 10^{-5}$	1,6	3	0,54
60	80	0	13,8	59	6,2	18				0,30
80	80	9	35,4	157	7,0	19	$2{,}3 \times 10^{-5}$	1,6	3	1,12
100	80	11	35,8	144	6,9	18	$1{,}3 \times 10^{-5}$	1,6	3	1,28
20	100	9	11,0	120	4,6	12	$1{,}2 \times 10^{-4}$	1,5	4	0,31
40	100	5	18,2	131	5,9	15	$6{,}0 \times 10^{-5}$	2,2	3	0,48
60	100	0	7,3	35	4,6	13				0,13
80	100	13	31,4	157	6,7	17	$3{,}1 \times 10^{-5}$	1,9	3	1,20
100	100	11	47,6	199	7,0	20	$1{,}8 \times 10^{-5}$	1,6	4	1,68

1. Не наблюдается зависимости числа нерешенных задач от размеров задачи. Максимальная доля нерешенных задач K_{no}/P составляет для малых задач 14%, для больших задач она доходит до 90%.

2. Наиболее устойчивыми оказались величины K_{av} и K_{max}. Диапазон изменения K_{av} для серий, в которых имелись нерешенные игры, составил для малых задач [4,4; 7,0], для больших задач – [7,8; 12,2] (для серий, в которых все P задач решены, диапазон изменения еще уже); соответственно, величина K_{max} изменяется в интервале [11; 20] для серий малых задач, в которых имеются нерешенные игры, и в интервале [11; 14] для серий полностью решенных малых задач, для больших задач интервалы другие: [23; 43] против [10; 33]. Полученные результаты подтверждают тот факт, что количество итераций, требующихся для решения задач линейного программирования, в значительной мере зависит от размеров решаемых задач.

Таблица 4. Решение больших задач с заполненностью матриц к =5%

Размеры задач		Не решено K_{no} задач	Начальные точки			*LP*-итерации		Функция Нэша	Дополнительность			Время счета
m	n		S_{min}	S_{av}	S_{max}	K_{av}	K_{max}	$\bar{N}_{av}$	Δ_{min}	Δ_{av}	Δ_{max}	T_{av}
200	200	3	19	88	151	7,8	23	$8{,}6\times10^{-6}$	1	2,0	3	7,15
400	200	1	37	159	365	8,6	25	$8{,}9\times10^{-7}$	1	1,0	1	10,31
600	200	0	15	167	394	7,8	25					12,41
800	200	1	96	219	368	9,4	25	$3{,}1\times10^{-6}$	1	1,0	1	81,71
1000	200	1	46	129	326	11,4	25	$4{,}9\times10^{-6}$	3	3,0	3	120,34
200	400	0	1	85	419	10,5	21					14,08
400	400	0	11	93	275	7,9	21					13,33
600	400	2	35	282	721	8,5	31	$4{,}1\times10^{-7}$	1	1,0	1	220,50
800	400	1	17	344	770	9,5	29	$1{,}3\times10^{-5}$	6	6,0	6	300,40
1000	400	0	25	182	347	8,2	28					69,66
200	600	1	9	33	115	10,7	23	$1{,}0\times10^{-5}$	3	3,0	3	29,12
400	600	3	50	264	854	8,8	38	$1{,}4\times10^{-5}$	1	1,7	2	225,44
600	600	3	144	276	469	7,8	49	$1{,}6\times10^{-5}$	2	2,7	4	168,70
800	600	0	1	331	1010	7,5	33					124,22
1000	600	0	2	243	110	4,6	15					9,32
200	800	1	24	93	200	7,7	23	$1{,}4\times10^{-6}$	2	2,0	2	79,75
400	800	1	7	185	835	11,0	31	$4{,}3\times10^{-6}$	2	2,0	2	723,70
600	800	0	8	123	328	7,2	24					56,32
800	800	9	99	99	99	9,3	38	$1{,}0\times10^{-5}$	1	4,4	9	1357,76
1000	800	3	137	424	1122	9,1	43	$9{,}5\times10^{-6}$	1	2,0	3	2510,30
200	1000	2	11	69	141	12,2	23	$2{,}1\times10^{-6}$	1	1,0	1	277,38
400	1000	1	13	129	242	10,9	24	$3{,}8\times10^{-6}$	1	1,0	1	854,49
600	1000	0	1	16	52	3,8	10					24,01
800	1000	1	34	320	731	8,5	32	$1{,}1\times10^{-6}$	3	3,0	3	1985,97
1000	1000	3	43	447	1800	7,5	35	$3{,}8\times10^{-6}$	1	2,0	4	834,33

3. Как показывают полученные результаты, нельзя предсказать, какое количество начальных стратегий (от 1 до $n + m$) потребуется для решения конкретной игры (см. столбцы S_{min} в табл. 4 и S_{av}, S_{max} в табл. 3–4). Такая же неопределенность обнаруживается при анализе столбцов, связанных с нарушением условий дополнительности для нерешенных игр. Учитывая также, что число нерешенных задач в серии также может значительно меняться, мы получаем объяснение тому факту, почему среднее время T_{av} решения одной задачи столь существенно варьируется.

4. Для каждой нерешенной задачи в серии из P задач определяется наилучшее значение функции Нэша (см. (9)), в табл. 3–4 приведено усредненное значение $\bar{N}_{av}$ (диапазон изменения для малых задач – $[1{,}3 \times 10^{-5}; 7{,}9 \times 10^{-4}]$, для больших задач он оказался на порядок лучше: [4,1 × 10–7; $1{,}6 \times 10^{-5}$)] и наихудшие значения функции Нэша (($3{,}13 \times 10^{-3}$ среди всех серий малых задач, $2{,}80 \times 10^{-5}$ среди всех серий больших задач). Эти числа позволяют оценить, насколько можно использовать предлагаемый метод для получения *приближенного* решения игры.

Таблица 5. Зависимость количества (доли) нерешенных задач от заполненности матриц

Задачи	Общее число	Заполненность матриц (к)				
		100%	20%	10%	5%	1%
Малые	2500	0	0,0036	0,0564	0,0652	–
Большие	250	0	0	0,04	0,148	0,496

5. Наконец, нет прямой зависимости между величинами $\bar{N}_{av}$ и максимальным, минимальным и усредненным значениями функции Δ, рассчитанным по ее наименьшим среди нерешенных задач в серии значениям. Так, по данным табл. 3, минимальное усредненное значение функции Нэша, равное $1{,}3 \times 10^{-5}$, получено в серии с $m = 100$, $n = 80$ и максимальное значение $7{,}9 \times 10^{-4}$ – в серии с $m = 60$, $n = 20$. В то же время, указанные в табл. 3 минимальные значения Δ_{av} и Δ_{max}, оба равные 1, обнаруживаются в серии задач с $m = 60$, $n = 40$, их максимальные значения, соответственно, равные 2,9 и 12, – в серии с $m = n = 60$. Функция Нэша и функция Δ оценивают близость к множеству точек Нэша, но делают это по-разному.

Была исследована зависимость числа нерешенных задач от заполненности матриц A_1 и A_1 (табл. 5). Для обоих семейств решаемых задач четко прослеживается рост отношения K_{no}/P при уменьшении процента к заполненности матриц.

Итак, проведенные численные эксперименты показали, что предлагаемый метод отыскивает точное решение многих билинейных игр, но не всех. Поэтому предпринимались попытки, не найдя точного решения задачи при помощи $n + m$ чистых начальных стратегий, продолжить поиск решения, проделав еще один либо несколько циклов $n + m$, с той лишь разницей, что вместо чистых начальных стратегий решено было использовать случайные смешанные начальные стратегии, генерируемые при помощи еще одного датчика случайных чисел *ur*. А именно, генерировалось либо n, либо m неотрицательных чисел, затем производилось их нормирование, с тем чтобы построенная таким образом смешанная начальная стратегия принадлежала соответствующему множеству X_2 либо X_1.

Опишем результаты численного тестирования с использованием случайных начальных стратегий для семейства малых задач при заполненности матриц к = 20%. Из 2500 задач при помощи чистых стратегий не было решено $K_{no} = 21$ задач, находящихся в 11 сериях из 25. Временны́е затраты на 1 решенную задачу составили $156{,}3/1079 \approx 0{,}146$ секунд, на одну нерешенную задачу – $26{,}0/21 \approx 1{,}238$ секунд. Один дополнительный цикл случайных смешанных начальных стратегий позволил найти решение 9 задач. Затем каждый дополнительный цикл трижды давал по одной решенной задаче. Время, затраченное на решение этих 12 задач, составило 41 секунду. Зато 9 последних задач не были решены даже при 15 ($n + m$) выборах случайных начальных стратегий! И время, затраченное на попытку их точного решения (239 секунд), оказалось примерно в 1,2 раза больше, чем было потрачено на решение остальных 1091 задач (197 секунд).

Поэтому, хотя в приведенном примере применение смешанных начальных стратегий позволило решить больше половины задач, не решенных при помощи чистых начальных стратегий, мы не можем рекомендовать применение случайного выбора начальных точек в качестве эффективного приема нахождения точного решения задачи. Нужно либо довольствоваться полученным приближенным решением игры, либо искать другие подходы к его улучшению.

СПИСОК ЛИТЕРАТУРЫ

Гольштейн Е.Г. (2002). Метод решения вариационных неравенств, определяемых монотонными отображениями // *Журнал вычислительной математики и математической физики*. Т. 42. № 7.

Гольштейн Е.Г. (2008). О монотонности отображения, связанного с неантагонистической игрой двух лиц // *Экономика и мат. методы*. Т. 44. Вып. 4.

Стрекаловский А.С., Орлов А.В. (2007). Биматричные игры и биматричное программирование. М.: Физматлит.

Mills H. (1960). Equilibrium Points in Finite Games // *J. of the Society for Industrial and Applied Mathematics*. Vol. 8. No. 2.

Поступила в редакцию
29.05.2013 г.

A Numerical Method for Solving Bimatrix Games

E.G. Golshtein, U.Kh. Malkov, N.A. Sokolov

We propose a method for solving bimatrix games based on a search of a global minimum of the Nash function. Choosing one by one some initial pure strategies, the method finds an exact solution of the game, if the complementarity condition holds, or it gives an acceptable approaching to the set of Nash points. The numerical tests of the method identified its advantages and disadvantages.

Keywords: convex game, Nash point, Nash function, bimatrix game, pure strategy, mixed strategy, complementarity
JEL Classification: C02, C72.