V. A. MĂRIEŞ

Method of Extending a Markov Chain for Data Processing on Electronic Computation Systems

Modern mathematical instruments allow us to more and more solve from the secrets of social and economic mechanism, to establish its developing tendencies in the future, and to establish as such a strategy of the development, as to make possible the touching of the wished objects.

One of the economic problems, which appear in the activity of people dealing with national-economy development planning or with the planning of one of its fields, is the prognosis problem, by trend-studying the development of the respective phenomenon or process. This problem can be studied with the help of the Markov chains and of quadratic programming.

Let T be a set of observations on the production of n assortments in the studied field, the probabilities p_i^t of state i at moment t, for $i = 1, 2, ..., n$ and $t = 1, 2, ..., T$, being known.

Let us determine the economic-phenomenon structure at the moment $t = T + 1$, i.e. the unknown probabilities $x_i^t (i = 1, 2, ..., n)$ for the production of the n assortments. The weights of the production of the n assortments, at the moment t, may be considered as probabilities $p_i^t (i = 1, 2, ..., n; t = 1, 2, ..., T)$. We can point out $T + 1$ n — dimension column-vectors:

$$p^t = \begin{bmatrix} p_1^t \\ p_2^t \\ \cdot \\ \cdot \\ \cdot \\ p_n^t \end{bmatrix} \text{ and } x^{T+1} = \begin{bmatrix} x_1^{T+1} \\ x_2^{T+1} \\ \cdot \\ \cdot \\ \cdot \\ x_n^{T+1} \end{bmatrix} \tag{1}$$

for $t = 1, 2, ..., T$;

with properties:

$$\begin{cases} e' \cdot p^t = 1; \\ t = 1, 2, \ldots, T; \end{cases} \qquad \text{and } e' \cdot x^{T+1} = 1; \tag{2}$$

where e is the n-dimension column-vector having all elements equal to the unity.

* * *

A method used so far for determining the unknown vector x^{T+1} is, broadly speaking, the following one:

There is, generally, a matrix

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ x_{n1} & x_{n2} & \cdots & x_{nn} \end{bmatrix}$$

called stochastic transition-function matrix, having the property

$$\begin{cases} p^{t+1} = \mathbf{X}' \cdot p^t; \\ t = 1, 2, \ldots, T-1; \end{cases} \tag{3}$$

The ranks of matrix $\mathbf{X}$ form the n — dimension vectors $x_k = (x_{k1}, x_{k2}, \ldots, x_{kn})$ which must comply with restrictions:

$$\begin{cases} x_k \cdot e = 1; \\ k = 1, 2, \ldots; n, \end{cases} \qquad \text{hence } \mathbf{X} \cdot e = e; \tag{4}$$

and

$$\begin{cases} 0 \leqslant x_{ij} \leqslant 1; \\ i = 1, 2, \ldots, n; \\ j = 1, 2, \ldots, n; \end{cases} \tag{5}$$

as the elements x_{ij} represent the stochastic transition functions, from state i to state j.

If we know transition matrix $\mathbf{X}$, we can determine the unknown vector of the weights of the n assortments, at the moment $T+1$, x^{T+1}, on the basis of relation:

$$x^{T+1} = \mathbf{X}' \cdot p^t; \tag{6}$$

whose components satisfy the Chapmann-Kolmogorov relation

$$\begin{cases} x_i^{T+1} = \sum_{j=1}^{n} p_j^T \cdot x_{ji}; \\ i = 1, 2, \ldots, n. \end{cases} \tag{7}$$

In general, the stochastic transition-function matrix $\mathbf{X}$ cannot be determined from the beginning such as to have property (3) and simultaneously to comply with restrictions (4) and (5). For this reason, a matrix $\mathbf{X}$ is sought for, which deviates as little as possible from property (3), but integrally com-

plies with restrictions (4) and (5). We consider the deviation squares sum as a measure of the deviation:

$$A(\mathbf{X}) = \sum_{t=1}^{T-1} (p^{t+1} - \mathbf{X}' \cdot p^t)^2 = \sum_{t=1}^{T-1} (p^{t+1} - \mathbf{X}' \cdot p^t) \cdot \mathbf{C} \cdot (p^{t+1} - \mathbf{X}' \cdot p^t) =$$

$$= \sum_{t=1}^{T-1} p^{t+1} \cdot \mathbf{C} \cdot p^{t+1} - 2 \sum_{t=1}^{T-1} (p^t)' \cdot \mathbf{X} \cdot \mathbf{C} \cdot p^{t+1} + \sum_{t=1}^{T-1} (p^t)' \cdot \mathbf{X} \cdot \mathbf{C} \cdot \mathbf{X}' \cdot p^t; \qquad (8)$$

where $\mathbf{C}$ is a symmetrical matrix, positively semi-defined, and

$$\mathbf{C}^{-1} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \cdot & \cdot & \cdots & \\ \cdot & \cdot & \cdots & \\ c_{n1} & c_{n2} & \cdots & c_{nn} \end{bmatrix}$$

The quadratic form $A(\mathbf{X})$ is minimal when the following quadratic form is minimal:

$$f(\mathbf{X}) = \frac{1}{2} \sum_{t=1}^{T-1} (p^t)' \cdot \mathbf{X} \cdot \mathbf{C} \cdot \mathbf{X}' \cdot p^t - \sum_{t=1}^{T-1} (p^t)' \cdot \mathbf{X} \cdot C \cdot p^{t+1}; \qquad (9)$$

Under these conditions the quadratic programming problem is enounced as follows: Determine the stochastic transition-function matrix $\mathbf{X}$ with property (3), which satisfies restrictions (4) and (5), such as to minimize quadratic forms (8) and (9), respectively.

In the first stage, matrix $\mathbf{X}_0$, which minimizes quadratic form (9), is determined, by taking into consideration only restrictions (4), and neglecting restrictions (5). With that in view, form the Lagrangian operator:

$$\mathbf{L}(\mathbf{X}, \mathbf{U}) = f(\mathbf{X}) + \mathbf{U}' \cdot (\mathbf{X} \cdot e - e)$$

where the multiplying matrix elements

$$\mathbf{U} = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ u_{21} & u_{22} & \cdots & u_{2n} \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ u_{n1} & u_{n2} & \cdots & u_{nn} \end{bmatrix}$$

are Lagrange multipliers.

For determining matrix $\mathbf{X}_0$, solve the homogeneous system, of the partial derivatives, with respect to $\mathbf{U}$ and $\mathbf{X}$:

$$\begin{cases} \mathbf{X} \cdot e = e; \\ \sum_{t=1}^{T-1} (p^t \cdot (p^t)' \cdot \mathbf{X} \cdot \mathbf{C} \cdot p^t (p^{t+1})' \cdot \mathbf{C}) + \mathbf{U} \cdot e' = 0; \end{cases} \qquad (10)$$

obtained on the basis of relations:

$$\begin{cases} \dfrac{\partial (p^t)' \cdot \mathbf{X} \cdot \mathbf{C} \cdot p^{t+1})}{\partial \mathbf{X}} = p^t \cdot (p^{t+1})' \cdot \mathbf{C}; \\ \dfrac{\partial (p^t)' \cdot \mathbf{X} \cdot \mathbf{C} \cdot \mathbf{X}' \cdot p^t)}{\partial \mathbf{X}} = 2p^t \cdot (p^t)' \cdot \mathbf{X} \cdot \mathbf{C}; \end{cases}$$

We obtain, by solving system (10):

$$\begin{cases} \mathbf{U} = 0; \\ \mathbf{X}_0 = \mathbf{S}^{-1} \cdot \mathbf{R}; \end{cases} \tag{11}$$

where we denoted:

$$S = \sum_{t=1}^{T-1} p^t (p^t)';$$

$$\mathbf{R} = \sum_{t=1}^{T-1} p^t (p^{t+1})';$$

$$S^{-1} = \begin{bmatrix} s_{11} & s_{12} & \cdots & s_{1n} \\ s_{21} & s_{22} & \cdots & s_{2n} \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ s_{n1} & s_{n2} & \cdots & s_{nn} \end{bmatrix}$$

As $\mathbf{X}_0$ does not depend on $\mathbf{C}$, this means that it is a solution in the first stage for any matrix $\mathbf{C}$, positively semi-defined. If the elements of matrix $\mathbf{X}_0$ satisfy also restrictions (5), it is the solution of the problem and the vector x^{T+1} can be determined on the basis of relation (6) or (7), respectively.

Otherwise, we go further to the second stage, by applying the Theil-Van de Panne algorithm. This combinatory algorithm takes into consideration restrictions (5), which it imposes under the form of equalities of a certain combination of matrix $\mathbf{X}_0$ index-pairs, which do not comply with restrictions (5). They are imposed as follows:

$$\begin{cases} x_{hk} \to r_{hk} = \begin{cases} 0, \text{ if } x_{hk} < 0; \\ 1, \text{ if } x_{hk} > 1; \end{cases} \\ (h,\ k) \in I \end{cases} \tag{12}$$

i.e.

$$\begin{cases} x_{hk} \to r_{hk} = e_h' \cdot \mathbf{X} \cdot e_k; \\ (h,\ k) \in I \end{cases}$$

where e_k is the k-th n-dimension unity column-vector.

In this case, the lagrangian is:

$$\mathbf{L}(\mathbf{X}, \mathbf{U}) = f(\mathbf{X}) + \mathbf{U}' \cdot (\mathbf{X} \cdot e - e) + \sum_{(h,\ k) \in \mathbf{I}} u_{hk} (e_h' \cdot \mathbf{X} \cdot e_k - r_{hk}) =$$

$$= \frac{1}{2} \sum_{t=1}^{T-1} (p^t)' \cdot \mathbf{X} \cdot \mathbf{C} \cdot \mathbf{X}' p^t - \sum_{t=1}^{T-1} (p^t)' \cdot \mathbf{X} \cdot \mathbf{C} \cdot p^{t+1} + \sum_{(h,\ k) \in \mathbf{I}} u_{hk} \cdot (e_h' \cdot \mathbf{X} \cdot e_k -$$

$$- r_{hk}) + \mathbf{U}' \cdot (\mathbf{X} \cdot e - e).$$

obtained on the basis of relations:

$$\begin{cases} \dfrac{\partial (p^t)' \cdot \mathbf{X} \cdot \mathbf{C} \cdot p^{t+1})}{\partial \mathbf{X}} = p^t \cdot (p^{t+1})' \cdot \mathbf{C}; \\ \dfrac{\partial (p^t)' \cdot \mathbf{X} \cdot \mathbf{C} \cdot \mathbf{X}' \cdot p^t)}{\partial \mathbf{X}} = 2p^t \cdot (p^t)' \cdot \mathbf{X} \cdot \mathbf{C}; \end{cases}$$

We obtain, by solving system (10):

$$\begin{cases} \mathbf{U} = 0; \\ \mathbf{X}_0 = \mathbf{S}^{-1} \cdot \mathbf{R}; \end{cases} \tag{11}$$

where we denoted:

$$S = \sum_{t=1}^{T-1} p^t (p^t)';$$

$$\mathbf{R} = \sum_{t=1}^{T-1} p^t (p^{t+1})';$$

$$S^{-1} = \begin{bmatrix} s_{11} & s_{12} & \cdots & s_{1n} \\ s_{21} & s_{22} & \cdots & s_{2n} \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ s_{n1} & s_{n2} & \cdots & s_{nn} \end{bmatrix}$$

As $\mathbf{X}_0$ does not depend on $\mathbf{C}$, this means that it is a solution in the first stage for any matrix $\mathbf{C}$, positively semi-defined. If the elements of matrix $\mathbf{X}_0$ satisfy also restrictions (5), it is the solution of the problem and the vector x^{T+1} can be determined on the basis of relation (6) or (7), respectively.

Otherwise, we go further to the second stage, by applying the Theil-Van de Panne algorithm. This combinatory algorithm takes into consideration restrictions (5), which it imposes under the form of equalities of a certain combination of matrix $\mathbf{X}_0$ index-pairs, which do not comply with restrictions (5). They are imposed as follows:

$$\begin{cases} x_{hk} \to r_{hk} = \begin{cases} 0, \text{ if } x_{hk} < 0; \\ 1, \text{ if } x_{hk} > 1; \end{cases} \\ (h,\ k) \in I \end{cases} \tag{12}$$

i.e.

$$\begin{cases} x_{hk} \to r_{hk} = e_h' \cdot \mathbf{X} \cdot e_k; \\ (h,\ k) \in I \end{cases}$$

where e_k is the k-th n-dimension unity column-vector.

In this case, the lagrangian is:

$$\mathbf{L}(\mathbf{X}, \mathbf{U}) = f(\mathbf{X}) + \mathbf{U}' \cdot (\mathbf{X} \cdot e - e) + \sum_{(h,\ k) \in \mathbf{I}} u_{hk} (e_h' \cdot \mathbf{X} \cdot e_k - r_{hk}) =$$

$$= \frac{1}{2} \sum_{t=1}^{T-1} (p^t)' \cdot \mathbf{X} \cdot \mathbf{C} \cdot \mathbf{X}' p^t - \sum_{t=1}^{T-1} (p^t)' \cdot \mathbf{X} \cdot \mathbf{C} \cdot p^{t+1} + \sum_{(h,\ k) \in \mathbf{I}} u_{hk} \cdot (e_h' \cdot \mathbf{X} \cdot e_k -$$

$$- r_{hk}) + \mathbf{U}' \cdot (\mathbf{X} \cdot e - e).$$

By partially differentiating with respect to X and putting equal to zero, we have:

$$\sum_{t=1}^{T-1} p^t (p^t)' \cdot \mathbf{X} \cdot \mathbf{C} - \sum_{t=1}^{T-1} p^t \cdot (p^{t+1})' \cdot \mathbf{C} + \sum_{(h,k)\in \mathbf{I}} u_{hk} \cdot e_h \cdot e_k' + \mathbf{U} \cdot e' = 0;$$

hence the solution of this iteration is:

$$\begin{cases} \mathbf{U} = -\dfrac{1}{e' \cdot \mathbf{C}^{-1} \cdot e} \sum\limits_{(h,k)\in \mathbf{I}} u_{hk} \cdot e_h \cdot e_k' \cdot \mathbf{C}^{-1} \cdot e; \\ \mathbf{X} = \mathbf{X}_0 - \sum\limits_{(h,k)\in I} u_{hk} \cdot S^{-1} \cdot c_h \cdot e_k' \cdot F; \end{cases} \tag{13}$$

where we denoted

$$F = \mathbf{C}^{-1} - \frac{\mathbf{C}^{-1} \cdot e \cdot e' \cdot \mathbf{C}^{-1}}{e' \cdot \mathbf{C}^{-1} \cdot e} = \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1n} \\ f_{21} & f_{22} & \cdots & f_{2n} \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ f_{n1} & f_{n2} & \cdots & f_{nn} \end{bmatrix}$$

and its elements can be calculated more simply by relations:

$$\begin{cases} f_{ij} = c_{ij} - \dfrac{\left(\sum\limits_{s=1}^{n} c_{is}\right) \cdot \left(\sum\limits_{j=1}^{n} c_{rj}\right)}{\sum\limits_{s=1}^{n} \sum\limits_{r=1}^{n} c_{rs}}; \\ i = 1, 2, \ldots, n; \\ j = 1, 2, \ldots, n; \end{cases} \tag{14}$$

In order to determine solution X of the relevant iteration, we must initially determine the Lagrange multipliers u_{hk}, $(h, k) \in I$, in the linear and non-homogeneous system:

$$\begin{cases} \sum\limits_{(h,k)\in \mathbf{I}} s_{ih} \cdot f_{kj} \cdot u_{hk} = x_{ij} - r_{ij}. \\ (i, j) \in I; \end{cases} \tag{15}$$

In this way, after at the most

$$N = \sum_{k=1}^{n} \mathrm{C}_n^k = 2^n - 1 \tag{16}$$

iterations, we obtain the transition matrix X, which has property (3) and observes all restrictions (4) and (5); we can obtain, with its help, the finally sought vector x^{T+1} on the basis of relation (6) or (7), respectively.

From the above stated, and from the conclusions drawn on practically using this method, we may deduce that it has the following shortcomings:

1. Since the Theil — Van de Panne algorithm is a combinatory procedure, it is very difficult to program, when using electronic data-processing systems, especially if $n > 2$.

2. The maximal number of necessary iterations, in the second stage, grows exponentially with the problem dimensions, so that high values are quickly attained and the combination programming is very difficult to carry out.

3. While solving the problem, the necessity appears to reverse an order — n matrix, a fact which implies a series of calculation and programming difficulties.

4. It is necessary to operate with generally big n-order matrixes,which need big memories, disregarding the fact that the program also needs a share of the internal memory of the electronic data-processing systems.

5. It is necessary, in order to obtain the unknown vector sought for, to calculate all components of the occurring vectors and matrix elements, as the determination of only part of them is not sufficient.

6. It is absolutely necessary to define a matrix $\mathbf{C}^{-1}$ and to compute matrix F, in order to obtain Lagrange's multipliers.

* * *

We shall now expound a simplified method of determining vector x^{T+1} with the help of electronic data-processing systems.

One may notice that, for obtaining the unknown vector x^{T+1}, only the known vector p^T and the stochastic transition-function matrix $\mathbf{X}$, found by the described method, are necessary. In this calculation, matrix $\mathbf{X}$ appears as an intermediary result, whose determination implies the previously stated series of difficulties.

The question arises: would it be possible to obtain the unknown vector x^{T+1} without effecting the calculation of matrix $\mathbf{X}$? The answer is affirmative, as one may notice that the above stated difficulties are the smaller, as n is the smaller. But when are these difficulties minimal? Of course, when $n = 2$.

Hence, if $n > 2$, the method's originality consists in decomposing and reducing the problem to n particular subproblems, with $n = 2$, such as to successively obtain the components of vector x^{T+1}. This simplification will have particularly favourable consequences in determining vector x^{T+1}. We shall further present the way of determining one of vector's x^{T+1} components, for the remaining ones the procedure being analogous.

We suppose that the first $i - 1$ components of vector x^{T+1} were determined and that we go further, to determine the next component x_i^{T+1}.

The set of $T + 1$ vectors, pointed out in (1), is considered to be:

$$\begin{cases} p^t = \begin{Bmatrix} \mathbf{P}_1^t \\ \mathbf{P}_2^t \end{Bmatrix}; \\ t = 1,\ 2,\ \ldots,\ T; \end{cases} \quad \text{and } x^{T+1,\,i} = \begin{Bmatrix} x_1^{T+1,\,i} \\ x_2^{T+1,\,i} \end{Bmatrix};$$

where

$$\begin{cases} \mathbf{P}_1^t = p_i^t; \\ \mathbf{P}_2^t = \sum\limits_{\substack{k=1 \\ k \neq i}}^{n} p_k^t; \\ t = 1, 2, \ldots, T. \end{cases} \tag{17}$$

On the basis of relations (2), and taking into account that solution x^{T+1}, regardless of the way of obtaining it, must be unique, the following properties result:

$$\begin{cases} \mathbf{P}_2^t = \sum\limits_{\substack{k=1 \\ k \neq 1}}^{n} p_k^t = \sum\limits_{k=1}^{n} p_k^t - p_i^t = 1 - \mathbf{P}_1^t \\ t = 1, 2, \ldots, T; \end{cases} \tag{18}$$

$$\begin{cases} x_1^{T+1, i} = x_i^{T+1}; \\ x_2^{T+1, i} = \sum\limits_{\substack{k=1 \\ k \neq i}}^{n} x_k^{T+1} = \sum\limits_{k=1}^{n} x_k^{T+1} - x_i^{T+1} = 1 - x_i^{T+1, i} \end{cases} \tag{19}$$

The stochastic transition-function matrix, sought for in this problem, is

$$\mathbf{X}^t = \begin{Vmatrix} x_{11}^t & x_{12}^t \\ x_{21}^t & x_{22}^t \end{Vmatrix},$$

and has property (3), and must satisfy restrictions (4) and (5), hence the relation

$$\begin{cases} x_{12}^t = 1 - x_{11}^t; \\ x_{22}^t = 1 - x_{21}^t, \end{cases} \tag{20}$$

In order to determine the solution $\mathbf{X}_0^t$ of the first stage, following matrixes will be calculated beforehand:

$$\mathbf{R} = \sum_{t=1}^{T-1} p^t (p^{t+1})'; \qquad S = \sum_{t=1}^{T-1} p^t (p^t)';$$

as well as the reverse of matrix S, i.e. S^{-1}.

If the elements of matrix $\mathbf{X}_0^t$, obtained with the help of relation (11), also satisfy restriction (5), it is a solution of the subproblem, hence vector $x^{T+1, i}$ can be determined with the help of relation (6) or (7), respectively, i. e. component $x_i^{T+1} = x_1^{T+1, i}$.

Otherwise, we must go further to the second stage, on applying the Theil-Van de Panne algorithm. With that end in view, we initilly calculate matrixes $\mathbf{C}^{-1}$ and F, with the help of relations (14). Thus, if matrix $\mathbf{C}^{-1}$ is considered a diagonal matrix, having as diagonal elements the means of the assortment weights in all T years, we get:

$$\mathbf{C}^{-1} = \begin{Vmatrix} a & 0 \\ 0 & 1 - 0 \end{Vmatrix}$$

and

$$F = \begin{Bmatrix} a(1-a) & a(a-1) \\ a(a-1) & a(1-a) \end{Bmatrix};$$

The symmetry and the particular form of matrix F are obtained owing to the form of matrix $\mathbf{C}^{-1}$, generated by dimension $n = 2$.

The Lagrage multipliers are obtained from system (15), by imposing restrictions (5) under the form (12).

Supposing that $x_{11}^{i0} < 0$, we obtain $u_{11} = \dfrac{x_{11}^{i0}}{s_{11} \cdot f_{11}}$.

On the basis of relation (13), we obtain the solution if iteration:

$$\mathbf{X}' = \begin{Bmatrix} x_{11}^{i} & x_{12}^{i} \\ x_{21}^{i} & x_{22}^{i} \end{Bmatrix} = \begin{Bmatrix} 0 & 1 \\ x_{21}^{i0} - \dfrac{x_{11}^{i0} \cdot s_{21}}{s_{11}} & x_{22}^{i0} - \dfrac{x_{11}^{i0} \cdot s_{21} \cdot f_{12}}{s_{11} \cdot f_{11}} \end{Bmatrix}$$

If this solution complies with restrictions (15), it is a solution of the subproblem and we can further calculate vector $x^{T+1,\,i}$ with the help of relation (6) or (7), respectively, whose component $x_1^{T+1,\,i}$ does not depend on matrixes $\mathbf{C}^{-1}$ and F, as can be seen:

$$x_1^{T+1,\,i} = x_{11}^{i} \cdot \mathbf{P}_1^T + x_{21}^{i} \cdot \mathbf{P}_2^T = \mathbf{P}_2^T \cdot O + \mathbf{P}_2^T \left(x_{21}^{i0} - \frac{x_{11}^{i0} \cdot s_{21}}{s_{11}} \right) = x_i^{T+1}; \quad (21)$$

Taking into account relations (19), (20) and (21), we notice that neither calculation of matrixes $\mathbf{C}^{-1}$; F, nor of all elements of matrix $\mathbf{X}_0^i$ and of all components of vector $x^{T+1,\,i}$ is necessary.

In case this iteration does not generate the subproblem solution, the solution of subproblem i, i.e. component $x_i^{T+1} = x_1^{T+1,\,i}$ may be obtained, according to relation (16), through maximum $N = 2^2 - 1 = 3$ iterations. The restrictions number is small and the solutions of subproblems may be calculated on an extremely short way, due to the automatic respecting of restrictions (4), as well as to the fact that various components and elements are expressed by linear expressions depending on other ones, as are relations (18), (19), (20).

The other components of vector x^{T+1} are determined by repeating the algorithm. The maximal number of necessary iterations, for obtaining vector x^{T+1}, is:

$$N = n(2^2 - 1) + n - 1 = 4 \cdot n - 1$$

if we also count among the iterations the n — 1 initial solutions $\mathbf{X}_0^i$.

Let us see for which problem's dimensions the new method has a smaller number of iterations. On the basis of the first method, we have $N_1 = 2^n - 1$ iterations, and on the basis of the new one, $N_2 = 4 \cdot n - 1$ iterations. From condition $N_1 \geqslant N_2$ we get $n \geqslant 5$.

From relation:

$$x_i^{T+1} = x_1^{T+1\cdot i} = x_{11}^i \cdot \mathbf{P}_1^T + x_{21}^i \cdot \mathbf{P}_2^T = x_{11}^i \cdot p_i^T + x_{21}^i \sum_{\substack{k=1 \\ k\neq i}}^{n} p_k^T = \sum_{k=1}^{i-1} x_{21}^i \cdot p_k^T +$$

$$+ x_{11}^i \cdot p_i^T + \sum_{k=i+1}^{n} x_{21}^i \cdot p_k^T$$

we notice that a transition matrix **X** can be supplementarily attached, but it does not satisfy either property (3) or restrictions (4), yet it satisfies relation (6) or (7), respectively. This matrix is written down as follows:

$$\mathbf{X} = \begin{Vmatrix} x_{11}^1 & x_{21}^2 & \dots & x_{21}^{i\,1} & x_{21}^1 & x_{21}^{i+1} & \dots & x_{21}^n \\ x_{21}^1 & x_{11}^2 & \dots & x_{21}^{i\,1} & x_{21}^1 & x_{21}^{i+1} & \dots & x_{21}^n \\ \cdot & \cdot & \dots & \cdot & \cdot & \cdot & \dots & \\ \cdot & \cdot & \dots & \cdot & \cdot & \cdot & \dots & \\ \cdot & \cdot & \dots & \cdot & \cdot & \cdot & \dots & \\ x_{21}^1 & x_{21}^2 & \dots & x_{11}^{i-1} & x_{21}^i & x_{21}^{i+1} & \dots & x_{21}^n \\ x_{21}^1 & x_{21}^2 & \dots & x_{21}^{i-1} & x_{11}^i & x_{21}^{i+1} & \dots & x_{21}^n \\ x_{21}^1 & x_{21}^2 & \dots & x_{21}^{i-1} & x_{21}^i & x_{11}^{i+1} & \dots & x_{21}^n \\ \cdot & \cdot & \dots & \cdot & \cdot & \cdot & \dots & \\ \cdot & \cdot & \dots & \cdot & \cdot & \cdot & \dots & \\ \cdot & \cdot & \dots & \cdot & \cdot & \cdot & \dots & \\ x_{21}^1 & x_{21}^2 & \dots & x_{21}^{i-1} & x_{21}^i & x_{21}^{i+1} & \dots & x_{21}^n \\ x_{21}^1 & x_{21}^2 & \dots & x_{21}^{i-1} & x_{21}^i & x_{21}^{i+1} & \dots & x_{21}^n \end{Vmatrix}$$

On the basis of these reasonings, the new method has the following advantages, as compared to the first one:

1. The Theil-Van de Panne algorithm is extremely simple to program, as it is a combinatory procedure applied in the case of the particular dimension $n = 2$.

2. The maximal number of necessary iterations, in the second stage of a subproblem, is constant and equal to 3, and the total number of iterations, necessary with the new method, grows only linearly with the problem's dimension. The total number of iterations, necessary with the new method, is much smaller when $n \geqslant 5$.

3. The calculation of the n-order matrixes is replaced by the calculation of some 2nd-order matrixes, a fact which implies minimal difficulties, both from the viewpoint of programming and of the occupied memory.

4. The program of reversing 2nd-order matrix, as well as the whole program, needs a small internal memory.

5. It is not necessary, in order to obtain the sought-for unknown vector, to know all vector components or all elements of the occurring matrixes, due to the frequent relations in which the unity is the sum of two quantities, of which it is sufficient to know only one.

6. It is not necessary to define matrix $\mathbf{C}^{-1}$ or to calculate matrix F or the Lagrange multipliers, respectively, and, in this way, the second stage of each subproblem gets extremely simplified.

We can consider, as a defficiency of the new method, that we cannot obtain the stochastic transition-function matrix **X**, which would satisfy rela-

tions (3), (4) and (5); as a matter of fact, the enunciated problem has not this object. We obtain supplementarily a matrix **X**, which satisfies only relation (6) or (7), respectively.

* * *

As the method is favourable, especially for $n \geqslant 5$, we give as example a problem with $n = 7$ dimensions, by supposing to know the evolution of the production weights, for 7 product groups of the national economy [4], during four years, according the data in the following table:

i	Product group \ Year / t	1966	1967	1968	1969
		1	2	3	4
1	Industrial electronic equipment	0.1556815	0.1571150	0.1228459	0.1227918
2	Medical electronic equipment	0.0117721	0.0237045	0.0248214	0.0253549
3	Professional electronic measuring and control instruments	0.0622026	0.0653685	0.0625623	0.0625115
4	Nuclear instruments	0.0125011	0.0137052	0.0158491	0.0150654
5	Telecommunication electronic equipment	0.2570435	0.2543253	0.2509409	0.2442375
6	Consumer electronic appliances	0.3216251	0.2917790	0.2982848	0.2855983
7	Electronic computers and nuclear accelerators	0.1731738	0.1936056	0.2246953	0.2444403

The problem was solved by solving 7 subproblems, as can be seen in Table 1; following solution was obtained:

$$\mathbf{X}^{T+1} = \begin{bmatrix} 0.123737 \\ 0.025318 \\ 0.063541 \\ 0.015156 \\ 0.240249 \\ 0.293503 \\ 0.266177 \end{bmatrix},$$

One may notice the problem meeded, out of a total of $N_2 = 27$ iterations, only $7 + 2 + 6 = 15$ iterations instead of maximum $N_1 = 127$ iterations.

Matrix **X**, which can be attached, is:

$$\mathbf{X} = \begin{bmatrix} 0.537992 & 0.019933 & 0.067778 & 0.011149 & 0.000000 & 0.318429 & 0.032175 \\ 0.065749 & 0.232329 & 0.067778 & 0.011149 & 0.000000 & 0.318429 & 0.032175 \\ 0.065749 & 0.019933 & 0.000000 & 0.011149 & 0.000000 & 0.318429 & 0.032175 \\ 0.065749 & 0.019933 & 0.067778 & 0.277149 & 0.000000 & 0.318429 & 0.032175 \\ 0.065749 & 0.019933 & 0.067778 & 0.011149 & 0.983669 & 0.318429 & 0.032175 \\ 0.065749 & 0.019933 & 0.067778 & 0.011149 & 0.000000 & 0.231154 & 0.032175 \\ 0.065749 & 0.019933 & 0.067778 & 0.011149 & 0.000000 & 0.318429 & 0.989473 \end{bmatrix}$$

If we want to find vector x^{T+1}, we consider vector x^{T+2} as being real, and we apply once more the described method.

The problem was solved with the help of an electronic data-processing system IBM 360/30, on the basis of the following logical diagram and of the source program, which can be written in FORTRAN language.

From the conceived logical diagram, one may notice that with mark no. 4, relation (18) is taken into account. Up to mark no. 5, matrixes **R** and S are calculated, and up to mark 8, matrix S is reversed, taking into account the fact that it is symmetrical. Up to mark no. 9, the first stage, i. e. $\mathbf{X}_0^i$, is calculated. There appear, in the second stage, three cases: the first one, when x_{11}^{i0} does not comply with restriction (5), the second one when x_{21}^{i0} does not comply with restriction (5), and the third one when, after the first two iterations, x_{11}^{i0} and x_{21}^{i0} do not comply with restriction (5). While testing the solution's cases, take into account the principale of excluded middle and relations (18), (19), (20). With mark no. 28, $x_i^{T+1}$is calculated, and with mark no. 29 it is written down. The program is taken again, for calculating next subproblem.

The effective calculation lasts about three minutes, and from the internal memory about 8 k Bytes are used.

The method was also verified by means of some examples, effected on the basis of both methods, the obtained solutions being very close.

REFERENCES

1. THEIL, H. VAN DE PANNE, C.: *Quadratic Programming as an Extension of Conventional Quadratic Maximization. Management Science*, nr. 7, 1961.
2. THEIL, H., GUIDO REY: *A Quadratic Programming Approach to the Estimation of Transition Probabilities.* „*Management Science*, nr. 7 1961.
3. TELSER, L. P.: *Least-Squares Estimates of Transition Probabilities.* „Measurement in Economics", cap. 11 *Studies in Matematical Economics and Econometrics in Memory of Iehuda Grünfeld*, Stanford University, Press, Stanford, California, 1963.
4. DIRECȚIA CENTRALĂ DE STATISTICĂ: *Anuarul Statistic al R.S.R.* 1969. Ed. by Direcția Centrală de Statistică, București, 1969

METHODE POUR PROLONGER UNE CHAINE MARKOV EN VUE DU TRAITEMENT DES DONNEES A L'AIDE DES SYSTEMES ELECTRONIQUES DE CALCUL

La prévision, en corrélation avec l'étude statistique de tendance dans un domaine, aide à la planification du développement de l'économie nationale ou d'un de ses compartiments, à la consolidation rigoureuse des décisions, en utilisant les méthodes scientifiques et à l'élimination des méthodes traditionnelles qui utilisent souvent l'expérience accumulée de façon empirique.

L'article traite d'une solution personnelle pour prolonger une chaîne Markov, finie multidimensionnelle, de telle façon que les écarts soient minimes, sans tenir compte des dimensions, utilisant la programmation quadratique et le traitement des données à l'aide des systèmes électroniques de calcul.

Cette méthode peut s'appliquer aux chaînes Markov ayant un nombre d'états, quelque grand qu'il soit, et permet de réaliser une simplification particulière du calcul, une réduction

importante du nombre d'itérations, une réduction de la mémoire interne réservée par le système électronique de traitement des données, une réduction substantielle du temps nécessaire au calcul, ainsi que la possibilité d'obtenir un programme source général.

МЕТОД ПРОДЛЕНИЯ ЦЕПИ МАРКОВА ДЛЯ ПЕРЕРАБОТКИ ДАННЫХ НА ЭЛЕКТРОННЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ

Прогноз, увязанный со статистическим изучением тенденции в данной области, служит для планирования развития народного хозяйства или одной из его областей, для наиболее точного обоснования решений путем научных методов и для устранения традиционных методов, часто основывающихся на эмпирически накопленном опыте.

Статья дает собственное решение вопроса продления многоразмерной конечной цепи Маркова с тем, чтобы отклонения были наименьшими, независимо от ее размера, на основе квадратичного программирования и переработки данных на электронных вычислительных системах.

Предлагаемый метод может быть применен к цепям Маркова с любым большим числом состояний и позволяет осуществлять особое упрощение расчета, значительное сокращение числа итераций, сокращение внутреннего запоминающего устройства, заложенного в электронную систему переработки данных, значительное сокращение времени расчета, а также дает возможность получать общую программу-источник.

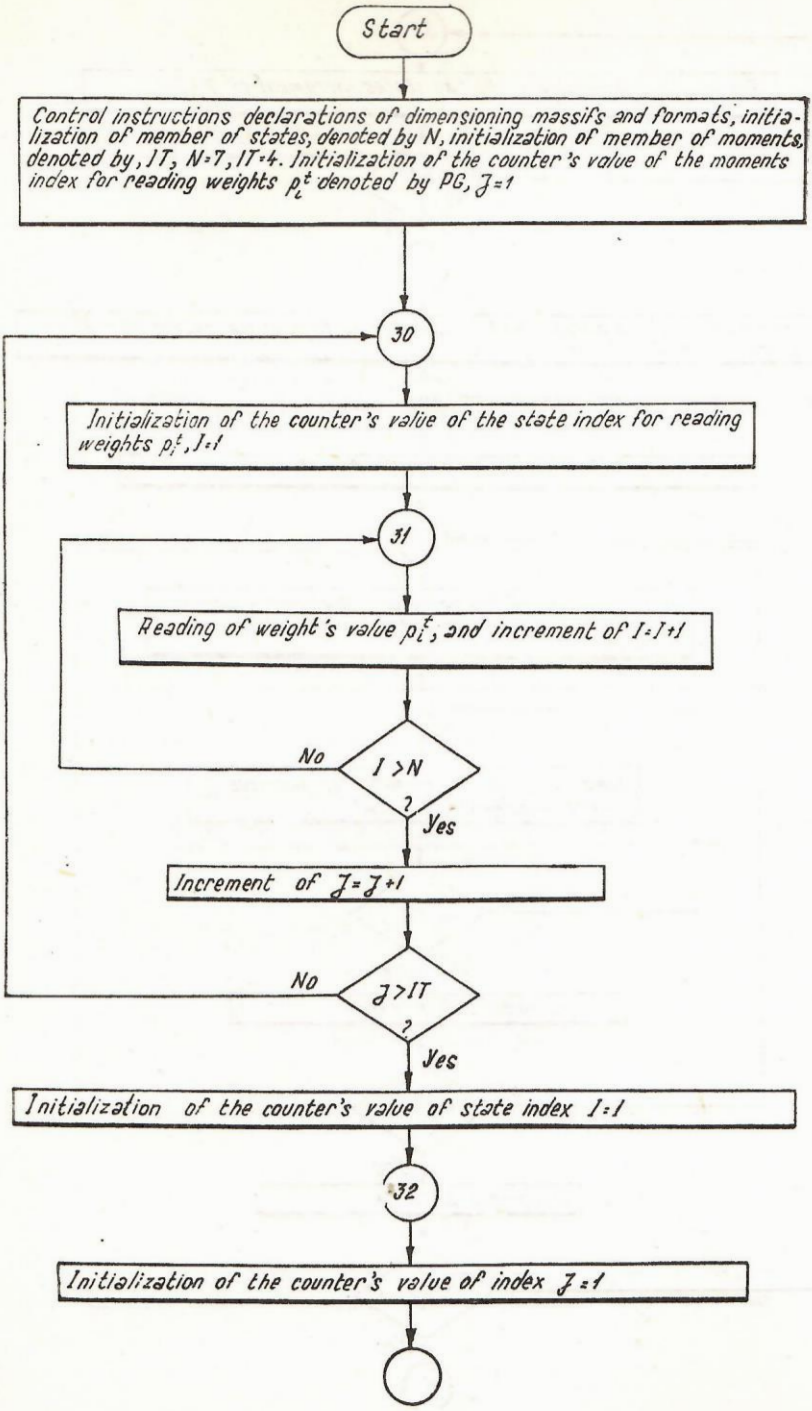
Start
Control instructions declarations of dimensioning massifs and formats, initialization of member of states, denoted by N, initialization of member of moments, denoted by, IT, N=7, IT=4. Initialization of the counter's value of the moments index for reading weights p_i^t denoted by PG, J=1
30
Initialization of the counter's value of the state index for reading weights p_i^t, I=1
31
Reading of weight's value p_i^t, and increment of I=I+1
No
I >N
?
Yes
Increment of J=J+1
No
J>IT
?
Yes
Initialization of the counter's value of state index I=1
32
Initialization of the counter's value of index J=1

Fig. 1

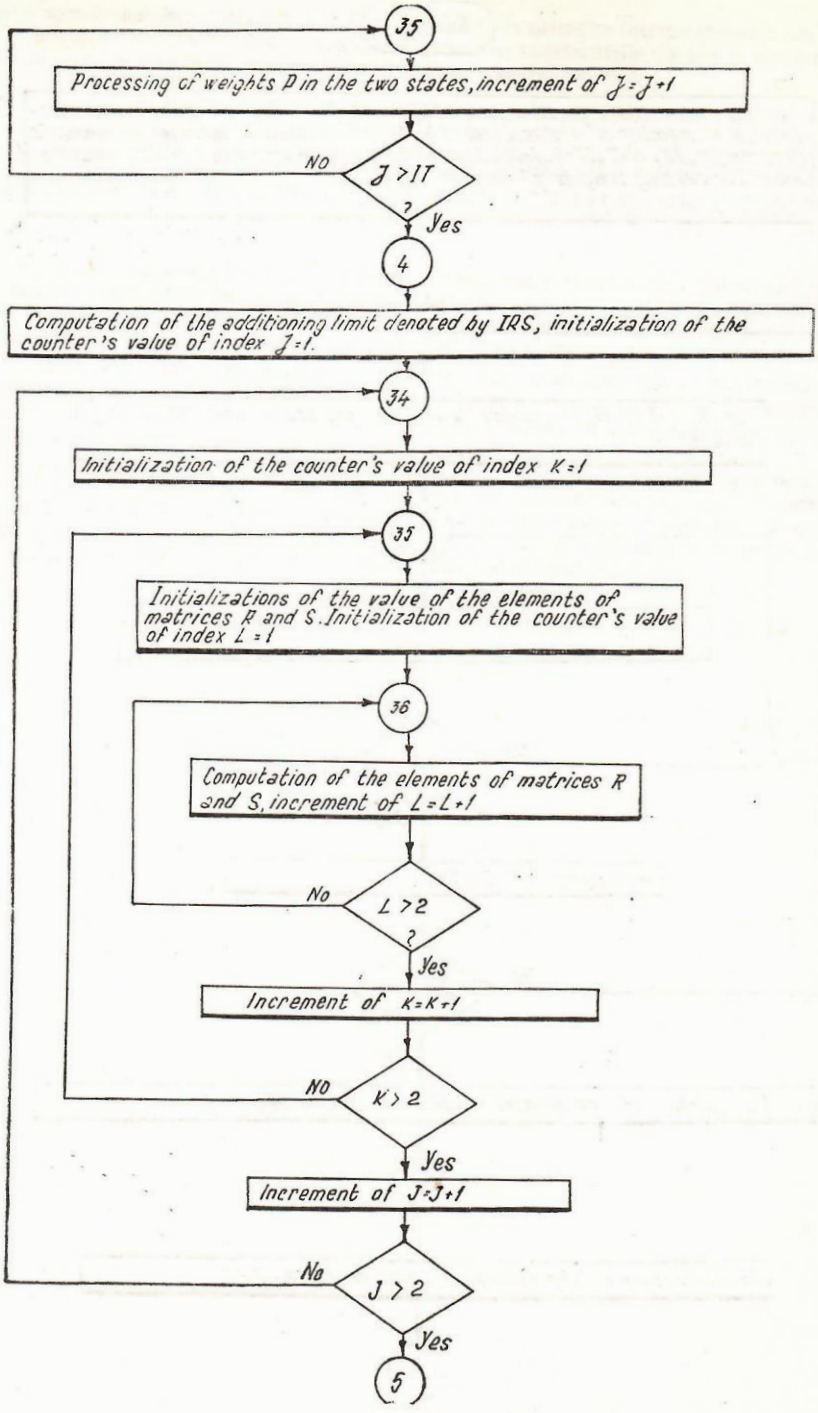
35
Processing of weights P in the two states, increment of J=J+1
No
J >11
?
Yes
4
Computation of the additioning limit denoted by IRS, initialization of the counter's value of index J=1.
34
Initialization of the counter's value of index K=1
35
Initializations of the value of the elements of matrices R and S. Initialization of the counter's value of index L=1
36
Computation of the elements of matrices R and S, increment of L=L+1
No
L >2
?
Yes
Increment of K=K+1
No
K > 2
Yes
Increment of J=J+1
No
J > 2
Yes
5

Fig. 2

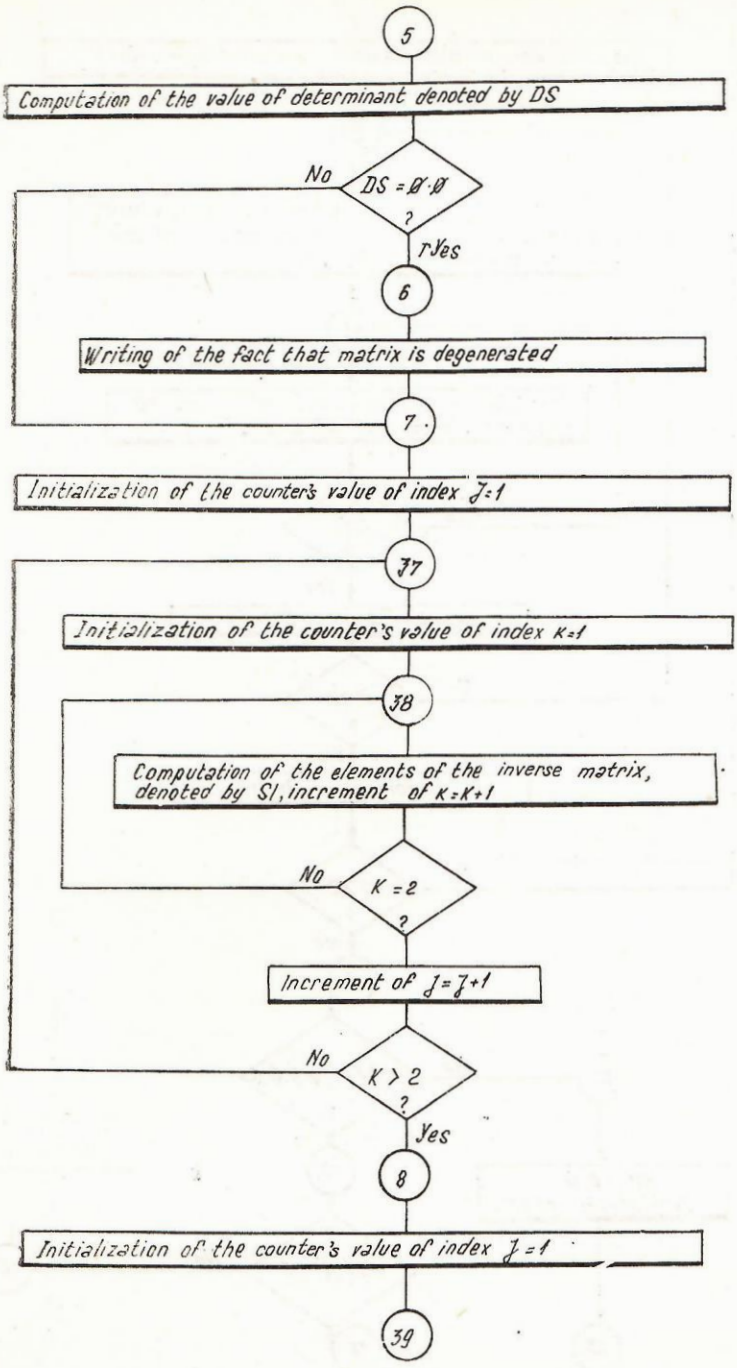
5
Computation of the value of determinant denoted by DS
No
DS = 0.0
?
Yes
6
Writing of the fact that matrix is degenerated
7
Initialization of the counter's value of index J=1
77
Initialization of the counter's value of index K=1
78
Computation of the elements of the inverse matrix, denoted by SI, increment of K=K+1
No
K = 2
?
Increment of J=J+1
No
K > 2
?
Yes
8
Initialization of the counter's value of index J=1
79

Fig. 3

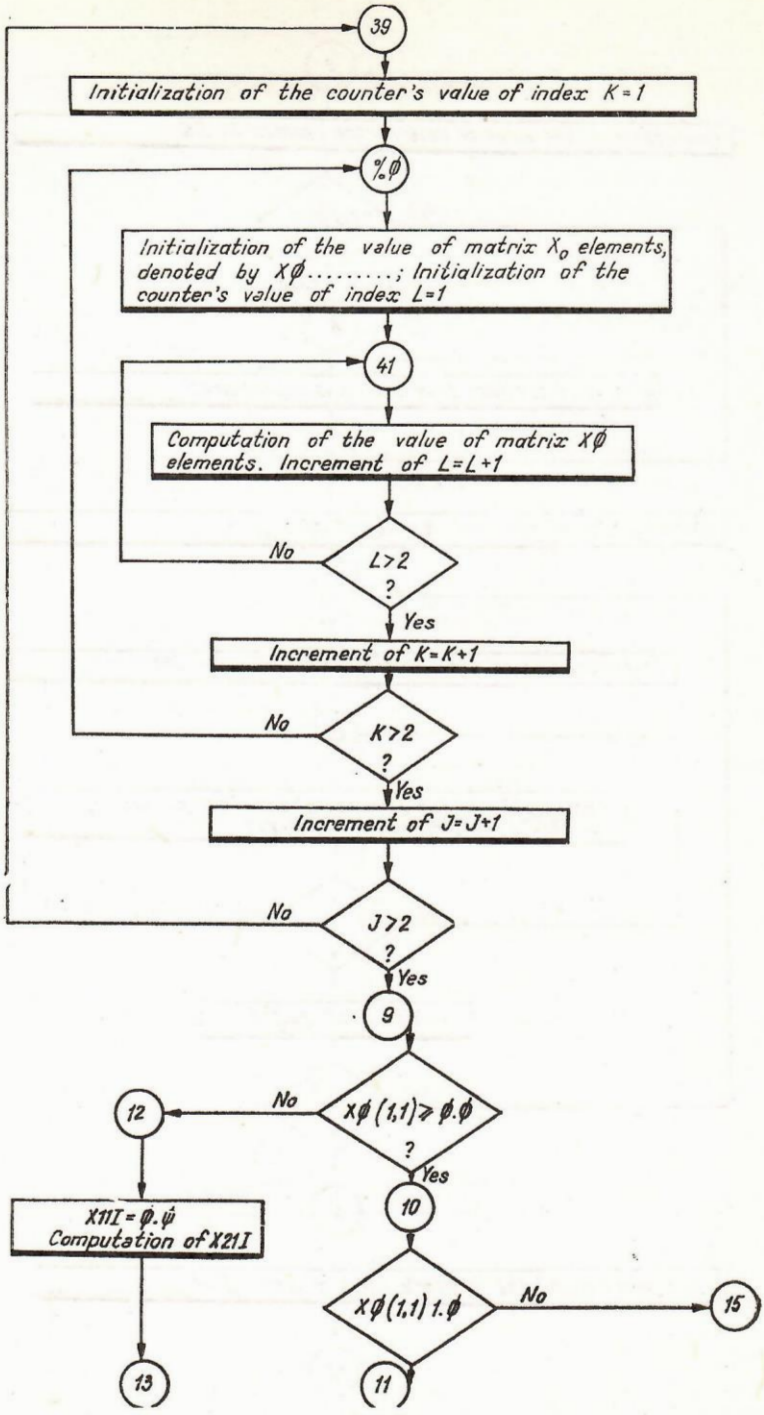
39
Initialization of the counter's value of index K=1
%ϕ
Initialization of the value of matrix X0 elements, denoted by Xϕ.....; Initialization of the counter's value of index L=1
41
Computation of the value of matrix Xϕ elements. Increment of L=L+1
No
L>2
?
Yes
Increment of K=K+1
No
K>2
?
Yes
Increment of J=J+1
No
J>2
?
Yes
9
No
Xϕ(1,1)≠ϕ.ϕ
?
Yes
12
10
X11I=ϕ.ψ
Computation of X21I
Xϕ(1,1) 1.ϕ
No
15
13
11

Fig. 4

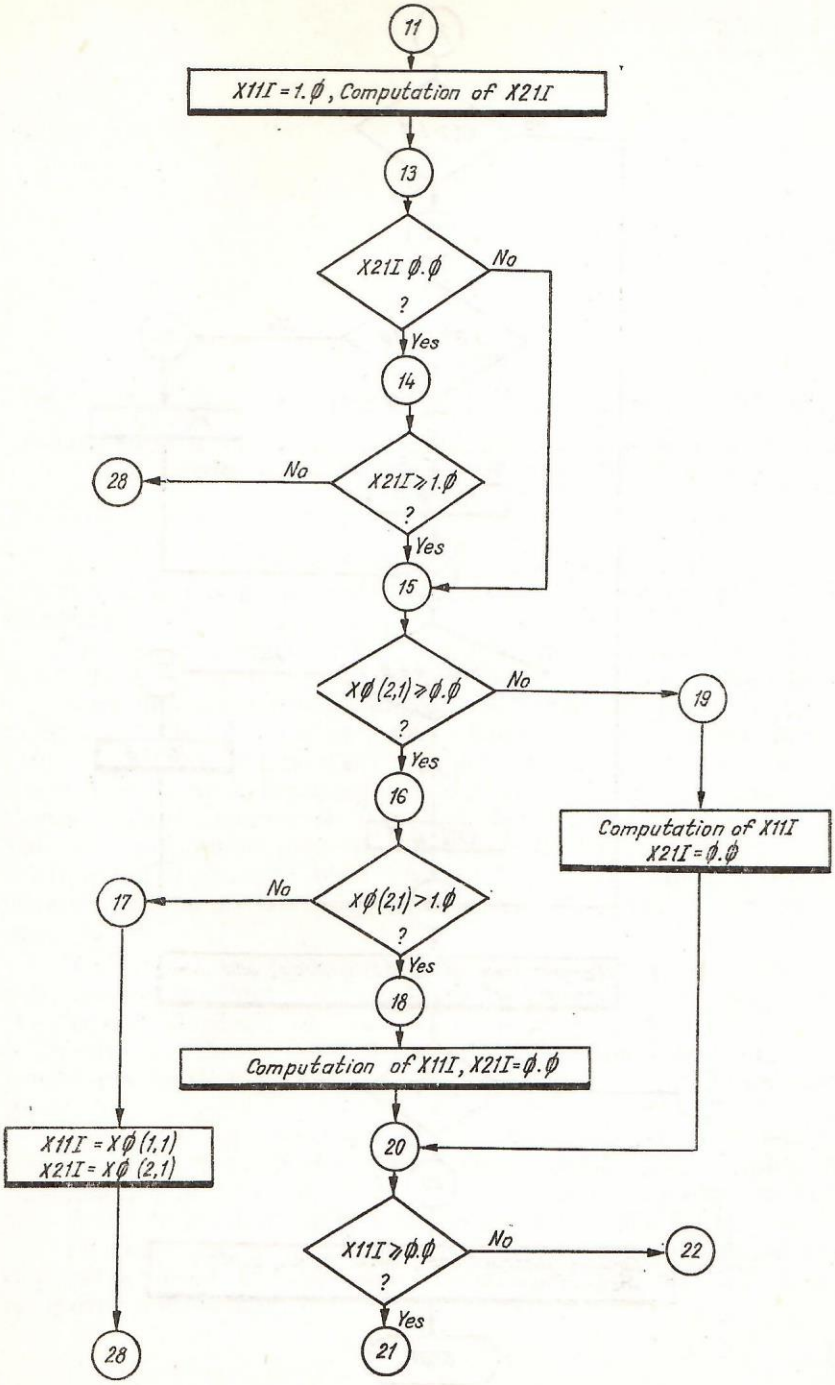
11
X11I = 1.Ø , Computation of X21I
13
X21I Ø.Ø
?
No
Yes
14
28
No
X21I ⩾ 1.Ø
?
Yes
15
XØ (2,1) ⩾ Ø.Ø
?
No
19
Yes
16
Computation of X11I
X21I = Ø.Ø
17
No
XØ (2,1) > 1.Ø
?
Yes
18
Computation of X11I, X21I = Ø.Ø
X11I = XØ (1,1)
X21I = XØ (2,1)
20
X11I ⩾ Ø.Ø
?
No
22
Yes
28
21

Fig. 5

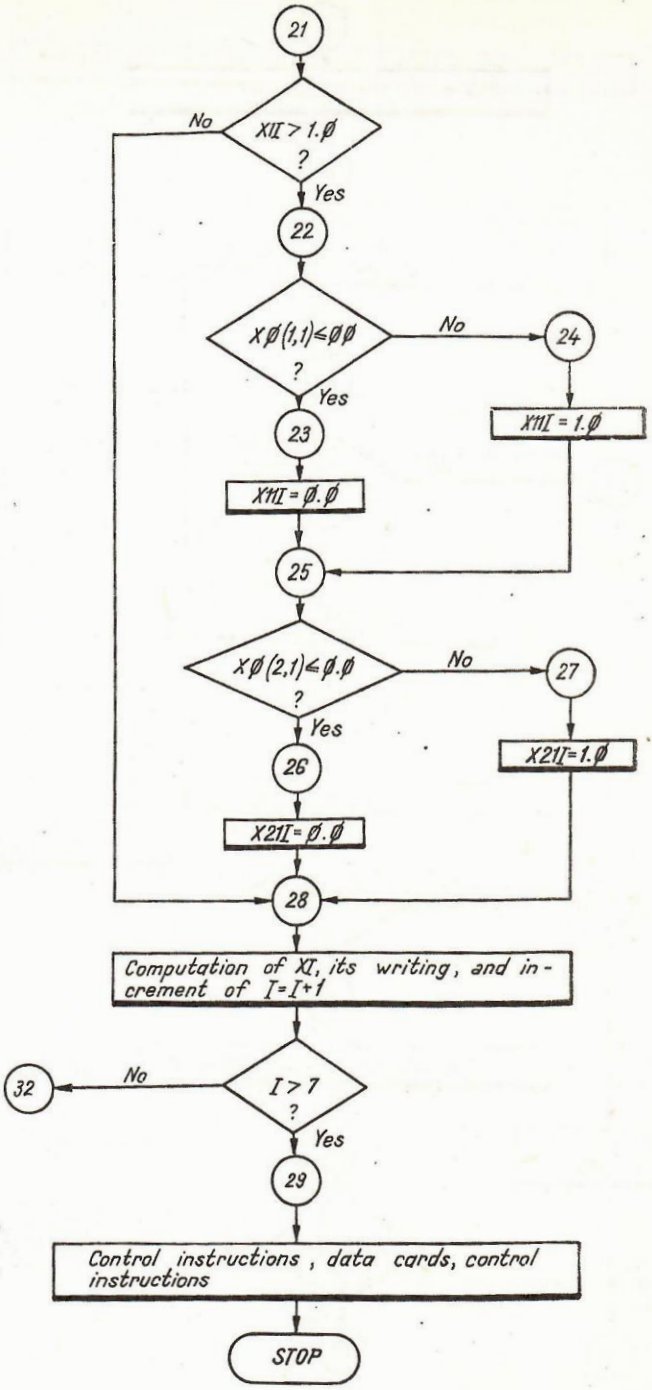
21
XII > 1.Ø
?
No
Yes
22
XØ(1,1)≤ØØ
?
No
24
Yes
23
XIII = 1.Ø
XIII = Ø.Ø
25
XØ(2,1)≤Ø.Ø
?
No
27
Yes
26
X21I = 1.Ø
X21I = Ø.Ø
28
Computation of XI, its writing, and increment of I=I+1
I > 7
?
No
32
Yes
29
Control instructions, data cards, control instructions
STOP

Fig. 6