Parallel Algorithms and Numerical Stability for Toeplitz Solvers*

Richard P. Brent
Computer Sciences Laboratory
Australian National University
rpb@cslab.anu.edu.au

August 1993

*Presented at *SIAM LASSC*, Seattle, August 1993.
 rpb143t

Outline

There are several surveys of algorithms for solving Toeplitz linear systems with an emphasis on their numerical properties and on possibilities for parallelism[1]. Today, I shall concentrate on a few interesting results.
An outline of the talk:

- Stability and weak stability.
- The Levinson-Durbin algorithm for positive definite symmetric Toeplitz matrices (Yule-Walker equations).
- The Bareiss algorithm for general Toeplitz matrices.
- The BBH algorithm for general Toeplitz matrices and least squares problems.
- A weakly stable algorithm for general Toeplitz solvers.
- Parallelism (mentioned as we go).

[1] See for example Bunch (1985), Brent (1991).

Notation

Let

$$A = \begin{pmatrix} a_0 & \cdots & a_{n-1} \\ \vdots & \ddots & \vdots \\ a_{1-m} & \cdots & a_{n-m} \end{pmatrix}$$

be a real $m \times n$ Toeplitz matrix, so

$$a_{i,j} = a_{j-i}$$

for $1 \le i \le m$, $1 \le j \le n$. We assume that $m \ge n$ and that A has rank n. Often, for simplicity, we assume $m = n$.

$A^T A$ has a Cholesky factorization

$$A^T A = R^T R,$$

where R is an upper triangular $n \times n$ matrix. We assume that the diagonal elements of R are positive, so R is unique. Also,

$$A = QR,$$

where Q is an $m \times n$ matrix with orthonormal columns, i.e.

$$Q^T Q = I.$$

Condition Number of A

If the singular values of A are $\sigma_1, \ldots, \sigma_n$, where $\sigma_1 \ge \ldots \ge \sigma_n > 0$, then the spectral condition number of A is

$$\kappa = \kappa_2(A) = \sigma_1 / \sigma_n.$$

We say that A is *well-conditioned* if $\kappa(A)$ is "small" in some sense.
For convenience in stating the error bounds, we often assume that σ_1 is of order unity.
Let B be a principal $k \times k$ submatrix of A.
If A is symmetric positive definite then (by an interlacing theorem for eigenvalues)

$$\kappa_2(B) \le \kappa_2(A).$$

However, in general this is not true –
B could be badly conditioned or even singular when A is well-conditioned.

A Useful Operator

A *displacement operator*

$$\mathcal{D} : \Re^{m \times n} \to \Re^{(m-1)\times(n-1)}$$

is defined as follows: for any $m \times n$ matrix B,

$$\mathcal{D}(B) = C,$$

where C is the $(m-1) \times (n-1)$ matrix with entries

$$c_{i,j} = b_{i+1,j+1} - b_{i,j},$$

$1 \le i < m, 1 \le j < n$.

Note that $\mathcal{D}B = 0$ iff B is Toeplitz.

Acronyms

BBH	=	Bojanczyk, Brent and de Hoog.
BBDH	=	Bojanczyk, Brent, Van Dooren and de Hoog.
BBHS	=	Bojanczyk, Brent, de Hoog and Sweet.

Error Bounds and "O" Notation

Let ε be the machine precision. It is convenient to subsume a polynomial in m and n into the "O" notation[2]. Thus, an error bound of the form $\|E\| = O(\varepsilon)$ will mean that

$$\|E\| \le P(m,n)\varepsilon$$

for some polynomial P and all sufficiently small ε. If the error bound depends on κ then this will be mentioned explicitly, e.g.

$$\|E\| = O(\kappa\varepsilon).$$

We shall ignore $O(\varepsilon^2)$ terms in the error analyses.
The meaning of "sufficiently small" may depend on κ (for example, we may need $\kappa^2\varepsilon < 1$).

[2] If the aim of error analysis is insight, then it is best not to obscure the results with unimportant details. To avoid ambiguity, we could write $O_{m,n}(\varepsilon)$ or $O^*(\varepsilon)$.

Numerical Quantities

We distinguish several classes of numerical quantities –

1. Exact values, e.g. input data such as a_i.
2. Computed values, usually indicated by a tilde, e.g. $\tilde{u}_i$.
3. Perturbed values given by error analysis, usually indicated by a hat, e.g. $\widehat{a}_{i,j}$, or by the argument ε, e.g. $a_{i,j}(\varepsilon)$.
   These are not computed, but the error analysis shows that they exist and gives bounds on their difference from the corresponding exact values.

Stability and Strong Stability

Consider algorithms for solving a nonsingular, $n \times n$ linear system $Ax = b$.
There are many definitions of numerical stability in the literature. Our definitions follow those of Bunch(1987). Definition 1 says that the *computed* solution has to be the *exact* solution of a problem which is close to the original problem. This is the classical *backward stability* of Wilkinson.

Definition 1 *An algorithm for solving linear equations is* stable *for a class of matrices* $\mathcal{A}$ *if for each* A *in* $\mathcal{A}$ *and for each* b *the computed solution* $\tilde{x}$ *to* $Ax = b$ *satisfies* $\widehat{A}\tilde{x} = \widehat{b}$*, where* $\widehat{A}$ *is close to* A *and* $\widehat{b}$ *is close to* b.

Note that the matrix $\widehat{A}$ does not have to be in the class $\mathcal{A}$. For example, $\mathcal{A}$ might be the class of nonsingular Toeplitz matrices, but $\widehat{A}$ need not be a Toeplitz matrix. If we do require $\widehat{A} \in \mathcal{A}$ we get what Bunch calls *strong stability*.

Closeness

In Definition 1, "close" means close in a relative sense, using some norm, i.e.

$$\|\widehat{A} - A\|/\|A\| = O(\varepsilon),\ \|\widehat{b} - b\|/\|b\| = O(\varepsilon).$$

Recall our convention that polynomials in n may be omitted from $O(\varepsilon)$ terms.
We are ruling out faster than polynomial growth in n, such as $O(2^n\varepsilon)$ or $O(n^{\frac{\log n}{4}}\varepsilon)$ (Gaussian elimination). Is this too strict ?

How Good is the Solution ?

Provided $\kappa\varepsilon$ is sufficiently small, stability implies that

$$\|\tilde{x} - x\|/\|x\| = O(\kappa\varepsilon).$$

Example – Gaussian Elimination

For Gaussian elimination with pivoting, Wilkinson shows that

$$\|\widehat{A} - A\| = O(g\varepsilon),$$

where $g = g(n)$ is the "growth factor". g depends on whether partial or complete pivoting is used. In practice g is usually small, even for partial pivoting. However, a well-known example shows that $g(n) = 2^{n-1}$ is possible for partial pivoting, and examples from boundary value problems show that exponential growth can occur in practice. Even for complete pivoting, it has not been proved that $g(n)$ is bounded by a polynomial in n. The conjecture $g(n) \le n$ was recently disproved by Gould.

Thus, $\mathcal{A}$ must be restricted to matrices with some special property (e.g. positive definiteness or diagonal dominance) or to matrices of fixed (or bounded) size in order for Gaussian elimination with pivoting to satisfy Definition 1.

[Is the definition is too strict ?]

Weak Stability

Definition 2 *An algorithm for solving linear equations is* weakly stable *for a class of matrices $\mathcal{A}$ if for each well-conditioned A in $\mathcal{A}$ and for each b the computed solution $\tilde{x}$ to $Ax = b$ is such that $\|\tilde{x} - x\|/\|x\|$ is small.*

In Definition 2, "small" means $O(\varepsilon)$, and "well-conditioned" means that $\kappa(A)$ is bounded by a polynomial in n. It is easy to see that stability implies weak stability.
Define the *residual*

$$r = A\tilde{x} - b.$$

It is well-known that

$$\frac{1}{\kappa}\frac{\|r\|}{\|b\|} \le \frac{\|\tilde{x} - x\|}{\|x\|} \le \kappa\frac{\|r\|}{\|b\|}.$$

Thus, for well-conditioned A, $\|\tilde{x} - x\|/\|x\|$ is small if and only if $\|r\|/\|b\|$ is small. (This gives an equivalent definition of weak stability.)

Cholesky Factorization

Consider the computation of the Cholesky factor R of A^TA, where A is an $m \times n$ matrix of full rank n. A good $O(mn^2)$ algorithm is to compute the QR factorization

$$A = QR$$

of A using Householder or Givens transformations. The computed matrices $\tilde{Q}$, $\tilde{R}$ satisfy

$$\widehat{A} = \widehat{Q}\tilde{R}$$

where $\widehat{Q}^T\widehat{Q} = I$, $\tilde{Q}$ is close to $\widehat{Q}$, and $\widehat{A}$ is close to A. The algorithm is stable.
A different algorithm is to compute (the upper triangular part of) A^TA, and then compute the Cholesky factorization of A^TA by the usual (stable) algorithm. The computed result $\tilde{R}$ is such that $\tilde{R}^T\tilde{R}$ is close to A^TA. However, this does not imply the existence of $\widehat{A}$ and $\widehat{Q}$ as above, unless A is well-conditioned (Stewart).
We may say that we have a *weakly stable* algorithm for computing R, whereas QR factorization is a *stable* algorithm for computing R.

The Levinson-Durbin Algorithm

In 1947, Levinson gave an algorithm for solving a symmetric $n \times n$ Toeplitz system in $O(n^2)$ operations and $O(n)$ storage. In linear prediction we want to solve a Toeplitz system with a special right-hand side, called the "Yule-Walker equations":

$$Ax = b,$$

where

$$b = -(a_1, a_2, \ldots, a_n)^T.$$

Durbin (1960) streamlined Levinson's algorithm for this special case.
In the linear prediction problem, A is symmetric positive definite, so we assume this when discussing the Levinson-Durbin algorithm. It is also convenient to assume that $a_0 = 1$ and that $\|b\|$ is not too small (if $\|b\|$ is small then A is close to I).

The Levinson-Durbin Recursion

The algorithm is defined by the recursion (for $j = 1, 2, \ldots, n$):

$$\begin{aligned} K_j &= -(a_j + x_{j-1,1}a_{j-1} + \cdots \\ &\quad +x_{j-1,j-1}a_1)/E_{j-1}, \\ x_j &= \begin{pmatrix} x_{j-1} + K_j x_{j-1}^R \\ K_j \end{pmatrix}, \\ E_j &= (1 - K_j^2)E_{j-1}, \end{aligned}$$

where $E_0 = 1$,

$$x_j = (x_{j,1}, x_{j,2}, \ldots, x_{j,j})^T,$$

and

$$x_j^R = (x_{j,j}, x_{j,j-1}, \ldots, x_{j,1})^T$$

is the reflection of x_j. One can verify by induction that x_j solves the j-th order Yule-Walker equations and

$$x = x_n = (x_{n,1}, x_{n,2}, \ldots, x_{n,n})^T$$

solves $Ax = b$.

Interpretation of K_j and E_j

The K_j are called "reflection coefficients" or "partial correlation coefficients" and satisfy

$$-1 < K_j < 1.$$

Note that the formula for K_j involves an inner product.

The E_j are the mean square prediction errors.

Matrix Factorization Interpretation

If

$$C = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ x_{n-1,1} & 1 & 0 & \cdots & 0 \\ x_{n-1,2} & x_{n-2,1} & 1 & & 0 \\ \vdots & \vdots & & \ddots & \vdots \\ x_{n-1,n-1} & x_{n-2,n-2} & & \ldots & 1 \end{pmatrix}$$

and

$$D = \mathrm{diag}(E_{n-1}, E_{n-2}, \ldots, E_0)$$

then

$$A^{-1} = CD^{-1}C^T,$$

so the Levinson-Durbin algorithm computes a Cholesky factorization of A^{-1}. The factorization can be used to find upper and lower bounds on $\|A^{-1}\|$ (which is much the same thing as $\kappa(A)$).

Note that A is sure to be poorly conditioned if E_{n-1} is small, i.e. if the mean square prediction error is small. (But this is what we want – Catch 22 !)

Theorem 1 (Cybenko, 1980)

$$\max\left(\frac{1}{E_{n-1}}, \frac{1}{\prod_{j=1}^{n-1}(1-K_j)}\right)$$
$$\le \|A^{-1}\|_1 \le \prod_{j=1}^{n-1} \frac{1+|K_j|}{1-|K_j|},$$

where

$$E_{n-1} = \prod_{j=1}^{n-1}(1-K_j^2),$$

and the K_j are the reflection coefficients.

Lemma 1 (Cybenko)

$$\|x\|_1 \ge \left|\prod_{j=1}^{n}(1+K_j) - 1\right|$$

Theorem 2 (Cybenko)
If floating-point arithmetic is used with machine precision ε, then the residual $r = A\tilde{x} - b$ satisfies

$$\|r\| = O\left(\varepsilon\left(\prod_{j=1}^{n}(1+|K_j|) - 1\right)\right).$$

Corollary 1
If all $K_j \ge 0$, then $\|r\|/\|x\| = O(\varepsilon)$.

Corollary 2
If all the reflection coefficients are non-negative, then the Levinson-Durbin algorithm is weakly stable.

Comparison with Cholesky

Cybenko notes that (in the case that $K_j \ge 0$) the bounds on the residual for the Levinson-Durbin method and Cholesky's method are of comparable size. However, this does not prove that the Levinson-Durbin method is stable (in the sense of Definition 1).

What if some $K_j < 0$?

Cybenko's analysis is not sharp if some of the reflection coefficients are negative, because of the absolute values $|K_j|$ in his inequalities[3]. However, from Cybenko's results we can deduce a bound which is similar to that for Gaussian elimination with partial pivoting. The proof is easy, but I have not seen it stated before. From Theorem 1,

$$\frac{1}{\kappa_1(A)} \le \frac{1}{\|A^{-1}\|_1} \le E_{n-1} = \prod_{1}^{n-1}(1-K_j^2),$$

so, from Theorem 2,

$$\frac{\|r\|}{\kappa\varepsilon} = O(M^{n-1}),$$

where

$$M = \max_{-1<K<1}(1+|K|)(1-K^2) = \frac{32}{27}$$

(the maximum occurs at $|K| = \frac{1}{3}$). Thus

[3]Koltracht and Lancaster (1986) have improved Cybenko's upper bound on $\kappa(A)$ in this case.

Theorem 3
For the Levinson-Durbin algorithm with positive definite symmetric A, but no restriction on the signs of the reflection coefficients,

$$\|r\| = O\left(\kappa\varepsilon\left(\frac{32}{27}\right)^n\right).$$

This is not too bad when compared with the bound $O(2^n\varepsilon)$ for Gaussian elimination with partial pivoting. In both cases, $\|r\|$ is usually much smaller than the corresponding bound.

Corollary 3
The Levinson-Durbin algorithm for solving the positive definite symmetric Yule-Walker equations of bounded size n is weakly stable.

Numerical experiments (BBHS, Varah) suggest that weak stability is all that we can expect to prove.

The Bareiss Algorithm

The algorithms of Levinson, Durbin, Trench and Zohar find an LU factorization of A^{-1} and (in the symmetric case) are related to the classical Szegö recursions for polynomials orthogonal on the unit circle. These algorithms involve inner products, so it is not obvious how to use more than $\Theta(n/\log n)$ processors efficiently in their parallel implementation.

Another class of algorithms, typified by the algorithm of Bareiss (1969), find an LU factorization of A, and (in the symmetric case) are related to the classical algorithm of Schur for the continued fraction representation of a holomorphic function in the unit disk. These algorithms avoid inner products, and it is straightforward to get speedups of order n when using n processors in parallel.

Can we use more than $O(n)$ processors to get speedup greater than n with reasonable efficiency ?

Numerical Properties of the Bareiss Algorithm

Sweet (1982–1993) and BBHS (1993) have shown that the numerical properties of the Bareiss algorithm when implemented in floating-point arithmetic are similar to those of Gaussian elimination (without pivoting).
Thus, the algorithm is stable for positive definite symmetric A. This is a stronger result than has been proved for the Levinson algorithm – we only showed that it was weakly stable for bounded n.
For general Toeplitz A the Bareiss algorithm is unstable, just like Gaussian elimination without pivoting. In fact, both break down immediately if $a_{1,1} = 0$, and exhibit instability if $a_{1,1}$ is small.
Sweet (1993) has shown that it is possible to introduce pivoting into the Bareiss algorithm to avoid instability[4]. However, in the worst case the overhead of pivoting is $O(n^3)$ so we no longer have a "fast" $O(n^2)$ algorithm.

[4] Using a connection between the Bareiss multipliers and the Trench-Zohar algorithm, Sweet also shows how to introduce pivoting into the Trench-Zohar algorithm.

Fast Orthogonal Factorization

In an attempt to achieve stability without pivoting, it is natural to consider algorithms for computing an orthogonal factorization

$$A = QR$$

of A. The first such $O(n^2)$ algorithm was introduced by Sweet (1982–84). Unfortunately, Sweet's algorithm is unstable[5]. Other $O(n^2)$ algorithms for computing the matrices Q and R or R^{-1} were given by BBH (1986), Chun *et al* (1987), Cybenko (1987), and Qiao (1988), but none of them has been shown to be stable (or weakly stable). In several cases examples show that they are not stable. Unlike the classical $O(n^3)$ Givens or Householder algorithms, the $O(n^2)$ algorithms do not form Q in a numerically stable manner as a product of matrices which are (close to) orthogonal.

[5] It depends on the condition of a submatrix of A – see Luk and Qiao (1987).

Experiments with the BBH Algorithm

Numerical experiments with the algorithm of Bojanczyk, Brent and de Hoog (BBH for short) suggest that the problem lies in the method for computing the orthogonal matrix Q; the computed upper triangular matrix $\tilde{R}$ is about as good as can be obtained by performing a Cholesky factorization of A^TA, provided the downdates involved in the algorithm are implemented in a certain way. This result has recently been proved by Bojanczyk, Brent and de Hoog (1993). As a consequence, the method of semi-normal equations (i.e. the solution of $R^TRx = A^Tb$) can be used to give a weakly stable algorithm for the solution of general Toeplitz systems and Toeplitz least squares problems.

Cholesky Downdating

The Cholesky downdating problem is: given a upper triangular matrix $R \in \Re^{n\times n}$ and a vector $x \in \Re^n$ such that $R^T R - xx^T$ is positive definite, find an upper triangular matrix U such that

$$U^T U = R^T R - xx^T.$$

The condition that $R^T R - xx^T$ be positive semi-definite is necessary for the existence of a real U. Thus, we would expect the downdating problem to be illconditioned if $R^T R - xx^T$ has small singular values, and this is what Stewart (1979) shows.

Error Analysis of Linpack Downdating

There are several algorithms for the Cholesky downdating problem. What is relevant to the BBH algorithm is the error analysis. Observe that there is an orthogonal matrix Q such that

$$\begin{pmatrix} x^T \\ U \end{pmatrix} = Q \begin{pmatrix} R \\ 0^T \end{pmatrix}.$$

Suppose the computed upper triangular matrix is $\tilde{U}$. To simplify the statement of the error bounds, suppose that $\|R\| = O(1)$.

Stewart (1979) has shown that, for the "Linpack" algorithm,

$$\begin{pmatrix} x^T(\varepsilon) \\ \tilde{U}(\varepsilon) \end{pmatrix} = Q(\varepsilon) \begin{pmatrix} R \\ 0^T \end{pmatrix},$$

where $Q(\varepsilon)$ is an exactly orthogonal matrix,

$$\|x(\varepsilon) - x\| = O(\varepsilon),$$

and

$$\|\tilde{U}(\varepsilon) - \tilde{U}\| = O(\varepsilon).$$

Analysis of BBDH "Algorithm C"

A similar result holds for downdating via "Algorithm C" of Bojanczyk, Brent, Van Dooren and de Hoog (BBDH).

We can regard $x(\varepsilon)$ as a (backward) perturbation of the input data x, and $\tilde{U}(\varepsilon)$ as a (forward) perturbation of the computed result $\tilde{U}$. Because of this mixture of forward and backward perturbations, a result of this form is sometimes called a "mixed" stability result. The mixed error bound implies that

$$\tilde{U}^T \tilde{U} = R^T R - xx^T + \varepsilon G(\varepsilon),$$

where

$$\|\varepsilon G(\varepsilon)\| = O(\varepsilon).$$

A similar result holds if a sequence of (updates and) downdates is performed, provided the intermediate and final results are positive definite.

Sketch of the BBH Algorithm

Suppose we are trying to find R such that $R^T R = A^T A$, where A is Toeplitz. First consider exact arithmetic. We partition A in two ways:

$$A = \left(\begin{array}{c|c} a_0 & y^T \\ \hline z & A_{-1} \end{array}\right) = \left(\begin{array}{c|c} A_{-1} & \overline{y} \\ \hline \overline{z}^T & a_{n-m} \end{array}\right),$$

where A_{-1} is an $(m-1) \times (n-1)$ Toeplitz matrix.
Similarly, we partition R in two ways:

$$R = \left(\begin{array}{c|c} r_{1,1} & u^T \\ \hline 0 & R_b \end{array}\right) = \left(\begin{array}{c|c} R_t & \overline{u} \\ \hline 0^T & r_{n,n} \end{array}\right),$$

where R_b and R_t are $(n-1) \times (n-1)$ upper triangular matrices.

Now

$$
\begin{aligned}
A^T A &= \left(\begin{array}{c|c} a_0^2 + z^T z & a_0 y^T + z^T A_{-1} \\ \hline a_0 y + A_{-1}^T z & A_{-1}^T A_{-1} + y y^T \end{array}\right) \\
&= \left(\begin{array}{c|c} A_{-1}^T A_{-1} + \overline{z}\,\overline{z}^T & \vdots \\ \hline \cdots & \cdot \end{array}\right).
\end{aligned}
$$

Similarly,

$$
\begin{aligned}
R^T R &= \left(\begin{array}{c|c} r_{1,1}^2 & r_{1,1} u^T \\ \hline r_{1,1} u & {R_b}^T R_b + u u^T \end{array}\right) \\
&= \left(\begin{array}{c|c} {R_t}^T R_t & \vdots \\ \hline \cdots & \cdot \end{array}\right).
\end{aligned}
$$

For future reference, it follows from these equations that

$$\mathcal{D}(A^T A) = y y^T - \overline{z}\,\overline{z}^T$$

and

$$\mathcal{D}(R^T R) = {R_b}^T R_b + u u^T - {R_t}^T R_t.$$

Equating $A^T A$ and $R^T R$, we obtain

$$r_{1,1}^2 = a_0^2 + z^T z,$$

$$
\begin{aligned}
r_{1,1} u &= a_0 y + A_{-1}^T z, \\
A_{-1}^T A_{-1} + y y^T &= {R_b}^T R_b + u u^T,
\end{aligned}
$$

and

$$A_{-1}^T A_{-1} + \overline{z}\,\overline{z}^T = {R_t}^T R_t.$$

Eliminating $A_{-1}^T A_{-1}$ gives the relation

$${R_b}^T R_b = {R_t}^T R_t + y y^T - u u^T - \overline{z}\,\overline{z}^T$$

which is the basis for the BBH algorithm.

Computation of R_b and R_t in the BBH Algorithm

Recall that

$$R = \left(\begin{array}{c|c} r_{1,1} & u^T \\ \hline 0 & R_b \end{array}\right) = \left(\begin{array}{c|c} R_t & \overline{u} \\ \hline 0^T & r_{n,n} \end{array}\right)$$

and

$${R_b}^T R_b = {R_t}^T R_t + y y^T - u u^T - \overline{z}\,\overline{z}^T.$$

If R_t were known, then R_b could be computed using one Cholesky updating step and two Cholesky downdating steps. Also, since updating and downdating algorithms can proceed by rows, knowledge of the first k rows of R_t is sufficient to allow the computation of the first k rows of R_b. It is easy to compute the first row of R. (For future reference, suppose that the computed first row of R is $(\tilde{r}_{1,1}, \tilde{u}^T)$.) It is clear that the k-th row of R_b defines the $(k+1)$-th row of R_t. Thus, we can compute R_t and R_b row by row.

Error Analysis of the BBH Algorithm

A straightforward extension of the mixed error analysis for Cholesky downdating applies to our problem of computing R_t and R_b. Provided a suitable variant of downdating is used, the computed results $\tilde{R}_t$ and $\tilde{R}_b$ satisfy

$$\tilde{R}_b^T \tilde{R}_b = \tilde{R}_t^T \tilde{R}_t + y y^T - \tilde{u}\tilde{u}^T - \overline{z}\,\overline{z}^T + \varepsilon G(\varepsilon)$$

where

$$\|\varepsilon G(\varepsilon)\| = O(\varepsilon).$$

Here y, $\overline{z}$ and $\tilde{u}$ are inputs to the up/downdating procedures. At this point we make no claims about the size of $\|\tilde{R}_b - R_b\|$ and $\|\tilde{R}_t - R_t\|$. All we need is that $\tilde{R}_b$ and $\tilde{R}_t$ exist and are bounded for sufficiently small ε.

The Computation of R in the BBH Algorithm

Because of the algorithm for their computation, the computed matrices $\tilde{R}_t$ and $\tilde{R}_b$ are related so that we can define the "computed R", say $\tilde{R}$, in a consistent manner by

$$\tilde{R} = \left(\begin{array}{c|c} \tilde{r}_{1,1} & \tilde{u}^T \\ \hline 0 & \tilde{R}_b \end{array}\right) = \left(\begin{array}{c|c} \tilde{R}_t & \vdots \\ \hline 0^T & \cdot \end{array}\right).$$

Thus

$$\tilde{R}^T\tilde{R} = \left(\begin{array}{c|c} \tilde{r}_{1,1}^2 & \tilde{r}_{1,1}\tilde{u} \\ \hline \vdots & \tilde{R}_b^T\tilde{R}_b + \tilde{u}\tilde{u}^T \end{array}\right) = \left(\begin{array}{c|c} \tilde{R}_t^T\tilde{R}_t & \vdots \\ \hline \cdots & \cdot \end{array}\right).$$

Recalling our definition of the operator $\mathcal{D}$, we have

$$\begin{aligned} \mathcal{D}(\tilde{R}^T\tilde{R}) &= \tilde{R}_b^T\tilde{R}_b + \tilde{u}\tilde{u}^T - \tilde{R}_t^T\tilde{R}_t \\ &= yy^T - \overline{z}\overline{z}^T + \varepsilon G(\varepsilon). \end{aligned}$$

Also,

$$\mathcal{D}(A^TA) = yy^T - \overline{z}\,\overline{z}^T.$$

Bounds on E and F

If $E = \tilde{R}^T\tilde{R} - A^TA$ and $F = \mathcal{D}(E)$ then

$$F = \varepsilon G(\varepsilon).$$

If $1 \le j \le i \le n$ then, by the definition of $\mathcal{D}(E)$,

$$e_{i,j} - e_{i-j+1,1} = f_{i-1,j-1} + f_{i-2,j-2} + \cdots + f_{i-j+1,1}.$$

The first row of $\tilde{R}^T\tilde{R}$ is $\tilde{r}_{1,1}(\tilde{r}_{1,1}, \tilde{u}^T)$, which is close to $r_{1,1}(r_{1,1}, u^T)$, so the first row of E has norm $O(\varepsilon)$. Also, E is symmetric. It follows that

$$\|E\| \le (n-1)\|F\| + O(\varepsilon) = O(\varepsilon),$$

where (as usual) a polynomial in n may be hidden by the "O" notation. Thus, after scaling to remove our assumption that $\sigma_1 = O(1)$, we have:

Theorem 4 (BBH, 1993)

If the BBH algorithm is used with the downdating steps performed in a suitable manner, then the computed Cholesky factor $\tilde{R}$ of A^TA satisfies

$$\|\tilde{R}^T\tilde{R} - A^TA\| / \|A^TA\| = O(\varepsilon).$$

Note that $\|\tilde{R}_b - R_b\|/\|R_b\|$ and $\|\tilde{R}_t - R_t\|/\|R_t\|$ may be of order $\kappa\varepsilon$, and that we have avoided[6] the computation of $Q = AR^{-1}$.

[6] An algorithm for computing Q is given in BBH (1986), but it is not recommended.

Normal and Semi-Normal Equations

Our aim is to solve a nonsingular $n \times n$ Toeplitz linear system

$$Ax = b,$$

using $O(n^2)$ arithmetic operations. In exact arithmetic, the *normal equations*

$$A^TAx = A^Tb$$

and the *semi-normal equations*

$$R^TRx = A^Tb$$

(where $R^TR = A^TA$) are equivalent to $Ax = b$. In most circumstances the use of the normal or semi-normal equations is not recommended, because the condition number $\kappa(A^TA)$ may be as large as $\kappa(A)^2$ (see Golub and Van Loan).

Use of the Semi-Normal Equations

When A is Toeplitz (but not symmetric positive definite) we may be able to justify use of the semi-normal equations. This is because we do not know any stable algorithm for solving $Ax = b$ directly with $O(n^2)$ arithmetic operations, but we can use the BBH algorithm to compute (a numerical approximation $\tilde{R}$ to) R in $O(n^2)$ operations, and then solve the seminormal equations in an additional $O(n^2)$ operations.

Weak Stability

Suppose $\tilde{R}$ is computed as in Theorem 4. We can compute an accurate approximation $\tilde{d}$ to $d = A^T b$ in $O(n^2)$ operations (using the obvious algorithm) or in $O(n \log n)$ operations (using the Fast Fourier Transform). Now solve the two triangular systems $\tilde{R}^T w = d$ and $\tilde{R}x = w$. We can expect to obtain a result $\tilde{x}$ for which

$$\|\tilde{x} - x\| / \|x\| = O(\kappa^2 \varepsilon),$$

where $\kappa = \kappa(A)$, provided $\kappa^2 \varepsilon \ll 1$. The residual $r = A\tilde{x} - b$ should satisfy

$$\|r\| / \|x\| = O(\kappa \varepsilon),$$

because $\|A^T r\| = \|A^T A \tilde{x} - A^T b\| = O(\varepsilon \|x\|)$. The method is *weakly stable* (according to Definition 2), although probably not *stable*.

Comment on the Error Bounds

In applications of the semi-normal equations, it is usually assumed that $\tilde{R}$ is computed via an orthogonal factorization of A, so there is a matrix $\widehat{A}$ such that $\tilde{R}^T \tilde{R} = \widehat{A}^T \widehat{A}$ and

$$\|\widehat{A} - A\| = O(\varepsilon).$$

However, in our case we only have $\|\tilde{R}^T \tilde{R} - R^T R\| = O(\varepsilon)$, which implies the weaker bound

$$\|\widehat{A} - A\| = O(\kappa \varepsilon)$$

by Stewart's perturbation analysis.

Iterative Refinement

As Åke Björck has shown for other applications of the semi-normal equations, it may be worth performing at least one step of iterative refinement (iterative improvement). The cost is a relatively cheap $O(n^2)$ operations.

Storage Requirements

The algorithm just described for the solution of the semi-normal equations requires working storage $O(n^2)$ words, because the upper triangular matrix R is not Toeplitz. However, it is possible to reduce the storage requirement to

- $O(n)$ words, at the expense of some increase in the error bounds, or to
- $O(n \log n)$ words, at no cost in the error bounds, but with a factor $O(\log n)$ in the time bound (using an idea of Griewank).

For details, see Bojancyzk, Brent and de Hoog (1993).

Toeplitz Least Squares Problems

If $A \in \Re^{m \times n}$ is Toeplitz with full rank n, then the semi-normal equations may be used to solve the least squares problem

$$\min \|Ax - b\|_2.$$

The use of semi-normal equations for the general full-rank linear least squares problem is discussed in detail by Björck (1987), and the only significant difference in our case is that an additional factor κ appears in some of the error bounds.

Conclusion and Open Problems

- The Levinson-Durbin algorithm is weakly stable for the symmetric positive definite Yule-Walker equations of bounded size n.
  Do we need the restriction on n ?
  Is weak stability the best possible result ?
  (Numerical experiments by Varah and BBHS suggest that it might be.)

- The Bareiss algorithm is stable for symmetric positive definite Toeplitz systems.
  What can be proved about stability and overhead of pivoted Bareiss ?
  How does it compare with the lookahead methods of Chan and Hansen, Freund and Zha, Gutknecht, etc ?

Conclusion and Open Problems cont.

- BBH with semi-normal equations is weakly stable for general nonsingular Toeplitz systems.
  Is weak stability the best possible result ?
  Is the $O(n)$ storage version less stable than the $O(n^2)$ storage version ?
  How does BBH compare with Cybenko's $AR = Q$ algorithm and the lookahead variant of Hansen and Gesmar ?
  Is there a fast, stable algorithm for computing $Q = AR^{-1}$?
  Are there comparable stability results for other fast Toeplitz QR algorithms ?

- How do the direct methods compare with iterative methods using circulant preconditioners ?

Some of the most relevant references:

References

[1] G. S. Ammar and W. B. Gragg, "Superfast solution of real positive definite Toeplitz systems", *SIMAX* 9 (1988), 61–76.

[2] E. H. Bareiss, "Numerical solution of linear equations with Toeplitz and vector Toeplitz matrices", *Numer. Math.* 13 (1969), 404–424.

[3] C. H. Bischof, C.-T. Pan and P. T. P. Tang, "A Cholesky up- and downdating algorithm for systolic and SIMD architectures", *SISSC* 14 (1993), 670-676.

[4] Å. Björck, "Stability analysis of the method of semi-normal equations for linear least squares problems", *Linear Alg. Appl.* 88/89 (1987), 31–48.

[5] A. W. Bojanczyk, R. P. Brent, P. Van Dooren and F. R. de Hoog, "A note on

downdating the Cholesky factorization", *SISSC* 8 (1987), 210–220.

[6] A. W. Bojanczyk, R. P. Brent and F. R. de Hoog, "QR factorization of Toeplitz matrices", *Numer. Math.* 49 (1986), 81–94.

[7] A. W. Bojanczyk, R. P. Brent and F. R. de Hoog, *Stability Analysis of Fast Toeplitz Linear System Solvers*, Report CMA-MR17-91, Centre for Mathematical Analysis, ANU, August 1991, 23 pp.

[8] A. W. Bojanczyk, R. P. Brent and F. R. de Hoog, *A Weakly Stable Algorithm for General Toeplitz Systems*, Technical Report, CMA, ANU, August 1993, to appear.

[9] A. W. Bojanczyk, R. P. Brent, F. R. de Hoog and D. R. Sweet, *On Stability of a Toeplitz Matrix Factorization Algorithm*, preprint, 1993.

[10] A. W. Bojanczyk and A. O. Steinhardt, "Stability analysis of a Householder-based

algorithm for downdating the Cholesky factorization", *SISSC* 12 (1991), 1255–1265.

[11] R. P. Brent, "Parallel algorithms for Toeplitz systems", *Numerical Linear Algebra, Digital Signal Processing and Parallel Algorithms* (edited by G. H. Golub and P. Van Dooren), Springer-Verlag, 1991, 75–92.

[12] J. R. Bunch, "Stability of methods for solving Toeplitz systems of equations", *SISSC* 6 (1985), 349–364.

[13] J. R. Bunch, "The weak and strong stability of algorithms in numerical linear algebra", *Linear Alg. Appl.* 88/89 (1987), 49–66.

[14] J. R. Bunch, "Matrix properties of the Levinson and Schur algorithms", *J. Numerical Linear Algebra with Applications* 1 (1992), 183–198.

[15] T. F. Chan and P. C. Hansen, "A look-ahead Levinson algorithm for

indefinite Toeplitz systems", *SIMAX* 13 (1992), 490–506.

[16] J. Chun, T. Kailath and H. Lev-Ari, "Fast parallel algorithms for QR and triangular factorization", *SISSC* 8 (1987), 899–913.

[17] G. Cybenko, "The numerical stability of the Levinson-Durbin algorithm for Toeplitz systems of equations", *SISSC* 1 (1980), 303–319.

[18] G. Cybenko, "Fast Toeplitz orthogonalization using inner products", *SIAM J. Sci. Stat. Computing* 8 (1987), 734–740.

[19] J.-M. Delosme and I. C. F. Ipsen, "Parallel solution of symmetric positive definite systems with hyperbolic rotations", *Linear Algebra Appl.* 77 (1986), 75–111.

[20] P. Delsarte, Y. V. Genin and Y. G. Kamp, "A generalisation of the Levinson algorithm for Hermitian Toeplitz matrices with any rank profile", *IEEE Trans.*

Acoustics, Speech and Signal Processing ASSP–33 (1985), 964–971.

[21] J. Durbin, "The fitting of time-series models", *Rev. Int. Stat. Inst.* 28 (1959), 229–249.

[22] R. W. Freund and H. Zha, *Formally biorthogonal polynomials and a look-ahead Levinson algorithm for general Toeplitz systems*, Technical Report 91.27, RIACS, NASA Ames Research Center, December 1991 (revised September 1992).

[23] I. Gohberg (editor), *I. Schur Methods in Operator Theory and Signal Processing* (Operator Theory: Advances and Applications, Volume 18), Birkhäuser Verlag, Basel, 1986.

[24] G. H. Golub and C. Van Loan, *Matrix Computations*, second edition, Johns Hopkins Press, Baltimore, Maryland, 1989.

[25] N. Gould, "On growth in Gaussian elimination with complete pivoting", *SIMAX* 12 (1991), 354–361.

[26] A. Griewank, “Achieving logarithmic growth of temporal and spatial complexity in reverse automatic differentiation”, *Optimization Methods and Software* 1 (1992), 35–54.

[27] P. C. Hansen and H. Gesmar, “Fast orthogonal decomposition of rank deficient Toeplitz matrices”, *Numerical Algorithms* 4 (1993), 151–166.

[28] T. Kailath, S. Y. Kung and M. Morf, “Displacement ranks of matrices and linear equations”, *J. Math. Anal. Appl.* 68 (1979), 395–407.

[29] T. Kailath, A. Vieira and M. Morf, “Inverses of Toeplitz operators, innovations and orthogonal polynomials”, *SIAM Review* 20 (1978), 106–119.

[30] A. N. Kolmogorov, “Interpolation and extrapolation of stationary random sequences”, *Izvestia Akad. Nauk SSSR* 5 (1941), 3–11 (in Russian).

[31] I. Koltracht and P. Lancaster, “Condition numbers of Toeplitz and block Toeplitz matrices”, in [23], 271–300.

[32] N. Levinson, “The Wiener RMS (Root-Mean-Square) error criterion in filter design and prediction”, *J. Math. Phys.* 25 (1947), 261–278.

[33] F. T. Luk and S. Qiao, “A fast but unstable orthogonal triangularization technique for Toeplitz matrices”, *Linear Algebra Appl.* 88/89 (1987), 495–506.

[34] C. C. Paige, “An error analysis of a method for solving matrix equations”, *Math. Comp.* 27 (1973), 355–359.

[35] C.-T. Pan, “A modification to the Linpack downdating algorithm”, *BIT* 30 (1990), 707–722.

[36] H. Park and L. Eldén, *Fast and accurate triangularization of Toeplitz matrices*, preprint, 14 April 1993.

[37] S. Qiao, “Hybrid algorithm for fast Toeplitz orthogonalization”, *Numer. Math.* 53 (1988), 351–366.

[38] I. Schur, “Über Potenzreihen, die im Innern des Einheitskreises beschränkt sind”, *J. fur die Reine und Angewandte Mathematik* 147 (1917), 205–232. English translation in [23], 31–59.

[39] G. W. Stewart, “Perturbation bounds for the QR factorization of a matrix”, *SIAM J. Numer. Anal.* 14 (1977), 509–518.

[40] G. W. Stewart, “The effect of rounding error on an algorithm for downdating a Cholesky factorization”, *J. Inst. Math. Appl.* 23 (1979), 203-213.

[41] D. R. Sweet, *Numerical Methods for Toeplitz Matrices*, PhD thesis, University of Adelaide, 1982.

[42] D. R. Sweet, “Fast Toeplitz orthogonalization”, *Numer. Math.* 43 (1984), 1–21.

[43] D. R. Sweet, “The use of pivoting to improve the numerical performance of algorithms for Toeplitz matrices”, *SIMAX* 14 (1993), 468–493.

[44] G. Szegö, *Orthogonal Polynomials*, AMS Colloquium publ. XXIII, AMS, Providence, Rhode Island, 1939.

[45] W. F. Trench, “An algorithm for the inversion of finite Toeplitz matrices”, *J. SIAM (SIAM J. Appl. Math.)* 12 (1964), 515–522.

[46] N. Wiener, *Extrapolation, Interpolation and Smoothing of Stationary Time Series, with Engineering Applications*, Technology Press and Wiley, New York, 1949 (originally published in 1941 as a Technical Report).

[47] S. Zohar, “Toeplitz matrix inversion: the algorithm of W. F. Trench”, *J. ACM* 16 (1969), 592–601.