How fast is fast fictitious play ?

Bernd Kummer [1]

10 September 2006

Dedicated to Professor Naum Shor

Abstract.
We present a modification of the fictitious play algorithm for matrix games by aggregating trivial original steps. The resulting new algorithm consists of comparable simple steps but has a much better rate of convergence. Modifications for solving large scale LP-programs, numerical results and some conjuncture concerning the total effort of elementary steps for ensuring an error $< \varepsilon$ will be discussed.

Key words.
matrix games, linear programming, fictitious play, aggregation of steps, numerical test.

1 Introduction

In 1951, Julia Robinson [6] proved that matrix games can be solved by the well-known fictitious playing algorithm proposed in [1]. This method, here denoted by R-method, requires few and simple operations (basically n additions) in each step and keeps the matrix fixed (in contrast to all standard methods for solving LP). Parallelization is very simple, too. Due to the close connection to linear programs, so already several authors used it for solving large-scale linear programs [3] and even for proposing a heuristic for certain discrete optimization problems [7].

Nevertheless, the convergence of the R-method is slow and (usually) the error does not monotonically decrease. In addition, there is almost no chance to take advantage of having already some good approximate solution.

The present paper shows how the R-method can be improved such that the new method preserves all advantages of the first one. The basic idea is very simple: we show that (many) uniform steps of the R-method can be aggregated in a cheep manner. This leads us to our modification modR1 which satisfies (with each rational matrix, cf. Theorem 2) an error estimate of the type

$$err_{modR1}(s)/err_{Rob}(s) \to 0 \quad \text{if } \ s \to \infty \text{ (the index of steps)}$$

and (with each integer matrix) for increasing σ and certain $s = s(\sigma) \geq \sigma$,

$$err_{modR1}(\sigma) = err_{Rob}(s) \quad \text{where } \ \sigma/s \ \leq \ err_{Rob}(s) \to 0.$$

Hence, at least for rational matrices, our order of convergence (measured by the needed steps) is better from the qualitative point of view, not only by some constant factor. This makes the new algorithm more attractive for solving large scale problems (games, linear programs or linear inequality systems).

[1]Institut für Mathematik, Humboldt–Universität zu Berlin, D-10099 Berlin. E–Mail: kummer@mathematik.hu-berlin.de

The paper is organized as follows. In order to simplify the considerations we formulate all statements for skew-symmetric matrices only. Thus, as a justification, we first mention that this class is sufficient general for solving arbitrary matrix games and linear programs. This will be done in section 2.

In section 3, the R-method, our algorithms modR1 and modR2 as well as their basic properties are presented.

In section 4, we present estimates for convergence and motivate the conjunction

$$s \le n\,\frac{a}{\delta} \qquad \text{where } a = \max_{i,j} a_{i,j} \tag{1.1}$$

concerning the number of steps for ensuring an error $\le \delta$ by modR1 and modR2, if A is a skew-symmetric (n,n) matrix. This formula was true for all tested matrices (more than 10^7) and implies that one needs no more than

$$s_E = K\ n^2\ \frac{a}{\delta}$$

elementary operations (with less than n^2 divisions and $K \approx 5$) for getting the related error. Notice that already the "exact" solution of $Ax = b$ requires $O(n^3)$ elementary operations. The estimates for solving the problem via transformation into a linear program are not better, cf. [5]. So, if the required relative error $\frac{\delta}{a}$ is not too small, in particular if $\frac{\delta}{a} > \frac{1}{n}$, the algorithms are very appropriate to solve (approximately) large scale problems.

Notations: Throughout, we denote the largest component of a vector $z \in \mathbb{R}^n$ by $\max z$ and the maximal entry of A by a. Further, $A_{.i}$ and $A_{i.}$ denote the ith column and row of A, respectively, and $S_n = \{x \in \mathbb{R}^n \mid \sum_i x_i = 1,\ x_i \ge 0\ \forall i\}$ denotes the unit-simplex in $\mathbb{R}^n$. For an (m,n) matrix A, a pair $(\hat{x},\hat{y}) \in S_m \times S_n$ is a (Nash-) equilibrium, if

$$x^T A\hat{y} \le \hat{x}^T A\hat{y} \le \hat{x}^T Ay \quad \forall\ (x,y) \in S_m \times S_n.$$

Solving the *matrix game* A means to find some equilibrium. If $m = n$ and $A^T = -A$ then A is called skew-symmetric.

2 Arbitrary matrix games and LP

If A is skew-symmetric, one easily sees due to $x^TAy = -y^TAx$ and $x^TAx = 0\ \forall x, y$ that the equilibria are just the pairs $(\hat{x},\hat{y}) \in S_n \times S_n$ such that $\hat{x}$ and $\hat{y}$ solve

$$Ax \le 0, \quad x \in S_n\ . \tag{2.1}$$

Solvability of (2.1) is ensured by the minimax Theorem (and by J. Robinsons Theorem). Considering skew-symmetric matrices only is sufficient for solving arbitrary matrix games and linear programming.

Linear programs.
The subsequent facts follow directly from duality of linear programming and have been observed (e.g.) also in [3] and [8]. Given a pair of dual problems

$$\begin{array}{lll} (P) & \max\ c^Tx & \text{s.t. } A\,x \le b,\ x \ge 0, \quad (A \text{ is a } (m,n)\text{-matrix }) \\ (D) & \min\ b^Ty & \text{s.t. } A^T\,y \ge c,\ y \ge 0 \end{array}$$

consider a solution $\theta = (\xi, \eta, \tau) \in S_{n+m+1}$ of the skew-symmetric matrix game

$$G = G(A, b, c) := \begin{pmatrix} 0 & -A^T & c \\ A & 0 & -b \\ -c^T & b^T & 0 \end{pmatrix}. \tag{2.2}$$

If $\tau > 0$, then

$$x = \xi/\tau \text{ and } y = \eta/\tau \tag{2.3}$$

is a pair of primal-dual solutions for (P, D) (since x and y are feasible and $b^T y \leq c^T x$). If (x, y) is a pair of primal-dual solutions to (P, D), then $\theta = (\xi, \eta, \tau)$ with components

$$\tau = (1 + \sum x_i + \sum y_k)^{-1} > 0, \quad \xi = \tau x \text{ and } \eta = \tau y \tag{2.4}$$

solves the game G (since $c^T x = b^T y$ holds for the solutions). Thus G has no solution with $\tau > 0$ iff (P) is unsolvable. The latter is true (by the Farkas Lemma) iff there is a solution $\theta = (\xi, \eta, 0)$ of G such that $-c^T \xi + b^T \eta < 0$.

The relation between (P, D) and G preserves particular structures of A and becomes more complicated only if G has solutions with $\tau = 0$ and $\tau > 0$. This happens iff (P) or its dual has an unbounded solution set. Then, after solving G, one may obtain a solution with $\tau = 0$ and $-c^T \xi + b^T \eta = 0$. Having such a solution, we only know that the direction (ξ, η) belongs to the recession cone of the primal-dual solution set, provided that (P) is solvable.

So, in accordance to (2.3) and (2.4), the minimal component τ^* over all solutions (ξ, η, τ) to G plays the role of a condition number to (P, D) where $\tau^* = 0$ stands for degeneration (unbounded or empty solution sets), and large τ^* ensure that the primal-dual solutions have small norms. For the subsequent algorithms, it makes no difficulty to start in such a way that $\tau > 0$ holds in the beginning (choose $\zeta = 0$ and select $i(0) = n + m + 1$).

Arbitrary matrix games.
Let M be any (m, n) matrix. The game M can be solved by a well-known transformation: First add a sufficiently large constant $c > 0$ to all entries of M in order to obtain a matrix A with positive entries $a_{i,j} > 0\ \forall i, j$ (this keeps the equilibria fixed). Next consider the above problems (P, D) for

$$c = 1_n = (1, ..., 1)^T \in \mathbb{R}^n, \quad \text{and} \quad b = 1_m = (1, ..., 1)^T \in \mathbb{R}^m, \quad \text{i.e.,}$$

$$(P)\ \max\ 1_n^T\, x \ \text{ s.t. }\ A\, x \leq 1_m,\ x \geq 0 \quad \text{and} \quad (D)\ \min\ 1_m^T\, y\ \text{ s.t. }\ A^T\, y \geq 1_n,\ y \geq 0.$$

Since (P) has a nonempty and bounded feasible region, solutions $\hat{x}, \hat{y}$ exist and satisfy $0 < 1_n^T \hat{x} = 1_m^T \hat{y}$. With $v = 1_n^T \hat{x},\ \hat{\xi} = x/v,\ \hat{\eta} = y/v$, this yields $\hat{\xi} \in S_n$ and $\hat{\eta} \in S_m$. Furthermore, the constraints ensure

$$\eta^T A\, \hat{x} \leq 1 \leq \hat{y}^T A\, \xi \quad \forall\, \xi \in S_n,\ \eta \in S_m.$$

Therefore, division by v tells us that $(\hat{\eta}, \hat{\xi})$ is an equilibrium pair for A and M. It can be determined via the skew-symmetric matrix G (2.2) for $b = 1_m,\ c = 1_n$ by using the given transformations.

3 The basic algorithms

In what follows we will throughout assume that

(A1) A is a (n, n) matrix, $A^T = -A$ and each column of A has a positive entry.

If $a_{k,i} \le 0 \; \forall k$ then the unit vector e^i satisfies (2.1). Clearly, (A1) implies $n \ge 3$. Let

$$\alpha(x) = \max_i A_{i.}\, x \tag{3.1}$$

be the error of $x \in S_n$. Solutions x^* minimize the non-negative, piecewise linear maximum function $\alpha(.)$ on S_n. Hence they can be determined by various methods, in particular by all algorithms of linear programming. Here, we are interested in fast and cheep methods for computing approximate solutions.

3.1 Julia Robinson's method

Since A is skew-symmetric the method can be described (see also [3]) as follows.

R-method.
Fix some $Z^0 = \zeta \in \mathbb{R}^n$ *such that* $\max \zeta = 0$, *and put* $s = 0$ *and* $Y^0 = 0 \in \mathbb{R}^n$.

Determine, in step $s \ge 0$, *some maximal component of* Z^s *(say its index is* $i = i(s)$*) and use the ith unit vector and the ith column of A, respectively, in order to put*

$$Y^{s+1} = Y^s + e^i, \quad Z^{s+1} = Z^s + A_{.i}, \qquad s := s + 1, \quad \textit{repeat} \qquad \diamond. \tag{3.2}$$

Comments:
The so-called *active* index $i = i(s)$ is not unique, in general. For the numerical tests described below, we then selected the smallest one.
To $s > 0$, elements $x^s := \frac{Y^s}{s} \in S_n$ are assigned. They obviously satisfy

$$Ax^s = \frac{AY^s}{s} = \frac{Z^s - Z^0}{s} \quad \text{and} \quad x^{s+1} = \frac{1}{s+1}\,[sx^s + e^i]. \tag{3.3}$$

Hence the error (3.1) satisfies

$$\alpha(x^s) = \frac{1}{s} \max Z^s + \beta_s \quad \text{where } \beta_s \in [0,\ \max \frac{-\zeta}{s}] \tag{3.4}$$

and can be controlled via $\max Z^s$ without computing Ax^s.
In the game theory, x is called a mixed strategy and the iterations improve x by "fictitious play". In accordance with J. Robinson [6], it holds

Theorem 1 *For all* $\varepsilon > 0$, *there exists some* $s(A, \varepsilon)$ *such that*

$$\frac{\max Z^s}{s} < \varepsilon \quad \textit{for all } s > s(A, \varepsilon) \textit{ and all feasible } \zeta. \qquad \diamond \tag{3.5}$$

The proof even shows that $s(A, \varepsilon)$ depends on ε and a only. In particular, it follows $\alpha(x^s) \to 0$ as $s \to \infty$. The convergence of the error is slow and $\alpha(x^{s+1}) > \alpha(x^s)$ may happen if x^s is close to a solution and s is small (compared with the distance).
The requirement $\max \zeta = 0$ is a device for proving convergence in [6]. In applications, the setting $\zeta = 0$ is appropriate. Then $i(s)$ denotes a maximal component of Ax^s.

3.2 Aggregation of R-steps and modR1

Let $i = i(s)$ be *active* at step s of the R-method (3.2). We ask for the maximal number m of steps $s+1, ..., s+m$ which do not change this situation. To find m, we suppose that the R-method uses the active index i as long as possible (if the index is not unique) and obtain the simple condition

$$Z_i^{s+p} \geq Z_k^{s+p} \quad \forall k, \ \forall p = 1, 2, ..., m$$

which means just

$$Z_i^s + p\ a_{ii} \geq Z_k^s + p\ a_{k,i} \quad \forall k$$

or equivalently (since $a_{i,i} = 0$),

$$p \leq \frac{Z_i^s - Z_k^s}{a_{k,i}} \qquad \forall k \ \text{ satisfying } a_{k,i} > 0.$$

Thus it holds $m = [q]$ where $[q]$ denotes the integer part of

$$q = \min_k \{\frac{Z_i^s - Z_k^s}{a_{k,i}} \mid 1 \leq k \leq n, \ a_{k,i} > 0\}. \tag{3.6}$$

The minimum exists due to (A1). Knowing m the next $m+1$ steps can be made at once by setting

$$s' = s + m + 1, \quad Z^{s'} = Z^s + (m+1)A_{.i} \ \text{ and } Y^{s'} = Y^s + (m+1)e^i. \tag{3.7}$$

At iteration s' of the R-method, some new i' is active, and we may continue to calculate (the next) m. This leads us to an algorithm where μ entails the role of $m+1$.

modR1.
Fix some $z(0) = \zeta \in \mathbb{R}^n$ *such that* $\max \zeta = 0$, *and put* $\sigma = 0$ *and* $y(0) = 0 \in \mathbb{R}^n$.
Step σ*: Choose* $i = i(\sigma)$ *such that* $\max z(\sigma) = z(\sigma)_i$, *determine*

$$q(\sigma) = \min_k \{ \ \frac{z(\sigma)_i - z(\sigma)_k}{a_{k,i}} \ \mid \ 1 \leq k \leq n, \ a_{k,i} > 0\}, \quad \mu = [q(\sigma)] + 1 \tag{3.8}$$

and put

$$y(\sigma+1) = y(\sigma) + \mu\ e^i, \ z(\sigma+1) = z(\sigma) + \mu\ A_{.i}, \qquad \sigma := \sigma + 1, \text{ repeat. } \ \diamond \tag{3.9}$$

Obviously, with identical initial points, the steps σ of modR1 and the steps $s = s(\sigma)$ which change the active index (for the σth time) in the R-method, correspond to each other. We compute $y(\sigma) = Y^{s(\sigma)}$ and $z(\sigma) = Z^{s(\sigma)}$ only. Furthermore, it holds

$$s(\sigma) = \|y(\sigma)\|_1 \text{ (sum-norm) } \ \text{and} \ \ s(\sigma+1) = s(\sigma) + \mu. \tag{3.10}$$

To all $\sigma > 0$, now the elements $x(\sigma) = \frac{y(\sigma}{s(\sigma)} \in S_n$ are assigned. They satisfy

$$Ax(\sigma) = \frac{z(\sigma) - \zeta}{s(\sigma)} \ , \quad x(\sigma+1) = \frac{1}{s(\sigma) + \mu}\ [s(\sigma)\ x(\sigma) + \mu\ e^{i(\sigma)}] \tag{3.11}$$

and (3.4) attains the form

$$\alpha(x(\sigma)) = \ \frac{1}{s(\sigma)} \max z(\sigma) + \beta_\sigma \quad \text{ where } \beta_\sigma \in [0, \ \max \frac{-\zeta}{s(\sigma)}] \ . \tag{3.12}$$

It is important (and the key of the modification) that (3.8) remains a cheep operation which depends again on Z^s (now $z(\sigma)$) and column $A_{.\ i}$ only.

If we delete (3.8) and simply put $\mu = 1$ instead, then modR1 decribes again the R-method. This allows a direct and simple comparison of both algorithms during the running time. We recommend the interested reader to do it while looking at the error, the active index and the quotient $s(\sigma)/\sigma$ as well.

Clearly, if $\mu = 1$ *follows at each step of* modR1, we aggregate nothing. However, this cannot happen for rational A though $\mu = 1$ *at certain steps* is not excluded, cf. Theorem 2.

Monotonicity and other modifications. Let $\zeta = 0$ and $\sigma > 0$. The quotient $q(\sigma)$ in (3.8) has been determined in such a way that, for $\lambda \geq 0$, the maximal component of $z(\sigma) + \lambda A_{.i(\sigma)}$ changes exactly at $\lambda = q$ from $i = i(\sigma)$ to another index, namely a minimizer k^* in (3.8). Due to $\zeta = 0$, the same is true for the maximal components of the function

$$f(\lambda) := A\ \xi(\lambda) \quad \text{where } \xi(\lambda) = \frac{1}{s(\sigma) + \lambda}\ [s(\sigma)\ x(\sigma) + \lambda\ e^{i(\sigma)}].$$

Furthermore, it holds

$$f_i(q) = f_{k^*}(q) = \alpha(\ \xi(q)\ ) \leq\ \alpha(\ x(\sigma)\ ) \ \text{ where } < \text{ holds for } q > 0.$$

If already $\alpha(x(\sigma)) \leq a_{k^*,i}$ (> 0), now the error $\alpha(\xi(\lambda))$ increases for $\lambda > q$ since

$$\alpha(\ \xi(\lambda)\ ) \geq A_{k^*.}\ \xi(\lambda) \text{ and } \frac{d}{d\lambda} A_{k^*.}\xi(\lambda) = \frac{s(\sigma)}{(s(\sigma) + \lambda)^2}\ [A_{k^*.}\ e^{i(\sigma)} - A_{k^*.}\ x(\sigma)] > 0.$$

The algorithm modR1 uses $\lambda = \mu = [q] + 1$ for defining $x(\sigma + 1) = \xi(\lambda)$, hence $\alpha(x(\sigma+1)) > \alpha(x(\sigma))$ is possible if q is small in comparison with $\lambda - q$. To reduce this effect one may define $x(\sigma + 1) = \xi(\lambda)$ with smaller $\lambda = \lambda(\sigma) \in (q, \mu]$ which means to replace μ by λ in formula (3.8) only. A similar replacement will define the subsequent algorithm modR2.

Notice, however, that the straightforward setting $\lambda = q$ is wrong since $q = 0$ cannot be excluded (and may really happen) in (3.8) and that, for $\lambda \neq \mu$, the resulting algorithm is no longer directly comparable with the R-method.

3.3 The modified algorithm modR2

Our iterations are now denoted by s instead of σ. We generate n-vectors $z(s)$, $y(s)$ by the following procedure.

modR2
Fix some $z(0) = \zeta \in \mathbb{R}^n$ such that $\max\zeta = 0$, and put $s = 0$ and $y(0) = 0 \in \mathbb{R}^n$.
Step s: Determine some $i = i(s)$ such that $\max z(s) = z(s)_i$ and put

$$h(s) = \min_k \{\ \frac{1 + z(s)_i - z(s)_k}{a_{k,i}}\ \mid\ 1 \leq k \leq n,\ a_{k,i} > 0\ \} \tag{3.13}$$

as well as

$$y(s+1) = y(s) + h(s)e^i, \quad z(s+1) = z(s) + h(s)A_{.i}, \quad s := s+1, \text{ repeat. } \diamond \tag{3.14}$$

If a maximizing index k of (3.13) coincides with a maximizing index k^* of (3.8) for $\sigma = s$, then we simply exchange μ of modR1 by $\lambda = h > q$ (as just discussed above). The next Lemma summarizes simple properties of the generated sequences.

Lemma 1 *It holds*

$$z(s) = \zeta + Ay(s), \tag{3.15}$$

and each index k^*, related to the minimum $h(s)$ in (3.13), fulfills with $i = i(s)$,

$$z(s+1)_{k^*} \;=\; \max z(s+1) \;=\; 1 + \max z(s) \;=\; 1 + s. \qquad \diamond \tag{3.16}$$

Proof. Equation (3.15) is obvious. By (3.13) and (3.14) we have $k^* \neq i$ and

$$z(s+1)_k = z(s)_k + h(s)\ a_{k,i} = z(s)_k + \frac{1 + z(s)_i - z(s)_{k^*}}{a_{k^*,i}}\ a_{k,i}.$$

Hence, for $k = k^*$, it follows $\frac{a_{k,i}}{a_{k^*,i}} = 1$ and $z(s+1)_{k^*} = 1 + z(s)_i$. Considering the cases of $a_{k,i} \leq 0$ and $a_{k,i} > 0$ separately and recalling $a_{i,i} = 0$, one also easily obtains $z(s+1)_k \leq z(s+1)_{k^*}\ \forall k$. $\square$

Therefore, the value $\max z(s)$, important for the error-estimate, now coincides with s, one may put $i(s+1) = k^*$ and replace $z_i(s) = s$ in (3.13). Since $h \geq \frac{1}{a}$, the sequence

$$t(s) = \sum_{s' \,<\, s} h(s') = \|y(s)\|_1$$

diverges and the assigned sequence $x(s) = \frac{y(s)}{t(s)} \in S_n$ fulfills, as under (3.11),

$$Ax(s) = \frac{z(s) - \zeta}{t(s)}\ , \qquad x(s+1) = \frac{1}{t(s) + h(s)}\ [t(s)x(s) + h(s)e^i] \tag{3.17}$$

and

$$\alpha(x(s)) = \frac{1}{t(s)} \max z(s) + \beta_s \quad \text{ where } \beta_s \in [0,\ \max \frac{-\zeta}{t(s)}]\ . \tag{3.18}$$

after $s > 0$ steps. This induces the key question: How fastly will $t(s)$ increase ? Theorem 3 ensures $\frac{t(s)}{s} \to \infty$, so $\alpha(x(s)) \to 0$ holds again.

The algorithm modR2 can be compared with the R-method and modR1, too: As in modR1, we aggregate the R-steps $s, s+1, ..., s+[q]$ with fixed active index i. After iteration $s + [q]$, the next R-step necessarily changes the active index. Only in this situation, modR2 requires to do something else by replacing now $m + 1$ in (3.7) with $h(s)$ from (3.13) in order to obtain $\max z(s) = s$.

4 Estimates of convergence

4.1 Theoretical comparison of the R-method and modR1

Theorem 2 *Let all $a_{i,j}$ be rational with common denominator Q, let $Z^0 = z(0) = 0$ and $s = s(\sigma)$ be the number of R-steps which corresponds to $\sigma > 0$ steps of modR1. Moreover, let $\delta(s) = \frac{\max Z^s}{s}$ be the error after these s ($\geq \sigma$) R-steps. Then, it holds*

$$\frac{s}{a}\ \delta(s) \;\leq\; \sigma \;\leq\; Q\ s\ \delta(s) \quad \text{ and } \quad \delta(s) \to 0 \text{ as } s \to \infty. \qquad \diamond \tag{4.1}$$

Proof. Since $a_{i,i} = 0$, the value $\max Z^s$ remains constant during the iterations $s+1, ..., s+m$ (m determined via (3.6)) and increases at iteration $s' = s+m+1$. Since all components of Z^s are sums of elements $a_{i,j}$, the value $\max Z^s$ increases at least by Q^{-1} and at most by a. Hence, it holds $Q^{-1}\ \sigma \ \leq \ \max Z^s \ \leq \ a\ \sigma$, and the error fulfills $Q^{-1}\ \frac{\sigma}{s} \leq \delta(s) \leq a\ \frac{\sigma}{s}$. In consequence, we have

$$\frac{\delta(s)}{a} \leq \frac{\sigma}{s} \quad \text{and} \quad \frac{\sigma}{s} \leq Q\ \delta(s).$$

Now, the assertion follows immediately by multiplying with s and taking into account that $\delta(s) \to 0$ as $s \to \infty$ is ensured by J. Robinson [6]. □

Remark 1 Thus we obtain an error $\alpha(x) \leq \delta$ already after σ steps where σ corresponds to $s \geq \sigma$ steps of the R-method and fulfills

$$\sigma \ \leq Q\ \delta\ s. \tag{4.2}$$

Since the error δ vanishes for increasing σ (or s), our order of convergence is better from a *qualitative* point of view (i.e., not only by some constant factor). ◇

Furthermore, (4.2) tells us that we are as faster (compared with the R-method) as smaller the error is required and as faster the R-method converges.

Each step of modR1 consists of n additions, n operations of the form $(u-v)/a_{i,j}$ and finding $\max Z$. Hence the new steps are more expensive than before, but they require (roughly speaking) only $5n$ *elementary operations.* The computing time per step increases about by factor 3 to 7 (depending on the generated code). For the subsequent example 3 and $n = 1000$, 10^5 iterations needed about 2.1 and 7.6 sec, respectively (all our tests used a x86 processor, 3 Ghz).

4.2 Estimates with and without assuming periodicity

The convergence of the R-method and modR1 (in the sense of $\alpha(x^s) \to 0$) are clear by [6]. One has only to note that modR1 (for rational and real matrices as well) aggregates steps of the R-method. Therefore, we consider here algorithm modR2.

Preliminaries.
Since $A = -A^T$ the equation $y^T A y = 0$ is evident and yields by (3.15) for each iteration, $y^T z = y^T \zeta \leq 0$. Thus, some component of $z(s)$, say the νth one, satisfies

$$z(s)_\nu \leq 0 \quad \text{where } \nu = \nu(s). \tag{4.3}$$

To simplify, we do not exploit here, that components k, which are never active for sufficiently large s, can be deleted in the current context.

Case 1: Let the νth component be maximal at some step $s' > s$. Then one obtains

$$\Delta z_\nu := z(s')_\nu - z(s)_\nu \geq s'. \tag{4.4}$$

In consequence, also $t = ||y||_1$ has to increase in a corresponding way, namely

$$\Delta t := t(s') - t(s) \geq \frac{s'}{a}. \tag{4.5}$$

Case 2: Now assume that component ν is not maximal during the iterations $s, ..., s'$ and write $s' = s + L + 1$. Then the algorithm behaves, during the L steps $s, .., s' - 1$ (where ν plays no role), as in the $(n-1)$-subgame $A^{(-\nu)}$ without strategy ν and with the initial point $\zeta_i = z(s)_i - s \ \forall i \neq \nu$. This fact is the key for induction proofs like in [6].

Periodic sequences.
Assume that case 1 happens M times at the iterations $s \in \{s_0, s_1, ..., s_{M-1}\}$ with $s'_m = s_{m+1}$. Setting $T_m := t(s_m)$, the inequality (4.5) yields for $m = 0, ..., M-1$,

$$T_M - T_0 \geq a^{-1} \sum_{m=0}^{M-1} s_{m+1} \geq M \ a^{-1} s_0 \ + \ \tfrac{1}{2} a^{-1} M(M+1). \tag{4.6}$$

Hence t increases quadratically with respect to the number of iteration segments the minimal component of z becomes maximal in which.

Definition: If positive integers p and s_0 exist such that the current case occures for

$$s_m = s_0 + mp, \qquad m = 0, 1, 2, ...$$

we call the iteration sequence *periodic*, otherwise *aperiodic*.

Roughly speaking, then the minimal components of z become maximal sufficiently fast (within p steps). In this situation, (4.6) ensures, by complete induction, that t increases quadratically

$$t(s_0 + mp) \geq t(s_0) + \tfrac{1}{2} \ p \ a^{-1} \ m(m+1) \ \text{ if } \varepsilon \geq 0, m > 0. \tag{4.7}$$

Since

$$\max z(s_0 + mp) = s_0 + mp,$$

the convergence of the error value z/t is completely described now. In particular, (4.7) yields for $\zeta = 0$,

$$\alpha(\ x(s_0 + mp) \ ) = \frac{\max z(s_0 + mp)}{t(s_0 + mp)} \leq \frac{2a \ (s_0 + mp)}{pm \ (m+1)} = \frac{2a \ s_0}{pm \ (m+1)} + \frac{2a}{m+1}.$$

This tells us that

$$\alpha(\ x(s) \ ) \leq \frac{C}{s} \qquad \forall s \tag{4.8}$$

holds with some $C > 0$ and shows that (1.1) is at least true up to some constant.

For $n = 3$, one easily sees that $t(.)$ can *at most* quadratically increase (in each column there is exactly one positive entry, so the active indices form a periodic sequence).

Aperiodicity.
*modR*2 may generate aperiodic sequences. Even both types of sequences may appear for the same matrix:

$$A = \begin{pmatrix} 0 & -1 & 0 & 1 & -2 \\ 1 & 0 & -1 & 0 & 1 \\ 0 & 1 & 0 & -1 & 0 \\ -1 & 0 & 1 & 0 & 0 \\ 2 & -1 & 0 & 0 & 0 \end{pmatrix}$$

Taking $\zeta = (0, 0, -6, 0, 0)$, the sequence of active indices becomes periodical :
(1 5 2 3 4) (1 5 2 3 4) ... .
For $\zeta = (\ 0, -145.5, -325,\ -1, -307)$, the index sequence attains the form
(1 2 3 4) ... (1 2 3 4) [1 5 2 3 4 1 5 2 3 4], (1 2 3 4) ... (1 2 3 4) [1 5 2 3 4 1 5 2 3 4], ...
where the number of repetitions of cycle (1 2 3 4) is 8 in the beginning and increases exactly by 2 after passing the 2 cycles (1 5 2 3 4). Periodicity is violated since the minimal component z_5 does not increase fast enough. The computed solution is $\frac{1}{6}(1, 2, 1, 2, 0)$ in both situations.

The exact analytical proof of this (a)periodical behavior required to determine all $z_i(s)$ explicitly for both sequences (via difference equations and composed affine mappings) and is quite technical. We recommend the interested reader to verify these statements by a computer. For the subgame without row/column 5, the algorithm determines the solution $\frac{1}{4}(1, 1, 1, 1)$, not the vertex $\frac{1}{2}(1, 0, 1, 0)$ of the solution set.

Another aperiodical example has the form

$$\hat{A} = \begin{pmatrix} 0 & 0 & 0 & -1 & 1 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & 1 & 0 & 0 & -1 \\ 1 & -1 & 0 & 0 & -2 \\ -1 & 0 & 1 & 2 & 0 \end{pmatrix}$$

and is of particular interest since $\hat{A} = G$ (2.2) corresponds to the simple linear program

$$(P) \quad \max x_1 \quad s.t.\ x_1 - x_2 \le 2,\ x_2 \le 1,\ x \ge 0.$$

The computed solution $\frac{1}{7}(3, 1, 1, 1, 1)$ yields the primal-dual solutions $x = (3, 1)$, $y = (1, 1)$. Now, there are two possible (shortest) cycles of active strategies: $C_1 = (4\ 5\ 1)$ and $C_2 = (4\ 2\ 3\ 5\ 1)$. After each of these cycles, $i = 4$ is again active, and, depending on

$$\tfrac{1}{2}\ [1 + z_4(s) - z_5(s)] \le 1 + z_4(s) - z_2(s),$$

a repetition of C_1 or C_2 follows. One sees quite simple that both cycles must be used since modR2 solves the game. The application of modR2 shows that there are arbitrarily long periods of repetitions of C_1 (z_4, z_5 and z_1 are positive there) and one can theoretically compute that p repetitions of C_1 transform z into some z' depending quadratically on p (in particular, z_2 increases quadratically). However, p repetitions of C_2, define an exponential (in p) transformation of z and disturb periodic behavior. Computing the sequence $z(s)$ explicitly is again rather hard.

Nevertheless, the crucial formula (4.9), which implies (1.1) for $\zeta = 0$, was true for both examples and all mentioned initial values. We permit all index sequences now.

Estimates for the general case.

Assume $\zeta = 0$. Formula (1.1) for modR2 then just requires, since $\alpha(x^s) = s/t(s)$,

$$t(s) \ge \frac{s^2}{n\ a}. \tag{4.9}$$

Remark 2 For each feasible ζ, (4.9) is valid if the maximal component $z_i(s')$ and the value $z_k(s')$ for the next active index $k = i(s' + 1)$ satisfy

$$z_i(s') - z_k(s') + 1 \ \ge\ \frac{2s' + 1}{n}, \qquad s' = 0, 1, ..., s - 1. \qquad \diamond \tag{4.10}$$

The proof is straightforward since, due to $h(s') \geq \frac{z_i(s')-z_k(s')+1}{a}$, (4.10) implies

$$a\ t(s) \geq \sum_{s'<s} (z_i(s') - z_k(s') + 1) \geq \frac{1}{n} \sum_{s'<s} (2s' + 1) = \frac{(s-1)s + s}{n} = \frac{s^2}{n}\ .$$

Obviously, (4.10) is one of many conditions which ensure (4.9) and (1.1) via

$$\sum_{s'<s} (z_i(s') - z_k(s') + 1) \geq \frac{s^2}{n}, \tag{4.11}$$

a condition which is equivalent to (4.9), provided that $a = 1$.

Nevertheless (4.10) hold for $n = 3$ and each ζ. Indeed, for $s' = 0$ (4.10) is trival. For $s > 0$, the vectors $z(s)$ satisfy $z(s)_{i(s)} = s,\ z(s)_{i(s-1)} = s-1$. The remaining component $z_k(s)$ is not positive. Since $a_{i(s),i(s-1)}$, used for the index change $i(s-1) \to i(s)$, is positive, we have $a_{i(s-1),i(s)} < 0$ which yields $k = i(s+1)$. Thus (4.10) follows from

$$z_i(s) - z_k(s) + 1 \geq s + 1 > \frac{2s+1}{3} \quad \forall s.$$

Next we verify convergence $\alpha(x^s) \to 0$ at all and apply a reduction argument as in [6].

Theorem 3 *(Convergence of modR2)*
There is some $c = c(A) > 0$, such that, for all initial vectors ζ and all iterations $s > 0$,

$$t(s) \geq c\ s^p \quad where \quad p \geq \frac{n-1}{n-2}. \tag{4.12}$$

Proof. We already know, that (4.12) is true for $n = 3$ and $p = 2$. Starting with $n = 4$ we may assume that, with certain $p > 1$ and $c > 0$, (4.12) is true (again with all initial vectors ζ') for all subgames A' of A which arise by deleting any row and the related column of A. Evidently, also $c \leq a^{-1}$ may be assumed. We put $q = 1/p \quad (< 1)$ and consider large s such that

$$1 + s^q < s \quad \text{and} \quad s^q/(1+s^q) > \tfrac{1}{2}. \tag{4.13}$$

Let $L = L(s)$ be the unique integer in the half-open interval $[s^q, 1+s^q)$ and N be the smallest integer satisfying $N \geq s/(1+s^q)$. Then

$$\begin{array}{lll} N & \geq\ s^{1-q}\ s^q/(1+s^q)\ >\ \tfrac{1}{2}\ s^{1-q} & \text{and} \\ NL & \leq\ [1 + s/(1+s^q)]\ (1+s^q)\ =\ 1 + s^q + s\ < 2s. & \end{array}$$

Beginning with step s we study the algorithm during the N (overlapping) intervals

$$J_k = [\ s+kL,\ s+kL+1,\ s+kL+2,\ \ldots\ ,\ s+(k+1)L\ ], \quad k = 0, 1, \ldots, N-1$$

of exactly $L + 1$ iteration steps.

If *some index i is not active* in J_k, i.e., $i(s') \neq i\ \forall s' \in J_k$, then the algorithm yields, for the L steps assigned to s' with $s + kL\ \leq\ s'\ <\ s + (k+1)L$, the same stepsizes $h(s')$ as in the subgame without row and column i and with the initial vector $\zeta' \in \mathbb{R}^{n-1}$, defined by

$$\zeta'_j = z(u)_j - \max z(u) \text{ for all } j \neq i \quad \text{where } u = s + kL.$$

Hence, t increases, during the L steps in $J_k \setminus \{s + (k + 1)L\}$, according to (4.12) at least by the amount $\Delta t_k \geq cL^p \geq cs$.

If *each index is active* in J_k, then (4.5) ensures that t increases during the same L steps in $J_k \setminus \{s + (k + 1)L\}$, at least by $\Delta t_k \geq a^{-1}s \geq cs$.

In consequence, t increases at least by cs in each interval $J_k \setminus \{s + (k + 1)L\}$. Considering all intervals we thus observe

$$t(3s) \geq t(s) + Ncs \geq t(s) + \tfrac{1}{2}\, s^{1-q}cs > \tfrac{1}{2}c\, s^{p'} \quad \text{with } p' = 2 - q > 1. \tag{4.14}$$

This inequality holds true for all s satisfying (4.13), i.e., for all sufficiently large $s > s_0$. Since t is strongly increasing, so (4.14) provides us with the existence of some (possibly small) $c' > 0$ such that $t(s) \geq c's^{p'}$ for all s.

Estimate of the constants: Since (4.12) holds for $n = 3$ and $p = 2$, the construction with $q = 1/p$ yields (worst) exponents $p(n)$ as $p(4) = 3/2, ..., p(n) = 2 - \frac{1}{p(n-1)} = \frac{n-1}{n-2}$. □

Remark 3 With the same proof, Theorem 3 holds for modR1 and integer matrices (to have again the applied inequality (4.5) in case 2). One has only to replace s by σ and $t(s)$ by $s(\sigma) = \|y(\sigma)\|$. Because of $\max z(\sigma) \leq a\sigma$, so (4.12) guarantees again convergence $\alpha(x(\sigma)) \to 0$ of the error for modR1 and, in consequence, also for the R-method.

5 Numerical tests

The estimates of Theorems 3 are not sharp. One reason arises from the fact that (with higher technical effort) the construction of the related intervals J_k could be improved. There is, however, a more important fact: After starting with the comfortable induction hypothesis in the proof, we used (like J. Robinson) only the following: If the algorithm "plays" a certain time only with $m < n$ active strategies, the values $t(s)$ increase as in any "worst" game B' with less than n strategies and entries $b_{i,k} \leq a$.

It has *not been exploited* that we are really playing in a *particular subgame* A' of A with a *specific* initial vector. This subgame-property completely determines the combinatorial structure (the sequence of active strategies) of the algorithm.

So it is not surprising that the *real numerical behavior* of the algorithm was much better for all checked examples.

5.1 Practical comparison of the R-method and modR1

We compare the results for 3 types of matrices and start with $\zeta = 0,\ i(0) = 1$.
Example 1.

$$A = \begin{pmatrix} 0 & 1 & -2 \\ -1 & 0 & 3 \\ 2 & -3 & 0 \end{pmatrix}$$

Example 2. A is the (n, n) matrix $a_{i,k} = 1 + i - [k/2]$ for $k > i$.
Here, $[r]$ denotes the integer part of r, and the remaining elements $a_{i,k}$ $(k \leq i)$ are given via $A^T = -A$ $(a \approx n/2)$.

Example 3: A is a (n, n) skew-symmetric Hilbert-type matrix such that, for $i < k$,

$a_{i,k} = i/(i+k)$ if $i+k$ is odd and $a_{i,k} = -i/(i+k)$ if $i+k$ is even.
Again, for $i \geq k$ the value of $a_{i,k}$ is given by $A^T = -A$ $(a \approx 1/2)$.

We summarize results for example 1 and for the examples 2, 3 with different n. Of course, the listed time is relative and depends on the used hardware. It is added only for getting more convenient comparisons. If the time for the R-method is not indicated, it was too big. Then we list the number $s(\sigma)$, cf. (3.10), of aggregated R-steps by σ steps of modR1.

Example and error		*steps/time*	*R − METH*	*steps/time*	*modR1*
(1) $n = 3$	$\delta = 1E-3$	$s = 1998000$	1 *sec*,	$\sigma =$ 1000	0 *sec*
(2) $n = 50$	$\delta = 1E-2$	$s = 5.8 * 10^6$	15 *sec*,	$\sigma =$ 7423	0 *sec*
	$\delta = 1E-3$	$s = 5.7 * 10^8$	1500 *sec*,	$\sigma =$ 74000	0.6 *sec*
	$\delta = 1E-4$	$s(\sigma)$:	$5.7 * 10^{10}$	$\sigma =$ 740000	6.4 *sec*
	$\delta = 1E-5$	$s(\sigma)$:	$5.7 * 10^{12}$	$\sigma =$ 7398000	63.4 *sec*
with $n = 200$:					
	$\delta = 1E-2$	$s = 52 * 10^6$	459 *sec*,	$\sigma =$ 122000	2 *sec*
	$\delta = 1E-3$	$s(\sigma)$:	$3.7 * 10^{10}$	$\sigma = 1.2 * 10^6$	21.2 *sec*
	$\delta = 1E-4$	$s(\sigma)$:	$3.7 * 10^{12}$	$\sigma = 12 * 10^6$	210 *sec*
	$\delta = 1E-5$	$s(\sigma)$:	$3.7 * 10^{14}$	$\sigma = 12 * 10^7$	2085 *sec*
(3) $n = 200$	$\delta = 2E-4$	$s = 10 * 10^6$	89 *sec*,	$\sigma =$ 27000	0.5 *sec*
	$\delta = 1E-4$	$s = 46 * 10^6$	405 *sec*,	$\sigma =$ 73000	1.2 *sec*
	$\delta = 1E-5$	$s(\sigma)$:	$3.5 * 10^{10}$	$\sigma = 1.8 * 10^6$	32 *sec*
with $n = 1000$:					
	$\delta = 2E-4$	$s = 5.7 * 10^6$	119 *sec*,	$\sigma =$ 6000	0.4 *sec*
	$\delta = 1E-4$	$s = 30 * 10^6$	620 *sec*,	$\sigma =$ 12000	0.7 *sec*
	$\delta = 1E-5$	$s(\sigma)$:	$4 * 10^{10}$	$\sigma = 1.9 * 10^6$	150 *sec*
with $n = 5000$:					
	$\delta = 2E-4$	$s = 13 * 10^6$	2873 *sec*,	$\sigma =$ 10000	5.8 *sec*
	$\delta = 1E-4$	$s(\sigma)$:	$3.9 * 10^7$	$\sigma =$ 12000	8.2 *sec*
	$\delta = 1E-5$	$s(\sigma)$:	$5 * 10^9$	$\sigma = 2 * 10^5$	139 *sec*

The number of steps to obtain a relative error $\alpha(x)/a < \delta = 5E-4$:

Example	*steps/time*	*R − METH*	*steps/time*	*modR1*
(1) $n = 3$	$s = 889000$	5.7 *sec*,	$\sigma =$ 1000	< 0.01 *sec*
(2) $n = 50$	$s = 3.7 * 10^6$	25 *sec*,	$\sigma =$ 6000	0.05 *sec*
with $n = 200$:	$s = 15 * 10^6$	55 *sec*,	$\sigma =$ 30000	0.34 *sec*
with $n = 1000$:	$s(\sigma)$:	$74 * 10^6$	$\sigma =$ 120000	7.7 *sec*
with $n = 5000$:	$s(\sigma)$:	$3.7 * 10^8$	$\sigma = 6 * 10^5$	383 *sec*
(3) $n = 200$	$s = 21 * 10^6$	189 *sec*,	$\sigma =$ 21000	0.36 *sec*
with $n = 1000$:	$s = 4.2 * 10^6$	106 *sec*,	$\sigma =$ 5000	0.38 *sec*
with $n = 5000$:	$s(\sigma)$:	$15 * 10^6$	$\sigma =$ 10000	6.8 *sec*

The reader may wonder why modR1 is surprisingly effective just for example 3 and high dimension. Due to the ill-posed matrix, the set of x with $\alpha(x) < \delta$ is quite big. Note that, e.g., for $x_j = 1/n \; \forall j$, it holds $\max Ax < \frac{1}{n}$. In contrast, the distance $\operatorname{dist}(x^\sigma, x^*)$ of the approximate solution to the exact one may be much greater.

The presented algorithms are very simple, so it makes no problems to test several own examples and formula (1.1) as well. The results above are part of a big collection of checked games with analog properties.

5.2 Practical behavior of modR1 and modR2

The algorithms modR1 and modR2 showed similar convergence behavior. modR1 is better for getting direct comparisons with the R-method, modR2 allows to control $\max z(s) = s$.

Both algorithms have been compared with the Feijer method:
Successive projection of x^s onto the n halfspaces $H_i = \{x \mid A_{i.}x \leq 0\}$, $H_{n+i} = \{x \mid x_i \geq 0\}$ and the hyperplane $H_{2n+1} = \{x \mid \sum x_i = 1\}$.
For points very close to a solution, the Feijer method (linear order of convergence but more expensive steps) was better. However, to obtain such approximations, it was clearly worser.

A similar effect can be observed when the methods are compared with nonsmooth Newton methods (cf. [2] or [4]), applied to $\alpha(x) = 0$ via active index sets, i.e., solve

$$L(x^s, \gamma): \quad A_{j.}\, x = 0, \quad \sum x_i = 1, \;\; x_i = 0 \text{ if } x_i^s < \gamma \qquad \text{for all } j \text{ satisfying } x_j^s \geq \gamma$$

and put $x^{s+1} = x, \; s = s + 1$. Here, $\gamma > 0$ should be small, in order to distinguish between "active and inactive" strategies at iteration s.
It is not hard to show that, generically in the class of skew symmetric matrices A, (2.1) has a unique solution x^* with an odd number of positive components and $L(x^s, \gamma)$ has the unique solution $x = x^*$ whenever $\delta := \gamma + \|x^s - x^*\| \leq \delta_0$ is sufficiently small.
But the radius δ_0 is usually small, and in fact, in tests with $n \approx 100$ and $a \approx 100$, the error δ had to be (often) very small, already in order to obtain feasible $x \geq 0$ and solvability of $L(x^s, \gamma)$ at all.

Algorithm modR2 has been also compared with the simplex method. The linear problems (P, D) have been taken
(i) with uniformly distributed random integers $a_{i,j} \in [1, 110], c_j \in [20, 80], b_i \in [100, 110]$.
(ii) with particularly degenerated matrices (2 identical constraints) of the same type.
For (i), with the required *relative error* $\delta_{opt} = 2E - 4$,
(measured by the maximal violation of the $n + m + 1$ crucial *LP- inequalities,*
and the maximal input $a := \max\{a_{i,j}, c_j, b_i\}$),
modR2 began to be better for $n \approx m \approx 1000$ (even more for (ii)). The mean values of

the time for the simplex method and for modR2 applied to G (2.2), were about:

		SI-method : modR2, $\delta_{opt} = 2 * 10^{-4}$,
for	$n = m = 1000$,	05 : 06 (sec)
for	$n = m = 1500$,	17 : 12
for	$n = m = 2000$,	36 : 22
for	$n = m = 2500$,	70 : 34
for	$n = m = 3000$,	122 : 57
for	$n = m = 3500$,	200 : 85
for	$n = m = 4000$,	270 : 105
for	$n = m = 4500$,	440 : 150
for	$n = m = 5000$,	794 : 184.

In all tests with modR1 and modR2 and $\zeta = 0$, the iteration number s for getting the absolute error $\alpha(x) \leq \delta$, fulfilled (1.1).

The formula has been checked, with modR2, for more than 10^7 (pseudo-) randomly generated games of dimension $5 \leq n \leq 5000$ and different δ (depending on n). In addition, various attempts of finding a conterexample had no success.

It has been also checked for a few number of games with higher dimension (until $n = 10^6$) and a large number of LP-problems and arbitrary matrix games, again with the maximal size of $(5000, 5000)$ and dense integer and "real" matrices. It was true for all assigned games G (2.2).

Arbitrary initial points.
If some approximate solution $\hat{x} \in S_n$ is already known, one can start at $\hat{x}$ by fixing some integer $s > 0$, setting $z(s) = sA\hat{x}$, $y(s) = s\hat{x}$ and passing to step s of the related algorithm (for modR2, $\max z(s'+1) = \max z(s')+1$ remains true, not so $\max z(s) = s$). The next steps, however, may generate points $x(s')$ far from $x(s) = \hat{x}$ (if s was small) or points which remain a long time very close to $\hat{x}$ (if s was big). To find a reasonable empirical value for s, one can determine the error at $\hat{x}$ and apply (1.1) to put $s \approx n \frac{a}{2\alpha(\hat{x})}$ (which is often big, too).

Nevertheless, *the general validity of (1.1) remains an open question*, and - besides to improve J. Robinsons algorithm- the hope that some interested reader has an idea for its proof or a counterexample, was a main reason for writing this paper.

References

[1] G.W. Brown. Iterative solution of games by ficticious play. *Activity Analysis of Production and Allocation*, 1951.

[2] F. Facchinei and J.-S. Pang. *Finite-Dimensional Variational Inequalities and Complementary Problems, Vol I and Vol II.* Springer, New York, 2003.

[3] S.I. Gass and P.M.R. Zafra. Modified ficticious play for solving matrix games and linear-programming problems. *Computers Oper. Res.*, 22:893–903, 1995.

[4] D. Klatte and B. Kummer. *Nonsmooth Equations in Optimization - Regularity, Calculus, Methods and Applications.* Nonconvex Optimization and Its Applications. Kluwer Academic Publ., Dordrecht-Boston-London, 2002.

[5] Y. Nesterov and A. Nemirovskii. *Interior–Point Polynomial Algorithms in Convex Programming.* SIAM Studies in Applied Mathematics. SIAM, Philadelphia, 1994.

[6] J. Robinson. An iterative method of solving a game. *Annals of Mathematics*, 54:296–301, 1951.

[7] M. Epelman T. Lambert and R. Smith. A fictious play approach to large-scale optimization. *Operations Research*, 53:477–489, 2005.

[8] N. N. Vorobiev. *Theory of games: Lessons for economics and cybernetics.* University press, Univ. of Leningrad, Leningrad, 1974. in Russian.