62 Fast Algorithms for Structured Matrix Computations

Michael Stewart
Georgia State University

A structured matrix is an $m \times n$ matrix with elements that can be determined from substantially fewer than mn parameters. This is a broad definition that includes sparse matrices, matrices with displacement structure, matrices with rank structure, and numerous types of matrices with specialized structure. Many common matrix operations can be performed using fast algorithms that exploit matrix structure to achieve asymptotic speed-up relative to conventional algorithms. For matrices with either displacement structure or rank structure, there are fast algorithms for the solution of linear systems, inversion, QR factorization, and matrix-vector multiplication. Fast algorithms typically achieve a factor of n speed-up over their conventional counterparts. Algorithms that do better than this are sometimes referred to as superfast.

This chapter focuses on methods for dense matrices with structure. The emphasis is on direct methods, with some limited coverage of the use of fast matrix-vector multiplication in iterative methods and the choice of preconditioners for structured systems. Displacement structure is used as a unifying context for the presentation of many of the fast algorithms. Toeplitz, Vandermonde, and Cauchy matrices are covered in detail. A notable omission is the topic of fast algorithms for matrices with rank structure.

Unless otherwise noted, all matrices are assumed to be complex.

62.1 Classes of Structured Matrices

Displacement equations were introduced to derive factorization algorithms and inversion formulas for matrices with an almost Toeplitz structure [FMK79]. Since then the concept has been generalized and has served as a unifying concept in the study of algorithms for a wide range of structured matrices. This approach allows uniform derivation of fast algorithms for LU, QR, and Cholesky factorization of Toeplitz, Vandermonde, Cauchy, Hankel, and related matrices.

Definitions:

Let $A \in \mathbb{C}^{m\times n}$, $E \in \mathbb{C}^{m\times m}$, $G \in \mathbb{C}^{m\times m}$, $F \in \mathbb{C}^{n\times n}$ and $H \in \mathbb{C}^{n\times n}$.

Let the vector $\mathbf{v}_n(\alpha)$ be given by

$$\mathbf{v}_n(\alpha) = \begin{bmatrix} 1 & \alpha & \alpha^2 & \cdots & \alpha^{n-1} \end{bmatrix}^{\mathrm{T}}$$

and $\mathbf{p}_n(\alpha)$ by

$$\mathbf{p}_n(\alpha) = \begin{bmatrix} p_0(\alpha) & p_1(\alpha) & \cdots & p_{n-1}(\alpha) \end{bmatrix}^{\mathrm{T}},$$

where each $p_k(\alpha)$ is a polynomial of degree k and $\alpha \in \mathbb{C}$.

Given a vector $\boldsymbol{v} \in \mathbb{C}^n$, $D_{\boldsymbol{v}}$ is the $n \times n$ diagonal matrix with diagonal elements taken in order from the elements of $\boldsymbol{v}$.

The matrix $Z_n = [z_{jk}]$ is the $n \times n$ downshift matrix given by $z_{jk} = 1$ for $j = k+1$ and $z_{jk} = 0$ otherwise.

The matrix $Z_{\delta,n}$ denotes $Z_n + \delta \mathbf{e}_1 \mathbf{e}_n^*$.

The matrix $\Sigma_{p,q}$ is the signature matrix $\Sigma_{p,q} = I_p \oplus -I_q$.

A matrix $T = [t_{jk}] \in \mathbb{C}^{m\times n}$ is **Toeplitz** if $t_{jk} = t_{j-k}$, that is if its elements are constant along diagonals.

A Toeplitz matrix $S = [s_{jk}] \in \mathbb{C}^{n\times n}$ is **circulant** if $s_{jk} = s_{(j-k) \bmod n}$.

A matrix $H = [h_{jk}] \in \mathbb{C}^{m\times n}$ is **Hankel** if $h_{jk} = h_{j+k}$; that is, if its elements are constant along cross diagonals.

A matrix $V \in \mathbb{C}^{m\times n}$ is **Vandermonde** if

$$V = \begin{bmatrix} \mathbf{v}_n(\alpha_1)^{\mathrm{T}} \\ \mathbf{v}_n(\alpha_2)^{\mathrm{T}} \\ \vdots \\ \mathbf{v}_n(\alpha_m)^{\mathrm{T}} \end{bmatrix}$$

for some set of distinct complex nodes $\{\alpha_1, \alpha_2, \ldots, \alpha_n\}$. Some authors use the transpose of V as the definition of a Vandermonde matrix.

Given a set of p distinct complex nodes $\{\alpha_1, \alpha_2, \ldots, \alpha_p\}$ and a multiset of size n formed from the α_k by including n_k repetitions of α_k, where $\sum_{k=1}^{p} n_k = n$, a matrix $V \in \mathbb{C}^{m\times n}$ is **confluent Vandermonde** if

$$V = \begin{bmatrix} V_1(\alpha_1) \\ V_2(\alpha_2) \\ \vdots \\ V_p(\alpha_p) \end{bmatrix},$$

where

$$V_k(\alpha) = \begin{bmatrix} \mathbf{v}_n(\alpha)^{\mathrm{T}} \\ \mathbf{v}_n'(\alpha)^{\mathrm{T}} \\ \vdots \\ \mathbf{v}_n^{(n_k-1)}(\alpha)^{\mathrm{T}} \end{bmatrix}.$$

A matrix $V \in \mathbb{C}^{m\times n}$ is **polynomial Vandermonde** if

$$V = \begin{bmatrix} \mathbf{p}_n(\alpha_1)^{\mathrm{T}} \\ \mathbf{p}_n(\alpha_2)^{\mathrm{T}} \\ \vdots \\ \mathbf{p}_n(\alpha_m)^{\mathrm{T}} \end{bmatrix}$$

for some set of distinct complex nodes $\{\alpha_1, \alpha_2, \ldots, \alpha_n\}$.

A matrix $C = [c_{jk}] \in \mathbb{C}^{m\times n}$ is **Cauchy** if $c_{jk} = 1/(\alpha_j - \beta_k)$ for $\alpha_j \neq \beta_k$, $j = 1, \ldots, m$ and $k = 1, \ldots, n$.

An operator $\Delta\colon \mathbb{C}^{m\times n} \to \mathbb{C}^{m\times n}$ of the form

$$\Delta(A) = A - GAH^*$$

is a **Stein type displacement operator**.

An operator $\Delta\colon \mathbb{C}^{m\times n} \to \mathbb{C}^{m\times n}$ of the form

$$\Delta(A) = EA - AH^*$$

is a **Sylvester type displacement operator**.

An operator $\Delta\colon \mathbb{C}^{m\times n} \to \mathbb{C}^{m\times n}$ of the form

$$\Delta(A) = EAF^* - GAH^*$$

is a **general displacement operator**.

The **displacement rank** of a matrix A with respect to a given displacement operator Δ is the rank of $\Delta(A)$.

An equation of the form

$$\Delta(A) = A - GAH^* = XY^*, \tag{62.1}$$

where $A \in \mathbb{C}^{m\times n}$, $X \in \mathbb{C}^{m\times r}$, and $Y \in \mathbb{C}^{n\times r}$ with $r = \text{rank}(\Delta(A))$ is a **Stein type displacement equation**.

An equation of the form

$$\Delta(A) = EA - AH^* = XY^*, \tag{62.2}$$

where $A \in \mathbb{C}^{m\times n}$, $X \in \mathbb{C}^{m\times r}$, and $Y \in \mathbb{C}^{n\times r}$ with $r = \text{rank}(\Delta(A))$ is a **Sylvester type displacement equation**.

An equation of the form

$$EAF^* - GAH^* = XY^*, \tag{62.3}$$

where $A \in \mathbb{C}^{m\times n}$, $X \in \mathbb{C}^{m\times r}$, and $Y \in \mathbb{C}^{n\times r}$ with $r = \text{rank}(\Delta(A))$ is a **general displacement equation**.

Given a displacement equation $\Delta(A) = XY^*$, the matrices X and Y are **generators** of A.

A matrix $T \in \mathbb{C}^{m\times n}$ is **Toeplitz-like** with displacement rank r if T satisfies a displacement equation of the form

$$T - Z_m T Z_n^* = XY^* \tag{62.4}$$

or

$$Z_{1,m}^* T - T Z_{\delta,n}^* = XY^*, \tag{62.5}$$

where $X \in \mathbb{C}^{m\times r}$ and $Y \in \mathbb{C}^{n\times r}$.

A matrix $H \in \mathbb{C}^{m\times n}$ is **Hankel-like** with displacement rank r if H satisfies a displacement equation of the form

$$Z_m H - H Z_n^* = XY^*,$$

where $X \in \mathbb{C}^{m\times r}$ and $Y \in \mathbb{C}^{n\times r}$.

A matrix $V \in \mathbb{C}^{m\times n}$ is **Vandermonde-like** with displacement rank r if V satisfies a displacement equation of the form

$$D_{\mathbf{v}} V - V Z_{\delta,n} = XY^*, \tag{62.6}$$

where $X \in \mathbb{C}^{m\times r}$ and $Y \in \mathbb{C}^{n\times r}$.

A matrix $C \in \mathbb{C}^{m\times n}$ is **Cauchy-like** with displacement rank r if C satisfies a displacement equation of the form

$$D_{\mathbf{v}} C - C D_{\mathbf{w}} = XY^*, \tag{62.7}$$

where $X \in \mathbb{C}^{m\times r}$ and $Y \in \mathbb{C}^{n\times r}$.

Facts:

1. Fast algorithms often exploit the fact that a matrix has low displacement rank with respect to a particular displacement operator. However, as seen in Eq. (62.5) and Eq. (62.4), the choice of displacement equation used to define a specific class of structured matrices can vary. Different displacement equations yield different algorithms. What is important algorithmically is that the displacement rank should not vary greatly with the particular choice of displacement operator.
2. [Dav79] A circulant matrix $S \in \mathbb{C}^{n\times n}$ commutes with the cyclic shift matrix and therefore satisfies the displacement equation
$$Z_{1,n}^* S - S Z_{1,n}^* = 0.$$
If $\mathbf{s}_j^*$ denotes row j of S, then this displacement equation is equivalent to $\mathbf{s}_{j+1}^* = \mathbf{s}_j^* Z_{1,n}^*$ so that the rows of a circulant are generated by repeated cyclic shifting of the first row.
3. A Toeplitz matrix $T \in \mathbb{C}^{m\times n}$ has displacement rank at most two with respect to the displacement equations
$$T - Z_m T Z_n^* = \begin{bmatrix} t_0 & 0 \\ t_{-1} & t_{-1} \\ \vdots & \vdots \\ t_{-m+1} & t_{-m+1} \end{bmatrix} \begin{bmatrix} 1 & t_1/t_0 & \cdots & t_{n-1}/t_0 \\ 0 & -t_1/t_0 & \cdots & -t_{n-1}/t_0 \end{bmatrix} \qquad (62.8)$$
and
$$Z_{1,m}^* T - T Z_{\delta,n}^* = \begin{bmatrix} t_{-1} - \delta t_{n-1} & 0 \\ \vdots & \vdots \\ t_{-m+1} - \delta t_{-m+n+1} & 0 \\ t_0 - \delta t_{-m+n} & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & t_1 - t_{-m+1} & \cdots & t_{n-1} - t_{-m+n-1} \end{bmatrix}. \qquad (62.9)$$
4. A Hermitian Toeplitz matrix $T \in \mathbb{C}^{n\times n}$ with $t_0 > 0$ satisfies
$$T - Z_n T Z_n^* = X \Sigma_{1,1} X^*,$$
where
$$X^* = \begin{bmatrix} \sqrt{t_0} & t_1/\sqrt{t_0} & \cdots & t_{n-1}/\sqrt{t_0} \\ 0 & t_1/\sqrt{t_0} & \cdots & t_{n-1}/\sqrt{t_0} \end{bmatrix}. \qquad (62.10)$$
5. A Hankel matrix $H \in \mathbb{C}^{m\times n}$ has displacement rank two with respect to the displacement equation $\Delta(H) = Z_m H - H Z_n^{\mathrm{T}} = XY^*$.
6. A Vandermonde matrix $V \in \mathbb{C}^{m\times n}$ with nodes $\alpha_1, \ldots, \alpha_m$ has displacement rank one with respect to the displacement equation
$$D_{\mathbf{v}} V - V Z_{\delta,n} = \begin{bmatrix} \alpha_1^n - \delta \\ \vdots \\ \alpha_m^n - \delta \end{bmatrix} \begin{bmatrix} 0 & \cdots & 0 & 1 \end{bmatrix}, \qquad (62.11)$$
where $\mathbf{v} = [\alpha_j]_{j=1}^m \in \mathbb{C}^m$.

7. A Cauchy matrix $C \in \mathbb{C}^{m\times n}$ has displacement rank 1 with respect to

$$D_{\mathbf{v}}C - CD_{\mathbf{w}} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix},$$

where $\mathbf{v} = [\alpha_j]_{j=1}^m \in \mathbb{C}^m$ and $\mathbf{w} = [\beta_j]_{j=1}^n \in \mathbb{C}^n$.

8. [SK95] A displacement operator $\Delta\colon \mathbb{C}^{m\times n} \to \mathbb{C}^{m\times n}$ is a linear transformation. If Δ is invertible, then it is possible to fully reconstruct A from $\Delta(A)$. A general displacement operator with lower triangular $E = [e_{jk}]_{j,k=1}^m$, $F = [f_{jk}]_{j,k=1}^n$, $G = [g_{jk}]_{j,k=1}^m$, and $H = [h_{jk}]_{j,k=1}^n$ is invertible if and only if

$$e_{jj}\overline{f}_{kk} - g_{jj}\overline{h}_{kk} \neq 0$$

for all choices of j and k. This fact can be specialized. A Sylvester type displacement operator is invertible if and only if $e_{jj} - \overline{h}_{kk}$ is always nonzero. A Stein type displacement operator is invertible if and only if $1 - g_{jj}\overline{h}_{kk}$ is always nonzero. Related conditions can be stated in terms of the generalized eigenvalues of the matrix pencils $E - \lambda G$ and $F - \lambda H$, in which case the assumption of triangularity of E, F, G, and H can be dropped. The triangular case covers all displacement operators associated with the algorithms considered in this chapter.

9. Several displacement equations allow simple reconstruction of A from $\Delta(A) = XY^*$. If $G^k = 0$ or $H^k = 0$ and if A satisfies the Stein type displacement equation (62.1), then

$$A = \sum_{j=0}^{k-1} (G)^j XY^*(H^*)^j.$$

If A satisfies an invertible Sylvester type displacement equation with $E = D_{\mathbf{e}}$, then row j of A, denoted by $\mathbf{a}_j^*$, can be obtained by solving

$$\mathbf{a}_j^*(e_j I - H^*) = \mathbf{x}_j^* Y^*,$$

where $\mathbf{x}_j^*$ is row j of X. If A satisfies an invertible Sylvester type displacement equation with $E = D_{\mathbf{e}}$ and $H = D_{\mathbf{h}}$, then individual elements of A can be computed from the formula

$$a_{jk} = \frac{\mathbf{x}_j^*\mathbf{y}_k}{e_j - \overline{h}_k},$$

where $\mathbf{y}_k^*$ is row k of Y.

Examples:

1. Given the multiset of nodes $\{\alpha_1, \alpha_1, \alpha_1, \alpha_2, \alpha_2\}$, a 5×5 confluent Vandermonde matrix formed from these nodes is

$$V = \begin{bmatrix} 1 & \alpha_1 & \alpha_1^2 & \alpha_1^3 & \alpha_1^4 \\ 0 & 1 & 2\alpha_1 & 3\alpha_1^2 & 4\alpha_1^3 \\ 0 & 0 & 2 & 6\alpha_1 & 12\alpha_1^2 \\ 1 & \alpha_2 & \alpha_2^2 & \alpha_2^3 & \alpha_2^4 \\ 0 & 1 & 2\alpha_2 & 3\alpha_2^2 & 4\alpha_2^3 \end{bmatrix}.$$

2. The Toeplitz matrix

$$T = \begin{bmatrix} 4 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 4 \end{bmatrix}$$

has displacement rank two with respect to the Stein type displacement equation

$$T - Z_3 T Z_3^* = \begin{bmatrix} 4 & 2 & 1 \\ 2 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 1 & 1 \\ 1/2 & 1/2 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 1/2 \\ 0 & 1 & 1/2 \end{bmatrix}.$$

This example illustrates Eq. (62.4).

62.2 Transformations of Structured Matrices

A structured matrix satisfying a displacement equation in which E, F, G, and H are all either diagonal matrices or shifts can be transformed to a Cauchy-like matrix using the discrete Fourier transform [Hei94, GKO95]. Such transformations are of great numerical significance. Most matrix structures are not preserved by the pivoting schemes typically used to stabilize Gaussian elimination. Cauchy-like structure is the exception; permuting the rows or columns of a Cauchy-like matrix results in another Cauchy-like matrix. Consequently, various pivoting schemes can be incorporated into fast solvers for Cauchy-like systems, making such solvers numerically robust even on indefinite and non-Hermitian systems. By first transforming a structured matrix to a Cauchy-like matrix, numerically robust solvers for Cauchy-like systems can be applied to Toeplitz, Hankel, Vandermonde, and other matrices with related displacement structures.

Definitions:

Let the $n \times n$ matrix $F_n = [f_{jk}] = \frac{1}{\sqrt{n}}[\omega_n^{(j-1)(k-1)}]$, where $\omega_n = e^{i2\pi/n}$, denote the unitary $n \times n$ discrete Fourier transform matrix.

Facts:

Let δ denote a positive real number and $\delta^{1/n}$ its positive nth root.

1. [Dav79]An $n \times n$ circulant matrix S with first row equal to

$$\mathbf{s}^T = \begin{bmatrix} s_0 & s_1 & \cdots & s_{n-1} \end{bmatrix}$$

has eigenvalue decomposition

$$S = F_n D_{\mathbf{v}} F_n^* \tag{62.12}$$

where $\mathbf{v} = \sqrt{n} F_n \mathbf{s}$.

2. [Hei94, HB97] The shift matrices $Z_{1,n}$, $Z_{-1,n}$, and $Z_{\delta,n}$ satisfy

$$F_n^* Z_{1,n}^* F_n = D_{\mathbf{v}_n(\omega_n)},$$

$$F_n^* D_{\mathbf{v}_n(\omega_{2n})}^* Z_{-1,n}^* D_{\mathbf{v}_n(\omega_{2n})} F_n = \omega_{2n} D_{\mathbf{v}_n(\omega_n)},$$

and

$$F_n^* D_{\mathbf{v}_n(\delta^{1/n})}^{-1} Z_{\delta,n}^* D_{\mathbf{v}_n(\delta^{1/n})} F_n = \delta^{1/n} D_{\mathbf{v}_n(\omega_n)}.$$

3. [Hei94, HB97] If $T \in \mathbb{C}^{m\times n}$ is a Toeplitz-like matrix satisfying
$$Z_{1,m}^* T - T Z_{-1,n}^* = XY^*,$$
then $C = F_m^* T D_{\mathbf{v}_n(\omega_{2n})} F_n$ is a Cauchy-like matrix satisfying
$$D_{\mathbf{v}_m(\omega_m)} C - C(\omega_{2n} D_{\mathbf{v}_n(\omega_n)}) = F_m^* XY^* D_{\mathbf{v}_n(\omega_{2n})} F_n. \qquad (62.13)$$
If $m = n$, then the sets of diagonal elements of $D_{\mathbf{v}_m(\omega_m)}$ and $\omega_{2n} D_{\mathbf{v}_n(\omega_n)}$ are disjoint and the matrix C satisfying Eq. (62.13) is uniquely determined.
4. [Gu98b] If $T \in \mathbb{C}^{m\times n}$ is a Toeplitz-like matrix satisfying
$$Z_{1,m}^* T - T Z_{\delta,n}^* = XY^*,$$
then $C = F_m^* T D_{\mathbf{v}_n(\delta^{1/n})} F_n$ is a Cauchy-like matrix satisfying
$$D_{\mathbf{v}_m(\omega_m)} C - C(\delta^{1/n} D_{\mathbf{v}_n(\omega_n)}) = F_m^* XY^* D_{\mathbf{v}_n(\delta^{1/n})} F_n. \qquad (62.14)$$
If $\delta > 1$, then the sets of diagonal elements of $D_{\mathbf{v}_m(\omega_m)}$ and $\delta^{1/n} D_{\mathbf{v}_n(\omega_n)}$ are disjoint and the matrix C satisfying Eq. (62.14) is uniquely determined.
5. If $V \in \mathbb{C}^{m\times n}$ is a Vandermonde-like matrix satisfying Eq. (62.6) with $\delta > 0$, then $C = V D_{v_n(\delta^{1/n})}^{-1} F_n$ is a Cauchy-like matrix satisfying
$$D_{\mathbf{v}} C - C(\delta^{1/n} D_{\mathbf{v}_n(\omega_n)}^*) = XY^* D_{\mathbf{v}_n(\delta^{1/n})}^{-1} F_n. \qquad (62.15)$$
If
$$\delta^{1/n} < \min_i |\alpha_i| \qquad \text{or} \qquad \delta^{1/n} > \max_i, |\alpha_i|$$
then the sets of diagonal elements of $\delta^{1/n} D_{\mathbf{v}_n(\omega_m)}$ and $D_{\mathbf{v}}$ are disjoint and the matrix C satisfying Eq. (62.15) is uniquely determined.
6. [HB97] Transformations of other structured matrices to Cauchy-like matrices can also be accomplished using real trigonometric transformations. Such transformations can be used to avoid the use of complex arithmetic.

Examples:

1. The Toeplitz matrix
$$T = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$
satisfies the displacement equation
$$Z_{1,3}^* T - T Z_{-1,3}^* = \mathbf{x}\mathbf{y}^* = \begin{bmatrix} 2 \\ 2 \\ 2 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$
and can be transformed to the Cauchy-like matrix
$$C = F_3^* T D_{\mathbf{v}_3(\omega_6)} F_3 = \begin{bmatrix} 1+\omega_6+\omega_6^2 & 1 & 1-\omega_6-\omega_6^2 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$
which satisfies the displacement equation
$$D_{\mathbf{v}_3(\omega_3)} C - \omega_6 C D_{\mathbf{v}_3(\omega_3)} = \begin{bmatrix} 2\sqrt{3} \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \end{bmatrix} = (F_3^* \mathbf{x})(\mathbf{y}^* D_{\mathbf{v}_3(\omega_6)} F_3).$$
This example illustrates Fact 3.

62.3 Generalized Schur Algorithms

It is a remarkable fact that under very general conditions the displacement structure of a matrix is inherited by its Schur complement. In particular, if E, F, G, and H are lower triangular, and if $A = [A_{jk}]$ with $a_{11} \neq 0$ has displacement rank r with respect to Eq. (62.3), then the Schur complement of a_{11} in A has displacement rank r with respect to a related displacement equation. The generators of the Schur complement can be computed from those of A using simple recurrences. Gaussian elimination can be interpreted as recursive Schur complementation so that an LU factorization of A, when it exists, can be computed using operations acting solely on the generators of A. This describes in outline the common features of the large class of fast generalized Schur algorithms for structured matrix factorization.

Instead of describing a Schur algorithm for the most general type of displacement structure, we focus on two specific equations that allow for particularly simple generator updates. The first equation is the Sylvester type displacement equation in which E and H are lower triangular. The second is the Stein type displacement equation in which G and H are strictly lower triangular.

Definitions:

Generators X and Y of a matrix satisfying a displacement equation are in **proper form** if

$$X = \begin{bmatrix} x_{11} & 0 \\ \mathbf{x}_{21} & X_{22} \end{bmatrix}, \qquad \text{and} \qquad Y = \begin{bmatrix} y_{11} & 0 \\ \mathbf{y}_{21} & Y_{22} \end{bmatrix}. \tag{62.16}$$

Facts:

Partition the $m \times n$ matrix A as

$$A = \begin{bmatrix} a_{11} & \mathbf{a}_{12}^* \\ \mathbf{a}_{21} & A_{22} \end{bmatrix}$$

and assume $a_{11} \neq 0$. Let $A_S = A/[a_{11}] = A_{22} - \frac{1}{a_{11}}\mathbf{a}_{21}\mathbf{a}_{12}^*$ be the Schur complement of a_{11} in A. The recurrences given here for computing generators of Schur complements have appeared in many papers in either more specialized or more general forms. They can be viewed in specialized forms of the general recurrences in [SK95].

The pseudocode presentation of each of the Schur algorithm variants uses recursion with the recursive function applied to structured Schur complements of decreasing size. No explicit termination criterion is given. The functions being defined compute triangular factors or generators of a matrix inverse. In actual implementation, the recursion would terminate when the function is called on a Schur complement that is small enough that a conventional factorization or inversion algorithm can be applied.

1. If A satisfies the Stein type displacement equation (62.1) with strictly lower triangular G and H partitioned as

$$G = \begin{bmatrix} 0 & 0 \\ \mathbf{g}_{21} & G_{22} \end{bmatrix}, \qquad \text{and} \qquad H = \begin{bmatrix} 0 & 0 \\ \mathbf{h}_{21} & H_{22} \end{bmatrix},$$

and generators X and Y in proper form (62.16), then A_S satisfies the displacement equation

$$A_S - G_{22}A_S H_{22}^* = X_S Y_S^*$$

where

$$X_S = \begin{bmatrix} G_{22}\mathbf{x}_{21} + \mathbf{g}_{21}x_{11} & X_{22} \end{bmatrix},$$
$$Y_S = \begin{bmatrix} H_{22}\mathbf{y}_{21} + \mathbf{h}_{21}y_{11} & Y_{22} \end{bmatrix}. \tag{62.17}$$

The first column and row of A can be computed from the generators X and Y by using

$$\begin{bmatrix} a_{11} & \mathbf{a}_{12}^* \end{bmatrix} = x_{11} \begin{bmatrix} \overline{y}_{11} & \mathbf{y}_{21}^* \end{bmatrix}, \qquad \text{and} \qquad a_{21} = \mathbf{x}_{21}\overline{y}_{11}. \tag{62.18}$$

Given generators X and Y for A that are not in proper form, when $a_{11} \neq 0$ it is possible to compute an invertible square matrix V such that XV and YV^{-*} are proper form generators for A. The construction of such a transformation when A is positive definite and $Y = X\Sigma_{p,q}$ is described in Section 62.4. The transformation of the generators to proper form can be accomplished using $O((m+n)r)$ operations. If the complexity of matrix-vector multiplication with G and H is $O(m+n)$, then the generators of A_S can be computed using $O((m+n)r)$ operations.

2. If A satisfies a Sylvester type displacement equation (62.2) with lower triangular E and H partitioned as

$$E = \begin{bmatrix} e_{11} & 0 \\ \mathbf{e}_{21} & E_{22} \end{bmatrix} \qquad \text{and} \qquad H = \begin{bmatrix} h_{11} & 0 \\ \mathbf{h}_{21} & H_{22} \end{bmatrix},$$

and generators X and Y partitioned as

$$\begin{bmatrix} \mathbf{x}_1^* \\ X_2 \end{bmatrix} \qquad \text{and} \qquad \begin{bmatrix} \mathbf{y}_1^* \\ Y_2 \end{bmatrix},$$

then A_S satisfies the displacement equation

$$E_{22}A_S - A_S H_{22}^* = X_S Y_S^*$$

where

$$X_S = X_2 - \tfrac{1}{a_{11}}\mathbf{a}_{21}\mathbf{x}_1^*,$$
$$Y_S = Y_2 - \tfrac{1}{\overline{a}_{11}}\mathbf{a}_{12}\mathbf{y}_1^*. \tag{62.19}$$

The first column and row of A can be obtained by solving

$$(E - \overline{h}_{11}I)\begin{bmatrix} a_{11} \\ \mathbf{a}_{21} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1^* \\ X_2 \end{bmatrix} \mathbf{y}_1,$$
$$\begin{bmatrix} a_{11} & \mathbf{a}_{12}^* \end{bmatrix}(e_{11}I - H^*) = \mathbf{x}_1^* \begin{bmatrix} \mathbf{y}_1 & Y_2^* \end{bmatrix}. \tag{62.20}$$

If E and H are banded with bandwidth that does not depend on n, then the systems can be solved using $O(m+n)$ operations and the generators of A_S can be computed using $O((m+n)r)$ operations.

3. If all leading principal minors of A are nonzero and if A satisfies either a Stein type displacement equation with strictly lower triangular G and H or a Sylvester displacement equation with lower triangular E and H, then repeated Schur complementation using the generator recurrences can be used to compute an LU factorization. The result is Algorithm 62.1. If $m > n$ and if the displacement equations are such that generator updates require $O((m+n)r)$ operations, then the complexity of LU factorization is $O(rmn)$.

Algorithm 62.1 (Generalized Schur Algorithm)

Let A be an $m \times n$ matrix with nonzero leading principal minors and satisfying Eq. (62.1) (or Eq. (62.2)). The function $(L, U) = \text{GSchur}(X, Y, E, F, G, H)$ computes an LU factorization of A. For the Stein equation it is assumed that $E = I$ and $F = I$. For the Sylvester equation it is assumed that $F = I$ and $G = I$.

```
function $(L, U) = \text{GSchur}(X, Y, E, F, G, H)$
    If        the displacement is of Stein type
      Then    Transform $G$ and $H$ to proper form.
    Let       $\begin{bmatrix} u_{11} & \mathbf{u}_{12}^* \end{bmatrix} = \begin{bmatrix} a_{11} & \mathbf{a}_{12}^* \end{bmatrix}$ using Eq. (62.18) (or Eq. (62.20)).
              $\mathbf{l}_{21} = \frac{1}{a_{11}}\mathbf{a}_{21}$ using Eq. (62.18) (or Eq. (62.20)).
    Compute   $X_S$ and $Y_S$ using Eq. (62.17) (or Eq. (62.19)).
    Let       $L = \begin{bmatrix} 1 & 0 \\ \mathbf{l}_{21} & L_{22} \end{bmatrix}, \quad U = \begin{bmatrix} u_{11} & \mathbf{u}_{12}^* \\ 0 & U_{22} \end{bmatrix}$
                  where $(L_{22}, U_{22}) = \text{GSchur}(X_S, Y_S, E_{22}, F_{22}, G_{22}, H_{22})$.
```

4. [KC94] Given an $n \times n$ invertible structured matrix, the generalized Schur algorithm can be applied to the augmented matrix

$$M = \begin{bmatrix} A & I \\ I & 0 \end{bmatrix}$$

to compute a generator representation of the inverse of a structured matrix. The Schur complement of A in M is $-A^{-1}$. If A satisfies the displacement equation $E_1 A - A H_1^* = X_{11} Y_{11}^*$ and if lower triangular E_2 and H_2 can be chosen so that $E_2 - H_1^* = X_{21} Y_{21}^*$ and $E_1 - H_2^* = X_{12} Y_{12}^*$ both have low rank, then M has low displacement rank with respect to

$$(E_1 \oplus E_2)M - M(H_1 \oplus H_2)^* = \begin{bmatrix} X_{11} & X_{12} & 0 \\ 0 & 0 & X_{21} \end{bmatrix} \begin{bmatrix} Y_{11}^* & 0 \\ 0 & Y_{12}^* \\ Y_{21}^* & 0 \end{bmatrix}.$$

Using this displacement representation of M, n steps of the Sylvester form of Algorithm 62.1 can be applied to the generators of M to compute generators of $-A^{-1}$. If A satisfies the Stein type equation $A - G_1 A H_1^* = X_{11} Y_{11}^*$ and it is possible to choose strictly lower triangular G_2 and H_2 so that $I - G_2 H_1^* = X_{21} Y_{21}^*$ and $I - G_1 H_2^* = X_{12} Y_{12}^*$ both have low rank, then M has low displacement rank with respect to

$$M - (G_1 \oplus G_2)M(H_1 \oplus H_2)^* = \begin{bmatrix} X_{11} & X_{12} & 0 \\ 0 & 0 & X_{21} \end{bmatrix} \begin{bmatrix} Y_{11}^* & 0 \\ 0 & Y_{12}^* \\ Y_{21}^* & 0 \end{bmatrix},$$

so that the Stein form of Algorithm 62.1 can be used to compute generators of $-A^{-1}$. This approach is applicable to Toeplitz matrices, in which case G_1 and H_1 are shift matrices and G_2 and H_2 can also be chosen to be shift matrices. Other augmented matrices can be used to achieve other ends, including QR factorization and the solution of a linear system.

62.4 Schur Algorithms for Positive Definite Matrices

The Stein variant of the generalized Schur algorithm can be adapted to compute the Cholesky factorization of a Hermitian positive definite structured matrix. In the case of a positive definite Toeplitz matrix, the algorithm has been widely used; see Section 62.6. The approach can be applied to any $n \times n$ positive definite matrix A satisfying a displacement equation of the form $A - GAG^* = X\Sigma_{p,q}X^*$ where G is strictly lower triangular. The algorithm is simply the Stein variant of Algorithm 62.1 with a transformation to proper form accomplished using $\Sigma_{p,q}$-unitary transformations.

Definitions:

For a given signature matrix $\Sigma_{p,q}$, a matrix S satisfying $S^*\Sigma_{p,q}S = \Sigma_{p,q}$ is **Σ-unitary** with respect to the signature $\Sigma_{p,q}$.

A matrix

$$S = \frac{1}{\sqrt{1-|\rho|^2}}\begin{bmatrix} 1 & \bar{\rho} \\ \rho & 1 \end{bmatrix},$$

where $|\rho| < 1$, is a **hyperbolic rotation**.

Facts:

1. If A is symmetric and if the displacement $A - GAG^*$ has p positive and q negative eigenvalues, then there is always a factorization of the displacement $A - GAG^* = X\Sigma_{p,q}X^*$. Such generators can be obtained using a variety of standard matrix decompositions applied to the displacement, including truncated LDL^* factorization. However, in considering such a factorization, it is important to note that Bunch-Parlett pivoting [GV96] is more reliable than Bunch-Kaufman pivoting in revealing rank deficiency in the displacement [SV97]. In most cases, generators are obtained using a more direct formula, or even an explicit formula in terms of the matrix elements as in Eq. (62.10).
2. A hyperbolic rotation can be computed and applied to introduce zeros into a column or row vector. If $|x| > |y|$ and $\rho = -y/x$, then

$$\frac{1}{\sqrt{1-|\rho|^2}}\begin{bmatrix} 1 & \bar{\rho} \\ \rho & 1 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \frac{x}{|x|}\sqrt{|x|^2-|y|^2} \\ 0 \end{bmatrix}$$

and

$$\frac{1}{\sqrt{1-|\rho|^2}}\begin{bmatrix} x & y \end{bmatrix}\begin{bmatrix} 1 & \rho \\ \bar{\rho} & 1 \end{bmatrix} = \begin{bmatrix} \frac{x}{|x|}\sqrt{|x|^2-|y|^2} & 0 \end{bmatrix}.$$

3. A hyperbolic rotation is Σ-unitary with respect to the signature $\Sigma_{1,1}$. A $(p+q)\times(p+q)$ hyperbolic rotation

$$S = I_{j-1} \oplus \begin{bmatrix} \frac{1}{\sqrt{1-|\rho|^2}} & 0 & \frac{\bar{\rho}}{\sqrt{1-|\rho|^2}} \\ 0 & I_{k-j-1} & 0 \\ \frac{\rho}{\sqrt{1-|\rho|^2}} & 0 & \frac{1}{\sqrt{1-|\rho|^2}} \end{bmatrix} \oplus I_{p+q-k}$$

acting in rows (or columns) j and k where $j \le p$ and $k > p$ is Σ-unitary with respect to $\Sigma_{p,q}$.
4. A hyperbolic rotation has triangular factorizations

$$\frac{1}{\sqrt{1-|\rho|^2}}\begin{bmatrix} 1 & \bar{\rho} \\ \rho & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \rho & \sqrt{1-|\rho|^2} \end{bmatrix}\begin{bmatrix} \frac{1}{\sqrt{1-|\rho|^2}} & \frac{\bar{\rho}}{\sqrt{1-|\rho|^2}} \\ 0 & 1 \end{bmatrix}$$

and

$$\frac{1}{\sqrt{1-|\rho|^2}}\begin{bmatrix} 1 & \rho \\ \bar{\rho} & 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{1-|\rho|^2}} & 0 \\ \frac{\bar{\rho}}{\sqrt{1-|\rho|^2}} & 1 \end{bmatrix}\begin{bmatrix} 1 & \rho \\ 0 & \sqrt{1-|\rho|^2} \end{bmatrix}.$$

For numerical stability, it is necessary to multiply by hyperbolic rotations in factored form, that is by multiplying by the triangular factors in sequence [BBV87]. For a hyperbolic rotation S, the first formula is stable for products of the form SX and the second is stable for products of the form XS.

5. For a symmetric positive-definite matrix A satisfying $A - GAG^* = X\Sigma_{p,q}X^*$, the Stein type generalized Schur can be arranged to compute a Cholesky factorization $A = C^*C$. The result is Algorithm 62.2. It can be shown that the matrix A is positive definite if and only if $|\hat{x}_{11}| > |\hat{x}_{13}|$ in each recursive application of Algorithm 62.2.

Algorithm 62.2 (Positive Definite Generalized Schur Algorithm)

Given an $n \times n$ positive definite matrix A satisfying $A - GAG^* = X\Sigma_{p,q}X^*$ for strictly lower triangular G, the function PDGSchur(X, G) computes the Cholesky factorization $A = C^*C$.

```
function $C = \text{PDGSchur}(X, G)$
    Let $X \begin{bmatrix} U & 0 \\ 0 & V \end{bmatrix} = \left[\begin{array}{cc|cc} x_{11} & 0 & x_{13} & 0 \\ \mathbf{x}_{21} & X_{22} & \mathbf{x}_{23} & X_{24} \end{array}\right]$
        where $U$ and $V$ are $p \times p$ and $q \times q$ Householder transformations.
    $X \begin{bmatrix} U & 0 \\ 0 & V \end{bmatrix} S = \left[\begin{array}{cc|cc} \hat{x}_{11} & 0 & 0 & 0 \\ \hat{\mathbf{x}}_{21} & X_{22} & \hat{\mathbf{x}}_{23} & X_{24} \end{array}\right]$
        where $S$ is a hyperbolic rotation acting in columns 1 and $p+1$
        with $\rho = -\overline{x}_{13}/\overline{x}_{11}$.
    $X_S = \left[\begin{array}{cc|cc} \mathbf{e}_{21}\hat{x}_{11} + E_{22}\hat{\mathbf{x}}_{21} & X_{22} & \hat{\mathbf{x}}_{23} & X_{24} \end{array}\right]$
    $C = \begin{bmatrix} \hat{x}_{11} & \hat{\mathbf{x}}_{21}^* \\ 0 & C_{22} \end{bmatrix}$, where $C_{22} = \text{PDGSchur}(X_S, G_{22})$.
```

62.5 Fast Algorithms for Cauchy-like Systems

Row and column permutations destroy most types of displacement structure. Consequently it is not generally possible to introduce pivoting for stability into most fast algorithms. Cauchy-like matrices, however, are an important exception. This advantage often motivates a preliminary transformation of matrices with other structures to Cauchy-like form prior to applying the generalized Schur algorithm.

Facts:

1. If P and Q are permutations and C is a Cauchy-like matrix satisfying Eq. (62.7), then the matrix PCQ satisfies
$$\hat{D}_{\mathbf{v}}PCQ - PCQ\hat{D}_{\mathbf{w}} = PXY^*Q,$$
where $\hat{D}_{\mathbf{v}} = PD_{\mathbf{v}}P^*$ and $\hat{D}_{\mathbf{w}} = Q^*D_{\mathbf{w}}Q$. Thus, PCQ is also a Cauchy-like matrix and pivoting schemes can be incorporated into fast solvers for Cauchy-like systems. The paper [Hei94] introduced the idea of transforming other structured matrices to Cauchy-like form in order to exploit pivoting and thereby improve the stability of fast algorithms, particularly algorithms applied to indefinite or nonsymmetric matrices. The paper presented a number of $O(n^2)$ and $O(n \log^3(n))$ methods.
2. The use of the Sylvester variant of Algorithm 62.1, in combination with partial pivoting, was investigated in [GKO95], where experiments were presented to illustrate

the numerical reliability of the method. However, the algorithm is not stable in all cases. A source of numerical instability was described and analyzed in [SB95].

3. [Gu98a] The basic approach of applying Algorithm 62.1 with pivoting can be made provably stable by using a stronger pivoting method, somewhat analogous to a structured form of complete pivoting, and by introducing a refactorization of the generators that is intended to limit growth of generator elements. The same approach can be extended to solve structured least squares problems [Gu98b].

62.6 Fast Algorithms for Toeplitz-like Systems

A broad range of direct fast algorithms have been proposed over many years for the solution of Toeplitz systems of equations. Iterative methods that exploit fast matrix-vector multiplication and effective, fast preconditioners have also been widely used.

Definitions:

The **Strang type preconditioner** for a $2m \times 2m$ Hermitian Toeplitz matrix $T = [t_{j-k}]$ is the circulant matrix S with first row given by

$$\mathbf{s}_0^* = \begin{bmatrix} t_0 & t_{-1} & \cdots & t_{-m} & t_{-m+1} & \cdots & t_{-1} \end{bmatrix}.$$

The **Chan type preconditioner** for an $n \times n$ Toeplitz matrix T is the circulant matrix S with first row given by

$$\mathbf{s}_0^* = \frac{1}{n} \begin{bmatrix} nt_0 & t_{-n+1} + (n-1)t_1 & 2t_{-n+2} + (n-2)t_2 & \cdots & (n-1)t_{-1} + t_{n-1} \end{bmatrix}.$$

A **multilevel Toeplitz matrix** is a matrix $T = [t_{\mathbf{j},\mathbf{k}}]$ with $t_{\mathbf{j},\mathbf{k}} = t_{\mathbf{j}-\mathbf{k}}$ and with rows and columns arranged by lexicographic ordering of index vectors

$$\mathbf{j} = (j_1, j_2, \ldots, j_p) \qquad \text{and} \qquad \mathbf{k} = (k_1, k_2, \ldots, k_p).$$

For $\mathbf{x} \in \mathbb{C}^n$, let $L(\mathbf{x})$ denote the $n \times n$ lower triangular Toeplitz matrix with $\mathbf{x}$ as its first column.

Facts:

1. Given an $n \times n$ positive definite Toeplitz matrix T satisfying

$$T - Z_n T Z_n^* = X \Sigma_{1,1} X^*,$$

   Algorithm 62.2 can be applied to factor T by setting $G = Z_n$. Only hyperbolic rotations, and not Householder transformations, are required. This algorithm was first described in this form in [Bar69]. It was later noted that the algorithm can be interpreted as a reformulation in matrix terms of the classical Schur algorithm for testing whether a power series is bounded in the interior of the unit circle [KS95]. The algorithm is numerically stable [BBH95].
2. Given an $n \times n$ positive definite Toeplitz T, the Levinson algorithm [Lev47] solves a linear system using $O(n^2)$ operations. The algorithm can also be interpreted as computing a triangular factorization of T^{-1}.
3. The algorithms of [Hei94, GKO95, Gu98a, Gu98b] can all be applied to solve Toeplitz systems by transforming a Toeplitz matrix to a Cauchy-like matrix using Eq. (62.13). The algorithms of [Gu98a, Gu98b] give the highest guarantee of numerical stability when the matrix is indefinite or non-Hermitian.

4. [KC94] If T is Toeplitz, then $A = T^*T$ has displacement rank at most four with respect to $\Delta(A) = A - Z_n A Z_n^*$. Block Toeplitz-like matrices can be used to compute QR factorizations of Toeplitz matrices. By applying n steps of the positive definite generalized Schur algorithm to

$$A = \begin{bmatrix} T^*T & T^* \\ T & 0 \end{bmatrix} = \begin{bmatrix} R^* \\ Q \end{bmatrix} \begin{bmatrix} R & Q^* \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ 0 & -QQ^* \end{bmatrix}$$

with $E = Z_n \oplus Z_n$, it is possible to determine the rows of R and Q^* from the triangular factors of A.
5. There are many other methods for fast QR factorization of Toeplitz and Toeplitz-like matrices. Although the presentation is based on Cholesky downdating, the algorithms of [Swe84, BBH86] are computationally similar to the generalized Schur algorithm [KC94]. Factorization algorithms of this type can be shown to provide weakly stable methods for the solution of Toeplitz least squares problems [SV97]. The methods of [Cyb80, Nag93] involve an inverse QR factorization, which computes R^{-1} instead of R. The method of [Ste07] generalizes the isometric Arnoldi algorithm of [Gra93] so that it can be used to orthogonalize the columns of a Toeplitz-like matrix.
6. Toeplitz matrices allow fast matrix-vector multiplication. Using Eq. (62.12), multiplication of a vector by an $n \times n$ circulant can be performed using $O(n\log(n))$ operations. An arbitrary $m \times n$ Toeplitz matrix T can be embedded as the leading principal submatrix of an $(m+n-1) \times (m+n-1)$ circulant as in the following:

$$S = \left[\begin{array}{c|c} T & S_{12} \\ \hline S_{21} & S_{22} \end{array}\right] = \left[\begin{array}{ccc|ccc} t_0 & t_1 & t_2 & t_{-3} & t_{-2} & t_{-1} \\ t_{-1} & t_0 & t_1 & t_2 & t_{-3} & t_{-2} \\ t_{-2} & t_{-1} & t_0 & t_1 & t_2 & t_{-3} \\ t_{-3} & t_{-2} & t_{-1} & t_0 & t_1 & t_2 \\ \hline t_2 & t_{-3} & t_{-2} & t_{-1} & t_0 & t_1 \\ t_1 & t_2 & t_{-3} & t_{-2} & t_{-1} & t_0 \end{array}\right].$$

A matrix-vector product $\mathbf{y} = T\mathbf{x}$ involving an $m \times n$ Toeplitz matrix T can be computed using $O((m+n)\log(m+n))$ operations by embedding T in a circulant and computing the product

$$\begin{bmatrix} T & S_{12} \\ S_{21} & S_{22} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ 0 \end{bmatrix} = \begin{bmatrix} \mathbf{y} \\ \mathbf{z} \end{bmatrix}$$

using the fast Fourier transform and Eq. (62.12).
7. [GF74, HR84] For an $n \times n$ nonsingular, Hermitian Toeplitz matrix T, the Gohberg-Semencul formula represents T^{-1} as the difference of products of lower triangular Toeplitz matrices

$$T^{-1} = L(\mathbf{x})L(\mathbf{x})^* - L(\mathbf{y})L(\mathbf{y})^*. \tag{62.21}$$

The formula is also implicit in the Toeplitz inversion algorithm of [Tre64] and is equivalent to

$$T^{-1} - Z_n T^{-1} Z_n^* = \begin{bmatrix} \mathbf{x} & \mathbf{y} \end{bmatrix} \Sigma_{1,1} \begin{bmatrix} \mathbf{x}^* \\ \mathbf{y}^* \end{bmatrix}$$

so that $\mathbf{x}$ and $\mathbf{y}$ are columns of a generator matrix for T^{-1}. These vectors can be obtained in a variety of ways. See Section 62.8 for an $O(n \log^2(n))$ divide-and-conquer algorithm. If $\mathbf{x}$ and $\mathbf{y}$ have been computed, Eq. (62.21) can be combined with circulant embeddings of $L(\mathbf{x})$ and $L(\mathbf{y})$ and Eq. (62.12) to multiply a vector by T^{-1} using $O(n \log(n))$ operations.

8. [Cha88] The Chan type preconditioner is the closest circulant approximation to T in the Frobenius norm.
9. [CS89, Cha89] Consider a family of Hermitian positive definite Toeplitz matrices T_n that are generated from a positive function $f(\theta) > 0$ defined on $[0, 2\pi)$. In particular, let the Fourier series of $f(\theta)$ be

$$f(\theta) = \sum_{j=-\infty}^{\infty} t_j e^{-ij\theta}$$

with $t_{-j} = \bar{t}_j$ and assume that $\{t_j\}_{j=-\infty}^{\infty} \in l_1$. Let T be the infinite Toeplitz matrix $T = [t_{j-k}]_{j,k=0}^{\infty}$ and let T_n be the $n \times n$ leading principal submatrix of T. Then T_n is positive definite. If S_n is the Strang or Chan preconditioner for T_n then $S_n^{-1}T_n$ has eigenvalues clustered around 1 in the sense that for any $\epsilon > 0$ there exist N and $M > 0$ such that for all $n > N$, at most M eigenvalues of $S_n^{-1}T_n - I$ have absolute value larger than ϵ. This is sufficient to show superlinear convergence of the preconditioned conjugate gradient algorithm [GV96] applied to a system $T_n\mathbf{x} = \mathbf{b}$. If matrix-vector multiplication by T_n and inversion of S_n are performed using Eq. (62.12) and the fast Fourier transform, the complete algorithm requires $O(n \log(n))$ operations.
10. Multilevel circulant matrices have been widely used and studied as preconditioners for multilevel Toeplitz systems. Such systems have diverse applications, including problems in image processing [HNO06]. This class of methods often works well in practice. However, the clustering results are weaker than for ordinary Toeplitz systems and it is not possible in general to guarantee superlinear convergence [CT99].

62.7 Fast Algorithms for Vandermonde Systems

Systems involving Vandermonde matrices, confluent Vandermonde matrices, and polynomial Vandermonde matrices arise in polynomial interpolation, computation of quadrature weights, and various problems in approximation. In addition to algorithms based on displacement structure, there are several more specialized algorithms.

Facts:

1. General displacement techniques can be applied to factor an $n \times n$ Vandermonde system using $O(n^2)$ operations. In particular, with $\delta = 0$, the Sylvester form of Algorithm 62.1 can be directly applied to the displacement equation (62.6) to compute an LU factorization of V. If the corresponding rows of X and diagonal of $D_{\mathbf{v}}$ are reordered correspondingly, partial pivoting can be used. In [KO97], this approach is developed more generally, in a form applicable to polynomial Vandermonde matrices in which the polynomials satisfy a three-term recurrence.
2. The transformation (62.15) can be used to transform a Vandermonde-like matrix into a Cauchy-like matrix. The algorithm of [Gu98a] or [Gu98b] can then be used to solve the corresponding Vandermonde linear system or least squares problem. This approach is backward stable.

Algorithm 62.3 (Primal Björck-Pereyra Algorithm)

Let V be an $n \times n$ Vandermonde matrix with distinct nodes $\{\alpha_1, \ldots, \alpha_n\}$. The following function computes the solution to the primal system $V\mathbf{a} = \mathbf{f}$.

```
function $\mathbf{a} = \text{primalBP}(\alpha_1, \ldots, \alpha_n, \mathbf{f})$
    $\mathbf{a} := \mathbf{f}$
    for $k = 1, \ldots, n$
        for $j = n, n-1, \ldots, k+1$
            $a_j := (a_j - a_{j-1})/(\alpha_j - \alpha_{j-k})$
    for $k = n-1, \ldots, 1$
        for $j = k, k+1, \ldots, n-1$
            $a_j := a_j - \alpha_k a_{j+1}$
```

3. The Björck-Pereyra algorithm [BP70] exploits the fact that a Vandermonde system $V\mathbf{a} = \mathbf{f}$ is equivalent to a polynomial interpolation problem with interpolation points (α_k, f_k). Such a problem can be solved using Newton interpolation and a nested product recurrence for computing the coefficients of the interpolating polynomial. Interpreted in matrix terms, the process can be represented as

$$\mathbf{a} = V^{-1}\mathbf{f} = U_0 \cdots U_{n-1} L_{n-1} \cdots L_0 \mathbf{f},$$

where the U_k are upper triangular and bidiagonal and the L_k are lower triangular and bidiagonal. Multiplication by the L_k corresponds to computing the Newton form of the interpolating polynomial. Multiplication by the U_k corresponds to multiplying out the Newton interpolating polynomial to get the coefficients of the powers of α. Given this factorization of V^{-1}, transposition can be used to derive recurrences for the dual Vandermonde system $V^*\mathbf{x} = \mathbf{b}$. Algorithm 62.3 solves a primal Vandermonde system. Algorithm 62.4 solves a dual Vandermonde system. Both algorithms require $O(n^2)$ operations to solve an $n \times n$ system. The algorithms naturally extend to the case of confluent Vandermonde systems [BE73], and also to polynomial Vandermonde systems for which the polynomials satisfy a three-term recurrence [Hig88]. The algorithm of [RO91] applies when the polynomials are Chebyshev polynomials. The algorithm of [BEG07] applies in the case of Szëgo polynomials.

Algorithm 62.4 (Dual Björck-Pereyra Algorithm)

Let V be an $n \times n$ Vandermonde matrix with distinct nodes $\{\alpha_1, \ldots, \alpha_n\}$. The following function computes the solution to the dual system $V^*\mathbf{x} = \mathbf{b}$.

```
function $\mathbf{x} = \text{dualBP}(\alpha_1, \ldots, \alpha_n, \mathbf{b})$
    $\mathbf{x} := \mathbf{b}$
    for $k = 1, \ldots, n$
        for $j = n, n-1, \ldots, k+1$
            $x_j := x_j - \overline{\alpha}_k x_{j-1}$
    for $k = n-1, \ldots, 1$
        for $j = k+1, \ldots, n$
            $x_j := x_j/(\overline{\alpha}_j - \overline{\alpha}_{j-k})$
        for $j = k, \ldots, n-1$
            $x_j := x_j - x_{j+1}$
```

4. The stability properties of the Björk-Pereyra algorithm are striking. Real Vandermonde matrices are known to be ill-conditioned. In fact, it is shown in [Tyr94] that an arbitrary real $n \times n$ Vandermonde matrix V satisfies $\kappa_2(V) \geq 2^{n-2}/\sqrt{n^{1/2}}$. Nevertheless, as originally noted in [BP70], the solution of specific Vandermonde systems using the Björck-Pereyra algorithms with real nodes is often surprisingly accurate. This behavior is in part explained by an error analysis in [Hig87], which shows that if $0 \leq \alpha_1 < \alpha_2 < \cdots < \alpha_n$ and if the vectors $\mathbf{f}$ and $\mathbf{b}$ have components with alternating signs, then the error in the computed solutions is small independent of the condition number of V.
5. There are also $O(n^2)$ algorithms for the inversion of Vandermonde and related matrices. Traub [Tra66] describes such an algorithm and gives a summary, with citations, of the repeated discovery of the method by many authors.

62.8 Superfast Algorithms

Most fast algorithms solve an $n \times n$ linear system using $O(n^2)$ operations. When implemented with aid of the fast Fourier transform, various doubling procedures can sometimes be applied to reduce the number of operations to $O(n \log^2(n))$ or $O(n \log^3(n))$. The Toeplitz case is the most extensively studied, but such techniques can be applied broadly to matrices with other displacement structures.

Algorithm 62.5 (Generic Superfast Divide-and-Conquer Inversion)

The is the recursive definition of a function $A^{-1} = \text{inverse}(A)$. In all references to matrices passed to or returned by the function, it is assumed that a structured representation of the matrix is used.

```
function $A^{-1} = \text{inverse}(A)$
    Let $n$ = number of rows of $A$
        $\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = A$, where $A_{11}$ is $n/2 \times n/2$.
        $A_{11}^{-1} = \text{inverse}(A_{11})$.
        $A_S = A_{22} - A_{21}A_{11}^{-1}A_{12}$
        $A_S^{-1} = \text{inverse}(A_S)$
        $A^{-1}$ be as in Eq. (62.22) with products computed using fast multiplication.
```

Facts:

For a given degree k polynomial $p(z)$, let $\overline{p}(z)$ denote the polynomial obtained by conjugating the coefficients of $p(z)$ and let $\tilde{p}(z) = z^k \overline{p}(1/z)$.

1. Given an $n \times n$ nonsingular matrix A with displacement structure and with nonzero leading principal minor A_{11}, the block inversion

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}^{-1} = \begin{bmatrix} A_{11}^{-1} + A_{11}^{-1}A_{12}A_S^{-1}A_{21}A_{11}^{-1} & -A_{11}^{-1}A_{12}A_S^{-1} \\ -A_S^{-1}A_{21}A_{11}^{-1} & A_S^{-1} \end{bmatrix}, \tag{62.22}$$

where $A_S = A_{22} - A_{21}A_{11}^{-1}A_{12}$, can be arranged to exploit the fact that A_{21}, A_{12}, A_{11}^{-1}, and A_S^{-1} all inherit displacement structure from A. The products required to form the blocks of the inverse can then be formed using a fast multiplication algorithm, typically using $O(n\log(n))$ operations. The details vary with the structure of the matrix A, but many algorithms follow the outline of Algorithm 62.5. The recursive formulation assumes that all leading principal submatrices of A are nonsingular. This approach was applied to Toeplitz matrices in [BA80] and to Cauchy-like matrices in [Hei94]. A general formulation is given in [Pan01]. If the multiplication steps use $O(n\log(n))$ operations, then the structured inversion algorithm requires $O(n\log^2(n))$ operations. However, the constants hidden by this complexity estimate are typically large.

2. [Hoo87, AG86, AG88] Consider an $n\times n$ positive definite Toeplitz matrix T satisfying

$$T - Z_nTZ_n^* = \begin{bmatrix} u_0 & v_0 \\ \vdots & \vdots \\ u_{n-1} & v_{n-1} \end{bmatrix} \Sigma_{1,1} \begin{bmatrix} \overline{u}_0 & \cdots & \overline{u}_{n-1} \\ \overline{v}_0 & \cdots & \overline{v}_{n-1} \end{bmatrix}.$$

The generators of T can be represented by polynomials, instead of a matrix, by defining

$$u_0(z) = u_0 + u_1z + \cdots + u_{n-1}z^{n-1} \qquad \text{and} \qquad v_0(z) = v_0 + v_1z + \cdots + v_{n-1}z^{n-1}.$$

The Schur algorithm, applied for k steps to the generator polynomials, takes the form

$$\begin{bmatrix} u_k(z) & v_k(z) \end{bmatrix} = \begin{bmatrix} u_0(z) & v_0(z) \end{bmatrix} \begin{bmatrix} a_k^{(0)}(z) & b_k^{(0)}(z) \\ \tilde{b}_k^{(0)}(z) & \tilde{a}_k^{(0)}(z) \end{bmatrix},$$

where $u_k(z)$ and $v_k(z)$ are the generator polynomials for the zero bordered Schur complement of the leading $k\times k$ principal submatrix of T and

$$\begin{bmatrix} a_k^{(l)}(z) & b_k^{(l)}(z) \\ \tilde{b}_k^{(l)}(z) & \tilde{a}_k^{(l)}(z) \end{bmatrix} = \prod_{j=l}^{k-1} \frac{1}{\sqrt{1-|\rho_j|^2}} \begin{bmatrix} 1 & \overline{\rho}_j \\ \rho_j & 1 \end{bmatrix} \begin{bmatrix} z & 0 \\ 0 & 1 \end{bmatrix}.$$

represents an accumulation of the generator transformations. If

$$\phi(z) = \frac{1}{t_0}(\tilde{a}_n^{(0)}(z) + b_n^{(0)}(z)) = \phi_0 + \phi_1 z + \cdots + \phi_{n-1}z^{n-1},$$

then

$$T^{-1} - Z_nT^{-1}Z_n^* = \begin{bmatrix} \phi_0 & 0 \\ \phi_1 & \phi_{n-1} \\ \vdots & \vdots \\ \phi_{n-1} & \phi_1 \end{bmatrix} \Sigma_{1,1} \begin{bmatrix} \overline{\phi}_0 & \overline{\phi}_1 & \cdots & \overline{\phi}_{n-1} \\ 0 & \overline{\phi}_{n-1} & \cdots & \overline{\phi}_1 \end{bmatrix}.$$

Thus, the polynomials give generators for T^{-1}. Multiplication of a vector by T^{-1} can then be implemented using $O(n\log(n))$ operations by embedding T^{-1} in a circulant and using the fast Fourier transform. If all polynomial multiplications are implemented using the fast Fourier transform, the polynomials $a_n^{(0)}(z)$ and $b_n^{(0)}(z)$ can be computed with $O(n\log^2(n))$ operations using the doubling procedure shown in Algorithm 62.6.

Algorithm 62.6 (Superfast Positive Definite Toeplitz Inversion)

Given a Schur complement of a Toeplitz matrix in the form of the polynomials $u_l(z)$ and $v_l(z)$ the following function performs k steps of the Schur algorithm. To perform a complete inversion the function should be called with SFSschur$(u_0(z), v_0(z), n)$.

```
function $(u_{l+k}(z), v_{l+k}(z), a_k^{(l)}(z), b_k^{(l)}(z)) = \text{SFSchur}(u_l(z), v_l(z), k)$
    Let $m = k/2$.
        $(u_{l+m}(z), v_{l+m}(z), a_m^{(l)}(z), b_m^{(l)}(z)) = \text{SFSchur}(u_l(z), v_l(z), m)$
        $\begin{bmatrix} u_{l+m}(z) & v_{l+m}(z) \end{bmatrix} = \begin{bmatrix} u_l(z) & v_l(z) \end{bmatrix} \begin{bmatrix} a_m^{(l)}(z) & b_m^{(l)}(z) \\ \tilde{b}_m^{(l)}(z) & \tilde{a}_m^{(l)}(z) \end{bmatrix}.$
        $(u_{l+k}(z), v_{l+k}(z), a_m^{(l+m)}(z), b_m^{(l+m)}(z)) = \text{SFSchur}(u_{l+m}(z), v_{l+m}(z), m)$
        $\begin{bmatrix} u_{l+k}(z) & v_{l+k}(z) \end{bmatrix} = \begin{bmatrix} u_{l+m}(z) & v_{l+m}(z) \end{bmatrix} \begin{bmatrix} a_m^{(l+m)}(z) & b_m^{(l+m)}(z) \\ \tilde{b}_m^{(l+m)}(z) & \tilde{a}_m^{(l+m)}(z) \end{bmatrix}.$
        $\begin{bmatrix} a_k^{(l)}(z) & b_k^{(l)}(z) \\ \tilde{b}_k^{(l)}(z) & \tilde{a}_k^{(l)}(z) \end{bmatrix} = \begin{bmatrix} a_m^{(l)}(z) & b_m^{(l)}(z) \\ \tilde{b}_m^{(l)}(z) & \tilde{a}_m^{(l)}(z) \end{bmatrix} \begin{bmatrix} a_m^{(l+m)}(z) & b_m^{(l+m)}(z) \\ \tilde{b}_m^{(l+m)}(z) & \tilde{a}_m^{(l+m)}(z) \end{bmatrix}$
```

3. Using a transformation of a nonsymmetric Toeplitz matrix to a Cauchy form and an associated interpolation problem, the algorithm of [VHK01] solves a nonsymmetric Toeplitz system with complexity $O(n \log^2(n))$.
4. The algorithm of [Ste03] adapts the approach of [Hoo87, AG86] so as to solve a Toeplitz system by using a divide-and-conquer form of back-substitution instead of by structured multiplication by the inverse.
5. The algorithm of [CGS07] exploits the presence of approximate low-rank blocks in a Cauchy-like matrix obtained from a transformation of a nonsymmetric Toeplitz matrix. The resulting algorithm gives an approximation to a solution to a system using $O(n \log(n))$ operations.

References

[AG86] G.S. Ammar and W.B. Gragg. The implementation and use of the generalized Schur algorithm. In C. I. Byrnes and A. Linquist, Eds., *Computational and Combinatorial Methods in Systems Theory*, pp. 265–279. North Holland, Amsterdam, 1986.

[AG88] G.S. Ammar and W.B. Gragg. Superfast solution of real positive definite Toeplitz systems. *SIAM J. Matrix Anal. Appl.*, 9: 61–76, 1988.

[Bar69] E.H. Bareiss. Numerical solution of linear equations with Toeplitz and vector Toeplitz matrices. *Numerische Mathematik*, 13: 404–424, 1969.

[BEG07] T. Bella, Y. Eidelman, I. Gohberg, I. Koltracht, and V. Olshevsky. A Björck-Pereyra-type algorithm for Szegö-Vandermonde matrices based on properties of unitary hessenberg matrices. *Lin. Alg. Appl.*, 420: 634–647, 2007.

[BA80] R.R. Bitmead and D.O. Anderson. Asymptotically fast solution of Toeplitz and related systems. *Lin. Alg. Appl.*, 34: 103–116, 1980.

[BE73] Å. Björck and T. Elfving. Algorithms for confluent Vandermonde systems. *Numerische Mathematik*, 21: 130–137, 1973.

[BP70] Å. Björck and V. Pereyra. Solution of Vandermonde systems of equations. *Math. Comp.*, 24: 893–903, 1970.

[BBH86] A. Bojanczyk, R.P. Brent, and F. de Hoog. QR factorization of Toeplitz matrices. *Numerische Mathematik*, 49: 81–94, 1986.

[BBH95] A. Bojanczyk, R. Brent, F. de Hoog, and D. Sweet. One the Stability of the Bareiss and related Toeplitz factorization algorithms. *SIAM J. Sci. Stat. Comp.*, 16: 40–58, 1995.

[BBV87] A. Bojanczyk, R.P. Brent, P. Van Dooren, and F. de Hoog. A note on downdating the Cholesky factorization. *SIAM J. Sci. Stat. Comp.*, 8: 210–221, 1987.

[CT99] S.S. Capizzano and E. Tyrtyshnikov. Any circulant-like preconditioner for multilevel matrices is not superlinear. *SIAM J. Matrix Anal. Appl.*, 21: 431–439, 1999.

[Cha89] R.H. Chan. Circulant preconditioners for hermitian Toeplitz systems. *SIAM J. Matrix Anal. Appl.*, 10: 545–550, 1989.

[CS89] R.H. Chan and G. Strang. Toeplitz equations by conjugate gradients with circulant preconditioner. *SIAM J. Sci. Stat. Comp.*, 10: 104–119, 1989.

[Cha88] T.F. Chan. An optimal circulant preconditioner for Toeplitz systems. *SIAM J. Sci. Stat. Comp.*, 9: 766–771, 1988.

[CGS07] S. Chandrasekaran, M. Gu, X. Sun, J. Xia, and J. Zhu. A superfast algorithm for Toeplitz systems of linear equations. *SIAM J. Matrix Anal. Appl.*, 29: 1247–1266, 2007.

[Cyb80] G. Cybenko. The numerical stability of the Levinson-Durbin algorithm for Toeplitz systems of equations. *SIAM J. Sci. Stat. Comp.*, 1: 303–310, 1980.

[Dav79] P.J. Davis. *Circulant Matrices.* Wiley-Interscience, New York, 1979.

[FMK79] B. Friedlander, M. Morf, T. Kailath, and L. Ljung. New inversion formulas for matrices classified in terms of their distance from Toeplitz matrix. *Lin. Alg. Appl.*, 27: 31–60, 1979.

[GF74] I. Gohberg and I. Feldman. *Convolution Equations and Projection Methods for Their Solution.* AMS, 1974.

[GKO95] I. Gohberg, T. Kailath, and V. Olshevesky. Fast Gaussian elimination with partial pivoting for matrices with displacement structure. *Math. Comp.*, 64: 1557–1576, 1995.

[GV96] G.H. Golub and C.F. Van Loan. *Matrix Computations*, 3rd ed. Johns Hopkins University Press, Baltimore, 1996.

[Gra93] W.B. Gragg. Positive definite Toeplitz matrices, the Arnoldi process for isometric operators, and Gaussian quadrature on the unit circle. *J. Comp. Appl. Math.*, 46: 183–198, 1993.

[Gu98a] M. Gu. Stable and efficient algorithms for structured systems of linear equations. *SIAM J. Matrix Anal. Appl.*, 19: 279–306, 1998.

[Gu98b] M. Gu. New fast algorithms for structured linear least squares problems. *SIAM J. Matrix Anal. Appl.*, 20: 244–269, 1998.

[HNO06] P.C. Hansen, J.G. Nagy, and D.P. O'Leary. *Deblurring Images: Matrices, Spectra, and Filtering.* SIAM, 2006.

[Hei94] G. Heinig. Inversion of generalized Cauchy matrices and other classes of structured matrices. In *Linear Algebra in Signal Processing*, pp. 95–114, 1994. The IMA Volumes in Mathematics and Its Applications, Volume 69.

[HB97] G. Heinig and A. Bojanczyk. Transformation techniques for Toeplitz and Toeplitz-plus-Hankel matrices. I. Transformations. *Lin. Alg. Appl.*, 254: 193–226, 1997.

[HR84] G. Heinig and K. Rost. *Algebraic Methods for Toeplitz-like Matrices and Operators.* Birkhäuser, Basel, 1984.

[Hig87] N.J. Higham. Error analysis of the Björck-Pereyra algorithms for solving Vandermonde systems. *Numerische Mathematik*, 50: 613–632, 1987.

[Hig88] N.J. Higham. Fast solution of Vandermonde-like systems involving orthogonal polynomials. *IMA J. Numer. Anal.*, 8: 473–486, 1988.

[Hoo87] F. de Hoog. A new algorithm for solving Toeplitz systems of equations. *Lin. Alg. Appl.*, 88/89: 123–138, 1987.

[KC94] T. Kailath and J. Chun. Generalized displacement structure for block-Toeplitz, Toeplitz-block, and Toeplitz-derived matrices. *SIAM J. Matrix Anal. Appl.*, 15: 114–128, 1994.

[KO97] T. Kailath and V. Olshevsky. Displacement-structure approach to polynomial Vandermonde and related matrices. *Lin. Alg. Appl.*, 261: 49–90, 1997.

[KS95] T. Kailath and A.H. Sayed. Displacement structure: theory and applications. *SIAM Review*, 37: 297–386, 1995.

[Lev47] N. Levinson. The Weiner RMS error criterion in filter design and prediction. *J. Math. Phys.*, 25: 261–278, 1947.

[Nag93] J.G. Nagy. Fast inverse QR factorization for Toeplitz matrices. *SIAM J. Sci. Comp.*, 14: 1174–1193, 1993.

[Pan01] V. Pan. *Structured Matrices and Polynomials: Unified Superfast Algorithms.* Birkhäuser, Boston, 2001.

[RO91] L. Reichel and G. Opfer. Chebyshev-Vandermonde systems. *Math. Comp.*, 57: 703–721, 1991.

[SK95] A.H. Sayed and T. Kailath. Fast algorithms for generalized displacement structures and lossless systems. *Lin. Alg. Appl.*, 219: 49–78, 1995.

[Ste03] M. Stewart. A superfast Toeplitz solver with improved numerical stability. *SIAM J. Matrix Anal. Appl.*, 25: 669–693, 2003.

[Ste07] M. Stewart. A generalized isometric Arnoldi algorithm. *Lin. Alg. Appl.*, 423: 183–208, 2007.

[SV97] M. Stewart and P. Van Dooren. Stability issues in the factorization of structured matrices. *SIAM J. Matrix Anal. Appl.*, 18: 104–118, 1997.

[Swe84] D.R. Sweet. Fast Toeplitz orthogonalization. *Numer. Math.*, 43: 1–21, 1984.

[SB95] D.R. Sweet and R.P. Brent. Error analysis of a partial pivoting method for structured matrices. In F. T. Luk, Ed., *Advanced Signal Processing Algorithms, Proceedings of SPIE-1995*, Volume 2563, pp. 266–280, 1995.

[Tra66] J.F. Traub. Associated polynomials and uniform methods for the solution of linear problems. *SIAM Review*, 8: 277–301, 1966.

[Tre64] W.F. Trench. An algorithm for the inversion of finite Toeplitz matrices. *J. SIAM*, 12: 515–522, 1964.

[Tyr94] E.E. Tyrtyshnikov. How bad are Hankel matrices. *Numerische Mathematik*, 67: 261–269, 1994.

[VHK01] M. Van Barel, G. Heinig, and P. Kravanja. A stabilized superfast solver for nonsymmetric Toeplitz systems. *SIAM J. Matrix Anal. Appl.*, 23: 494–510, 2001.