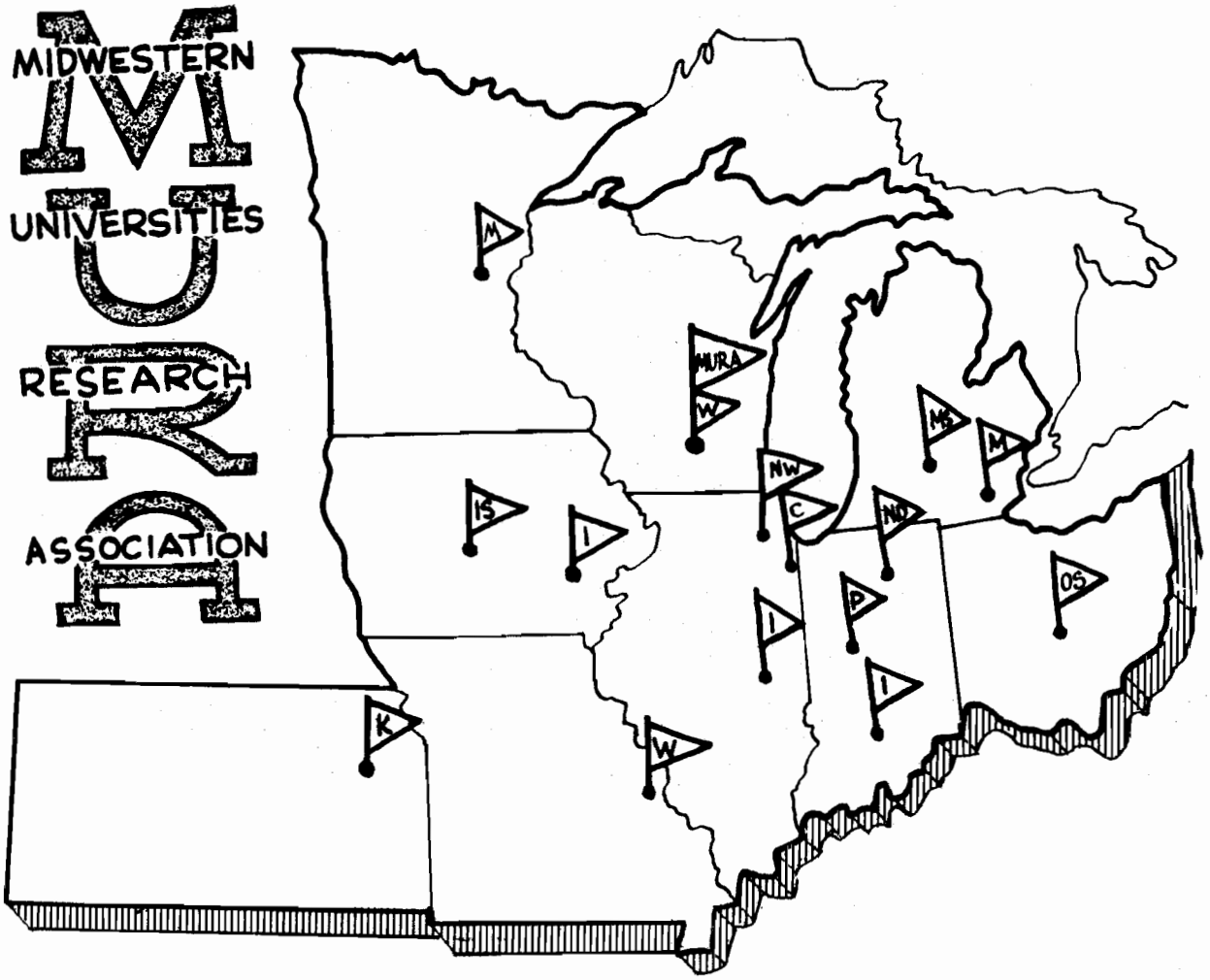
594

Fermilab Library
0 1160 0038873 0

MIDWESTERN UNIVERSITIES RESEARCH ASSOCIATION



CONVERGENCE RATE OF THE NINE-POINT EXTRAPOLATED LIEBMANN ALGORITHM

Harry H. Denman

REPORT **NUMBER** 594

LEGAL NOTICE

This report was prepared as an account of Government sponsored work. Neither the United States, nor the Commission, nor any person acting on behalf of the Commission:

A. Makes any warranty or representation, expressed or implied, with respect to the accuracy, completeness, or usefulness of the information contained in this report, or that the use of any information, apparatus, method, or process disclosed in this report may not infringe privately owned rights; or

B. Assumes any liabilities with respect to the use of, or for damages resulting from the use of any information, apparatus, method, or process disclosed in this report.

As used in the above, "person acting on behalf of the Commission" includes any employee or contractor of the Commission, or employee of such contractor, to the extent that such employee or contractor of the Commission, or employee of such contractor prepares, disseminates, or provides access to, any information pursuant to his employment or contract with the Commission, or his employment with such contractor.

Printed in USA. Price $0.50. Available from the

Office of Technical Services
U. S. Department of Commerce
Washington 25, D. C.

MURA-594
UC-28, Particle Accelerators and
High Voltage Machines
TID-4500 (15th Edition)

MIDWESTERN UNIVERSITIES RESEARCH ASSOCIATION*

2203 University Avenue, Madison, Wisconsin

CONVERGENCE RATE OF THE NINE-POINT EXTRAPOLATED LIEBMANN ALGORITHM

Harry H. Denman**

September 18, 1960

ABSTRACT

A quartic equation is developed which determines the optimum value for the over-relaxation parameter α and the value of the corresponding convergence factor A_b^2 for the numerical solution of Laplace's equation in a rectangle, using the nine-point extrapolated Liebmann algorithm.

*AEC Research and Development Report. Research supported by the U. S. Atomic Energy Commission, Contract No. AT(11-1)-384.

**Department of Physics, Wayne State University, Detroit, Michigan

I. INTRODUCTION

We wish to find an approximate solution of Laplace's equation $\nabla^2\Phi = 0$ in a rectangular region, where the values of Φ are given on the boundary. We set up an array of mesh or equally spaced points in the region, as shown in Fig. 1. There are $(N_1 - 1)(N_2 - 1)$ points in the interior of the region. The points $i = 0$, $i = N_1$, and $j = 0$, $j = N_2$ are on the boundaries. Then Φ_{ij} refers to the value of Φ at the mesh point (i, j). Since we shall be using an iterative method, $\Phi_{ij}^{(n)}$ refers to the value of our approximate solution at the mesh point (i, j) for the nth iteration. The points will be handled from left to right (j fixed, $i = 1, 2, \ldots N_1 - 1$) and from bottom to top during an iteration. The Liebmann algorithms* are characterized by the use of the new values of $\Phi_{ij}^{(n)}$ in each iteration when dealing with the mesh points handled after (i, j) in the nth iteration. In the nine-point method, we consider the effects of the values of Φ at all nearest-neighboring mesh points $(i + 1, j)$, $(i - 1, j)$, $(i, j + 1)$ $(i, j - 1)$, and the next nearest neighbors $(i + 1), j + 1)$, $(i + 1, j - 1)$, $i - 1, j - 1)$, $(i - 1, j + 1)$ on $\Phi_{ij}^{(n)}$. The difference equation involving the values of Φ on this set of points and most closely approximating Laplace's equation is

$$- 20\,\Phi_{ij} + 4\,(\Phi_{i+1,j} + \Phi_{i-1,j} + \Phi_{i-1,j} + \Phi_{i,j-1}) + \Phi_{i+1,j+1} + \\ + \Phi_{i-1,j+1} + \Phi_{i-1,j-1} + \Phi_{i+1,j-1} = 0 .$$

If Φ_{ij} is the exact solution of this difference equation, and if we let the

---

*For background material on iterative methods and the Liebmann algorithms, see references (1) and (2).

error $\phi_{ij}^{(n)}$ in our iterative solution $\Phi_{ij}^{(n)}$ be defined by:

$$\phi_{ij}^{(n)} = \Phi_{ij}^{(n)} - \Phi_{ij} ,$$

then for the extrapolated nine-point Liebmann algorithm, both $\Phi_{ij}^{(n)}$ and $\phi_{ij}^{(n)}$ are given by

$$\phi_{ij}^{(n+1)} = \phi_{ij}^{(n)} + \alpha \Big[- 20\, \phi_{ij}^{(n)} + 4 \big(\phi_{i-1,\,j}^{(n+1)} + \phi_{i+1,\,j}^{(n)} + \phi_{i,\,j-1}^{(n+1)} + \phi_{i,j+1}^{(n)} \big) + \phi_{i+1,\,j+1}^{(n)} + \phi_{i-1,\,j+1}^{(n)} + \phi_{i+1,j-1}^{(n+1)} + \phi_{i-1,j-1}^{(n+1)} \Big] \quad (1)$$

where α is the over-relaxation parameter, $\phi_{ij}^{(o)}$ is arbitrary in the interior, and $\phi^{(n)} = 0$ on the boundaries for all n.

II. SOLUTION OF THE DIFFERENCE EQUATION

As a solution of (1), we try*

$$\phi_{ij}^{(n)} = A^{2n+j} B^{i} \sin i\theta \sin j\phi \quad . \quad (2)$$

Substituting in (1), we find that (2) is a solution of (1) if

$$A^2 = 1 - 20\alpha + 4\alpha \left[B \cos\theta + A^2 B^{-1} \cos\theta + 2 A \cos\phi \right] + 2\alpha A \cos\theta \cos\phi \, (B + B^{-1}) \quad (3)$$

---

*Note that the algorithm (1) is not symmetric with respect to inversion of its application in the x and y directions. If we first apply it in the y direction (i fixed), and then increase x (i), the results are slightly different, since the pattern of "new" and "old" values of ϕ is changed. This is not true in the case of the five-point extrapolated Liebmann algorithm. Thus the solution (2) is <u>not</u> symmetric in i and j, as it was for the five-point algorithm.

and

$$0 = 2B - 2A^2 B^{-1} + A\cos\phi \quad (B - B^{-1}) . \tag{4}$$

Equation (4) can be written:

$$AB^{-1}(2A + \cos\phi) = (2 + A\cos\phi)B, \tag{5}$$

and (3) becomes:

$$A^2 = (1 - 20\alpha) + 4\alpha(2 + A\cos\phi)B\cos\theta + 8\alpha A\cos\phi . \tag{6}$$

Multiplying (5) by B, we obtain

$$A(2A + \cos\phi) = (2 + A\cos\phi)B^2 . \tag{7}$$

Squaring (6) and using (7), we obtain a quartic equation in A, so that (2) is a solution of the difference equation (1) only if A satisfies

$$A^4 - 16\alpha\cos\phi(1 + 2\alpha\cos^2\theta)A^3 + 2\left[20\alpha - 1 + 8\alpha^2(4\cos^2\phi - 4\cos^2\theta - \cos^2\theta\cos^2\phi)\right]A^2 - 16\alpha\cos\phi(20\alpha - 1 + 2\alpha\cos^2\theta)A + (20\alpha - 1)^2 = 0 . \tag{8}$$

From (2) we see that the error will converge to 0 at all points on the mesh as the iterative equation (1) is applied (n increases) only if $|A| < 1$, and that the smaller A is, the more rapid the convergence. Also, in order for the error to be 0 on the boundaries, θ and ϕ must be given by the following expressions:

$$\theta = \frac{\pi}{N_1} r, \qquad r = 1, 2, \ldots, N_1 - 1$$

$$\phi = \frac{\pi}{N_2} s, \qquad s = 1, 2, \ldots, N_2 - 1 .$$

From the results for the five-point extrapolated Liebmann method, we expect the largest value of A, which gives the slowest convergence and therefore dominates the convergence process, to be nearly 1 in magnitude if r

and s are chosen as 1 and the mesh is small (i.e., N_1 and N_2 large). Also, the value of α leading to the smallest real value of A for $r = s = 1$ (let these values be α_b and A_b) is expected to be slightly less than twice the "nominal" value (the value used in the Richardson algorithm). The nominal value in this case is $\alpha = 1/20$, so that α_b is expected to be slightly less than 1/10. With this guide let us consider the case of an infinite mesh, so $\theta = \phi = 0$. Then (8) becomes

$$A^4 - 16\alpha(1 + 2\alpha)A^3 + 2(20\alpha - 1 - 8\alpha^2)A^2 - 16\alpha(22\alpha - 1)A + (20\alpha - 1)^2 = 0. \tag{9}$$

One of the roots of (9) is $A = 1$, for all α. Thus for a very large mesh, one root of (8) approaches 1. If we take $\alpha = 1/10$ in (9), the equation becomes

$$A^4 - 1.92A^3 + (1.84)A^2 - 1.92A + 1 = 0. \tag{10}$$

The roots of (10) are 1, 1, and $0.04 \pm i\sqrt{.9984}$. Thus, for a very fine mesh, one root of (8) approaches 1, and for α near the optimum value, we obtain another root near unity, and a complex pair of roots of modulus 1, with small real and large imaginary parts. These results have been verified numerically, as will be discussed later.

III. APPROXIMATE SOLUTION

It is of fundamental interest to learn how the convergence rate depends on N_1 and N_2 for α chosen in an optimum way. With the information that the real roots of interest are near unity in magnitude, we rewrite (7) as

$$B^2 = 1 - \frac{2(1 - A^2)}{2 + A\cos\phi}. \tag{11}$$

For A near 1, we approximate

$$B \doteq 1 - \frac{1 - A^2}{2 + A \cos\phi} \quad . \tag{12}$$

Substitution in (6) yields the approximate quadratic

$$A^2 (1 - 4\alpha \cos\theta) - 4\alpha \cos\phi \, (\cos\theta + 2) A + 20\alpha - 1 - 4\alpha \cos\theta = 0, \tag{13}$$

or

$$A^2 (1/4\alpha - \cos\theta) - \cos\phi \, (\cos\theta + 2) A + 5 - 1/4\alpha - \cos\theta = 0.$$

Note that for $\cos\theta = \cos\phi = 1$ and $\alpha = 0.1$, the roots of (13) are 1, 1, which are the real roots of (8) for this limiting case. In the five-point extrapolated Liebmann procedure, the optimum value of α (for the smallest convergence factor) is found when the roots of a quadratic equation are equal (see Reference 1). If we <u>assume</u> here that the optimum value of α is that for which the roots of (13) are equal, we obtain the quadratic equation in α_b

$$X \alpha_b^2 - 5 \alpha_b + \frac{1}{4} = 0 \tag{14}$$

where

$$X = \cos^2\phi \, (\cos^2\theta + 4\cos\theta + 4) - 4\cos^2\theta + 20\cos\theta \, . \tag{15}$$

The solutions of (14) are

$$\alpha_b = \frac{5 \pm \sqrt{25 - X}}{2X} = \frac{1}{10} \, \frac{1 \pm \sqrt{1 - X/25}}{X/25} \quad . \tag{16}$$

The + value is discarded since it leads to values of α greater than 0.1, which produces all complex roots A in (8) and (13). Using $\alpha_b = \dfrac{0.1}{1 + \sqrt{1 - X/25}}$, from (16)

$$A_b^2 \doteq \frac{20\alpha_b - 1 - 4\alpha_b \cos\theta}{1 - 4\alpha_b \cos\theta} = 1 - \frac{2(1 - 10\alpha_b)}{1 - 4\alpha_b \cos\theta}$$

or

$$A_b^2 \doteq 1 - 2 \frac{\sqrt{25 - X}}{5 - 2\cos\theta + \sqrt{25 - X}} \quad . \tag{17}$$

Substituting for X from (15),

$$A_b^2 \doteq \frac{(5 - 2\cos\theta) - \sqrt{(5 - 2\cos\theta)^2 - \cos^2\phi \quad (\cos\theta + 2)^2}}{(5 - 2\cos\theta) + \sqrt{(5 - 2\cos\theta)^2 - \cos^2 \quad (\cos\theta + 2)^2}} \quad . \tag{18}$$

For $\theta = \phi$, $\cos\theta = \cos\phi = 1 - a$, $a = \frac{1}{2}\left(\frac{\pi}{N}\right)^2$, where $N_1 = N_2 = N$, $X \doteq 25 - 36\,a$, to first order in a. Then

$$A_b^2 \doteq 1 - \frac{12\sqrt{a}}{3 + 2a + 6\sqrt{a}} \doteq 1 - 4\sqrt{a} \; . \tag{18'}$$

For comparison, the five-point extrapolated Liebmann algorithm yields an optimum value $A_b^2 = 1 - 2\sqrt{2}\sqrt{a}$. The important result, if our approximation is good, is that the convergence rate $(1 - A^2)$ is proportional to $\sqrt{a} = \frac{1}{\sqrt{2}}\frac{\pi}{N}$, so that it is of order $\frac{1}{N}$.

IV. COMPUTER SOLUTIONS

Because of the uncertainties in the approximation procedure above, and since an exact solution of (8) and optimization with respect to α appeared difficult analytically, it was decided to solve (8) on the computer for various values of α, N_1 and N_2, and vary α until the minimum maximum real root A_b was obtained. A floating-point FORTRAN program was written by T. Edwards and V. Pratt which varied α by fixed increments, and found the roots of (8). For reasonable running times, the increments in α were of order 0.00125, but this was too large to give accurate results for A_b. Therefore a search routine was added which reduced the change in α as the optimum point (α_b, A_b) was approached. Still, the smallest change in α possible with floating-point numbers ($\sim 10^{-9}$) was large enough to produce appreciable

changes in A_b. Therefore, a fixed point routine which also searched for (α_b, A_b) was written, so that the minimum $\Delta\alpha$ is now about 3×10^{-11}. With the results of the earlier program (floating point) as first approximations, the fixed-point program was used to gain more precision. By this technique, about five-place accuracy in A_b was obtained.

In all cases tried, the best value A_b was obtained where two real roots of (8) became the same, within the accuracy of the program. This seems to be a characteristic of this problem, and use was made of this fact in the fixed-point program. That is, since a double root x for a polynomial equation $P_4(\alpha, x) = 0$ gives

$$\frac{dP_4}{dx} = 0 \tag{19}$$

at this point also, the computer program first sought the root of the cubic equation (19), and then sought the roots of P_4 nearby. It was always found, in the cases tried, that these three roots were very close together, so that apparently the optimum value of α does yield two equal roots A_b for (8).

The computer results for α_b and A_b^2 for $N_1 = 10\ (10)\ 100$ and $N_2 = 10\ (10)\ 100$ are given in Table I. The value of α_b is first in each entry, with the corresponding value of A_b^2 just below.

The value of K^*, which is the equivalent to A_b^2 for the five-point extrapolated Liebmann algorithm, was calculated also in each case, and in general gave a value for the convergence rate ($1 - A_b^2$ or $1 - K^*$) of 10 to 16% <u>less</u> than for the nine-point case. Thus, although the nine-point algorithm (1) may be expected to take longer to calculate since more points

must be handled, it converges faster, and to a more accurate solution of Laplace's equation.

In addition to the cases for Table I, calculations were also carried out for the cases $N_1 = N_2 = 200$, 400, 1000, in an attempt to test the accuracy of our approximation formulae (16), (17), and (18). The results for these cases, plus the 10 x 10 case, are summarized in Table II. These figures show that the results of the approximation formula do approach those of the numerical solutions of (8) for N_1 and N_2 large. The relative error is appreciable for small N, but the indication that the convergence rate of the same order and slightly improved for the nine-point results as compared to the five-point results is verified. These "handy" formulae (16) to (18) can be used to obtain first approximations to α_b and A_b for the computer program, and thus save computer time. Alternately, values from Table I and Table II could be used, with an interpolation or extrapolation formula, to obtain first estimates for different N_1 and/or N_2.

The complex pair of roots of (8), which accompanies the real equal pair in the computer solutions of (8), does approach, as a limiting value, the results given under equation (10) as N_1 and N_2 become very large (and, according to the computer results, α approaches 0.1). For example, using the second floating-point program (which searches for α_b automatically), for $N_1 = N_2 = 100$, the complex pair is $-0.0389 \pm i\,0.972$, and for $N_1 = N_2 = 400$, the complex pair is $-0.03973 \pm i\,0.9924$. This agreement is a good check on the computer program.

V. CONCLUSIONS AND DISCUSSION

A solution is obtained here for the functions satisfying the difference equation (1) for the nine-point extrapolated Liebmann algorithm. An auxilliary condition on the convergence parameter A appearing in the solution is given by the quartic equation (8). The value of the overrelaxation parameter α such that the real roots of (8) have their minimum maximum value is to be found. Such values were obtained numerically, using the 704, for a large number of cases, as summarized in Tables I and II. Approximate expressions for α_b and A_b^2 were also obtained, which are more accurate as the number of points in the mesh is increased.

Several minor questions remain unanswered at present. Are all four roots of (8) actually solutions of (1)? The complex pair obtained (for $\alpha < 0.1$) seem to be extraneous, but there is no proof of this as yet. Also, does the best (minimax) value A_b <u>always</u> occur where there is a real <u>pair</u> of roots of (8) (plus a complex pair)?

The results indicate that this algorithm will always converge faster (fewer iterations) than the corresponding five-point formula, and it converges to a more accurate solution of Laplace's equation. However, the total time required will probably be greater, due to the greater number of points involved in each calculation. A combined use, first of the five-point formula, followed by the nine-point, might be worthwhile in a number of cases.

VI. ACKNOWLEDGEMENTS

The author would like to thank Dr. L. Jackson Laslett, who suggested the problem, and Mr. T. Edwards and Mr. Val Pratt, who carried out the programming and coding for the computer.

REFERENCES

(1) Convergence Rates of Iterative Treatments of Partial Differential Equations. S. P. Frankel, MTAC 4, p. 65 (1950).

(2) MURA-327, Convergence Rates of Iterative Treatments of Laplace's Equation, M. J. Freiser and L. J. Laslett.

TABLE I

N_1		10	20	30	40	50
10	α_b	.07607 20059	.07998 22256	.08089 00316	.08122 70330	.08138 67465
	A_b^2	.47304	.55473	.57399	.58117	.58458
20		.07996 76244	.08624 73806	.08805 35485	.08878 91726	.08915 38516
		.55503	.69018	.72988	.74614	.75422
30		.08087 32498	.08805 17207	.09036 88850	.09138 76609	.09191 70000
		.57434	.72992	.78131	.80407	.81593
40		.08120 95461	.08878 68340	.09138 71794	.09259 23202	.09324 35774
		.58153	.74619	.80408	.83112	.84579
50		.08136 89448	.08915 13054	.09191 63347	.09324 33987	.09398 23977
		.58495	.75428	.81595	.84580	.86248
60		.08145 65657	.08935 66439	.09222 33218	.09363 18336	.09443 38428
		.58683	.75884	.82284	.85456	.87270
70		.08150 97576	.08948 31507	.09241 60866	.09388 06322	.09472 83865
		.58797	.76165	.82718	.86019	.87938
80		.08154 44278	.08956 63929	.09254 45709	.09404 88507	.09493 03706
		.58871	.76350	.83007	.86399	.88396
90		.08156 82646	.08962 39946	.09263 42874	.09416 75507	.09507 44550
		.58922	.76478	.83209	.86668	.88723
100		.08158 53487	.08966 54673	.09269 93081	.09425 42554	.09518 05946
		.58959	.76571	.83355	.86864	.88964

N_1/N_2		60	70	80	90	100
10	α_b	.08147 45354	.08152 78274	.08156 25621	.08158 64430	.08160 35585
	A_b^2	.58645	.58759	.58833	.58884	.58921
20		.08935 92943	.08948 58610	.08956 91407	.08962 67676	.08966 82579
		.75878	.76159	.76344	.76472	.76565
30		.09222 40747	.09241 68877	.09254 54011	.09263 51367	.09270 01706
		.82283	.82716	.83005	.83207	.83353
40		.09363 20948	.09388 09373	.09404 91819	.09416 78983	.09425 46143
		.85456	.86018	.86399	.86667	.86863
50		.09443 39239	.09472 85100	.09493 05188	.09507 46187	.09518 07685
		.87270	.87938	.88396	.88723	.88964
60		.09493 34467	.09526 46822	.09549 48075	.09566 06992	.09578 39412
		.88403	.89155	.89678	.90055	.90336
70		.09526 46401	.09562 50112	.09587 82870	.09606 26487	.09620 07256
		.89155	.89974	.90551	.90971	.91285
80		.09549 47414	.09587 82631	.09615 05090	.09635 04109	.09650 12531
		.89678	.90551	.91171	.91627	.91971
90		.09566 06182	.09606 26101	.09635 03963	.09656 33317	.09672 51033
		.90055	.90971	.91627	.92112	.92482
100		.09578 38504	.09620 06775	.09650 12291	.09672 50940	.09689 62024
		.90336	.91285	.91971	.92482	.92872

TABLE II ($N_1 = N_2$)

N_1 (= N_2)	α_b Eq. (16)	α_b Computer	A_b^2 Eq. (18)	A_b^2 Eq. (18')	A_b^2 Computer	Five-point algorithm K^* (corresonds to A_b^2)
10	.0792	.0761	.4051	.1151	.4730	.5279
100	.0974	.0969	.9150	.9111	.9287	.9391
200	.0987	.0984	.9565	.9556	.9637	.9691
400	.0993	.0992	.9780	.9778	.9817	.9844
1000	.09973	.09968	.9912	.9911	.9926	.9937

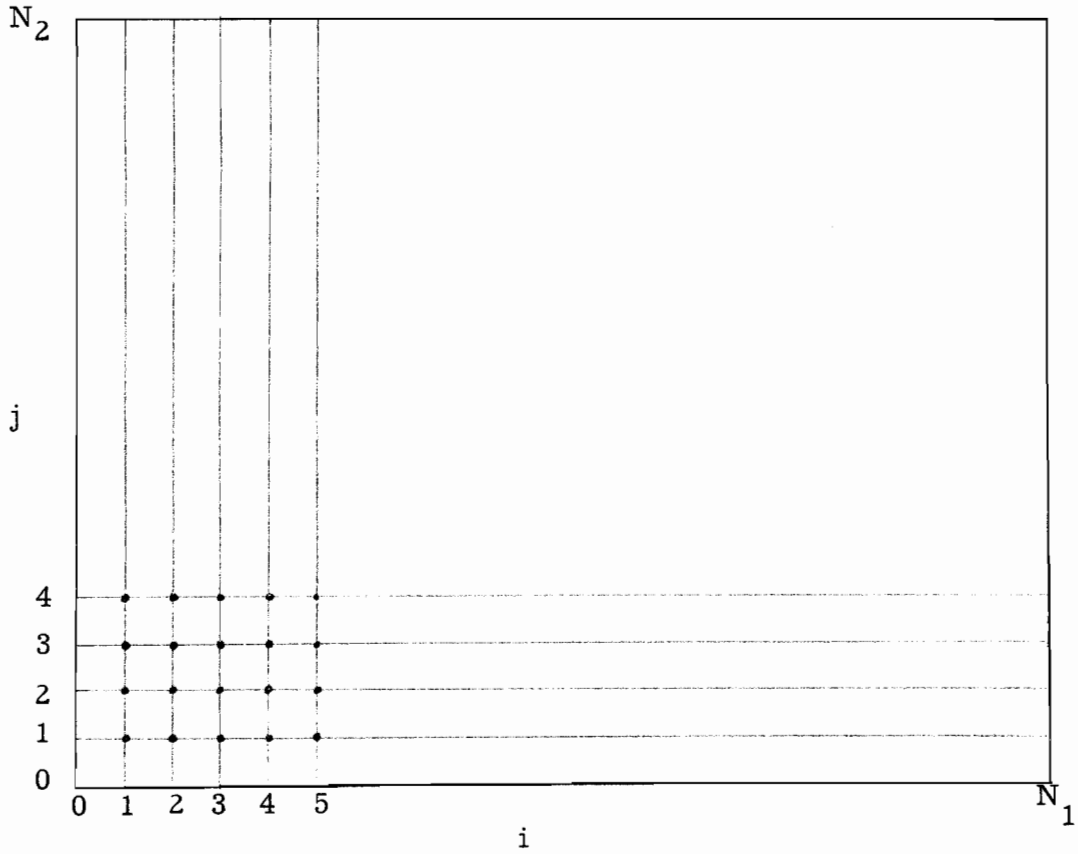


Fig. 1