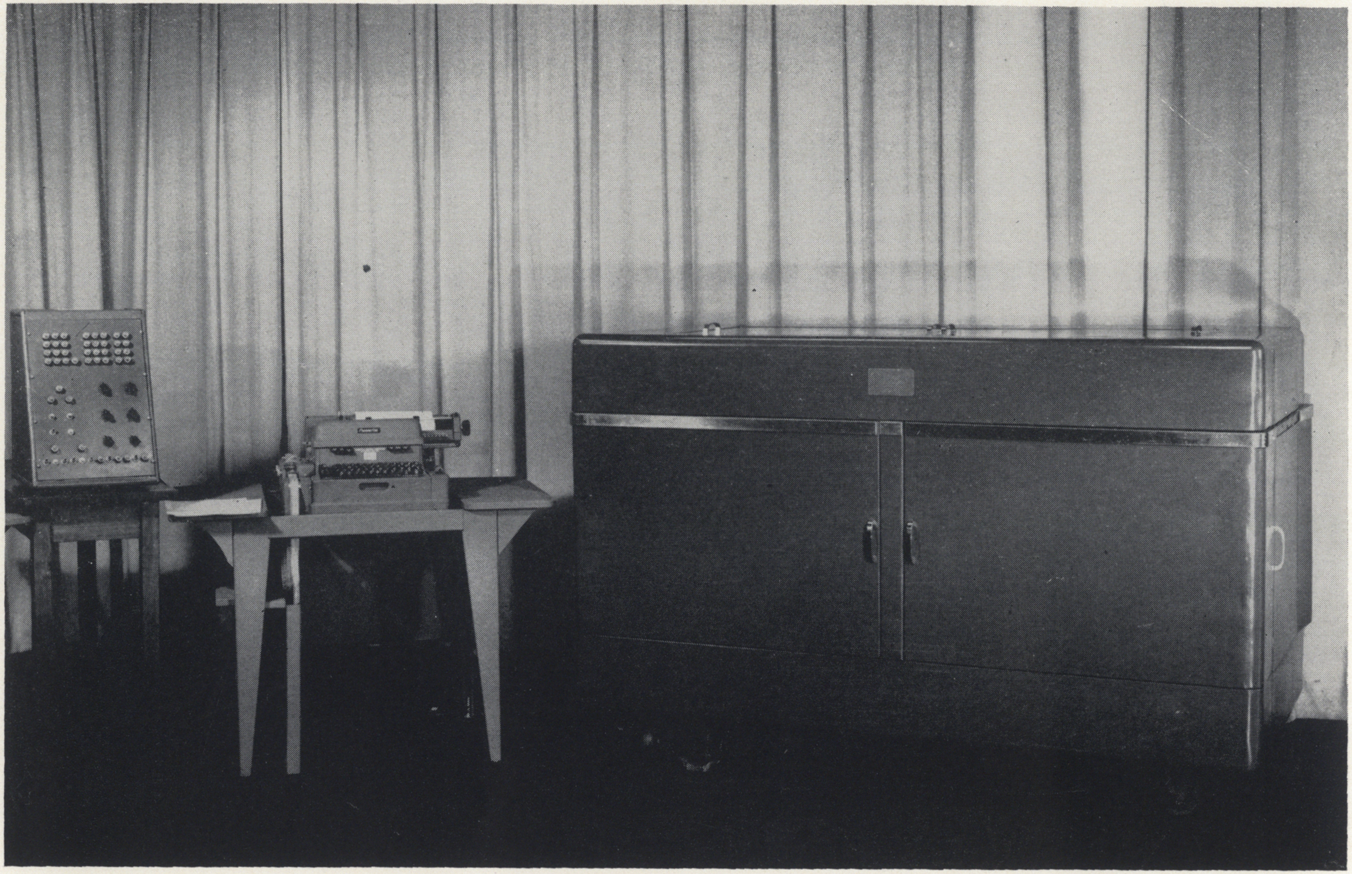

THE CIRCLE COMPUTER

Computing Eigenvalues and Eigenvectors of a Symmetric Matrix on the ILLIAC

One of the programs in the library of programs for the University of Illinois' electronic digital computer, known as the ILLIAC, is a program for finding the eigenvalues and eigenvectors of a symmetric matrix. The iterative method used is the rotation of axes method discussed by H. H. GOLDSTINE[1] in an unpublished paper and referred to by TAUSSKY & TODD[2] as JACOBI'S method.[3] It consists essentially of performing a sequence of orthogonal transformations on the matrix, where each transformation is designed to reduce a selected off-diagonal element to zero. GOLDSTINE[1] shows that the sum of the squares of the off-diagonal elements is reduced, during a single transformation, by the amount $2a^2_{jk}$ (a_{jk} is the element reduced to zero by the transformation) and that the process produces a sequence of matrices whose limit is a diagonal matrix. His expression for an upper bound on the number of transformations required to diagonalize an nth order matrix is $[\ln (t_0/t_i)](n^2 - n)/2$, where t_0 is the sum of the squares of the off-diagonal elements of the original matrix and t_i is the same quantity after the ith transformation. Here it is assumed that a_{jk} is always greater than the average off-diagonal element in absolute value. Our results so far indicate that this bound is from ten to twenty times greater than the number actually required by our program. The eigenvectors are obtained by multiplying together the orthogonal matrices used in the successive transformations.

When the eigenvalues are not close, and the element a_{jk} is small, reduction of a_{jk} to 0 leaves the other elements unchanged in the first approximation. For the angle of rotation is given by the relation

$$\tan 2\phi = 2a_{jk}/(a_{jj} - a_{kk}).$$

When the off-diagonal elements are very small ϕ is of the order of a_{jk} ($a_{jj} \neq a_{kk}$). Now off-diagonal elements are transformed by

$$\begin{aligned} a_{rj} &= a_{rj}\cos\phi + a_{rk}\sin\phi = a_{rj}(1 - \phi^2/2! + \cdots) + a_{rk}(\phi - \phi^3/3! + \cdots) \\ a_{rk} &= -a_{rj}\sin\phi + a_{rk}\cos\phi = a_{rj}(-\phi + \phi^3/3! - \cdots) \\ &\qquad\qquad + a_{rk}(1 - \phi^2/2! + \cdots) \end{aligned}$$

and hence are unchanged if second order terms are neglected. Thus one sweep through the off-diagonal elements reduces them to zero (up to terms of second order).

The purpose of this paper is to display some results of an investigation into the relative merits of

(1) two approaches to the problem of how to select the off-diagonal element a_{jk} mentioned above, and
(2) two approaches to the problem of when to apply the convergence test so as to terminate the process after convergence.

The two approaches mentioned in (1) are
- (a) to select the off-diagonal elements in sequence along successive rows of the matrix, and
- (b) to select the largest off-diagonal element each time.

The two approaches mentioned in (2) are
- (c) to apply the convergence test after each transformation, and

(d) to apply the convergence test after each group of $(n^2 - n)/2$ transformations, where n is the order of the matrix. The convergence test used was a test of the size of t_i using double precision.

Method (a) is the simplest to program for an electronic computer but will require more transformations for convergence than method (b). Thus the accumulated round-off errors should be less using method (b). Method (c) enables one to terminate the process as soon as the process converges, but requires many applications of the test. Method (d) applies the test only after going through the off-diagonal elements once. This means that over-iterating will result and in the extreme case $(n^2 - n - 2)/2$ unnecessary transformations will be performed.

The library program mentioned in the first paragraph is called program 42. It uses methods (a) and (d) and requires a total of 190 storage locations in the memory. Two modifications of this program have been written which are slightly longer. Program 42A uses methods (a) and (c) and program 42B uses methods (b) and (c). Tables 1–5 display the results obtained using these three programs on seven matrices of each of the orders 20, 16, 12, 8, and 4. They contain

A. The time required to diagonalize each matrix,
B. The number of orthogonal transformations required, and
C. An indication of the accuracy.

Tables 6–9 contain the eigenvalues of the thirty-five matrices. In order to conserve space only five decimal places are included.

The time referred to in A is merely the computation time and does not include the time required for input or output of data. The accuracy of the process (item C above) is determined by forming the sum of the squares of the components of the n residual vectors,

$$r_i = Ax_i - \pi_i x_i \qquad i = 1, 2, \cdots, n$$

where x_i and π_i are the eigenvectors and eigenvalues, respectively, of A. This sum of squares is small and is scaled by 2^{30} before being printed.

Of the thirty-five matrices used in this investigation five were correlation matrices which were available (numbers 1, 8, 15, 22, and 29) and the remaining thirty were matrices generated by the machine. The method employed to generate the elements of these matrices was to square a number and use the middle digits of the product. Each new number then was used to generate the following number. The computation times for diagonalizing the five correlation matrices are slightly longer than those for the machine generated matrices due to the fact that certain changes were made in the ILLIAC, just after the five correlation matrices were diagonalized, which increased the speed of certain arithmetic operations.

Several conclusions can be drawn from an inspection of the results. The simplest program (number 42) using methods (a) and (d) was the fastest despite the fact that it over-iterated. However program 42B, using method (b), was in general the most accurate in the sense that the sum of squares of residuals was smallest. Obviously, the method (d) is superior to method (c). Program 42 never required more than seven sweeps through the off-diagonal elements, i.e., no more than 7 $(n^2 - n)/2$ transformations were required for convergence. It appears that method (a) required about one and one-half times as many transformations as method (b).

Matrix	Time Min.	Sec.	Number of Transformations	$n = 20$ $2^{30}\Sigma r_{ij}^2$
		PROGRAM 42		
1	6	39	1330	.00 345
2	5	57	1330	.00 331
3	5	56	1330	.00 286
4	5	56	1330	.00 310
5	5	6	1140	.00 251
6	5	56	1330	.00 305
7	5	6	1140	.00 296
		PROGRAM 42A		
1	15	18	1154	.00 272
2	14	25	1161	.00 279
3	14	27	1164	.00 282
4	14	17	1149	.00 326
5	13	32	1087	.00 230
6	14	23	1159	.00 314
7	13	56	1122	.00 263
		PROGRAM 42B		
1	19	34	683	.00 145
2	19	19	685	.00 138
3	19	32	692	.00 131
4	19	32	692	.00 167
5	19	18	684	.00 148
6	19	17	684	.00 141
7	19	13	681	.00 165

TABLE 1

Matrix	Time Min.	Sec.	Number of Transformations	$n = 16$ $2^{30}\Sigma r_{ij}^2$
		PROGRAM 42		
8	3	28	840	.00 191
9	3	6	840	.00 163
10	3	6	840	.00 149
11	3	5	840	.00 142
12	2	39	720	.00 131
13	2	39	720	.00 104
14	2	39	720	.00 152
		PROGRAM 42A		
8	6	54	726	.00 163
9	6	23	724	.00 126
10	6	23	725	.00 106
11	6	21	721	.00 133
12	5	53	667	.00 126
13	5	50	663	.00 106
14	5	37	637	.00 154
		PROGRAM 42B		
8	8	10	425	.00 063
9	8	0	432	.00 066
10	7	53	426	.00 061
11	8	4	436	.00 070
12	7	47	421	.00 063
13	7	54	427	.00 072
14	7	57	430	.00 059

TABLE 2

Matrix	Time Min.	Time Sec.	Number of Transformations	$\frac{n = 12}{2^{30}\sum r_{ij}^2}$
		PROGRAM 42		
15	1	17	396	.000 600
16	1	9	396	.000 591
17	1	9	396	.000 447
18	1	9	396	.000 450
19	1	9	396	.000 502
20	1	9	396	.000 579
21	1	9	396	.000 600
		PROGRAM 42A		
15	2	10	344	.000 681
16	1	59	340	.000 492
17	1	58	338	.000 506
18	1	58	339	.000 369
19	2	2	351	.000 407
20	2	2	351	.000 501
21	2	1	347	.000 631
		PROGRAM 42B		
15	2	33	217	.000 233
16	2	41	239	.000 307
17	2	33	228	.000 277
18	2	35	230	.000 233
19	2	25	216	.000 256
20	2	32	227	.000 257
21	2	39	236	.000 232

TABLE 3

Matrix	Time Min.	Time Sec.	Number of Transformations	$\frac{n = 8}{2^{30}\sum r_{ij}^2}$
		PROGRAM 42		
22		20	140	.000 125
23		18	140	.000 134
24		18	140	.000 119
25		18	140	.000 118
26		18	140	.000 111
27		22	168	.000 165
28		18	140	.000 102
		PROGRAM 42A		
22		28	126	.000 160
23		28	135	.000 113
24		26	129	.000 112
25		25	121	.000 106
26		28	134	.000 088
27		29	143	.000 208
28		28	135	.000 102
		PROGRAM 42B		
22		35	97	.000 072
23		32	91	.000 047
24		30	88	.000 085
25		32	94	.000 063
26		32	93	.000 056
27		34	96	.000 043
28		34	97	.000 081

TABLE 4

Matrix	Time Min.	Time Sec.	Number of Transformations	$n = 4$ $2^{30}\sum r_{ij}^2$
		PROGRAM 42		
29		2.5	24	.0000 212
30		2.4	24	.0000 095
31		3.0	30	.0000 070
32		2.4	24	.0000 075
33		2.3	24	.0000 137
34		2.4	24	.0000 110
35		2.4	24	.0000 086
		PROGRAM 42A		
29		2.4	19	.0000 138
30		2.3	20	.0000 056
31		2.9	25	.0000 113
32		2.1	19	.0000 029
33		2.3	20	.0000 093
34		2.4	22	.0000 078
35		2.4	22	.0000 143
		PROGRAM 42B		
29		2.4	17	.0000 055
30		2.3	16	.0000 051
31		2.4	16	.0000 053
32		2.4	16	.0000 030
33		2.4	17	.0000 042
34		2.4	16	.0000 067
35		2.4	19	.0000 090

TABLE 5

EIGENVALUES

1	2	3	4	5	6	7
+.72951	+.70548	+.61214	−.60855	+.34710	+.65011	+.55872
−.75242	−.54004	−.38161	−.52812	+.56507	+.36704	−.71648
−.54106	−.50730	+.45455	−.43533	−.74842	−.73229	+.60475
+.41564	−.23097	−.72650	+.78575	+.54166	+.68674	−.33592
−.58256	−.64287	−.53514	+.46933	−.61324	+.57402	−.53408
−.28102	+.58653	−.43796	−.64543	−.27147	−.59379	+.69971
+.52031	+.26919	+.75447	+.27829	−.49548	−.23771	+.36562
+.58046	−.10555	+.55998	+.71874	−.52582	+.42814	−.37732
−.41469	+.29675	−.30252	−.32482	+.52150	−.49613	−.60475
−.29046	+.52572	+.12832	−.22613	−.45968	+.24109	+.29305
−.33468	−.37614	−.27717	+.59033	−.34006	+.34885	−.20574
+.54923	−.43715	−.07906	+.35744	−.10921	−.13801	+.27637
+.02789	+.48254	+.30587	+.07084	+.29205	−.46256	−.07809
+.18720	+.35102	+.48978	+.21038	−.08419	+.27996	−.00028
−.18534	−.16190	+.21471	−.11323	+.03378	+.05406	−.12379
+.13149	−.24689	−.23067	+.09255	+.18037	+.20607	+.41737
+.29696	+.06814	−.21203	+.02895	−.19826	−.41526	+.20575
+.05071	+.03994	+.23868	−.27643	+.00225	−.28314	−.28980
−.13306	+.17470	−.03577	−.06418	+.24630	−.05462	+.06703
−.05810	+.21225	+.02712	−.16923	+.07460	−.18670	+.13132

TABLE 6

EIGENVALUES

8	9	10	11	12	13	14
+.62833	+.64790	+.65658	−.54089	+.26411	+.42710	+.60556
−.55535	−.50376	−.45344	−.38215	+.49948	−.47585	+.13948
−.64081	−.43511	+.46503	−.51589	−.49670	+.57891	−.68004
+.39019	−.05532	−.61847	+.59733	+.48355	−.24169	+.55796
−.50721	−.57457	−.28238	+.47695	−.60610	−.59440	+.51185
−.33063	+.47444	−.36903	−.47087	−.39261	+.62438	−.54204
+.32001	+.17786	+.53729	+.07674	−.52051	+.15495	−.24080
+.51316	+.02452	+.35702	+.66319	−.33774	−.19197	+.31760
−.29364	+.20854	−.24871	−.14883	+.40512	−.32898	−.43488
−.13233	+.51314	+.16411	−.23124	−.15733	+.27840	−.03264
−.19180	−.32012	−.21132	+.36210	−.26548	−.31017	+.33752
+.24308	−.24645	−.01296	+.06539	−.12244	+.28416	−.17560
−.02683	+.33257	+.12396	+.19461	+.19613	+.00504	−.32918
+.17290	+.31621	+.26874	−.07243	+.01189	+.06621	+.26982
+.02344	−.12027	+.19060	−.08335	+.05150	−.07676	+.05161
+.10861	−.18881	−.10576	+.10039	+.13602	+.18048	−.09026

TABLE 7

EIGENVALUES

15	16	17	18	19	20	21
+.53175	+.46872	+.54439	−.44956	+.26122	+.38378	+.57151
−.54732	−.53186	−.38947	−.32509	+.44123	−.46303	+.05192
−.44538	−.33823	+.37032	−.30085	−.42957	+.51761	−.51638
+.33602	+.14008	−.50660	+.57480	+.24992	−.27931	+.45467
−.14756	−.40958	−.22885	+.29884	−.58716	−.48722	+.39733
−.16680	+.40960	−.33077	−.28068	−.32414	+.30840	−.30701
+.20161	+.17662	+.29323	+.08503	−.21857	+.09502	−.23531
+.35719	−.01891	+.18885	+.44081	−.30374	+.01101	+.12498
−.30020	+.27312	−.13267	−.11625	+.16273	−.11119	−.34457
+.04760	+.32131	+.21655	−.04640	−.02549	+.19796	−.10689
−.06842	−.20311	−.02120	+.21271	−.26387	−.20737	+.23361
+.12093	−.24408	+.05281	+.04666	+.03641	+.23940	−.09563

TABLE 8

EIGENVALUES

22	23	24	25	26	27	28
+.39745	+.39005	+.42526	−.30457	+.17009	+.21844	+.50218
−.37863	−.41997	−.34751	−.28376	+.34870	−.29635	+.08990
−.35548	−.14337	+.32709	−.12749	−.30897	+.28240	−.25758
+.22612	+.16712	−.27489	+.21208	+.13591	−.07422	+.32270
−.04768	−.25903	−.19429	+.29716	−.45210	−.32743	+.02900
+.02841	+.27762	−.03599	−.03213	−.26639	+.46710	−.39965
+.12765	+.06557	+.17299	+.07477	−.16227	+.03887	−.17113
+.06131	−.02717	+.09796	+.37703	−.09466	−.01919	−.05480

29	30	31	32	33	34	35
+.12263	+.18964	+.26306	−.28179	+.19975	+.18424	+.16572
−.32853	−.21833	−.26037	−.12093	−.01266	−.14856	−.07177
−.03726	−.02269	+.14103	+.08960	−.28262	+.28318	−.22650
+.19951	+.05038	−.14230	+.16096	−.21660	+.03009	+.12923

TABLE 9

[1] Institute for Advanced Study, Princeton, 1949.
[2] O. Taussky & J. Todd, "Systems of equations, matrices and determinants," *Mathematics Magazine*, v. 26, 1952, p. 71–88.
[3] C. G. J. Jacobi, "Ein leichtes Verfahren, die in der Theorie der Säkularstörungen vorkommenden Gleichungen numerisch aufzulösen," *Jn. reine angew. Math.*, v. 30, 1846, p. 51–95.

University of Illinois
Urbana, Ill.

Robert T. Gregory