11.4

NATIONAL BUREAU OF STANDARDS REPORT

2501

COMPUTATIONAL EXPERIENCE IN SOLVING LINEAR PROGRAMS

by

A. Hoffman, M. Mannos, D. Sokolowsky and N. Wiegmann

NBS

U. S. DEPARTMENT OF COMMERCE
NATIONAL BUREAU OF STANDARDS

U. S. DEPARTMENT OF COMMERCE.
Sinclair Weeks, *Secretary*

NATIONAL BUREAU OF STANDARDS
A. V. Astin, *Director*

THE NATIONAL BUREAU OF STANDARDS

The scope of activities of the National Bureau of Standards is suggested in the following listing of the divisions and sections engaged in technical work. In general, each section is engaged in specialized research, development, and engineering in the field indicated by its title. A brief description of the activities, and of the resultant reports and publications, appears on the inside of the back cover of this report.

Electricity. Resistance Measurements. Inductance and Capacitance. Electrical Instruments. Magnetic Measurements. Applied Electricity. Electrochemistry.

Optics and Metrology. Photometry and Colorimetry. Optical Instruments. Photographic Technology. Length. Gage.

Heat and Power. Temperature Measurements. Thermodynamics. Cryogenics. Engines and Lubrication. Engine Fuels. Cryogenic Engineering.

Atomic and Radiation Physics. Spectroscopy. Radiometry. Mass Spectrometry. Solid State Physics. Electron Physics. Atomic Physics. Neutron Measurements. Infrared Spectroscopy. Nuclear Physics. Radioactivity. X-Rays. Betatron. Nucleonic Instrumentation. Radiological Equipment. Atomic Energy Commission Instruments Branch.

Chemistry. Organic Coatings. Surface Chemistry. Organic Chemistry. Analytical Chemistry. Inorganic Chemistry. Electrodeposition. Gas Chemistry. Physical Chemistry. Thermochemistry. Spectrochemistry. Pure Substances.

Mechanics. Sound. Mechanical Instruments. Aerodynamics. Engineering Mechanics. Hydraulics. Mass. Capacity, Density, and Fluid Meters.

Organic and Fibrous Materials. Rubber. Textiles. Paper. Leather. Testing and Specifications. Polymer Structure. Organic Plastics. Dental Research.

Metallurgy. Thermal Metallurgy. Chemical Metallurgy. Mechanical Metallurgy. Corrosion.

Mineral Products. Porcelain and Pottery. Glass. Refractories. Enameled Metals. Concreting Materials. Constitution and Microstructure. Chemistry of Mineral Products.

Building Technology. Structural Engineering. Fire Protection. Heating and Air Conditioning. Floor, Roof, and Wall Coverings. Codes and Specifications.

Applied Mathematics. Numerical Analysis. Computation. Statistical Engineering. Machine Development.

Electronics. Engineering Electronics. Electron Tubes. Electronic Computers. Electronic Instrumentation.

Radio Propagation. Upper Atmosphere Research. Ionospheric Research. Regular Propagation Services. Frequency Utilization Research. Tropospheric Propagation Research. High Frequency Standards. Microwave Standards.

Ordnance Development.
Electromechanical Ordnance.
Ordnance Electronics.
These three divisions are engaged in a broad program of research and development in advanced ordnance. Activities include basic and applied research, engineering, pilot production, field testing, and evaluation of a wide variety of ordnance matériel. Special skills and facilities of other NBS divisions also contribute to this program. The activity is sponsored by the Department of Defense.

Missile Development. Missile research and development: engineering, dynamics, intelligence, instrumentation, evaluation. Combustion in jet engines. These activities are sponsored by the Department of Defense.

- Office of Basic Instrumentation
- Office of Weights and Measures.

NATIONAL BUREAU OF STANDARDS REPORT

NBS PROJECT **NBS REPORT**

1102-10-1104 May 15, 1953 2501

COMPUTATIONAL EXPERIENCE IN SOLVING LINEAR PROGRAMS*

by

A. Hoffman, M. Mannos, D. Sokolowsky and N. Wiegmann
National Bureau of Standards

*This work was supported (in part) by the Office of Scientific Research, USAF.

COMPUTATIONAL EXPERIENCE IN SOLVING LINEAR PROGRAMS*

by

A. Hoffman, M. Mannos, D. Sokolowsky and N. Wiegmann
National Bureau of Standards

Introduction

This paper is a discussion of three methods which have been employed to solve problems in linear programming, and a comparison of results which have been yielded by their use on the Standards Eastern Automatic Computer (SEAC) at the National Bureau of Standards.

A linear program is essentially a scheme to run an organization or effect a plan efficiently, i.e., it is a technique of management which serves to minimize costs, maximize returns or achieve other ends of a similar nature. To illustrate the kind of "life situation" to which linear programming is applicable, and the technique of formulating the circumstances mathematically, let us examine a particular problem. For this purpose, we choose a simplification of the so-called "caterer problem" of W. Jacobs.

A caterer knows that in connection with the meals he has arranged to serve during the next $\underline{n}$ days, he will need $r_j (> 0)$ fresh napkins on the j-th day, $j = 1, 2, \ldots, n$. Laundering takes

*This work was supported (in part) by the Office of Scientific Research, USAF. The coding and operation of the methods was performed by R. Bryce, I. Diehm, L. Gainen, B. Handy, Jr., B. Heindish, N. Levine, F. Meek, S. Pollack, R. Shepherd and O. Steiner.

p days; that is, a soiled napkin sent for laundering at the end of the j-th day is returned in time to be used again, on the (j+p)th day. Having no usable napkins on hand or in the laundry, the caterer will meet his early needs by purchasing napkins at a cents each. Laundering costs b cents per napkin. How does he arrange matters to meet his needs and minimize his outlays for the n days?

Before expressing the caterer's problem algebraically, two conventions of notation will be stated. The subscript j throughout has the range 1,2,...,n; every equation involving j is to hold for the entire range of values. Quantities with subscripts outside this range are always zero.

Let x_j represent the napkins purchased for use on the j-th day; the remaining requirements, if any, are supplied by laundered napkins. Of the r_j napkins which have been used on that day plus any other soiled napkins on hand, let y_j be the number sent to the laundry and s_j the stock of soiled napkins left. Consequently

(1) $$y_j + s_j - s_{j-1} = r_j$$

The stock of fresh napkins on hand on the j-th day must be at least as great as the need. Thus

(2) $$\sum_{i=1}^{j} x_i + \sum_{i=1}^{j} y_{i-p} \geq \sum_{i=1}^{j} r_i$$

The total cost to be minimized, subject to the constraints (1) and (2) on the nonnegative variable x_j, y_j, s_j, is

$$\sum_{j=1}^{n} (a\ x_j + b\ y_j)$$

This is a mathematical formulation of the problem the caterer wishes to solve.

If desired, the equation (1) can be changed into inequalities. For example, (1) is equivalent to the pair of inequalities

$$y_j + s_j - s_{j-1} \geqq r_j$$

$$-y_j - s_j + s_{j-1} \geqq -r_j.$$

If we make this change, then it is clear that the problem just described is, mathematically, a special case of the following: Let $A = (a_{ij})$ be a given m x n matrix; $b = (b_1,\ldots,b_m)$ an m-dimensional vector, $c = (c_1,\ldots,c_n)$ an n-dimensional vector. For all vectors $x = (x_1,\ldots,x_n)$ satisfying

(3) $$x_i \geqq 0$$

and

(4)
$$\begin{aligned} a_{11}\, x_1 + a_{12}\, x_2 + \ldots + a_{1n}\, x_n &\geqq b_1 \\ a_{21}\, x_1 + \ldots + a_{2n}\, x_n &\geqq b_2 \\ &\vdots \\ a_{m1}\, x_1 + \ldots + a_{mn}\, x_n &\geqq b_m \end{aligned}$$

(briefly, $Ax \geqq b$).

minimize $(c,x) = c_1x_1+\ldots+c_nx_n$.

The foregoing is the mathematical statement of the general linear programming problem. In geometric language, it is to find a point on a convex polyhedron (the region satisfying (3) and (4)) at which a given linear form $(c_1x_1+\ldots+c_nx_n)$ is a minimum.

The Computation Laboratory of the National Bureau of Standards, with the sponsorship and close cooperation of the Planning and Research Division of the Office of the Air Comptroller, U. S. Air Force, has been engaged in the task of discovering ane evaluating methods for computational attack on this problem, and this paper is in a sense a progress report on a part of this work (see also Orden [10]).

The three techniques that have received most attention so far are (a) the "Simplex" method, devised by George Dantzig, (b) the Fictitious Play method of George Brown and (c) the Relaxation Method of T. S. Motzkin. Each will be described in more detail in the next section, but for the present it is appropriate to remark that the simplex method is a finite algorithm, and the other two are infinite processes. Further, the other two methods are designed to solve not the linear programming problem _per se_ but two related problems: fictitious play finds a solution to a matrix game - i.e., a zero-sum 2-person game in normalized form - and relaxation finds an x satisfying (4) - i.e., solves a system of linear inequalities. It is known, however (see Gale, Kuhn and Tucker [6], Dantzig [4], Orden [9] that the three problems (i) solving a linear program, (ii) solving a matrix game, (iii) solving a system of linear inequalities are in general equivalent in that each of (i), (ii) and (iii) can be so formulated that it becomes either of the other two.

For purposes of comparison, the following experiment was undertaken. Several symmetric matrix games (i.e., games whose matrices were skew-symmetric) were attacked by each method in turn and the results studied with respect to the accuracy achieved and the time required to obtain this accuracy. Many conjectures about the relative merits of the three methods by various criteria could only be verified by actual trial. Apart from the descriptions of the methods, the paper is concerned principally with the results of the experiment, but some other aspects of the comparison, revealed more strikingly by other computations, will also be mentioned.

The games in question have as payoff matrices the submatrices of order 5,6,7,8,9,10 obtained from the following 10x10 array by deleting the last five rows and columns, the last four rows and columns, etc.

$$\begin{bmatrix} 0 & 1 & -2 & -1 & 3 & -2 & -1 & -4 & 1 & -2 \\ -1 & 0 & -1 & 1 & 2 & -2 & 1 & 1 & -1 & -1 \\ 2 & 1 & 0 & -3 & 1 & 1 & 3 & -3 & 1 & -1 \\ 1 & -1 & 3 & 0 & 1 & -1 & 4 & 2 & -1 & 5 \\ -3 & -2 & -1 & -1 & 0 & -1 & -5 & 6 & 1 & 6 \\ 2 & 2 & -1 & 1 & 1 & 0 & -2 & -1 & -1 & 3 \\ 1 & -1 & -3 & -4 & 5 & 2 & 0 & 2 & 1 & -4 \\ 4 & -1 & 3 & -2 & -6 & 1 & -2 & 0 & 1 & -1 \\ -1 & 1 & -1 & 1 & -1 & 1 & -1 & -1 & 0 & -5 \\ 2 & 1 & 1 & -5 & -6 & -3 & 4 & 1 & 5 & 0 \end{bmatrix}$$

2. Description of the Methods

(a) The Simplex Method.

The simplex method solves the problem: minimize

$$c_1x_1+\dots+c_nx_n$$

for all vectors $x = (x_1,\dots,x_n)$ satisfying

$$x_j \geqq 0$$
$$Ax = b$$

where $A = (a_{ij})$ is an m x n matrix and $b = (b_1, \ldots, b_m)$.

This differs slightly from the formulation of the general linear programming problem in which $Ax \geqq b$, but the inequalities can be made into equations by appending dummy nonnegative variables.

An algebraic description of the process* is given in Dantzig [5] and Orden [9], and will not be repeated here. While it is not excessively complicated it is somewhat lengthy, and we merely remark now that it very much resembles elimination methods for solving equations. Even for those familiar with the algebra, however, (as well as for novices), the following geometric interpretation is illuminating. First, to state the problem in geometric language: if $A_1, \ldots, A_n$ are the m-dimensional column vectors of A, let $A_1', \ldots, A_n'$ be the (m+1)-dimensional vectors obtained from $A_1, \ldots, A_n$ by appending $c_1, \ldots, c_n$ respectively as the (m+1)th coordinate. Let C be the cone in (m+1)-space spanned by these vectors. Let B be the line in (m+1)-space consisting of all points whose first m coordinates are $b_1, \ldots, b_m$. The object of the computation is to find the lowest point of B which is also in C, i.e., the point of B whose (m+1)th coordinate is a minimum.

The computation proceeds in the following way: assume that m of the vectors $A_1', \ldots, A_n'$, say $A_{v_1}', \ldots, A_{v_m}'$ are given which are linearly independent and have the property that the m-dimensional cone D they span contains a point of B. (Such a set

*Interesting variations suggested by Charnes [3] and Wolfe [12] have not yet been tested.

of vectors may have to be given initially by an artificial device which we shall not describe here). Of all the remaining vectors A_i', let us look at the subset of those which are on the side of the hyperplane containing D that does not contain the positive (m+1)th coordinate axis. These vectors are all "lower" than D. Each of these vectors can be joined to the hyperplane containing D by a line segment parallel to the (m+1)th coordinate axis. Let A_i' be the vector with the property that this line segment has maximal length - i.e., A_i' is the "lowest" of the low vectors. Then A_i' and a certain set of m-1 of the vectors $A_{V_1}',\ldots,A_{V_m}'$ have the property that the m-dimensional cone they span contains a point of B, and this point will be lower than the intersection of D with B. We replace the discarded vector of the set $A_{V_1}',\ldots,A_{V_m}'$ with A_i' and proceed. This replacement process is an iteration in the simplex method, and clearly the computation must stop after a finite number of iterations with the desired lowest point of the intersection of C and B.

The symmetric games were formulated for the simplex method as follows:

Let $A = (a_{ij})$ be the n x n game matrix. We wish to find an $x = (x_1,\ldots,x_n)$ such that

$$Ax \leqq 0$$
$$x_i \geqq 0$$
$$\sum x_i = 1$$

This is equivalent to

$$(a_{11} + 1)\, x_1 + (a_{12} + 1)\, x_2 + \dots + (a_{1n} + 1)\, x_n + w_1 = 1$$

$$\vdots$$

$$(a_{n1} + 1)\, x_1 + \dots + (a_{nn} + 1)\, x_n + (a_{nn} + 1)\, x_n + w_n = 1$$

$$x_i \geqq 0 \qquad (i = 1,\dots,n)$$

$$w_i \geqq 0 \qquad (i = 1,\dots,n);$$

maximize

$$x_1 + \dots + x_n,$$

(thus minimize $-x_1 - x_2 - \dots - x_n$) which is suitable for simplex computation. We omit the proof of the equivalence as well as the justification for choosing this particular way of formulating the game as a simplex computation, since both depend on technical reasons irrelevant to the main purpose of the paper.

(b) Fictitious Play.

It is well known (see Dantzig [4]), that a computational scheme that will solve symmetric games can be adapted to the solution of linear programming problems. The fictitious play method, devised by G. Brown [2] and proved valid by J. Robinson [11], is a procedure for solving an arbitrary matrix game, but the computation is simpler (particularly from the standpoint of storage) if the game is symmetric. And since our primary interest is in linear programs rather than games per se, we have confined our attention to the symmetric case.

If $A = (a_{ij})$ is skew-symmetric, our object is to solve the system of linear inequalities

$$
(5) \quad
\begin{array}{l}
a_{11}\, x_1 + a_{21}\, x_2 + \ldots + a_{n1}\, x_n \geqq 0 \\
a_{12}\, x_1 + a_{22}\, x_2 + \ldots + a_{n2}\, x_n \geqq 0 \\
\quad\vdots \\
a_{1n}\, x_1 + a_{2n}\, x_2 + \ldots + a_{nn}\, x_n \geqq 0
\end{array}
$$

subject to the conditions $x_i \geqq 0$ $(i = 1,2,\ldots,n)$, $\sum_{i=1}^{n} x_i = 1$.

The fictitious play method consists of forming two sequences of n-dimensional vectors, V(0), V(1), V(2),..., and T(0), T(1), T(2),..., with V(0) = T(0) = 0. V(N) (N=1,2,...) is obtained by adding the r-th row of A to V(N-1), where the r-th coordinate of V(N-1) is the minimum of the coordinates of V(N-1) and of smallest index among all the coordinates equal to the minimum. The smallest index criterion is used in order to be specific but is in no way essential.* T(N) is the same as T(N-1) except for the r-th coordinate which is larger by 1. In effect, the j-th component $T_j(N)$ of T(N) represents the number of times the j-th row of the matrix A has been selected by the above criterion in forming V(N). Hence, if $V_j(N)$ denotes the j-th component of V(N), we have

$$V_j(N) = a_{i1}\, T_1(N) + a_{2j}\, T_2(N) + \ldots + a_{nj}\, T_j(N) = \sum_{i=1}^{N} a_{ij}\, T_i(N)$$

Upon setting $x_i(N) = \frac{T_i(N)}{N}$, this is equivalent to

$$(6) \quad \frac{V_j(N)}{N} = a_{ij}\, x_1(N) + a_{2j}\, x_2(N) + \ldots + a_{nj}\, x_n(N) = \sum_{i=1}^{N} a_{ij} x_i(N)$$

*This procedure is followed in the illustrative example on page 12 but not in the SEAC code.

Since $T_i(N) \geqq 0$, and $T_1(N) + T_2(N)+\dots+T_n(N) = N$, it follows that $x_i(N) \geqq 0$ and $x_1(N) + x_2(N)+\dots+x_n(N) = 1$. If the first player follows the strategy $(x_1(N), x_2(N),\dots,x_n(N))$ his expectation is the least of the expressions (6), and except in the trivial case that the first row of A consists of nonnegative elements, $\min\limits_j \frac{V_j(N)}{N}$ will be negative, which, by (6), implies that the inequalities (5) will not be satisfied. It is however, the main result of J. Robinson's paper that

$$\lim_{N \to \infty} \frac{\min\limits_j V_j(N)}{N} = 0$$

This implies (See Hoffman [7], McKinsey [8]), that the vector $x(N) = (x_1(N),\dots,x_n(N))$ approaches the convex set of all solutions to (5) as N increases indefinitely, though it does not imply (indeed, it is not true) that x(N) converges. Of course, if the first player is willing to follow a strategy such that his expected loss is no greater than ϵ, he may follow the strategy x(N), where $\min\limits_j \frac{V_j(N)}{N} \geqq -\epsilon$.

These considerations suggest that the speed of convergence of the fictitious play method should be determined by deciding how "long" it takes for the process to arrive at a vector x(N) such that the corresponding expected loss is no greater than ϵ for a given decreasing sequence of positive numbers ϵ.

At least two criteria are relevant in measuring "how"long" it takes to attain an ϵ: the size of N and the time consumed on the computer. Both are given in the table of results. A third criterion is suggested by the manner in which the procedure was coded. It is apparent from J. Robinson's proof and readily verifiable by experience that as the computation proceeds the same row vector of A will be added to V(N) for a large stretch of successive values of N. Hence, the code picks out, not only the row to be added to V(N), say A_r, but decides how many times A_r is to be added to V(N) before some other row is added; i.e. it determines a number S(N) such that

$$V(N+1) - V(N) = \ldots = V(N+S(N)) - V(N+S(N) - 1) = A_r$$

but

$$V(N+S(N) + 1) - V(N + S(N)) \neq A_r$$

The number S(N) is the least positive integer not less than

$$M_{r,j} = \min_{a_{rj}<0} \frac{V_r(N) - V_j(N)}{a_{rj}}$$

if $M_{r,j}$ is not an integer (i.e. $S(N) = [M_{r,j}]$) and $S(N) = [M_{r,j}] + 1$ if $M_{r,j}$ is an integer. In the latter case ties resulting between the j-th and k-th components with $k > j$ are resolved in favor of the k-th component.

Then $V(N+S(N)) = V(N) + S(N)\cdot A_r$, $T(N+S(N)) = T(N) + S(N)\cdot\delta_r$, (where δ_r is the r-th unit vector.)

The foregoing computation and subsequent changes in V(N) and T(N) are essentially computational "steps" in the SEAC code and therefore the number of such steps it takes to attain a given ϵ is a third proper criterion by which to measure the convergence rate of the process.

Before listing the experimental data and the conclusions, let us illustrate, except for a slight modification, the preceding descriptions of the process by an example.

The modification consists of letting $S(N)=[M_{r,j}]$ if either $M_{r,j}$ is not an integer or is an integer with $j<r$ and $S(N)=[M_{r,j}]+1$ if $M_{r,j}$ is an integer with $j>r$.

Let G be the payoff matrix

$$G = \begin{bmatrix} 0 & 0 & 2 & -1 \\ 0 & 0 & -1 & 2 \\ -2 & 1 & 0 & 0 \\ 1 & -2 & 0 & 0 \end{bmatrix}$$

which is the same as the illustrative matrix in J. Robinson's paper. The first component of V(0) guides the choice of the row of G added to V(0). Hence the first row of G is added to V(0) in order to obtain V(1). In the next step the fourth component is taken as the minimum and so the fourth row of G is added to V(1) yielding V(2). In the next five steps the second component is the minimum and 5 times the second row of G is added to V(2), giving V(7). In the next two steps one observes that the third component is the minimum and so adds 2 times the third row of G to V(7) to get V(9). It is the first, fourth, second and third components in that order which give rise to the minimum component. Due to the simplicity of the payoff matrix which has but one negative element in each row the minimum component keeps passing through the same cycle as above. For example, in the next cycle the first component is a minimum 13 successive times, the fourth component 2 times, the second component 28 times, and the third component 2 times. That is, S(9) = 13, S(22) = 2, S(24) = 28, and S(52) = 2. Similarly, in the next cycle one may verify that S(54) = 58, S(112) = 2, S(114) = 118, S(232) = 2. It is then observed that for any of the following cycles the first component appears as a minimum 4 times as many plus 6 as it does in the previous cycle. The second component follows this pattern;

whereas, the third and fourth components appear as a minimum, twice in each of the succeeding cycles. In the first cycle then, the first component appears once as the minimum component and in the n-th cycle ($n \geq 2$) the first component appears as the minimum in $13.4^{n-2} + 6(4^{n-3}+4^{n-4}+\ldots+4+1) = 13.4^{n-2}+2.4^{n-2}-2 = 15.4^{n-2} - 2$ successive steps. In the first cycle the second component appears 5 times as the minimum component and in the n-th cycle ($n \geq 2$) the second component appears as a minimum in $28.4^{n-2}+6(4^{n-3}+4^{n-4}+\ldots+4+1) = 28.4^{n-2}+2.4^{n-2}-2 = 30.4^{n-2}-2$ successive steps. The third component enters twice as the minimum component in each cycle; whereas, the fourth component is a minimum one time in the first cycle and 2 times in each cycle thereafter. It is quite clear that $x_3(N) \to 0$ and $x_4(N) \to 0$ as $N \to \infty$. If one computes the x(N) for each N at the end of each cycle, it is seen that $x_1 = 1/3$ and $x_2 = 2/3$. However, by computing x(N) for various other sequence of N's it may be observed that $x_1(N)$ and $x_2(N)$ come close to any value between 1/3 and 2/3 provided their sum is 1. While the sequences of $x_1(N)$ and $x_2(N)$ oscillate as $N \to \infty$, one may pick out a subsequence of the N's such that the corresponding subsequence of the x(N)'s converges to any given optimal strategy.

(c) Relaxation Method.

There have been several proposed versions of the relaxation method and what we call the relaxation method here might more properly be termed the "furthest hyperplane" method.

The object of the computation is to fi-l a point which satisfies a finite system of linear inequalities

$$\sum_{j=1}^{n} a_{ij} x_j + b_i \geq 0 \quad (i = 1,2,\ldots,m)$$

The set of points which satisfy one of these m inequalities is called a half-space and the set of points which satisfy the corresponding equation is called the <u>bounding hyperplane</u> of the half-space. A point satisfies the entire system of inequalities (i.e. is a solution) if and only if it lies at the intersection of the m half-spaces.

The procedure is inductive, producing an infinite sequence of points $x^0, x^1, x^2, \ldots$)which converge to a solution (see Agmon [1]), provided one exists. x^0 is arbitrary. Assuming we have x^k, ($k = 0,1,2,\ldots$ x^{k+1} is obtained as follows:

If x^k is a solution, $x^{k+1} = x^k$

If x^k is not a solution, there are one or more of the given half spaces which do not contain it. Among the bounding hyperplanes of these half-spaces, let α be one at a maximum distance from x^k and let p be the point of α nearest x^k. Then

$$x^{k+1} = x^k + t(p-x^k) \text{ where } 0 < t < 2.$$

Three values of t were tried, namely $t = 3/4$ (undershoot), $t = 1$ (normal-here $x^{k+1} = p$), and $t = 3/2$ (overshoot).

We now describe the process algebraically along with a summary of the machine procedure. The code does not use the algebraic formulation which would yield the fastest computational procedure (the reader can easily concoct such a procedure using the matrix AA^T), for the naive method followed required less internal storage.

Let the set of inequalities be normalized, $\sum_{j=1}^{n} a_{ij}^2 = 1$, $(i = 1,2,\ldots,m)$. Let $y_i = \sum_{j=1}^{n} a_{ij} x_j + b_i$, choose an initial set of values for x: $x_1^{(0)},\ldots,x_n^{(0)}$ and obtain a corresponding set of y: $y_1^{(0)},\ldots,y_n^{(0)}$. If all the y_i are nonnegative, the $x_j^{(0)}$ form a solution. If not choose the largest (in absolute value) of the negative $y_i^{(0)}$, call this $y_k^{(0)}$, and form new values x_j according to $x_j^{(1)} = x_j^{(0)} - t \cdot a_{kj}^{(0)}$ where $0 < t < 2$. Substitute the $x_j^{(1)}$ into the system and obtain a new system of y: $y_i^{(1)}$. Continue in this way to form a sequence $x_1^{(i)}$, $x_2^{(i)},\ldots,x_n^{(i)}$ for $i = 1,2,\ldots$. The machine procedure has been as follows: Given the m x n matrix $A = (a_{ij})$, the problem is scaled so that $|\max a_{ij}| \leqq 1$. Each inequality is multiplied by 10^{-4} so that the b_i are properly scaled. This scaling is to be kept in mind in interpreting the final results. Next conversion is made from the decimal to the binary system and the system is normalized. The matrix A and the b_i's are stored in the machine. The initial choice is $x_i^{(0)} = 0$ for $i = 1,2,\ldots,n$ in all the problem work.

In order to measure convergence, it is reasonable to compare the size of the negative $y_i^{(k)}$ relative to k itself. In this case the following procedure was adopted: an $\epsilon > 0$ was chosen and a solution was considered as obtained when the minimum

$y_i \geqq -\epsilon$. The ϵ was made progressively smaller and, on the basis of previous experience was taken successively as 2^{-2}, 2^{-6}, 2^{-10}, 2^{-11}, $2^{-12}, \ldots, 2^{-22}$. The time required to satisfy a given ϵ was also noted in each instance.

The game problem was transformed into a pure inequality problem in the following manner: If the m x n game matrix is $A = (a_{ij})$, the expected payoff for player one, if he engages in a mixed strategy $x_1, x_2, \ldots, x_n$, is given by $M = \min \sum_{i=1}^{n} a_{ij} x_i$, where $\sum x_i = 1$, $x_i \geqq 0$. Since the game is symmetric, the value is zero so that the problem reduces to solving

$$-Ax \geqq 0, \quad x \geqq 0 \quad \text{and} \quad \sum_{i=1}^{m} x_i = 1,$$

or to solving the (2m+2)x m system of inequalities

$$-Ax \geqq 0, \quad x \geqq 0, \quad \sum_{i=1}^{m} x_i \geqq 1, \quad -\sum_{i=1}^{m} x_i \geqq -1.$$

3. <u>Numerical Results</u>

(a) <u>Simplex Method</u>.

Table I gives the answers obtained, the number of iterations required and the time consumed on the machine by the computation.

	5x5	6x6	7x7	8x8	9x9	10x10
x_1	0.00000	0.00000	0.00000	0.00000	0.12341	0.03690
x_2	0.59999	0.00000	0.00000	0.00000	0.00000	0.00000
x_3	0.19999	0.20000	0.19999	0.04761	0.00000	0.08487
x_4	0.19999	0.20000	0.20000	0.26190	0.25949	0.22509
x_5	0.00000	0.00000	0.00000	0.00000	0.06012	0.10332
x_6		0.60000	0.59999	0.57142	0.02531	0.12915
x_7			0.00000	0.09523	0.03481	0.00000
x_8				0.02380	0.06645	0.00000
x_9					0.43037	0.39483
x_{10}						0.02582
No. of iterations	6	4	5	6	7	11
Time (mins)	10	8	8½	9	12	15

Note that the number of iterations is about n for each of these n x 2n linear programming problems. This is in accord with our general experience using the simplex method on m x n problems that a solution takes approximately m iterations unless the artificial device mentioned in the description of the simplex method given in 2(a) is needed. In that case it takes about 2n iterations to reach a solution. These estimates are completely heuristic, but they are based on over fifty simplex computations of various sizes and are probably the right order of magnitude. The success that the simplex method has enjoyed is based largely on the fact that the number of iterations required has not been larger.

(b) The Fictitious Play Method.

For each of the problems many answers were printed as different ϵ 's were attained.

Let us look in detail at the results of the 6x6 game, which illustrate the typical properties of the convergence rates. In table II are given the approximate solutions $x_i(N)$ corresponding to the various values of ϵ. The step in going from V(N) to V(N+S(N)) is called an S-step. The time is counted from the beginning of the computation, excluding the time taken to print out the answers.

TABLE II

	$\epsilon = 2^{-2}$	$\epsilon = 2^{-6}$	$\epsilon = 2^{-10}$
x_1	0.0169491525	0.0002732987	0.0000013102
x_2	0	0	0
x_3	0.1016949152	0.1866630226	0.1990320137
x_4	0.1864406779	0.1997813610	0.1999989518
x_5	0	0	0
x_6	0.6949151542	0.6132823175	0.6009677241
TIME	0:02	0:12	2:04
S-steps	7	52	742
N	59	3659	763,234

	$\epsilon = 2^{-11}$	$\epsilon = 2^{-12}$
x_1	0.0000003290	0.0000000824
x_2	0	0
x_3	0.1995141064	0.1997565760
x_4	0.1999997367	0.1999999340
x_5	0	0
x_6	0.6004858277	0.6002434074
TIME	3:50	6:40
S-steps	1480	2956
N	3,039,349	12,130,279

Observe that for ϵ sufficiently small the number of S-steps required to "attain the ϵ " doubled as ϵ was halved, while N quadrupled. This phenomenan held for all the games solved as part of the experiment and for others not part of the experiment that were solved by the Brown method.

No arithmetic relationship between the computing time (required to attain a given ϵ) and the size of the matrix could be determined.

(c) The Relaxation Method.

For the same reasons as given above for the fictitious play method, we present below (table III) the results of the 6x6 game obtained using the three methods of relaxation. (N = number of iterations.)

TABLE III

$\epsilon = 2^{-2}$		Undershoot	Normal	Overshoot
x_1	=	0.125000	0.015152	-0.034309
x_2	=	0.125000	0.015152	0.217439
x_3	=	0.125000	0.242424	0.225306
x_4	=	0.125000	0.090909	0.262019
x_5	=	0.125000	0.090909	0.136145
x_6	=	0.125000	0.166667	0.558348
N	=	2	3	5
T	=	0:02	0:03	0:05

$\epsilon = 2^{-6}$		Undershoot	Normal	Overshoot
x_1	=	-0.014057	0.000000	-0.014191
x_2	=	0.054008	0.043413	-0.004277
x_3	=	0.217143	0.210530	0.203763
x_4	=	0.174677	0.177211	0.203559
x_5	=	0.010341	-0.002738	0.029268
x_6	=	0.545075	0.546727	0.585016
N	=	50	33	12
T	=	0:56	0:37	0:13

$\epsilon = 2^{-10}$		Undershoot	Normal	Overshoot
x_1	=	-0.000697	-0.000923	-0.000269
x_2	=	0.003456	0.002975	-0.000320
x_3	=	0.201578	0.201360	0.198920
x_4	=	0.198840	0.198293	0.200325
x_5	=	-0.000877	0.000000	-0.000288
x_6	=	0.596372	0.596994	0.599880
N	=	198	121	37
T	=	3:42	2:15	0:41

$\epsilon = 2^{-11}$	Undershoot	Normal	Overshoot
x_1 =	-0.000372	-0.000251	-0.000412
x_2 =	0.001661	0.001701	-0.000462
x_3 =	0.200672	0.200691	0.199648
x_4 =	0.199325	0.199442	0.200473
x_5 =	-0.000408	-0.000238	-0.000140
x_6 =	0.598102	0.598654	0.600318
N =	236	144	39
T =	0:04	2:41	0:44

$\epsilon = 2^{-12}$	Undershoot	Normal	Overshoot
x_1 =	-0.000179	-0.000216	-0.000219
x_2 =	0.000920	0.000866	-0.000194
x_3 =	0.200323	0.200202	0.199861
x_4 =	0.199665	0.199755	0.200260
x_5 =	-0.000067	-0.000172	0.000176
x_6 =	0.598970	0.599354	0.600318
N =	271	168	43
T =	5:03	3:08	0:48

$\epsilon = 2^{-13}$	Undershoot	Normal	Overshoot
x_1 =	-0.000099	-0.000093	0.000110
x_2 =	0.000436	0.000399	-0.000034
x_3 =	0.200179	0.200086	0.199827
x_4 =	0.199822	0.199877	0.200029
x_5 =	-0.000104	-0.000070	-0.000055
x_6 =	0.599505	0.599696	0.600185
N =	310	193	48
T =	5:47	3:36	0:54

$\epsilon = 2^{-14}$	Undershoot	Normal	Overshoot
x_1 =	-0.000023	0.000000	-0.000041
x_2 =	0.000226	0.000177	0.000036
x_3 =	0.200085	0.200040	0.199914
x_4 =	0.199904	0.199916	0.200052
x_5 =	-0.000033	-0.000041	-0.000035
x_6 =	0.599760	0.599772	0.600149
N =	349	212	55
T =	6:31	3:57	1:02

$\epsilon = 2^{-15}$	Undershoot	Normal	Overshoot
x_1 =	−0.000026	−0.000017	−0.000027
x_2 =	0.000116	0.000110	0.000017
x_3 =	0.200047	0.200055	0.199932
x_4 =	0.199953	0.199957	0.200003
x_5 =	−0.000028	−0.000017	0.000021
x_6 =	0.599867	0.599912	0.600099
N =	383	233	66
T =	7:09	4:21	1:14

$\epsilon = 2^{-16}$	Undershoot	Normal	Overshoot
x_1 =	−0.000004	−0.000015	−0.000012
x_2 =	0.000056	0.000058	0.000003
x_3 =	0.200025	0.200011	0.199957
x_4 =	0.199974	0.199981	0.200028
x_5 =	−0.000014	−0.000007	−0.000015
x_6 =	0.599946	0.599953	0.600060
N =	427	256	78
T =	7:58	4:47	1:27

$\epsilon = 2^{-17}$	Undershoot	Normal	Overshoot
x_1 =	−0.000007	−0.000002	−0.000003
x_2 =	0.000030	0.000025	−0.000008
x_3 =	0.200013	0.200005	0.199984
x_4 =	0.199988	0.199989	0.200010
x_5 =	−0.000007	−0.000006	−0.000003
x_6 =	0.599965	0.599980	0.600023
N =	457	283	105
T =	8:31	5:17	1:58

$\epsilon = 2^{-18}$	Undershoot	Normal	Overshoot
x_1 =	−0.000003	−0.000002	−0.000001
x_2 =	0.000016	0.000013	0.000002
x_3 =	0.200006	0.200004	0.199993
x_4 =	0.199994	0.199995	0.200005
x_5 =	−0.000003	−0.000001	−0.000003
x_6 =	0.599986	0.599990	0.600013
N =	499	305	125
T =	9:19	5:42	2:20

$\epsilon = 2^{-19}$	Undershoot	Normal	Overshoot
x_1 =	-0.000001	-0.000002	-0.000001
x_2 =	0.000007	0.000006	0.000000
x_3 =	0.200003	0.200003	0.199996
x_4 =	0.199998	0.199998	0.200003
x_5 =	-0.000002	-0.000002	-0.000001
x_6 =	0.599993	0.599993	0.600005
N =	541	322	147
T =	10:05	6:01	2:45

$\epsilon = 2^{-20}$	Undershoot	Normal	Overshoot
x_1 =	-0.000001	0.000000	-0.000001
x_2 =	0.000003	0.000003	0.000000
x_3 =	0.200001	0.200001	0.199999
x_4 =	0.199999	0.199999	0.200000
x_5 =	-0.000001	-0.000001	0.000001
x_6 =	0.599996	0.599998	0.600004
N =	579	355	163
T =	10:48	6:38	3:03

$\epsilon = 2^{-21}$	Undershoot	Normal	Overshoot
x_1 =	0.000000	0.000000	0.000000
x_2 =	0.000002	0.000002	0.000000
x_3 =	0.200001	0.200001	0.199999
x_4 =	0.199999	0.200000	0.200001
x_5 =	0.000000	0.000000	0.000000
x_6 =	0.599998	0.599998	0.600001
N =	614	368	184
T =	11:28	6:52	3:26

$\epsilon = 2^{-22}$	Undershoot	Normal	Overshoot
x_1 =	0.000000	0.000000	0.000000
x_2 =	0.000001	0.000001	0.000000
x_3 =	0.200000	0.200000	0.200000
x_4 =	0.200000	0.200000	0.200000
x_5 =	0.000000	0.000000	0.000000
x_6 =	0.599999	0.599999	0.600001
N =	653	388	201
T =	12:11	7:15	3:45

Observe that overshoot converged faster than normal, which in turn converged faster than undershoot. This held consistently for all the games.

Further, for ϵ sufficiently small, there is an approximately uniform increase in the number of iterations required to "attain a given ϵ" as ϵ is halved. For the 6x6 game, for example, from $\epsilon = 2^{-6}$ to $\epsilon = 2^{-20}$, the additional iterations required to go from $\epsilon = 2^{-i}$ to $\epsilon = 2^{-(i+1)}$ were approximately 38 (for undershoot), 24 (for normal), 12 (for overshoot).

The experiment did not reveal any arithmetic relationship between the size of the matrix and the computing time.

Conclusions

Any relative evaluation of proposed computation schemes requires specification of the size of the problem considered, the accuracy demanded and the amount of computation time reasonable to invest in obtaining this accuracy. Let us assume (in accordance with the requirements of most of the practical problems that have so far arisen in our work) that four or five decimal digits are required in the answer, and that the size of the matrix A is say, 7x7 or greater.

Then the simplex method is outstanding among the three. In the large size games considered in the experiment, the simplex method achieved answers to this precision in a third or a fourth of the time required by the most favorable of the others. This occurred despite the fact that simplex was to use magnetic tapes for storage of most of the numbers arising in the computation, whereas the other methods stored all the numbers within the high speed memory. It is estimated that about 4/5 of the machine time required for simplex on these games was spent in bringing the needed numbers from the tape to the high speed memory and taking the numbers from the memory to the tape. (Improvements in tape performance subsequent to these computations have reduced this ratio to about $\frac{1}{2}$). The other methods would be completely impractical if tape had to be used, and that is why only the simplex method has solved moderately large problems (where the matrix A is about 50x70). Even assuming a very large memory so that, for instance, a large problem could be coded for relaxation in the most efficient way, then the fact that simplex could be done internally would favor it even more. It is true, however, that because simplex is more complicated algebraically, it is possible by clever coding to fit some problems into the high speed memory when using fictitious play or relaxation that could not be so accommodated if simplex were employed. One such large problem arose in our work: the computation matrix was 48x71 for the simplex method, but formulated as a symmetric game and using ingenious coding devices, it could be done within the high speed memory by fictitious play. Nevertheless, simplex was completed in half the time that fictitious play required to obtain the same accuracy.

Is there then an area of usefulness for the infinite methods? The answer is yes, for problems satisfying the following conditions: They are small enough to be done entirely within the memory, and the precision demanded is very slight or very great. Two objections to the simplex method are: (i) in general, there is no reason to believe that an answer from an early iteration has any meaning at all, so there is no provision for doing less work if one is content with small accuracy: and (ii) when the answers are finally obtained, there is no way to improve them to obtain greater precision. (Wolfe's proposed variation [12] of the simplex method will help (ii), but it is questionable that it would involve less work than the procedure suggested below).

If the purposes of the computation require only one or two decimals in the answer, then one is perhaps better off using the infinite methods. This is verified in the 6x6 problem, which, indeed, favors fictitious play over relaxation for this purpose (which favorable position held in general). If the purposes of the computation demand greater precision than the simplex answers yield, then it is reasonable to use the simplex answers as a starting point for one of the other methods. And here, the more favorable convergence rate for relaxation (see the comments in 3(b) and 3(c) over fictitious play favors the use of the former.

References

Several of the references (indicated with *) are taken from Activity Analysis of Production and Allocation, edited by T. C. Koopmans, New York, 1951.

[1] S. Agmon, The Relaxation Method for Linear Inequalities, to appear in Annals of Mathematics.

*[2] G. W. Brown, Iterative Solution of Games by Fictitious Play, p. 379.

[3] A. Charnes, Optimality and Degeneracy in Linear Programming, Econometrica, vol. 20, No. 2(1952), p. 160.

*[4] G. B. Dantzig, A Proof of the Equivalence of the Programming Problem and the Game Problem, p. 330.

*[5] G. B. Dantzig, Maximization of a Linear Function of Variables Subject to Linear Inequalities, p. 339.

*[6] D. Gale, H. W. Kuhn, A. W. Tucker, Linear Programming and the Theory of Games, p. 317.

[7] A. J. Hoffman, On Approximate Solutions of Systems of Linear Inequalities, Journal of Research of the National Bureau of Standards, vol. 49, No. 4 (1952), p. 263.

[8] J. C. C. McKinsey, Introduction to the Theory of Games, New York, 1952.

[9] A. Orden, Application of the Simplex Method to a Variety of Matrix Problems, in Symposium on Linear Inequalities and Programming, edited by A. Orden and L. Goldstein, Washington, 1952, p. 28.

[10] A. Orden, Solution of Systems of Linear Inequalities on a Digital Computer, Proceedings of the Association for Computing Machinery, May 1952, Pittsburgh, Pa.

[11] J. Robinson, An Iterative Method of Solving a Game, Annals of Mathematics, vol. 54(1951), p. 296.

[12] P. Wolfe, The Epsilon-Technique and the Artificial Basis in the Simplex Solution of the Linear Programming Problem, Planning Research Division, Office of Air Comptroller, United States Air Force, 1951 (mimeographed).

THE NATIONAL BUREAU OF STANDARDS

Functions and Activities

The functions of the National Bureau of Standards are set forth in the Act of Congress, March 3, 1901, as amended by Congress in Public Law 619, 1950. These include the development and maintenance of the national standards of measurement and the provision of means and methods for making measurements consistent with these standards; the determination of physical constants and properties of materials; the development of methods and instruments for testing materials, devices, and structures; advisory services to Government Agencies on scientific and technical problems; invention and development of devices to serve special needs of the Government; and the development of standard practices, codes, and specifications. The work includes basic and applied research, development, engineering, instrumentation, testing, evaluation, calibration services, and various consultation and information services. A major portion of the Bureau's work is performed for other Government Agencies, particularly the Department of Defense and the Atomic Energy Commission. The scope of activities is suggested by the listing of divisions and sections on the inside of the front cover.

Reports and Publications

The results of the Bureau's work take the form of either actual equipment and devices or published papers and reports. Reports are issued to the sponsoring agency of a particular project or program. Published papers appear either in the Bureau's own series of publications or in the journals of professional and scientific societies. The Bureau itself publishes three monthly periodicals, available from the Government Printing Office: The Journal of Research, which presents complete papers reporting technical investigations; the Technical News Bulletin, which presents summary and preliminary reports on work in progress; and Basic Radio Propagation Predictions, which provides data for determining the best frequencies to use for radio communications throughout the world. There are also five series of nonperiodical publications: The Applied Mathematics Series, Circulars, Handbooks, Building Materials and Structures Reports, and Miscellaneous Publications.

Information on the Bureau's publications can be found in NBS Circular 460, Publications of the National Bureau of Standards ($1.00). Information on calibration services and fees can be found in NBS Circular 483, Testing by the National Bureau of Standards (25 cents). Both are available from the Government Printing Office. Inquiries regarding the Bureau's reports and publications should be addressed to the Office of Scientific Publications, National Bureau of Standards, Washington 25, D. C.

NBS