1976-04

A numerical comparison of Toeplitz equation solving algorithms

Bell, Edison L.

NPS-53Fe76041

NAVAL POSTGRADUATE SCHOOL

Monterey, California

A NUMERICAL COMPARISON OF TOEPLITZ EQUATION SOLVING ALGORITHMS

Edison L. Bell

and

Richard Franke

April 1976

NAVAL POSTGRADUATE SCHOOL
Monterey, California

Rear Admiral Isham Linder
Superintendent

Jack R. Borsting
Provost

ABSTRACT

This report presents the results of a test of the numerical accuracy of some Toeplitz equation-solving algorithms. A typical autocorrelation function of signal plus noise was used to form the Toeplitz coefficient matrix. Thirty separate data sets of systems of order 4 through 128 were formed, and the resulting equations were solved by each of four different algorithms. IMSL'S LEQT1F Gauss elimination procedure, run in double precision, was used as the standard for comparison of accuracies. The results show that the Levinson algorithm is to be recommended for small (order $\leq$ 16) systems to which it is applicable. Otherwise, the algorithm of choice is the Bareiss algorithm.

This report was prepared by:

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: NPS-53Fe76041
2. GOVT ACCESSION NO.:
3. RECIPIENT'S CATALOG NUMBER:

4. TITLE (and Subtitle): A NUMERICAL COMPARISON OF TOEPLITZ EQUATION SOLVING ALGORITHMS

5. TYPE OF REPORT & PERIOD COVERED: Final July 75 - April 76

6. PERFORMING ORG. REPORT NUMBER:

7. AUTHOR(s): Edison L. Bell and Richard Franke

8. CONTRACT OR GRANT NUMBER(s):

9. PERFORMING ORGANIZATION NAME AND ADDRESS:
Naval Postgraduate School
Monterey, CA 93940

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS:

11. CONTROLLING OFFICE NAME AND ADDRESS:

12. REPORT DATE: April 1976

13. NUMBER OF PAGES:

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office):

15. SECURITY CLASS. (of this report): UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Toeplitz equations
linear prediction
autocorrelation matrix
numerical comparisons

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report presents the results of a test of the numerical accuracy of some Toeplitz equation-solving algorithms. A typical autocorrelation function of signal plus noise was used to form the Toeplitz coefficient matrix. Thirty separate data sets of systems of order 4 through 128 were formed, and the resulting equations were solved by each of four different algorithms. IMSL's LEQT1F Gauss elimination procedure, run in double precision, was used as the standard for comparison of accuracies. The results show that the

Levinson algorithm is to be recommended for small (order $\leq$ 16) systems to which it is applicable. Otherwise, the algorithm of choice is the Bareiss algorithm.

Table of Contents

1.0 Introduction

This report outlines the results of a numerical comparison of several methods for solving Toeplitz equations. A moderately thorough review of recent literature in the fields of electrical engineering, mathematics, and computing revealed the existence of three different algorithms tailored specifically for solution of these equations. One is applicable only in special (but important) cases; we include it in our comparison.

The Toeplitz matrix arises in a variety of problems in electrical engineering. The most familiar of these are probably (1) problems in electromagnetic theory involving integral equations with difference kernels such as in the numerical solution of Hallen's equation for a thin cylindrical antenna [1], and (2) problems in discrete linear filtering and prediction theory where the Toeplitz matrix arises from the calculation of the autocorrelation matrix [4], [5], [8], [9], [10].

The elements of a Toeplitz matrix depend only on the difference between column and row number, j - i, rather than on i and j independently. A general Toeplitz matrix is of the form

$$T_{n+1} = \begin{pmatrix} \rho_0 & \rho_1 & \rho_2 & - & - & \rho_n \\ \rho_{-1} & \rho_0 & \rho_1 & - & - & \rho_{n-1} \\ \rho_{-2} & \rho_{-1} & \rho_0 & - & - & \rho_{n-2} \\ \cdot & & & & & \cdot \\ \cdot & & & & & \cdot \\ \rho_{-n} & \rho_{-n+1} & \rho_{-n+2} & - & - & \rho_0 \end{pmatrix}$$

and the system of equations to be solved is

$$T_{n+1} X = C$$

where X is an nx1 vector of unknowns and C is the nx1 vector of constants.

It is easily observed that T_{n+1} is persymmetric, that is, symmetric about the cross diagonal which extends from lower left to upper right. Further, the elements on a given codiagonal are all the same. In applications the Toeplitz matrix is frequently symmetric in which case $\rho_{-i} = \rho_i$. We will only consider real symmetric Toeplitz matrices here.

A very popular algorithm for solution of equation of this type is that of Trench [1], [2], [7], which actually computes the inverse of the coefficient matrix. Bareiss [3] gave an elimination method which takes advantage of the nature of the Toeplitz matrix. A very simple recursive method known as Levinson's algorithm [4], [5] is applicable in the case of linear prediction. In addition to the above, we also included a standard Gauss elimination procedure, as implemented by IMSL* in subroutine LEQT1F.

2.0 Features of Tested Algorithms.

We will not describe the algorithms in detail since the references are readily available. Instead we will only address ourselves to a brief discussion of timing and storage requirements.

We assume that the matrix is specified by its first row or column and that the constant vector is also given. Storage requirements beyond these two vectors will be given.

*International Mathematical and Statistical Libraries,
7500 Bellaire Blvd.
Houson, TX 77036

2.1 Levinson's Algorithm [4] , [5]

This algorithm is extremely simple and fast, but is applicable only to a special system of Toeplitz equations of the form:

$$(1)\qquad \begin{pmatrix} \rho_0 & \rho_1 & - & - & \rho \\ \rho_1 & \rho_0 & - & - & \rho_{n-1} \\ . & . & & & . \\ . & . & & & . \\ \rho_n & \rho_{n-1} & - & - & \rho_0 \end{pmatrix} X = \begin{pmatrix} \rho_1 \\ \rho_2 \\ . \\ . \\ \rho_{n+1} \end{pmatrix}$$

The algorithm requires $0(n^2)$ multiplications and divisions and only $n+1$ auxiliary storage locations. Thus it is quite attractive for use in the special instance where it is applicable.

2.2 Trench Inversion Algorithm [2], [7]

This algorithm assumes that T_{n+1} is strongly nonsingular; that is, each principal minor is nonzero. This condition does not seem to be unduly restrictive for the problems in which we are interested. Because the algorithm computes the inverse of T_{n+1} , it is then necessary to multiply the constant vector by the inverse to obtain the solution of the system of equations. The number of multiplications and divisions required by the algorithm is $0(n^2)$ thus the total number required for solution of the equations is also $0(n^2)$. Auxiliary storage required can be as small as $2n+1$, although storage of the complete inverse is simpler and requires an additional $(n+1)^2$ locations.

2.3 Bareiss Algorithm [3]

This algorithm solves for the solution of the system of equations directly, using an elimination procedure. The algorithm requires $0(n^2)$ multiplications and divisions. About $4n+4$ auxiliary storage locations are necessary. Coding of the algorithm is difficult unless one can commit much larger (about $2(n+1)^2$ locations) blocks of auxiliary storage.

2.4 Gauss Elimination

The subroutine LEQT1F coded by IMSL is a standard Gauss elimination routine which uses scaled partial pivoting, the numerical properties of which are well documented [6]. This subroutine was used in both double and single precision form. The single precision form was used for comparison with the other routines, and double precision was used as a standard for comparison of accuracies.

3.0 Results

The results of the study are presented in two tables. One gives the typical run times, while the other lists the error in the solutions obtained. The algorithms were programmed in Fortran IV for the IBM 360 model 67 at the Naval Postgraduate School. Single precision (Real *4) was used, which on the 360 embodies a mantissa of 6 hexadecimal digits, about the equivalent of 6-7 decimal digits. The programs were compiled under the H compiler, which generates optimized object code.

The Toeplitz matrix of coefficients was formed from a 250-point autocorrelation function of a baseband signal pulse plus noise. Systems of order 4 through 128 were generated by sampling this autocorrelation

function at different rates. Five separate sets of data were generated by starting the sampling at different points; then thirty distinct systems of equations were solved by each of the four algorithms, and by the double precision LEQT1F. Because of the special form of the equations required by the Levinson algorithm (Section 2.1), the constant vector was taken as required by that algorithm.

Specifically, the sequence of elements for the first row of the Toeplitz coefficient matrix were selected, for each data set, as follows. Let $a_1, a_2, .. a_{250}$ denote the elements of the first row of the auto-correlation matrix previously mentioned. For each data set we begin with an initial index, k, and a function s_n which was either taken as $s_n = 1$ or $s_n = \frac{128}{n+1}$, where $n+1$ is the order the system to be selected, and was taken as 4, 8, 16, 32, 64, or 128. The subset of elements selected from the a_i sequence was then $a_{k+j\,s_n}$, $j = 1, 2 \ldots, n+1$. Five different data sets were chosen in this manner.

Our goal in testing the algorithms was to try to obtain some information about their execution times, and more particularly, about their accuracy. Thus our test included quite large systems of equations, not because they are of practical interest, but rather to determine something about the accuracy limits of the algorithms.

While there are some variations, the information in Tables 1 - 3 show the following: (i) The Levinson algorithm is about 3 times as fast as the Trench algorithm, which is about the same speed as the Bareiss algorithm for small systems and about 5% faster for large systems. The Bareiss algorithm is about $2 + \frac{n+1}{16}$ times as fast as Gauss elimination, for large values of $n+1$. For small values ($n+1$ less than about 16) this relation

does not hold. This last result is to be expected since Gauss elimination requires $O(n^3)$ multiplications and divisions. (ii) The desirability of the algorithms in terms of accuracy is generally the reverse of their rating in terms of execution speeds. The Levinson algorithm gave results good to only 3 significant digits on one system of order 16, and in two instances failed completely (errors within an order of magnitude of the solution) on systems of order 32. On one system of order 64, the algorithm did well, while generally failing for order 128. This algorithm is intended for positive definite matrices, which our test matrices did not always satisfy.

The Trench algorithm has larger errors in virtually every case than does the Levinson algorithm. The Bareiss algorithm does well, almost as well overall as standard Gauss elimination, although it did completely fail on two systems, one of order 32. This latter system caused all the algorithms to fail, however.

The standard Gauss elimination algorithm completely failed on only one system, the above noted one of order 32. Performance was marginal (about two digits accurate) on 3 other systems. This information, we believe, reveals more about the inherent difficulties of the equations than the capabilities of the method, which are well known. Thus, any particular algorithm should not be faulted for failing when Gauss elimination fails.

Comparing the algorithms on this basis for various allowable errors yields the information in Table 4. The success of the Bareiss algorithm for higher accuracies is probably a result of fewer arithmetic

operations required, as this may be more important than numerical stability on the small systems of equations involved there.

Finally, we observe that the tables of maximum errors and rms errors yield essentially the same information, since the rms errors are usually 2-4 times smaller than the maximum errors. In table 4 the allowed rms error was tabulated at a value of one-half the allowed maximum error to partially compensate for that fact.

Order (n+1)	Levinson	Trench	Bareiss	Gauss
4	.7	1.7	1.6	2.9
8	1.8	6.2	6.0	9.8
16	6.2	20.2	20.3	43.6
32	22.2	74.4	76.2	245.0
64	80.2	281.3	294.1	1034.4
128	315.6	1107.3	1168.5	11118.0

TABLE 1. Average observed execution times (msec)

Case	n+1	Levinson	Trench	Bareiss	Gauss
	4	1.5(-5)	1.7(-5)	1.4(-5)	2.4(-5)
	8	4.3(-5)	6.7(-6)	6.7(-6)	4.3(-5)
I	16	5.4(-5)	1.7(-4)	1.2(-4)	7.2(-5)
max norm of	32	4.4(-2)	3.3(-2)	7.2(-4)	3.0(-4)
solution	64	1.0(-2)	1.3(-2)	2.0(-3)	5.7(-4)
1.4(1)	128	6.8(0)	2.6(1)	7.0(-2)	1.8(-1)
	4	4.9(-5)	2.2(-4)	6.9(-5)	6.7(-5)
	8	6.6(-5)	1.7(-4)	8.6(-5)	9.0(-5)
II	16	1.5(-4)	2.6(-4)	9.1(-5)	1.0(-4)
max norm of	32	1.5(-4)	6.1(-4)	8.6(-5)	9.0(-5)
solution	64	3.0(-4)	1.2(-3)	8.8(-5)	9.1(-5)
1.7(0)	128	1.7(-1)	2.6(-2)	6.4(-4)	9.8(-4)
	4	6.1(-5)	1.0(-4)	1.1(-5)	1.5(-5)
	8	9.7(-4)	1.1(-3)	1.7(-5)	7.2(-5)
III	16	8.5(-3)	1.2(-2)	1.4(-5)	2.0(-5)
max norm of	32	3.2(0)	2.0(0)	3.1(-2)	1.1(-2)
solution	64	4.7(-2)	3.2(-1)	1.1(-1)	1.6(-3)
4.6(0)	128	4.4(-1)	3.0(1)	1.8(0)	4.0(-3)
	4	3.2(-5)	3.1(-5)	2.0(-5)	2.8(-5)
	8	6.6(-5)	5.2(-5)	3.8(-6)	7.0(-6)
IV	16	3.5(-4)	5.1(-4)	9.3(-5)	2.6(-5)
max norm of	32	1.1(1)	1.3(1)	1.5(1)	4.4(-1)
solution	64	6.0(0)	9.6(-1)	1.1(-1)	2.7(-2)
1.7(0)	128	1.7(-1)	2.0(-1)	7.3(-3)	3.0(-3)
	4	7.7(-7)	1.5(-6)	1.0(-6)	9.5(-7)
	8	8.3(-7)	1.6(-6)	1.3(-6)	8.3(-7)
V	16	3.5(-6)	6.1(-6)	6.2(-6)	5.2(-6)
max norm of	32	7.6(-5)	1.0(-4)	4.7(-5)	3.0(-5)
solution	64	3.2(-1)	3.0(-2)	3.0(-3)	5.9(-4)
1.7(0)	128	1.7(-1)	2.6(-2)	6.4(-4)	9.8(-4)

TABLE 2: Maximum observed errors

Case	n+1	Levinson	Trench	Bareiss	Gauss
	4	1.0(-5)	9.4(-6)	8.6(-6)	1.4(-5)
	8	2.2(-5)	4.8(-4)	4.3(-6)	2.6(-5)
I	16	2.7(-5)	1.2(-4)	8.3(-5)	4.1(-5)
rms	32	2.2(-2)	1.7(-2)	2.8(-4)	1.3(-4)
of solution	64	3.4(-3)	6.4(-3)	1.0(-3)	3.0(-4)
3.5(0)	128	1.8(0)	7.4(0)	2.4(-2)	5.0(-2)
	4	3.9(-5)	1.2(-4)	4.2(-5)	4.1(-5)
	8	4.4(-5)	1.0(-4)	4.4(-5)	4.4(-5)
II	16	7.2(-5)	1.1(-4)	4.7(-5)	4.8(-5)
rms	32	7.8(-5)	1.4(-4)	4.2(-5)	4.0(-5)
of solution	64	9.8(-5)	1.8(-4)	4.3(-5)	4.2(-5)
6.5(-1)	128	3.8(-2)	6.2(-3)	2.4(-4)	4.3(-4)
	4	4.3(-5)	5.5(-5)	7.5(-6)	1.0(-5)
	8	6.3(-4)	6.2(-4)	8.7(-6)	4.3(-5)
III	16	4.6(-3)	6.2(-3)	8.4(-6)	1.0(-5)
rms	32	1.5(0)	9.1(-1)	1.5(-2)	5.3(-3)
of solution	64	2.2(-2)	1.4(-1)	5.3(-2)	7.2(-4)
4.8(0)	128	1.6(-1)	1.3(1)	7.6(-1)	1.5(-3)
	4	3.1(-5)	2.8(-5)	1.6(-5)	2.3(-5)
	8	4.0(-5)	3.6(-5)	2.5(-6)	3.1(-6)
IV	16	1.8(-4)	3.1(-4)	5.1(-5)	1.2(-5)
rms	32	6.9(0)	8.1(0)	9.4(0)	2.7(-1)
of solution	64	2.2(0)	3.3(-1)	3.6(-2)	8.3(-3)
5.8(0)	128	7.4(-2)	7.0(-2)	2.9(-3)	8.6(-4)
	4	6.0(-7)	7.6(-7)	8.1(-7)	6.8(-7)
	8	5.1(-7)	7.5(-7)	7.5(-7)	5.3(-7)
V	16	2.2(-6)	2.8(-6)	3.5(-6)	3.4(-6)
rms	32	2.9(-5)	4.2(-5)	2.3(-5)	1.5(-5)
of solution	64	1.6(-1)	1.3(-2)	1.2(-3)	3.0(-4)
7.1(-1)	128	3.8(-2)	6.2(-3)	2.4(-4)	4.3(-4)

TABLE 3: rms errors

Maximum error allowed	Levinson	Trench	Bareiss	Gauss
1.0(-5)	3 - 75%	3 - 75%	5 - 125%	4
1.0(-4)	12 - 70%	6 - 35%	18 - 106%	17
1.0(-3)	17 - 74%	15 - 65%	21 - 91%	23
1.0(-2)	18 - 69%	17 - 65%	24 - 92%	26
1.0(-1)	21 - 75%	23 - 82%	26 - 93%	28
rms error allowed				
5.0(-6)	3 - 75%	3 - 75%	5 - 125%	4
5.0(-5)	12 - 67%	7 - 39%	16 - 89%	18
5.0(-4)	16 - 70%	16 - 78%	21 - 91%	23
5.0(-3)	19 - 73%	17 - 65%	24 - 92%	26
5.0(-2)	23 - 79%	23 - 79%	27 - 93%	29

TABLE 4:

Successful runs as a percent of those which were successful with Gauss elimination.

4.0 Conclusion

On the basis of our tests for numerical accuracy, we recommend the use of Levinson's algorithm for small $(n+1 \leq 16)$ systems to which it is applicable. The size of system successfully solved by Levinson's algorithm is probably larger if positive definiteness can be assured, but otherwise it appears large errors may occur. If Levinson's algorithm is not applicable, the algorithm of choice is the Bareiss algorithm. It is slightly slower than the Trench algorithm on large systems, but almost always gives better results, and is nearly as good as Gauss elimination in most cases. Unless the inverse of the matrix is explicitly needed, we cannot recommend the use of the Trench algorithm at all.

REFERENCES

1. Douglas H. Preis, "The Toeplitz Matrix: It's occurence in antenna problems and a rapid inversion algorithm," IEEE Trans. on Ant. & Prop., March 1972, pp. 204-206.

2. S. Zohar, "Toeplitz matrix inversion: the algorithm of W. F. Trench," JACM 16(1969) 592-601.

3. Edwin H. Bareiss, "Numerical solution of linear equations with Toeplitz and Vector Toeplitz Matrices," Numer Math 13 (1969) 404-424.

4. M. J. Knudsen, "Real-Time Linear-Predictive Coding of Speech on the SPS-41 Triple Micro Processor MACHINE," IEEE Trans. on Acoustics, Speech, and Signal Processing. Vol. ASSP-23, No. 1, Feb 1975, 140-145.

5. J. Makhoul, "Linear Prediction: A Tutorial Review," Proc. IEEE, Vol 63, No. 4, April 1975, 565-566.

6. George Forsythe and Cleve B. Moler, "Computer Solution of Linear Algebraic Systems," Prentice-Hall, 1967.

7. W. F. Trench, "An Algorithm for the Inversion of Finite Toeplitz Matrices," J. SIAM 12 (1964), 515-522.

8. D. C. Farden and L. L. Scharf, "Statistical Design of Nonrecursive Digital Filters," IEEE Trans. Acoustics, Speech, and Signal Processing, ASSP-22, No. 3, June 1974, 188-196.

9. P. Butler, "Comments on Statistical Design of Nonrecursive Digital Filters," IEE Trans. on Acoustics, Speech, and Signal Processing, Vol ASSP-23, No. 5, Oct. 1975, 494-495.

10. D. C. Farden and L. L. Scharf, "Authors' Reply," OP CIT, 495-497.

Distribution List

	No. of copies
Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
Library Naval Postgraduate School Monterey, California 93940	2
Dean of Research Naval Postgraduate School Monterey, California 93940	2
Professor Ladis D. Kovach Chairman, Department of Mathematics Naval Postgraduate School Monterey, California 93940	1
Professor C. Comstock Department of Mathematics Naval Postgraduate School Monterey, California 93940	1
Professor Frank D. Faulkner Department of Mathematics Naval Postgraduate School Monterey, California 93940	1
Dr. Richard Lau Office of Naval Research Pasadena, California 91100	1
LT Edison L. Bell 1731 Leasure Way Crofton, Maryland 21114	5
Professor Richard Franke Department of Mathematics Naval Postgraduate School Monterey, California 93940	5