STUDIA

UNIVERSITATIS BABEŞ-BOLYAI

MATHEMATICA

3

1988

CLUJ-NAPOCA

Anul XXXIII 1988

STUDIA
UNIVERSITATIS BABEŞ-BOLYAI

MATHEMATICA

3

Redacţia: 3400 CLUJ-NAPOCA, str. M. Kogălniceanu, 1 ● Telefon 1 61 01

SUMAR — CONTENTS — SOMMAIRE

2

STUDIA UNIV. BABEŞ-BOLYAI, MATHEMATICA, XXXIII, 3, 1988

A HEURISTIC SEARCH TYPE ALGORITHM FOR SOLVING NONLINEAR ECUATION SYSTEMS

BALÁZS MÁRTON ERNŐ*

Received: June 12, 1987

REZUMAT. — Tipuri euristice de algoritmi pentru rezolvarea unor sisteme de ecuaţii neliniare. În această lucrare se aduc unele îmbunătăţiri algoritmilor pentru rezolvarea sistemului nelinear prezentat în [1].

1. **Introduction.** In this paper we give some improvements to the nonlinear system solving algorithms given in [1]. For approximating the solutions of the system

$$f_j(x_1, x_2, \ldots, x_n) = 0,\ j = 1, 2, \ldots, m \tag{1}$$

in a previously given domain $S \subset R^n$, we suggested two algorithms based on the excluding method of K a l o v i c s [3]. In the followings we present this method.

The method is based on the following theorem:

THEOREM 1. (see [2] Theorem B) *Let $G \subset R^n$ be an open set and $D \subset G$ a compact and convex domain. Assume that the function $f: G \to R$ is twice continuously differentiable and that there is the positive real K such that*

$$K > \frac{1}{2} \max \left\{ \left[\sum_{i=1}^{n} \sum_{k=1}^{n} \left(\frac{\partial^2 f(x)}{\partial x_i \partial x_k} \right)^2 \right] :\ x \in D \right\}. \tag{2}$$

If $a \in D$, $f(a) \neq 0$ and

$$p(x) := p(f, a, x) := f(a) + f'(a)(x - a) + K \| x - a \|^2 \cdot \operatorname{sign} f(a) \tag{3}$$

then

$$[f(x) - p(x)] \cdot \operatorname{sign} f(a) > 0$$

for all $x \in D$, $x \neq a$.

The method for approximating the solutions of the equation $f(x) = 0$ may be formulated as follow:

1. Let T be a hyperparallelipiped such that $D \subseteq T$, and let us consider a lattice which divides T into a finite number of hyperparallelipipeds.

2. Let a be a node of the lattice such that $f(a) = 0$ and T_1 a hyperparallelipiped for which all the vertexes are nodes of the lattice. If for all the vertexes x of T_1 the relation

$$p(x) \cdot \operatorname{sign} f(a) > 0 \tag{4}$$

* *Research Institute for Computer Techniques and Informatics (ITCI), Str. Republicii no. 109, 3400 Cluj-Napoca, Romania*

holds, then according to Theorem 1, T_1 contains no solutions of the equation. This means that from T we may exclude the reunion of all hyperparallelipipeds with the same property.

3. If there remained unexcluded hyperparalellipipeds in T, we choose another node a and apply the method repeatedly until a covers all the vertexes of the unexcluded hyperparallelipipeds.

4. To examine the unexcluded domain we choose a refinement of the lattice and repeat the above givens. We stop the refinement when the maximal distance between two adjacent nodes of the lattice (the step of the lattice) is smaller than a previously given bound, which gives the precision of the approximation. We accept a hyperparallelipiped from the unexcluded domain to contain a solution of the equation if it contains a point x such that $|f(x)|$ is smaller then a given positive real.

The advantage of this method over the most popular iterative methods is that it has no points of divergence. It also has the interesting property of finding an approximation for each solution of the equation in the inspected domain. The major drawback of the method seems to be its low speed which is due to the sysematic inspection of all the possibilities.

The above given method is characterised by the following theorem and its consequences.

THEOREM 2. (see [2] Theorem C and its consequences) *Let*

$$d := \frac{1}{2K_j\sqrt{n}}\left(\sqrt{nK_j|f_j(a)| + ||f_j'(a)||^2} - ||f_j(a)||\right),$$

$$||f'(a)|| := \left[\sum_{i=1}^{n}\left(\frac{\partial f_j(a)}{\partial x_i}\right)^2\right]^{\frac{1}{2}}.$$

$$K_j > \frac{1}{2}\max\left\{\max\left\{\sum_{k=1}^{n}\left|\frac{\partial^2 f_j(x)}{\partial x_i \partial x_k}\right| : i = \overline{1, n}\right\} : x \in T\right\},\ j = \overline{1, m}$$

and

$$d_{j_0} = \max\{d_j : j = 1, 2, \ldots, m\}.$$

If the step of the lattice is smaller than d, then using a node a the domain excluded with the above given method contains at least one lattice-hyperparallelipiped.

Consequence 1. *Let $\varepsilon > 0$ and assume that for some j in $\{1, 2, \ldots, n\}$ in a domain U of T*

$$|f_j(x)| > \varepsilon$$

for all $x \in U$. In this case U will be excluded in a finite number of steps.

Consequence 2. *If $x^* \in T$ is not contained in any excluded rectangle of any lattice, then x^* is a solution of system (1).*

Consequence 3. *If (1) has no solution in T, then T will be excluded in a finite number of steps.*

Consequence 4. *If x^* is a solution of system* (1), *then for arbitrary $r > 0$ we can find a cube with the lengh of its sides r which contains x^*.*

2. **Algorithms using the excluding method.** Obviously the method given in the previous section is also valid for equidistant lattices. Using this kind of lattices has the advantage that it simplifies the computations and makes implementation easier. We shall call the hyperparallelipipeds in an equidistant lattice cubes. Another advantage of using equidistant lattices is that in R for defining a cube we need $n + 1$ informations (the n co-ordinates of a vertex and the lenght of its sides), while for defining a hyperparallelipiped we need $2n$ informations (the n co-ordinates and the n lenghts of its sides), which means an economy of $n - 1$ memory locations for each cube.

In [1] we gave two alforithms using this excluding method. These are similar to the depth-first and bredth-first search algorithms respectively, used in inspecting a solution space [4]. In the followings we present these algorithms with slight modofications. In both of the algorithms a list is used to store the unexcluded cubes at each moment.

Algorithm 1. (depth-first type algoritm)

In this algoritm the list used to store the unexcluded cubes is organized according to the last-in-first-out principle (LIFO).

Step 1. Let us consider a lattice with step d and introduce into the list the cubes which cover T, with side-lenghts $l = d$.

Step 2. If the list is empty stop.

Step 3. Take the cube from the top of the list, let this be C and let a be the vertex in the definition of C.

Step 4. If C may be excluded using a then go to Step 2.

Step 5. If $|f(a)| < \eta(\eta) > 0)$ and the side-lenght of C is smaller then the desired precision ($l < \varepsilon$), then choose a point x in C as the approximation of the solution in this cube; go to Step 2.

Step 6. Divide C into k^n cubes with the side-lenghts $l = 1/k$, go to Step 2.

Algorithm 2. (bredth-first type algorithm)

In this algorithm the list is organised according to the first-in-first-out principle (FIFO).

Step 1. Let us consider a lattice with step d and introduce into the list the unmarked cubes with their side-lenghts $l = d$ which cover T (a marked cube in the list means that its defining vertex has been used for exclusions).

Step 2. If the list is empty stop.

Step 3. Let C be the cube on the top of the list and a the vertex in the definition of C. If C is marked, go to Step 6.

Step 4. Delete from the list all the cubes which may be excluded from T using node a.

Step 5. If C was not excluded mark it and move it from the top of the list to its end; go to Step 2.

Step 6. If the side-lengths of the cubes (they all are equal at each moment) is smaller than the desired precision ($\varepsilon > 0$), then return as approximation of a solution an arbitrarily chosen point from each cube in the list which contains a point x such that $|f(x)|<\eta$ ($\eta > 0$); stop.

Step 7. Divide each cube C in the list into k cubes with side-lengths $l = 1/k$ and introduce them unmarked into the list; go to Step 2.

Obviously the choice of the positive integer k is arbitrary for both of the algorithms, although it should be made such that the memory requirements remain as low as possible and the bookkeeping operations simple. In the implementations we made we used $k = 2$.

Between the two algorithms we can make the usual comparison made in generally between depth-first search and bredth-first search algorithms 4].

Algorithms 4 keeps on the list all the unexcluded cubes with the side-lenght l while Algorithm 1 divides one cube at a time and immediately tries to exclude the obtained cubes. This results in a larger memory requirement for Algorithm 2.

The other difference between the two algorithms is that the first one proceeds searching one solution at a time while the second one works on finding all the solutions in parallel.

3. On informed algorithms. Both of the algorithms presented in the previous section are uninformed, i.e. the choice of the node used in an excluding step is arbitrary. It is desirable to use at every step the node which excludes the largest domain possible. To find such a node might be more difficult than solving the system itself. It seems to be more efficient to find a function which can choose "almost every time" the node which excludes the largest domain. We mean here a function like those used in heuristic serch (see [4]). In the followings we give such a function.

THEOREM 3. *Let $f: G \to R$ be as in Theorem 1 and $a, b \in G$ such that*

$$|f(a)| - \frac{1}{4K}\,||f'(a)||^2 > |f(b)| - \frac{1}{4K}\,||f'(b)||^2 \tag{5}$$

If $x^1, x^2 \in G$, $p(f, b, x^2) = 0$ and $||x^1 - x^a|| > ||x^2 - x^b||$, where x^a and x^b are the extremal points of the polinomials $p(f, a, x)$ and $p(f, b, x)$ respectively, then $p(f, a, x^1) \cdot \operatorname{sign} f(a) \geq 0$.

Proof. The points x^a and x^b have the co-ordinates

$$x_i^a = a_i - \frac{1}{2K}\,\frac{\partial f(a)}{\partial x_i} \cdot \operatorname{sign} f(a),\ i = \overline{1, n} \text{ and} \tag{6}$$

$$x_i^b = b_i - \frac{1}{2K}\,\frac{\partial f(b)}{\partial x_i} \cdot \operatorname{sign} f(b),\ i = \overline{1, n} \text{ respectively.}$$

The distance of a point $x \in G$ to x^a is

$$||x - x^a|| = \left[\sum_{i=1}^{n}\left(x_i - a_i + \frac{1}{2K}\frac{\partial f(a)}{\partial x_i} \cdot \operatorname{sign} f(a)\right)^2\right]^{\frac{1}{2}} \tag{7}$$

$$= \left\{\sum_{i=1}^{n}\left[(x_i - a_i)^2 + \frac{1}{K}\frac{\partial f(a)}{\partial x_i}(x_i - a_i) \cdot \operatorname{sign} f(a) + \frac{1}{4K^2}\left(\frac{\partial f(a)}{\partial x_i}\right)^2\right]\right\}^{\frac{1}{2}}$$

$$= \left[||x - a||^2 + \frac{1}{K}f'(a)(x - a) \cdot \operatorname{sign} f(a) + \frac{1}{4K^2}||f'(a)||^2\right]^{\frac{1}{2}}.$$

Analogously we obtain:

$$||x - x^b||^2 = ||x - b||^2 + \frac{1}{K}f'(b)(x - b) \cdot \operatorname{sign} f(b) + \frac{1}{4K^2}||f'(b)||^2$$

According to the hypothesis $p(f, b, x^2) = 0$, i.e.

$$f(b) + f'(b)(x^2 - b) + K||x^2 - b||^2 \cdot \operatorname{sign} f(b) = 0,$$

thus

$$|f(b)| + f'(b)(x^2 - b) \cdot \operatorname{sign} f(b) + K||x^2 - b||^2 = 0.$$

Using (7) we obtain the followings:

$$0 = |f(b)| + K\left[\frac{1}{K}f'(b)(x^2 - b) \cdot \operatorname{sign} f(b) + ||x^2 - b||\right]$$

$$= |f(b)| + K\left[||x^2 - x^b||^2 - \frac{1}{4K^2}||f'(b)||^2\right]$$

$$= |f(b)| - \frac{1}{4K}||f'(b)||^2 + K||x^2 - x^b||^2 \leqslant$$

$$< |f(a)| - \frac{1}{4K}||f'(a)||^2 + K||x^1 - a||^2$$

$$= |f(a) + K\left[||x^1 - x^a||^2 - \frac{1}{4K^2}||f'(a)||^2\right]$$

$$= |f(a)| + K\left[\frac{1}{K}f'(a)(x^1 - a) \cdot \operatorname{sign} f(a) + ||x^1 - a||^2\right]$$

$$= [f(a) + f'(a)(x^1 - a) + ||x^1 - a||^2 \operatorname{sign} f(a)] \operatorname{sign} f(a)$$

$$= p(f, a, x^1) \cdot \operatorname{sign} f(a),$$

which completes the proof.

Consequence 1. *Let* $a \in G$ *and* $x \in G$ *such that* $p(f, a, x) = 0$.
If $y \in G$ *is such that* $||y - x^a|| < ||x - x^a||$, *then*
$p(f, a, y)$ sign $f(a) > 0$;
If $y \in G$ *is such that* $||y - x^a|| = ||x - x^a||$, *then*
$p(f, a, y)$ sign $f(a) = 0$;
If $y \in G$ *is such that* $||y - x^a|| < ||x - x^a||$, *then*
$p(f, a, y)$ sign $f(a) < 0$.

Consequence 2. *Let* $a, b \in G$ *which satisfie relation* (5) *and* $x^1, x^2 \in G$ *such that* $p(f, a, x^1) \cdot$ sign $f(a) > 0$ *and* $||x^2 - x^a|| > ||x^2 - x^b||$, *then* $p(f, b, x^2) \cdot$ sign $f(b) > 0$.

This latter consequence means that for two points $a, b \in G$, using the one for which the value of the function $h: G \to R$ defined by

$$h(x) = |f(x)| - \frac{1}{4K} ||f'(x)||^2$$

is greater, we can exclude a larger domain.

In the case of searching the solutions of an equation, in a given rectangular domain T using the excluding method, the fuction may not always give the best choice. This is because the domain which could be excluded from G may not be entirely contained in T. However h gives a choice criterion for the node which will be used for exclusion. Based on this "heuristic" function we give an informed version of Algorithm 2.

Algorithm 3. (best-first type algorithm)

Step 1. Let us consider a lattice with step d and let $C_1, C_2, \ldots, C_m$ be the cubes with side-lenght $l = d$ for which $G \subset C_i$; for C_i $(i = \overline{1, m})$ compute $h(a_i)$ (where a_i is the vertex in the definition of C_i) and introduce them into the list unmarked.

Step 2. If the list is empty stop.

Step 3. Let C be an unmarked cube in the list for which $h(a)$ is maximal; if there are no unmarked cubes in the list go to Step 5, otherwise mark C.

Step 4. Delete from the list all the cubes which may be excluded from T using node a; go to Step 2.

Step 5. If the side-lengths of the cubes in the list are smaller than the desired precision $(\eta > 0)$, then return as approximation of a solution an arbitrarily chosen point from each cube which contains a point x such that $|f(x)| < \eta$ $(\eta > 0)$; stop.

Step 6. Divide each cube C in the list into k^n cubes with the side-lengths $l = 1/k$; for each cube obtained compute the value of the function h in the vertex from its definition and introduce them into the list unmarked; go to Step 2.

To compute the values of the function h in nodes of the lattice we don't need extra informations since both $f(a)$ and $f'(a)$ are used in constructing the polinomials $p(f, a, x)$ aswell. In order to avoid recomputation of these values it is useful to store them together with the definitions of the cubes. For this reason we suggest that in an implementation of the algorithm a cube should be a record structure with the following components:

— the coordinates of the vertex: array of n reals;

— the side-length: real;

— the value of f in the vertex: real;

— the values of the partial derivates in the vertex: array of n reals;

— the value of h in the vertex: real.

A further improvement to the algorithm would be if in Step 4 one should not examine all the cubes in the list. According to Consequence 1 of Theorem 3 the test should be made only to those cubes for which each vertex x satisfies the relation $||x - x^a|| < ||x^j - x^a||$ (where x is such that $p(f, a, x)=0$. Obviously it is not worth computing these distances for each vertex of each cube to be tested but the sequence in which the testing of cubes is made can be chosen in such a manner that the above given condition tells when the testing may be stopped.

Let v be a vertex in the definition of a cube in the list at Step 4. We organise the testing on levels: level 0 are those cubes which have x as one of their vertexes; level 1 are those untested cubes which are adjacent to level 0; ... level k are those untested cubes which are adjacent to level $k - 1$.

THEOREM 4. *If in level k no cubes were excluded, then in levels $k > k_0$ no cubes will be excluded either.*

Proof. If in level k no cubes were excluded then for each vertex v of each cube on level k $p(f, a, v_k) \cdot \operatorname{sign} f(a) > 0$. Since the distances of the closest vertexes on level k to v are $k \cdot l$ all the vertexes of the cubes on level $k+1$ are at a distance greater or equal than $(k + 1) \cdot l$ to v. This means that for each vertex of the cubes on level $k > k_0$, $p(f, a, v_k) \operatorname{sign} f(a) < 0$.

Remark. v may be chosen arbitrarily but it is convenient to choose it as close as possible to x.

The algorithm is rather complicated but it might be useful in isolating the solutions of complex systems to a level at which fast iterative methods can be used.

4. **Implementation notes.** As we mentioned at the end of the previous section the algorithm presented should be used in combination with a fast converging method e.g. Newton's method. This means that the cubes which satisfie the precision criterion are passed to such an algorithm.

In the algorithm there are several processes which may be executed concurrently:

— managing the database which contains the unexcluded cubes at each moment (adding, searching, deleting cubes);

— testing of cubes (which may be done concurrently);

— dividing and constructing cubes (which also may be done in parallel).

If the algorithm is used to approximate the solutions of a system of form (1), then the algorithm may be executed concurrently for each $f(x) = 0$, $j=\overline{1,m}$. This means that in applying the algorithm for such systems there are two levels of parallelism.

In a future paper we shall give a computer program written in ADA which is intended to be the base of a larger equation solving system.

REFERENCES

1. Balázs, M.E.: *Algorithms Using Excluding Methods For Nonlinear System Solving.* "Babeş–Bolyai" University Cluj-Napoca, Faculty of Mathematics Seminar on Computer Science, Preprint Nr. 2, (1986).
2. Balázs, M. — Kolumbán, J.: *Solving Nonlinear Systems Using an Excluding Method.* Proc. of the Colloquium on Approximation and Optimization, Cluj-Napoca (1984).
3. Kalovics, F.: *Nemlineáris egyenletrendszerek megoldása érintő paraboloid módszerrel.* NME Közleményei (Miskolc) IV. sorozat, Természettudományok 23 (1977).
4. Pearl, J.: *Heuristics.*
5. Szabó, Z.: *Über gleichungslosende Iterationen ohne Divergenzpunkt, I–II.* Publ. Math. Debrecensis 20 (1973).

STUDIA UNIV. BABEŞ-BOLYAI, MATHEMATICA, XXXIII, 3, 1988

ON THE APPROXIMATION OF A CUBIC SPLINE CURVE BY CIRCULAR ARCS

IULIU VLAIC* and ANGELA VASIU**

Received: April 15, 1988

REZUMAT. — Aproximarea unei curbe cubice spline prin arcuri circulare. Prezenta notă dă o soluţie numerică pentru determinarea curbelor plane, definite prin puncte discrete, în vederea determinării profilelor plane prin prelucrare pe maşini-unelte, cu comandă numerică. Algoritmul dat, calculează mai întîi o curbă spline care este apoi aproximată printr-un lanţ de arce de cerc tangente unul celuilalt. Este prezentat un program pentru calculator în limbajul BASIC pe calculatorul de birou HEWLETT—PACKARD 9845B.

1. This paper presents a numerical solution to determine a plane curve, defined by n given points. First, a cubic spline interpolation is calculated, then this is approximated by a string of circular arcs tangent each other in such a way that the error in each point is within a given limit prescribed by the user.

The computation and graphics representation programme is realized in BASIC programming language on the Hewlett—Packard 9845B desktop computer. The graphics is plotted within HP 9872B plotter.

2. **Theoretical Formulation.** Given a plane profile defined by points which are known by their coordinates, we ask:

1. The theoretical cubic spline function which interpolates the profile;

2. The approximation of the spline function by a string of circular arcs tangent each other such that the error in each point does not exceed a given ε.

2.1. Let (x_0, y_0), (x_1, y_1), ..., (x_n, y_n) be the given points with:

$$x_0 < x_1 < \ldots < x_n \tag{1}$$

The spline function $F_s: [x_0, x_n] \to \mathbf{R}$ is composed of arcs of polynominals of degree three, joined continuously with continuous first and second derivatives. For the first and the last points we have the additional conditions of tangence or curvature:

$$\begin{cases} F_s'(x_0) = \operatorname{tg}(\alpha_0) \\ F_s'(x_n) = \operatorname{tg}(\alpha_n) \end{cases} \tag{2}$$

or

$$\begin{cases} F_s''(x_0) = c_0 \\ F_s''(x_n) = c_n \end{cases} \tag{3}$$

where the constants α_0, α_n and c_0, c_n respectively are supplied by the user.

* *Mechanical Entreprise of Cugir, 2566 Cugir, Romania.*
** *University of Cluj-Napoca, Faculty of Mathematics and Physics, 3400 Cluj-Napoca, Romania.*

By [6] the function F_s has the following expression:

$$F_s(x) = y_i + (x - x_i)(y_{i+1} - y_i)/(x_{i+1} - x_i) + \\ + \frac{1}{6}(x - x_i)(x - x_{i+1})[F_s''(x_i) + F_s''(x_{i+1}) + F_s''(x)] \tag{4}$$

where

$$x \in [x_i, x_{i+1}], \; i = \overline{0, n}$$

In case of condition (2), one obtains the solution of the corresponding system of equations by matrix methods.

If in (3) we take $c_0 = c_n = 0$, then we determine F_s as an iterative solution of a linear system of equations, using, for instance, the method of Young, where the precision of the solution can be fixed by the user. This last case is described by the computer programme in Table 1.

3. **The Algorithm of Approximation by Circular Arcs.** The approximation of F_s by a string of circular arcs is determined as follows below. The circle passing through distinct points (a_0, b_0) and (a_1, b_1) and having $m = \operatorname{tg} \alpha$ as the slope of its tangent in (a_0, b_0) has the centre:

$$x_c = a_0 + mb_0 - my_c \tag{5}$$
$$y_c = A/B$$

where

$$A = a_0^2 + b_0^2 - a_1^2 - b_1^2 + 2(a_1 - a_0)(a_0 + mb_0)$$
$$B = 2[b_1 - b_0 - m(a_1 - a_0)]$$

and the radius:

$$R = [(x_c - a_0)^2 + (a_c - b_0)^2]^{1/2} \tag{6}$$

If $B = 0$ the circular arc is replaced by a line segment. If $\alpha = \frac{\pi}{2}$ the (5) formulas become:

$$y_c = b_0$$
$$x_c = [a_1^2 - a_0^2 + (b_1 - b_0)^2]/[2(a_1 - a_0)] \tag{7}$$

if $a_1 \neq a_0$.

The next circular arc has a common tangent with the preceding circular arc in (a_1, b_1) which becomes (a_0, b_0). We have:

$$m = -(a_1 - x_c)/(b_1 - y_c) \text{ if } b_1 \neq y_c$$

and

$$m = \pm\infty \text{ if } b_1 = y_c.$$

Let ε be the precision of approximation of F_s by circular arc, which is given of the user. If $g; [a_0, a_1] \to \mathbf{R}$ defines the circular arc obtained with

the formulas (5) and (6), we have:

$$F_s(a_0) = g(a_0),\ F_s(a_1) = g(a_1) \qquad (8)$$

because the points (a_0, b_0), (a_1, b_1) are, at the same time, on the theoretical curve and on the circular arc. This means that

$$y = |F_s - g| : [a_0, a_1] \to \mathbf{R}$$

has Q maximum, which we ask not to depasse ε.

We ask that:

$$\max y < \varepsilon \text{ when } x \in [a_0, a_1] \qquad (9)$$

In the domain $[x_0, x_n]$ with a variable Δx step we obtain points: (a_0, b_0), (a_1, b_1) ..., by which we consider the circular arcs as we have considered above. The last point of every calculated circular arc, except the last point x_n, is the first point of the next circular arc. The step Δx is modified, increasing it, and so we obtain different arcs, and we test for every one a condition of type (9). We retain the last string of circular arcs for which (9) is still satisfied. In this way we minimize the number of the circles of approximation of F_s by circular arcs.

In Table 1 is given an algorithm to determine a cubic spline of interpolation. We denote by (x_i, y_i), $i = \overline{0, n}$ the given points, by $(x, F_s(x))$ an arbitrary point of the theoretical spline curve; by Pas x, the step Δx of the crossing of the $[x_0, x_n]$ interval. D1 $\div$ D10, E_i, F_i, G_i $i = \overline{0, n}$ are some auxiliar variables.

A complet programme, in view of processing a profile on a machine-tool, contains after the determination of the F_s, the processings of approximation of F_s by circular arcs and records the final data in CL—FILE (Cataloging File), [2], [3], [5].

We mention that a post-processing is asociated for the machine-tools with numerical processing command, which "compile" the profile in machine instructions. Different codes are used: ISO, APT etc. To this end, the authors have created such a post-processing, for the machine AGIE CUT, DEM 29 (Switzerland).

The approximation algorithm with arcs, for the cubic-spline function, remains unchanged for any plane contour type, i.e. for any contour defined by implicit, explicit or parametric functions.

4. **An example of using of the programme.** Given the plane curve, by the individual points P0[—150, 20], P1[—120, 10], P2[—80, —10], P3[—40, —5], P4[—10, 5], P5[20, 20], P6[40, 30], P7[70, 45], P8[100, 50], P9[130,45], P10[150, 35]. We ask:

1) the graphic representation of the spline-function of interpolation; we choose Pas x = 0, 1;

2) the graphic of the approximated curve, by the calculated circular arcs.

Using the programme given in Table 1 it resulted the cubic-spline curve of interpolation, with the graphic representation given in Fig. 1. The indivi-

dual given points, which define the function, are noted by the '+' sign. The approximated curve is represented in Fig. 2, at the same scale with the theoretical profile from Fig. 1. The ends of the circle arcs are noted by "*".

All figures are plotted on the HEWLETT—PACKARD 9845B desktop computer.

Table 1

The computation and graphics representation programme, in BASIC programming language, for a cubic-spline function of interpolation

```
10    REM  ********************************************
20    REM  *** DETERMINATION OF A PLANE CURVE  *********
30    REM  *** DEFINED BY N DATA POINTS              ***
40    REM  ***                                      ***
50    REM  *** APPLICATION OF CUBIC—SPLINE          ***
60    REM  *** FUNCTION OF INTERPOLATION            ***
70    REM  ********************************************
80    REM
90    REM          CUGIR 1987
100   REM
110   Select = 16
120   INPUT "Selective code of the printer?" Select
130   PRINTER IS Select
140   DEG
150   INPUT "Number of points N = ?"
160   INPUT "The graphics representation scale ?" Sc
170   PLOTTER IS "9872A"
180   SCALE — 190, 190, —125, 125
190   FRAME
200   PEN 1
210   MASS STORAGE IS ": F8"
220   PLOT — 175, 0
230   PLOT 175, 0
240   PLOT 172, 2
250   PLOT 172, —2
260   PLOT 175, 0
270   PENUP
280   PLOT 0, —90
290   PLOT 0, 110
300   PLOT —2, 107
310   PLOT .2, 107
320   PLOT 0, 110
330   PENUP
340   MOVE —5, —5
350   LABEL "0"
360   MOVE 170, —5
370   LABEL "X"
380   MOVE —5, 100
390   LABEL "Y"
400   PLOT —175, 100
410   PLOT —145, 100
420   PENUP
430   LORG 5
440   MOVE —175, 100
450   LABEL ":"
460   MOVE —145, 100
470   LABEL ":"
480   MOVE —175, 110
```

```
490     LABEL "0"
500     MOVE -145, 110
510     LABEL "30"
520     LORG 1
530     CSIZE 3
540     MOVE -175,-110
550     LABEL "Fig. 1 — The graphics of interpolation spline-function for individual
            points"
560     LABEL "
570     LABEL " "
580     OPTION BASE 0
590     DIM F(100), G(100), E(100), X(100), Y(100)
600     FOR I=0 TO N
610     PRINT I;
620     INPUT The rectangular coordinates of the points
            (X, Y, CONT )?", X(I), Y(I)
630     PRINT "X = "; X(I);"     Y = "; Y(I)
640     NEXT I
650     GRAPHICS
660     LORG 5
670     FOR I=0 TO N
680     MOVE X(I), Y(I)
690     LABEL "+"
700     PENUP
710     NEXT I
720     LORG 1
730     GOSUB Print
740     PRINT "If there are some changing introduce the index of the point which must
            be changed. Otherwise press the clak 'CONT' "
2)
760     Edite: I=PI
770     INPUT "Introduce the index which want to change" I
780     IF I=PI THEN 920
790     I=INT(I)
800     IF (I<0) OR (I>N) THEN Edition"
810     DISP "Introduce coord. for the point nr."; I; "(X,Y, CONT)"
820     INPUT " ", X(I), Y(I)
830     PRINT USING Image; I, X(I), Y(I)
840     GOTO Edition
850     Print: PRINT LIN (2), SPA(12); "DATE INITIALE"
860     FOR I=0 TO N
870     PRINT USING Image; I, X(I), Y(I)
880     Image: IMAGE "Point nr. "DDDD";" 5X, "X="K, 5X, "Y="";
890     NEXT I
900     PRINT I IN(2)
910     RETURN
920     INPUT "Pas abscisa Pasx = ?" Pasx
930     D10 =.001     ! Precission of approximation"
940     FOR I=1 TO N-1
950     D1=X(I)
960     D2=X(I-1)
970     D3=X(I+1)
980     D4=Y(I)
990     D5=Y(I-1)
1000    D6=Y(I+1)
1010    D7=D1-D2
1020    D8=D3-D2
1030    E(I)=.5*D7/D8
1040    D9=((D6-D4)/(D3-D1)-(D4-D5)/D7)/D8
1050    F(I) = 2*D9
```

```
1060    G(I)=3*D9
1070    NEXT I
1080    F(0)=0
1090    F(N)=0
1100    D3=8-4*SQR(3)
1110    D1=0
1120    FOR I=1 TO N-1
1130    D9=D3*(-F(I)-E(I)*F(I-1)-(.5-E(I))*F(I+1)+G(I))
1140    D8=ABS(D9)
1150    IF D8>D1 THEN 1180
1160    IF D1 > =D10 THEN 1110
1170    GOTO 1200
1180    F(I)=D9+F(I)
1190    NEXT I
1200    FOR I=0 TO N-1
1210    G(I)=(F(I+1)-F(I))/(X(I+1)-X(I))
1220    NEXT I
1230    PEN 1
1240    FOR X=X(0) TO X(N) STEP Pasx
1250    GOSUB Interpolar
1260    PLOT Sc*X, Sc*F
1270    NEXT X
1280    X=X(N)
1290    GOSUB Interpolar
1300    PLOT Sc*X, Sc*F
1310    PENUP
1320    MOVE 999, 999
1330    STOP
1340    Interpolar : IF (X > = X(0)) AND (X < =X(N)) THEN 1370
1350    DISP "ARGUMENT OUT OF LIMITS"
1260    STOP
1370    D9=X
1380    GOSUB Unu
1390    RETURN
1400    Unu: I=0
1410    IF D9> =X(0) THEN 1470
1420    PRINTER IS 0
1430    PRINT "ARGUMENT OUT OF LIMITS!"
1440    PRINT " X(0)="; X(0);" X(N)="; X(N);" X=";X
1450    PRINTER IS 16
1460    STOP
1470    I=I+1
1480    IF I > N THEN 1420
1490    IF D9 > X(I) THEN 1470
1500    I=I-1
1510    D8=X-X(I)
1520    D9=X-X(I+1)
1530    D7=D8*D9
4540    D15=F(I)+D8*G(I)
1550    D6=1/6
1560    D1=D6*(F(I)+F(I+1)+D15)
1570    D3=(Y(I+1)-Y(I))/(X(I+1)-X(I))
1580    F=D3*D8+Y(I)+D7*D1
1590    RETURN
1600    END
```

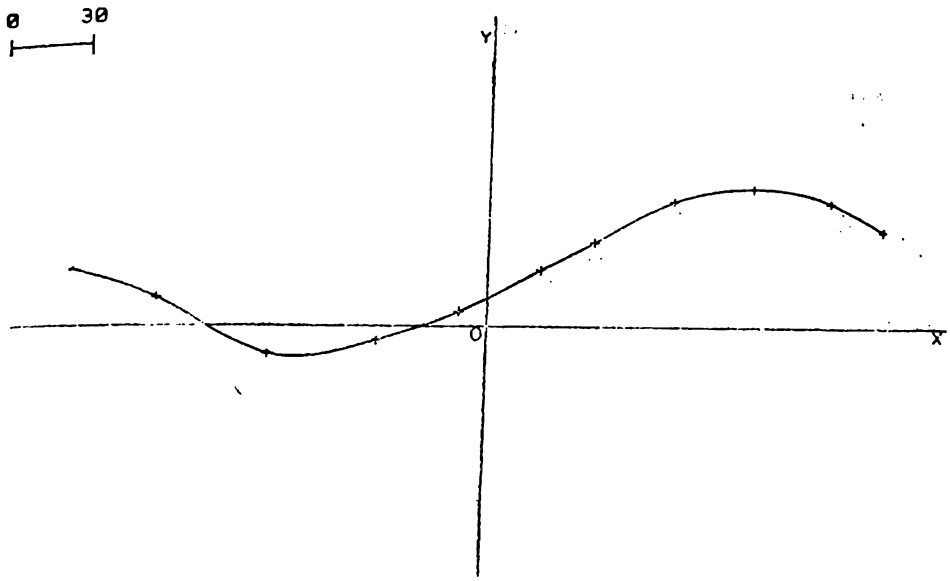


Fig. 1. The cubic-spline curve of interpolation, for exemple in paragraph 4.

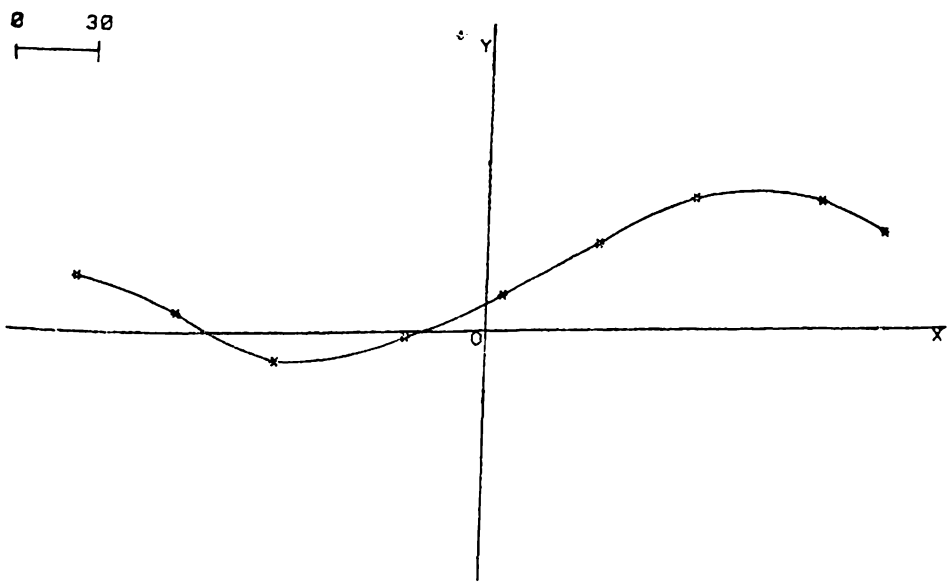


Fig. 2. The grafic of cubic-spline curve, approximated by circular arcs.

BIBLIOGRAPHY

1. Bacvalov, N.: Methodes numeriques, Editions Mir, Moscou, 1976.
2. Bezier, P.: Emploi des machines a commande numerique, Masson et Cie, Paris, 1970
3. Daulte, J. J.: Eine Methode zur Beschreibung zweidimensionale Kurven, Fertigung 5, 1
Verlag Rundschau, Hallwag, Bern.
4. Micula, G.: Funcţii spline şi aplicaţii, Editura Tehnică, Bucureşti, 1978.
5. Vlaic, I., Bernath, A.: Controlul numeric computerizat în realizarea unor profile a
trare, utilizînd unele metode de interpolare. Sesiunea de referate şi comunicări tehnico-ştii
fice Matematica şi Producţia; Deva, 12.05.1980.
6. * * * Numerical Analysis Library, vol. I, Part. No. 09845—10351, 1979. System 45B HE
LETT—PACKARD 3404 East Harmony Road, Fort Collins Colorado 80525, Palo Alto,
lifornia, USA.

STUDIA UNIV. BABEŞ-BOLYAI, MATHEMATICA, XXXIII, 3, 1988

MONTE CARLO INTEGRATION ON SIMPLEX

PETRU BLAGA*

Received: April 15, 1988

REZUMAT. — Integrare Monte Carlo pe simplex. În prezenta lucrare se studiază evaluarea unei integrale multiple pe simplex, folosind metode probabilistice. Se utilizează reducerea dispersiei estimatorului folosit cu ajutorul metodei separării părţii esenţiale şi respectiv metoda alegerii esenţiale. De asemenea se consideră două scheme combinate ale acestora. Metodele considerate folosesc polinoamele lui Bernstein definite pe simplex. Experimentele numerice confirmă utilitatea aplicării acestor metode de reducere a dispersiei. Rezultatele obţinute sînt comparate cu cele obţinute în cazul aplicării metodei Monte Carlo clasice.

0. It is known that a definite integral can be estimated using probabilistic methods, and these methods are preferably to approximate the definite integrals when multiple integrals are considered. The integral is looked as expectation a certain random variable, which is an unknown parameter. The estimation of this parameter, i.e. the definite integral, can be obtained if one performs a sampling from random variable considered, and taking an unbiased estimation function for this parameter. Generally, this method is not fast-converging ratio to volume of sampling, and efficiency depends on the variance of estimator. For increasing the efficiency must to reduce the variance as much as possible. Two important methods for reducing of the variance are known: the method of control variates, and the method of importance sampling [3].

To evaluate the multiple integrals on the unit hypercube, the two methods were used in [5]. The Bernstein polynomials were considered to reduce the variance with the above mentioned methods. We consider the same problem in the case when the integration domain is the n-dimensional standard simplex. Some numerical results are presented when there are applied the two methods to reduce of variance, and combinated schemes of these. All these techniques are raported to crude Monte Carlo method.

1. Let S_n be the n-dimensional standard simplex, i.e. $S_n = \{(x_1, \ldots, x_n) \in \mathbb{R}^n \mid x_1, \ldots, x_n \geq 0,\ x_1 + \ldots + x_n < 1\}$, and let f be an absolute integrable function defined on S_n. The approximating value of the integral

$$\int \ldots \int_{S_n} f(x_1, \ldots, x_n)dx_1 \ldots dx_n \tag{1}$$

can be obtained using probabilistic interpolation of integral

$$I = \int \ldots \int_{S_n} f(x_1, \ldots, x_n)(n!\, dx_1 \ldots dx_n)$$

* *University of Cluj-Napoca, Faculty of Mathematics and Physics, 3400 Cluj-Napoca, Romania.*

as the expectation of random variable $f(X_1, \ldots, X_n)$, where $(X_1, \ldots, X_n)$, is an uniform distributed random vector over the simplex S_n.

If one considers the estimation function

$$\alpha_N = \frac{1}{N} \sum_{k=1}^{N} f(X_1^{(k)}, \ldots, X_n^{(k)}),$$

where $(X_1^{(k)}, \ldots, X_n^{(k)})$, $k = \overline{1, N}$, are independent and identically unform distributed random vectors over the simplex S_n, then α_N is an unbiased estimation function for the parameter I. The variance of α_N is σ_C^2/N, where σ_C^2 is the variance of $f(X_1, \ldots, X_n)$. Thus, α_N converges with probability one to I as $N \to \infty$. With this method, the *crude Monte Carlo method* named, we have

$$\int \ldots \int_{S_n} f(x_1, \ldots, x_n) dx_1 \ldots dx_n \approx \frac{1}{n!} \hat{\alpha}_N,$$

where

$$\hat{\alpha}_N = \frac{1}{N} \sum_{k=1}^{N} f(x_1^{(k)}, \ldots, x_n^{(k)}),$$

and $(x_1^{(k)}, \ldots, x_n^{(k)})$, $k = \overline{1, N}$, are uniform random number vectors over S_n.

2. In *method of control variates* the reducing of variance is obtained if the integral I is writed as $I = I_1 + I_2$, where

$$I_1 = \int \ldots \int_{S_n} [f(x_1, \ldots, x_n) - g(x_1, \ldots, x_n)](n!\, dx_1 \ldots dx_n),$$

and

$$I_2 = \int \ldots \int_{S_n} g(x_1, \ldots, x_n)(n!\, dx_1 \ldots dx_n).$$

The function g is selected such as to be theoretically integrated and to mimic the behaviour of function f. Thus, the estimation of integral I is reduced at the estimation of integral I_1.

In the following we consider that $g = B_m(f)$, where $B_m(f)$ is the Bernstein polynomial of global degree m relative to f:

$$B_m(f; x_1, \ldots, x_n) = \sum_{i_1=0}^{m} \sum_{i_2=0}^{m-i_1} \ldots \sum_{i_n=0}^{m-i_1-\ldots-i_{n-1}} f\left(\frac{i_1}{m}, \frac{i_2}{m}, \ldots, \frac{i_n}{m}\right) \cdot$$

$$\cdot \binom{m}{i_1} \binom{m-i_1}{i_2} \ldots \binom{m-i_1-\ldots-i_{n-1}}{i_n} x_1^{i_1} x_2^{i_2} \ldots x_n^{i_n} (1 - x_1 - \ldots - x_n)^{m-i_1-\ldots-i_n}.$$

Then we have that

$$I_2 = \binom{m+1}{n} \sum_{i_1=0}^{m} \ldots \sum_{i_n=0}^{m-i_1-\ldots-i_n-i_{n-1}} f\left(\frac{i_1}{m}, \ldots, \frac{i_n}{m}\right).$$

To evaluate the integral I_1 one considers the estimating function

$$\beta_N = \frac{1}{N} \sum_{k=1}^{N} e(X_1^{(k)}, \ldots, X_n^{(k)}),$$

where $c = f - B_m(f)$ and $(X_1^{(k)}, \ldots, X_n^{(k)})$, $k = \overline{1, N}$, are independent and identically uniform distributed random vectors over the simplex S_n. This estimating function is an unbiased estimating function for the parameter I_1. The variance of β_N is σ_V^2/N, with σ_V^2 the variance of $e(X_1, \ldots, X_n)$, where $(X_1, \ldots, X_n)$ is uniform random vector over the simplex S_n.

Using this method we have

$$\int \cdots \int_{S_n} f(x_1, \ldots, x_n)\, dx_1 \ldots dx_n \approx \frac{1}{n!} (\hat{\beta}_N + I_2),$$

with

$$\hat{\beta}_N = \frac{1}{N} \sum_{k=1}^{N} e(x_1^{(k)}, \ldots, x_n^{(k)}),$$

where $(x_1^{(k)}, \ldots, x_n^{(k)})$, $k = \overline{1, N}$, are uniform random number vectors over the simplex S_n.

To approximate the integral (1) with the method of control variates as well as with the crude Monte Carlo method it is necessary to generate the uniform random number vectors $(x_1^{(k)}, \ldots, x_n^{(k)})$, $k = \overline{1, N}$, over the simplex S_n. Rejection method can be used, but for large n this method is inefficiently because the rejection probability is also largely. In [6] was proposed a new method to generate uniform random number vectors over S_n and this method was compared with rejection method. The method presented in [6] follows from a result given in [2]. Namely, if $X_1, \ldots, X_{n+1}$ are independently, identic exponential distributed random variables ($\lambda = 1$), then the random vector $(Y_1, \ldots, Y_n)$ with the component $Y_i = X_i/(X_1 + \ldots + X_{n+1})$ is uniformly distributed over S_n. Taking into account this result following generate algorithm is given in [6]:

Step 1. Generate $u_1, \ldots, u_{n+1}$, uniformly over (0, 1),

Step 2. Calculate $y_i = \ln u_i$, $i = \overline{1, n+1}$,

Step 3. Calculate $s = y_1 + \ldots + y_{n+1}$,

Step 4. Calculate $x_i = y_i/s$, $i = \overline{1, n}$.

The vector $(x_1, \ldots, x_n)$ is uniformly random number vector over the simplex S_n.

3. *Method of importance sampling* consists to consider a new density function g that mimics the properties of function f and also to be simple because a sampling relative to this density function will be neccessary. Then we have

$$I = \int \cdots \int_{S_n} h(x_1, \ldots, x_n) g(x_1, \ldots, x_n) dx_1 \ldots dx_n,$$

where $h = n!f$

In the following we consider that $g = \tilde{B}_m(f)$, where $\tilde{B}_m(f)$ is the normalized Bernstein polynomial of global degree m relative to f, i.e. $\tilde{B}(f) = B(f)/T$, with

$$T = \int_{S_n} \dots \int B_m(f; x_1, \dots, x_n)\, dx_1 \dots dx_n =$$

$$= \frac{1}{(m+1)(m+2)\dots(m+n)} \sum_{i_1=0}^{m} \dots \sum_{i_n=0}^{m-i_1-\dots-i_{n-1}} f\left(\frac{i_1}{m}, \dots, \frac{i_n}{m}\right).$$

Taking into account that $B_m(f)$ is a positive linear operator we have that $\tilde{B}_m(f)$ is positively when $f > 0$. Alternatively, it is added an appropriate positive constant to function f. Futher on the positivity of function f is considered.

The estimation of integral I is given by estimating function

$$\gamma_N = \frac{1}{N} \sum_{k=1}^{N} h(X_1^{(k)}, \dots, X_n^{(k)}),$$

where $(X_1^{(k)}, \dots, X_n^{(k)})$, $k = \overline{1, N}$, are independent and identically distributed random vectors who have the common probability density function $\tilde{B}_m(f)$ on the simplex S_m. The estimating function γ_N is an unbiased estimating function for the parameter I and the variance of γ_N is σ_S^2/N, with σ_S^2 the variance of $h(X_1, \dots, X_n)$, where the random vector $(X_1, \dots, X_n)$ is $\tilde{B}_m(f)$ distributed over S_n. Hence, we have

$$\int_{S_n} \dots \int f(x_1, \dots, x_n)\, dx_1 \dots dx_n \approx \hat{\gamma}_N,$$

where

$$\hat{\gamma}_N = \frac{1}{N} \sum_{k=1}^{N} h(x_1^{(k)}, \dots, x_n^{(k)}) = \frac{T}{N} \sum_{k=1}^{N} \frac{f(x_1^{(k)}, \dots, x_n^{(k)})}{B_m(f; x_1^{(k)}, \dots, x_n^{(k)})}$$

with $(x_1^{(k)}, \dots, x_n^{(k)})$, $k = \overline{1, N}$, random number vectors, $\tilde{B}_m(f)$ distributed over S_n.

To generate a random number vector $(x_1, \dots, x_n)$, $\tilde{B}_m(f)$ distributed over S_n, one considers an urn which contains balls of $M = \binom{m+n}{n}$ colours denoted by $(i_1, \dots, i_n)$, $i_1 = \overline{0, m}$, $i_2 = \overline{0, m - i_1}$, $\dots$, $i_n = \overline{0, m - i_1 - \dots - i_{n-1}}$. Let $B_{(i_1, \dots, i_n)}$ be the event of drawing out a ball labeled by $(i_1, \dots, i_n)$ and the probability of this event

$$P(B_{(i_1, \dots, i_n)}) = f\left(\frac{i_1}{m}, \dots, \frac{i_n}{m}\right) \Big/ \sum_{i_1=0}^{m} \dots \sum_{i_n=0}^{m-i_1-\dots-i_{n-1}} f\left(\frac{i_1}{m}, \dots, \frac{i_n}{m}\right).$$

f a ball of colour $(i_1, \ldots, i_n)$ is drawing one considers random number vector $x_1, \ldots, x_n)$ which is sampling value vector with Dirichlet distribution $D(i_1+1, \ldots, i_n + 1;\ m - i_1 - \ldots - i_n + 1)$ [9]. This sampling value vector corresponds to random vector $(X_1, \ldots, X_n)$ with $\tilde{B}_m(f)$ probability density function over S_n. Namely, if the distribution function of random vector $(X_1, \ldots, X_n)$ is F and ρ is the corresponding probability density function then using the theorem on compound probabilities it results that

$$F(x_1, \ldots, x_n) = P(X_1 < x_1, \ldots, X_n < x_n) =$$

$$= \sum_{i_1=0}^{m} \cdots \sum_{i_n=0}^{m-i_1-\ldots-i_{n-1}} P(B_{(i_1, \ldots, i_n)}) P(X_1 < x_1, \ldots, X_n < x_n \mid B_{(i_1, \ldots, i_n)}) =$$

$$= \frac{1}{(m+1) \ldots (m+n)T} \sum_{i_1=0}^{m} \cdots$$

$$\cdots \sum_{i_n=0}^{m-i_1-\ldots-i_{n-1}} f\left(\frac{i_1}{m}, \ldots, \frac{i_n}{m}\right) P(X_1 < x_1, \ldots, X_n < x_n \mid B_{(i_1, \ldots, i_n)}).$$

But, for a selecting colour $(i_1, \ldots, i_n)$ the random vector has the probability density function

$$\rho_{(i_1, \ldots, i_n)}(x_1, \ldots, x_n) = \frac{(m+n)!}{i_1! \ldots i_n! (m - i_1 - \ldots - i_n)!} x_1^{i_1} \ldots x_n^{i_n} \times$$

$$\times (1 - x_1 - \ldots - x_n)^{m-i_1-\ldots-i_n}$$

over the simplex S_n. Hence

$$\rho(x_1, \ldots, x_n) = \frac{1}{(m+1) \ldots (m+h)T} \sum_{i_1=0}^{m} \cdots$$

$$\cdots \sum_{i_n=0}^{m-i_1-\ldots-i_{n-1}} f\left(\frac{i_1}{m}, \ldots, \frac{i_n}{m}\right) \cdot \rho_{(i_1, \ldots, i_n)}(x_1, \ldots, x_n) =$$

$$= \frac{1}{T} \sum_{i_1=0}^{m} \cdots \sum_{i_n=0}^{m-i_1-\ldots-i_{n-1}} f\left(\frac{i_1}{m}, \ldots, \frac{i_n}{m}\right) \times$$

$$\times \binom{m}{i_1}\binom{m-i_1}{i_2} \ldots \binom{m-i_1-\ldots-i_{n-1}}{i_n} \cdot x_1^{i_1} \ldots x_n^{i_n} (1 - x_1 - \ldots - x_n)^{m-i_1-\ldots-i_{n-1}},$$

therefore $\rho = \tilde{B}_m(f)$.

Using these results one gives an algorithm to generate a random number vector, $\tilde{B}_m(f)$ distributed:

Step 1. A correspondence one-to-one

$$r: \{1, 2, \ldots, M\} \to \{(i_1, \ldots, i_n) \mid i_1 = \overline{0, m}, \ldots, i_n = \overline{0, m-i-\ldots-i_{n-1}}\}$$

one defines, $M = \binom{m+n}{n}$.

Step 2. Calculate $p_k = P(B_{r(k)})$, $k = \overline{1, M}$.

Step 3. Generate uniform x over (0, 1).

Step 4. If $x \in [p_1 + \ldots + p_{j-1}, p_1 + \ldots + p_j)$, then generate Dirichlet $D(i_1 + 1, \ldots, i_n + 1; m - i_1 - \ldots - i_n + 1)$ random number vector $(x_1, \ldots, x_n)$ [7, 8], with $r(j) = (i_1, \ldots, i_n)$.

Step 5. $(x_1, \ldots, x_n)$ is $\tilde{B}_m(f)$ distributed.

5. In the following we consider an estimate of the integral I by combining the two methods to reduce of variance. In the first one applies the method of importance sampling then the method of control variates is applied. That is why one writes $I = I_3 + I_4$, where

$$I_3 = \int_{S_n} \ldots \int [h(x_1, \ldots, x_n) - B_m(h; x_1, \ldots, x_n)]\, B_m(f; x_1, \ldots, x_n) dx_1 \ldots dx_n$$

and

$$I_4 = \int_{S_n} \ldots \int B_m(h; x_1, \ldots, x_n) B_m(f; x_1, \ldots, x_n) dx_1 \ldots dx_n,$$

with the function h the same of the previous section, i.e. $h = n!\, f / B_m(f)$.

The integral I_4 can be calculated using the formula

$$\int_{S_n} \ldots \int B_m(v; x_1, \ldots, x_n) B_m(w; x_1, \ldots, x_n) dx_1 \ldots dx_n =$$

$$= \sum_{i_1=0}^{m} \ldots \sum_{i_n=0}^{m-i_1-\ldots-i_{n-1}} \sum_{j_1=0}^{m} \ldots \sum_{j_n=0}^{m-j_1-\ldots-j_{n-1}} v\left(\frac{i_1}{m}, \ldots, \frac{i_n}{m}\right) w\left(\frac{j_1}{m}, \ldots, \frac{j_n}{m}\right) \times$$

$$\times \frac{(n!)^2}{(2m+n)!} \binom{i_1 + j_1}{i_1} \ldots \binom{i_n + j_n}{i_n} \binom{2m - i_1 - \ldots - i_n - j_1 - \ldots - j_n}{m - i_1 - \ldots - i_n - j_1 - \ldots - j_n}.$$

To estimate the integral I_3 one considers the estimating function

$$\delta_N = \frac{1}{N} \sum_{k=1}^{N} u(X_1^{(k)}, \ldots, X_n^{(k)}),$$

where $u = h - B_m(h)$ and $(X_1^{(k)}, \ldots, X_n^{(k)})$, $k = \overline{1, N}$, are independently random vectors with common $\tilde{B}_m(f)$ probability density function. Of course, the function f must to be positively, otherwise it is incremented by a suitable constant. The estimating function δ_N is unbiased and it has the variance σ_{SV}^2/N, where σ_{SV}^2 is the variance of $u(X_1, \ldots, X_n)$, with $(X_1, \ldots, X_n)$ a random vector $B_m(f)$ distributed.

By this combinated scheme we have that

$$\int \cdots \int_{S_n} f(x_1, \ldots, x_n)dx_1 \ldots dx_n \approx \frac{1}{n!}(\hat{\delta}_N + I_4),$$

with

$$\hat{\delta}_N = \frac{1}{N}\sum_{k=1}^{N} u(x_1^{(k)}, \ldots, x_n^{(k)}),$$

where $(x_1^{(k)}, \ldots, x_n^{(k)})$, $k = \overline{1, N}$, are random number vectors $\tilde{B}_m(f)$ distributed over the simplex S_n. To generate these random number vectors one follows the algorithm presented in the previous section.

6. In this part one considers invers combining scheme than the scheme presented in the section five. After the application of the method of control variates, the importance sampling method is applied. We assume that $e = f - B_m(f)$ is nonnegative. Then the integral I is writed in the form $I = I_2 + I_5$, where I_2 is that from the section two and

$$I_5 = \int \cdots \int_{S_n} z(x_1, \ldots, x_n)\tilde{B}_m(e; x_1, \ldots, x_n)dx_1 \ldots dx_n,$$

with $z = n!\, e/\tilde{B}_m(e)$.

One takes the estimation function

$$\varepsilon_N = \sum_{k=1}^{N} z(X_1^{(k)}, \ldots, X_n^{(k)}),$$

where the random vectors $(X_1^{(k)}, \ldots, X_n^{(k)})$, $k = \overline{1, N}$, are independent and identically $\tilde{B}_m(e)$ distributed. The estimatin function ε_N is unbiased for the parameter I_5 and it has the variance σ_{CS}^2/N, with σ_{CS}^2 the variance of $z(X_1, \ldots, X_n)$, where $(X_1, \ldots, X_n)$ is a random vector with $\tilde{B}_m(e)$ probability density function over S_n. Hence we have

$$\int \cdots \int_{S_n} f(x_1, \ldots, x_n)dx_1 \ldots dx_n \approx \frac{1}{n!}(\hat{\varepsilon}_N + I_2),$$

where

$$\hat{\varepsilon}_N = \frac{1}{N}\sum_{k=1}^{N} z(x_1^{(k)}, \ldots, x_n^{(k)},$$

with $(x_1^{(k)}, \ldots, x_n^{(k)})$, $k = \overline{1, N}$, random number vectors, $\tilde{B}_m(e)$ distributed over the simplex S_n. To generate these random number vectors one proposes the algorithm from the fourth section, by e replacing the function f.

7. Numerical experiments have been performed by the two methods to reduce of variance and by the two combining schemes. It was considered the

bidimensional case ($n = 2$) and the integrand function $f(x, y) = 1/(1+x+y)$. The results are presented in Table 1. In all the four methods the Bernstein polynomials of global degree $m = 2, 3, 4$ were considered. The variances of the estimators were conputed by numerical methods using a romanian computer CORAL—4030 in double precision. In all the cases the variances were reported to variance of the crude Monte Carlo method.

Table 1

Type of scheme	$m=2$		$m=3$		$m=4$	
	(1)	(2)	(1)	(2)	(1)	(2)
Crude Monte Carlo	0.0096597	100.	0.0096597	100.	0.0096597	100.
Control variate	0.0002210	2.29	0.0001021	1.06	0.0000580	0,60
Importance sampling	0.0001364	1.41	0.0000635	0.66	0.0000362	0.37
Importance sampling-control variate	0.0000439	0.45	0.0000118	0.12	0.0000041	0.04
Control variate-importance sampling	0.0000505	0.52	0.0000129	0.13	0.0000045	0.05

(1) — variance, (2) — per cent of crude Monte Carlo method.

REFERENCES

1. Ermakov, S. M., *Metoda Monte Carlo şi probleme înrudite*, Ed. Tehnică, Bucureşti, 1976.
2. Feller, W., *An Introduction to Probability Theory and Its Applications*, vol. 2, Wiley, New York, 1966.
3. Hammersley, J. M., Handscomb, D. C., *Monte Carlo Methods*. Methuen—John Wiley, London—New York, 1964.
4. Lorentz, G. G., *Bernstein Polynomials*, University of Toronto Press, Toronto, 1953.
5. Rosenberg, L., *Bernstein Polynomials and Monte Carlo integration*, SIAM J. Numer. Anal. 4, No. 4, 1967, 566—574.
6. Ştefănescu, Şt., *Algoritmi pentru generarea de puncte uniform repartizate într-un simplex din spaţiul R^n*, Stud. Cerc. Mat. 37, No. 1, 1985, 83—93.
7. Văduva, I., *Modele de simulare cu calculatorul*, Ed. Tehnică, Bucureşti, 1977.
8. Văduva, I., *Computer generation of random vectors based on transformation of uniformly distributed vector*, in "Proceedings of the Seventh Conf. on Probability Theory" (Braşov, Romania, 1982), Ed. Acad. R.S.R., Bucureşti, 1984, pp. 589—598.
9. Wilks, S., *Mathematical Statistics (Russian)*, Izd. „Nauka", Moskva, 1967.

TUDIA UNIV. BABEŞ-BOLYAI, MATHEMATICA, XXXIII, 3, 1988

A SYSTOLIC ARRAY FOR NUMERICAL INTEGRATION BY USING THE TRAPEZOIDAL AND SIMPSON FORMULAS

OCTAV BRUDARU*

Received: November 12, 1987

REZUMAT. — **Tablou sistolic pentru integrarea numerică prin formulele trapezului şi Simpson.** În prezenta lucrare se prezintă un şir de implementare pentru formula trapezului şi a lui Simpson, care permite calculul integralelor de forma (1).

1. **Introduction.** The first papers on systolic computation [3], [4] have proved clearly that the systolic arrays could lead to devices that would have remarcable performances. Since then, the interest has never ceased and great progress has been made in this domain [6], [8].

The purpose of this paper is to present a systolic array implementing the trapezoidal and Simpson formulas, which is able to compute with a constant rate of time

$$I(k) = \int_{a(k)}^{b(k)} f_k(x)\,dx \tag{1}$$

$k = 1, \ldots, K$. We suppose that $f_k : [a(k), b(k)] \to R$ is given by an arithmetic expression, $k = 1, \ldots, K$. The proposed solution is based on the results concerning the design of systolic arrays dedicated to the pipelined computation of real functions given by arithmetic expressions ([1]).

In Section 2 we outline the trapezoidal and Simpson formulas. In Section 3 we give the systolic network implementing these formulas and analyse its performances.

2. **Outline of the methods.** For a fixed $k \in \{1, \ldots, K\}$, if applied to the integral (1), the trapezoidal rule [2, p. 594], [7, p. 107] becomes

$$I(k) = [f(k\,;\, x(k, 0)) + f(k\,;\, x(k, n(k))) + 2\sum_{i=1}^{n(k)} f(k\,;\, x(k, i))]h(k)/2 \tag{2}$$

where $f(k\,;\, x) = f_k(x)$, $x(k, i) = x(k, 0) + ih(k)$, $i = 0, \ldots, n(k)$,

$$x(k, 0) = a(k) \text{ and } h(k) = (b(k) - a(k))/n(k).$$

The Simpson's rule ([2, p. 596], [7, p. 108]) gives the following form to $I(k)$

$$I(k) = [f(k\,;\, x(k, 0)) + f(k\,;\, x(k, n(k)) + 4S_1(k) + 2S_2(k)]h(k)/3 \tag{3}$$

* *Universitatea „Al. I. Cuza", Seminarul Matematic „A. Myller", 6600 Iaşi, Romania.*

where $n(k) = 2m(k)$, $S_1(k) = f(k\,;\, x(k, 1)) + f(k\,;\, x(k, 3)) + \ldots + f(k\,;\, x(k, n(k) - 1))$, $S_2(k) = f(k, x(k, 2)) + f(k\,;\, x(k, 4)) + \ldots + f(k\,;\, x(k, n(k) - 2))$.

The type of formula required by the computation of $I(k)$ is given b $ty(k)$, where $ty(k) = 0$ ($ty(k) = 1$) if $I(k)$ is given by (2) ((3)), $k = 1, \ldots, K$

3. **The systolic network.** In [1] and here, the clock tick (CT) is the tim needed to execute a division or both a multiplication and an addition. Furthe the variable t designates the time which is a count of the number of CTs We suppose that each processor is active during every pulse number.

The systolic network implementing the trapezoidal and Simpson formulas denoted by SN, needs the processors depicted in Fig. 1.

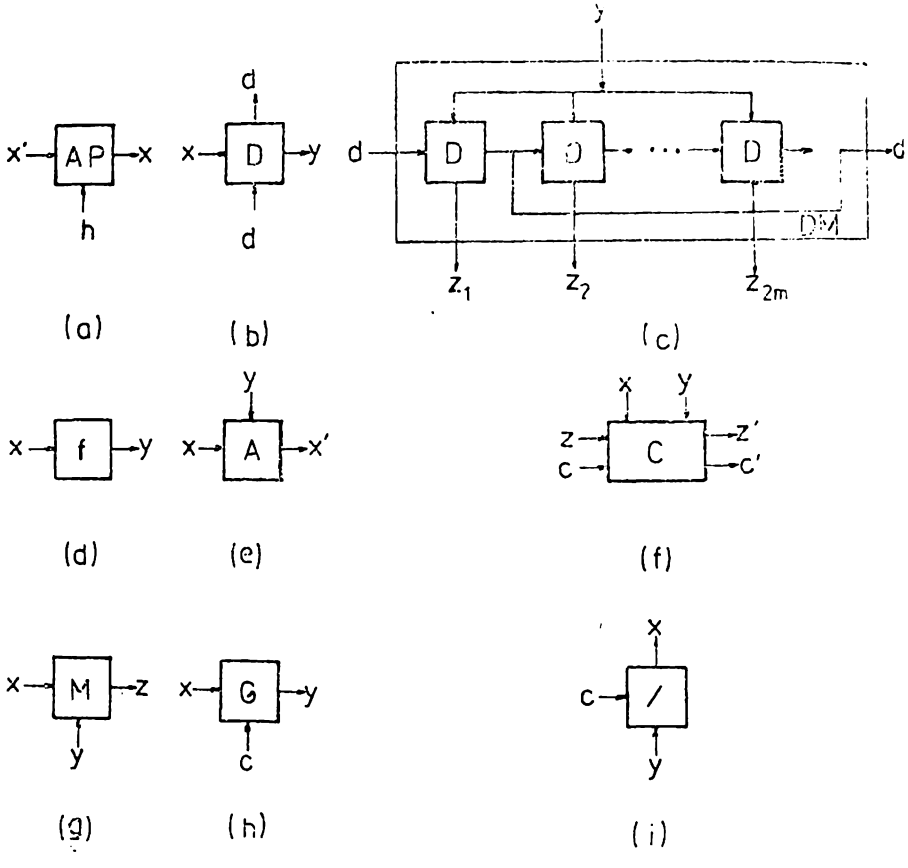


Fig. 1: The processors needed by the systolic network.

These processors are des ribed below.

a) The AP processor has the registers Rx and Rh which contain th values $x_0 - h$ and h, respectively. As soon as it receives '1' as a control sig

nal, it executes $x'(t+1) = Rx + Rh$, $Rx \leftarrow x'(t+1)$. Eventually, a single input can be used to load both Rx and Rh.

b) The D processor works so that $d'(t+1) = d(t)$ and if $d(t) = 1$ then $y(t+1) = x(t)$ otherwise y does not emit.

c) The cell DM is a demultiplexer and works so that $d'(t+1) = d(t)$ and if $d(t) = 1$ and $d(t+i) = 0$, then $z_i(t+i) = y(t+i-1)$ and z_k, $k \neq i$ does not emit at this time, $i = 1, \ldots, 2m+2$.

d) The subarray whose label is "f" accomplishes the computation $y(t+RT(f)) = f(x(t))$, where $RT(f)$ is the response time of a systolic array able to pipeline the computation of the function f and having 1 CT as period. Some techniques to design such an array for a given f are presented in [1].

e) The A processor performs $x'(t+1) = x(t) + y(t)$.

f) The C cell works so that $c'(t+2) = c(t)$ and if $c(t) = 1$ then $z'(t+2) = z(t) + 4x(t) + 2y(t+1)$ otherwise $(c(t) = 0)$ $z'(t+2) = z(t) + 2x(t) + 2y(t+1)$.

g) The M processor executes $z(t+1) = x(t)y(t)$.

h) The G processor work so that if $c(t) = 1$ then $y(t+1) = x(t)$, otherwise $(c(t) = 0)$ $y(t+1) = 0$.

i) This processor works so that if $c(t) = 0$ then $x(t+1) = y(t)/2$ and for $c(t) = 1$, $x(t+1) = y(t)/3$.

The entire SN network is presented in Fig. 2. Let us analyse the manner in which the own activity of SN and the $I/0$ operations are syncronised.

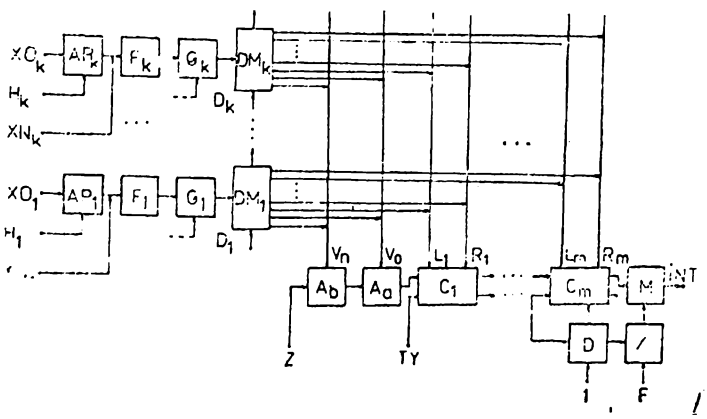


Fig. 2: Systolic implementation of trapezoidal and Simpson formulas.

If L is the label of an input (output) of a processor in SN, then let us denote by $L(t)$ the value entering (entering from) that one. Let us denote by t_k the moment in whcih $I(k)$ emerges from INT, $k = 1, \ldots, K$. Because SN has one CT as period it results that $t_k = t_0 + k$, where t_0 will be determined by an initial condition.

Let us consider a fixed $k \in \{1, \ldots, K\}$. From $\mathrm{INT}(t_k) = I(k)$ we must have $F(t_k - 2) = h(k)$. Because of the activity of G_k, which sends zeros as soon as the last value of f_k was sent by F_k, and by supposing $n(k) < 2m+1$,

we can take, without loss of generality, $f(k; x(k, j)) = 0$, $j = n(k) + 1, \ldots, 2m + 2$. On the other hand, let us observe that the c-input of C_{m-1} receives the values of $ty(k)$ during the $t_k - 3\,CT$ and therefore this signal reaches the division proccesor at the right time $(t_k - 2)$.

The pipe of serially connected processors $A_b, A_a, C_1, \ldots, C_m$ computes the value of the sum in brackets of (2) or (3), say SUM(k). This sum enters as zero A_b, accumulates the terms $f(k; x(k, n(k)), f(k; x(k, 0))$ and enters the z-input of C_1. Now, C_1 must know which formula must be applied. Because SUM(k) goes through $2m + 2$ delays before emerging from C_m during $t_k - 1C_i$ it results that $Z(t_k - 2m - 3) = 0$. Thus, $TY(t_k - 2m - 1) = ty(k)$ and the value goes to the right of the pipe together $S(k)$ indicating to each C-cell its left vertical input is multiplied by 2 (if $c(k) = 0$) or by 4 (if $c(k)=1$).

Also, we have that $V_b(t_k - 2m - 3) = f(k; x(k, n(k)))$, $V_a(t_k - 2m - 2) = f(k; x(k, 0))$, $L_i(t_k - 2m - 3 + 2i) = f(k; x(k, 2i - 1))$ and $R_i(t_k - 2m - 3 + 2i + 1) = f(k; x(k, 2i))$, $i = 1, \ldots, m$.

The first computed value of f_k (i.e. $f(k; x(k, n(k)))$ enters V_b during the $t - 2m - 3\,CT$s. It results that this value must enter DM_k at $t_k - 2m - 4\,CT$. From $D_k(t_k - 2m - 4) = 1$ we obtain for $k = 1$ that $D_1(t_0 - 2m - 3) = 1$. On the other hand, let us observe that the array F_k sends $f(k; x(k, n(k)))$ to G_k at $t_k - 2m - 5$ and therefore $XN_k(t_k - 2m - 5 - RT(f_k)) = x(k, n(k))$.

Clearly, AP_k begins its activity at $t_k - 2m - 4 - RT(f_k)\,CT$ and emits $x(k, 0), x(k, 1), \ldots, x(k, n(k) - 1)$ while it receives $n(k)$ true values on the control path. It results that $x(k, 0)$ and $h(k)$ must be loaded in AP_k no later than $t_k - 2m - 5 - RT(f_k)$.

Consequently, the moments to send the first values through the inputs XN_k, $k = 1, \ldots, K$, D_1, Z, TY and F, are $S(XN_k) = t_0 + k - 2m - 5 - RT(f_k)$, $k = 1, \ldots, K$, $S(D_1) = t_0 - 2m - 3$, $S(Z) = t_0 - 2m - 2$, $S(TY) = t_0 - 2$ and $S(F) = t_0 - 1$, respectively.

We obtain t_0 by taking

$$\min(\{S(D_1), S(Z), S(TY), S(F)\} \cup \{S(XN_k)/k = 1, \ldots, K\}) =$$

$$= \min(\{S(D_1)\} \cup \{S(XN_k)/k = 1, \ldots, K\}) = 0.$$

As a consequence of the above analysis, we can state the following.

THEOREM. *If* $D_1(t_0 - 2m - 3) = 1$, $D_1(t_0 - 2m - 3 + j) = 0$, $j > 1$, $XN_k(t_k - 2m - 5 - RT(f_k)) = x(k, n(k))$, $Z(t_k - 2m - 3) = 0$ *and* $TY(t_k - 2m - 1) = ty(k)$, $k = 1, \ldots, K$ *then* $\mathrm{INT}(t_k) = I(k)$, $k = 1, \ldots, K$ *where* $t_k = t_0 + k = 1, \ldots, K$ *and* t_0 *is given by* (4).

Let us remark that the entire processing takes $t_0 + K\,CT$s as compared with the time needed by an usual sequential algorithm. Observe that the time could be reduced if the arrays F_k, $k = 1, \ldots, K$ are arranged so that $RT(f_k) < RT(f_{k+1})$, $k = 1, \ldots, K - 1$.

In order to save pins, a single input may be used instead of XO_k, F and XN_k, by using this input to send the corresponding input value during three succesive CTs, $k = 1, \ldots, K$. If $f_k = f$, $k = 1, \ldots, K$, then SN becomes more regular, the control path and the executing of the reset comand are more simple.

Let us remark that following the definition given in [5], the presented is a systolic one. On the other hand, the design uses simple processing elements, which are locally connected. The $I/0$ operations are made by the processors placed on the boundary of the array. The proposed network can be modulary extended in order to handle a larger value of K.

The necessity to accomodate a larger value of M can be avoided by splitting the interval $[a(k), b(k)]$ in (1) into smaller ones by preserving $h(k)$, although SN could be extended by adding simple C-cells at the right end of the pipe and modifying the connections between the D-processors in DM_k, $k = 1, \ldots, K$. For this reason the A-cells were placed on the left end of the pipe. Such an extension could be made more easy by using only inner product step processors instead of the existing processors, but the last ones are more complicated then first ones.

Also, this design satisfies the basic features of a systolic array as they are given in [6].

The proposed network can be modified or extended in order to implement some new formulas, but this is a topic for another work.

REFERENCES

1. Brudaru, O., *Systematic Synthesis of Systolic Arrays for Some Real Functions Computation*, Tech. Rep. no. 2, Computer Centre, Politechnical Inst. of Iaşi, February 1987.
2. Demidovitch, B., Maron, I., *Elements de calcul numerique*, Edition Mir, Moscou, 1973.
3. Foster, M., Kung, H. T., *Design of Special-Purpose VLSI Chips. Example and Opinions*, Proc. 7th Internat. Symposium on Computer Architectures, May 1980.
4. Kung, H., Leiserson, C., *Systolic Arrays (for VLSI)*, Proc Sparse Matrix Proc., 1978, Society for Industrial and applied Mathematics, 1978, 256–282.
5. Leiserson, C. E., Saxe, J. B., *Optimizing Synchronous Circuitry by Retiming*, Journal of VLSI and Computer Systems 1, 1983, 41–67.
6. Quinton, P., *An Introduction to Systolic Architectures*, Future Parallel Computers, LNCS, (272), P. Treleaven, M. Vanneschi (eds.), Springer–Verlang, 1986, 387–400.
7. Scheid, F., *Theory and Problems of Numerical Analysis*, Schaum Outline Series, McGraw Hill, New York, 1968.
8. Schreiber, R., *A Survey of Systolic Computation*, Research Report, Computer Science Departement, Rensselaer Polytechnic Institute Troy, New York, 1986.

STUDIA UNIV. BABEŞ-BOLYAI, MATHEMATICA, XXXIII, 3, 1988

SIMULTANEOUS CONSTRUCTION OF SIMILAR MATRICES AND THE SIMILARITY TRANSFORMATION

ALEXANDER ABIAN*

Received: July 15, 1988

REZUMAT. — Construirea simultană a matricilor similare şi transformarea de similaritate. Fie M o matrice pătratică. Se dă o metodă pentru construirea simultană a matricilor E şi D pentru care $M = EDE^{-1}$. Metoda poate fi folosită pentru a construi o matrice D avînd anumite proprietăţi dorite.

The entries of all matrices are from a field, say, the real numbers.

Let matrix $M = \begin{pmatrix} 2 & 1 & 2 \\ 0 & 0 & -4 \\ 0 & 1 & 4 \end{pmatrix}$ be given. With M let us consider the following configuration:

	c_1	c_2	c_3			
	1	0	0			
	0	1	0			
	0	0	1			
r_1	2	1	2	1	0	0
r_2	0	0	−4	0	1	0
r_3	0	1	4	0	0	1

(1)

where on the top and to the right of M the unit 3 by 3 matrix is placed

We call the first three columns c_1, c_2, c_3 the *columns of the configuration*. We call the last three rows r_1, r_2, r_3 the *rows of the configuration.*

In (1), let us add to the second row the product of the third row by 2. We indicate this operation by:

$$2r_3 \rightarrow r_2 \tag{2}$$

and we record the result of operation (2) performed on (1) as follows:

1	0	0			
0	1	0	$2r_3 \rightarrow r_2$		
0	0	1			
2	1	2	1	0	0
0	0	−4	0	1	0
0	1	4	0	0	1

(3)

* *Iowa State University Ames, Department of Mathematics, Iowa 50011, U.S.A*

yields

$$\left(\begin{array}{ccc|ccc} 1 & 0 & 0 & & & \\ 0 & 1 & 0 & & & \\ 0 & 0 & 1 & & & \\ \hline 2 & 1 & 2 & 1 & 0 & 0 \\ 0 & 2 & 4 & 0 & 1 & 2 \\ 0 & 1 & 4 & 0 & 0 & 1 \end{array}\right) \tag{4}$$

We immediately follow operation (2) by its dual performed on (4). By the *dual* of (2), we mean "adding to the third column the product of the second column by -2". We indicate the operation dual of (2) by:

$$-2c_2 \rightarrow c_3 \tag{5}$$

and we record the result of operation (5) performed on (4) as follows:

$$\left(\begin{array}{ccc|c} 1 & 0 & 0 & \\ 0 & 1 & 0 & -2c_2 \rightarrow c_3 \\ 0 & 0 & 1 & \\ \hline 2 & 1 & 2 & 1\ 0\ 0 \\ 0 & 2 & 4 & 0\ 1\ 2 \\ 0 & 1 & 4 & 0\ 0\ 1 \end{array}\right) \tag{6}$$

yields

$$\left(\begin{array}{ccc|ccc} 1 & 0 & 0 & & & \\ 0 & 1 & -2 & & & \\ 0 & 0 & 1 & & & \\ \hline 2 & 1 & 0 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 & 1 & 2 \\ 0 & 1 & 2 & 0 & 0 & 1 \end{array}\right) \tag{7}$$

We observe that after performing in succession operation (2) and its dual (5) on (1), we obtain configuration (7) with the following properties:

In (7) *the top square matrix is the inverse of the right square matrix.* (8)

In (7) *the product of the top and left and right square matrices (in this order) is equal to the original left square matrix in* (1). (9)

Indeed, it is easy to verify that

$$\begin{pmatrix} 2 & 1 & 2 \\ 0 & 0 & -4 \\ 0 & 1 & 4 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & -2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 2 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{pmatrix} \tag{10}$$

Clearly, (10) describes a similarity transformation since it is of the form $M = PAP^{-1}$.

From (3) to (7) it follows that our method produces a matrix similar to the originally given matrix M simultaneously with the corresponding similarity

transformation. Moreover, as (10) shows, the similar matrix thus obtained i of a more desirable form than M inasmuch as it has more zeros than M.

Let us continue by performing another operation followed by its dua on (7) with the aim of constructing a matrix similar to M of a simpler form yet.

For instance, in (7), let us add to the first column the third column. W indicate this operation by:

$$c_3 \mapsto c_1 \tag{11}$$

and we record the result of operation (11) performed on (7) as follows:

$$\left.\begin{array}{ccc|c} 1 & 0 & 0 & \\ 0 & 1 & -2 & c_3 \mapsto c_1 \\ 0 & 0 & 1 & \end{array}\right| \left.\begin{array}{ccc} & & \\ & & \\ & & \end{array}\right| \tag{12}$$

$$\left.\begin{array}{ccc|ccc} 2 & 1 & 0 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 & 1 & 2 \\ 0 & 1 & 2 & 0 & 0 & 1 \end{array}\right.$$

yields

$$\left.\begin{array}{ccc|c} 1 & 0 & 0 & \\ -2 & 1 & -2 & \\ 1 & 0 & 1 & \end{array}\right| \left.\begin{array}{ccc} & & \\ & & \\ & & \end{array}\right| \tag{13}$$

$$\left.\begin{array}{ccc|ccc} 2 & 1 & 0 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 & 1 & 2 \\ 2 & 1 & 2 & 0 & 0 & 1 \end{array}\right.$$

We immediately follow operation (11) by its dual performed on (13). Th dual of (11) is: "adding to the thrid row the product of the first row by -1" We indicate the operation dual of (11) by:

$$-r_1 \mapsto r_3 \tag{14}$$

and we record the result of operation (14) performed on (13) as follows:

$$\left.\begin{array}{ccc|c} 1 & 0 & 0 & \\ -2 & 1 & -2 & -r_1 \mapsto r_3 \\ 1 & 0 & 1 & \end{array}\right| \left.\begin{array}{ccc} & & \\ & & \\ & & \end{array}\right| \tag{15}$$

$$\left.\begin{array}{ccc|ccc} 2 & 1 & 0 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 & 1 & 2 \\ 2 & 1 & 2 & 0 & 0 & 1 \end{array}\right.$$

yields

$$\left.\begin{array}{ccc|c} 1 & 0 & 0 & \\ -2 & 1 & -2 & \\ 1 & 0 & 1 & \end{array}\right| \left.\begin{array}{ccc} & & \\ & & \\ & & \end{array}\right| \tag{16}$$

$$\left.\begin{array}{ccc|ccc} 2 & 1 & 0 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 & 1 & 2 \\ 0 & 0 & 2 & -1 & 0 & 1 \end{array}\right.$$

We observe again that in configuration (16) the two properties described by (8) and (9) prevail. Indeed,

$$\begin{pmatrix} 2 & 1 & 2 \\ 0 & 0 & -4 \\ 0 & 1 & 4 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & -2 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 2 \\ -1 & 0 & 1 \end{pmatrix} \tag{17}$$

Clearly, (17) describes a similarity transformation since it is of the form $M = EDE^{-1}$.

So far, for the simultaneous construction of similar matrices and the coresponding similarity transformations we have used transformations of the form :

Adding to r_j row the product of r_i row by any real number s followed by adding to c_i column the product of c_j column by $-s$ where $i \neq j$. (18)

We symbolize (18) as follows:

$$sr_i \mapsto r_j \text{ followed by } -sc_j \mapsto c_i \text{ with } i \neq j \text{ and } s \text{ any real.} \tag{19}$$

Remark 1. In (18) as well as (19), the order in which an operation and its dual are performed is immaterial, i.e., (19) is equivalent to:

$$-sc_j \mapsto c_i \text{ followed by } sr_i \mapsto r_j \text{ with } i \neq j \text{ and } s \text{ any real} \tag{20}$$

For some theoretical reasons [2, p. 147] matrix M (appearing on the left side of the equality sign in (17)) cannot be similar to a matrix which is of a simpler form than matrix D (appearing as the middle matrix on the right side of the equality sign in (17)). In fact, D is the so-called Jordan canonical form [1, p. 17] of M.

Accordingly, no further entry of matrix D in (17) can be reduced to 0 by a further application of operations (19) or (20) on the configuration (16).

However, there are two other pairs of operations which can be performed on (16) and which yield matrices similar to D with the following properties. The result of one of the pairs of operations is to change the position of 1 appearing in D. The result of the other pair of operations is to replace 1 appearing in D by any nonzero real number h.

The above two pairs of operations are:

Exchanging r_i row with r_j row followed by exchanging c_i column with c_j column. (21)

and

Multiplying r_i row by a nonzero real number h followed by diviing c_i column by h (22)

The pairs of operations (21) and (22) are symbolized respectively as:

$$r_i \leftrightarrow r_j \text{ followed by } c_i \leftrightarrow c_j \tag{23}$$

$$r_i \to hr_i \text{ followed by } c_i \to \frac{1}{h} c_i \text{ with } h \neq 0 \tag{24}$$

It can be readily verified that Remark 1 is also applicable to (23) and (24).

Let us apply an instance of (23) on (26), e.g.,

$$r_1 \leftrightarrow r_3 \textit{ followed by } c_1 \leftrightarrow c_3$$

We record the results as follows:

$$\left.\begin{array}{ccc|c} 1 & 0 & 0 & \\ -2 & 1 & -2 & r_1 \leftrightarrow r_3 \\ 1 & 0 & 1 & \end{array}\right| \qquad \left.\begin{array}{ccc} 2 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{array}\right| \quad \left.\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 1 & 2 \\ -1 & 0 & 1 \end{array}\right|$$

yields

$$\left.\begin{array}{ccc|} 1 & 0 & 0 & \\ -2 & 1 & -2 & \\ 1 & 0 & 1 & \end{array}\right| \qquad \left.\begin{array}{ccc} 0 & 0 & 2 \\ 0 & 2 & 0 \\ 2 & 1 & 0 \end{array}\right| \quad \left.\begin{array}{ccc} -1 & 0 & 1 \\ 0 & 1 & 2 \\ 1 & 0 & 0 \end{array}\right|$$

and

$$\left.\begin{array}{ccc|c} 1 & 0 & 0 & \\ -2 & 1 & -2 & c_1 \leftrightarrow c_3 \\ 1 & 0 & 1 & \end{array}\right| \qquad \left.\begin{array}{ccc} 0 & 0 & 2 \\ 0 & 2 & 0 \\ 2 & 1 & 0 \end{array}\right| \quad \left.\begin{array}{ccc} -1 & 0 & 1 \\ 0 & 1 & 2 \\ 1 & 0 & 0 \end{array}\right|$$

yields

$$\left.\begin{array}{ccc|} 0 & 0 & 1 & \\ -2 & 1 & -2 & \\ 1 & 0 & 1 & \end{array}\right| \qquad \left.\begin{array}{ccc} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 2 \end{array}\right| \quad \left.\begin{array}{ccc} -1 & 0 & 1 \\ 0 & 1 & 2 \\ 1 & 0 & 0 \end{array}\right|$$

We observe that, as expected, the two properties (8) and (9) prevail in (26). Indeed,

$$\begin{pmatrix} 2 & 1 & 2 \\ 0 & 0 & -4 \\ 0 & 1 & 4 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 \\ -2 & 1 & -2 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 2 \end{pmatrix} \begin{pmatrix} -1 & 0 & 1 \\ 0 & 1 & 2 \\ 1 & 0 & 0 \end{pmatrix}$$

We also observe that, as mentioned above, the result of operations performed on (16) is to change the position of 1 in the matrix appearing

he middle on the right side of the equality sign in (17). Clearly, (27) also lescribes a similarity transformation.

Finally, we apply an instance of (24) on (26), e.g.,

$$r_3 \to 2r_3 \text{ followed by } c_3 \to \frac{1}{2} c_3 \tag{28}$$

We record the results as follows:

$$\begin{array}{ccc|c} 0 & 0 & 1 & \\ -2 & 1 & -2 & r_3 \to 2r_3 \\ 1 & 0 & 1 & \\ \hline 2 & 0 & 0 & -1\ 0\ 1 \\ 0 & 2 & 0 & 0\ 1\ 2 \\ 0 & 1 & 2 & 1\ 0\ 0 \end{array}$$

yields

$$\begin{array}{ccc|ccc} 0 & 0 & 1 & & & \\ -2 & 1 & -2 & & & \\ 1 & 0 & 1 & & & \\ \hline 2 & 0 & 0 & -1 & 0 & 1 \\ 0 & 2 & 0 & 0 & 1 & 2 \\ 0 & 2 & 4 & 2 & 0 & 0 \end{array}$$

and

$$\begin{array}{ccc|ccc} 0 & 0 & 1 & & & \\ -2 & 1 & -2 & c_3 \to \frac{1}{2} c_3 & & \\ 1 & 0 & 1 & & & \\ \hline 2 & 0 & 0 & -1 & 0 & 1 \\ 0 & 2 & 0 & 0 & 1 & 2 \\ 0 & 2 & 4 & 2 & 0 & 0 \end{array}$$

yields

$$\begin{array}{ccc|ccc} 0 & 0 & 1/2 & & & \\ -2 & 1 & -1 & & & \\ 1 & 0 & 1/2 & & & \\ \hline 2 & 0 & 0 & -1 & 0 & 1 \\ 0 & 2 & 0 & 0 & 1 & 2 \\ 0 & 2 & 2 & 2 & 0 & 0 \end{array}$$

We rewrite the similarity transformation indicated by (28) in the usual way:

$$\begin{pmatrix} 2 & 1 & 2 \\ 0 & 0 & -4 \\ 0 & 1 & 4 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1/2 \\ -2 & 1 & -1 \\ 1 & 0 & 1/2 \end{pmatrix} \begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 2 & 2 \end{pmatrix} \begin{pmatrix} -1 & 0 & 1 \\ 0 & 1 & 2 \\ 2 & 0 & 0 \end{pmatrix} \tag{29}$$

As mentioned above, and as shown by (29), the result of operations (28 performed on (26) is to replace 1, appearing in the middle matrix on the righ side of the equality sign in (27), by 2.

Remark 2. In carrying out our method, there is no need to repeat som of the configurations. For instance, the repetition (3) of (1) and (6) of (4 etc. are unnecessary since they can be combined. We have repeated thes configrations for the sake of clarity of our exposition.

Remark 3. We recall [2, p. 147] that a matrix is called *tridiagonal* (o of Jacobi type) if all of its entries below the first subdiagonal and above th first superdiagonal are zero.

Using our method, it can be readily verified that any square matri M can be made similar to a tridiagonal matrix T. In fact, by repeated (i necessary) applications of (19), (23), (24), our method constructs simultaneoly matrices T, H, H^{-1} such that $M = HTH^{-1}$.

Remark 4. We observe that throughout the entire process of tridiagon lization of a square matrix M by our method, we do not need the charact ristic polynomial of M nor do we need the eigenvalues or eigenvectors of M But then, precisely for this reason, we cannot expect that our method woul yield the Jordan canonical form [1, p. 17] of a square matrix without requiri to solve some polynomial equations in order to make judicious choices for in (19). The fact that the Jordan canonical form was obtained in (17) wit out any reference to the characteristic polynomial of M was quite accidenta

For instance, let us try to contruct the similarity matrix A given by

$$A = \begin{pmatrix} 5 & 6 \\ -2 & -2 \end{pmatrix} \qquad (3$$

which would yield the Jordan canonical form of A. Clearly, A is already the tridiagonal form. Let us try to replace 6 in A by 0 through operation of type (19). The tempting choice of $s = 3$ in (19) for the operation $3r_2 \mapsto r$ would replace 6 in A by 0 but then the dual operation $-3c_1 \mapsto c_2$ would re place that 0 by 3. So, the choice of s must be made more judiciously. To this end, we perform operations (19) on our initial configuration involving and then determine the suitable s. Accordingly, we have:

$$\begin{array}{cc|c} 1 & 0 & \\ 0 & 1 & sr_2 \mapsto r_1 \\ \hline 5 & 6 & 1 \quad 0 \\ -2 & -2 & 0 \quad 1 \end{array}$$

yields

$$\begin{array}{cc|cc} 1 & 0 & & \\ 0 & 1 & & \\ \hline 5-2s & 6-2s & 1 & s \\ -2 & -2 & 0 & 1 \end{array}$$

yields

$\begin{matrix}1 & 0\\ 0 & 1\end{matrix}$		$-sc_1 \mapsto c_2$
$\begin{matrix}5-2s & 6-2s\\ -2 & -2\end{matrix}$	$\begin{matrix}1 & s\\ 0 & 1\end{matrix}$	

yields

$\begin{matrix}1 & 0\\ 0 & 1\end{matrix}$		$-sc_1 \mapsto c_2$
$\begin{matrix}5-2s & 6-2s\\ -2 & -2\end{matrix}$	$\begin{matrix}1 & s\\ 0 & 1\end{matrix}$	

yields

$\begin{matrix}1 & -s\\ 0 & 1\end{matrix}$		
$\begin{matrix}3-2s & 2s^2-7s+6\\ -2 & -2+2s\end{matrix}$	$\begin{matrix}1 & s\\ 0 & 1\end{matrix}$	

(31)

Now, we determine s in (31) in such a way that $2s^2 - 7s + 6 = 0$. Thus, we have to solve a quadratic equation. A root of the equation is 2. Substituting $s = 2$ in (31) we obtain (32). Next, we perform operations (19) on (32) in such a way that the newly obtained 0 is not affected and we determine s so that -2 in the lower left 2 by 2 matrix in (32) is replaced by 0. Combining operations (19) in one configuration, we have:

$\begin{matrix}1 & -2\\ 0 & 1\end{matrix}$		$\begin{matrix}sr_1 \mapsto r_2\\ -sc_2 \mapsto c_1\end{matrix}$
$\begin{matrix}1 & 0\\ -2 & 2\end{matrix}$	$\begin{matrix}1 & 2\\ 0 & 1\end{matrix}$	

(32)

yields

$\begin{matrix}1+2s & -2\\ -s & 1\end{matrix}$		
$\begin{matrix}1 & 0\\ -2-s & 2\end{matrix}$	$\begin{matrix}1 & 2\\ s & 2s+1\end{matrix}$	

(33)

We determine s in (33) in such a way that $-2-s=0$. The root of this linear equation is -2. Substituting $s = -2$ in (33) we obtain:

$\begin{matrix}-3 & -2\\ 2 & 1\end{matrix}$		
$\begin{matrix}1 & 0\\ 0 & 2\end{matrix}$	$\begin{matrix}1 & 2\\ -2 & -3\end{matrix}$	

(34)

which simultaneously describes the similarity transformation and the resultin Jordan canonical form of matrix A given by (30). We rewrite (34) in th usual way:

$$\begin{pmatrix} 5 & 6 \\ -2 & -2 \end{pmatrix} = \begin{pmatrix} -3 & -2 \\ 2 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ -2 & -3 \end{pmatrix} \tag{35}$$

As mentioned above, we obtained (35) without determining the characte- ristic equation or the eigenvalues or the eigenvectors of matrix A given by (30). However, we had to solve a quadratic equation. Obviously, (35) provides us with the eigenvelues 1 and 2 of A and with their corresponding eigen- vectors $(-3, 2)$ and $(-2, 1)$.

REFERENCES

1. Householder, A. S., *The Theory of Matrices in Numerical Analysis*, Dover Publ. Inc. N. Y., 1964.
2. Gourlay, A. R. and Watson, G. A., *Computational Methods for Matrix Eigenproblems* John Wiley and Sons, N.YI., 1973.

STUDIA UNIV. BABEŞ-BOLYAI, MATHEMATICA, XXXIII, 3, 1988

ON THE EXTENSIBILITY OF PROGRAMMING LANGUAGES IN DYNAMIC MEANING

ILIE PARPUCEA*

Received: July 14, 1988

REZUMAT. — **Cu privire la extensibilitatea limbajelor de programare în sens dinamic.** Noţiunea de extensibilitate în limbajul de programare se îmbogăţeşte cu noi elemente. Extensibilitatea dinamică permite ridicarea gradului de elasticitate în programare.

1. Introduction. The programming medium constitutes a relatively new notion in the field of computer programming, largely embraced by the projectors of economic, scientific and social applications. In the following we shall mean by programming medium an association of software components allowing:

— computer programming in a specific programming language;

— possibilities of edition for programs and documentary texts;

— facilities of high level program repair (adjusting), without resorting to the assembling language specific to the computer;

— compatibility ensuring with other programming media, with respect to data organization and management.

These components will act as an entity, ensuring the management and the optimum partition of the computer resources. The user must dispose of a complete set of instructions allowing the optimum exploitation of the programming facilities.

The problem of programming language extensibility appears to be needed by the increase of programming efficiency and the diversification of the programming facilities at programmer's disposal. The methods of language extension known till now allow a so-called extensibility in static meaning. This denomination points out the fact that the grammar as model of syntactical specification is fixed to the soft product implementation; the user cannot operate modifications (adaptation to the programming real needs).

2. Dynamic Extensibility with Algebraic Specification of Programming Languages. The extensibility of programming languages at the grammatical level does not constitute a new problem. But the mode of implementation of this one on concrete cases is still far from exploitating the offered advantages. This extension mode constitutes a complex problem, knottily to apply.

The adequate development of a mathematical apparatus concerning the HAS hierarchy has for purpose the necessity of creating a formal mechanism for specifying a concept of abstract calcultion system, structurally developed

* *University of Cluj-Napoca, Computing Data Center, 3400 Cluj-Napoca, Romania.*

in order to be considered as semantics for a programming language [1]. We shall mean by formal specification of an abstract calculation system the elaboration of an algebraic model for the respective system and a symbolism allowing the representation of the calculation concepts implicated in the abstract calculation system.

The concept of abstract calculation system as support for the semantics of a programming language is specified by means of the heterogeneous algebra mechanism, under the form of the algebraic structure:

$$\mathcal{A} = \{D = (D_{pi})_{i\in I} \cup (D_{cj})_{j\in J},\ \Sigma S = (\Sigma S(i))_{i\in I} \cup (\Sigma S(j))_{j\in J},\ F\},$$

where:

$(D_{pi})_{i\in I}$ = the set of primitive calculation objects;

$(D_{cj})_{j\in J}$ = the set of composed calculation objects;

ΣS = the operator scheme formed by the primitive and composed operation schemes;

F = the symbol of the function which associates to each operation scheme σ from HAS(i) a heterogeneous operation $F(\sigma)$ specific to HAS($i+1$).

The behaviour features of the operation schemes are specified for each one under the form of formal identities.

The objects of abstract calculation system are represented as formal expressions, organized as a heterogeneous algebra of words, of the following form

$$\mathfrak{W} = \{W = (W_i)_{i\in I} \cup (W_j)_{j\in J},\ \Sigma S = (\Sigma S(i))_{i\in I} \cup (\Sigma S(j))_{j\in J},\ F\},$$

where the significance of the notations is similar to that of the notations from the support algebra for the semantics [1].

The notion of dynamic extensibility imposes a dynamic character to the algebrae $\mathcal{A}$ and $\mathfrak{W}$, allowing the definition of new semantic forms which enrich the algebra $\mathcal{A}$ (a collection of semantic forms). According to these semantic forms, taking into account their representation, the semantic correspondent which will enrich the algebra $\mathfrak{W}$ will be automatically generated.

An estimate morphism $f: \mathfrak{W} \to \mathcal{A}$ is inductively defined between $\mathfrak{W}$ and $\mathcal{A}$. Every element $w \in \mathfrak{W}$ represents in the programming language associated for specification either a program, instruction, set of instructions between begin and end, or a calculation process. An element w is of the form $w_1w_2\ldots w_no$ structured on depth levels [1].

The estimate process by means of the morphism f involves a detailed analysis of w, the separation of the component parts at subword level to the free generator level. In other words, an analysis is performed in order to identify all the syntactic components. After these ones are established, one associates to each syntactic component the semantic correspondent from $\mathcal{A}$. This correspondence being established, it will imply the verification of the placement of the syntactic form between the limits of the semantic formalism. We recall that the semantic formalism imposes certain restrictions to the actual parameters of the semantic formalism.

Taking into account the significance and the representation form of w, the above defined estimate morphism f has a recursive character. The esti-

mate of w and the establishment of the component syntactic forms will make the estimate process to be taken again, corresponding to the depth levels of the subwords w.

For $w \in \mathfrak{W}$ and $w = w_1 w_2 \ldots w_n$, we have $f(w) = f(x_1) \circ f(w_2) \circ \ldots \circ f(w_n)$, where $\circ$ represents the concatenation operation of the semantic units. If the estimate of the function f on one of the components w_k fails, then the estimate of w will finally fail, too; this will determine the inexistence of a semantic associate for w. As a matter of fact, w can have complex forms, entailing the same property for $f(w)$.

In order to understand the form of $f(w)$, we define two operations in X, strictly needed by the symbolization of the final form of $f(w)$:

— the concatenation operation $\circ$, $\forall \sigma_1, \sigma_2 \in \mathfrak{M}$, $\sigma_1 \circ \sigma_2 = \sigma\circ\sigma_2$ (where $\sigma_1\sigma_2$ means a sequence of semantic components);

— the stratification operation (analogous to the brackets in algebraic expressions); using the brackets, the depth level of a word w in an expression w will correspond to a closing level of a pair of brackets.

The result of the estimate $f(w)$ will appear as a stratification on levels of the semantic forms corresponding to the syntactic forms. In other words, $f(w)$ will appear as a translation of w into the language of the semantic forms. The form $f(w)$ can constitute a starting form for the analysis of the semantic correctness of the expressions w. In the concrete case of the programming languages, this amounts to the analysis of the correctness of the programs.

To each formal expression $w \in \mathfrak{W}$ corresponds an object of dynamic calculation (calculation process) or a static object data type).

The extensibility, considered both conceptually and as a mode of performing, must be seen at this level; this fact ensures the naturalness of the notion specification, too, in the frame of the specification of the programming language. In this meaning, the extensibility has a dynamic character and can be performed by either the construction of new composed objects and new operation schemes, or the enrichment of the properties of the existing objects and the extension of the existing operation schemes. This extension of the operation schemes must be seen as the extension of an algebraic relationship. Let TIP1 and TIP2 be two specified abstractions and let TIP3 be a bew composed abstraction whose specification is based on TIP1 and TIP2. The new operation schemes corresponding to the definition of the new abstraction will be defined as follows:

$$\sigma: \text{TIP1} \times \text{TIP2} \rightarrow \text{TIP3},$$

where TIP1 and TIP2 are injected into TIP3 as structured supports or as parameters for expressing the objects from TIP3.

3. Examples. We present further down some examples of dynamic extensibility.

Example 1. This constitutes a mode of extending the comparison operations ($<$, $<=$, $>$, $>=$) to the arrays having the same dimensions. We mean by this that only two arrays of the same dimensions can be compared.

The comparison $A <= B$, where A and B are two arrays, will have truth value if for $A(i, j)$ and $B(i, j)$, $i, j = \overline{1, n}$, we have $A(i, j) <= B(i, j)$. Such

an extension has not a great enough importance for being statically ensured at the instant of the programming language specification. This must constitute a programming option, allowing to the user more elasticity in programming. But, if this extension is performed in a parametrized manner at the compiler level, this will allow to perform sensibly simplified programs.

Example 2. In close connection with the first example, we can also perform the extension of some types of action objects. For inszance, in the following instruction:

IF COND THEN ACTIUNE1 ELSE ACTIUNE2

the condition COND can be performed by using composition operations extended to arrays of real numbers. Such models of extension can also be imagined for the arithmetic operations $(+, -, \cdot, /)$.

Example 3. This example constitutes an algebraic model of hierarchized specification of the data types in a programming language. The mathematical models compel more and more recognition as to the specification and implementation of the languages. They gain permanently ground against the artisanal methods of specification, implementation and extension of the man-machine communication languages. The formulation of the problems by means of the mathematical apparatus with all the corresponding notions and concepts allows the study and establishment of well substantiated algorithms as to the specification, implementation and extensibility of the programming languages. In the following, using the HAS hierarchies, we shall present an algebraic model of data stratification in a certain programming language.

We choose as zero level of the hierarchy the following homogeneous algebra:

$$\mathcal{A}_0 = \{D_0,\ \Omega_0,\ F_0 : D_0 \to I\},$$

where:

D_0 = the support of the algebra, consisting of the set of all possible data in a programming language;

Ω_0 = the set of operations defined on the support D_0;

F_0 = a function which associates to every element x from D_0 its representation length in standard units: $\forall x \in D_0,\ F_0(x) = l(x)$, where $l(x)$ is the representation length of x in storage standard units;

I = a subset of the natural number set, which represents the set of all values of the function F_0 and will constitute the index set for the next level in the hierarchy.

The next level of the hierarchy is defined on the basis of the zero level and has the following form:

$$\mathcal{A}_1 = \{D_1 = (D)_{i \in I},\ (\Sigma S_o)_{o \in \Omega_o},\ F_0 : D_1 \to I,\ F_1\},$$

where:

D_1 = a first partitioning of the elements of the support D_0 in classes of data types, the partitioning criterion being the representation length;

ΣS_o = the set of operation schemes corresponding to the definition of the new object types (classes of data types), or calculation with objects only. For

$o \in \Omega_0$, the function m points out the n-arity, $m(o) = n$. If $b = b_1 b_2 \ldots b_n o$, then an operation scheme:

$$\sigma = (n,\ o,\ F_0(b_1)\ F_0(b_2) \ldots F_0(b_n) F_0(b))$$

is associated. When o runs over the operation domain, Ω_0, and for a fixed o it is $(b_1, b_2, \ldots, b_n) \in D_0^n$ which varies, the result is the set of all operation schemes which can be defined in the frame of the level 1 of the hierarchy;

F_1 = the symbol of a function which associates to each operation scheme a heterogeneous operation scheme specific to the level 1.

If $\sigma = (n,\ o,\ F_0(b_1)\ F_0(b_2) \ldots F_0(b_n)\ F_0(b))$ is an operation scheme, then $F_1(\sigma)$ is a specific operation in $\mathcal{A}_1$ defined as follows:

$$F_1(\sigma) : D_{F_0(b_1)} \times D_{F_0(b_2)} \times \ldots \times D_{F_0(b_n)} \to D_{F_0(b_0)}.$$

The domain and co-domain of the operation $F_1(\sigma)$ are inherited from the previous level; what is specific in the new hierarchy level is its manner of action. If o satisfies the commutativity property, which is expressed as follows:

$$o(b_1,\ b_2,\ \ldots,\ b_n) = o(b_{t1},\ b_{t2},\ \ldots,\ b_{tn}), \tag{1}$$

where $(t1, t2, \ldots, tn)$ is a permutation of the sequence $(1, 2, \ldots, n)$, then this property will be transmitted to the algebra $\mathcal{A}_1$, too: the function F_1 associates to the schemes:

$$\sigma_1 = (n,\ o,\ F_0(b_1)F_0(b_2) \ldots F_0(b_n)F_0(b))$$

and:

$$\sigma_2 = (n,\ o,\ F_0(b_{t1})F_0(b_{t2}) \ldots F_0(b_{tn})F_0(b))$$

the same law of composition, namely $F_1(\sigma_1) = F_1(\sigma_1)$. In other words:

$$F_1(\sigma_1) : D_{F_0(b_1)} \times D_{F_0(b_2)} \times \ldots \times D_{F_0(b_n)} \to D_{F_0(b)}$$

and:

$$F_1(\sigma_2) : D_{F_0(b_{t1})} \times D_{F_0(b_{t2})} \times \ldots \times D_{F_0(b_{tn})} \to D_{F_0(b)}$$

are chosen to be equal functions, leaving aside a permutation of the domain of definition.

The equality (1) defines an equivalence relationship on D_0. In this way one easily notices that ΣS_0 can be seen as the set of the equivalence classes specified by the equality (1). This finding allows to reduce a number of operations specified by $\mathcal{A}_0$ in $\mathcal{A}_1$.

Taking into account the mode in which the function F_0 is defined, the set D_1 is identified with the set of equivalence classes $D_1/\text{Ker } F_0$, where:

$$\text{Ker } F_0 = \{(b_1,\ b_2) \in D_1 \times D_2 \mid F_0(b_1) = F_0(b_2)\}.$$

In order to define the level 2 of the HAS hierarchy, one must have in view the manner of data interpretation. This starts from the fact that a certain datum represented in a certain representation mode can be interpreted

differently. For instance, a value a represented on an octet can be interpreted as either integral value or character. The significance of the interpretation depends ob the type of the variable attached to the respective value. In the case of the level 1 of the HAS hierarchy, $D_{F_0(b)}$, where $F_0(b_1) = 1$, represents the class of data types which are represented on an octet. But, generally, in a programming language, the set of data types represented on an octet (of length 1) consists of: the boolean type, the character type, the byte type, etc.

In order to facilitate the mode of expression, we consider $D_1 = (D_i)_{i \in I}$, where I is the co-domain of the function F_0. Taking into account the fact that I is a finite set, the same property results also for the family of sets $(D_i)_{i \in I}$.

The connection between the interpretation mode and the representation length of unstructured-type data in a programming language is illustrated by the following two-dimensional matrix:

$$A_{m \times n} = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1j} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2j} & \dots & a_{2m} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ a_{i1} & a_{i2} & \dots & a_{ij} & \dots & a_{im} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nj} & \dots & a_{nm} \end{pmatrix}$$

where:

$$a_{ij} = \begin{cases} 0, \text{ if for the representation length } i \text{ the type (interpretation mode) } j \text{ does not exist;} \\ 1, \text{ in the opposite case.} \end{cases}$$

The index of line $i = \overline{1, n}$ constitutes the representation length of the data, while the index of column $j = \overline{1, m}$ represents the index associated to the set of the interpretation modes. We gather all interpretation modes in the set M and all possible representation lengths in the set N. Lines equal to zero can exist in the matrix; these ones correspond to an inexistent representation dimension for a certain language for which the hierarchy is constructed.

Every D_i must be seen as a reunion of subsets D_{ij}, $j = \overline{1, m}$, which can have common elements. In other words, D_i explodes in the subsets corresponding to the unstructured data types. Taking into account the above specifications, the level 2 of the HAS hierarchy is defined as follows:

$$\mathcal{A}_2 = \{D_2 = (D_{ij})_{i \in N, j \in M}, (\Sigma So_{ij}), F_2 : D_1 \times M \to N \times M, F_2'\},$$

where:

D_{ij} = the support of the data type of length i, interpreted in the mode j;

ΣSo_{ij} = the set of operation schemes for defining the new types of data, or calculation schemes in the frame of the existing data of length i, interpreted in the mode j;

F_2 = function which establishes by its values the index set for the family of sets D_2: $\forall(b, j) \in D_1 \times M$, $F_2(b, j) = (F_0(b), j)a_{F_2(b)j}$;

F_2' = the symbol of a function which associates to each scheme $\sigma = (n, o, b_1b_2 \ldots b_nk)$ and to an interpretation mode j a heterogeneous operation scheme specific to the level 2 of the hierarchy.

The support of the heterogeneous algebra $\mathcal{A}_2$ being established, the indexation of the family of sets D_{ij} can be reconsidered with a single index i, in order to simplify the mode of defining the function F_2'. For this purpose we re-define the analytical expression of the function F_2 as follows:

$$F_2(b, j) = (m(F_0(b) - 1) + (j + 1))a_{F_2(b)j},$$

where the definition elements are those previously defined.

For a given σ, $F_2'(\sigma)$ will represent a specific operation in $\mathcal{A}_2$, defined as follows:

$$F_2'(\sigma, j) : D_{F_2(b_1, j)} \times D_{F_2(b_2, j)} \times \ldots \times D_{F_2(b_n, j)} \to D_{F_2(b, j)},$$

where the domain and the co-domain are inherited from the previous level, while the mode of action will be specific to the new level.

Now there is still only a step to the definition of the structured data types (RECORD, FILE, SET, etc.). The level 3 of hte hierarchy will keep its support from the previous level, enriching itself with only a series of operations characteristic to the new defined data types. We make the convention to define the level 3 of the hierarchy as follows:

$$\mathcal{A}_3 = \{D_3 = (D_i)_{i \in I}, (\Sigma S_i), F_3'\},$$

where the components of the heterogeneous algebra have significances similar to those of the level 2 of the HAS hierarchy. The obtaining of structured-type objects on the basis of unstructured calculation objects is connected to the action mode of the function F_3'. What is characteristic for this level, where composed (structured) calculation objects are obtained, is the concatenation operation of the already defined calculation objects.

REFERENCES

1. Rus, T., *Mecanisme formale pentru specificarea limbajelor*, Ed. Acad. R.S.R., Bucureşti, 1983
2. Purdea, I., Pic, G., *Tratat de algebră modernă*, Vol. 1, Ed. Acad. R.S.R., Bucureşti, 1977.
3. Parpucea, I., *The HAS Hierarchy and the Extensibility of the Programming Languages*, Studia Univ. Babeş—Bolyai, ser. Mathematica, *32* (1987), No. 3, 3—14.

STUDIA UNIV. BABEŞ-BOLYAI, MATHEMATICA, XXXIII, 3, 1988

HIERARCHICAL CLASSIFICATION FOR LINEAR CLUSTERS

D. DUMITRESCU* and C. LENART*

Received: July 20, 1988

REZUMAT. — Clasificarea ierarhică pentru clusteri liniari. În lucrare se propune o metodă de clasificare ierarhică divizivă. Se consideră că aglomerările de puncte (nori, clusteri) din mulţimea datelor de clasificat au forme liniare. Admitem că o clasă de puncte poate fi descrisă cu succes de o mulţime nuanţată. Se propune un algoritm (GFL) pentru detectarea subclusterilor liniari ai unei clase nuanţate. Acest algoritm se utilizează pentru a construi o ierarhie nuanţată ce descrie structura de clasificare a unei mulţimi de date. În final se prezintă algoritmul de clasificare ierarhică FDH. Algoritmul permite detectarea claselor liniare in absenţa oricăror informaţii a priori privind structura datelor.

Introduction. A divisive hierarchical classification algorithm has been proposed in the papers [2], [4], [5], [6]. The aim of this paper is to extend this classification procedure in order to detect linear shape clusters. The method to obtain the principal component of a fuzzy class given in [3] may also be used for the classification of linear clusters.

1. Linear cluster substructure of a fuzzy class. Let $X = \{x^1, \ldots, x^p\}$, $x^j \in \mathbf{R}^d$ be a data set. The cluster substructure of a fuzzy class C of points from X may be described as a fuzzy partition of C ([2], [4]).

Let $P = \{A_1, \ldots, A_n\}$ be a fuzzy partition of C, where C is a fuzzy set on X.

The linear shape cluster may be reprezented by a direction u^i and a point v^i. The prototype L^i of the fuzzy class A_i is thus

$$L^i = (v^i, u^i).$$

The linear variety corresponding to this prototype is

$$V_i(v^i, u^i) = \{y \in \mathbf{R}^d \mid y = v^i + tu^i, \ t \in \mathbf{R}\}.$$

Let d be a norm induced metric. The distance of a point x to V_i is

$$d(x, V_i) = (\| x - v^i \|^2 - (x - v^i, u^i)^2)^{1/2}.$$

The scalar product is

$$(x, y) = x^T M y,$$

where M is a symmetric positive definite matrix.

* *University of Cluj-Napoca, Faculty of Mathematics and Physics, 3400 Cluj-Napoca, Romania.*

In order define a dissimilarity between a point x^j and the prototype L^i we'll use the local metric d induced by d and the fuzzy class A_i. According to [2] we may write

$$d_i(x^j, V_i) = A_i(x^j)\ d(x, V_i). \tag{4}$$

The dissimilarity $D_i(x^j, L^i)$ between x^j and L^i is therefore

$$D_i(x^j, L^i) = (A_i(x^j))^2(\| x^j - v^i \|^2 - (x^j - v^i, u^i)^2). \tag{5}$$

The inadequancy between the class A_i and its reprezentation by the prototype L^i can be expressed using D_i as

$$I(A_i, L^i) = \sum_{j=1}^{p} D_i(x^j, L^i). \tag{6}$$

Let $L = (L^1, \ldots, L^n)$ be the representation of the fuzzy partition P. The inadequacy between P and L is given by

$$J(P, L) = \sum_{i=1}^{n} I(A_i, L^i). \tag{7}$$

Therefore we may write

$$J(P, L) = \sum_{i=1}^{n} \sum_{j=1}^{p} (A_i(x^j))^2(\| x^j - v^i \|^2 - (x^j - v^i, u^i)^2). \tag{8}$$

The optimal fuzzy partition and its reprezentation may be viewed as minimizing the criterion function $J : F_n(C) \times (\mathbf{R}^{dn} \times \mathbf{R}^{dn}) \to \mathbf{R}$ given by (8). Therefore we have the minimization problem

$$\begin{cases} \text{minimize } J(P, L) \\ P \in F_n(C) \\ L \in \mathbf{R}^{dn} \times \mathbf{R}^{dn} \end{cases} \tag{9}$$

$F_n(C)$ is the family of all fuzzy partition of C having n atoms.

In order to solve the problem (9) we may use the next:

PROPOSITION 1. *$P \in F_n(C)$ is a local minimum of the function $J(\cdot, L)$ if and only if*

$$A_i(x^j) = \frac{C(x^j)}{\sum_{k=1}^{n} \frac{d^2(x^j, L^i)}{d^2(x^j, L^k)}} \quad i = 1, \ldots, n;\ j = 1, \ldots, p. \tag{10}$$

Proof. See [2], [4].

PROPOSITION 2. *$L \in \mathbf{R}^{dn} \times \mathbf{R}^{dn}$ is a local minimum of the function $J(P,$ if and only if*

(i)
$$v^i = \frac{\sum_{j=1}^{p} (A_i(x^j))^2 x^j}{\sum_{j=1}^{p} (A_i(x^j))^2}, \quad i = 1, \ldots, n. \tag{11}$$

(ii) *$u^i =$ is the unit eigenvector corresponding to the largest value of the scalar matrix S_i of the class A_i,*

$$S_i = M\left(\sum_{j=1}^{p} (A_i(x^j))^2 (x^j - v^i)(x^j - v^i)^T\right) M. \tag{12}$$

Proof. The proof given in [1] for the particular case $C = X$ also holds for C a fuzzy set on X, $C \neq \emptyset$.

The Fuzzy n-Lines algorithm [1] may now be generalized to detect the cluster substructure of a fuzzy class.

GENERALIZED FUZZY n-LINES ALGORITHM (GFL).

S_1. Fix n, $2 \leqslant n \leqslant p - 1$; fix a scalar product on $\mathbf{R}^d$.
S_2. Initialize a fuzzy partition $P = \{A_1, \ldots, A_n\}$ of C.
S_3. Compute $L^i = (v^i, u^i)$, $i = 1, \ldots, n$, using (11) and (12).
S_4. Compute a new fuzzy partition P^2 those atoms are given by (10)
S_5. If $\| P^2 - P^1 \| \leqslant \varepsilon$ then stop. Otherwise set $P^1 := P^2$ and go to S_3.

2. Fuzzy hierarchical classification of linear clusters. A divisive procedure to obtain a binary fuzzy hierarchy ([2], [4], [5]) may be developed using the GFL algorithm. In this hierarchy we have to consider only actual clusters. In this respect we define a clustering degree of a binary fuzzy partition.

Let $P = \{A_1, A_2\}$ be a fuzzy partition of a fuzzy class C. The clustering degree $R(P)$ of P is defined by

$$R(P) = \frac{\sum_x A_{1,\frac{1}{2}}(x) + \sum_x A_{2,\frac{1}{2}}(x)}{\sum_x C(x)}. \tag{13}$$

where $A_{i,\frac{1}{2}}$ is the $\frac{1}{2}$-cut of the fuzzy set A_i i.e.

$$A_{i,\frac{1}{2}}(x) = \begin{cases} A_i(x) & \text{if } A_i(x) > \frac{1}{2} \\ 0, & \text{otherwise} \end{cases}$$

$R(P)$ is also called the polarization coefficient and it measures the structuredness degree ([2], [4]) of a binary fuzzy partition. It is easy to see that $0 \leqslant R(P) \leqslant 1$.

We admit that a binary fuzzy partition $P = \{A_1, A_2\}$ describes "real" clusters iff the next condition are fulfilled:

(i) $\exists x \in X,\ A_i(x) > \frac{1}{2},\ \forall i = 1, 2.$ (14a)

(ii) $R(P) \geqslant t$, where t is an appropriate threshold. (14b)

The algorithm corresponding to the divisive procedure to obtain a fuzzy hierarchy is the next:

FUZZY DIVISIVE HIERARCHICAL (FDH) CLUSTERING ALGORITHM FOR LINEAR CLUSTERS

S1. Set $l := 0,\ N := 0,\ P = \{X, \emptyset\}$.

S2. For every $C \in P^l$, C not marked and $C \neq \emptyset$ perform S4. Allocate every marked C to P^{l+1}.

S3. If $P^{l+1} = P^l$ then stop.
Otherwise set $l := l + 1$ and go to S2.

S4. Using the GFL algorithm calculate the fuzzy partition $P = \{A_1, A_2\}$ of C and its representation $(L^1, L^2);\ L^i = (u^i, v^i)$. If P describes real clusters (conditions (14)), then allocate A_1 and A_2 to P^{l+1}. Otherwise mark C and set $N := N + 1$.

It is easy to see that this algorithm leads to a binary fuzzy hierarchy The fuzzy partitions $P^0, P^1, \ldots, P^l$ represent a chain with respect to the refinement relation ([4]).

REFERENCES

1. Bezdek, J. C., Coray C., Gunderson, R., Watson, J., *Detection and characterization of cluster substructure*, Siam J. Appl. Math., 40 (1981), 339–371.
2. Dumitrescu, D., *Numerical methods in fuzzy hierarchical pattern recognition I, II*, Studia Univ. Babeş–Bolyai, Math., 4 (1986), 31–36; 1 (1987), 24–30.
3. Dumitrescu, D., Toadere, T., *Principal components of a fuzzy class*, Studia Univ. Babeş–Bolyai, Math., 3 (1987). 24–28.
4. Dumitrescu, D., *Hierarchical pattern classification*, Fuzzy Sets and Systems, 28(1988), 145–162.
5. Dumitrescu, D., *Divisive hierarchical classification*, Economic Computation and Economic Cybernetics Studies and Ressarch, Bucarest, 3 (1987), 31–38.
6. Dumitrescu, D., *Hierarchical detection of linear cluster substructure*, Univ. of Cluj-Napoca, Fac. Math. Res. Seminars, 2 (1986), 21–28.
7. Dumitrescu, D., *Hierarchical classification in pattern recognition*, Proc. of the Coll. on Informatics, Iassy, 1985, vol. II, 638–643.

STUDIA UNIV. BABEŞ-BOLYAI, MATHEMATICA, XXXIII, 3, 1988

CLASSIFICATION WITH FUZZY RELATIONS

CRISTIAN LENART*

Received: July 22, 1988

REZUMAT. — Clasificare cu relaţii nuanţate. În lucrare se fundamentează matematic o tehnică de clasificare bazată pe aproximarea relaţiilor nuanţate prin relaţii clasice de echivalenţă. Sînt date două teoreme de existenţă a elementului de cea mai bună aproximare în raport cu norma Cebişev, respectiv norma integrală. Printr-un exemplu numeric se arată că această metodă furnizează rezultate mai „plauzibile" decît metoda descompunerii convexe a relaţiilor nuanţate [1].

Introduction. The aim of this paper is to give a new classification meth with fuzzy relations. This method is based on the approximation of a fuz relation with classical (hard) equivalence relations. It is well known that a equivalence relation induces a hard partition on the data set. We study t existence of the best approximation of a fuzzy relation with respect to t Tchebishew and the integral norm. A numerical example will show that o method is a more natural approach to the classification problem than the co vex decomposition of a fuzzy relation [1].

1. Definitions and notations. Let X be a non-empty set. A fuzzy set o X is a function $A : X \to [0, 1]$. A fuzzy relation on X is a fuzzy set o $X \times X$.

A fuzzy relation R is a similarity one if it satisfies the following axiom

a) $R(x, x) = 1$, $(\forall)\ x \in X$ (reflexivity)

b) $R(x, y) = R(y, x)$, $(\forall) x, y \in X$ (simmetry)

c) $R(x, y) \geqslant \sup_{z \in X} [\max(R(x, z) + R(z, y) - 1, 0)]$ (max $-\Delta$ transitivity

Bezdek and Harris [1] proved that R is a similarity relation iff $1-R$ is a pseudometric on X. Let $\mathcal{R}(X)$ be the family of all similarity relations o X and $\mathcal{R}_0(X)$ the family of all classical equivalence relations on X.

2. The best approximation of fuzzy relations. In this section we stud the existence of the best approximation of a fuzzy similarity relation wit classical equivalence relations in some subspaces of $\mathbf{R}^{X \times X}$ with respect to di ferent norms.

Let us first consider the bounded functions subspace and the Tchebishe norm:

$$||f - g|| = \sup_{(x,y)} |f(x, y) - g(x, y)|$$

* *University of Cluj-Napoca, Faculty of Physics and Mathematics, 3400 Cluj-Napoca, Romania.*

THEOREM 1. *For every* $R \in \mathcal{R}(X)$ *there exists* $R^* \in \mathcal{R}_0(X)$ *which is the best approximation with respect to the Tchebishew norm. Moreover, if:*

$$\rho = \inf\{r \mid [R(x_0, x_1) \geqslant r, \ldots, R(x_{n-1}, x_n) \geqslant r] \Rightarrow R(x_0, x_n) > 1 - r\}$$

then

$$\| R - R^* \| = \rho \tag{1}$$

Proof. Let R^* be defined as follows: $R^*(x, y) = 1 \Leftrightarrow (\exists)\ x = x_0, x_1, \ldots, x_{n-1}, x_n = y$ a sequence of elements in X such that $R(x_k, x_{k+1}) > \rho$, $(\forall)k = 0, 1, \ldots, n-1$. It is obvious that $R^* \in \mathcal{R}_0(X)$.

Let $x, y \in X$. If $R^*(x, y) = 1 \Rightarrow (\exists)\ x = x_0, x_1, \ldots, x_{n-1}, x_n = y \in X$ such that $R(x_k, x_{k+1}) > \rho$, $(\forall)k = 0, 1, \ldots, n-1 \Rightarrow (\exists)\ \varepsilon_0 > 0$ such that $R(x_k, x_{k+1}) \geqslant \rho + \varepsilon$, $(\forall)\ 0 < \varepsilon \leqslant \varepsilon_0 \Rightarrow R(x, y) > 1 - \rho - \varepsilon$, $(\forall)\ 0 < \varepsilon \leqslant \varepsilon_0$ (according to (1)) $\Rightarrow 1 - R(x, y) \leqslant \rho \Rightarrow$

$$| R^*(x, y) - R(x, y) | \leqslant \rho \tag{2}$$

On the other hand, if $R^*(x, y) = 0$, then $R(x, y) \leqslant \rho$ (according to the definition of R^*) $\Rightarrow$ (2). Hence the inrquality (2) holds for all $x, y \in X$.

According to (1), for every $\varepsilon > 0$, there exists a sequence $x = x_0, x_1, \ldots, x_n = y$ such that $R(x_0, x_1) \geqslant \rho - \varepsilon, \ldots, R(x_{n-1}, x_n) \geqslant \rho - \varepsilon$ and $R(x, y) \leqslant 1 - \rho + \varepsilon$ (otherwise $\rho - \varepsilon$ would belong to the set of which inf is ρ). If $R^*(x, y) = 1$, we have:

$$| R^*(x, y) - R(x, y) | \geqslant \rho - \varepsilon \tag{3}$$

If $R^*(x, y) = 0$ then there exists k such that $R^*(x_k, x_{k+1}) = 0$ (otherwise the transitivity of R^* would imply $R^*(x, y) = 1$). Hence from $R(x_k, x_{k+1}) \geqslant \rho - \varepsilon$, we get:

$$| R^*(x_k, x_{k+1}) - R(x_k, x_{k+1}) | \geqslant \rho - \varepsilon \tag{4}$$

From (3) and) (4) it follows that for every $\varepsilon > 0$ there exist $x, y \in X$ such that:

$$| R^*(x, y) - R(x, y) | \geqslant \rho - \varepsilon \tag{5}$$

From (2) and (5) it follows that $\| R - R^* \| = \rho$.

We now assume that R^* is not a best approximations. This means that there exists $R_1 \in \mathcal{R}_C(X)$ with $\| R - R_1 \| < \rho - \varepsilon$, $\varepsilon > 0$. In the same way as above we determine a sequence $x_0, x_1, \ldots, x_n$ such that $R(x_0, x_1) \geqslant \rho - \varepsilon, \ldots, R(x_{n-1}, x_n) \geqslant \rho - \varepsilon$ and $R(x, y) \leqslant 1 - \rho + \varepsilon$. If there exists k such that $R_1(x_k, x_{k+1}) = 0$, then $|R_1(x_k, x_{k+1}) - R(x_k, x_{k+1})| \geqslant \rho - \varepsilon$. This contradicts the inequality $\| R - R_1 \| < \rho - \varepsilon$. Hence $(\forall)k = 0, 1, \ldots, n-1$ we have $R_1(x_k, x_{k+1}) = 1$. According to the transitivity of R_1 we have $R_1(x, y) = 1$. It then follows that $|R_1(x, y) - R(x, y)| \geqslant \rho - \varepsilon$. Contradiction. Hence, R^* is a best approximation of R.

Remark. The best approximation is not unique.

Let us now consider $X \times X$ as a measurable space with the finite measure μ. Let $\mathcal{R}_m(X)$ be the family of all fuzzy measurable relations (as ordinary

functions on $X \times X$) · $L_2(X \times X)$ is a Hilbert space with the integral no induced by the scalar product:

$$\|f\| = \left(\int_{X \times X} f^2 \, d\mu \right)^{1/2}.$$

THEOREM 2. *For every* $R \in \mathcal{R}_m(X)$ *there exists a unique best approxima* R^* *of* R *in* $\overline{\text{conv}\,(\mathcal{R}_0(X) \cap \mathcal{R}_m(X))}$.

Proof. The space $L_2(X \times X)$ is Hilbert and $R \in \mathcal{R}_m(X)$ implies R $\in L_2(X \times X)$ because $X \times X$ has finite measure and R is bounded. The $\overline{\text{conv}\,(\mathcal{R}_0(X) \cap \mathcal{R}_m(X))}$ is convex (as a closure of a convex set). On the ot hand being a closed set in a complete metric space, it is also complete. T assertion of the theorem now follows from the Beppo—Levi theorem (see, nstance [3]).

Remark. The assertion of the theorem is not trivial, which means t there exist fuzzy relations, even similarity ones, which do not lie $\overline{\text{conv}\,(\mathcal{R}_0(X) \cap \mathcal{R}_m(X))}$.

Let us consider the relation R_4 on $X_4 = \{1, 2, 3, 4\}$ given by $R_4(i, j)$ $= a_{ij}$, $(\forall) i, j \in X_4$, where $A = (a_{ij})_{i=\overline{1,4},\, j=\overline{1,4}}$ is the following matrix:

$$A = \begin{bmatrix} 1 & 0.3 & 0.6 & 0 \\ 0.3 & 1 & 0.7 & 0 \\ 0.6 & 0.7 & 1 & 0.2 \\ 0 & 0 & 0.2 & 1 \end{bmatrix}.$$

Let $X = [1, n + 1]$ and R be the fuzzy similarity relation defined by:

$$R(x, y) = \begin{cases} R_4([x], [y]) & \text{if } 1 \leqslant x < n + 1,\ 1 \leqslant y < n + 1 \\ 0 & \text{if } (x = n + 1 \text{ or } y = n + 1) \text{ and } x \neq y \\ 1 & \text{if } x = y = n + 1 \end{cases}$$

In [2] it is shown that the distance between R and $\overline{\text{conv}\,\mathcal{R}_0(X)}$ is strict po tive.

3. Numerical example. Let X be a set with four elements. We consi the fuzzy similarity relation on X given by the matrix:

$$R = \begin{bmatrix} 1 & 0.4 & 0.3 & 0.3 \\ 0.4 & 1 & 0 & 0 \\ 0.3 & 0 & 1 & 0 \\ 0.3 & 0 & 0 & 1 \end{bmatrix}.$$

It is not difficult to see that $R \in \text{conv}\,\mathcal{R}_0(X)$ and R has the unique conv decomposition in $\mathcal{R}_0(X)$:

$$R = 0.4\, R_1 + 0.3\, R_2 + 0.3\, R_3,$$

where

$$R_1 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad R_2 = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad R_3 = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

Thus, according to the convex decomposition method [1], R is approximated by R_1. The best approximation of R with respect to the Tchebishew norm is:

$$R^* = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

It is easy to see that:

$$||R - R^*|| = 0.4 < ||R - R_1|| = 0.6$$

An algorithm for the computation of a best approximation of a fuzzy similarity relation on a finite set with classical equivalence relations is given in [2].

REFERENCES

1. Bezdek J. C., Harris J. D., *Fuzzy Partitions and Relations: an Axiomatic Basis for Clustering*, Reprint from J. Fuzzy Sets and Systems 1 (1978), 111–127.
2. Lenart C., *Pattern Recognition Algorithms in A.I.*, Thesis, University of Cluj-Napoca, 1988.
3. Muntean I., *Functional Analysis*, Lectures Notes, Univ. of Cluj-Napoca, 1974.

STUDIA UNIV. BABEŞ-BOLYAI, MATHEMATICA, XXXIII, 3, 1988

CONSECUTIVE RETRIEVAL WITH MINIMUM REDUNDANCE

LEON ŢÂMBULEA*

Received: July 27, 1988

REZUMAT. — Regăsirea consecutivă cu redundanţă minimă. În acest articol sînt date cîteva rezultate privind proprietatea de regăsire consecutivă cu redundanţă minimă. Aceste rezultate arată că problema poate fi redusă la determinarea unui drum hamiltonian în două grafe, dacă mulţimea de întrebări verifică anumite restricţii. Construirea celor două grafe se bazează pe colecţia de date şi mulţimea de întrebări ce se adresează acestei colecţii.

Let C be a data collection (data base) stored on the medium (supp S which is considered to be linear. Suppose that the data requests fro are formulated as questions, and let Q be the set of these questions. Fo given question $q \in Q$, denote by $q(C)$ its answer, namely the elements i C needed by the question q. In order to determine the answer $q(C)$, a t $t(q)$ (answer time) is needed.

S. P. G h o s h [1] discovered the property of consecutive retrieval, wh is a manner of organizing the data collections, in which there are minim the two important factors: the needed medium for storing the collectio and the answer time for the questions from Q.

DEFINITION 1. The collection C has the property of consecutive retri (the CR property) with respect to Q if all data from $q(C)$ are consecutiv stored on the medium S, $\forall q \in Q$.

The CR property does not exist for every pair (Q, C); that is why seve organization models approximating this property were proposed. Such a mo is based on the consecutive retrieval with minimum redundance, proposed [2]. In the frame of this model, certain data from the collection C may several times stored (hence the redundance appears), but every question m have its answer stored consecutively on the medium S, while the total nu ber of stored data must be minimum. Such a storage mode (with redunda and fulfilling the CR property) does always exist. It is sufficient to store data in the order:

$$q_1(C),\ q_2(C),\ \ldots,\ q_n(C),$$

if $Q = \{q_1, q_2, \ldots, q_n\}$.

DEFINITION 2. [3, 6]. If p is the total number of stored data, then mean by redundance of a storage mode the ratio $r = (p - m)/m$, where is the number of element from C (the value $p - m$ specifies the number supplementary stored data).

* *University of Cluj-Napoca, Faculty of Mathematics and Physics, 3400 Cluj-Napoca, Roman a.*

Remark 1. If every record from C is stored only once on the medium S, then r from Definition 2 is zero.

In the following there are analysed different situations of determining the consecutive retrieval with minimum redundance, which reduce to the determination of the optimal Hamiltonian path in a graph. In [4, 5] the problem is studied in the general case, but the obtained algorithms are very complex. In [2, 3] different limits (bounds) for the redundance value are given. As one shows in [7], the problem in the general case is very difficult.

We shall firstly consider the case in which the number of elements in Q is 3 or 4. Let be $Q = \{q_1, q_2, \ldots, q_n\}$. We construct the completely unoriented graph $G_1 = \{X, U, c\}$, where:

$X = \{q_1, q_2, \ldots, q_n\} = Q$;

$U =$ all $\binom{n}{2}$ edges from the complete graph;

$c(q_i, q_j) =$ the number of data from the answer $q(C)$ of the following composed question (conjunctions of questions or negations of questions from Q):

$$q = \bar{q}_1 \wedge \ldots \wedge \bar{q}_{i-1} \wedge q_i \wedge \bar{q}_{i+1} \wedge \ldots \wedge \bar{q}_{j-1} \wedge q_j \wedge \bar{q}_{j+1} \wedge \ldots \wedge \bar{q}_n.$$

For the question $q \in Q$ we shall denote by $a(q)$ the address of the first datum from $q(C)$ which appears on the storage medium S. Also denote:

$$q_i^b = \begin{cases} q_i, & \text{if } b = 1; \\ \bar{q}_i, & \text{if } b = 0; \end{cases}$$

$$p(q_1^{i_1} \ldots q_n^{i_n}) = q(C),$$

where $q = q_1^{i_1} \wedge \ldots \wedge q_n^{i_n}$.

THEOREM 1. *Let be* $Q = \{q_1, \ldots, q_n\}$, $C = \{d_1, \ldots, d_m\}$. *If* $n \leqslant 4$, *then the minimum redundance is:*

$$r = (s - d)/m, \tag{1}$$

where:

$$s = \sum_{\substack{(i_1, \ldots, i_n) \in \{0,1\}^n \\ i_1 + \ldots + i_n \geqslant 2}} |p(q_1^{i_1} \ldots q_n^{i_n})|,$$

while d *is the length of the maximum valued Hamiltonian path in the above constructed graph* G_1. *If this Hamiltonian path is* $(q_{j_1}, q_{j_2}, \ldots, q_{j_n})$, *then:*

$$a(q_{j_1}) \leqslant a(q_{j_2}) \leqslant \ldots \leqslant a(q_{j_n}).$$

Proof. For $n = 3$, it is given in [3] a D. E. Knuth's result, in which the minimum redundance is:

$$r = (|p(q_1q_2q_3)| + \min(|p(\bar{q}_1q_2q_3)|, |p(q_1\bar{q}_2q_3)|, p(q_1q_2\bar{q}_3)|))/m. \tag{2}$$

If we denote:

$$Y = \{|p(\bar{q}_1q_2q_3)|, |p(q_1\bar{q}_2q_3)|, |p(q_1q_2\bar{q}_3)\} = \{c(q_2, q_3), c(q_1, q_3), c(q_1, q_2)\},$$

then from (2) we obtain (1), where:

$$d = \begin{cases} |p(q_1q_2\bar{q}_3)| + |p(\bar{q}_1q_2q_3)| = c(q_1, q_2) + c(q_2, q_3), \text{ if } \min Y = c(q_1, q_3), \\ |p(q_1\bar{q}_2q_3)| + |p(\bar{q}_1q_2q_3)| = c(q_1, q_3) + c(q_2, q_3), \text{ if } \min Y = c(q_1, q_2), \\ |p(q_1q_2\bar{q}_3)| + |p(q_1\bar{q}_2q_3)| = c(q_1, q_2) + c(q_1, q_3), \text{ if } \min Y = c(q_2, q_3). \end{cases}$$

In the graph G_1 there are three Hamiltonian paths, and d is just the length of the minimum valued Hamiltonian path. If, for instance, $d = c(q_1, q_2) + c(q_2, q_3)$, then, storing the data such that $a(q_1) \leqslant a(q_2) \leqslant a(q_3)$, they can appear in the order: $p(q_1\bar{q}_2\bar{q}_3)$, $p(q_1\bar{q}_2q_3)$, $p(q_1q_2\bar{q}_3)$, $p(q_1q_2q_3)$, $p(\bar{q}_1q_2\bar{q}_3)$, $p(\bar{q}_1q_2q_3)$ $p(\bar{q}_1\bar{q}_2q_3)$. $p(q_1\bar{q}_2q_3)$, $p(q_1q_2q_3)$, and the redundance of this storage mode is (1).

Now we shall consider $n = 4$. Suppose that:

$$a(q_{i_1}) \leqslant a(q_{i_2}) \leqslant a(q_{i_3}) \leqslant a(q_{i_4}), \tag{3}$$

and denote $a_j = a(q_{i_j})$, $j = 1, 2, 3, 4$.

From the address a_1 (on the medium) the set $q_{i_1}(C)$ must be stored, while from the address a_2 — the set $q_{i_2}(C)$. We shall obtain the minimum redundance if $q_{i_1}(C)$ will be stored in the following order:

— the set $A_1 = q_{i_1}(C) - q_{i_2}(C) = p(q_{i_1}\bar{q}_{i_2})$ from the address a_1;

— the set $B_1 = q_{i_1}(C) \cap q_{i_2}(C) = p(q_{i_1}q_{i_2})$ from the address a_2.

Since from the address a_2 the set $q_{i_2}(C)$ must be stored, while from the address a_3 — the set $q_{i_3}(C)$, then we can firstly store B_1, then $A_2 = q_{i_3}(C) - B_1 - q_{i_3}(C) = p(\bar{q}_{i_1}q_{i_2}\bar{q}_{i_3})$, and finally the elements which have remained from $q_{i_2}(C)$, namely $B_2 = q_{i_2}(C) \cap q_{i_3}(C) - B_1$, stored starting with the address a_3.

Analogously, the set $q_{i_3}(C)$ can be stored in the following order: B_2 from the adress a_3, $A_3 = q_{i_3}(C) - B_2 - q_{i_4}(C) = p(q_{i_1}q_{i_2}\bar{q}_{i_4}) \cup p(q_{i_1}q_{i_2}\bar{q}_{i_4})$ after B_2, and $B_3 = q_{i_3}(C) \cap q_{i_4}(C) - B_2$ starting from the address a_4, followed by $A_4 = q_{i_4}(C) - B_3 = p(\bar{q}_{i_1}q_{i_2}q_{i_3}q_{i_4}) \cup p(\bar{q}_{i_2}q_{i_3}q_{i_4})$.

Observe from the above described storage mode that the CR property does exist because: $q_{i_1}(C) = A_1 \cup B_1$, $q_{i_2}(C) = B_1 \cup A_2 \cup B_2$, $q_{i_3}(C) = B_2 \cup A_3 \cup B_3$, $q_{i_4}(C) = B_3 \cup A_4$. As one can notice, there is no other possibility of storage fulfilling (3) and with a smaller redundance. The redundance obtained with this storage mode is:

$$r = \frac{1}{m}(s - (|p(q_{i_1}q_{i_2}\bar{q}_{i_3}\bar{q}_{i_4})| + |p(\bar{q}_{i_1}q_{i_2}q_{i_3}\bar{q}_{i_4})| + |p(\bar{q}_{i_1}\bar{q}_{i_2}q_{i_3}q_{i_4})|)) =$$

$$= \frac{1}{m}(s - (c(q_{i_1}, q_{i_2}) + c(q_{i_2}, q_{i_3}) + c(q_{i_3}, q_{i_4}))).$$

The redundance is minimum for that permutation (i_1, i_2, i_3, i_4) for which the value:

$$d = c(q_{i_1}, q_{i_2}) + c(q_{i_2}, q_{i_3}) + (q_{i_3}, q_{i_4})$$

is maximum. The value d is the length of a Hamiltonian path in the graph G_1.

For the next result we shall consider a completely oriented graph $G_2 = \{X, U, c\}$, constructed starting from the sets C and Q as follows:

$$X = \{q_0, q_1, \ldots, q_n\} = Q \cup \{q_0\};$$

U = the arcs of the completely oriented graph;

$$c(q_i, q_j) = |\overline{q_i(C)} \cup q_j(C)| = |q_j(C) - q_i(C)|;$$

$$q_0(C) = \emptyset.$$

THEOREM 2. *If* $Q = \{q_1, q_2, \ldots, q_n\}$ *fulfils:*

$$q_i(C) \cap q_j(C) \cap q_k(C) = \emptyset, \ \forall i, j, k \in \{1, 2, \ldots, n\}, \ i \neq j, j \neq k, k \neq i, \quad (4)$$

then the minimum redundance is:

$$r = d/m - 1, \quad (5)$$

where $m = |C|$, *while* d *is the length of the minimum valued Hamiltonian path in the graph* G_2, *path starting from* q_0.

Proof. Suppose that the sets $q_i(C)$ are consecutively stored on the medium and the following condition is fulfilled:

$$a(q_{i_1}) \leqslant a(q_{i_2}) \leqslant \ldots \leqslant a(q_{i_n}). \quad (6)$$

Let be $M_i = q_{j_i}(C)$, $i = 0, 1, \ldots, n$. We construct the following sets (using hereafter the notation $AB = A \cap B$):

$$B_0 = B_n = \emptyset;$$
$$B_i = M_i \cap M_{i+1}, \ i = 1, 2, \ldots, n-1;$$
$$A_i = M_i - B_{i-1} \cup B_i, \ i = 1, 2, \ldots, n.$$

We have $M_i = B_{i-1} \cup A_i \cup B_i$, $i = 1, 2, \ldots, n$, and from (4) we obtain:

$$B_i \cap B_{i+1} = M_i \cap M_{i+1} \cap M_{i+1} = \emptyset.$$

If the data are stored on the medium as follows:

$$A_1, B_1, A_2, B_2, \ldots, A_{n-1}, B_{n-1}, A_n, \quad (7)$$

then the CR property takes place, and the sets $A_1, B_1, B_2, \ldots, B_{n-1}$ will be stored starting from the addresses $a(q_{i_1}), \ldots, a(q_{i_n})$. In the condition (6) the minimum redundance is given by the sequence (7). The number of data from (7) is:

$$d = \sum_{i=1}^{n} |A_i| + \sum_{i=1}^{n-1} |B_{i-1}| =$$
$$= |M_1 \bar{B}_1| + \sum_{i=2}^{n-1} |M_i \bar{B}_{i-1} \bar{B}_i| + |M_n \bar{B}_{n-1}| + \sum_{i=1}^{n-1} |\bar{B}_{i-1}| =$$
$$= |M_1(\bar{M}_1 \cup \bar{M}_2)| + \sum_{i=2}^{n-1} |M_i(\bar{M}_{i-1} \cup \bar{M}_i)(\bar{M}_i \cup \bar{M}_{i+1})| +$$

$$+ |M_n(\overline{M}_{n-1} \cup \overline{M}_n)| + \sum_{i=1}^{n-1} |M_i M_{i+1}| =$$

$$= |M_1 \overline{M}_2| + \sum_{i=2}^{n-1} |M_{i-1} M_i M_{i+1}| + |\overline{M}_{n-1} M_n| + \sum_{i=1}^{n-1} |M_i M_{i+1}| =$$

$$= |M_1(\overline{M}_2 \cup M_2)| + \sum_{i=2}^{n-1} |M_i(\overline{M}_{i-1}\overline{M}_{i+1} \cup M_{i+1})| + |\overline{M}_{n-1} M_n|.$$

Since $\overline{M}_i \cap M_i = \emptyset$, $i = 2, 3, \ldots, n$, $M_{i-1} \cap M_i \cap M_{i+1} = \emptyset$ (from (4)), $i = 2, 3, \ldots, n-1$, and $M_0 = \emptyset$, we obtain:

$$d = |M_1| + \sum_{i=2}^{n-1} |M_i(\overline{M}_{i-1}\overline{M}_{i+1} \cup \overline{M}_{i-1} M_{i+1} \cup M_{i-1} M_{i+1})| + |M_{n-1} M_n| =$$

$$= |M_1| + \sum_{i=2}^{n-1} |M_i \overline{M}_{i-1} \cup M_{i-1} M_i M_{i+1}| + |\overline{M}_{n-1} M_n| =$$

$$= |M_1| + \sum_{i=2}^{n-1} |\overline{M}_{i-1} M_i| = \sum_{i=1}^{n} |\overline{M}_{i-1} M_i|.$$

We obtained that in the condition (6) the redundance is of the form (5), where:

$$d = \sum_{i=1}^{n} |\overline{M}_{i-1} M_i| = \sum_{i=1}^{n} |\overline{q_{j_{i-1}}(C)} \cap |q_{j_i}(C)| = \sum_{i=1}^{n} c(q_{j_{i-1}}, q_{j_i}),$$

and $q_{j_0} = q_0$. Therefore d is the length of the Hamiltonian path $q_0, q_{j_1}, \ldots, q_{j_n}$. Taking into account all permutations $(j_1, j_2, \ldots, j_n)$, we reach the minimum redundance for that permutation from which we obtain a Hamiltonian path of minimum length in the graph G_2, path starting from the vertex q_0.

REFERENCES

1. Ghosh, S. P., *File Organization: The Consecutive Retrieval Property*, Comm. ACM, *15* (1972) 802–808.
2. Ghosh, S. P., *Consecutive Storage of Relevant Records with Redundancy*, Comm. ACM, *18* (1975), 464–471.
3. Ghosh, S. P., *Data Base Organization for Data Management*, Academic Press, 1977.
4. Gorski, J., *On Consecutive Storage of Records*, Math. Found of Computer Science (Proc. 5-th Symp., Gdansk, Sept. 6–10, 1976), *45* (1976), 304–310.
5. Gorski, J., *On the Optimal Arrangement for Consecutive Storage of Objects*, ICS PAS Reports, 279, Warszawa, 1977.
6. Ikeda, H., *A Query Set with Representative Records and Consecutive Retrieval File Organization with Consecutive Retrieval Property*, Proc. Conf. on Consec. Retr. Prop., ICS PAS Reports, 438, Warszawa, 1981, 88–95.
7. Lipski, W., *On String Containing All Subsets as Substrings*, Discrete Math., *21* (1978), 253–259.

STUDIA UNIV. BABEŞ-BOLYAI, MATHEMATICA, XXXIII, 3, 1988

DISCRETE FIXED POINT THEOREMS

IOAN A. RUS*

Received: July 24, 1988

REZUMAT. — Teoreme discrete de punct fix.
Teoria discretă a punctului fix (teoria punctului fix în teoria mulţimilor, în teoria mulţimilor ordonate şi în teoria generală a categoriilor) a cunoscut în ultimi 25 de ani o largă dezvoltare (a se vedeea [3], [18]—[21]). Printre altele menţionăm că S. Eilenberg a stabilit (a se vedea [11], pag. 17—18 şi 168) o variantă discretă a principiului contracţiilor şi a aplicat această variantă în teoria automatelor. În prezenta lucrare se stabilesc teoreme discrete de punct fix de tip Eilenberg şi se dau citeva aplicaţii în analiza neliniară. O problemă deschisă: Care sînt implicaţiile rezultatelor din această lucrare în teoria automatelor?

1. Introduction. The discrete fixed point theory has been, in the last years, subject to an intensive development (see [3], [18] — [21]. The following results are the principal results in the set-theoretical approach for the fixed point theory:

THEOREM 1. (A b i a n [2]). *Let X be a nonempty set. Then $f: X \to X$ has a fixed point if and only if there exists a subset $Y \subset X$, such that for every subset $Z \subset Y$,*

$$Z \cap f(Z) = \emptyset \text{ implies } Y \setminus (Z \cup f(Z) \cup f^{-1}(Z)) \neq \emptyset.$$

THEOREM 2. (A b i a n [1]; see [26]. *Let X be a nonempty set. Then $f: X \to X$ has a fixed point if and only if X is not a union of three mutually disjoint set X_1, X_2, X_3, such that $X_i \cap f(X_i) = \emptyset$ for $i = 1, 2, 3$.*

THEOREM 3. (D e a c o n e s c u [9]). *Let $f: X \to X$ be an injective mapping and $Y \subset X$ a maximal total f-variant subset of X. Then*

$$F_f = (X \setminus Y) \cap (X \setminus f(Y)) \cap (X \setminus f^{-1}(Y)).$$

An other example. S. Eilenberg (see [11] pp. 17—18, 168) formulated a discrete analog of the contractions principle, which has applications in automata theory. The Eilenberg's result is the following:

THEOREM 4. *Let X be set, $R_n \subset X \times X$, $n \in \mathbb{N}$, a sequence of equivalence relations and $f: X \to X$ such that:*

(i) $X \times X = R_0 \supset R_1 \supset \cdots \supset R_n \supset \cdots$,

(ii) $\bigcap_0^\infty R_n = \Delta$ (*the diagonal in* $X \times X$),

* *University of Cluj-Napoca, Faculty of Mathematics and Physics, 3400 Cluj-Napoca, Romania*

(iii) *if* $\{x_n\}$ *is any sequence in* X *such that* $(x_n, x_{n+1}) \in R_n$ *for each* n, *then there is a* $x \in X$ *such that* $(x_n, x) \in R_n$ *for all* $n \in \mathbb{N}$,

(iv) *for all* $n \in N$, $(x, y) \in R_n$ *implies* $(f(x), f(y)) \in R_{n+1}$

Then $F_f = \{x^*\}$ *and* $(f^n(x_0), x^*) \in R_n$, *for all* $x_0 \in X$ *and* $n \in N$.

Oue purpose in this note is to give some discrete fixed point theorems of Eilenberg type. Some applications in nonlinear analysis are given.

Throughout this paper we follow terminologies and notations in [22].

2. General discrete fixed point principles. Let X be a set and $R_n \subset X \times X$, $n \in \mathbb{N}$, a sequence of symmetric binary relations in X. Throughout this paper we suppose that:

(a) $X \times X = R_0 \supset R_1 \supset \cdots \supset R_n \supset \cdots$,

(b) $\bigcap_0^\infty R_n = \Delta(X)$,

(c) if $\{x_n\}$ is any sequence in X such that $(x_n, x_{n+1}) \in R_n$, for all $n \in N$, then there is a unique $x^* \in X$ such that $(x_n, x^*) \in R_n$ *for all* $n \in N$

Let $f: X \to X$ be a mapping. We have

LEMMA 1. *If for all* $n \in N$, $(x, f(x)) \in R_n$, $(y, f(y)) \in R_n$ imply $(f(x), f(y)) \in R_{n+1}$, *then* card $F_f \leqslant 1$.

Proof. Let $x^*, y^* \in F_f$. From (b) we have $(x^*, f(x^*)) \in R_n$, $(y^*, f(y^*)) \in R_n$, for all $n \in N$. These imply $(x^*, y^*) \in R_n$, for all $n \in N$. This implies $x^* = y^*$.

LEMMA 2. *If for all* $n \in N$, $(x, f(x)) \in R_n$ *implies* $(f(x), f^2(x)) \in R_{n+1}$, and $(f^n(x_0), x^*) \in R_n$, $\forall n \in N \Rightarrow (f^{n+1}_{(x_0)}, f(x^*)) \in R_{n+1} \forall n \in N$, *then* $F_f \neq \emptyset$.

Proof. For all $x_0 \in X$, $(x_0, f(x_0)) \in R_0$. This implies $(f(x_0), f^2(x_0)) \in R_1$. Thus we have $(f^n(x_0), f^{n+1}(x_0)) \in R_n$, for all $n \in N$. From (c) there is a unique $x^* \in X$ such that $(f^n(x_0), x^*) \in R_n$, for all $n \in N$. On the other hand, $(f^n(x_0), x^*) \in R_n$ implies $(f(x^*), f^{n+1}(x_0)) \in R_{n+1}$, i.e., $(f^n(x_0), f(x^*)) \in R_n$, for all $n \in N$. From (c) we have $x^* \in F_f$.

THEOREM 5. *If for all* $n \in N$, $((x, y) \in R_n$ *implies* $((f(x), f(y)) \in R_{n+1})$, *then* $F_f = \{x^*\}$, *and* $(f^n(x_0), x^*) \in R_n$, *for all* $n \in N$ *and* $x_0 \in X$.

Proof. From Lemma 2 we have $F_f \neq \emptyset$, and if $x_0 \in X$, then there is $x^* \in F_f$ such that $(f^n(x_0), x^*) \in R_n$ for all $n \in N$. Now, let $x^*, y^* \in F_f$. We have $(x^*, y^*) \in R_0$. This implies $(x^*, y^*) \in R_n$, for all $n \in N$, i.e., $x^* = y^*$.

From Lemma 1 and 2 we have

THEOREM 6. *If for all* $n \in N$,

(i) $((x, f(x)) \in R_n, (y, f(y)) \in R_n)$ *imply* $(f(x), f(y)) \in R_{n+1}$,

(ii) $(x, f(x)) \in R_n$ *implies* $(f(x), f^2(x)) \in R_{n+1}$,

(iii) $(f^n(x_0), x) \in R_n$, $\forall n \in N \Rightarrow (f^{n+1}_{(x_0)}, f(x)) \in R_{n+1} \forall n \in N$,

then $F_f = \{x^*\}$ *and* $(f^n(x_0), x^*) \in R_n$ *for all* $n \in N$.

From Lemma 1.3.3. in [22] and the Theorem 5 we have

THEOREM 7. *Let $k \in N^*$. If for all $n \in N$, $(x, y) \in R_n$ implies $(f^k(x), f^k(y)) \in R_{n+1}$ then $F_f = \{x^*\}$.*

3. Retraction method. Let X be a nonempty set and $Y \subset X$. A mapping $\rho : X \to Y$ is called a retraction of X onto Y if $\rho|_Y = 1_Y$. A mapping $f : Y \to X$ is retractible onto Y (see [5], [23] if there is a retraction $\rho : X \to Y$ such that if $x \in \rho(f(Y) - Y)$, then $f(x) \in (X - \rho^{-1}(x)) \cup \{x\}$.

The following general result is essential in the fixed point theory of non self-mappings.

LEMMA 3 (see [5], [23]. *Let X be a set, Y a nomepty subset of X. If a mapping $f : Y \to X$ is retractible onto Y by means of a retraction $\rho : X \to Y$, then $F_f = F_{\rho of}$.*

We have

THEOREM 8. *Let X be a set, Y a nonmepty subset of X, $\rho : X \to Y$ a retraction and $f : Y \to X$ a mapping. We suppose that*

(i) *for all $n \in \mathrm{N}$, $(x, y) \in R_n$ implies $(\rho(x), \rho(y)) \in R_n$,*

(ii) *for all $n \in \mathrm{N}$, $(x, y) \in R_n$ implies $(f(x), f(y)) \in R_{n+1}$*

(iii) *f is retractible onto Y by means of ρ.*

Then $F_f = \{x^*\}$.

Proof. Let $(x, y) \in R_n$. From (ii), $(f(x), f(y)) \in R_{n+1}$, and from (iii), $(\rho(f(x)), \rho((y)) \in R_{n+1}$. Now the proof follows from the Theorem 5 and the Lemma 3.

4. Applications.

4.1. Metric spaces. From the Theorem 5 we have.

THEOREM 9 (see [22]). *Let (X, d) be a bounded complete metric space and $f : X \to X$ a (ρ, a)-contraction. Then $F_f = \{x^*\}$ and for all $x_0 \in X$, $\{f^n(x_0)\}$ converges to x^*.*

Proof. We take $R_n : = \{(x, y) \in X \mid d(x, y) \leqslant a^n \ \delta(X)\}$.

From the Theorem 8 we have

THEOREM 10. *Let X be a Hilbert space and $f : \bar{B}(0; R) \to X$, a strict (δ, a)-contraction. If $x \in \bar{B}(0, R)$, $f(x) = \lambda(x)$ imply $\lambda \leqslant 1$, then $F_f = \{x^*\}$.*

Proof. Let $\rho : X \to \bar{B}(0; R)$ be the radial retraction. The mapping f is retractible onto $\bar{B}(0; R)$ by means of ρ. The proof follows from the theorem 8.

4.2. Uniform spaces (see [7]). Let (X, U) be a Hausdorff separable complete uniform space where U is a uniform base $\{R_n\}_{n \in \mathrm{N}}$ such that $X \times X = R_0 \supset R_1 \supset \ldots \supset R_n \supset \ldots$ From the Theorem 5 we have

THEOREM 11 (see [10], [12], [16], [17], [25]). *Let $f : X \to X$ be a mapping such that for all $n \in \mathrm{N}$, $(x, y) \in R_n$ implies $(f(x), f(y)) \in R_{n+1}$. Then $F_f = \{x^*\}$ and $f^n(x_0) \xrightarrow{U} x^*$, for all $x_0 \in X$.*

5. **Remarks.** *Remark 1.* For other fixed point theorems in sets see: [1], [2], [9], [19], [22].

Remark 2. For some fixed point theorems in order set see: [3], [4], [8], [11], [13], [15], [18] — [20], [23], [24].

Remark 3. For a categorical point of view in the fixed point theory see [19] — [21].

Remark 4 (see [14]). Let X be a set, $R \subset X \times X$ and $f: X \to X$ a mapping. By definition $x \in X$ is a R-fixed point of f if $(x, f(x)) \in R$. If we take R_n, $n \in \mathbb{N}$, such that $\bigcap_0^\infty R_n = R \supset \Delta(X)$, we obtain a set-theoretic approach for the R-fixed point theory.

6. Problem. The Eilenberg's theorem has applications in automata theory. Are the theorems 5, 6 and 8 applications in automata theory?

REFERENCES

1. A. Abian, *A fixed point theorem*, Nieuw Archief voor Wiskunde, 16 (1968), 184–185.
2. A. Abian, *A fixed point theorem for mappings*, J. Math. Anal. Appl., 24 (1968), 146–1
3. K. Baclawski, *Combinatorics: trend and examples*, in New directions in applied mathematics (P. J. Hilton and G. S. Young ed.) 1–10, Springer–Verlag, Berlin, 1981.
4. K. Balcawski, A. Björner, *Fixed points in partially ordered sets*, Advances in Math. 31 (1979), 263–287.
5. R. F. Brown, *Retraction mapping principle in Nielsen fixed point theory*, Pacific J. Math. 115 (1984), 277–297.
6. R. Cristescu, *Structuri de ordine în spaţii liniare normate*, Ed. Şt. şi Enciclop., Bucureş 1983.
7. A. Császár, *General topology*, Akadémiai Kiadó, Budapest, 1978.
8. M. Deaconescu, *Fixed points of decreasing contractions*, Preprint Nr. 3 (Cluj-Napoca 1984, 24–28.
9. M. Deaconescu, *The fixed point set for injective mappings*, Stud. Univ. „Babeş–Bolyai 29 (1984), 13–15.
10. E. Dubinski, *Fixed points in non-normal spaces*, Ann. Acad. Sc. Fennicae, Sercie A. I 331, 1963.
11. J. Dugundji, A. Granas, *Fixed point theory*, Vol. 1, PWN, Warszawa, 1982.
12. O. Hadžić, *Fixed point theory in topologica vector spaces*, Novi Sad, 1984.
13. E. M. Jawhari, D. Misane, M. Pauzet, *Retracts: graphs and ordered sets from metric point of view*, Contemporary Math., 57 (1986), 175–226.
14. V. Klee, *Stability of the fixed point property*, Colloq. Math., 8 (1961), 43–46.
15. S. C. Kleene, *Introduction to metamathematic*, North–Holland, Amsterdam, 1952.
16. J. C. Mathews, D. W. Curtis, *Relatively contractive relations in pairs of generalized* form spaces, J. London Math. Soc., 44 (1969), 100–106.
17. J. Reinermann, *Über das Verfahren der sukzessiven Näherung in der Fixpunkt-the kontrahierender Abbildungen*, Köln, 1970.
18. I. Rival, *The problem of fixed points in ordered sets*, Ann. Discrete Math., 8 (1980), 283–2
19. I. A. Rus, *Teoria punctului fix în structuri algebrice*, Uriv. Cluj, 1971.
20. I. A. Rus, *Principii şi aplicaţii ale teoriei punctului fix*, Ed. Dacia, Cluj-Napoca, 197
21. I. A. Rus, *Punct de vedere categorial în teoria punctului fix*, Sem. intinerant de ecuaţii fun onale, aproximare şi convexitate, Timişoara, 1980, 205–209.
22. I. A. Rus, *Generalized contractions*, Preprint Nr. 3 (Cluj-Napoca), 1983, 1–130.
23. I. A. Rus, *Retraction method in the fixed point theory in ordered structures*, Preprint N (Cluj-Napoca), 1988, 1–8.
24. O. Stănăşilă, *Noţiuni şi tehnici de matematică discretă*, Ed. Şt., Encicl. Bucureşti, 1
25. P. V. Subrahmanyam, *A new approach to fixed points and coincidences*, Bull. Acad. Sc., 25 (1977), 776–780.
26. K. Wišniewski, *On functions without fixed points*, Ann. Soc. Mat. Pol., 17 (1973), 227–

STUDIA UNIV. BABEŞ–BOLYAI, MATHEMTICA, XXXIII, 3, 1988

A SHEPARD-TAYLOR APPROXIMATION FORMULA

GH. COMAN* and L. ŢÂMBULEA*

Received: January 5, 1989

REZUMAT. — O formulă de aproximare Shepard—Taylor. În lucrare se construieşte o formulă de interpolare de tip Shepard de grad de exactitate $m-1$, $m \in \mathbb{N}^*$. Pe un exemplu concret se ilustrează comportarea funcţiei interpolatoare, pentru $m = 1, 2$, în raport cu parametrul μ, parametru ce apare în construcţia acesteia.

Let P_k, $P_k = (x_k, y_k)$, $k = \overline{0, n}$, be distinct points in $\mathbf{R}^2$ and $D = [a, b] \times [c, d]$ be a rectangle that contains all these points P_k. Let, also, f be a real-valued function defined on D and such that there exist the derivatives $f^{(i,j)}(x_k, y_k)$, $k = \overline{0, n}$; $i, j \in \mathbb{N}$, $i + j \leqslant m$ with $m \geqslant 1$.

The goal of this paper is to study the following interpolation formula

$$f = \sum_{k=0}^{n} A_k(T_m f) + R_m f \tag{1}$$

where A_k is the Shepard's function:

$$A_k(x, y) = \prod_{\substack{i=0 \\ i \neq k}}^{n} [r_i(x, y)]^{\mu} \Big/ \left(\sum_{i=0}^{n} \prod_{\substack{j=0 \\ j \neq i}}^{n} [r_j(x, y)]^{\mu} \right),$$

with $r_i(x, y)$ the distance between the points (x, y) and (x_i, y_i), $\mu \in \mathbf{R}_+$ and $T_m f$ is the bivariate Taylor interpolation polynomial:

$$(T_m f)(x, y) = \sum_{i+j<m} \frac{(x - x_k)^i}{i!} \frac{(y - y_k)^j}{j!} f^{(i,j)}(x_k, y_k).$$

The formula (1) is named a Shepard—Taylor approximation formula. The corresponding interpolation operator is denoted by ST_m, i.e.

$$ST_m f = \sum_{k=0}^{n} A_k(T_m f).$$

THEOREM 1. If $\mu \geqslant m$ then $(ST_m f)^{(i,j)}(x_k, y_k) = f^{(i,j)}(x_k, y_k)$, $k = \overline{0, n}$; $i, j \in \mathbf{N}$, $i + j < m$.

* *University of Cluj-Napoca, Faculty of Mathemtics and Physics, 3400 Cluj-Napoca, Romania.*

Proof. First, we observe that $A_k^{(i,j)}(x_\nu, y_\nu) = 0$ for $\nu = \overline{0, n}$, $\nu \neq k$ and $i, j \in \mathbf{N}$, $1 \leqslant i + j < m$. To this end, one considers $A_k(x, y) = g_k(x, y)/h_k(x, y)$ where

$$g_k(x, y) = \prod_{\substack{i=0 \\ i \neq k}}^{n} [r_i(x, y)]^\mu; \quad h_k(x, y) = \sum_{i=0}^{n} \prod_{\substack{j=0 \\ j \neq i}}^{n} [r_i(x, y)]^\mu.$$

By a straightforward computation one observes that $g_k^{(i,j)}(x_\nu, y_\nu) = 0$, $\nu = \overline{0, n}$, $\nu \neq k$ and $g_k^{(i,j)}(x_k, y_k) = h_k^{(i,j)}(x_k, y_k)$ for $i, j \in \mathbf{N}$, $1 \leqslant i + j < m$, if $\mu \geqslant m$. Hence, $A_k^{(i,j)}(x_\nu, y_\nu) = 0$, $\nu = \overline{0, n}$, $\nu \neq k$ and $i, j \in \mathbf{N}$, $1 \leqslant i + j < m$. So,

$$(ST_m f)^{(i,j)}(x_\nu, y_\nu) = \sum_{k=0}^{n} (A_k \cdot T_m f)^{(i,j)}(x_\nu, y_\nu) =$$

$$= \sum_{k=0}^{n} A_k(x_\nu, y_\nu)(T_m f)^{(i,j)}(x_\nu, y_\nu) = f^{(i,j)}(x_\nu, y_\nu),$$

for all $\nu = \overline{0, n}$ and $i, j \in \mathbf{N}$, $i + j < m$.

THEOREM 2. $ST_m f = f$ for all $f \in \mathrm{P}_{m-1}^2$ (the set of all bivariate polynomials of the total degree at most $m - 1$).

Proof. Let $P \in \mathbf{P}_{m-1}^2$ be given. Then $T_m P = P$ and, taking into account that $\sum_{k=0}^{n} A_k(x, y) = 1$,

$$(ST_m P)(x, y) = \sum_{k=0}^{n} A_k(x, y)(T_m P)(x, y) = P(x, y) \sum_{k=0}^{n} A_k(x, y) = P(x, y).$$

Next, one considers the interpolation formula generated by the operator ST_m:

$$f = ST_m f + RT_m f.$$

THEOREM 3. If $f \in \mathbf{B}_{p,q}(a, c)[2]$ with $p + q = m$, then

$$(RT_m f)(x, y) = \sum_{j<q} \int_a^b K_{m-j,j}(x, y; s)\, f^{(m-j,j)}(s, c)ds +$$

$$+ \sum_{i<p} \int_c^d K_{i,m-i}(x, y; t) f^{(i,m-i)}(a, t)dt + \iint_D K_{p,q}(x, y; s, t)\, f^{(p,q)}(s, t)dsdt,$$

where

$$K_{m-j,j}(x, y; s) = \frac{(y - c)^j}{(m - j - 1)!j!} \left\{ (x - s)_+^{m-j-1} - \sum_{k=0}^{n} A_k(x, y)[(x - s)_+^{m-j-1} + (x_k - s)_+^{m-j-1}] \right\}, \quad j < q$$

$$K_{i,m-i}(x, y; t) = \frac{(x-a)^i}{(m-i-1)!\,i!}\left\{(y-t)_+^{m-i-1} - \sum_{k=0}^{n} A_k(x, y)\,[(y_k - t) \overset{0}{+} (y - y_k) + (y_k - t)_+]^{m-i-1}\right\}, \quad i < p$$

$$K_{p,q}(x, y; s, t) = \frac{1}{(p-1)!\,(q-1)!}\left\{(x-s)_+^{p-1}(y-t)_+^{q-1} - \sum_{k=0}^{n} A_k(x, y)\right.$$

$$\left.\cdot\,[(x_k - s) \overset{0}{+} (x - x_k) + (x_k - s)_+]^{p-1}[(y_k - t) \overset{0}{+} (y - y_k) + (y_k - t)_+]^{q-1}\right\}.$$

More than that, if $f^{(m-j,j)}(.,c) \in C[a, b]$, for $j < q$; $f^{(i,m-i)}(0,.) \in C[c, d]$, for $i < p$, $f^{(p,q)} \in C(D)$ and $||\cdot||$ is the uniform norm, then

$$|(RT_m f)(x, y)| \leqslant \sum_{j<q} H_{m-j,j}(x, y)\,||f^{(m-j)}(.,c)|| + \sum_{i<p} H_{i,m-i}(x, y)\,||f^{(i,m-i)}(a,.)|| + H_{p,q}(x, y)\,||f^{(p,q)}||,$$

where

$$H_{m-j,j}(x, y) = \frac{(y-c)^j}{j!\,(m-j)!}\sum_{k=0}^{n} A_k(x, y)(x - x_k)^{m-j}, \text{ for } m-j \text{ even and}$$

$$H_{m-j,j}(x, y) = \frac{(y-c)^j}{j!\,(m-j)!}\sum_{k=0}^{n} A_k(x, y)\{[1 - 2(x_k - x) \overset{0}{+} 1](x - x_k)^{m-j} + 2[(x_k - x) \overset{0}{+} (x - x_k) + (x_k - x)_+]^{m-j}\}$$

if $m-j$ is odd,

$$H_{i,m-i}(x, y) = H_{m-i,i}(y, x), \text{ for } c = a$$

and

$$H_{p,q}(x, y) = \frac{1}{p!\,q!}\left\{(x-a)^p(y-c)^q - \sum_{k=0}^{n} A_k(x, y)\,[g(a)h(c) - 2g(x)h(y)]\right\},$$

for p and q even numbers,

$$H_{p,q}(x, y) = \frac{1}{p!\,q!}\left\{(x-a)^p(y-c)^q - \sum_{k=0}^{n} A_k(x, y)\,[g(a)h(c) - 2g(a)h(y) + 2g(x)h(y)]\right\}$$

for p even and q odd,

$$H_{p,q}(x, y) = \frac{1}{p!\,q!}\left\{(x-a)^p(y-c)^q - \sum_{k=0}^{n} A_k(x, y)\,[g(a)h(c) - 2g(x)h(c) + 2g(x)h(y)]\right\}$$

for p odd and q even, respectively.

$$H_{p,q}(x,y)=\frac{1}{p!\,q!}\Big\{(x-a)^p(y-c)^q-\sum_{k=0}^{n}A_k(x,y)\,[g(a)h(c)-2g(a)h(y)-$$
$$-2g(x)h(c)+2g(x)h(c)]\Big\}$$

if both p and q are odd, with

$$g(a)=(x-a)^p-(x-x_k)^p$$
$$g(x)=[(x_k-x)^0_+(x-x_k)+(x_k-x)_+]^p-(x-x_k)^p$$
$$h(c)=(y-c)^q-(y-y_k)^q$$
$$h(y)=[(y_k-y)^0_+(y-y_k)+(y_k-y)_+]^q-(y-y_k)^q.$$

Proof. From theorem 2 it follows that $RT_mf=0$ for all $f\in\mathbf{P}^2_{m-1}$. S for $f\in\mathbf{B}_{p,q}(a\cdot c)$, we can apply the Peano's theorem [2], and the integ representation (2) follows. From (2), one obtains

$$|(RT_mf)(x,y)|\leqslant\sum_{j<q}\frac{(y-c)^j}{j!}\,||f^{(m-j,j)}(.,c)||\int_a^b|G_{m-j,j}(x;s)|ds+$$
$$+\sum_{i<p}\frac{(x-a)^i}{i!}\,||f^{(i,m-i)}(a,.)||\int_c^d|G_{i,m-i}(y;t)|dt+||f^{(p,q)}||\iint_D|K_{p,q}(x,y;s,t)|dsdt$$

where

$$G_{m-j,j}(x;s)=\frac{1}{(m-j-1)!}\Big\{(x-s)^{m-j-1}_+-\sum_{k=0}^{n}A_k(x,y)\,[(x_k-s)^0_+(x-x_k)+$$
$$+(x_k-s)_+]^{m-j-1}\Big\}$$

and $G_{i,m-i}(y;t)=G_{m-i,i}(y,t)$.

Now, let us suppose that the given points (x_k,y_k), $k=\overline{0,n}$, are suc that $x_{k-1}<x_k$. Thus, if $G'_{m-j,j}(x,.)=G_{m-j,j}(x;.)|_{[x_{r-1},x_r]}$ (the restriction the function $G_{m-j,j}$ to the interval $[x_{r-1},x_r]$), then we observe that

$$G^s_{m-j,j}(x;s)=\frac{1}{(m-j-1)!}\begin{cases}(x-s)^{m-j-1}\sum_{k=0}^{r-1}A_k(x,y), & \text{for } s\leqslant x\\ (-1)^{m-j}(s-x)^{m-j-1}\sum_{k=r}^{n}A_k(x,y), & \text{for } s>x.\end{cases}$$

Hence, $G_{m-j,j}\geqslant 0$ for $m-j$ even and $G_{m-j,j}\geqslant 0$ if $s\leqslant x$ and $G_{m-j,j}\leqslant$ if $s>x$, for $m-j$ odd. So,

$$\int_a^b|G_{m-j,j}(x;s)|ds=\frac{1}{(m-j)!}\sum_{k=0}^{n}A_k(x,y)(x-x_k)^{m-j},$$

for $m-j$ even, respectively

$$\int_a^b |G_{m-j,j}(x;s)|\,ds = \frac{1}{(m-j)!}\sum_{k=0}^{n} A_k(x,y)\{[1-2(x_k-x)_+^0](x-x_k)^{m-j} +$$

$$+ 2[(x_k-x)_0^+(x-x_k) + (x_k-x)_+]^{m-j}\}$$

for $m-j$ odd, and the expression of the function $H_{m-j,j}$ follows. In an analoyous way, one obtains the function $H_{i,m-i}$. In order to determine the function $H_{p,q}$;

$$H_{p,q}(x,y) = \iint_D K_{p,q}(x,y;s,t)\,ds\,dt, \tag{4}$$

we must study the sign of the function $K_{p,q}$ on D. To do this, we introduce the notations: $D_{xy} = [a, x]\,x(x, y]$, $D_{xd} = [a, x]\times[y, d]$, $D_{by} = [x, b]\times[c, y]$, $D_{bd} = [x, b]\times[y, d]$. Now, one considers the grid defined by the lines that passed through the points (x_k, y_k), $k = \overline{0, n}$, and are parallel to the coordinate axes. In the hypothesis that the points (x_k, y_k) are such that $x_{k-1} < x_k$, $k = \overline{1, n}$, one denotes by D_ν the element $[x_\nu, x_{\nu+1}]\times[y_\nu, y_l]$ with $l \in \{1, \ldots, n\}$ of this grid and by $K^\nu_{p,q}(x, y;.,.)$ the restriction of the function $K_{p,q}(x, y;.,.)$ to D_ν. We have

$$K^\nu_{p,q}(x,y;s,t) = \begin{cases} (x-s)^{p-1}(y-t)^{q-1}\sum\limits_{k\in I_\nu} A_k(x,y), & (s,t)\in D_{xy} \\ -(x-s)^{p-1}(y-t)^{q-1}\sum\limits_{k\in J_\nu} A_k(x,y), & (s,t)\in D-D_{xy}, \end{cases}$$

where $I_\nu = \{\mu\,|\,(x_\mu, y_\mu)\in D_{bd},\ \mu\in\{\nu,\ldots,n\}\}$ and $J_\nu = \{0,\ldots,n\} - I_\nu$. It follows that: 1) if p and q are even numbers then $K_{p,q}\geqslant 0$ on $D - D_{bd}$ and $K_{p,q}\leqslant 0$ on D_{xd}; 2) if p is even and q is odd then $K_{p,q}\geqslant 0$ on $D - D_{xd}$ and $K_{p,q}\leqslant 0$ on D_{xd}; 3) if p is odd and q is even then $K_{p,q}\geqslant 0$ on $D - D_{by}$ and $K_{p,q}\leqslant 0$ on D_{by}; 4) if both p and q are odd numbers then $K_{p,q}\geqslant 0$ on D_{xy} and $K_{p,q}\leqslant 0$ on $D - D_{xy}$. Taking into account the sign of the function $K_{p,q}$ in the relation (4), the proof follows by a straightforward computation.

Next, we illustrate the behaviour of the approximation function ST_mf with to the number μ for $m = 1, 2$ and for the function f given by $f(x, y) = -(x^2+y^2)$ on the domain $D = [-1,1]\times[-1,1]$, with the interpolating points: $P_1 = (-1,-1)$; $P_2 = (-1,1)$; $P_3 = (1,-1)$; $P_4 = (1,1)$; $P_5 = (-1/2, -1/2)$; $P_6 = (-1/2, 1/2)$; $P_7 = (1/2, -1/2)$; $P_8 = (1/2, 1/2)$; $P_9 = (0,0)$. In the following figures there are given the graph of the function f respectively the graphs of the approximations ST_1f and ST_2f for $\mu = 1, 2, 4$.

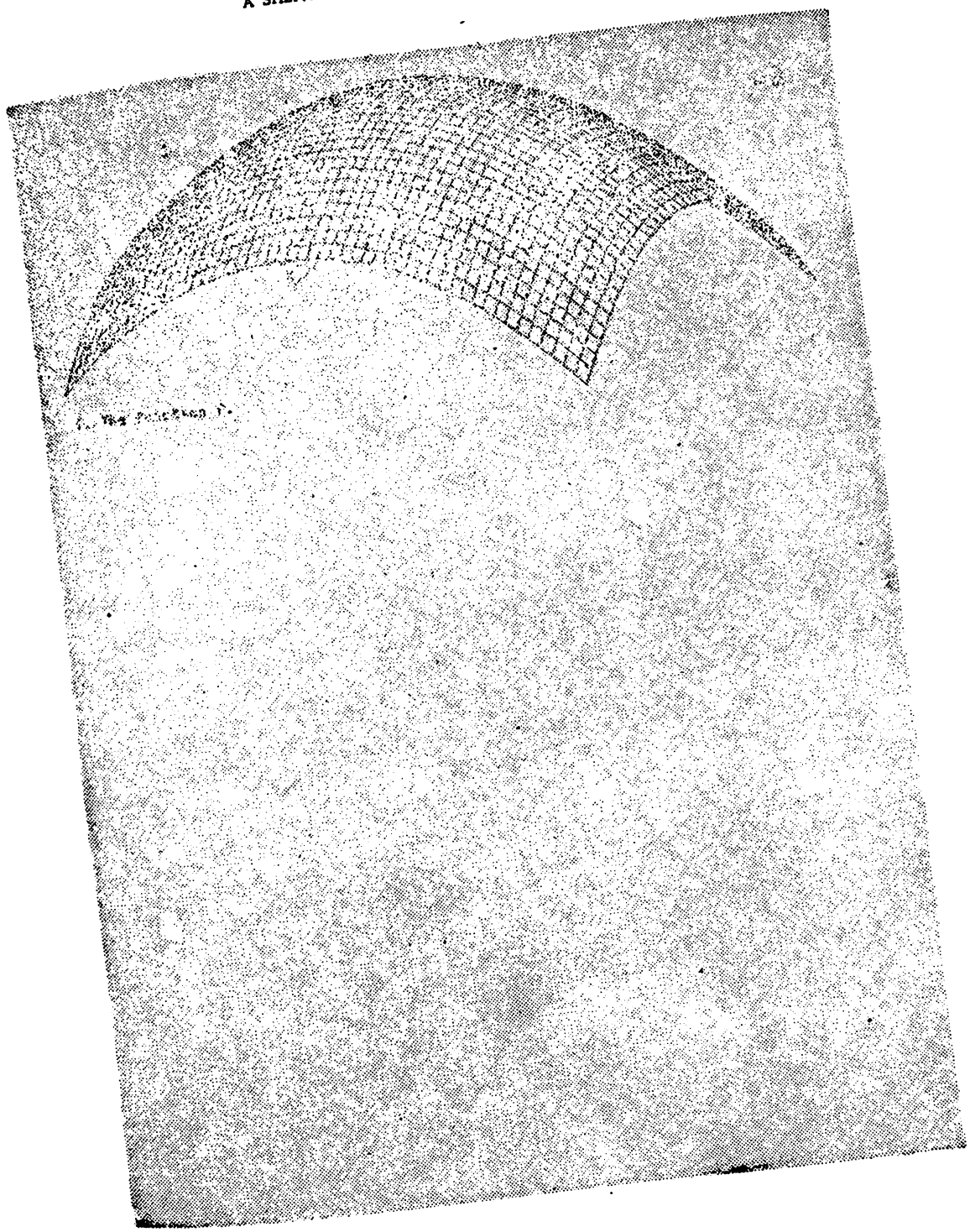
As, it can be sen the remark from the paper [4] is comfirmed; i.e. for $0<\mu\leqslant 1$ ST_1f has cusps at the data pounts (x_i, y_i) and for $\mu>1$ it has flat

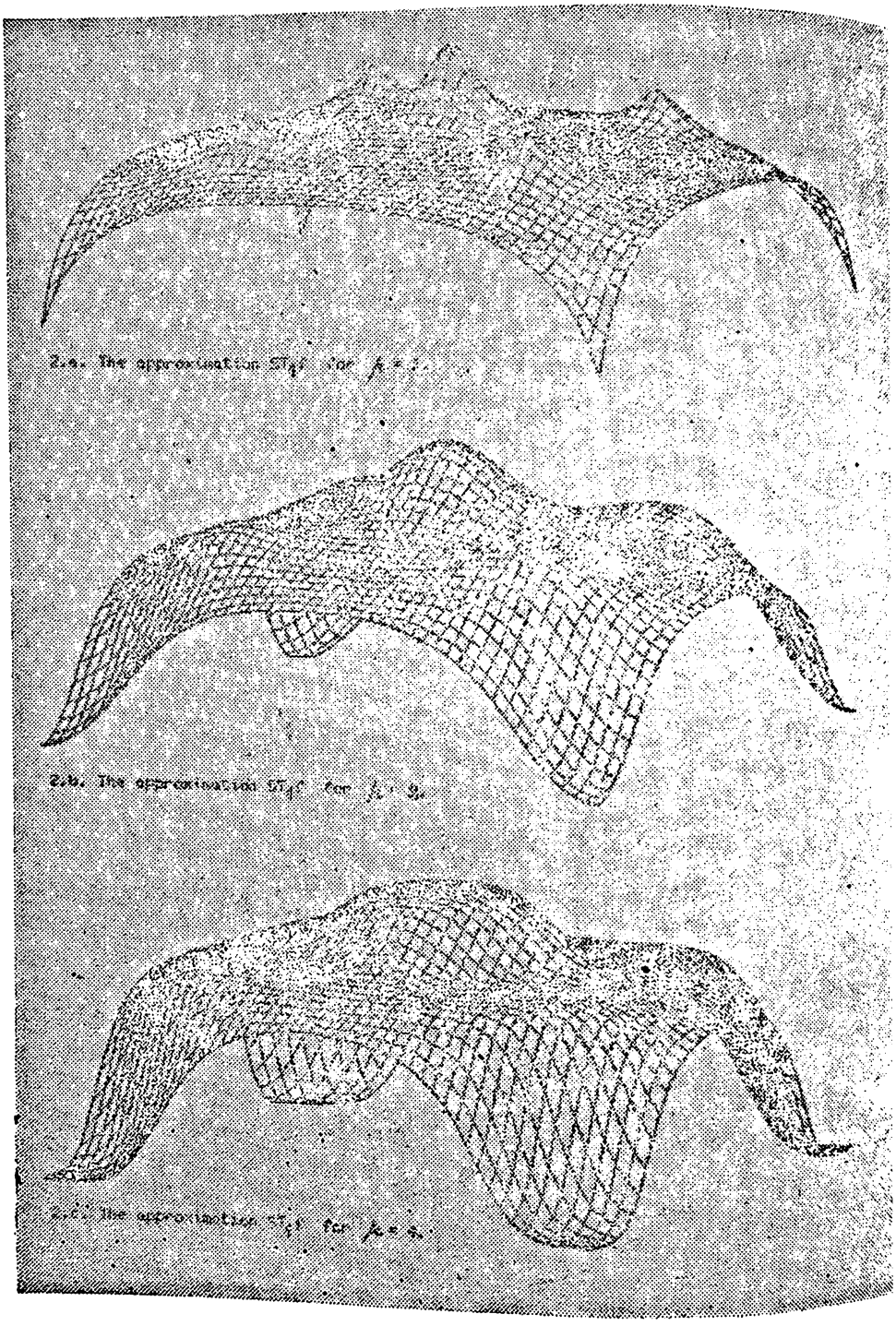
spots at these points. If μ is relatively large, then the surface $z = (ST_1 f)(x, y)$ tends to become very flat near the data points.

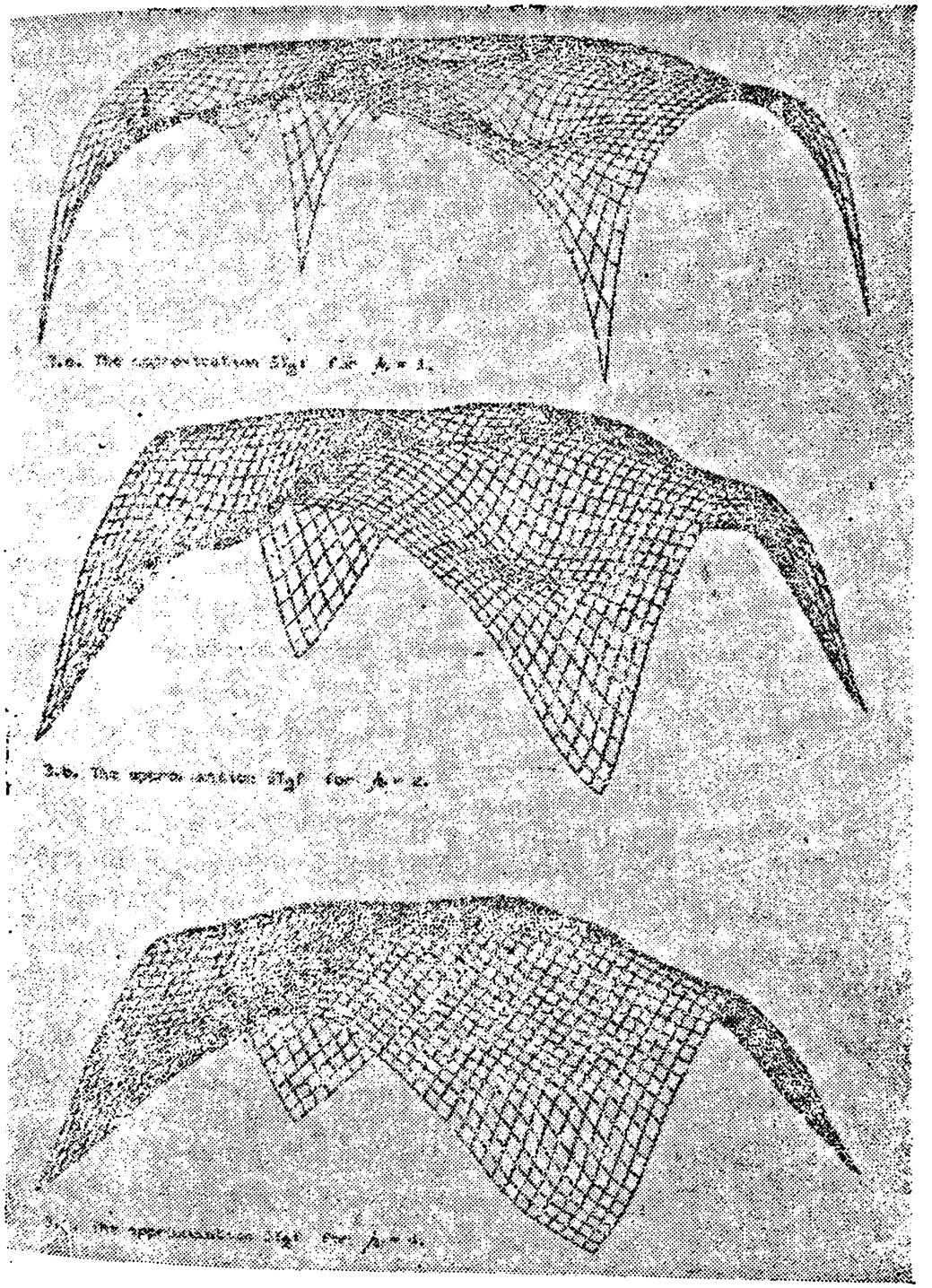
The functions $ST_2 f$ has the same behavior but with a diminishing cusps and flat vicinity.

REFERENCES

1. Gh. Coman, *Shepard—Taylor interpolation*, "Babeş—Bolyai" University, Faculty of Mathematics and Physics, Research Seminars, Preprint Nr. 6, 1988, 5—14.
2. A. Sard, *Linear Approximation*, "Amer. Math. Society", Providence 1963.
3. D. Shepard, *A two-dimensional interpolation function for iregularly spaced data*, "Proc. 1964 ACM Nat. Conf.", 517—524.
5. L. L. Schumaker, *Fitting surfaces to scattered data*, In: "Approximation Theory II" (ed. by G. G. Lorentz, C. K. Chui, L. L. Schumaker), Acad. Press, Inc. 1976, 203—268.







STUDIA UNIV. BABEŞ–BOLYAI, MATHEMATICA, XXXIII, 3, 1988

A CHARACTERIZATION OF ELEMENTS OF BEST APPROXIMATION IN REAL NORMED LINEAR SPACES

SEVER SILVESTRU DRAGOMIR*

Received: April 20, 1988

REZUMAT. — O caracterizare a elementelor de cea mai bună aproximare în spaţii liniare normate reale. În lucrare se dă o caracterizare a elementelor de cea mai bună aproximare în spaţii liniare normate reale folosind funcţionale liniare şi continue.

1. Introduction. Let $(X, ||\cdot||)$ be a real normed linear space and the mappings $\langle , \rangle_i, \langle , \rangle_s : X \times X \to \mathbf{R}$ given by

$$\langle x, y\rangle_i = \lim_{t \uparrow 0} \frac{\|y + tx\|^2 - \|y\|^2}{2t}, \quad x, y \in X; \tag{1}$$

$$\langle x, y\rangle_s = \lim_{t \downarrow 0} \frac{\|y + tx\|^2 - \|y\|^2}{2t}, \quad x, y \in X \tag{2}$$

(see [3] p. 35).

For details concerning the properties of these mappings we send to [3] p. 38 and [1] p. 389.

Another functional in connection with $\langle,\rangle_i$ and $\langle,\rangle_s$ is the following:

$$\tau(x, y) = \lim_{t \downarrow 0} \frac{\|x + ty\| - \|x\|}{t}, \quad x, y \in X, \quad [4] \text{ p. } 82. \tag{3}$$

It is easy to see that $\|x\|\tau(x, y) = \langle y, x\rangle_s = -\langle -y, x\rangle_i$, $x, y \in X$.

1.1. DEFINITION. The element $x \in X$ is called *orthogonal in Birkhoff sens* over $y \in X$ iff $\|x + ty\| \geq \|x\|$, for all $t \in \mathbf{R}$. We note that $x \perp y$.

In paper [2] R. C. James proves the following result:

1.2. THEOREM. *Let $(X, ||\cdot||)$ be a real normed linear space. Then the following assertions are equivalent*

$$x \perp (\alpha x + y) \tag{4}$$

$$-\tau(x, -y) \leq \alpha\|x\| \leq \tau(x, y). \tag{5}$$

It is easy to see that the relation (5) is equivalent with

$$\langle y, x\rangle_i \leq \alpha\|x\|^2 \leq \langle y, x\rangle_s \tag{6}$$

what means that $x \perp y$ iff

$$\langle y, x\rangle_i \leq 0 \leq \langle y, x\rangle_s. \tag{7}$$

* *Secondary School, Băile Herculane, 1600 Băile Herculane, Caraş-Severin County, Romania.*

Now let G be a proper linear subspace not dense in X and $\mathfrak{P}_G(x) := \{y_0 \mid \|y_0 - x\| = \inf_{y \in G} \|y - x\|\} \subset G$ the set of elements of best approximation reffering to $x \in X - \bar{G}$.

1.3. DEFINITION. The proper linear subspace $E \subset X$ is called *proximinal* in X iff for every $x \in X$ the set $\mathfrak{P}_G(x)$ is non-void.

Finally, we present the well-known result which give a caracterisation of elements of best approximation in terms of Birkhoff orthogonality [4] p. 85:

1.4. LEMMA. *Let $(X, \|\cdot\|)$ be a normed linear space, G a linear subspace in X, $x \in X - \bar{G}$ and $g \in G$. Then $g \in \mathfrak{P}_G(x)$ iff $x - g \perp G$.*

For details concerning the theory of elements of best approximation in normed linear spaces we send to excelent monography of Ivan Singer [4].

2. The Characterization of Elements of Best Approximation.

The main purposes of this section are to give a characterization of elements of best approximation in real normed linear spaces in terms of continuous linear functionals and to apply this result in theory of continuous linear functionals representation on smooth and/or reflexive normed linear spaces.

2.1. THEOREM. *Let $(X, \|\cdot\|)$ be a real normed linear space $f \in X^*$, $f \neq 0$, $g_0 \in \operatorname{Ker}(f)$ and $x_0 \in X - \operatorname{Ker}(f)$. Then the following sentences are equivalent*

(i) $g_0 \in \mathfrak{P}_{\operatorname{Ker}(f)}(x_0)$;

(ii) *for every $x \in X$ we have*

$$\left\langle x, \frac{f(x_0)(x_0 - g_0)}{\|x_0 - g_0\|^2} \right\rangle_i \leqslant f(x) \leqslant \left\langle x, \frac{f(x_0)(x_0 - g_0)}{\|x_0 - g_0\|^2} \right\rangle_s \tag{1}$$

and

$$\|f\| = \frac{|f(x_0)|}{\|x_0 - g_0\|}. \tag{2}$$

Proof. "(i) $\Rightarrow$ (ii)". If $g_0 \in \mathfrak{P}_{\operatorname{Ker}(f)}(x_0)$, then by Lemma 1.4. we have $x_0 - g_0 \perp \operatorname{Ker}(f)$. Putting $w := x_0 - g_0$ and since $f(x)w - f(w)x \in \operatorname{Ker}(f)$ for every $x \in X$, we obtain $w \perp (f(x)w - f(w)x)$ what implies

$$\langle f(x)w - f(w)x, w\rangle_i \leqslant 0 \leqslant \langle f(x)w - f(w)x, w\rangle_s, \quad x \in X.$$

Using the properties of $\langle ,\rangle_i$, $\langle ,\rangle_s$ [3] pp. 38, we obtain

$$\langle f(x)w - f(w)x, w\rangle_p = f(x)\|w\|^2 + \langle -f(w)x, w\rangle_p, \quad x \in X, \ p = s \text{ or } p = i.$$

Since $w \perp \operatorname{Ker}(f)$, $w \neq 0$, we have $f(w) \neq 0$ what implies a) $f(w) > 0$ or b) $f(w) < 0$.

a). If $f(w) > 0$, we obtain

$0 \geqslant f(x)\|w\|^2 + \langle -f(w)x, w\rangle_i = f(x)\|w\|^2 - \langle x, f(w)w\rangle_s$ from where results

$$f(x) \leqslant \left\langle x, \frac{f(w)}{\|w\|^2} w \right\rangle_s, \quad x \in X. \tag{3}$$

Similarly, we have $0 \leqslant f(x)\|w\|^2 + \langle -f(w)x, w\rangle_s = f(x)\|w\|^2 - \langle x, f(w)w\rangle_i$ from where result

$$f(x) \geqslant \left\langle x, \frac{f(w)}{\|w\|^2} w \right\rangle_i, \quad x \in X.$$

b). Firstly, we remark that for every $y, z \in X$ we have

$\langle y, z\rangle_i = -\langle -y, z\rangle_s = -\langle y, -z\rangle_s$ and $\langle y, -z\rangle_i = -\langle -y, -z\rangle_s = -\langle y, z\rangle_s$.

If $f(w) < 0$, applying the properties of $\langle , \rangle_{i,s}$ we obtain $0 \geqslant f(x)\|w\|^2 + \langle -f(w)x, w\rangle_i = f(x)\|w\|^2 - \langle x, f(w)w\rangle_s$, from where results

$$f(x) \leqslant \left\langle x, \frac{f(w)}{\|w\|^2} w \right\rangle_s, \quad x \in X.$$

Similar, we have

$$f(x) \geqslant \left\langle x, \frac{f(w)}{\|w\|^2} w \right\rangle_i, \quad x \in X.$$

Since $f(w) = f(x_0)$ by the relations (3), (4), (5), (6) we obtain (1).

Now, let be $u := \frac{f(x_0)(x_0 - g_0)}{\|x_0 - g_0\|^2}$. Then we have $f(x) \leqslant \langle x, u\rangle_s \leqslant \|x\|\|u\|$, $x \in X$ and $f(x) \geqslant \langle x, u\rangle_i = -\langle -x, u\rangle_s \geqslant -\|x\|\|u\|$, $x \in X$, from where results $\|f\| \leqslant \|u\|$.

On the other hand $\|f\| \geqslant \frac{f(u)}{\|u\|} \geqslant \frac{\langle u, u\rangle_i}{\|u\|} = \|u\|$ what implies the relation (2).

"(ii) $\Rightarrow$ (i)". By the relation (1) it results

$$\left\langle x, \frac{f(x_0)(x_0 - g_0)}{\|x_0 - g_0\|^2} \right\rangle_i \leqslant 0 \leqslant \left\langle x, \frac{f(x_0)(x_0 - g_0)}{\|x_0 - g_0\|^2} \right\rangle_s, \quad x \in \mathrm{Ker}(f).$$

Since $f(x_0) \neq 0$ we have $x_0 - g_0 \perp \mathrm{Ker}(f)$ what implies $g_0 \in \mathfrak{P}_{\mathrm{Ker}(f)}(x_0)$.

2.2. COROLARY. *Let $(X, \|\cdot\|)$ be a real normed linear space with a Gateaux differentiable norm, $f \in X^*$, $f \neq 0$, $g_0 \in \mathrm{Ker}(f)$ and $x_0 \in X - \mathrm{Ker}(f)$. Then the following sentences are equivalent:*

(i) $g_0 \in \mathfrak{P}_{\mathrm{Ker}(f)}(x_0)$;

(ii) *for every $x \in X$ we have*

$$f(x) = \left\langle x, \frac{f(x_0)(x_0 - g_0)}{\|x_0 - g_0\|^2} \right\rangle_s, \quad x \in X$$

and

$$\|f\| = \frac{|f(x_0)|}{\|x_0 - g_0\|}.$$

CONSEQUENCES.

1. Let $(X, \|\cdot\|)$ be a real reflexive Banach space. Then for every $f \in X^*$, $f \neq 0$ and $x_0 \in X - \mathrm{Ker}(f)$ there exists $g_0 \in \mathrm{Ker}(f)$ such that the relations (1) and (2) are satisfied. In addition, if $(X, \|\cdot\|)$ is a smooth reflexive Banach space, then the relations (7) and (8) are satisfied.

2. Let $(X, (,))$ be a real Hilbert space. Then for every $f \in X^*$, $f \neq 0$ and $x_0 \in X - \mathrm{Ker}(f)$ there exists $g_0 \in \mathrm{Ker}(f)$ such that

$$f(x) = \left(x, \frac{f(x_0)(x_0 - g_0)}{\|x_0 - g_0\|}\right), \quad x \in X, \quad \|f\| = \frac{|f(x_0)|}{\|x_0 - g_0\|}. \tag{9}$$

2.3. THEOREM. *Let $(X, \|\cdot\|)$ be a real normed linear space, G a closed linear subspace in X, $x_0 \in X - G$ and $g_0 \in G$. Then the following sentences are equivalent:*

(i) $g_0 \in \mathfrak{P}_G(x_0)$;

(ii) *for every $f \in (G \oplus [x_0])^*$ such that $G = \mathrm{Ker}(f)$ we have*

$$\left\langle x, \frac{f(x_0)(x_0 - g_0)}{\|x_0 - g_0\|^2} \right\rangle_i \leqslant f(x) \leqslant \left\langle x, \frac{f(x_0)(x_0 - g_0)}{\|x_0 - g_0\|^2} \right\rangle_s \tag{10}$$

for every $x \in G \oplus [x_0]$, and

$$\|f\| = \frac{|f(x_0)|}{\|x_0 - g_0\|}. \tag{11}$$

The proof is evident by Theorem 2.1..

2.4. COROLARY. *Let $(X, \|\cdot\|)$ be a smooth normed linear space, G a closed linear subspace in X, $x_0 \in X - G$ and $g_0 \in G$. Then the following sentences are equivalent:*

i) $g_0 \in \mathfrak{P}_G(x_0)$;

ii) *for every $f \in (G \oplus [x_0])^*$ such that $G = \mathrm{Ker}\, f$) we have*

$$f(x) = \left\langle x, \frac{f(x_0)(x_0 - g_0)}{\|x_0 - g_0\|^2} \right\rangle_s, \quad x \in G \oplus [x_0] \tag{12}$$

and

$$\|f\| = \frac{|f(x_0)|}{\|x_0 - g_0\|}. \tag{13}$$

3. The Characterization of Proximinal Linear Subspaces. The main purposes of this section are to give two characterizations of proximinal linear subspaces in a real normed linear space and to apply these results in theory of continuous linear functional representation on smooth and/or reflexive normed linear spaces.

3.1. THEOREM. *Let $(X, \|\cdot\|)$ be a real normed linear space and $f \in X^*$, $f \neq 0$. Then the following sentences are equivalent:*

(i) $\mathrm{Ker}(f)$ *is proximinal in* X;

(ii) *there exists* $u \in X$, $u \neq 0$ *such that*

$$\langle x, u\rangle_i \leqslant f(x) \leqslant \langle x, u\rangle_s, \quad x \in X \text{ and } \|f\| = \|u\|.$$

We use the following lemma:

3.2. LEMA. ([4] pp. 87). *Let $(X, \|\cdot\|)$ be a real normed linear space and H a hiperplan in X such that $0 \in H$. Then H is proximinal iff there exists $z \in X - \{0\}$ such that $z \perp H$.*

Theorem's proof. "(i) $\Rightarrow$ (ii)". If $\mathrm{Ker}(f)$ is proximinal, then there exists $w \in X - \{0\}$ such that $w \perp \mathrm{Ker}(f)$. Consequently (see Theorem 2.1.) for every $x \in X$ we have

$$\left\langle x, \frac{f(w)}{\|w\|^2} w \right\rangle_i \leqslant f(x) \leqslant \left\langle x, \frac{f(w)}{\|w\|^2} w \right\rangle_s, \quad x \in X \text{ and } \|f\| = \frac{|f(w)|}{\|w\|^2}\|w\|.$$

Putting $u := \dfrac{f(w)}{\|w\|^2} w$, one gets (1).

"(ii) $\Rightarrow$ (i)". It is evident.

3.3. COROLLARY. *Let $(X, \|\cdot\|)$ be a smooth normed linear space, $f \in X^*$, $f \neq 0$. Then the following sentences are equivalent*

(i) $\mathrm{Ker}(f)$ *is proximinal,*

(ii) *there exists* $u \in X$, $u \neq 0$ *such that*

$$f(x) = \langle x, u\rangle_s, \quad x \in X \text{ and } \|f\| = \|u\|.$$

CONSEQUENCES.

1. Let $(X, \|\cdot\|)$ be a real normed linear space, $f \in X^*$, $f \neq 0$ such that $S_f := \{x \in \mathrm{Ker}(f), \|x\| \leqslant 1\}$ is strictly sequential compact in $\sigma(X, X^*)$. Then there exists $u \in X$, $u \neq 0$ such that the relation (1) is satisfied. In addition if $(X, \|\cdot\|)$ is a smooth space, then the relation (2) is satisfied.

The proof is evident by Klee's theorem ([4] pp. 91) and by Theorem 3.1..

2. Let $(X, \|\cdot\|)$ be a real normed linear space and $f \in X^*$, $f \neq 0$. Then for every E a finite dimensional linear space in X there exists $u \in E$ such that

$$\langle x, u\rangle_i \leqslant f(x) \leqslant \langle x, u\rangle_s, \quad x \in X \text{ and } \|f\|_E = \|u\|.$$

In addition, if X is a smooth normed linear space then there exists $u \in E$ such that

$$f(x) = \langle x, u\rangle, \quad x \in E \text{ and } \|f\|_E = \|u\|.$$

3. Let $(X, \|\cdot\|)$ be a real reflexive Banach space. Then for every $f \in X^*$, $f \neq 0$ there exists $u \in X$, $u \neq 0$ such that the relation (1) is satisfied. If $(X, \|\cdot\|)$ is a smooth reflexive Banach space, then there exists $u \in X$, $u \neq 0$, such that the relation (2) is satisfied.

REMARK. The second part of consequence 3. represents the implication "(i) $\Rightarrow$ (ii)" of Tapia's theorem ([1] pp. 400) with an other proof.

Further, we shall give a theorem of characterization for the proximinal linear subspaces in a real normed linear space in terms of orthogonality.

3.4. THEOREM. *Let $(X, \|\cdot\|)$ be a real normed linear space and G a closed linear subspace in X. Then the following sentences are equivalent*

(i) *G is proximinal in X;*

(ii) *for every $x \in X$, there exists $x' \in G$ and $x'' \in G\perp$ such that*

$$x = x' + x''. \tag{5}$$

The proof results by Lemma 1.4. by simple computation. We omit the details.

CONSEQUENCES.

1. Let $(X, \|\cdot\|)$ be a real normed linear space and G a closed linear subspace in X such that $S_G := \{g \in G, \|g\| \leqslant 1\}$ is strictly sequential compact in $\sigma(X, X^*)$. Then for every $x \in X$ there exists $x' \in G$ and $x'' \in G\perp$ such that the relation (5) is satisfied.

2. Let $(X, \|\cdot\|)$ be a real normed linear space and G a finite dimensional linear subspace in X. Then for every $x \in X$ there exists $x' \in G$ and $x'' \in G$ such that the relation (5) is satisfied.

3. Let $(X, \|\cdot\|)$ be a real reflexive Banach space and G a closed linear subspace in X. Then for every $x \in X$ there exists $x' \in G$ and $x'' \in G\perp$ such that the relation (5) is satisfied.

Finally, we shall point out a theorem of characterization for the proximinal linear subspaces in a real normed linear space in terms of continuous linear functionals.

3.5. THEOREM. *Let $(X, \|\cdot\|)$ be a normed space and G a closed linear subspace in X. Then the following sentences are equivalent:*

(i) *G is proximinal in X;*

(ii) *for every $x_0 \in X - G$ and $f \in (G \oplus [x_0])^*$ such that $\mathrm{Ker}(f) = G$ there exists $u \in G \oplus [x_0]$ such that:*

$$\langle x, u\rangle_i \leqslant f(x) \leqslant \langle x, u\rangle_s, \quad x \in G \oplus [x_0] \text{ and } \|f\| = \|u\|. \tag{6}$$

In addition, if $(X, \|\cdot\|)$ is a smooth normed linear space then the relation (i) is equivalent with the following

(iii) *for every $x \in X - G$ and $f \in (G \oplus [x_0])^*$ such that $\mathrm{Ker}(f) = G$, there exists $u \in G \oplus [x_0]$ with the property*

$$f(x) = \langle x, u\rangle_s, \quad x \in G \oplus [x_0] \text{ and } \|f\| = \|u\|. \tag{7}$$

REFERENCES

1. G. Dinca, *Metode variaționale și aplicații*. Ed. Tehnică, București, 1980.
2. R. C. James, *Orthogonality and linear functionals in normed linear spaces*, Trans. Amer. Soc., 61, (1947), 265—292.
3. N. Pavel, *Ecuații differențiale asociate unor operatori neliniari pe spații Banach*, Ed. București, 1977.
4. I. Singer, *Cea mai bună aproximare în spații vectoriale normate prin elemente din subspații vectoriale*, Ed. Acad., București, 1967.

In cel de al XXXIII-lea an (1988) *Studia Universitatis Babeş-Bolyai* apare în specialităţile:

matematică

fizică

chimie

geologie-geografie

biologie

filosofie

ştiinţe economice

ştiinţe juridice

istorie

filologie

In the XXXIII-rd year of its publication (1988), *Studia Universitatis Babeş-Bolyai* is issued as follows:

mathematics

physics

chemistry

geology-geography

biology

philosophy

economic sciences

juridical sciences

history

philology

Dans sa XXXIII-e année (1988), *Studia Universitatis Babeş-Bolyai* paraît dans les spécialités:

mathématiques

physique

chimie

géologie-géographie

biologie

philosophie

sciences économiques

sciences juridiques

histoire

philologie