UNIVERSITÀ DEGLI STUDI DI PISA
DIPARTIMENTO DI INFORMATICA

DOTTORATO DI RICERCA IN INFORMATICA
Università di Pisa-Genova-Udine

Ph.D. Thesis: TD-1/94

Matrix Algebras and Discrete Transforms

Enrico Bozzo

Abstract. Matrices having displacement structure, such as Toeplitz-like and Hankel-like matrices, arise in many problems of applied mathematics. Discrete transforms, such as the Discrete Fourier Transform (DFT), are a basic tool in scientific computing. Matrix algebras can be used to study these two important topics and to clarify their relationships.

In the first part of this thesis, after a review of the best known matrix algebras, we introduce some new algebras, in particular algebras related to discrete trigonometric transforms and algebras of dimension higher than the matrix order. These algebras lead to the synthesis of new inversion formulas, based on suitable displacement operators, for Toeplitz-like and Hankel-like matrices.

In the second part we devise a family of new algorithms for computing some discrete trigonometric transforms by means of the eigenpolynomials of the matrices of certain algebras. These algorithms are related to Bruun's algorithm for DFT computation. We perform a deep analysis of Bruun's algorithm: we show that it rests on a particular factorization of the Fourier matrix, suggest for it an autosort variant and discuss its numerical stability.

March 1994

ADDR: Corso Italia 40 - 56125 Pisa - Italy - TEL:+39-50-887000 - TLX:590291 DIPISAI
E_MAIL: bozzo@di.unipi.it

Acknowledgements

I wish to thank my advisor, Dario Bini, for the ideas and the enthusiasm that he has communicated to me in these years.

The possibility of working in the group of Numerical Analysis of the University of Pisa has been a privilege. Thanks to Milvio Capovani that has introduced me in the group. For their warmth and friendliness as well as for the many pleasing and useful discussions I thank Roberto Bevilacqua, Nazzareno Bonanni, Claudia Fassino, Luca Gemignani, Mauro Leoncini, Carla Manni, Francesco Romani, Stefano Serra. A special thank goes to Paolo Zellini for his precious suggestions and his encouragement. A very special thank goes to Ornella Menchi for her everyday help on everything.

Infine la mia Emanuela ed i miei genitori, con il loro amore e la loro fiducia, mi hanno aiutato in modo decisivo a superare i momenti difficili di questi anni.

Contents

Introduction.

Various developments in numerical mathematics rest on the relationship between structural and computational properties of sets of matrices. On one hand, the clever exploitation of structure properties has led to new algorithms for solving numerical problems. By using these algorithms, a dramatic speed up has beeen achieved with respect to the methods previously emploied. On the other hand, the discovery of the computational properties of certain sets of matrices has encouraged (a) the characterization of other sets with analogous properties (b) the attempt to extend to wider sets of matrices the same properties in a suitably weakened form.

For their importance in many applications Toeplitz and Hankel matrices [44] are among the most useful and studied dense structured matrices. In the efficient exploitation of their structure a major role is played by matrix algebras, in particular matrix algebras associated with discrete transforms such as the discrete Fourier transform (DFT) and certain trigonometric transforms. Let us recall that these algebras have been used in the study of spectral and computational properties of various classes of Toeplitz matrices [14, 15, 73] and in the development of preconditioning strategies for the conjugate gradient method [18, 19, 32, 31, 49]. Moreover, matrix algebras have found application in the derivation of matrix decompositions in relation with the concept of displacement rank [37, 50, 38, 4, 5, 41, 42, 43, 33, 21] leading to the discovery of various inversion formulas for Toeplitz, Hankel, Toeplitz-plus-Hankel and other related matrices. These formulas include the Gohberg-Semencul [44] inversion formulas for Toeplitz matrices as a particular case.

The circulant matrices [29] are probably the best known example of matrix algebra. There is a tight connection between circulants and DFT that was known well before the introduction of the fast Fourier transform (FFT) [53, 68] in 1965. The FFT shed a new light on circulants, allowing a better understanding of their computational relevance and opening the way towards the applications to Toeplitz matrices already mentioned and many others, including DFT computation itself. In fact, we recall that today important algorithms for computing the DFT (e.g. Rader and Winograd algorithms [53]) use the product of a circulant matrix by a vector as a basic tool.

Besides circulant matrices, another matrix algebra is of special interest: it is known as τ algebra and has been introduced in [14]. As circulants are associated with DFT, τ algebra is associated with a particular version of the discrete Sine tranform [54, 70]. Having circulants and τ algebra as basic examples it is interesting to investigate other pairs algebra-transform with analogous characteristics.

In the first part of this thesis, we will introduce a family of algebras that we call $\tau_{\varepsilon\varphi}$ algebras. Some of these algebras are shown to be associated with certain versions of Sine and Cosine transforms [70, 47, 69]. Actually τ algebra turns out to be one of them. Moreover, we will introduce two algebras having dimension about $2n$ associated with DFT. The structure of all these algebras will be deeply investigated, having as term of comparison Toeplitz and Hankel structures.

The new algebras introduced in this thesis allow us to obtain new inversion formulas for Toeplitz, Toeplitz-plus-Hankel and Toeplitz-plus-Hankel like matrices (the concept of Toeplitz-plus-Hankel like matrix will be defined in relation with the concept of displacement rank). These formulas are interesting in various respects. In particular, one of them involves algebras of dimension higher than n, and is the first formula of this kind in the literature. As an application, it is shown how the derived formulas can be exploited in order to compute the product of the inverse of one of the cited matrices by a vector. We obtain cost bounds that match the best bounds known in the literature for Toeplitz matrices and improve the ones relative to Toeplitz-plus-Hankel matrices.

In the second part of this thesis, the discrete transforms associated with the matrix algebras previously discussed come to the foreground. Actually certain properties of matrix algebras themselves suggest an unusual approach to their computation. As a result we obtain a family of new algorithms for computing discrete trigonometric transforms. These algorithms have a cost of about $2n \log n$ real arithmetic operations for computing the Sine, Cosine or Hartley transform of a (real) data sequence, matching the costs of the best algorithms today available for these transforms.

The algoritms here proposed are deeply related with Bruun's algorithm [27]. Bruun's algorithm belongs to the family of "real factor" [53, 56, 55] algoritms for DFT computation. Actually, it is not a well known algorithm even if some recent results [72, 61] indicate a renewed interest in it. In the view of the mentioned relation we perform a deep analysis of Bruun's algorithm. Two interesting characteristics of Bruun's algorithm will emerge.

1. The algorithm can be used in order to compute the cyclic convolution of two real data sequences working in place and without the explicit use of any data permutation.

2. It is possible to obtain certain autosort variants of the algorithm similar, in some sense, to Stockham algorithms [68]. The implementation of these variants on a vector computer could have a certain interest.

The structure of the thesis is as follows. There are two parts, each divided in two chapters. Matrix algebras are the subject of the first part. Discrete transforms remain on the background but actually they guide us towards certain matrix algebras. In the first chapter after a brief discussion of the most important theoretical results on matrix algebras, we will introduce various new algebras discussing their structure with reference to Toeplitz and Hankel structure. In the second chapter we will use the new algebras in order to obtain new inversion formulas for Toeplitz, Toeplitz-plus-Hankel and Toeplitz-plus-Hankel like matrices. The second part of the thesis deals with Bruun's algorithm and with a family of algorithms for computing trigonometric transforms related to it. Specifically, in chapter 3 we introduce all these algorithms while in chapter 4 we proceed with a deep analysis of Bruun's algorithm.

The most important results obtained in chapters 1 and 2 have been collected in [24] and [23]. Furthermore in [9] some ideas presented in chapter 1 are developed, leading to the precise characterization of a wide class of algebras of Toeplitz-plus-Hankel matrices. The results obtained in chapter 3 have been collected in [13].

Part I

Chapter 1

Matrix algebras.

A matrix algebra is a linear space of matrices closed under matrix multiplication. Our main interest is in discussing the relations existing between algebraic properties (closure, commutativity), structure properties (number and spatial arrangement of the free parameters) and computational properties (binding with fast discrete transform algorithms) of these sets of matrices. Unfortunately, a complete theory able to explain these relations is not available, so that we will present and compare the most important approaches that have been used in the literature for their study.

The circulant matrices [29] are probably the best known example of matrix algebra. The interest in the circulant matrices dates back to the last century, however only after the introduction of the fast Fourier transform their computational relevance has been fully understood, and other related algebras, such as the one known as τ [14], have been discovered and studied.

These sets of matrices have an intrisic interest and various applications. In particular, they have been widely exploited in the area of the computations with Toeplitz and Hankel matrices, both in a sequential and in a parallel environment.

In this chapter, besides circulants and τ algebra, some new algebras will be introduced. In the next chapter, by using the concept of displacement rank, we will show how it is possible to represent an arbitrary matrix as sum of products of pairs of matrices belonging to these algebras.

The outline of this chapter is as follows. After some preliminaries, in section 1.3 we will discuss some basic facts about algebras of simultaneously diagonalizable matrices and algebras associated with transforms. Section 1.4 concerns algebras of matrices that are polynomials in a given matrix, and section 1.5 deals with centralizers. The three families of algebras introduced in sections 1.3, 1.4 and 1.5 are overlapping to some extent as we will see in section 1.6. In section 1.7 we will discuss the concept of cross-sum condition that underlies the structure of many matrix algebras, circulants and τ algebra included. In section 1.8 some important examples of algebras are discussed: among these, certain algebras related to τ algebra that we call $\tau_{\varepsilon\varphi}$ algebras [24], and two algebras of dimension greater than n [23].

1.1 Preliminaries.

In this section, after setting terminology and notation, we recall some results from matrix theory that will be useful in order to characterize the structure of certain matrix algebras. For a detailed discussion of these results the reader is referred to [45].

1.1.1 Basic terminology and notations.

Throughout this thesis with $\mathsf{C}^{n\times m}$ ($\mathsf{R}^{n\times m}$) we will denote the set of $n \times m$ matrices over the complex (real) field C (R). If $A \in \mathsf{C}^{n\times m}$ writing $A = (a_{i\,j})$ we mean that $a_{i\,j}$ is the $(i,\, j)$ entry of A. Sometimes we will use the alternative notation $[A]_{i\,j} = a_{i\,j}$. If $A = (a_{i\,j})$ then $A^T = (a_{j\,i})$ is the transpose of A and $A^H = (\bar{a}_{j\,i})$ is the conjugate transpose of A. If A is an $n \times m$ matrix then both A^T and A^H are $m \times n$. In the case where $m = 1$ we set $\mathsf{C}^n = \mathsf{C}^{n\times 1}$ ($\mathsf{R}^n = \mathsf{R}^{n\times 1}$). The elements of C^n (R^n) will be called vectors

and denoted with boldface lowercase letters such as $\mathbf{a}$, $\mathbf{b}$.

If $A \in \mathsf{C}^{n\times n}$ and $A = A^T$ then A is said to be symmetric (note that we do not require that the elements of A are real). If $A = A^H$ then A is said to be Hermitian. If $AA^H = A^HA$ then A is said to be normal. Given a nonsingular matrix $A \in \mathsf{C}^{n\times n}$, its inverse will be denoted with A^{-1} so that $AA^{-1} = A^{-1}A = I$ being I the identity matrix. If $A^H = A^{-1}$ then A is said to be unitary. If $A \in \mathsf{R}^{n\times n}$ and $A^T = A^{-1}$ then A is said to be orthogonal. We will use A^{-T} and A^{-H} as shorthands for $(A^{-1})^T$ and $(A^{-1})^H$ respectively.

The columns of the identity matrix are called canonical vectors and are denoted with $\mathbf{e}_k$, where $k = 1, \ldots, n$. Moreover, we will set $\mathbf{e} = \sum_{k=1}^{n} \mathbf{e}_k$ and $\tilde{\mathbf{e}} = \sum_{k=1}^{n} (-1)^{(k-1)}\mathbf{e}_k$.

The matrix

$$J = \begin{pmatrix} 0 & \cdots & 0 & 1 \\ \vdots & \iddots & \iddots & 0 \\ 0 & \iddots & \iddots & \vdots \\ 1 & 0 & \cdots & 0 \end{pmatrix}$$

will be used extensively. Note that $J^T = J$ and that $J^2 = I$. Given a vector $\mathbf{x} \in \mathsf{C}^n$ we set $J\mathbf{x} = \hat{\mathbf{x}}$.

If $A \in \mathsf{C}^{n\times n}$ and $JAJ = A^T$ then A is said to be persymmetric. If $JAJ = A$ then A is said to be centrosymmetric. If a matrix is both symmetric and persymmetric then it is centrosymmetric.

Besides J, we will use frequently the matrices

$$Z = \begin{pmatrix} 0 & 0 & \cdots & \cdots & 0 \\ 1 & \ddots & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 & 0 \end{pmatrix} \quad \text{and} \quad K = \begin{pmatrix} 1 & 0 & \cdots & \cdots & 0 \\ 0 & & & \iddots & 1 \\ \vdots & & \iddots & \iddots & 0 \\ \vdots & \iddots & \iddots & \iddots & \vdots \\ 0 & 1 & 0 & \cdots & 0 \end{pmatrix}.$$

A matrix $A \in \mathsf{C}^{n\times n}$ is Toeplitz if $A = (a_{i-j})$, and is Hankel if $A = (a_{i+j-2})$ with $i, j = 1, \ldots, n$. For example for $n = 4$ we have respectively

$$A = \begin{pmatrix} a_0 & a_{-1} & a_{-2} & a_{-3} \\ a_1 & a_0 & a_{-1} & a_{-2} \\ a_2 & a_1 & a_0 & a_{-1} \\ a_3 & a_2 & a_1 & a_0 \end{pmatrix} \quad \text{and} \quad A = \begin{pmatrix} a_0 & a_1 & a_2 & a_3 \\ a_1 & a_2 & a_3 & a_4 \\ a_2 & a_3 & a_4 & a_5 \\ a_3 & a_4 & a_5 & a_6 \end{pmatrix}.$$

Note that Toeplitz matrices are persymmetric and Hankel matrices are symmetric. Moreover, note that if A is Toeplitz (Hankel) then both JA and AJ are Hankel (Toeplitz).

1.1.2 Simultaneous reduction of commuting sets of matrices to triangular or diagonal form.

Definition 1.1 An arbitrary (finite or infinite) set of matrices $\mathcal{M} \subseteq \mathsf{C}^{n\times n}$ is a commuting set if each pair in the set commutes under row by column multiplication. A simultaneously diagonalizable (triangularizable) set is one for which there is a single nonsingular matrix S such that $S^{-1}AS$ is diagonal (upper triangular) for every $A \in \mathcal{M}$. If S can be taken to be unitary the set is said to be simultaneously unitarily diagonalizable (triangularizable).

For commuting sets the following results hold (see [45, pp. 81 and 103]).

Theorem 1.2 *If $\mathcal{M} \subseteq \mathsf{C}^{n\times n}$ is a commuting set then it is simultaneously unitarily triangularizable.*

Corollary 1.3 *Let $\mathcal{M} \subseteq \mathsf{C}^{n\times n}$ be a set of normal matrices. Then $\mathcal{M}$ is a commuting set if and only if it is a simultaneously unitarily diagonalizable set.*

More generally, for a commuting set of diagonalizable matrices we have the next result (see [45, p. 52]).

Theorem 1.4 *Let $\mathcal{M} \subseteq \mathsf{C}^{n\times n}$ be a set of diagonalizable matrices. Then $\mathcal{M}$ is a commuting set if and only if it is a simultaneously diagonalizable set.*

1.1.3 Nonderogatory matrices.

The minimal polynomial and the Jordan canonical form of a matrix are closely related. This relation yields an interesting observation (see [45, p. 135]).

Theorem 1.5 *Let $A \in \mathsf{C}^{n\times n}$. Then the following conditions are equivalent:*

(a) *every eigenvalue of A has geometric multiplicity 1;*

(b) *the minimal polinomial and the characteristic polynomial of A are equal.*

If a matrix satisfies the conditions (a) or (b) listed in the previous theorem then it is said to be *nonderogatory*.

Example 1.6 Let $A \in \mathsf{C}^{n\times n}$ be diagonalizable. Since every eigenvalue of A has the same algebraic and geometric multiplicity we have that A is nonderogatory if and only if it has n distinct eigenvalues. As a second example consider the lower Hessenberg matrix

$$X = \begin{pmatrix} * & b_1 & & O \\ \vdots & \ddots & \ddots & \\ \vdots & & \ddots & b_{n-1} \\ * & \cdots & \cdots & * \end{pmatrix},$$

where $*$ denote unimportant elements. If $b_i \neq 0$, with $i = 1, \ldots, n-1$, then X is nonderogatory. To prove this assertion suppose that λ is an eigenvalue of X. Then the matrix $X - \lambda I$ is singular. On the other hand, its rank must be at least $n-1$ since, by the hypothesis on the b_i, its last $n-1$ columns are linearly independent. Since the matrix $X - \lambda I$ is singular its rank cannot be n and thus it must be $n-1$. This implies that the dimension of its nullspace, i.e., the geometric multiplicity of λ, is 1.

Nonderogatory matrices arise in connection with the following problem. Let $p(x) = \sum_{i=0}^{m} a_i x^i$, $a_i \in \mathsf{C}$, be a given polynomial. If $A \in \mathsf{C}^{n\times n}$ we set $p(A) = \sum_{i=0}^{m} a_i A^i$, where $A^0 = I$. Clearly, for every A and $p(x)$ we have $Ap(A) = p(A)A$. Is there any other matrix commuting with A? If the matrix A is nonderogatory the answer to this question is no. Moreover, the converse is also true, leading to the following characterization of nonderogatory matrices (see [45, pp. 135–137]).

Theorem 1.7 *A matrix $A \in \mathsf{C}^{n\times n}$ is nonderogatory if and only if every matrix that commutes with A is a polynomial in A.*

1.2 Definitions and basic properties.

For an abstract presentation of the concept of algebra, and various examples the reader is referred to [67]. The following definition embodies the special case here addressed. Remember that with the usual entrywise addition and product by an element of C the set $\mathsf{C}^{n\times n}$ is endowed with the structure of *linear space* over C. In order to define the concept of algebra we need to enrich the operation set with row by column matrix multiplication.

Definition 1.8 *A set of matrices $\mathcal{A} \subseteq \mathsf{C}^{n\times n}$ is an algebra or, more specifically, a matrix algebra if and only if it is a linear subspace of $\mathsf{C}^{n\times n}$ closed under row by column matrix multiplication.*

The concepts of basis and dimension of an algebra $\mathcal{A} \subseteq \mathsf{C}^{n\times n}$ are defined as in any linear space. An algebra $\mathcal{A} \subseteq \mathsf{C}^{n\times n}$ is said to be *commutative* if, for every A, $B \in \mathcal{A}$ it turns out that $AB = BA$; it is said to be an *algebra with identity* if $I \in \mathcal{A}$.

Example 1.9 Diagonal matrices in $\mathsf{C}^{n\times n}$ constitute a commutative algebra of dimension n, while upper (lower) triangular matrices constitute a noncommutative algebra of dimension $n(n+1)/2$. Both diagonal

and triangular matrices are algebras with identity. Let n be even. If A, $O \in \mathsf{C}^{n/2\times n/2}$, being O the zero matrix, then matrices of the form

$$\begin{pmatrix} O & O \\ A & O \end{pmatrix}$$

constitute a commutative algebra of dimension $n^2/4$. This algebra does not contain the identity matrix. However, observe that if $\mathcal{A}$ is a matrix algebra and if $I \notin \mathcal{A}$ then $\mathcal{A}' = \{\alpha I + A \mid \alpha \in \mathsf{C}\,,\ A \in \mathcal{A}\}$ is an algebra with identity such that $\mathcal{A} \subset \mathcal{A}'$ and $\dim \mathcal{A}' = 1 + \dim \mathcal{A}$. We notice that $n^2/4 + 1$ is the greatest possible dimension for a commutative algebra [63]. As a last example, note that the set of unitary matrices is closed under row by column multiplication but it is not a linear space, thus it is not a matrix algebra.

The Cayley-Hamilton theorem states that every matrix in $\mathsf{C}^{n\times n}$ satisfies its own characteristic polynomial. This has a simple but important consequence.

Proposition 1.10 *If $\mathcal{A} \subseteq \mathsf{C}^{n\times n}$ is a matrix algebra and $A \in \mathcal{A}$ is a nonsingular matrix, then $A^{-1} \in \mathcal{A}$.*

Given two algebras $\mathcal{A}$, $\mathcal{B} \subseteq \mathsf{C}^{n\times n}$ an *isomorphism* betweeen $\mathcal{A}$ and $\mathcal{B}$ is a bijective application $\phi : \mathcal{A} \to \mathcal{B}$ such that, for every A, $B \in \mathcal{A}$ and every a, $b \in \mathsf{C}$

$$\begin{aligned} \phi(aA + bB) &= a\phi(A) + b\phi(B), \\ \phi(AB) &= \phi(A)\phi(B). \end{aligned}$$

Observe that if $A \in \mathcal{A}$ is a nonsingular matrix then $\phi(A)$ is nonsingular and

$$\phi(A^{-1}) = \phi(A)^{-1}.$$

If there is an isomorphism between two algebras $\mathcal{A}$ and $\mathcal{B}$, they are said to be isomorphic. Note that isomorphic algebras have the same dimension. Moreover if $\mathcal{A}$ and $\mathcal{B}$ are isomorphic then $\mathcal{A}$ is commutative if and only if $\mathcal{B}$ is so, and $I \in \mathcal{A}$ if and only if $I \in \mathcal{B}$. As an example, let $\mathcal{A} \subseteq \mathsf{C}^{n\times n}$ be an algebra and let $V \in C^{n\times n}$ be a nonsingular matrix. We set

$$D(V, \mathcal{A}) = \{VAV^{-1} \mid A \in \mathcal{A}\}. \tag{1.1}$$

Clearly $D(V, \mathcal{A})$ is an algebra isomorphic to $\mathcal{A}$.

A subset $\mathcal{S}$ of an algebra $\mathcal{A}$ will be called a *system of generators* for $\mathcal{A}$ if every element of $\mathcal{A}$ is a linear combination of elements of $\mathcal{S}$ and of their products. If $\mathcal{S}$ contains m elements then $\mathcal{A}$ is said to be m-generated. By using again Cayley-Hamilton theorem we easily prove that if $\mathcal{A} \subseteq \mathsf{C}^{n\times n}$ is one-generated then $\dim \mathcal{A} \leq n$.

Example 1.11 The algebra of the diagonal matrices is one-generated and one can take $\mathcal{S} = \{D\}$, where D is whatever diagonal matrix with distinct and nonzero diagonal entries. The algebra of upper triangular matrices cannot be one-generated, since its dimension is greater than n, however it is two-generated. It is simple to show that $\mathcal{S} = \{D\,,\ T\}$, where D is whatever diagonal matrix with distinct and nonzero diagonal entries and, for example

$$T = \begin{pmatrix} 1 & \cdots & \cdots & 1 \\ 0 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 \end{pmatrix},$$

is a system of generators for the algebra.

In general, it is not easy to determine the structure of an algebra once a set of generators is given. However in certain cases an elegant graph theoretic approach can be used. The paper [51] is a recent reference on this subject.

We close this section mentioning an interesting result discussed in [8]. Suppose that $\mathcal{S} = \{A\,,\ B\}$ is the set of generators of an algebra $\mathcal{A} \subseteq \mathsf{C}^{n\times n}$. With properly chosen A and B we could have $\mathcal{A} = \mathsf{C}^{n\times n}$. However, if A and B commute the following result holds.

Theorem 1.12 *Let $\mathcal{S} = \{A\,,\ B\}$ be a set of generators for the algebra $\mathcal{A} \subseteq \mathsf{C}^{n\times n}$. If A and B commute then $\dim \mathcal{A} \leq n$.*

1.3 Matrix algebras associated with transforms.

Let $\mathcal{D} \subseteq \mathsf{C}^{n\times n}$ be the algebra of the diagonal matrices (see example 1.9) and let $V \in \mathsf{C}^{n\times n}$ be nonsingular. We define (see (1.1))

$$D(V) = D(V, \mathcal{D}) = \{VDV^{-1} \,|\, D \in \mathcal{D}\}. \tag{1.2}$$

The set $D(V)$ is an algebra isomorphic to $\mathcal{D}$ so that it is a commutative algebra with identity and has dimension n. The family of the algebras of this kind will be indicated with the symbol $\mathbf{D}$. Formally $\mathbf{D} = \{D(V) | V \in \mathsf{C}^{n\times n}, V \text{ nonsingular}\}$.

If $A \in D(V)$ then $A = VD_AV^{-1}$, where D_A is a diagonal matrix whose diagonal entries are the eigenvalues of A. In [14] the different role played by D_A and V in the above representation is remarked in an attractive, though informal, way. The matrix V is the same for all the matrices in $D(V)$ and thus has to be related to what the matrices in $D(V)$ have in common, i.e., a certain layout or *structure*. On the other hand, the matrix D_A fully specifies a particular matrix A in $D(V)$ and one may say that D_A has to do with the *informational content* of A.

As we already pointed out, the algebras in $\mathbf{D}$ are all isomorphic to the algebra $\mathcal{D}$ of the diagonal matrices. For this reason from an algebraic viewpoint they are, in a certain sense, too simple to be of some interest. However, from a computational viewpoint the situation is completely different for at least two reasons.

First, it seems that if we impose on a matrix algebra certain algebraic and/or structure prescriptions of interest in the applications then the algebra turns out to belong to $\mathbf{D}$ or, roughly speaking, it can be described in terms of algebras belonging to $\mathbf{D}$. There is a growing literature on this subject. The interested reader can see for example [10]. As a very simple example observe that for any algebra $\mathcal{A}$ made up by Hermitian matrices, we have $\mathcal{A} = D(V)$ with V unitary. In fact, suppose A, $B \in \mathcal{A}$. Since A and B are Hermitian they are normal. Now, since $AB \in \mathcal{A}$ we have that AB is Hermitian and thus $AB = (AB)^H = B^HA^H = BA$. Hence $\mathcal{A}$ is a commuting set of normal matrices and we can apply corollary 1.3 obtaining that $\mathcal{A}$ is a unitarily diagonalizable set of matrices.

Second, certain algebras in $\mathbf{D}$ present structures that can be exploited in the field of computations with Toeplitz and Hankel matrices. Furthermore, these structures are often obtained when V is chosen to be a discrete transform matrix, such as the Fourier matrix (see section 1.8.2). For these matrices there are algorithms, known as *fast transform algorithms*, that, given an arbitrary vector $\mathbf{x} \in \mathsf{C}^n$, allow to compute both the products $V\mathbf{x}$ and $V^{-1}\mathbf{x}$ with $O(n \log n)$ arithmetic operations. Concerning the Fourier matrix for example, various of these algorithms exist. One is the fast Fourier transform [68]. We call *algebras associated with transforms* the algebras $D(V) \in \mathbf{D}$ such that V is a transform matrix. We stress that the importance of these algebras relies in the link between interesting structures and fast transform algorithms.

Let us consider more formally the problem of computing with $D(V)$ algebras. Let, for every $A \in D(V) \subseteq \mathsf{C}^{n\times n}$, be $A = VD_AV^{-1}$, with $D_A \in \mathcal{D}$.

(a) Given A, $B \in D(V)$, by virtue of the isomorphism between $D(V)$ and $\mathcal{D}$ we have, for every a, $b \in \mathsf{C}$

$$\begin{aligned} D_{aA+bB} &= aD_A + bD_B, \\ D_{AB} &= D_AD_B. \end{aligned}$$

Moreover, if A is nonsingular

$$D_{A^{-1}} = D_A^{-1}.$$

Thus if D_A and D_B are known D_{aA+bB}, D_{AB} and $D_{A^{-1}}$ can be computed with $O(n)$ operations.

(b) Given $A \in D(V)$ and a vector $\mathbf{x} \in \mathsf{C}^n$, if D_A is known, we can compute both $A\mathbf{x}$ and $A^{-1}\mathbf{x}$ with $f(n) + i(n) + O(n)$ arithmetic operations, where $f(n)$ and $i(n)$ are the number of operations required to compute the products by a vector of V and V^{-1} respectively. In particular for algebras associated with transforms we have $f(n) = O(n \log n)$ and $i(n) = O(n \log n)$ and thus $f(n)+i(n)+O(n) = O(n \log n)$.

Though representing a matrix $A \in D(V)$ by means of its eigenvalues is convenient, we need to join this representation to the entries of A. Under simple conditions on V it is possible to establish a bijective mapping between the eigenvalues of the matrices in $D(V)$ and the elements of their k-th row or column. In this way a matrix in $D(V)$ is completely specified by these elements as well as by its eigenvalues. The following theorem comes from [22].

Theorem 1.13 *Given a nonsingular matrix $V \in \mathsf{C}^{n\times n}$ and an integer $k = 1, \ldots, n$, the following two conditions are equivalent*

(a) *The k-th row of V (respectively column of V^{-1}) has no zero entries.*

(b) *Given arbitrarily a vector $\mathbf{x} \in \mathsf{C}^n$, exists and it is unique a matrix $A \in D(V)$ such that $\mathbf{x}$ is the k-th row (respectively column) of A.*

Proof. We prove the theorem in the case where $\mathbf{x}$ is desired as k-th row of A. The case where it is desired as column is analogous.

We consider first the implication (a)⇒(b). Since by hypothesis $V^T\mathbf{e}_k$ has no zero entries it is possible to define the vector $\mathbf{w} \in \mathsf{C}^n$ as

$$[\mathbf{w}]_i = \frac{1}{[V^T\mathbf{e}_k]_i}.$$

By using $\mathbf{w}$ we define $D_A \in \mathcal{D}$ as

$$D_A = \mathrm{Diag}(\mathbf{w} \circ V^T\mathbf{x}),$$

where $\circ$ denotes entrywise vector product. Now, for the matrix

$$A = VD_AV^{-1} = V\mathrm{Diag}(\mathbf{w} \circ V^T\mathbf{x})V^{-1} \in D(V) \tag{1.3}$$

we have $\mathbf{e}_k^T A = \mathbf{x}^T$. This proves the existence. Let now $A = VD_AV^{-1}$, $B = VD_BV^{-1} \in D(V)$ be such that $\mathbf{e}_k^T A = \mathbf{e}_k^T B = \mathbf{x}^T$. This implies $\mathbf{e}_k^T VD_A = \mathbf{e}_k^T VD_B$. Thus $D_A = D_B$, using again the hypothesis.

We turn now to the implication (b)⇒(a). By hypothesis we have that for $i = 1, \ldots, n$ there exists a matrix $A = VD_AV^{-1} \in D(V)$ such that $\mathbf{e}_k^T A = \mathbf{e}_i^T V^{-1}$. This implies $\mathbf{e}_k^T VD_A = \mathbf{e}_i^T$ and thus $[V^T\mathbf{e}_k]_i \neq 0$. ∎

In order to compute D_A from the k-th row or column of A the product of V^T or V^{-1} by a suitable vector should be computed. If V is a transform matrix usually these products can be computed with $O(n \log n)$ arithmetic operations.

Remark 1.14 With reference to the theorem 1.13, if $V^T\mathbf{e}_k$ has no zero entries we can consider the matrix $\tilde{V} = V\,\mathrm{Diag}(\mathbf{w})$ such that $\mathbf{e}_k^T\tilde{V} = \mathbf{e}^T$, where $\mathbf{e}$ has every entry equal to 1. Consider the algebra $D(\tilde{V})$ and the matrices $A_r \in D(\tilde{V})$ defined as

$$A_r = \tilde{V}\mathrm{Diag}(\tilde{V}^T\mathbf{e}_r)\tilde{V}^{-1} \qquad r = 1, \ldots, n.$$

Then $\mathbf{e}_r$ is the k-th row of A_r, so that these matrices are a base of $D(\tilde{V})$. Moreover, the matrices A_r have a property that is best appreciated by setting $\tilde{V} = (v_{i\,j})$ and $\tilde{V}\mathbf{e}_j = \mathbf{v}_{\cdot\,j}$. With this notation we have

$$A_r\mathbf{v}_{\cdot\,j} = v_{r\,j}\mathbf{v}_{\cdot\,j} \qquad r,\, j = 1, \ldots, n.$$

In [7], this property of the A_r has been called *eigenvector-eigenvalue duality.*

Is it possible to extend the concept of algebra associated with a transform to algebras of higher dimension and, thus, of higher informational content? Obviously, the main problem is to save the binding between structure and fast algorithms. As we will see, this in certain cases will be possible. The theory of centralizers will be the natural framework for dealing with this extension.

1.4 Matrix algebras and polynomials.

Let $X \in \mathsf{C}^{n\times n}$ and let $X^0 = I$. Then

$$P(X) = \left\{ \sum_{k=0}^{n-1} a_k X^k \,|\, a_k \in \mathsf{C} \right\} \tag{1.4}$$

is a commutative matrix algebra with identity, as it can be easily checked. We set $\mathbf{P} = \{P(X) \,|\, X \in \mathsf{C}^{n\times n}\}$.

Proposition 1.15 *Let $P(X) \in \mathbf{P}$.*

1. *Every matrix in $P(X)$ is symmetric (respectively persymmetric, centrosymmetric, diagonalizable) if and only if X is symmetric (respectively persymmetric, centrosymmetric, diagonalizable).*
2. *The dimension of $P(X)$ is equal to the degree of the minimal polynomial of X. Thus $\dim P(X) \le n$. In particular $\dim P(X) = n$ if and only if X is nonderogatory.*

Let $P(X) \in \mathbf{P}$ and let $p(x)$ be the minimal polynomial of X. Suppose that $p(x)$ has degree d. Then there is a natural isomorphism between the algebra $P(X)$ and the algebra of the polynomials of degree less than d in which operations are performed modulo $p(x)$. Thus, if we represent $A = \sum_{k=0}^{d-1} a_k X^k \in P(X)$ using the coefficients a_k we can

- add matrices in $P(X)$ with d arithmetic operations;
- multiply matrices in $P(X)$ with $O(d \log d)$ arithmetic operations by computing a product of polynomials of degree less than d and then dividing by $p(x)$ the result;
- compute the inverse of A, if it exists, with $O(d \log^2 d)$ arithmetic operations, by computing the reciprocal of $\sum_{k=0}^{d-1} a_k x^k$ modulo $p(x)$ (see [20]).

Though theoretically interesting, this binding with polynomial arithmetic hides an important difficulty that concerns the problem of multiplying A by a vector $\mathbf{v}$. If A is represented using the coefficients a_k there are only few cases where the product $A\mathbf{v}$ can be computed efficiently. The most important is the case where X is a Frobenius matrix of the form

$$X = \begin{pmatrix} 0 & \cdots & 0 & c_0 \\ 1 & \ddots & \vdots & \vdots \\ & \ddots & 0 & \vdots \\ O & & 1 & c_{n-1} \end{pmatrix},$$

so that, as well known, $p(x) = x^n - \sum_{i=0}^{n-1} c_i x^i$. Since we have $X^i \mathbf{e}_1 = \mathbf{e}_{i+1}$, we have also $A\mathbf{e}_1 = (a_0, \cdots, a_{n-1})^T$. Thus the elements in the first column of a matrix in $P(X)$ give directly the coefficients of its representation as polynomial in X. In addition we have

$$A\mathbf{v} = A(\sum_{k=0}^{n-1} [\mathbf{v}]_{k+1} X^k)\mathbf{e}_1$$

so that the problem of computing $A\mathbf{v}$ can be reduced to the problem of computing the product of matrices in $P(X)$.

By the above considerations our attention is attracted by algebras $P(X) \in \mathbf{P}$ where X is a Frobenius matrix. Actually, in [33], Di Fiore and Zellini study the structure of a much more wider class of algebras $P(X) \in \mathbf{P}$. Specifically, they consider the case where $X \in \mathsf{C}^{n\times n}$ is a lower Hessenberg matrix

$$X = \begin{pmatrix} r_{11} & b_1 & & O \\ r_{21} & r_{22} & \ddots & \\ \vdots & & \ddots & b_{n-1} \\ r_{n1} & r_{n2} & \cdots & r_{nn} \end{pmatrix} \tag{1.5}$$

(with obvious changes they results apply to the case where X is an upper Hessenberg matrix). In this way they are able to encompass and unify various previously known results. Following [33], in the case where X is lower Hessenberg we will set $H_X = P(X)$ and call H_X Hessenberg Algebra (HA).

Proposition 1.16 *If $b_i \neq 0$ for all i, then X is nonderogatory, so that in this hypothesis* $\dim H_X = n$.

By generalizing some ideas presented in [7], Di Fiore and Zellini provide a convenient basis for H_X. Let $A_k = p_{k-1}(X)$, where $p_k(\lambda)$, for $k = 1, \ldots, n$, is the characteristic polynomial of the $k \times k$ top left submatrix of X and $p_0(\lambda) = 1$. Clearly $H_X = \{\sum_{k=1}^{n} a_k A_k\}$. Moreover, the matrices A_k have the following important property.

Theorem 1.17 ([33]) *We have*

$$\mathbf{e}_1^T A_k = \mathbf{e}_k^T \prod_{i=1}^{k-1} b_i.$$

Proof. The polynomials $p_k(\lambda)$ obey to the following recurrence

$$\begin{aligned} &p_0(\lambda) = 1, \qquad p_1(\lambda) = \lambda - r_{1\,1}, \\ &p_j(\lambda) = (\lambda - r_{j\,j})p_{j-1}(\lambda) - \sum_{m=1}^{j-1} r_{j\,m}(\prod_{i=m}^{j-1} b_i)p_{m-1}(\lambda), \qquad j = 2, \ldots, n-1. \end{aligned}$$

Thus we have

$$\begin{aligned} &A_1 = I, \qquad A_2 = X - r_{1\,1}I, \\ &A_{j+1} = (X - r_{j\,j}I)A_j - \sum_{m=1}^{j-1} r_{j\,m}(\prod_{i=m}^{j-1} b_i)A_m, \qquad j = 2, \ldots, n-1. \end{aligned}$$

Therefore $\mathbf{e}_1^T A_1 = \mathbf{e}_1^T$, $\mathbf{e}_1^T A_2 = b_1 \mathbf{e}_2^T$ and, proceding by induction we find

$$\begin{aligned} \mathbf{e}_1^T A_{j+1} &= \mathbf{e}_1^T (X - r_{j\,j}I)A_j - \sum_{m=1}^{j-1} r_{j\,m}(\prod_{i=m}^{j-1} b_i)\mathbf{e}_m^T(\prod_{i=1}^{m-1} b_i) = \\ &= \mathbf{e}_1^T A_j(X - r_{j\,j}I) - (\prod_{i=1}^{j-1} b_i)\sum_{m=1}^{j-1} r_{j\,m}\mathbf{e}_m^T = \\ &= (\prod_{i=1}^{j-1} b_i)\mathbf{e}_j^T(X - r_{j\,j}I) - (\prod_{i=1}^{j-1} b_i)\sum_{m=1}^{j-1} r_{j\,m}\mathbf{e}_m^T = \\ &= \mathbf{e}_{j+1}^T \prod_{i=1}^{j} b_i. \end{aligned}$$

■

If $b_i \neq 0$ for all i we can set $X_k = \left(1/\prod_{i=1}^{k-1} b_i\right) A_k$. Being the set $\{A_k \mid k = 1, \ldots, n\}$ a basis for H_X the set $\{X_k \mid k = 1, \ldots, n\}$ is a basis as well and, obviously, $\mathbf{e}_1^T X_k = \mathbf{e}_k^T$. Given a vector $\mathbf{a} = (a_k) \in \mathsf{C}^n$ we set

$$H_X(\mathbf{a}) = \sum_{k=1}^{n} a_k X_k, \tag{1.6}$$

being this notation motivated from the fact that $\mathbf{e}_1^T H_X(\mathbf{a}) = \mathbf{a}^T$. Hence, if $b_i \neq 0$ for all i, the first row of a matrix belonging to H_X can be freely specified. Moreover, two matrices in H_X having the same first row must be equal. This has to be compared with theorem 1.13. The notation (1.6) will be used extensively in the following.

The following properties of the matrices X_k are important (see also [33]).

Theorem 1.18 *Let $b_i \neq 0$ for all i.*

(1) *We have*

$$X_k X_j = \sum_{i=1}^{n} [X_j]_{ki} X_i. \tag{1.7}$$

Moreover, $X_k X_j = X_j X_k$ and therefore $\mathbf{e}_k^T X_j = \mathbf{e}_j^T X_k$.

(2) *If X is centrosymmetric then $X_n = J$.*

(3) *If X is persymmetric then $X_k \mathbf{e}_j = X_{n+1-j} \mathbf{e}_{n+1-k}$.*

(4) *If X is symmetric then $X_k \mathbf{e}_j = X_j \mathbf{e}_k$ and*

$$X_k X_j = \sum_{i=1}^{n} [X_i]_{kj} X_i. \tag{1.8}$$

This property states that the multiplication table of the X_k has the same structure of that of the matrix $H_X(\mathbf{a}) = \sum_{k=1}^{n} a_k X_k$.

(5) *If X is tridiagonal and $r_{i+1\,i} \neq 0$, for $i = 1, \ldots, n-1$, then X_n is nonsingular.*

Proof.

(1) We have $X_k X_j \in H_X$ and $\mathbf{e}_1^T X_k X_j = \mathbf{e}_k^T X_j$. Thus $X_k X_j$ belongs to H_X and has $X_j^T \mathbf{e}_k$ as first row. Hence from (1.6) we have $X_k X_j = \sum_{i=1}^{n} [X_j]_{ki} X_i$. Moreover, $X_k X_j = X_j X_k$ since the X_i belong to H_X and H_X is a commutative algebra. Therefore

$$\mathbf{e}_k^T X_j = \mathbf{e}_1^T X_k X_j = \mathbf{e}_1^T X_j X_k = \mathbf{e}_j^T X_k.$$

(2) Setting $j = n$ in $\mathbf{e}_k^T X_j = \mathbf{e}_j^T X_k$ we have $\mathbf{e}_k^T X_n = \mathbf{e}_n^T X_k$. Now, since X is centrosymmetric, the X_k are centrosymmetric as well. Hence $\mathbf{e}_n^T X_k = \mathbf{e}_n^T J J X_k = \mathbf{e}_1^T X_k J = \mathbf{e}_k^T J$.

(3) From $\mathbf{e}_j^T X_k = \mathbf{e}_k^T X_j$, by transposing both members we obtain $X_k^T \mathbf{e}_j = X_j^T \mathbf{e}_k$. Now, if X is persymmetric then every X_k is so. Therefore $X_k \mathbf{e}_{n+1-j} = X_j \mathbf{e}_{n+1-k}$, for $j = 1, \ldots, n$. By renaming the indexes we have $X_k \mathbf{e}_j = X_{n+1-j} \mathbf{e}_{n+1-k}$, for $j = 1, \ldots, n$.

(4) If X is symmetric then every X_k is so. Thus we have $X_k \mathbf{e}_j = X_j \mathbf{e}_k$ that implies $[X_j]_{ik} = [X_k]_{ij}$ and thus, $[X_j]_{ki} = [X_i]_{kj}$. Using this equality and (1.7) we find (1.8).

(5) We use again the relation $\mathbf{e}_k^T X_n = \mathbf{e}_n^T X_k$. Note that, since X is tridiagonal, then X^p, for $p = 1, \ldots, n-1$, is $(2p+1)$-diagonal. In particular, X_k is $(2k-1)$-diagonal, since it is a polinomial of degree $k-1$ in X. Moreover

$$[X_k]_{n\,n+1-k} = (1/(\prod_{i=1}^{k-1} b_i))[X^{k-1}]_{n\,n+1-k}, \qquad k = 2, \ldots, n.$$

Now, an easy induction shows that

$$[X^{k-1}]_{n\,n+1-k} = \prod_{i=n+1-k}^{n-1} r_{i+1\,i}.$$

Setting $d_k = (\prod_{i=n+1-k}^{n-1} r_{i+1\,i})/(\prod_{i=1}^{k-1} b_i)$, for $k = 2, \ldots, n$, we have

$$X_n = \begin{pmatrix} 0 & \cdots & 0 & 1 \\ \vdots & \iddots & d_2 & * \\ 0 & \iddots & \iddots & \vdots \\ d_n & * & \cdots & * \end{pmatrix},$$

where $*$ denote unimportant entries. This shows the nonsingularity of X_n.

∎

Remark 1.19 Note that formula (1.7) holds for every set of matrices X_k, $k = 1, \ldots, n$, forming the basis of a matrix algebra and such that $\mathbf{e}_1^T X_k = \mathbf{e}_k^T$. If in addition we require that the X_k should be symmetric we immediately get the (1.8).

1.5 Matrix algebras and centralizers.

If $A \in \mathsf{C}^{n\times n}$ then the set

$$Z(A) = \left\{X \in \mathsf{C}^{n\times n} \mid AX = XA\right\}$$

is a matrix algebra known as *centralizer* of A. The theory of centralizers is discussed in [29]. Here we briefly recall the main results. Much of what is known about centralizers has been discovered in the study of matrix equations: for an introduction to this huge subject the reader may consult [39, 46].

Given a matrix $A \in \mathsf{C}^{n\times n}$ it is possible to give a description of the matrices in $Z(A)$ by using the Jordan canonical form of A. The reader can see [29] for such a description, that can be taken as starting point for proving the following result.

Theorem 1.20 *If $A \in \mathsf{C}^{n\times n}$ then $\dim Z(A) \geq n$. In particular, the following statements are equivalent:*

(1) *A is nonderogatory;*

(2) *$Z(A) = P(A)$;*

(3) *$\dim Z(A) = n$;*

(4) *$Z(A)$ is commutative.*

The inequality $\dim Z(A) \geq n$ is the main reason of our interest in centralizers: this theory allows us to deal with algebras of dimension greater than n.

The matrices in $Z(A)$ have a particularly simple structure in the case where A is diagonalizable, as the following results show (see [29, pp. 192–203]).

Theorem 1.21 *Let $A \in \mathsf{C}^{n\times n}$ be diagonalizable and let $V \in \mathsf{C}^{n\times n}$ be a nonsingular matrix such that*

$$A = V \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{pmatrix} V^{-1},$$

where $\lambda_1, \cdots, \lambda_n$, are the eigenvalues of A. Let S_A be an $n \times n$ matrix defined as follows

$$[S_A]_{i\,j} = \begin{cases} 1 & \text{if } \lambda_i = \lambda_j \\ 0 & \text{otherwise.} \end{cases}$$

Then we have

$$Z(A) = \left\{X \in \mathsf{C}^{n\times n} \mid X = V(S_A \circ M)V^{-1}\right\},$$

where $\circ$ denotes entrywise matrix product and M is an arbitrary matrix in $\mathsf{C}^{n\times n}$.

Corollary 1.22 *If A is diagonalizable and if $\lambda_1, \cdots, \lambda_d$ are the distinct eigenvalues of A then we have*

$$\dim Z(A) = \sum_{i=1}^{d} \sigma^2(\lambda_i),$$

where $\sigma(\lambda_i)$ is the algebraic (and geometric) multiplicity of the eigenvalue λ_i.

1.6 Comparison.

In this section we want to compare the families of matrix algebras introduced so far. Recall that

- $\mathbf{D} = \{D(V) \,|\, V \in \mathsf{C}^{n\times n} \text{ nonsingular}\}$;
- $\mathbf{P} = \{P(A) \,|\, A \in \mathsf{C}^{n\times n}\}$.

In addition we set

- $\mathbf{Z} = \{Z(A) \,|\, A \in \mathsf{C}^{n\times n}\}$.

Let us begin by examining the set $\mathbf{P} \cap \mathbf{Z}$. If $\mathcal{A} \in \mathbf{P} \cap \mathbf{Z}$ then $\dim \mathcal{A} = n$, since from $\mathcal{A} \in \mathbf{P}$ we get $\dim \mathcal{A} \leq n$ and from $\mathcal{A} \in \mathbf{Z}$ we get $\dim \mathcal{A} \geq n$. Moreover, if $\mathcal{A} = P(A)$ (or $\mathcal{A} = Z(A)$) and $\dim \mathcal{A} = n$ then A has to be nonderogatory and thus $P(A) = Z(A)$ and $\mathcal{A} \in \mathbf{P} \cap \mathbf{Z}$. This implies that $\mathbf{P} - \mathbf{Z}$ contains only algebras of dimension lower than n and $\mathbf{Z} - \mathbf{P}$ only algebras of dimension greater than n.

It is easy to establish the following inclusion.

Proposition 1.23 *We have* $\mathbf{D} \subseteq \mathbf{P} \cap \mathbf{Z}$.

Proof. Let $\mathcal{A} \in \mathbf{D}$ with $\mathcal{A} = D(V)$. Moreover let $A = VD_AV^{-1} \in D(V)$ be a matrix with n distinct eigenvalues. Obviously we have $P(A) \subseteq D(V) \subseteq Z(A)$. Now, since A is diagonalizable and has distinct eigenvalues, it is nonderogatory, so that by theorem 1.20 we have $P(A) = Z(A)$. Thus $P(A) = D(V) = Z(A)$. This proves that $D(V) \in \mathbf{P}$ and $D(V) \in \mathbf{Z}$ so that $D(V) \in \mathbf{P} \cap \mathbf{Z}$. ∎

Obviously $\mathbf{D} \neq \mathbf{P} \cap \mathbf{Z}$ since for example any Hessenberg algebra H_X (see section 1.4), with X nonderogatory but not diagonalizable, belongs to $\mathbf{P} \cap \mathbf{Z}$ but not to $\mathbf{D}$. We collect in figure 1.1 the results of this discussion.

Figure 1.1: Relations between the families of algebras $\mathbf{D}$, $\mathbf{P}$ and $\mathbf{Z}$.

1.7 Cross sum condition.

Let us consider the set of all the matrices $A \in \mathsf{C}^{n\times n}$, with $A = (a_{i\,j})$, $i, j = 1, \ldots, n$, satisfying the *cross-sum* condition

$$a_{i-1\,j} + a_{i+1\,j} = a_{i\,j-1} + a_{i\,j+1}, \qquad i, j = 1, \ldots, n, \tag{1.9}$$

where the following *boundary conditions* hold

$$a_{0\,j} = a_{i\,0} = a_{n+1\,j} = a_{i\,n+1} = 0, \qquad i, j = 1, \ldots, n \tag{1.10}$$

This set of matrices was introduced in [14] and was shown to be a matrix algebra, denoted with the symbol τ. In [14] it was pointed out that, once the first row of a matrix $A \in \tau$ has been arbitrarily fixed, the remaining entries of A are univocally determined by the (1.9) and (1.10). For example, if $n = 5$ and we set $a_{1\,j} = a_j$, $j = 1, \ldots, 5$, then we have

$$A = \begin{pmatrix} a_1 & a_2 & a_3 & a_4 & a_5 \\ a_2 & a_1 + a_3 & a_2 + a_4 & a_3 + a_5 & a_4 \\ a_3 & a_2 + a_4 & a_1 + a_3 + a_5 & a_2 + a_4 & a_3 \\ a_4 & a_3 + a_5 & a_2 + a_4 & a_1 + a_3 & a_2 \\ a_5 & a_4 & a_3 & a_2 & a_1 \end{pmatrix}.$$

Actually, conditions (1.9) and (1.10) can be restated in matrix notation in a much more compact form. To this end, let us consider the matrix

$$T = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ 1 & 0 & 1 & \ddots & \vdots \\ 0 & 1 & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & 0 & 1 \\ 0 & \cdots & 0 & 1 & 0 \end{pmatrix}.$$

Given $A \in \mathsf{C}^{n\times n}$ an easy direct computation shows that A satisfies condition (1.9) assuming the boundary conditions (1.10) if and only if $AT = TA$. This leads to the following conclusion.

Proposition 1.24 *We have $\tau = Z(T)$.*

In the view of proposition 1.24, the theory developed so far gives us plenty of information about the algebra τ.

1. Every eigenvalue of T has geometric multiplicity 1 (see example 1.6), so that T is nonderogatory. Thus, by theorem 1.20, we have that $\tau = Z(T)$ is a commutative algebra of dimension n and that $\tau = P(T)$.

2. Notice that T is symmetric, persymmetric and thus centrosymmetric. By proposition 1.15 we have that every matrix in $\tau = P(T)$ is so. Moreover T is an Hessenberg matrix such that (see (1.5)) $b_i \neq 0$, for $i = 1, \ldots, n-1$. Thus we can construct the n matrices $T_k = p_{k-1}(T)$, $k = 1, \ldots, n$, being $p_k(\lambda)$ the characteristic polynomial of the $k \times k$ top left submatrix of T, with $p_0(\lambda) = 1$. The matrices T_k constitute a basis of the algebra τ. Moreover, we can apply theorem 1.18. We have $\mathbf{e}_j^T T_k = \mathbf{e}_k^T T_j$. Using the centrosymmetry of T we obtain $T_n = J$. In addition, using the symmetry of T we have $T_k \mathbf{e}_j = T_j \mathbf{e}_k$ and

$$T_k T_j = \sum_{i=1}^{n} [T_i]_{k\,j} T_i.$$

3. Since T has real entries and is symmetric, it is diagonalizable by means of an orthogonal transformation V. Being diagonalizable and nonderogatory, T must have distinct eigenvalues, so that, by the results obtained in section 1.6, we have $\tau = D(V)$. As we will see, the matrix V can be explicitly located and turns out to be a discrete transform matrix.

The algebra τ has been extensively used in the study of the spectral and computational properties of Toeplitz matrices [14, 15, 73]. More recently, it has found application in the development of preconditioning strategies for the conjugate gradient method [18, 32, 31], and in relation with the concept of displacement rank (see the next chapter of this thesis). Here, we want to discuss in some detail two observations that stay, we believe, at the very heart of the importance that the algebra τ has. In brief these two observations are:

(a) the cross-sum condition is a distinctive characteristic of Toeplitz and Hankel structures;

(b) the matrix T has certain trigonometric sequences as "near" eigenvectors.

Observation (a) is taken, with some adaptations, from [12], and explains why τ algebra, and also other algebras satisfying the cross-sum condition (see section 1.8.1), have been used in the field of computations with Toeplitz and Hankel matrices. Observation (b) is taken from [68] and points out the binding between these algebras and fast trigonometric transforms (see again section 1.8.1).

Let us begin by (a). Let $\mathcal{T}$ and $\mathcal{H}$ be respectively the linear spaces of Toeplitz and Hankel matrices. We have $\dim \mathcal{T} = \dim \mathcal{H} = 2n-1$. Moreover, the space $\mathcal{T} \cap \mathcal{H}$ has dimension 2, since it is spanned by the two chessboard-structured matrices

$$\begin{pmatrix} 1 & 0 & 1 & \cdots \\ 0 & 1 & 0 & \cdots \\ 1 & 0 & 1 & \\ \vdots & \vdots & & \ddots \end{pmatrix}, \qquad \begin{pmatrix} 0 & 1 & 0 & \cdots \\ 1 & 0 & 1 & \cdots \\ 0 & 1 & 0 & \\ \vdots & \vdots & & \ddots \end{pmatrix}.$$

Thus we have $\dim(\mathcal{T} + \mathcal{H}) = 4n-4$. Let us consider the linear operator $\mathcal{L}_T : \mathsf{C}^{n\times n} \to \mathsf{C}^{n\times n}$ such that $\mathcal{L}_T(A) = AT - TA$. Note that τ is the nullspace of $\mathcal{L}_T$. In addition, we define the linear space

$$\mathcal{B} = \{\mathbf{e}_1\mathbf{u}_1^T + \mathbf{e}_n\mathbf{u}_2^T + \mathbf{u}_3\mathbf{e}_1^T + \mathbf{u}_4\mathbf{e}_n^T \,|\, \mathbf{u}_i \in \mathsf{C}^n\}$$

and we let $\mathcal{X}$ denote the inverse image under $\mathcal{L}_T$ of $\mathcal{B}$, i.e.,

$$\mathcal{X} = \{A \in \mathsf{C}^{n\times n} \,|\, \mathcal{L}_T(A) \in \mathcal{B}\}.$$

Note that if $A = (a_{i\,j}) \in \mathcal{X}$, i.e. $\mathcal{L}_T(A) = \mathbf{e}_1\mathbf{u}_1^T + \mathbf{e}_n\mathbf{u}_2^T + \mathbf{u}_3\mathbf{e}_1^T + \mathbf{u}_4\mathbf{e}_n^T$ for some vectors $\mathbf{u}_i$, then A satisfies the cross-sum condition with the boundary conditions

$$\begin{aligned} a_{0\,i} &= [\mathbf{u}_1]_i, & a_{i\,0} &= -[\mathbf{u}_3]_i, \\ a_{n+1\,i} &= [\mathbf{u}_2]_i, & a_{i\,n+1} &= -[\mathbf{u}_4]_i. \end{aligned}$$

Clearly, the following result holds.

Proposition 1.25 *We have $\mathcal{T} \subseteq \mathcal{X}$ and $\mathcal{H} \subseteq \mathcal{X}$, so that $\mathcal{T} + \mathcal{H} \subseteq \mathcal{X}$.*

Proof. Both Toeplitz and Hankel matrices satisfy the cross-sum condition setting in a suitable way the boundary conditions. ∎

The next question is: what is the dimension of the space $\mathcal{X}$? The answer is shown in the next theorem. The proof that we give is based on the use of certain equalities, known as orthogonality conditions, that will be proved in the next chapter (see section 2.1).

Theorem 1.26 *We have* $\dim \mathcal{X} = 4n-4$.

Proof. First of all, remember that τ is the nullspace of the operator $\mathcal{L}_T$. Hence, if we denote with $\mathcal{L}_T(\mathcal{X}) \subseteq \mathcal{B}$ the image of $\mathcal{X}$ under $\mathcal{L}_T$, we have

$$\dim \mathcal{X} = \dim(\tau \cap \mathcal{X}) + \dim \mathcal{L}_T(\mathcal{X}). \tag{1.11}$$

From proposition 1.25 we get $\dim \mathcal{X} \geq 4n-4$, and obviously we have $\dim(\tau \cap \mathcal{X}) = \dim \tau = n$. Now, we want to obtain some information on $\dim \mathcal{L}_T(\mathcal{X})$.

If $B = \mathbf{e}_1\mathbf{u}_1^T + \mathbf{e}_n\mathbf{u}_2^T + \mathbf{u}_3\mathbf{e}_1^T + \mathbf{u}_4\mathbf{e}_n^T \in \mathcal{L}_T(\mathcal{X})$ then we have $AT - TA = B$ for some $A \in \mathcal{X}$. By virtue of theorem 2.4 we have that the following equalities must hold

$$\mathbf{e}_1^T T_k \mathbf{u}_1 + \mathbf{e}_n^T T_k \mathbf{u}_2 + \mathbf{u}_3^T T_k \mathbf{e}_1 + \mathbf{u}_4^T T_k \mathbf{e}_n = 0, \qquad k = 1, \ldots, n.$$

By using the properties of the T_k (see theorem 1.18) the preceding equalities become

$$\mathbf{e}_k^T \mathbf{u}_1 + \mathbf{e}_{n+1-k}^T \mathbf{u}_2 + \mathbf{u}_3^T \mathbf{e}_k + \mathbf{u}_4^T \mathbf{e}_{n+1-k} = 0, \qquad k = 1, \ldots, n.$$

Observe that we can write them in the more compact form

$$\mathbf{u}_1 + J\mathbf{u}_2 + \mathbf{u}_3 + J\mathbf{u}_4 = 0.$$

This means that the elements in, say, the first column of B are completely determined when the remaining $3n-4$ nonzero entries of B are given. Thus $\dim \mathcal{L}_T(\mathcal{X}) \leq 3n-4$. This implies that equality (1.11) can hold only if $\dim \mathcal{X} = 4n-4$ and $\dim \mathcal{L}_T(\mathcal{X}) = 3n-4$. This proves the thesis and, consequently, the inclusion $\tau \subseteq \mathcal{T} + \mathcal{H}$. ∎

By using proposition 1.25 and theorem 1.26 we obtain the following

Corollary 1.27 *We have $\mathcal{X} = \mathcal{T} + \mathcal{H}$.*

In other words, a matrix A satisfies the cross-sum condition under suitable boundary conditions if and only if A is the sum of a Toeplitz and of an Hankel matrix.

The next proposition points out how certain centralizers are subsets of $\mathcal{T} + \mathcal{H}$.

Proposition 1.28 *Set*

$$T^{\times} = T + \varepsilon \mathbf{e}_1\mathbf{e}_1^T + \varphi \mathbf{e}_n\mathbf{e}_n^T + \alpha \mathbf{e}_n\mathbf{e}_1^T + \beta \mathbf{e}_1\mathbf{e}_n^T = \begin{pmatrix} \varepsilon & 1 & 0 & \cdots & 0 & \beta \\ 1 & 0 & 1 & \ddots & & 0 \\ 0 & 1 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 1 & 0 \\ 0 & & \ddots & 1 & 0 & 1 \\ \alpha & 0 & \cdots & 0 & 1 & \varphi \end{pmatrix}.$$

Then we have $Z(T^{\times}) \subseteq (\mathcal{T} + \mathcal{H})$.

In the next section we will study some interesting examples of these centralizers. Moreover, in a forthcoming paper [9] we will show that, under certain conditions, the centralizers $Z(T^{\times})$ are the only maximal algebras included in the space $\mathcal{T} + \mathcal{H}$.

Let us turn now to observation (b). We follow, in the substance, the treatment in [68]. We set $f_i = \sin((ai+b)\theta)$, or in turn $f_i = \cos((ai+b)\theta)$, for any integer i, where a and b are integers and $\theta \in \mathsf{R}$. In both cases, by using simple trigonometric identities we have

$$f_{i-1} + f_{i+1} = 2f_i \cos(a\theta).$$

Adding f_{i-1} to f_{i+1} is exactly what we do when we apply the matrix T to a vector whose entries are the f_i themselves.

Theorem 1.29 *Suppose that a, b, p and q are integers with $p < q$ and $\theta \in \mathsf{R}$. If $f_i = \sin((ai+b)\theta)$ or $f_i = \cos((ai+b)\theta)$ then*

$$T \begin{pmatrix} f_p \\ f_{p+1} \\ \vdots \\ f_{q-1} \\ f_q \end{pmatrix} = 2\cos(a\theta) \begin{pmatrix} f_p \\ f_{p+1} \\ \vdots \\ f_{q-1} \\ f_q \end{pmatrix} - \begin{pmatrix} f_{p-1} \\ 0 \\ \vdots \\ 0 \\ f_{q+1} \end{pmatrix}. \tag{1.12}$$

Proof. We have

$$T \begin{pmatrix} f_p \\ f_{p+1} \\ \vdots \\ f_{q-1} \\ f_q \end{pmatrix} = \begin{pmatrix} f_{p+1} \\ f_p + f_{p+2} \\ \vdots \\ f_{q-2} + f_q \\ f_{q-1} \end{pmatrix} = \begin{pmatrix} 2f_p\cos(a\theta) - f_{p-1} \\ 2f_{p+1}\cos(a\theta) \\ \vdots \\ 2f_{q-1}\cos(a\theta) \\ 2f_q\cos(a\theta) - f_{q+1} \end{pmatrix} = 2\cos(a\theta) \begin{pmatrix} f_p \\ f_{p+1} \\ \vdots \\ f_{q-1} \\ f_q \end{pmatrix} - \begin{pmatrix} f_{p-1} \\ 0 \\ \vdots \\ 0 \\ f_{q+1} \end{pmatrix}.$$

∎

1.8 A survey of new and classic matrix algebras.

In this section we present some important examples of matrix algebras that are subsets of the space $\mathcal{T}+\mathcal{H}$ (see section 1.7). First we will consider a family of algebras strictly related to the algebra τ. Then we will turn to circulant and skewcirculant matrices, setting the ground for introducing two algebras of dimension higher than n and that include circulants and skewcirculants respectively.

1.8.1 The algebras $\tau_{\varepsilon\varphi}$.

Suppose that ε, $\varphi \in \mathsf{R}$ and set

$$T_{\varepsilon\varphi} = \begin{pmatrix} \varepsilon & 1 & 0 & \cdots & 0 \\ 1 & 0 & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & 0 & 1 \\ 0 & \cdots & 0 & 1 & \varphi \end{pmatrix}. \tag{1.13}$$

Obviously $T_{00} = T$ (see section 1.7). Analogously to what we have seen for the matrix T, for every ε, $\varphi \in \mathsf{R}$ the matrix $T_{\varepsilon\varphi}$ is diagonalizable by means of an orthogonal matrix, that we will denote with $M_{\varepsilon\varphi}$, and has n distinct eigenvalues. Thus we get $Z(T_{\varepsilon\varphi}) = P(T_{\varepsilon\varphi}) = D(M_{\varepsilon\varphi})$. Moreover, observe that $T_{\varepsilon\varphi}$ is a Hessenberg matrix so that $P(T_{\varepsilon\varphi})$ is a Hessenberg algebra, and the results discussed in section 1.4 for these algebras hold. Here, following [24], we will denote these algebras with the more expressive symbol $\tau_{\varepsilon\varphi}$.

In some special cases, by means of theorem 1.29, it is possible to obtain the matrices $M_{\varepsilon\varphi}$ in closed form.

1. Consider $\varepsilon = 0$ and $\varphi = 0$. We have $\tau_{00} = \tau$. The matrix

$$M_{00} = \sqrt{\frac{2}{n+1}}\,(\sin\frac{ij\pi}{n+1}), \qquad i,\, j = 1, \ldots, n, \tag{1.14}$$

is symmetric and orthogonal. Moreover, the following equality holds

$$T_{00}M_{00} = M_{00}\,\mathrm{Diag}((2\cos\frac{j\pi}{n+1})), \qquad j = 1, \ldots, n, \tag{1.15}$$

i.e., the columns of M_{00} are eigenvectors of the matrix T_{00}, and its eigenvalues are the numbers $2\cos\frac{j\pi}{n+1}$, with $j = 1, \ldots, n$. In order to prove this it is handy to use theorem 1.29, setting $p = 1$, $q = n$, $a = 1$, $b = 0$, $\theta = \theta_j = \frac{j\pi}{n+1}$ and $f_i = \sin\frac{ij\pi}{n+1}$. This yields $f_0 = f_{n+1} = 0$ for every $j = 0, \ldots, n-1$.

The structure of the generic matrix in $\tau_{00} = \tau$ has been shown for $n = 5$ in section 1.7.

2. Consider $\varepsilon = \varphi = 1$ and $\varepsilon = \varphi = -1$. The two matrices

$$M_{11} = \sqrt{\frac{2}{n}}\,(k_j\cos\frac{(2i+1)j\pi}{2n}), \qquad i,\, j = 0, \ldots, n-1, \tag{1.16}$$

$$M_{-1-1} = \sqrt{\frac{2}{n}}\,(k_j\sin\frac{(2i-1)j\pi}{2n}), \qquad i,\, j = 1, \ldots, n. \tag{1.17}$$

where $k_j = 1/\sqrt{2}$ for $j = 0$ and $j = n$ and $k_j = 1$ otherwise, are orthogonal. Moreover, the two following relations hold

$$T_{11}M_{11} = M_{11}\,\mathrm{Diag}((2\cos\frac{j\pi}{n})), \qquad j = 0, \ldots, n-1, \tag{1.18}$$

$$T_{-1-1}M_{-1-1} = M_{-1-1}\,\mathrm{Diag}((2\cos\frac{j\pi}{n})), \qquad j = 1, \ldots, n. \tag{1.19}$$

These relations can be proved by using theorem 1.29. The algebras τ_{11} and τ_{-1-1} are strictly related as well as the matrices M_{11} and M_{-1-1}. In fact setting $D_{-1} = \mathrm{Diag}(((-1)^j))$ with $j = 0, \ldots, n-1$, we have

$$T_{-1-1} = -D_{-1}T_{11}D_{-1}, \tag{1.20}$$
$$M_{11}J = D_{-1}M_{-1-1}. \tag{1.21}$$

As an example, we display a generic matrix belonging to τ_{11} in the case $n = 5$

$$\begin{pmatrix} a_1 & a_2 & a_3 & a_4 & a_5 \\ a_2 & a_1 - a_2 + a_3 & a_2 - a_3 + a_4 & a_3 - a_4 + a_5 & a_4 \\ a_3 & a_2 - a_3 + a_4 & \sum_{i=1}^{5}(-1)^{i-1}a_i & a_2 - a_3 + a_4 & a_3 \\ a_4 & a_3 - a_4 + a_5 & a_2 - a_3 + a_4 & a_1 - a_2 + a_3 & a_2 \\ a_5 & a_4 & a_3 & a_2 & a_1 \end{pmatrix}.$$

3. Consider $\varepsilon = 1$, $\varphi = -1$ and $\varepsilon = -1$, $\varphi = 1$. We set

$$M_{1-1} = \sqrt{\frac{2}{n}}\,(\cos\frac{(2i+1)(2j+1)\pi}{4n}), \qquad i, j = 0, \ldots, n-1, \tag{1.22}$$
$$M_{-11} = \sqrt{\frac{2}{n}}\,(\sin\frac{(2i+1)(2j+1)\pi}{4n}), \qquad i, j = 0, \ldots, n-1. \tag{1.23}$$

The matrices M_{1-1} and M_{-11} are symmetric and orthogonal and we have (for the proof, use once again theorem 1.29)

$$T_{1-1}M_{1-1} = M_{1-1}\,\mathrm{Diag}((2\cos\frac{(2j+1)\pi}{2n})), \qquad j = 0, \ldots, n-1, \tag{1.24}$$
$$T_{-11}M_{-11} = M_{-11}\,\mathrm{Diag}((2\cos\frac{(2j+1)\pi}{2n})), \qquad j = 0, \ldots, n-1. \tag{1.25}$$

Moreover we have relations analogous to the (1.20) and (1.21)

$$T_{1-1} = -D_{-1}T_{-11}D_{-1}, \tag{1.26}$$
$$M_{1-1}J = D_{-1}M_{-11}. \tag{1.27}$$

As an example, we display a generic matrix belonging to τ_{1-1} in the case $n = 5$

$$\begin{pmatrix} a_1 & a_2 & a_3 & a_4 & a_5 \\ a_2 & a_1 - a_2 + a_3 & a_2 - a_3 + a_4 & a_3 - a_4 + a_5 & a_4 - 2a_5 \\ a_3 & a_2 - a_3 + a_4 & a_1 - a_2 + a_3 - a_4 + a_5 & a_2 - a_3 + a_4 - 2a_5 & a_3 - 2a_4 + 2a_5 \\ a_4 & a_3 - a_4 + a_5 & a_2 - a_3 + a_4 - 2a_5 & a_1 - a_2 + a_3 - 2a_4 + 2a_5 & a_2 + 2\sum_{i=3}^{5}(-1)^i a_i \\ a_5 & a_4 - 2a_5 & a_3 - 2a_4 + 2a_5 & a_2 + 2\sum_{i=3}^{5}(-1)^i a_i & a_1 - 2\sum_{i=2}^{5}(-1)^i a_i \end{pmatrix}.$$

All the matrices in (1.14,1.16,1.17,1.22,1.23) are matrices of real discrete transforms computable with quite efficient fast algorithms [70]. For example, particular attention has been devoted to the computation of the transforms $\mathbf{x} \to M_{11}\mathbf{x}$ and $\mathbf{x} \to M_{11}^T\mathbf{x}$, for their importance in image and signal processing [2, 69, 47].

The following proposition will be useful.

Proposition 1.30 *Let $\mathbf{x} \in \mathsf{C}^n$. For $(\varepsilon, \varphi) \in \{(0,0), (1,1), (-1,-1), (1,-1), (-1,1)\}$ exists and is unique a matrix $\tau_{\varepsilon\varphi}(\mathbf{x}) \in \tau_{\varepsilon\varphi}$ such that $\mathbf{e}_1^T\tau_{\varepsilon\varphi}(\mathbf{x}) = \mathbf{x}^T$.*

Proof. The first row of the matrices $M_{\varepsilon\varphi}$, for $(\varepsilon, \varphi) \in \{(0,0), (1,1), (-1,-1), (1,-1), (-1,1)\}$ does not contain zero entries. Thus we can exploit theorem 1.13. Specifically, we have

$$\tau_{\varepsilon\varphi}(\mathbf{x}) = M_{\varepsilon\varphi}\mathrm{Diag}(\mathbf{v}_{\varepsilon\varphi} \circ M_{\varepsilon\varphi}^T\mathbf{x})M_{\varepsilon\varphi}^T$$

where $\circ$ denotes entrywise vector product and where $[\mathbf{v}_{\varepsilon\varphi}]_i = 1/[M_{\varepsilon\varphi}^T\mathbf{e}_1]_i$. ∎

1.8.2 Circulant and skewcirculant matrices.

The Fourier matrix is defined as

$$F = \frac{1}{\sqrt{n}}(\omega^{ij}), \qquad i,\, j = 0, \ldots, n-1,$$

where $\omega = e^{-\mathbf{i}\frac{2\pi}{n}}$, being $\mathbf{i}$ the complex unit. Recall that $F^{-1} = F^H$, i.e., F is unitary. For aesthetic reasons, besides F, we will use the matrix $\Omega = \sqrt{n}F$.

Set $D_\omega = \mathrm{Diag}\left((\omega^i)\right)$, $i = 0, \ldots, n-1$, and consider the matrix

$$C = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & & \ddots & \ddots & 1 \\ 1 & 0 & \cdots & 0 & 0 \end{pmatrix}. \tag{1.28}$$

It is well known [29] that

$$C = FD_\omega F^H, \tag{1.29}$$

so that C is diagonalizable and has n distinct eigenvalues. The algebra $P(C) = D(F) = Z(C)$, that we will denote with $\mathcal{C}$, is the algebra of the circulant matrices. Observe that C is a Hessenberg matrix so that we can use notation 1.6 and, given $\mathbf{a} = (a_k) \in \mathsf{C}^n$, we set $\mathcal{C}(\mathbf{a}) = \sum_{k=0}^{n-1} a_k C^k$. For example, for $n = 4$ we have

$$\mathcal{C}(\mathbf{a}) = \begin{pmatrix} a_0 & a_1 & a_2 & a_3 \\ a_3 & a_0 & a_1 & a_2 \\ a_2 & a_3 & a_0 & a_1 \\ a_1 & a_2 & a_3 & a_0 \end{pmatrix}.$$

From the relation (1.29) we obtain

$$\mathcal{C}(\mathbf{a}) = F\mathrm{Diag}(\Omega\mathbf{a})F^H. \tag{1.30}$$

Now, let $\rho = e^{-\mathbf{i}\frac{\pi}{n}}$, let $D_\rho = \mathrm{Diag}\left((\rho^i)\right)$, $i = 0, \ldots, n-1$, and consider the matrix

$$S = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & & \ddots & \ddots & 1 \\ -1 & 0 & \cdots & 0 & 0 \end{pmatrix}. \tag{1.31}$$

Since $S = \rho D_\rho C D_\rho^H$, from equality (1.29) we get

$$S = D_\rho F \rho D_\omega F^H D_\rho^H, \tag{1.32}$$

so that S is diagonalizable and has n distinct eigenvalues. The algebra $P(S) = D(D_\rho F) = Z(S)$, that we will denote with $\mathcal{S}$, is the algebra of the skewcirculant matrices. As before, we set $\mathcal{S}(\mathbf{a}) = \sum_{k=0}^{n-1} a_k S^k$. For example for $n = 4$ we have

$$\mathcal{S}(\mathbf{a}) = \begin{pmatrix} a_0 & a_1 & a_2 & a_3 \\ -a_3 & a_0 & a_1 & a_2 \\ -a_2 & -a_3 & a_0 & a_1 \\ -a_1 & -a_2 & -a_3 & a_0 \end{pmatrix}.$$

From relation (1.32) we obtain

$$\mathcal{S}(\mathbf{a}) = D_\rho F\mathrm{Diag}(\Omega D_\rho \mathbf{a})F^H D_\rho^H. \tag{1.33}$$

Remark 1.31 Let $\mathcal{T}$ be the linear space of the Toeplitz matrices. Then we have

$$\mathcal{T} = \mathcal{C} + \mathcal{S}. \tag{1.34}$$

Actually, the inclusion $\mathcal{C} + \mathcal{S} \subseteq \mathcal{T}$ is obvious. Moreover, we have $\dim \mathcal{T} = 2n - 1$ and

$$\dim(\mathcal{C} + \mathcal{S}) = \dim \mathcal{C} + \dim \mathcal{S} - \dim(\mathcal{C} \cap \mathcal{S}) = 2n - \dim(\mathcal{C} \cap \mathcal{S}).$$

From $\mathcal{C} \cap \mathcal{S} = \{aI \,|\, a \in \mathsf{C}\}$ it follows (1.34). See also [49].

Among the matrices F, D_ω, C, D_ρ and S and the matrices J, K (see section 1.1.1) and

$$K_- = JS = \begin{pmatrix} -1 & 0 & \cdots & \cdots & 0 \\ 0 & & & \iddots & 1 \\ \vdots & & \iddots & \iddots & 0 \\ \vdots & \iddots & \iddots & \iddots & \vdots \\ 0 & 1 & 0 & \cdots & 0 \end{pmatrix}$$

there are deep relations as the following proposition points out (see also [29, 19, 6]).

Proposition 1.32

(a) We have $C^H = C^{-1}$, $S^H = S^{-1}$, $J^2 = K^2 = K_-^2 = I$. Moreover $KC = C^H K$ and $K_- S = S^H K_-$.

(b) We have $KF = F^H$, $K = F^2$ and $KF = FK$.

(c) We have $K_- D_\rho F = -D_\rho^H F^H$.

(d) We have $J = F D_\omega F$.

Proof. The equalities listed in (a) are immediate. The equalities $KF = F^H$, $K = F^2$ and $KF = FK$ can be established by an easy direct computation. In order to prove (c) observe that $D_\rho K_- D_\rho = -K$. Hence,

$$K_- D_\rho F = D_\rho^H D_\rho K_- D_\rho F = -D_\rho^H KF = -D_\rho^H F^H.$$

Lastly, from (1.29) we have $C = JK = F D_\omega F^H$. Thus

$$J = F D_\omega F^H K = F D_\omega F.$$

■

1.8.3 The algebra $\mathcal{C}_e$.

We define the linear space

$$\mathcal{C}_e = \mathcal{C} + K\mathcal{C} = \{C_1 + KC_2 \,|\, C_1, C_2 \in \mathcal{C}\}.$$

Lemma 1.33 *We have*

$$\dim \mathcal{C}_e = \begin{cases} 2n-2 & \text{if } n \text{ is even,} \\ 2n-1 & \text{if } n \text{ is odd.} \end{cases}$$

Proof. We have

$$\dim(\mathcal{C} + K\mathcal{C}) = \dim \mathcal{C} + \dim K\mathcal{C} - \dim(\mathcal{C} \cap K\mathcal{C}) = 2n - \dim(\mathcal{C} \cap K\mathcal{C}).$$

Now, $A \in \mathcal{C} \cap K\mathcal{C}$ iff there exist two diagonal matrices D_1 and D_2 such that $A = F D_1 F^H$ and $A = KF D_2 F^H$. Using proposition 1.32, we get that equality $F D_1 F^H = KF D_2 F^H$ holds iff $D_1 = K D_2$.

When n is even this last equality is equivalent to $D_1 = D_2 = a\,\mathbf{e}_1\mathbf{e}_1^T + b\,\mathbf{e}_{\frac{n}{2}+1}\mathbf{e}_{\frac{n}{2}+1}^T$, with $a, b \in \mathsf{C}$. Thus A must have the following form

$$A = F(a\mathbf{e}_1\mathbf{e}_1^T + b\mathbf{e}_{\frac{n}{2}+1}\mathbf{e}_{\frac{n}{2}+1}^T)F^H = \frac{a}{n}\,\mathbf{e}\mathbf{e}^T + \frac{b}{n}\,\tilde{\mathbf{e}}\tilde{\mathbf{e}}^T = \frac{a}{n}\begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & & \vdots \\ \vdots & & \ddots & \vdots \\ 1 & \cdots & \cdots & 1 \end{pmatrix} + \frac{b}{n}\begin{pmatrix} 1 & -1 & \cdots & -1 \\ -1 & 1 & & \vdots \\ \vdots & & \ddots & \vdots \\ -1 & \cdots & \cdots & 1 \end{pmatrix}.$$

In particular $\dim(\mathcal{C} \cap K\mathcal{C}) = 2$ and the thesis follows.

The case where n is odd is left to the reader. ∎

Actually, $\mathcal{C}_e$ is an algebra, as the following theorem states.

Theorem 1.34 *For the matrix*

$$C_e = C + C^H = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 & 1 \\ 1 & 0 & 1 & \ddots & & 0 \\ 0 & 1 & 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & & \ddots & \ddots & \ddots & 1 \\ 1 & 0 & \cdots & 0 & 1 & 0 \end{pmatrix}$$

we have $\mathcal{C}_e = Z(C_e)$.

Proof. We proceed through four steps.

1. We have $\mathcal{C} \subseteq Z(C_e)$. In fact let $A \in \mathcal{C}$, then $AC = CA$ and moreover $AC^H = C^H A$, since $C^H = C^{-1}$ (proposition 1.32 (a)). Adding these two equalities we find $A(C + C^H) = (C + C^H)A$, hence $A \in Z(C_e)$.

2. We have $K\mathcal{C} \subseteq Z(C_e)$. This follows from the relations $KC = C^H K$ (proposition 1.32 (a)) and $KC^H = CK$.

3. We have $\mathcal{C} + K\mathcal{C} \subseteq Z(C_e)$, as a consequence of 1 and 2.

4. We have
$$\dim Z(C_e) = \begin{cases} 2n-2 & \text{if } n \text{ is even,} \\ 2n-1 & \text{if } n \text{ is odd.} \end{cases}$$
In fact, since
$$C_e = C + C^H = FD_\omega F^H + FD_\omega^H F^H = F(D_\omega + D_\omega^H)F^H = F\mathrm{Diag}((2\cos\frac{2\pi i}{n}))F^H, \qquad (1.35)$$
we can apply corollary 1.22.

The thesis is a straightforward consequence of 3, 4 and of lemma 1.33. ∎

Remark 1.35 Using (1.35) and proposition 1.15, we get that $P(C_e)$ is the algebra of the circulant symmetric matrices.

Remark 1.36 Obviously, $\mathcal{C}_e$ contains $\mathcal{C}$. It is interesting to note that $\mathcal{C}_e$ contains also the algebra related to the Hartley transform introduced in [19].

Some interesting questions arise. Given two vectors $\mathbf{u}$, $\mathbf{v} \in \mathsf{C}^n$, does there exist a matrix $A \in \mathcal{C}_e$ having $\mathbf{u}$ and $\mathbf{v}$ as first and last row respectively? Is it unique? How can we find two vectors $\mathbf{a}$ and $\mathbf{b}$ such that $A = \mathcal{C}(\mathbf{a}) + K\mathcal{C}(\mathbf{b})$? In the following theorem we answer these questions in the case where n is even.

Theorem 1.37 *Let n be even, and let $\mathbf{u}$, $\mathbf{v} \in \mathsf{C}^n$. A matrix $A \in \mathcal{C}_e$ such that $\mathbf{e}_1^T A = \mathbf{u}^T$ and $\mathbf{e}_n^T A = \mathbf{v}^T$ there exists if and only if*

$$\begin{aligned} \mathbf{e}^T(\mathbf{v}-\mathbf{u}) &= 0, & (1.36)\\ \tilde{\mathbf{e}}^T(\mathbf{v}+\mathbf{u}) &= 0. & (1.37) \end{aligned}$$

When A exists, it is unique, and will be indicated with $\mathcal{C}_e(\mathbf{u},\mathbf{v})$. Moreover, we have $\mathcal{C}_e(\mathbf{u},\mathbf{v}) = \mathcal{C}(\mathbf{a}) + K\mathcal{C}(\mathbf{b})$, where $\mathbf{a}$ is an arbitrary solution of the linear system

$$(C - C^H)\mathbf{x} = \mathbf{v} - C^H\mathbf{u},$$

and $\mathbf{b} = \mathbf{u} - \mathbf{a}$.

Proof. Let $\mathbf{a}$, $\mathbf{b} \in \mathsf{C}^n$ and $\mathcal{C}(\mathbf{a}) + K\mathcal{C}(\mathbf{b}) \in \mathcal{C}_e$. The conditions $\mathbf{e}_1^T(\mathcal{C}(\mathbf{a}) + K\mathcal{C}(\mathbf{b})) = \mathbf{u}^T$ and $\mathbf{e}_n^T(\mathcal{C}(\mathbf{a}) + K\mathcal{C}(\mathbf{b})) = \mathbf{v}^T$ lead to the system

$$\begin{cases} \mathbf{a} + \mathbf{b} = \mathbf{u} \\ C\mathbf{a} + C^H\mathbf{b} = \mathbf{v} \end{cases},$$

or, equivalently

$$\begin{cases} \mathbf{b} = \mathbf{u} - \mathbf{a} \\ (C - C^H)\mathbf{a} = \mathbf{v} - C^H\mathbf{u} \end{cases}. \tag{1.38}$$

Clearly system (1.38) has solution iff system $(C - C^H)\mathbf{a} = \mathbf{v} - C^H\mathbf{u}$ has solution. In view of (1.29) this system becomes $F(D_\omega - D_\omega^H)F^H\mathbf{a} = \mathbf{v} - C^H\mathbf{u}$, or, more explicitly

$$(D_\omega - D_\omega^H)F^H\mathbf{a} = F^H\mathbf{v} - D_\omega^H F^H\mathbf{u}. \tag{1.39}$$

Since $\mathbf{e}_i^T(D_\omega - D_\omega^H)\mathbf{e}_i = 0$ if and only if $i = 1,\ n/2+1$, a solution for (1.39) exists if and only if

$$\begin{aligned} \mathbf{e}_1^T(F^H\mathbf{v} - D_\omega^H F^H\mathbf{u}) &= \tfrac{1}{\sqrt{n}}\mathbf{e}^T(\mathbf{v}-\mathbf{u}) = 0,\\ \mathbf{e}_{\frac{n}{2}+1}^T(F^H\mathbf{v} - D_\omega^H F^H\mathbf{u}) &= \tfrac{1}{\sqrt{n}}\tilde{\mathbf{e}}^T(\mathbf{v}+\mathbf{u}) = 0. \end{aligned}$$

This ends the proof of the first assertion in the thesis.

Now, observe that if $\mathbf{a}$ is a particular solution of the system (1.39), the general solution has the form $\mathbf{a} + a\mathbf{e} + b\tilde{\mathbf{e}}$, where a and b are arbitrary complex parameters. Thus

$$\mathcal{C}(\mathbf{a} + a\mathbf{e} + b\tilde{\mathbf{e}}) + K\mathcal{C}(\mathbf{u} - \mathbf{a} - a\mathbf{e} - b\tilde{\mathbf{e}}) = \mathcal{C}(\mathbf{a}) + K\mathcal{C}(\mathbf{u} - \mathbf{a}),$$

since $\mathcal{C}(\mathbf{e}) = K\mathcal{C}(\mathbf{e})$ and $\mathcal{C}(\tilde{\mathbf{e}}) = K\mathcal{C}(\tilde{\mathbf{e}})$ (compare with lemma 1.33). Thus we have unicity and the last part of the thesis. ∎

For the sake of completeness, let us note that in the case where n is odd only the condition (1.36) is needed in order to guarantee the existence and the unicity of A.

The previous theorem indicates a handy way to switch from $\mathcal{C}_e(\mathbf{u},\mathbf{v})$ to $\mathcal{C}(\mathbf{a}) + K\mathcal{C}(\mathbf{b})$. In fact, provided that a solution of the linear system $(C - C^H)\mathbf{x} = \mathbf{y}$ exists, we can compute it with a cost of $n-2$ additions. For this purpose, let $\mathbf{x} = (x_k) \in \mathsf{C}^n$, with $k = 1, \ldots, n$. By arbitrarily choosing x_1 (x_2) by means of a forward substitution we get the odd (even) entries of $\mathbf{x}$.

By virtue of (1.30) and of proposition 1.32 (b) we have

$$\mathcal{C}(\mathbf{a}) + K\mathcal{C}(\mathbf{b}) = F\left(\operatorname{Diag}(\Omega\mathbf{a}) + K\operatorname{Diag}(\Omega\mathbf{b})\right)F^H. \tag{1.40}$$

1.8.4 The algebra $\mathcal{S}_e$.

We define $\mathcal{S}_e$ as the linear space

$$\mathcal{S}_e = \mathcal{S} + K_-\mathcal{S} = \{S_1 + K_-S_2 \,|\, S_1, S_2 \in \mathcal{S}\}.$$

Lemma 1.38 *We have*

$$\dim \mathcal{S}_e = \begin{cases} 2n & \text{if } n \text{ is even,} \\ 2n-1 & \text{if } n \text{ is odd.} \end{cases}$$

Proof. We have

$$\dim(\mathcal{S} + K_-\mathcal{S}) = \dim \mathcal{S} + \dim K_-\mathcal{S} - \dim(\mathcal{S} \cap K_-\mathcal{S}) = 2n - \dim(\mathcal{S} \cap K_-\mathcal{S}).$$

Now, $A \in \mathcal{S} \cap K_-\mathcal{S}$ iff there exist two diagonal matrices D_1 and D_2 such that $A = D_\rho F D_1 F^H D_\rho^H$ and $A = K_- D_\rho F D_2 F^H D_\rho^H$. The equality $D_\rho F D_1 F^H D_\rho^H = K_- D_\rho F D_2 F^H D_\rho^H$ holds iff $D_\rho F D_1 = K_- D_\rho F D_2$, and in turn, by virtue of proposition 1.32 (c), this is equivalent to $D_\rho F D_1 = -D_\rho^H F^H D_2$. Hence we have $F D_\omega F D_1 = -D_2$, that is (see again proposition 1.32 (d)) $J D_1 = -D_2$.

If n is even $J D_1 = -D_2$ implies $D_1 = D_2 = 0$. Thus $\dim(\mathcal{S} \cap K_-\mathcal{S}) = 0$ and the thesis follows.

If n is odd $J D_1 = -D_2$ implies $D_1 = -D_2 = a\, \mathbf{e}_{\frac{n+1}{2}} \mathbf{e}_{\frac{n+1}{2}}^T$ where a is a complex parameter. Thus A must have the form

$$A = D_\rho F(a \mathbf{e}_{\frac{n+1}{2}} \mathbf{e}_{\frac{n+1}{2}}^T) F^H D_\rho^H = a/n\, \tilde{\mathbf{e}} \tilde{\mathbf{e}}^T = a/n \begin{pmatrix} 1 & -1 & \cdots & 1 \\ -1 & 1 & & \vdots \\ \vdots & & \ddots & \vdots \\ 1 & \cdots & \cdots & 1 \end{pmatrix}.$$

In particular $\dim(\mathcal{S} \cap K_-\mathcal{S}) = 1$ and the thesis follows as well. ∎

Actually, $\mathcal{S}_e$ is an algebra, as the following theorem states (since the proof is analogous to that of theorem 1.34, it is omitted).

Theorem 1.39 *For the matrix*

$$S_e = S + S^H = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 & -1 \\ 1 & 0 & 1 & \ddots & & 0 \\ 0 & 1 & 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & & \ddots & \ddots & \ddots & 1 \\ -1 & 0 & \cdots & 0 & 1 & 0 \end{pmatrix},$$

we have $\mathcal{S}_e = Z(S_e)$.

We close this section with a theorem having the same purpose of theorem 1.37.

Theorem 1.40 *Let n be even, and let $\mathbf{u}$, $\mathbf{v} \in \mathsf{C}^n$. Then there exists and it is unique a matrix $A \in \mathcal{S}_e$ such that $\mathbf{e}_1^T A = \mathbf{u}^T$ and $\mathbf{e}_n^T A = \mathbf{v}^T$. It will be indicated with $\mathcal{S}_e(\mathbf{u}, \mathbf{v})$. We have $\mathcal{S}_e(\mathbf{u}, \mathbf{v}) = \mathcal{S}(\mathbf{a}) + K_-\mathcal{S}(\mathbf{b})$, where $\mathbf{a}$ is the solution of the linear system*

$$(S - S^H)\mathbf{x} = -\mathbf{v} - S^H \mathbf{u},$$

and $\mathbf{b} = \mathbf{a} - \mathbf{u}$.

Proof. Let $\mathbf{a}$, $\mathbf{b} \in \mathsf{C}^n$ and $\mathcal{S}(\mathbf{a}) + K_-\mathcal{S}(\mathbf{b}) \in \mathcal{S}_e$. The conditions $\mathbf{e}_1^T(\mathcal{S}(\mathbf{a}) + K_-\mathcal{S}(\mathbf{b})) = \mathbf{u}^T$ and $\mathbf{e}_n^T(\mathcal{S}(\mathbf{a}) + K_-\mathcal{S}(\mathbf{b})) = \mathbf{v}^T$ lead to the system

$$\begin{cases} \mathbf{a} - \mathbf{b} = \mathbf{u} \\ -S\mathbf{a} + S^H \mathbf{b} = \mathbf{v} \end{cases},$$

equivalent to

$$\begin{cases} \mathbf{b} = \mathbf{a} - \mathbf{u} \\ (S - S^H)\mathbf{a} = -\mathbf{v} - S^H \mathbf{u} \end{cases}. \tag{1.41}$$

Now, using (1.32) we have

$$S - S^H = D_\rho F(\rho D_\omega - \bar{\rho} D_\omega^H) F^H D_\rho^H = D_\rho F \mathrm{Diag}((-2\mathbf{i} \sin \frac{\pi(2i+1)}{n})) F^H D_\rho^H.$$

Since we have assumed n to be even, we have that $\sin \frac{\pi(2i+1)}{n} \neq 0$, for $i = 0, \ldots, n-1$. Hence $S - S^H$ is nonsingular and this implies that the system (1.41) has unique solution. The thesis follows. ∎

For the sake of completeness, we note that in the case where n is odd we have existence and unicity of $\mathcal{S}_e(\mathbf{u}, \mathbf{v})$ iff $\tilde{\mathbf{e}}^T(\mathbf{v} - \mathbf{u}) = 0$. The proof is left to the reader.

We can easily switch from $\mathcal{S}_e(\mathbf{u}, \mathbf{v})$ to $\mathcal{S}(\mathbf{a}) + K_- \mathcal{S}(\mathbf{b})$, since a solution of the system $(S - S^H)\mathbf{x} = \mathbf{y}$ can be computed with a cost of about $2n$ additions-subtractions. In fact, let $\mathbf{x} = (x_k) \in \mathsf{C}^n$, with $k = 1, \ldots, n$. We can compute x_1 and x_n by solving the system

$$\begin{cases} \mathbf{e}^T \mathbf{y} = -2x_1 + 2x_n \\ \tilde{\mathbf{e}}^T \mathbf{y} = 2x_1 + 2x_n \end{cases},$$

(note that computing $\mathbf{e}^T\mathbf{y}$ and $\tilde{\mathbf{e}}^T\mathbf{y}$ costs about n additions). Starting with x_1 (x_n) with a forward (backward) substitution we get the odd (even) entries of $\mathbf{x}$ (altogether, this costs about n additions). By virtue of (1.33) and of proposition 1.32 (c) and (d) we have

$$\mathcal{S}(\mathbf{a}) + K_- \mathcal{S}(\mathbf{b}) = D_\rho F \left(\mathrm{Diag}(\Omega D_\rho \mathbf{a}) - J \, \mathrm{Diag}(\Omega D_\rho \mathbf{b}) \right) F^H D_\rho^H. \tag{1.42}$$

Remark 1.41 Since the matrix $S_e = S + S^H$ is symmetric, by virtue of theorem 1.39 we have that $\mathcal{S}_e(\mathbf{u}, \mathbf{v}) \in \mathcal{S}_e$ implies $\mathcal{S}_e^T(\mathbf{u}, \mathbf{v}) \in \mathcal{S}_e$. Obviously $\mathcal{S}_e^T(\mathbf{u}, \mathbf{v})\mathbf{e}_1 = \mathbf{u}$ and $\mathcal{S}_e^T(\mathbf{u}, \mathbf{v})\mathbf{e}_n = \mathbf{v}$.

Remark 1.42 In remark 1.31 we have seen that $\mathcal{T} = \mathcal{C} + \mathcal{S}$. Let $\mathcal{H}$ be the linear space of the Hankel matrices. It is well known [44] that $\mathcal{H} = J\mathcal{T}$. Now, since $K = JC$ and $K_- = JS$, we have respectively $\mathcal{C}_e = \mathcal{C} + J\mathcal{C}$ and $\mathcal{S}_e = \mathcal{S} + J\mathcal{S}$. Hence,

$$\mathcal{C}_e + \mathcal{S}_e = \mathcal{C} + J\mathcal{C} + \mathcal{S} + J\mathcal{S} = \mathcal{C} + \mathcal{S} + J(\mathcal{C} + \mathcal{S}) = \mathcal{T} + J\mathcal{T} = \mathcal{T} + \mathcal{H}.$$

Thus $\mathcal{T} + \mathcal{H}$ shares with $\mathcal{T}$ the property of being sum of two algebras.

Chapter 2

On the use of certain matrix algebras in matrix displacement decomposition.

The concept of displacement rank, introduced in [50] and thereafter studied by various authors [37, 4, 5, 12, 20, 21, 9, 38, 41, 42, 44], has important applications in the field of computations with Toeplitz and other types of dense structured matrices, both in a sequential and in a parallel environment.

The main idea is as follows. Given a matrix A we look for a linear operator $\mathcal{D}$ such that it maps A into a low rank matrix $\mathcal{D}(A)$. Using suitable techniques we can operate with $\mathcal{D}(A)$ in place of A, taking advantage of its low rank. It is important to be able to easily recover A from its image $\mathcal{D}(A)$. The formulas of Gohberg-Semencul [44], Ammar-Gader [4], Bini-Pan [20, 12] and Gohberg-Olshevsky [41] allow to recover a matrix from its image under different linear operators, in terms of a sum of products of simple structured matrices. More precisely, these formulas involve various kinds of matrices belonging to commutative algebras (Toeplitz triangulars, circulants and others) that can be efficiently multiplied by a vector using the fast Fourier transform.

In a recent paper [33], Di Fiore and Zellini study the representation of an arbitrary square matrix as sum of products of matrices belonging to Hessenberg Algebras. This way, they are able to find a unifying approach to derive all the formulas listed above. Moreover, they propose new formulas in which an arbitrary $n \times n$ matrix is expressed as sum of products of τ algebra matrices and of matrices having a τ algebra submatrix of order $n-1$ or $n-2$.

In this chapter, we extend some results in [33] obtaining new classes of decomposition formulas. As important particular case of the new formulas, we show how an arbitrary matrix can be expressed as sum of products of matrices belonging to $\tau_{\varepsilon\varphi}$ algebras (see section 1.8.1). In this way we find new inversion formulas for Toeplitz and Toeplitz-plus-Hankel matrices.

The formulas here presented fit naturally in the framework that Di Fiore and Zellini set and have the computational advantage that all the transforms to be computed have the same size, say n. Actually, fast transform algorithms achieve their best efficiency when n is a power of two, and the need of computing transforms of sizes n and $n-1$ or n and $n-2$ may be a serious drawback, as noted in [54]. In addition, in the particular case of an $n \times n$ real symmetric Toeplitz matrix T whose $(n-1)$-order principal minor is nonzero (for example a real positive definite Toeplitz matrix), we are able to exploit our formulas in order to compute the product of T^{-1} by a vector with 8 real fast Fourier transforms of order n, thus improving both [33] and [4], and matching the best result known so far, given in [5].

In this chapter, we also suggest a decomposition formula in which the algebras $\mathcal{C}_e$ and $\mathcal{S}_e$ are exploited (see sections 1.8.3 and 1.8.4). Remember that these algebras have dimension about $2n$ and are noncommutative, while all the algebras mentioned above have dimension n and are commutative. We are able to exploit the higher dimension of $\mathcal{C}_e$ and $\mathcal{S}_e$ in order to reduce, with respect to other decompositions, the number of matrix products that have to be added for representing certain matrices, in particular Toeplitz-plus-Hankel like matrices. Consider the linear operator $\mathcal{D}_X(A) = AX - XA$ and the

matrix T_{00} (see section 1.8.1). A matrix A has been defined in [20] to be Toeplitz-plus-Hankel like iff the rank of $\mathcal{D}_{T_{00}}(A)$ is bounded by a constant independent of the dimension of the matrix. This class includes Toeplitz-plus-Hankel matrices as well as their inverses, see [20].

The most important results obtained in this chapter have been collected in the papers [24] and [23].

The chapter is organized as follows. In section 2.1 we discuss the important concept of "orthogonality condition" concerning the linear operators of the form $\mathcal{D}_X(A) = AX - XA$. This concept will be used throughout sections 2.2 and 2.3, where decomposition formulas involving Hessenberg algebras and the algebras $\mathcal{C}_e$ and $\mathcal{S}_e$ respectively, are established. After some preliminaries contained in section 2.4, in section 2.5 we present some inversion formulas for Toeplitz and Toeplitz plus Hankel matrices discussing their computational implications in section 2.6. In these two sections we rely on the formulas involving Hessenberg algebras. Further applications to Toeplitz-plus-Hankel like matrices are presented in section 2.7, where we rely on the formula involving the algebras $\mathcal{C}_e$ and $\mathcal{S}_e$.

2.1 Displacement operators and orthogonality conditions.

Given an operator $\mathcal{L} : \mathsf{C}^{n\times n} \to \mathsf{C}^{n\times n}$ and given $A \in \mathsf{C}^{n\times n}$, we will call $\mathcal{L}$-rank of A the rank of the matrix $\mathcal{L}(A)$.

In 1979 Kailath, together with other authors, introduced and studied the linear operator $\mathcal{K} : \mathsf{C}^{n\times n} \to \mathsf{C}^{n\times n}$, defined as

$$\mathcal{K}(A) = A - ZAZ^T. \tag{2.1}$$

The operator $\mathcal{K}$, and other related operators, according to their form, were called *displacement operators*. In particular the $\mathcal{K}$-rank of A was called *displacement rank* of A. This terminology was extended to the linear operators studied thereafter in the literature in the spirit of the work of Kailath.

The main motivation for studying displacement operators is the extension of effective techniques for Toeplitz matrix computations to more general classes of dense structured matrices. We will consider the operator $\mathcal{K}$ in order to clarify this point. Concerning $\mathcal{K}$ we can show the following basic result.

Theorem 2.1 *The operator $\mathcal{K}$ is nonsingular. Moreover if $A \in \mathsf{C}^{n\times n}$ and if the $\mathcal{K}$-rank of A is not greater than d, so that $\mathcal{K}(A) = \sum_{i=0}^{d} \mathbf{x}_i\mathbf{y}_i^T$ for some $\mathbf{x}_i$,$\mathbf{y}_i \in \mathsf{C}^n$, then we have*

$$A = \sum_{i=0}^{d} L(\mathbf{x}_i)L(\mathbf{y}_i)^T, \tag{2.2}$$

where $L(\mathbf{v})$ denotes the lower triangular Toeplitz matrix having the vector $\mathbf{v}$ as first column.

Proof. In order to prove the nonsingularity of $\mathcal{K}$ it is sufficient to show that the only matrix in its nullspace is the zero matrix. Suppose that a given matrix A is such that $\mathcal{K}(A) = O$. This implies

$$\begin{aligned} A &= ZAZ^T, \\ ZAZ^T &= Z^2A(Z^2)^T, \\ &\vdots \\ Z^{n-1}A(Z^{n-1})^T &= Z^nA(Z^n)^T. \end{aligned}$$

Thus $A = Z^nA(Z^n)^T = O$ since $Z^n = O$.

Concerning the second assertion in the thesis, for the matrix in (2.2) we have

$$\begin{aligned} \mathcal{K}(A) &= A - ZAZ^T = A - Z\sum_{i=0}^{d} L(\mathbf{x}_i)L(\mathbf{y}_i)^TZ^T = \\ &= A - \sum_{i=0}^{d} L(\mathbf{x}_i)ZZ^TL(\mathbf{y}_i)^T = A - \sum_{i=0}^{d} L(\mathbf{x}_i)(I - \mathbf{e}_1\mathbf{e}_1^T)L(\mathbf{y}_i)^T = \\ &= \sum_{i=0}^{d} L(\mathbf{x}_i)\mathbf{e}_1\mathbf{e}_1^TL(\mathbf{y}_i)^T = \sum_{i=0}^{d} \mathbf{x}_i\mathbf{y}_i^T. \end{aligned}$$

The proof is completed by observing that it is impossible to find any other matrix having the same image under $\mathcal{K}$, being it a nonsingular linear operator. ∎

The previous theorem shows how it is possible to recover a matrix A from its image under $\mathcal{K}$ in terms of a sum of products of Toeplitz triangular matrices. The minimum number of products that have to be added in order to recover A is equal to the $\mathcal{K}$-rank of A. Clearly, if this rank is small, and in particular if it is independent of the matrix dimension n, formula (2.2) becomes computationally attractive in order to multiply A by a vector. Now, for Toeplitz matrices and for their inverses this rank can be shown to be not greater than two. This way, from theorem 2.1 it follows that the inverse of a Toeplitz matrix can be expressed as sum of two products of a lower triangular Toeplitz matrix by an upper one. Various formulas of this kind, which are known as Gohberg-Semencul formulas, can be found in [44]. Formula (2.2) can be seen as a generalization of a Gohberg-Semencul formula that has computational interest for matrices that are "Toeplitz like", in the sense that their $\mathcal{K}$-rank is small [37].

In the successive work on the concept of displacement rank various other displacement operators have been studied with the aim of obtaining new and more efficient *displacement decomposition formulas* (this terminology is used for the formulas that allow to recover a matrix from its image under a given displacement operator). In particular, in this thesis we will consider only operators of the form

$$\mathcal{D}_X(A) = AX - XA, \tag{2.3}$$

with $X \in \mathsf{C}^{n\times n}$. The main difference betweeen these operators and the operator $\mathcal{K}$ in (2.1) is that they are singular. Given X, the nullspace of the operator D_X is the algebra $Z(X)$ (see paragraph 1.5). Obviously, this implies that the range of D_X cannot be the whole $\mathsf{C}^{n\times n}$. Gader, in [38], studying a certain displacement operator $\mathcal{G}$, singular though not of the form (2.3), pointed out the need of characterizing the range of $\mathcal{G}$ in order to obtain a displacement decomposition formula based on it. Specifically, he found necessary and sufficient conditions on a matrix $B \in \mathsf{C}^{n\times n}$ in order that $B = \mathcal{G}(A)$ for some $A \in \mathsf{C}^{n\times n}$. He called such conditions *orthogonality conditions*.

Gader's ideas have been extended to all the operators having the form (2.3) in [33], and have been further discussed in [23], that we will follow here, adding some further consideration.

Let $A = (a_{i\,j})$ and $B = (b_{i\,j})$ be in $\mathsf{C}^{n\times n}$ and let "$\circ$" denote entrywise matrix product $A\circ B = (a_{i\,j}b_{i\,j})$. We denote with "tr" the trace operator, defined as $\mathrm{tr}(A) = \sum_{i=1}^{n} a_{i\,i}$. From the definition it follows immediately that

$$\mathrm{tr}(AB) = \mathrm{tr}(BA) \tag{2.4}$$

for every A and B in $\mathsf{C}^{n\times n}$. In addition, we have

$$\mathbf{e}^T(A \circ B)\mathbf{e} = \sum_{i,j=1}^{n} a_{i\,j}b_{i\,j} = \mathrm{tr}\,(AB^T).$$

The following lemma yields a necessary condition in order that a matrix belongs to the range of the operator $\mathcal{D}_X$.

Lemma 2.2 *If $A, B, X \in \mathsf{C}^{n\times n}$ and $XB^T = B^TX$, then we have*

$$\mathbf{e}^T(\mathcal{D}_X(A) \circ B)\mathbf{e} = \mathrm{tr}(\mathcal{D}_X(A)B^T) = 0.$$

Proof. We have

$$\begin{aligned}\mathbf{e}^T(\mathcal{D}_X(A) \circ B)\mathbf{e} = \mathrm{tr}(\mathcal{D}_X(A)B^T) &= \mathrm{tr}\,\left((AX - XA)B^T\right) = \mathrm{tr}\,(AXB^T - XAB^T) = \\ &= \mathrm{tr}\,(AB^TX - XAB^T) = \mathrm{tr}\,(AB^TX) - \mathrm{tr}\,(XAB^T) = 0.\end{aligned}$$

∎

In other words, if a matrix M belongs to the range of $\mathcal{D}_X$ then we have $\mathrm{tr}(MB^T) = 0$ for every matrix $B \in Z(X^T)$. The converse is also true.

Lemma 2.3 *Let $M, B, X \in \mathsf{C}^{n\times n}$. If*

$$\mathbf{e}^T(M \circ B)\mathbf{e} = \mathrm{tr}(MB^T) = 0$$

for every B in $Z(X^T)$ then M belongs to the range of $\mathcal{D}_X$.

Proof. The set $\mathcal{M}$ of the matrices M satisfying the hypothesis is actually a linear space and, by virtue of lemma 2.2, it contains the range of the operator $\mathcal{D}_X$. Now both this range and the space $\mathcal{M}$ have dimension $n^2 - \dim Z(X)$ so that they must be equal. ∎

By means of lemma 2.2 it is possible to prove the following result, that will be used extensively in the derivation of decomposition formulas based on displacement operators having the form $\mathcal{D}_X$.

Theorem 2.4 *Let $A, B, X \in \mathsf{C}^{n\times n}$. If $\mathcal{D}_X(A) = \sum_{m=1}^{\alpha} \mathbf{x}_m\mathbf{y}_m^T$ and $XB^T = B^TX$ then*

$$\sum_{m=1}^{\alpha} \mathbf{x}_m^T B\mathbf{y}_m = 0.$$

Proof. We have

$$\sum_{m=1}^{\alpha} \mathbf{x}_m^T B\mathbf{y}_m = \sum_{m=1}^{\alpha} \mathbf{e}^T\left((\mathbf{x}_m\mathbf{y}_m^T) \circ B\right)\mathbf{e} = \mathbf{e}^T\left(\left(\sum_{m=1}^{\alpha} \mathbf{x}_m\mathbf{y}_m^T\right) \circ B\right)\mathbf{e} = \mathbf{e}^T\left(\mathcal{D}_X(A) \circ B\right)\mathbf{e} = 0.$$

∎

2.2 Decomposition formulas involving Hessenberg algebras.

In this section we obtain decomposition formulas where a square matrix $A \in \mathsf{C}^{n\times n}$ is represented as sum of products of matrices belonging to particular Hessenberg Algebras (HA, see section 1.4). Throughout this section we set X equal to the Hessenberg matrix defined in (1.5) with $[X]_{ij} \in \mathsf{C}$ and we assume $b_i \neq 0$ for $i = 1, \ldots, n-1$.

In [33] two wide classes of decomposition formulas are obtained. In the first, an arbitrary matrix A is expressed as sum of products of Toeplitz triangular matrices and of matrices in the HA H_X, where X besides Hessenberg is taken to be Toeplitz. This class includes as particular cases various Gohberg-Semencul type formulas (one of these was shown in theorem 2.1). Moreover, it includes the Bini-Pan type formulas [12, 20], and the Gader type formulas [38], where H_X is chosen to be respectively the τ algebra and the algebra of the circulant matrices. The second class of formulas involves the HA H_X and $H_{X'}$, with X' defined by $X = X' + (r_{n1} - \beta)\mathbf{e}_n\mathbf{e}_1^T$ and X persymmetric. These formulas include the Gohberg-Olshevsky type formulas exploiting ϵ-circulant matrices [41].

Here, at first we extend this second class by modifying a generic element in the secondary diagonal of X (theorem 2.5). Secondly, we apply this technique in the principal diagonal of a symmetric (theorem 2.6) or persymmetric-tridiagonal matrix X (theorems 2.7 and 2.8) obtaining new important classes of decomposition formulas. They include, as particular cases, formulas involving $\tau_{\varepsilon\varphi}$ algebras (corollaries 2.9 and 2.10). This way, we obtain new representations of the inverses of Toeplitz or Toeplitz-plus-Hankel matrices, for some aspects having better computational properties than the ones in [4, 33, 21] (see sections 2.5 and 2.6).

Set

$$X = X' + (r_{n+1-j\,j} - \beta)\mathbf{e}_{n+1-j}\mathbf{e}_j^T, \qquad j = 1, \ldots, \lfloor \tfrac{n}{2} \rfloor + 1,$$

where $\beta \in \mathsf{C}$ and, if n is even and $j = \frac{n}{2} + 1$ ($r_{\frac{n}{2}\,\frac{n}{2}+1} = b_{\frac{n}{2}}$), $\beta \neq 0$.

Theorem 2.5 *If X is persymmetric and the matrices X_j and X_j' are invertible, then the equalities $\mathcal{D}_X(A) = \sum_{m=1}^{\alpha} \mathbf{x}_m\mathbf{y}_m^T$ and $\mathcal{D}_{X^T}(A) = \sum_{m=1}^{\alpha} \mathbf{x}_m\mathbf{y}_m^T$ imply, respectively,*

$$(r_{n+1-j\,j} - \beta)A = \sum_{m=1}^{\alpha} H_{X'}(JX_j'^{-1}\mathbf{x}_m)H_X(X_j^{-T}\mathbf{y}_m) + (r_{n+1-j\,j} - \beta)H_X(X_j^{-T}A^T\mathbf{e}_j) = \quad (2.5)$$

$$= -\sum_{m=1}^{\alpha} H_X(JX_j^{-1}\mathbf{x}_m)H_{X'}(X_j'^{-T}\mathbf{y}_m) + (r_{n+1-j\,j} - \beta)H_X(JX_j^{-1}A\mathbf{e}_{n+1-j}) \quad (2.6)$$

and

$$(r_{n+1-j\,j}-\beta)A = -\sum_{m=1}^{\alpha} H_X(X_j^{-T}\mathbf{x}_m)^T H_{X'}(JX_j'^{-1}\mathbf{y}_m)^T + (r_{n+1-j\,j}-\beta)H_X(X_j^{-T}A\mathbf{e}_j)^T = \quad (2.7)$$

$$= \sum_{m=1}^{\alpha} H_{X'}(X_j'^{-T}\mathbf{x}_m)^T H_X(JX_j^{-1}\mathbf{y}_m)^T + (r_{n+1-j\,j}-\beta)H_X(JX_j^{-1}A^T\mathbf{e}_{n+1-j})^T. \quad (2.8)$$

Proof. For (2.5) using the linearity of $\mathcal{D}_X$ and the persymmetry of X and X' we have

$$\mathcal{D}_X\left(\sum_{m=1}^{\alpha} H_{X'}(JX_j'^{-1}\mathbf{x}_m)H_X(X_j^{-T}\mathbf{y}_m)\right) =$$

$$= (r_{n+1-j\,j}-\beta)\sum_{m=1}^{\alpha}\left(H_{X'}(JX_j'^{-1}\mathbf{x}_m)\mathbf{e}_{n+1-j}\mathbf{e}_j^T - \mathbf{e}_{n+1-j}\mathbf{e}_j^T H_{X'}(JX_j'^{-1}\mathbf{x}_m)\right)H_X(X_j^{-T}\mathbf{y}_m) =$$

$$= (r_{n+1-j\,j}-\beta)\left(\sum_{m=1}^{\alpha}\mathbf{x}_m\mathbf{y}_m^T - \mathbf{e}_{n+1-j}\sum_{m=1}^{\alpha}\mathbf{x}_m^T JH_X(X_j^{-T}\mathbf{y}_m)\right) = (r_{n+1-j\,j}-\beta)\sum_{m=1}^{\alpha}\mathbf{x}_m\mathbf{y}_m^T.$$

The last equality holds by virtue of the following relations which hold, for $i = 1, \ldots, n$, by theorems 1.18 and 2.4,

$$\sum_{m=1}^{\alpha}\mathbf{x}_m^T JH_X(X_j^{-T}\mathbf{y}_m)\mathbf{e}_i = \sum_{m=1}^{\alpha}\mathbf{x}_m^T J\sum_{k=1}^{n}[X_j^{-T}\mathbf{y}_m]_k X_k\mathbf{e}_i =$$

$$= \sum_{m=1}^{\alpha}\mathbf{x}_m^T JX_{n+1-i}JX_j^{-T}\mathbf{y}_m = \sum_{m=1}^{\alpha}\mathbf{x}_m^T X_{n+1-i}^T X_j^{-T}\mathbf{y}_m = 0. \quad (2.9)$$

Thus we have

$$(r_{n+1-j\,j}-\beta)A - \sum_{m=1}^{\alpha} H_{X'}(JX_j'^{-1}\mathbf{x}_m)H_X(X_j^{-T}\mathbf{y}_m) = H_X(\mathbf{z})$$

for a vector $\mathbf{z} \in \mathsf{C}^n$. Since $\mathbf{e}_j^T\sum_{m=1}^{\alpha} H_{X'}(JX_j'^{-1}\mathbf{x}_m)H_X(X_j^{-T}\mathbf{y}_m) = \mathbf{0}^T$ (see (2.9)), $\mathbf{z}$ is defined by the equality

$$(r_{n+1-j\,j}-\beta)\mathbf{e}_j^T A = \mathbf{z}^T X_j,$$

so that we obtain (2.5). As regards (2.6) proceed in a similar way. The formulas (2.7) and (2.8) follow from (2.5) and (2.6), respectively, and from the equality $\mathcal{D}_X(A^T) = -\mathcal{D}_{X^T}(A)^T$. ∎

Observe that if $j = 1$, then $X_j = X_j' = I$. Thus the case $j = 1$ [33] is the most significative.

In the next theorem assume

$$X = X' + (r_{ii}-\beta)\mathbf{e}_i\mathbf{e}_i^T, \qquad i = 1, \ldots, n.$$

Theorem 2.6 *If X is symmetric and the matrices X_i and X_i' are invertible, then the equality $\mathcal{D}_X(A) = \sum_{m=1}^{\alpha}\mathbf{x}_m\mathbf{y}_m^T$ implies*

$$(r_{ii}-\beta)A = \sum_{m=1}^{\alpha} H_{X'}(X_i'^{-1}\mathbf{x}_m)H_X(X_i^{-1}\mathbf{y}_m) + (r_{ii}-\beta)H_X(X_i^{-1}A^T\mathbf{e}_i) = \quad (2.10)$$

$$= -\sum_{m=1}^{\alpha} H_X(X_i^{-1}\mathbf{x}_m)H_{X'}(X_i'^{-1}\mathbf{y}_m) + (r_{ii}-\beta)H_X(X_i^{-1}A\mathbf{e}_i). \quad (2.11)$$

Proof. For (2.10) using the linearity of $\mathcal{D}_X$ and the symmetry of X we have

$$\mathcal{D}_X\left(\sum_{m=1}^{\alpha} H_{X'}(X_i'^{-1}\mathbf{x}_m)H_X(X_i^{-1}\mathbf{y}_m)\right) =$$

$$= (r_{ii}-\beta)\sum_{m=1}^{\alpha}\left(H_{X'}(X_i'^{-1}\mathbf{x}_m)\mathbf{e}_i\mathbf{e}_i^T - \mathbf{e}_i\mathbf{e}_i^T H_{X'}(X_i'^{-1}\mathbf{x}_m)\right)H_X(X_i^{-1}\mathbf{y}_m) =$$

$$= (r_{ii}-\beta)\left(\sum_{m=1}^{\alpha}\mathbf{x}_m\mathbf{y}_m^T - \mathbf{e}_i\sum_{m=1}^{\alpha}\mathbf{x}_m^T H_X(X_i^{-1}\mathbf{y}_m)\right) = (r_{ii}-\beta)\sum_{m=1}^{\alpha}\mathbf{x}_m\mathbf{y}_m^T.$$

The last equality holds by virtue of the following relations which hold, for $j = 1, \ldots, n$, by theorems 1.18 and 2.4

$$\sum_{m=1}^{\alpha}\mathbf{x}_m^T H_X(X_i^{-1}\mathbf{y}_m)\mathbf{e}_j = \sum_{m=1}^{\alpha}\mathbf{x}_m^T\sum_{k=1}^{n}[X_i^{-1}\mathbf{y}_m]_k X_k\mathbf{e}_j = \sum_{m=1}^{\alpha}\mathbf{x}_m^T X_j X_i^{-1}\mathbf{y}_m = 0. \tag{2.12}$$

Thus, we have

$$(r_{ii}-\beta)A - \sum_{m=1}^{\alpha}H_{X'}(X_i'^{-1}\mathbf{x}_m)H_X(X_i^{-1}\mathbf{y}_m) = H_X(\mathbf{z})$$

for a vector $\mathbf{z} \in \mathsf{C}^n$. Since $\mathbf{e}_i^T\sum_{m=1}^{\alpha}H_{X'}(X_i'^{-1}\mathbf{x}_m)H_X(X_i^{-1}\mathbf{y}_m) = \mathbf{0}^T$ (see (2.12)), the vector $\mathbf{z}$ is defined by the equality

$$(r_{ii}-\beta)\mathbf{e}_i^T A = \mathbf{z}^T X_i$$

so that we obtain (2.10). The formula (2.11) follows from (2.10) and from the equality $\mathcal{D}_X(A^T) = -\mathcal{D}_X(A)^T$. ∎

Observe that in the cases $i = 1$ and $i = n$ the assumptions on X_i and X_i' are satisfied.

In the next two theorems assume

$$X = X' + (r_{11}-\beta)(\mathbf{e}_1\mathbf{e}_1^T + \mathbf{e}_n\mathbf{e}_n^T).$$

Theorem 2.7 *If X is persymmetric and tridiagonal and $r_{i+1\,i} \neq 0$, for $i = 1, \ldots, n-1$ then the equalities $\mathcal{D}_X(A) = \sum_{m=1}^{\alpha}\mathbf{x}_m\mathbf{y}_m^T$ and $\mathcal{D}_{X^T}(A) = \sum_{m=1}^{\alpha}\mathbf{x}_m\mathbf{y}_m^T$ imply, respectively,*

$$(r_{11}-\beta)(A + X_n A X_n^{-1}) =$$

$$= \sum_{m=1}^{\alpha}H_{X'}(\hat{\mathbf{x}}_m)H_X(X_n^{-T}\mathbf{y}_m) + (r_{11}-\beta)H_X((A + X_n A X_n^{-1})^T\mathbf{e}_1) = \tag{2.13}$$

$$= -\sum_{m=1}^{\alpha}H_X(\hat{\mathbf{x}}_m)H_{X'}(X_n'^{-T}\mathbf{y}_m) + (r_{11}-\beta)H_X(J(A + X_n A X_n^{-1})\mathbf{e}_n) \tag{2.14}$$

and

$$(r_{11}-\beta)(A + X_n^{-T} A X_n^T) =$$

$$= -\sum_{m=1}^{\alpha}H_X(X_n^{-T}\mathbf{x}_m)^T H_{X'}(\hat{\mathbf{y}}_m)^T + (r_{11}-\beta)H_X((A + X_n^{-T} A X_n^T)\mathbf{e}_1)^T = \tag{2.15}$$

$$= \sum_{m=1}^{\alpha}H_{X'}(X_n'^{-T}\mathbf{x}_m)^T H_X(\hat{\mathbf{y}}_m)^T + (r_{11}-\beta)H_X(J(A + X_n^{-T} A X_n^T)^T\mathbf{e}_n)^T. \tag{2.16}$$

Proof. For (2.13) using the linearity of $\mathcal{D}_X$, the persymmetry of X and the equality (see theorem 1.18)

$$[X_n]_{i\,j} = [X_n']_{i\,j} = \begin{cases} (\prod_{l=1}^{i-1} b_l)^{-1}\prod_{l=n+1-i}^{n-1} r_{l+1\,l} & \text{if } i = n+1-j \\ 0 & \text{otherwise} \end{cases},$$

we have

$$\mathcal{D}_X\left(\sum_{m=1}^{\alpha}H_{X'}(\hat{\mathbf{x}}_m)H_X(X_n^{-T}\mathbf{y}_m)\right) =$$

$$
\begin{aligned}
&= (r_{11}-\beta)\sum_{m=1}^{\alpha}\left(H_{X'}(\hat{\mathbf{x}}_m)(\mathbf{e}_1\mathbf{e}_1^T+\mathbf{e}_n\mathbf{e}_n^T)-(\mathbf{e}_1\mathbf{e}_1^T+\mathbf{e}_n\mathbf{e}_n^T)H_{X'}(\hat{\mathbf{x}}_m)\right)H_X(X_n^{-T}\mathbf{y}_m)= \\
&= (r_{11}-\beta)\left(\sum_{m=1}^{\alpha}X_n'\mathbf{x}_m\mathbf{y}_m^TX_n^{-1}+\sum_{m=1}^{\alpha}\mathbf{x}_m\mathbf{y}_m^T\right. \\
&\qquad\left.-\mathbf{e}_1\sum_{m=1}^{\alpha}\mathbf{x}_m^TJH_X(X_n^{-T}\mathbf{y}_m)-\mathbf{e}_n\sum_{m=1}^{\alpha}\mathbf{x}_m^TJX_n'H_X(X_n^{-T}\mathbf{y}_m)\right)= \\
&= (r_{11}-\beta)\left(\sum_{m=1}^{\alpha}X_n'\mathbf{x}_m\mathbf{y}_m^TX_n^{-1}+\sum_{m=1}^{\alpha}\mathbf{x}_m\mathbf{y}_m^T\right)=(r_{11}-\beta)\mathcal{D}_X(A+X_nAX_n^{-1}).
\end{aligned}
$$

The last but one equality follows from (2.9) and from the following relation

$$
\sum_{m=1}^{\alpha}\mathbf{x}_m^TJX_n'H_X(X_n^{-T}\mathbf{y}_m)=\sum_{m=1}^{\alpha}\mathbf{x}_m^TJH_X(X_n^{-T}\mathbf{y}_m)X_n'=\mathbf{0}^TX_n'=\mathbf{0}^T.
$$

Now, we have

$$
(r_{11}-\beta)(A+X_nAX_n^{-1})-\sum_{m=1}^{\alpha}H_{X'}(\hat{\mathbf{x}}_m)H_X(X_n^{-T}\mathbf{y}_m)=H_X(\mathbf{z})
$$

for a vector $\mathbf{z}\in\mathsf{C}^n$. Since $\mathbf{e}_1^T\sum_{m=1}^{\alpha}H_{X'}(\hat{\mathbf{x}}_m)H_X(X_n^{-T}\mathbf{y}_m)=\mathbf{0}^T$ (see (2.9) for $j=n$), the vector $\mathbf{z}$ is defined by the equality

$$
(r_{11}-\beta)(A+X_nAX_n^{-1})=\mathbf{z}^T
$$

so that we obtain (2.13). As regards (2.14) proceed in a similar way. The formulas (2.15) and (2.16) follow from (2.13) and (2.14), respectively, and from the equality $\mathcal{D}_X(A^T)=-\mathcal{D}_{X^T}(A)^T$. ∎

If X is also symmetric we can state the following

Theorem 2.8 *If X is symmetric and persymmetric and $\mathcal{D}_X(A)=\sum_{m=1}^{\alpha}\mathbf{x}_m\mathbf{y}_m^T$, then*

$$
\begin{aligned}
(r_{11}-\beta)(A+JAJ) &= \sum_{m=1}^{\alpha}H_{X'}(\mathbf{x}_m)H_X(\mathbf{y}_m)+(r_{11}-\beta)H_X((A+JAJ)^T\mathbf{e}_1)= && (2.17)\\
&= -\sum_{m=1}^{\alpha}H_X(\mathbf{x}_m)H_{X'}(\mathbf{y}_m)+(r_{11}-\beta)H_X((A+JAJ)\mathbf{e}_1). && (2.18)
\end{aligned}
$$

Proof. Exploit theorems 1.18 and 2.7. ∎

We now obtain two corollaries of the previous theorems choosing H_X and $H_{X'}$ to be $\tau_{\varepsilon\varphi}$ algebras. From theorem 2.6, $i=1$, we have

Corollary 2.9 *If $\mathcal{D}_{T_{\varepsilon\varphi}}(A)=\sum_{m=1}^{\alpha}\mathbf{x}_m\mathbf{y}_m^T$ then*

$$
\begin{aligned}
(\varepsilon-\beta)A &= \sum_{m=1}^{\alpha}\tau_{\beta\varphi}(\mathbf{x}_m)\tau_{\varepsilon\varphi}(\mathbf{y}_m)+(\varepsilon-\beta)\tau_{\varepsilon\varphi}(A^T\mathbf{e}_1)= && (2.19)\\
&= -\sum_{m=1}^{\alpha}\tau_{\varepsilon\varphi}(\mathbf{x}_m)\tau_{\beta\varphi}(\mathbf{y}_m)+(\varepsilon-\beta)\tau_{\varepsilon\varphi}(A\mathbf{e}_1). && (2.20)
\end{aligned}
$$

An analogous corollary can be derived from theorem 2.6, $i=n$. We leave this task to the reader. From theorem 2.8 we obtain

Corollary 2.10 *If $\mathcal{D}_{T_{\varepsilon\varepsilon}}(A)=\sum_{m=1}^{\alpha}\mathbf{x}_m\mathbf{y}_m^T$ and $A=JAJ$ then*

$$
\begin{aligned}
2(\varepsilon-\beta)A &= \sum_{m=1}^{\alpha}\tau_{\beta\beta}(\mathbf{x}_m)\tau_{\varepsilon\varepsilon}(\mathbf{y}_m)+2(\varepsilon-\beta)\tau_{\varepsilon\varepsilon}(A^T\mathbf{e}_1)= \\
&= -\sum_{m=1}^{\alpha}\tau_{\varepsilon\varepsilon}(\mathbf{x}_m)\tau_{\beta\beta}(\mathbf{y}_m)+2(\varepsilon-\beta)\tau_{\varepsilon\varepsilon}(A\mathbf{e}_1).
\end{aligned}
$$

2.3 A decomposition formula involving algebras of dimension higher than n.

In this section we will express an arbitrary matrix in $\mathsf{C}^{n\times n}$ as sum of products of matrices in the algebras $\mathcal{C}_e$ and $\mathcal{S}_e$ (see section 1.8.3). We will assume n to be even. The decomposition formula that we obtain is the first in the literature that makes use of algebras of dimension higher than n. This allows us to reduce, with respect to other decompositon formulas, the number of matrix products that have to be added to represent certain matrices, in particular Toeplitz-plus-Hankel like matrices (see section 2.7).

Before proceding we need a couple of lemmas.

Lemma 2.11 *Let $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{y}_1$, $\mathbf{y}_2 \in \mathsf{C}^n$ and let*

$$A = \mathbf{x}_1\mathbf{y}_1^T + \mathbf{x}_2\mathbf{y}_2^T = (\,\mathbf{x}_1\,\mathbf{x}_2\,)\begin{pmatrix}\mathbf{y}_1^T\\ \mathbf{y}_2^T\end{pmatrix}.$$

If the matrix

$$\Delta(\mathbf{y}_1,\mathbf{y}_2) = \begin{pmatrix}\mathbf{e}^T\mathbf{y}_1 & \mathbf{e}^T\mathbf{y}_2\\ \tilde{\mathbf{e}}^T\mathbf{y}_1 & \tilde{\mathbf{e}}^T\mathbf{y}_2\end{pmatrix} = \begin{pmatrix}\mathbf{e}^T\\ \tilde{\mathbf{e}}^T\end{pmatrix}(\,\mathbf{y}_1\,\mathbf{y}_2\,)$$

is nonsingular then there exist vectors $\mathbf{z}_1$, $\mathbf{z}_2$, $\mathbf{w}_1$, $\mathbf{w}_2 \in \mathsf{C}^n$ such that

$$A = (\,\mathbf{z}_1\,\mathbf{z}_2\,)\begin{pmatrix}\mathbf{w}_1^T\\ \mathbf{w}_2^T\end{pmatrix} \tag{2.21}$$

and

$$\begin{aligned}\mathbf{e}^T(\mathbf{w}_2-\mathbf{w}_1) &= 0,\\ \tilde{\mathbf{e}}^T(\mathbf{w}_2+\mathbf{w}_1) &= 0.\end{aligned} \tag{2.22}$$

Proof. Set $M = \Delta(\mathbf{y}_1,\mathbf{y}_2)^{-1}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}$ and verify that (2.21) and (2.22) hold for the vectors

$$\begin{aligned}(\,\mathbf{z}_1\,\mathbf{z}_2\,) &= (\,\mathbf{x}_1\,\mathbf{x}_2\,)\,M^{-T},\\ (\,\mathbf{w}_1\,\mathbf{w}_2\,) &= (\,\mathbf{y}_1\,\mathbf{y}_2\,)\,M.\end{aligned}$$

∎

Let $A \in \mathsf{C}^{n\times n}$ and let $\mathcal{D}_X(A) = \sum_{m=1}^{\alpha}\mathbf{x}_m\mathbf{y}_m^T$. Obviously, it is not restrictive to assume α even so that we have

$$\mathcal{D}_X(A) = \sum_{m=1}^{\alpha/2}(\,\mathbf{x}_{2m-1}\,\mathbf{x}_{2m}\,)\begin{pmatrix}\mathbf{y}_{2m-1}^T\\ \mathbf{y}_{2m}^T\end{pmatrix}. \tag{2.23}$$

Lemma 2.12 *Let $\mathcal{D}_X(A)$ be as in (2.23). Then there exist vectors $\mathbf{z}_i$, $i = 1, \ldots, \alpha+2$, and $\mathbf{w}_i$, $i = 1, \ldots, \alpha$, such that*

$$\mathcal{D}_X(A) = \sum_{m=1}^{\alpha/2}(\,\mathbf{z}_{2m-1}\,\mathbf{z}_{2m}\,)\begin{pmatrix}\mathbf{w}_{2m-1}^T\\ \mathbf{w}_{2m}^T\end{pmatrix} + (\,\mathbf{z}_{\alpha+1}\,\mathbf{z}_{\alpha+2}\,)\begin{pmatrix}\mathbf{e}_1^T\\ \mathbf{e}_n^T\end{pmatrix}, \tag{2.24}$$

moreover, $\mathbf{e}^T(\mathbf{w}_{2m}-\mathbf{w}_{2m-1}) = 0$ and $\tilde{\mathbf{e}}^T(\mathbf{w}_{2m}+\mathbf{w}_{2m-1}) = 0$ for $m = 1, \ldots, \alpha/2$.

Proof. We can write $\mathcal{D}_X(A)$ in the form

$$\mathcal{D}_X(A) = \sum_{m=1}^{\alpha/2}(\,\mathbf{x}_{2m-1}\,\mathbf{x}_{2m}\,)\begin{pmatrix}\mathbf{y}_{2m-1}^T + a_m\mathbf{e}_1^T\\ \mathbf{y}_{2m}^T + a_m\mathbf{e}_n^T\end{pmatrix} - (\,\mathbf{x}_{\alpha+1}\,\mathbf{x}_{\alpha+2}\,)\begin{pmatrix}\mathbf{e}_1^T\\ \mathbf{e}_n^T\end{pmatrix}, \tag{2.25}$$

where $(\,\mathbf{x}_{\alpha+1}\,\mathbf{x}_{\alpha+2}\,) = \sum_{m=1}^{\alpha/2} a_m\,(\,\mathbf{x}_{2m-1}\,\mathbf{x}_{2m}\,)$. By suitably choosing the complex parameters a_m, we can force the matrices $\Delta(\mathbf{y}_{2m-1}+a_m\mathbf{e}_1, \mathbf{y}_{2m}+a_m\mathbf{e}_n)$ to be nonsingular for every $m = 1, \ldots, \alpha/2$. We get the thesis by applying, when needed, lemma 2.11 to the matrices on the right hand side of equality (2.25). ∎

Now, we state and prove our main theorem. We will use the linear operator

$$\mathcal{D}_{S_e}(A) = AS_e - S_eA,$$

whose kernel is $\mathcal{S}_e$, the relation

$$C_e = S_e + 2(\mathbf{e}_n\mathbf{e}_1^T + \mathbf{e}_1\mathbf{e}_n^T)$$

and the proof technique largely exploited in section 2.2.

Theorem 2.13 *Let n be even and $A \in \mathsf{C}^{n\times n}$. Let*

$$\mathcal{D}_{S_e}(A) = \sum_{m=1}^{\alpha} \mathbf{x}_m\mathbf{y}_m^T = \sum_{m=1}^{\alpha/2} (\,\mathbf{x}_{2m-1}\,\mathbf{x}_{2m}\,) \begin{pmatrix} \mathbf{y}_{2m-1}^T \\ \mathbf{y}_{2m}^T \end{pmatrix},$$

with

$$\begin{aligned} \mathbf{e}^T(\mathbf{y}_{2m-1} - \mathbf{y}_{2m}) &= 0, \\ \tilde{\mathbf{e}}^T(\mathbf{y}_{2m-1} + \mathbf{y}_{2m}) &= 0. \end{aligned}$$

Then

$$A = \frac{1}{2}\sum_{m=1}^{\alpha/2} \mathcal{S}_e^T(\mathbf{x}_{2m-1}, \mathbf{x}_{2m})\mathcal{C}_e(\mathbf{y}_{2m}, \mathbf{y}_{2m-1}) + \mathcal{S}_e^T(A\mathbf{e}_1, A\mathbf{e}_n). \tag{2.26}$$

Proof. We have

$$\begin{aligned} &\mathcal{D}_{S_e}\left(\frac{1}{2}\sum_{m=1}^{\alpha/2} \mathcal{S}_e^T(\mathbf{x}_{2m-1}, \mathbf{x}_{2m})\mathcal{C}_e(\mathbf{y}_{2m}, \mathbf{y}_{2m-1}) + \mathcal{S}_e^T(A\mathbf{e}_1, A\mathbf{e}_n)\right) = \\ &= \frac{1}{2}\sum_{m=1}^{\alpha/2} \left\{\mathcal{S}_e^T(\mathbf{x}_{2m-1}, \mathbf{x}_{2m})\mathcal{C}_e(\mathbf{y}_{2m}, \mathbf{y}_{2m-1})S_e - S_e\mathcal{S}_e^T(\mathbf{x}_{2m-1}, \mathbf{x}_{2m})\mathcal{C}_e(\mathbf{y}_{2m}, \mathbf{y}_{2m-1})\right\} = \\ &= \frac{1}{2}\sum_{m=1}^{\alpha/2} \mathcal{S}_e^T(\mathbf{x}_{2m-1}, \mathbf{x}_{2m}) \left\{\mathcal{C}_e(\mathbf{y}_{2m}, \mathbf{y}_{2m-1})(C_e - 2(\mathbf{e}_n\mathbf{e}_1^T + \mathbf{e}_1\mathbf{e}_n^T)) - \right. \\ &\qquad\qquad \left. (C_e - 2(\mathbf{e}_n\mathbf{e}_1^T + \mathbf{e}_1\mathbf{e}_n^T))\mathcal{C}_e(\mathbf{y}_{2m}, \mathbf{y}_{2m-1})\right\} = \\ &= \sum_{m=1}^{\alpha/2} \mathcal{S}_e^T(\mathbf{x}_{2m-1}, \mathbf{x}_{2m}) \left\{(\mathbf{e}_n\mathbf{e}_1^T + \mathbf{e}_1\mathbf{e}_n^T)\mathcal{C}_e(\mathbf{y}_{2m}, \mathbf{y}_{2m-1}) - \right. \\ &\qquad\qquad \left. \mathcal{C}_e(\mathbf{y}_{2m}, \mathbf{y}_{2m-1})(\mathbf{e}_n\mathbf{e}_1^T + \mathbf{e}_1\mathbf{e}_n^T)\right\} = \\ &= \sum_{m=1}^{\alpha/2} \left\{\mathbf{x}_{2m-1}\mathbf{y}_{2m-1}^T + \mathbf{x}_{2m}\mathbf{y}_{2m}^T - \mathcal{S}_e^T(\mathbf{x}_{2m-1}, \mathbf{x}_{2m})\mathcal{C}_e(\mathbf{y}_{2m}, \mathbf{y}_{2m-1})(\mathbf{e}_n\mathbf{e}_1^T + \mathbf{e}_1\mathbf{e}_n^T)\right\}. \end{aligned}$$

Now, we want to evaluate

$$\sum_{m=1}^{\alpha/2} \mathbf{e}_k^T\mathcal{S}_e^T(\mathbf{x}_{2m-1}, \mathbf{x}_{2m})\mathcal{C}_e(\mathbf{y}_{2m}, \mathbf{y}_{2m-1})\mathbf{e}_i,$$

for $k = 1, \ldots, n$, and $i = 1, n$.

We consider the case where $i = 1$, since the case where $i = n$ can be reduced to it by virtue of the equality $\mathcal{D}_{S_e}(AJ) = \mathcal{D}_{S_e}(A)J$.

By hypothesis we can find two vectors $\mathbf{f}_m$, $\mathbf{g}_m \in \mathsf{C}^n$ such that $\mathcal{C}_e(\mathbf{y}_{2m}, \mathbf{y}_{2m-1}) = \mathcal{C}(\mathbf{f}_m) + K\mathcal{C}(\mathbf{g}_m)$. Thus,

$$\mathcal{C}_e(\mathbf{y}_{2m}, \mathbf{y}_{2m-1})\mathbf{e}_1 = K\mathbf{f}_m + \mathbf{g}_m.$$

Moreover, by virtue of theorem 1.37, we can set

$$\begin{cases} \mathbf{f}_m = (C - C^H)^{\div}(\mathbf{y}_{2m-1} - C^H\mathbf{y}_{2m}) \\ \mathbf{g}_m = \mathbf{y}_{2m} - \mathbf{f}_m \end{cases},$$

where $(C-C^H)^{\div}$ is the Moore-Penrose pseudoinverse of $(C-C^H)$ (see for example [29]). Hence,

$K\mathbf{f}_m+\mathbf{g}_m=(K-I)(C-C^H)^{\div}\mathbf{y}_{2m-1}+(I-(K-I)(C-C^H)^{\div}C^H)\mathbf{y}_{2m}=N\mathbf{y}_{2m-1}+(I-NC^H)\mathbf{y}_{2m}$,

where $N=(K-I)(C-C^H)^{\div}$.

Now we want to evaluate $\mathbf{e}_k^T\mathcal{S}_e^T(\mathbf{x}_{2m-1},\mathbf{x}_{2m})$. By virtue of theorem 1.40, we can find two vectors $\mathbf{a}_m$, $\mathbf{b}_m\in\mathsf{C}^n$ such that $\mathcal{S}_e^T(\mathbf{x}_{2m-1},\mathbf{x}_{2m})=\mathcal{S}(\mathbf{a}_m)+K_-\mathcal{S}(\mathbf{b}_m)$, and we have

$$\mathbf{e}_k^T\mathcal{S}_e^T(\mathbf{x}_{2m-1},\mathbf{x}_{2m})=\mathbf{e}_k^T(\mathcal{S}(\mathbf{a}_m)+K_-\mathcal{S}(\mathbf{b}_m))=\mathbf{a}_m^TS^{k-1}-\mathbf{b}_m^TS^{k-1\,H}. \tag{2.27}$$

The vectors $\mathbf{a}_m$, $\mathbf{b}_m$, $\mathbf{x}_{2m-1}$ and $\mathbf{x}_{2m}$ are related by the following linear system

$$\begin{cases}\mathbf{a}_m^TJ+\mathbf{b}_m^TS^H=-\mathbf{x}_{2m-1}^TS^H\\ \mathbf{a}_m^TJ+\mathbf{b}_m^TS=\mathbf{x}_{2m}^T\end{cases}. \tag{2.28}$$

Hence,

$$\begin{cases}\mathbf{a}_m^T=-(\mathbf{x}_{2m-1}^T+\mathbf{x}_{2m}^TS^H)(S-S^H)^{-1}J\\ \mathbf{b}_m^T=(\mathbf{x}_{2m}^T+\mathbf{x}_{2m-1}^TS^H)(S-S^H)^{-1}\end{cases}.$$

Substituting these expressions in (2.27) we have

$$\begin{aligned}&\mathbf{e}_k^T\mathcal{S}_e^T(\mathbf{x}_{2m-1},\mathbf{x}_{2m})=\\&\quad=-(\mathbf{x}_{2m-1}^T+\mathbf{x}_{2m}^TS^H)(S-S^H)^{-1}JS^{k-1}-(\mathbf{x}_{2m}^T+\mathbf{x}_{2m-1}^TS^H)(S-S^H)^{-1}S^{k-1\,H}=\\&\quad=-\mathbf{x}_{2m-1}^T(S-S^H)^{-1}(JS^{k-1}+S^{k\,H})-\mathbf{x}_{2m}^T(S-S^H)^{-1}(JS^k+S^{k-1\,H})=\\&\quad=-\mathbf{x}_{2m-1}^T(S-S^H)^{-1}S^{k-1\,H}(I+K_-)J-\mathbf{x}_{2m}^T(S-S^H)^{-1}S^{k-1\,H}(I+K_-)=\\&\quad=\mathbf{x}_{2m-1}^TM_k(I+K_-)J+\mathbf{x}_{2m}^TM_k(I+K_-),\end{aligned}$$

where $M_k=-(S-S^H)^{-1}S^{k-1\,H}\in\mathcal{S}_e$.

Thus we have

$$\begin{aligned}&\sum_{m=1}^{\alpha/2}\mathbf{e}_k^T\mathcal{S}_e^T(\mathbf{x}_{2m-1},\mathbf{x}_{2m})\mathcal{C}_e(\mathbf{y}_{2m},\mathbf{y}_{2m-1})\mathbf{e}_1=\\&\qquad=\sum_{m=1}^{\alpha/2}(\mathbf{x}_{2m-1}^TM_k(I+K_-)J+\mathbf{x}_{2m}^TM_k(I+K_-))(N\mathbf{y}_{2m-1}+(I-NC^H)\mathbf{y}_{2m}).\end{aligned}$$

In order to evaluate this expression, we need the following matrix identities whose simple proof is left to the reader:

$$(I+K_-)N=O, \tag{2.29}$$
$$(I+K_-)J=(I+K_-)C^H, \tag{2.30}$$
$$(I+K_-)J(K-I)=(I+K_-)(C-C^H), \tag{2.31}$$
$$(C-C^H)(C-C^H)^{\div}=I-\tfrac{1}{n}(\mathbf{e}\mathbf{e}^T+\tilde{\mathbf{e}}\tilde{\mathbf{e}}^T). \tag{2.32}$$

Using (2.29) we have

$$\begin{aligned}&\sum_{m=1}^{\alpha/2}(\mathbf{x}_{2m-1}^TM_k(I+K_-)J+\mathbf{x}_{2m}^TM_k(I+K_-))(N\mathbf{y}_{2m-1}+(I-NC^H)\mathbf{y}_{2m})=\\&=\sum_{m=1}^{\alpha/2}\{\mathbf{x}_{2m-1}^TM_k((I+K_-)JN\mathbf{y}_{2m-1}+(I+K_-)J(I-NC^H)\mathbf{y}_{2m})+\mathbf{x}_{2m}^TM_k(I+K_-)\mathbf{y}_{2m}\}.\end{aligned}$$

Now, using (2.30), (2.31) and (2.32) we find that

$$\begin{aligned}&(I+K_-)JN\mathbf{y}_{2m-1}+(I+K_-)J(I-NC^H)\mathbf{y}_{2m}=\\&\quad=(I+K_-)J(K-I)(C-C^H)^{\div}\mathbf{y}_{2m-1}+(I+K_-)J(I-(K-I)(C-C^H)^{\div}C^H)\mathbf{y}_{2m}=\\&\quad=(I+K_-)(C-C^H)(C-C^H)^{\div}\mathbf{y}_{2m-1}+((I+K_-)C^H-(I+K_-)(C-C^H)(C-C^H)^{\div}C^H)\mathbf{y}_{2m}=\\&\quad=(I+K_-)(I-\tfrac{1}{n}(\mathbf{e}\mathbf{e}^T+\tilde{\mathbf{e}}\tilde{\mathbf{e}}^T))\mathbf{y}_{2m-1}+\tfrac{1}{n}(I+K_-)(\mathbf{e}\mathbf{e}^T+\tilde{\mathbf{e}}\tilde{\mathbf{e}}^T)C^H\mathbf{y}_{2m}=\\&\quad=(I+K_-)\mathbf{y}_{2m-1}+\tfrac{1}{n}(I+K_-)(-\mathbf{e}\mathbf{e}^T\mathbf{y}_{2m-1}+\mathbf{e}\mathbf{e}^T\mathbf{y}_{2m}-\tilde{\mathbf{e}}\tilde{\mathbf{e}}^T\mathbf{y}_{2m}-\tilde{\mathbf{e}}\tilde{\mathbf{e}}^T\mathbf{y}_{2m-1})=\\&\quad=(I+K_-)\mathbf{y}_{2m-1}.\end{aligned}$$

Thus, by virtue of theorem 2.4 we have

$$\sum_{m=1}^{\alpha/2}\{\mathbf{x}_{2m-1}^T M_k((I+K_-)JN\mathbf{y}_{2m-1}+(I+K_-)J(I-NC^H)\mathbf{y}_{2m})+\mathbf{x}_{2m}^T M_k(I+K_-)\mathbf{y}_{2m}\}=$$

$$=\sum_{m=1}^{\alpha/2}\{\mathbf{x}_{2m-1}^T M_k(I+K_-)\mathbf{y}_{2m-1}+\mathbf{x}_{2m}^T M_k(I+K_-)\mathbf{y}_{2m}\}=0. \tag{2.33}$$

Now consider the matrix

$$H=A-\frac{1}{2}\sum_{m=1}^{\alpha/2}\mathcal{S}_e^T(\mathbf{x}_{2m-1},\mathbf{x}_{2m})\mathcal{C}_e(\mathbf{y}_{2m},\mathbf{y}_{2m-1})$$

Equality (2.33) implies that H belongs to $\mathcal{S}_e$, i.e., the kernel of the operator $\mathcal{D}_{S_e}$, and that $H\mathbf{e}_i=A\mathbf{e}_i$, for $i=0$ and $i=n-1$. Hence $H=\mathcal{S}_e^T(A\mathbf{e}_0,A\mathbf{e}_{n-1})$ and formula (2.26) follows. ∎

2.4 Displacements of inverses of Toeplitz and Toeplitz-plus-Hankel matrices.

Since we want to use the formulas derived so far in order to represent the inverses of Toeplitz and Toeplitz-plus-Hankel matrices, we need to study the image of such matrices under the operators $\mathcal{D}_{T_{\mathcal{E}\varphi}}$ and $\mathcal{D}_{S_e}$.

Let $T=(t_{i-j})$ and $H=(h_{i+j-2})$, with $i,\,j=1,\,\ldots,\,n$, be a Toeplitz and a Hankel matrix. Let T and $T+H$ be nonsingular, let $S=T^{-1}$ and $W=(T+H)^{-1}$ and let $\mathbf{s}_{\cdot\,i}$, $\mathbf{s}_{i\,\cdot}$ ($\mathbf{w}_{\cdot\,i}$, $\mathbf{w}_{i\,\cdot}$) be the i-th column and the i-th row of S (W), respectively. Set

$$\begin{aligned}\mathbf{a}&=(0,\quad t_{-n+1},\quad\ldots,\quad t_{-1})^T,\\ \mathbf{b}&=(t_1,\quad\ldots,\quad t_{n-1},\quad 0)^T,\\ \mathbf{c}&=(0,\quad h_0,\quad\ldots,\quad h_{n-2})^T,\\ \mathbf{d}&=(h_n,\quad\ldots,\quad h_{2n-2},\quad 0)^T.\end{aligned}$$

If $\gamma=S\mathbf{a}$ and $\delta=S\mathbf{b}$, then we have [44]

$$\begin{aligned}\mathcal{D}_Z(S)&=SZ-ZS=\gamma\hat{\mathbf{s}}_{\cdot\,1}^T-\mathbf{s}_{\cdot\,1}\hat{\gamma}^T,\\ \mathcal{D}_{Z^T}(S)&=SZ^T-Z^TS=\delta\hat{\mathbf{s}}_{\cdot\,n}^T-\mathbf{s}_{\cdot\,n}\hat{\delta}^T,\end{aligned}$$

where $\hat{\mathbf{x}}=J\mathbf{x}$ for every $\mathbf{x}\in\mathsf{C}^n$. Consequently (see section 1.8.1) we obtain the following

Proposition 2.14 *We have*

$$\mathcal{D}_{T_{\mathcal{E}\varphi}}(S)=(\gamma-\varphi\mathbf{e}_n)\hat{\mathbf{s}}_{\cdot\,1}^T-\mathbf{s}_{\cdot\,1}(\hat{\gamma}^T-\varepsilon\mathbf{e}_1^T)+(\delta-\varepsilon\mathbf{e}_1)\hat{\mathbf{s}}_{\cdot\,n}^T-\mathbf{s}_{\cdot\,n}(\hat{\delta}^T-\varphi\mathbf{e}_n^T).$$

Proof. Exploit the equality $T_{\mathcal{E}\varphi}=Z+Z^T+\varepsilon\mathbf{e}_1\mathbf{e}_1^T+\varphi\mathbf{e}_n\mathbf{e}_n^T$. ∎

Analogously, setting

$$\begin{aligned}\mathbf{x}_1&=W(\mathbf{b}+\mathbf{c}),&\qquad\mathbf{x}_2&=W(\mathbf{a}+\mathbf{d}),\\ \mathbf{x}_3&=W^T(\hat{\mathbf{a}}+\mathbf{c}),&\qquad\mathbf{x}_4&=W^T(\hat{\mathbf{b}}+\mathbf{d}),\end{aligned}$$

yields [44]

$$\mathcal{D}_{T_{00}}(W)=\mathbf{x}_1\mathbf{w}_{1\,\cdot}^T+\mathbf{x}_2\mathbf{w}_{n\,\cdot}^T-\mathbf{w}_{\cdot\,1}\mathbf{x}_3^T-\mathbf{w}_{\cdot\,n}\mathbf{x}_4^T.$$

From this equation we easily obtain the two following propositions.

Proposition 2.15 *We have*

$$\mathcal{D}_{T_{\varepsilon\varphi}}(W) = (\mathbf{x}_1 - \varepsilon\mathbf{e}_1)\mathbf{w}_{1\,\cdot}^T + (\mathbf{x}_2 - \varphi\mathbf{e}_n)\mathbf{w}_{n\,\cdot}^T - \mathbf{w}_{\cdot\,1}(\mathbf{x}_3^T - \varepsilon\mathbf{e}_1^T) - \mathbf{w}_{\cdot\,n}(\mathbf{x}_4^T - \varphi\mathbf{e}_n^T).$$

Proposition 2.16 *We have*

$$\mathcal{D}_{S_e}(W) = (\mathbf{x}_1 + \mathbf{e}_n)\mathbf{w}_{1\,\cdot}^T + (\mathbf{x}_2 + \mathbf{e}_1)\mathbf{w}_{n\,\cdot}^T - \mathbf{w}_{\cdot\,1}(\mathbf{x}_3^T + \mathbf{e}_n^T) - \mathbf{w}_{\cdot\,n}(\mathbf{x}_4^T + \mathbf{e}_1^T).$$

2.5 Representation of the inverses of Toeplitz and Toeplitz-plus-Hankel matrices.

In this section we show how corollaries 2.9 and 2.10, together with propositions 2.14 and 2.15, can be exploited in order to represent the inverse of a Toeplitz or of a Toeplitz-plus-Hankel matrix. Further results, based on theorem 2.13 and on proposition 2.16, on the representation of the inverse of a Toeplitz-plus-Hankel matrix will be presented in section 2.7.

2.5.1 Formulas for Toeplitz inverses.

At this point the following result is immediate.

Theorem 2.17 *Let T be a nonsingular Toeplitz matrix and let $S = T^{-1}$. We have*

$$\begin{aligned}(\varepsilon - \beta)S &= \tau_{\beta\varphi}(\gamma - \varphi\mathbf{e}_n)\tau_{\varepsilon\varphi}(\hat{\mathbf{s}}_{\cdot\,1}) - \tau_{\beta\varphi}(\mathbf{s}_{\cdot\,1})\tau_{\varepsilon\varphi}(\hat{\gamma} - \varepsilon\mathbf{e}_1) \\ &\quad + \tau_{\beta\varphi}(\delta - \beta\mathbf{e}_1)\tau_{\varepsilon\varphi}(\hat{\mathbf{s}}_{\cdot\,n}) - \tau_{\beta\varphi}(\mathbf{s}_{\cdot\,n})\tau_{\varepsilon\varphi}(\hat{\delta} - \varphi\mathbf{e}_n) = \qquad (2.34)\\ &= -\tau_{\varepsilon\varphi}(\gamma - \varphi\mathbf{e}_n)\tau_{\beta\varphi}(\hat{\mathbf{s}}_{\cdot\,1}) + \tau_{\varepsilon\varphi}(\mathbf{s}_{\cdot\,1})\tau_{\beta\varphi}(\hat{\gamma} - \beta\mathbf{e}_1) \\ &\quad - \tau_{\varepsilon\varphi}(\delta - \varepsilon\mathbf{e}_1)\tau_{\beta\varphi}(\hat{\mathbf{s}}_{\cdot\,n}) + \tau_{\varepsilon\varphi}(\mathbf{s}_{\cdot\,n})\tau_{\beta\varphi}(\hat{\delta} - \varphi\mathbf{e}_n). \qquad (2.35)\end{aligned}$$

Proof. Use proposition 2.14 and corollary 2.9. ∎

Now let T be symmetric. In this case we have $\gamma = \hat{\delta}$ and $\hat{\mathbf{s}}_{n-i+1\,\cdot} = \mathbf{s}_{i\,\cdot}$ for all i. The above theorem can be readily rewritten in this particular case.

Theorem 2.18 *If T is symmetric we have*

$$\begin{aligned}(\varepsilon - \beta)S &= (\tau_{\beta\varepsilon}(\gamma - \varepsilon\mathbf{e}_n)J + \tau_{\beta\varepsilon}(\hat{\gamma} - \beta\mathbf{e}_1))\tau_{\varepsilon\varepsilon}(\mathbf{s}_{\cdot\,1}) - (\tau_{\beta\varepsilon}(\mathbf{s}_{\cdot\,1})J + \tau_{\beta\varepsilon}(\hat{\mathbf{s}}_{\cdot\,1}))\tau_{\varepsilon\varepsilon}(\gamma - \varepsilon\mathbf{e}_n) = \\ &= -\tau_{\varepsilon\varepsilon}(\gamma - \varepsilon\mathbf{e}_n)(\tau_{\beta\varepsilon}(\hat{\mathbf{s}}_{\cdot\,1}) + J\tau_{\beta\varepsilon}(\mathbf{s}_{\cdot\,1})) + \tau_{\varepsilon\varepsilon}(\mathbf{s}_{\cdot\,1})(\tau_{\beta\varepsilon}(\hat{\gamma} - \beta\mathbf{e}_1) + J\tau_{\beta\varepsilon}(\gamma - \varepsilon\mathbf{e}_n)).\end{aligned}$$

Proof. Set $\varphi = \varepsilon$ in theorem 2.17 and use the fact that $J \in \tau_{\varepsilon\varepsilon}$. ∎

However, if T is symmetric we can more conveniently exploit corollary 2.10.

Theorem 2.19 *If T is symmetric we have*

$$\begin{aligned}(\varepsilon - \beta)S &= \tau_{\beta\beta}(\hat{\gamma} - \beta\mathbf{e}_1)\tau_{\varepsilon\varepsilon}(\mathbf{s}_{\cdot\,1}) - \tau_{\beta\beta}(\mathbf{s}_{\cdot\,1})\tau_{\varepsilon\varepsilon}(\hat{\gamma} - \varepsilon\mathbf{e}_1) = \qquad (2.36)\\ &= -\tau_{\varepsilon\varepsilon}(\hat{\gamma} - \varepsilon\mathbf{e}_1)\tau_{\beta\beta}(\mathbf{s}_{\cdot\,1}) + \tau_{\varepsilon\varepsilon}(\mathbf{s}_{\cdot\,1})\tau_{\beta\beta}(\hat{\gamma} - \beta\mathbf{e}_1). \qquad (2.37)\end{aligned}$$

Remark 2.20 It is well known [44, 33] that if S is such that $s_{11} \neq 0$ then

$$\gamma = -\frac{1}{s_{11}} Z\mathbf{s}_{\cdot\,n}.$$

2.5.2 Formulas for Toeplitz-plus-Hankel inverses.

We turn now to the case of a general Toeplitz-plus-Hankel matrix.

Theorem 2.21 *We have*

$$\begin{aligned}(\varepsilon-\beta)W &= \tau_{\beta\varphi}(\mathbf{x}_1-\beta\mathbf{e}_1)\tau_{\varepsilon\varphi}(\mathbf{w}_{1\,\cdot})+\tau_{\beta\varphi}(\mathbf{x}_2-\varphi\mathbf{e}_n)\tau_{\varepsilon\varphi}(\mathbf{w}_{n\,\cdot})\\ &\quad -\tau_{\beta\varphi}(\mathbf{w}_{\cdot\,1})\tau_{\varepsilon\varphi}(\mathbf{x}_3-\varepsilon\mathbf{e}_1)-\tau_{\beta\varphi}(\mathbf{w}_{\cdot\,n})\tau_{\varepsilon\varphi}(\mathbf{x}_4-\varphi\mathbf{e}_n)= \qquad (2.38)\\ &= -\tau_{\varepsilon\varphi}(\mathbf{x}_1-\varepsilon\mathbf{e}_1)\tau_{\beta\varphi}(\mathbf{w}_{1\,\cdot})-\tau_{\varepsilon\varphi}(\mathbf{x}_2-\varphi\mathbf{e}_n)\tau_{\beta\varphi}(\mathbf{w}_{n\,\cdot})\\ &\quad +\tau_{\varepsilon\varphi}(\mathbf{w}_{\cdot\,1})\tau_{\beta\varphi}(\mathbf{x}_3-\beta\mathbf{e}_1)+\tau_{\varepsilon\varphi}(\mathbf{w}_{\cdot\,n})\tau_{\beta\varphi}(\mathbf{x}_4-\varphi\mathbf{e}_n). \qquad (2.39)\end{aligned}$$

Proof. Use proposition 2.15 and corollary 2.9. ∎

If $T=T^T$ then $T+H$ is symmetric and we have $\mathbf{w}_{i\,\cdot}=\mathbf{w}_{\cdot\,i}$, $\mathbf{x}_1=\mathbf{x}_3$ and $\mathbf{x}_2=\mathbf{x}_4$. If $JHJ=H$ then $T+H$ is persymmetric and in this case we have $\mathbf{w}_{i\,\cdot}=\hat{\mathbf{w}}_{\cdot\,n-i+1}$, $\hat{\mathbf{x}}_3=\mathbf{x}_2$ and $\hat{\mathbf{x}}_4=\mathbf{x}_1$. If both these conditions hold then $T+H$ is centrosymmetric and we can rewrite theorem 2.21 as follows.

Theorem 2.22 *If $T+H$ is centrosymmetric we have*

$$\begin{aligned}(\varepsilon-\beta)W &= (\tau_{\beta\varepsilon}(\mathbf{x}_1-\beta\mathbf{e}_1)+\tau_{\beta\varepsilon}(\hat{\mathbf{x}}_1-\varepsilon\mathbf{e}_n)J)\tau_{\varepsilon\varepsilon}(\mathbf{w}_{\cdot\,1})-(\tau_{\beta\varepsilon}(\mathbf{w}_{\cdot\,1})+\tau_{\beta\varepsilon}(\hat{\mathbf{w}}_{\cdot\,1})J)\tau_{\varepsilon\varepsilon}(\mathbf{x}_1-\varepsilon\mathbf{e}_1)=\\ &= -\tau_{\varepsilon\varepsilon}(\mathbf{x}_1-\varepsilon\mathbf{e}_1)(\tau_{\beta\varepsilon}(\mathbf{w}_{\cdot\,1})+J\tau_{\beta\varepsilon}(\hat{\mathbf{w}}_{\cdot\,1}))+\tau_{\varepsilon\varepsilon}(\mathbf{w}_{\cdot\,1})(\tau_{\beta\varepsilon}(\mathbf{x}_1-\beta\mathbf{e}_1)+J\tau_{\beta\varepsilon}(\hat{\mathbf{x}}_1-\varepsilon\mathbf{e}_n)).\end{aligned}$$

Proof. Set $\varphi=\varepsilon$ in theorem 2.21, and use the fact that $J\in\tau_{\varepsilon\varepsilon}$. ∎

However, a more convenient expression for W can be obtained by means of corollary 2.10.

Theorem 2.23 *If $T+H$ is centrosymmetric we have*

$$\begin{aligned}(\varepsilon-\beta)W &= \tau_{\beta\beta}(\mathbf{x}_1-\beta\mathbf{e}_1)\tau_{\varepsilon\varepsilon}(\mathbf{w}_{\cdot\,1})-\tau_{\beta\beta}(\mathbf{w}_{\cdot\,1})\tau_{\varepsilon\varepsilon}(\mathbf{x}_1-\varepsilon\mathbf{e}_1)= \qquad (2.40)\\ &= -\tau_{\varepsilon\varepsilon}(\mathbf{x}_1-\varepsilon\mathbf{e}_1)\tau_{\beta\beta}(\mathbf{w}_{\cdot\,1})+\tau_{\varepsilon\varepsilon}(\mathbf{w}_{\cdot\,1})\tau_{\beta\beta}(\mathbf{x}_1-\beta\mathbf{e}_1). \qquad (2.41)\end{aligned}$$

Proof. Use proposition 2.15 and corollary 2.10. ∎

2.6 Applications to the solution of real Toeplitz and Toeplitz-plus-Hankel systems of equations.

As it is well known, many algorithms for the solution of a Toeplitz system of equations $T\mathbf{z}=\mathbf{b}$ have the following two-stage structure.

1. Given T compute the information relevant to obtain a displacement representation for $S=T^{-1}$. For example, if T is symmetric and $s_{11}\neq 0$ only $\mathbf{s}_{\cdot\,1}$ has to be computed by virtue of remark 2.20.
2. Compute $\mathbf{z}=S\mathbf{b}$ exploiting the properties of the matrices involved in the displacement representation of S.

Obviously the same kind of algorithm can be used for the solution of a Toeplitz-plus-Hankel system of equations $(T+H)\mathbf{z}=\mathbf{b}$.

In this paragraph we look for an efficient implementation of stage 2 using the formulas of section 2.5 in the case of real systems of equations, making, when possible, a comparison with the literature on the subject [42, 4, 5, 33]. Other results concerning general (complex) Toeplitz-plus-Hankel systems will be presented in section 2.7.

As it is customary, we will use the number of fast Fourier transforms that have to be executed as a measure of cost for our algorithms. More precisely with $\mathrm{FFT}_{\mathsf{R}}(n)$ we denote the cost the FFT of a real vector of order n. Remember that $\mathrm{FFT}_{\mathsf{R}}(2n)\simeq 2\mathrm{FFT}_{\mathsf{R}}(n)$. Moreover, the computations of the

transforms $\mathbf{x} \to M_{11}^T\mathbf{x}$, $\mathbf{x} \to M_{11}\mathbf{x}$, $\mathbf{x} \to M_{-11}^T\mathbf{x}$ and $\mathbf{x} \to M_{-11}\mathbf{x}$ where $\mathbf{x}$ is a real vector, have a cost $\simeq \mathrm{FFT}_{\mathsf{R}}(n)$ as well [69, 70, 47].

It is worthwhile to distinguish between computations that involve the vector $\mathbf{b}$ and computations that involve only elements of S and thus can be embodied in the *preconditioning* stage [42], if any is performed. So we use the notation $x\mathrm{FFT}_{\mathsf{R}}(n) + y\mathrm{FFT}_{\mathsf{R}}(n)$ in order to indicate a cost of $x + y$ real FFT's, y of which can be performed only once if the system has to be solved for many differents known terms vectors.

2.6.1 Toeplitz and Toeplitz-plus-Hankel general systems.

By means of the formulas (2.34-2.35) and (2.38-2.39) it is possible to compute $S\mathbf{b}$ or $W\mathbf{b}$ with a cost of $10\mathrm{FFT}_{\mathsf{R}}(n) + 8\mathrm{FFT}_{\mathsf{R}}(n) + O(n)$ arithmetic operations. To prove this, consider for example formula (2.39) and set $\varepsilon = \varphi = 1$, $\beta = -1$. By virtue of proposition 1.30, this implies

$$\begin{aligned} W &= \frac{1}{2}M_{11}\left\{-\Lambda_{11}(M_{11}^T(\mathbf{x}_1 - \mathbf{e}_1))M_{11}^T M_{-11}\Lambda_{-11}(M_{-11}^T\mathbf{w}_{1\,\cdot})\right. \\ &\qquad - \Lambda_{11}(M_{11}^T(\mathbf{x}_2 - \mathbf{e}_n))M_{11}^T M_{-11}\Lambda_{-11}(M_{-11}^T\mathbf{w}_{n\,\cdot}) \\ &\quad +\Lambda_{11}(M_{11}^T\mathbf{w}_{\cdot\,1})M_{11}^T M_{-11}\Lambda_{-11}(M_{-11}^T(\mathbf{x}_3 + \mathbf{e}_1)) \\ &\qquad \left. + \Lambda_{11}(M_{11}^T\mathbf{w}_{\cdot\,n})M_{11}^T M_{-11}\Lambda_{-11}(M_{-11}^T(\mathbf{x}_4 - \mathbf{e}_n))\right\} M_{-11}^T, \end{aligned}$$

where $\Lambda_{\varepsilon\varphi}(\mathbf{x}) = \mathrm{Diag}(\mathbf{v}_{\varepsilon\varphi} \circ \mathbf{x})$ (see proposition 1.30).

2.6.2 Toeplitz symmetric and Toeplitz-plus-Hankel symmetric and persymmetric systems.

If T is symmetric or $T + H$ is symmetric and persymmetric using formulas (2.36-2.37) and (2.40-2.41) respectively it is possible to compute $S\mathbf{b}$ and $W\mathbf{b}$ with the cost of $6\mathrm{FFT}_{\mathsf{R}}(n) + 4\mathrm{FFT}_{\mathsf{R}}(n) + O(n)$. In fact, consider for example formula (2.37) and set $\varepsilon = 1$ and $\beta = -1$. Observe that, by proposition 1.30 we have

$$\tau_{-1-1}(\mathbf{x}) = M_{-1-1}\mathrm{Diag}(\mathbf{v}_{-1-1} \circ M_{-1-1}^T\mathbf{x})M_{-1-1}^T,$$

whence, using relation (1.21) we have

$$\begin{aligned} \tau_{-1-1}(\mathbf{x}) &= D_{-1}M_{11}\mathrm{Diag}(\mathbf{v}_{11} \circ M_{11}^T D_{-1}\mathbf{x})M_{11}^T D_{-1} = \\ &= D_{-1}M_{11}\Lambda_{11}(M_{11}^T D_{-1}\mathbf{x})M_{11}^T D_{-1}. \end{aligned}$$

Formula (2.37) becomes

$$\begin{aligned} S &= \frac{1}{2}M_{11}\left\{-\Lambda_{11}(M_{11}^T(\hat{\gamma} - \mathbf{e}_1))M_{11}^T D_{-1}M_{11}\Lambda_{11}(M_{11}^T D_{-1}\mathbf{s}_{\cdot\,1}) + \right. \\ &\qquad \left. \Lambda_{11}(M_{11}^T\mathbf{s}_{\cdot\,1})M_{11}^T D_{-1}M_{11}\Lambda_{11}(M_{11}^T D_{-1}(\hat{\gamma} + \mathbf{e}_1))\right\} M_{11}^T D_{-1}. \end{aligned} \tag{2.42}$$

Setting $\varepsilon = 1$ and $\beta = -1$ in (2.41) we have an analogous formula for W. It can be simply obtained from (2.42) by replacing $\hat{\gamma}$ and $\mathbf{s}_{\cdot\,1}$ with $\mathbf{x}_1$ and $\mathbf{w}_{\cdot\,1}$ respectively.

The algorithm that can be obtained from the preceding representation of S (W) has theoretically the same cost of that suggested in [33]. However, it has the practical advantage that the transforms to be computed have the same size, say n. This is particularly favourable when n is a power of 2. In the algorithm proposed in [33] it is necessary to compute Sine transforms of size n and $n - 2$, and if n is a power of two the computation of a Sine transform of order $n - 2$ may be quite more expensive with respect to that of a Sine transform of order n.

Toeplitz positive definite systems.

Let T be real and symmetric and let $s_{11} \neq 0$. Note that this includes the important special case where T is positive definite. It is possible to compute $S\mathbf{b}$ with a cost of $6\mathrm{FFT}_{\mathsf{R}}(n) + 2\mathrm{FFT}_{\mathsf{R}}(n) + O(n)$. For

this purpose, consider again the representation (2.42) and take into account that $\hat{\gamma} = -\frac{1}{s_{11}} Z^T \mathbf{s}_{\cdot\,1}$ (see remark 2.20). Thus,

$$\begin{aligned} S &= \frac{1}{2s_{11}} M_{11} \left\{ \Lambda_{11}(M_{11}^T(Z^T\mathbf{s}_{\cdot\,1} + s_{11}\mathbf{e}_1)) M_{11}^T D_{-1} M_{11} \Lambda_{11}(M_{11}^T D_{-1}\mathbf{s}_{\cdot\,1}) \right. \\ &\qquad \left. - \Lambda_{11}(M_{11}^T\mathbf{s}_{\cdot\,1}) M_{11}^T D_{-1} M_{11} \Lambda_{11}(M_{11}^T D_{-1}(Z^T\mathbf{s}_{\cdot\,1} - s_{11}\mathbf{e}_1)) \right\} M_{11}^T D_{-1}. \end{aligned}$$

Let us consider the four transforms

$$\begin{array}{ll} M_{11}^T\mathbf{s}_{\cdot\,1}, & M_{11}^T D_{-1}\mathbf{s}_{\cdot\,1}, \\ M_{11}^T Z^T\mathbf{s}_{\cdot\,1}, & M_{11}^T D_{-1} Z^T\mathbf{s}_{\cdot\,1}. \end{array} \tag{2.43}$$

We note that the following relation hold [2]

$$M_{11}^T = \sqrt{\frac{2}{n}} \mathrm{Re}\left(\mathrm{Diag}(k_i \rho_{2n}^i)(\rho_n^{ij})\right), \qquad i,\, j = 0, \ldots, n-1,$$

where $\rho_n = e^{-u_m^{(n)}\frac{\pi}{n}}$ being $u_m^{(n)}$ the complex unit. The matrix $R = (\rho_n^{ij})$, with $i,\, j = 0,\, \ldots,\, n-1$, is the $n \times n$ left upper corner of the Fourier matrix of order $2n$ defined as $F_{2n} = (\rho_{2n}^{2ij})$, for $i,\, j = 0,\, \ldots,\, 2n-1$. Observe that F_{2n} has the following form:

$$F_{2n} = \begin{pmatrix} R & D_{-1}R \\ RD_{-1} & (-1)^n D_{-1} R D_{-1} \end{pmatrix}.$$

Thus, if we compute

$$F_{2n} \begin{pmatrix} \mathbf{s}_{\cdot\,1} \\ 0 \end{pmatrix} \quad \text{and} \quad F_{2n} Z^T \begin{pmatrix} \mathbf{s}_{\cdot\,1} \\ 0 \end{pmatrix}, \tag{2.44}$$

with $O(n)$ arithmetic operations we can recover the transforms in (2.43). Moreover

$$F_{2n} Z^T = F_{2n}(P - \mathbf{e}_{2n}\mathbf{e}_1^T),$$

where $P = Z^T + \mathbf{e}_{2n}\mathbf{e}_1^T$ is the unit circulant matrix [29] and is such that $F_{2n}P = \mathrm{Diag}(\rho_n^{-i}) F_{2n}$, $i = 0,\, \ldots,\, 2n-1$. Thus

$$F_{2n} Z^T = -F_{2n}\mathbf{e}_{2n}\mathbf{e}_1^T + \mathrm{Diag}(\rho_n^{-i}) F_{2n}.$$

We deduce that once $F_{2n} \begin{pmatrix} \mathbf{s}_{\cdot\,1} \\ 0 \end{pmatrix}$ has been computed, the other transform in (2.44) can be recovered with $O(n)$ arithmetic operations.

The present result improves both the bound $6\mathrm{FFT}_{\mathsf{R}}(n) + 4\mathrm{FFT}_{\mathsf{R}}(n) + O(n)$ given in [33] and the bound $7\mathrm{FFT}_{\mathsf{R}}(n) + 2\mathrm{FFT}_{\mathsf{R}}(n) + O(n)$ given in [4], matching the best result known so far, given in [5].

2.7 Applications to Toeplitz-plus-Hankel like matrices.

Let $A \in \mathsf{C}^{n\times n}$ and let $\mathcal{D}_{S_e}(A) = \sum_{m=1}^{\alpha} \mathbf{x}_m \mathbf{y}_m^T$. Following [42, 43] we define the matrices

$$X = (\mathbf{x}_1, \cdots, \mathbf{x}_\alpha) \qquad \text{and} \qquad Y = (\mathbf{y}_1, \cdots, \mathbf{y}_\alpha)$$

to be an α-length $\mathcal{D}_{S_e}$-generator of A. In this section, we show how formula (2.26) can be exploited for the computation of the product of the matrix A, given through an α-length $\mathcal{D}_{S_e}$-generator, by a vector $\mathbf{b}$. As we can expect, the best results are obtained when α is sufficiently "low". The class of matrices having a low-length $\mathcal{D}_{S_e}$-generator is the class of the Toeplitz-plus-Hankel like matrices as it has been defined in [20]. In fact, in [20] a matrix A is defined to be Toeplitz-plus-Hankel like if the rank of $\mathcal{D}_{T_{00}}(A)$ is bounded by a constant independent of the dimension of the matrix. Since $S_e = T_{00} - \mathbf{e}_{n-1}\mathbf{e}_0^T - \mathbf{e}_0\mathbf{e}_{n-1}^T$, we have $|\mathrm{rk}\,\mathcal{D}_{T_{00}}(A) - \mathrm{rk}\,\mathcal{D}_{S_e}(A)| \le 4$. Thus we could use $\mathcal{D}_{S_e}$ in place of $\mathcal{D}_{T_{00}}$ in the definition of the class of the Toeplitz-plus-Hankel like matrices.

Using formula (2.26) we can reduce the computation of $A\mathbf{b}$ to the computation of a certain number of products by $\mathbf{b}$ of matrices in $\mathcal{S}_e$ and $\mathcal{C}_e$. Proceding as in section 2.6, we denote with $\mathrm{FFT}_{\mathsf{C}}(n)$ the cost of the fast Fourier transform of a complex vector of order n. Moreover we denote with $x\,\mathrm{FFT}_{\mathsf{C}}(n) + y\,\mathrm{FFT}_{\mathsf{C}}(n)$ a cost of $x + y$ such transforms, y of which can be executed in a preprocessing stage.

Theorem 2.24 *Let n and α be even integers, and let $A \in \mathsf{C}^{n\times n}$ be given by an α-length $\mathcal{D}_{S_e}$-generator and by its first and last columns. Then A can be multiplied by a vector with a cost of $(\alpha+3)\mathrm{FFT}_{\mathsf{C}}(n)+(2\alpha+2)\mathrm{FFT}_{\mathsf{C}}(n)+O(n)$.*

Proof. Let $X=(\mathbf{x}_1,\cdots,\mathbf{x}_\alpha)$ and $Y=(\mathbf{y}_1,\cdots,\mathbf{y}_\alpha)$ be an α-length $\mathcal{D}_{S_e}$-generator of A. We have

$$\mathcal{D}_{S_e}(A)=\sum_{m=1}^{\alpha/2}\left(\begin{array}{cc}\mathbf{x}_{2m-1} & \mathbf{x}_{2m}\end{array}\right)\left(\begin{array}{c}\mathbf{y}_{2m-1}^T\\ \mathbf{y}_{2m}^T\end{array}\right).$$

We have shown in lemma 2.12 how it is possible to compute vectors $\mathbf{z}_i$ and $\mathbf{w}_i$ such that

$$\mathcal{D}_X(A)=\sum_{m=1}^{\alpha/2}\left(\begin{array}{cc}\mathbf{z}_{2m-1} & \mathbf{z}_{2m}\end{array}\right)\left(\begin{array}{c}\mathbf{w}_{2m-1}^T\\ \mathbf{w}_{2m}^T\end{array}\right)+\left(\begin{array}{cc}\mathbf{z}_{\alpha+1} & \mathbf{z}_{\alpha+2}\end{array}\right)\left(\begin{array}{c}\mathbf{e}_1^T\\ \mathbf{e}_n^T\end{array}\right),$$

and that $\mathbf{e}^T(\mathbf{w}_{2m}-\mathbf{w}_{2m-1})=0$ and $\tilde{\mathbf{e}}^T(\mathbf{w}_{2m}+\mathbf{w}_{2m-1})=0$ for $m=1,\ldots,\alpha/2$. By means of theorem 2.13 we obtain

$$\begin{aligned}A &= \frac{1}{2}\sum_{m=1}^{\alpha/2}\mathcal{S}_e^T(\mathbf{z}_{2m-1},\mathbf{z}_{2m})\mathcal{C}_e(\mathbf{w}_{2m},\mathbf{w}_{2m-1})+\frac{1}{2}\mathcal{S}_e^T(\mathbf{z}_{\alpha+1},\mathbf{z}_{\alpha+2})J+\mathcal{S}_e^T(A\mathbf{e}_1,A\mathbf{e}_n)=\\ &= \frac{1}{2}\sum_{m=1}^{\alpha/2}\mathcal{S}_e^T(\mathbf{z}_{2m-1},\mathbf{z}_{2m})\mathcal{C}_e(\mathbf{w}_{2m},\mathbf{w}_{2m-1})+\mathcal{S}_e^T(\mathbf{p},\mathbf{q}),\end{aligned}$$

where $\mathbf{p}=A\mathbf{e}_1+\frac{1}{2}\mathbf{z}_{\alpha+2}$ and $\mathbf{q}=A\mathbf{e}_n+\frac{1}{2}\mathbf{z}_{\alpha+1}$. The matrices in $\mathcal{C}_e$ and $\mathcal{S}_e$ that appear in the previous representation can be expressed as shown in formulas (1.40) and (1.42), respectively, with a cost that amounts to $(2\alpha+2)\mathrm{FFT}_{\mathsf{C}}(n)+O(n)$ arithmetic operations. Now, let $\mathcal{C}_e(\mathbf{u},\mathbf{v})=\mathcal{C}(\mathbf{a})+J\mathcal{C}(\mathbf{b})$ and let $\Lambda(\mathbf{u},\mathbf{v})=\mathrm{Diag}(\Omega\mathbf{a})+J\mathrm{Diag}(\Omega\mathbf{b})$. Analogously, let $\mathcal{S}_e^T(\mathbf{u},\mathbf{v})=\mathcal{S}(\mathbf{c})+J\_\mathcal{S}(\mathbf{d})$ and let $\Gamma(\mathbf{u},\mathbf{v})=\mathrm{Diag}(\Omega D_\rho\mathbf{c})-K\mathrm{Diag}(\Omega D_\rho\mathbf{d})$. We have

$$\begin{aligned}A &= \frac{1}{2}\sum_{m=1}^{\alpha/2}D_\rho F\Gamma(\mathbf{z}_{2m-1},\mathbf{z}_{2m})F^HD_\rho^HF\Lambda(\mathbf{w}_{2m}\mathbf{w}_{2m-1})F^H+D_\rho F\Gamma(\mathbf{p},\mathbf{q})F^HD_\rho^H=\\ &= D_\rho F\left\{\frac{1}{2}\left(\sum_{m=1}^{\alpha/2}\Gamma(\mathbf{z}_{2m-1},\mathbf{z}_{2m})F^HD_\rho^HF\Lambda(\mathbf{w}_{2m}\mathbf{w}_{2m-1})\right)F^H+\Gamma(\mathbf{p},\mathbf{q})F^HD_\rho^H\right\}.\end{aligned}$$

Using this representation we can multiply A by a vector with $(\alpha+3)\mathrm{FFT}_{\mathsf{C}}(n)+O(n)$ arithmetic operations. ∎

It is important to point out that both Toeplitz-plus-Hankel matrices and their inverses have a 4-length $\mathcal{D}_{S_e}$-generator (see proposition 2.16). Thus we have the following result.

Corollary 2.25 *Let T, $H\in\mathsf{C}^{n\times n}$, n even, be a Toeplitz and a Hankel matrix, respectively, such that $T+H$ is nonsingular. Let $W=(T+H)^{-1}$ be given by its 4-length $\mathcal{D}_{S_e}$-generator, and by its first and last columns. Then W can be multiplied by a vector with a cost of $7\,\mathrm{FFT}_{\mathsf{C}}(n)+10\,\mathrm{FFT}_{\mathsf{C}}(n)+O(n)$.*

The present result improves the bound $15\,\mathrm{FFT}_{\mathsf{C}}(n)+12\,\mathrm{FFT}_{\mathsf{C}}(n)+O(n)$, not explicitly stated but easily obtainable from the formula given in [21, p. 20]. Moreover, it also improves the bound $10\,\mathrm{FFT}_{\mathsf{C}}(n)+8\,\mathrm{FFT}_{\mathsf{C}}(n)+O(n)$ that can be obtained re-writing one of the formulas given in [24] for matrices with complex entries.

Part II

Chapter 3

Fast discrete transforms by means of eigenpolynomials.

Given an $n \times n$ matrix $A = (a_{ij})$, with $i, j = 0, \ldots, n-1$, let us associate with the j-th column of A the *eigenpolynomial* $a_j(x) = \sum_{i=0}^{n-1} a_{ij} x^i$. Given a vector $\mathbf{b} = (b_i)$ we may look at the transform $\mathbf{b} \to A\mathbf{b}$ as to the application associating the coefficients of the polynomial

$$\sum_{j=0}^{n-1} b_j a_j(x) \tag{3.1}$$

with the vector $\mathbf{b}$. In this way, the problem of computing $\mathbf{b} \to A\mathbf{b}$ turns into the problem of computing the linear combination of eigenpolynomials (3.1).

This approach to the computation of the discrete transforms is somewhat unusual and is here examined through various examples. Specifically, we consider the cases where A is the Fourier, Hartley, Sine and Cosine matrix, i.e., the matrix defining the Discrete Fourier Transform (DFT), the Discrete Hartley Transform (DHT), the Discrete Sine Transform (DST) and the Discrete Cosine Transform (DCT), respectively. The definitions of these transform matrices are collected in section 3.2.

For each eigenpolynomial of each transform matrix we determine a compact representation in terms of a rational function. This representation seems to be intimately connected with the existence of a "fast", i.e., having cost $O(n \log n)$, algorithm for computing the transform itself.

Concerning DFT, we use the eigenpolynomials as a unifying tool for describing some important algorithms, namely the radix-2 Fast Fourier Transform (FFT), the radix-4 FFT, the split-radix FFT and Bruun's algorithm. While radix-2 FFT and radix-4 FFT are very well known in the scientific community [26, 53, 68] and popularity of split-radix FFT is increasing [34, 68], Bruun's algorithm [27] is totally unknown among nonspecialits. However, the structure of the eigenpolynomials associated with Hartley, Sine and Cosine matrices is such that an extension of Bruun's method to the computation of the respective transforms is almost immediate. In this way we have obtained new algorithms for these transforms that we have called Bruun Fast Hartley, Sine and Cosine Transforms (BFHT, BFST and BFCT, respectively). The total operation count for BFHT, BFST and BFCT is of $2n \log n + O(n)$ real operations, thus matching the costs of the best algorithms actually known for the computation of the respective transforms.

The overall structure of BFHT, BFST and BFCT is substantially the same and reflects that of Bruun's algorithm. In order to achieve a better comprehension of this structure, in the next chapter we undertake a deep analysis of Bruun's algorithm, showing how it is related to a particular factorization of the Fourier matrix.

The outline of this chapter is as follows. In section 3.1 we link the concept of eigenpolynomial, here presented, with some relevant literature. In section 3.2 we set some notations and in section 3.3 we rewrite different FFT algorithms using eigenpolynomials. The study of the structure of the eigenpolynomials associated with Hartley, Sine and Cosine matrices is made in section 3.4. In section 3.5 we deal with the extension of Bruun's algorithm to these trigonometric transforms.

In this and in the next chapter indexes of vectors and matrices run from 0 to $n-1$, unless otherwise stated.

3.1 On the concept of eigenpolynomial.

The term "eigenpolynomial" was introduced by Delsarte and Genin to denote a polynomial whose coefficients are the entries of an eigenvector of a certain matrix. In this respect, particular interest has been devoted to the study of the zeros of the eigenpolynomials of Toeplitz matrices [30, 17] due to the role that they play in the optimization of rejection filters. The definition given at the beginning of this chapter matches that of Delsarte and Genin if we observe that the columns of a transform matrix are eigenvectors of the matrices belonging to the *algebra* associated with the transform (see section 1.3). In section 1.8 we met various examples of these algebras, including the algebra of the circulant matrices [29], associated with the Fourier transform, the $\tau_{\varepsilon\varphi}$ algebras, associated with various versions of the Sine and Cosine tranforms. Recently [19] the Hartley algebra, associated with the Hartley transform has been described.

The structure of these algebras is somehow related to Toeplitz structure. For example, circulants are a subclass of Toeplitz matrices and therefore it makes sense to ask what are the zeros of their eigenpolynomials, in Delsarte-Genin sense, or, equivalently, what are the zeros of the eigenpolynomials associated with the columns of the Fourier matrix. The (very easy) answer is given in theorem 3.1. Analogous questions could be raised for the other algebras cited so far. Some of the relative eigenpolynomials are studied in section 3.4.

3.2 Transform matrices.

In this section we introduce the notation used for the Fourier, Hartley, Sine and Cosine matrices in this and in the next chapter. We met some of these matrices in chapter 1 equipped with a normalizing constant. Here and in the next chapter this constant is unnecessary.

- The Fourier matrix is defined as

$$F_n = (\omega_n^{ij}) \qquad i,\, j = 0, \ldots, n-1,$$

  where $\omega_n = e^{-\mathbf{i}\frac{2\pi}{n}} = \cos\frac{2\pi}{n} - \mathbf{i}\sin\frac{2\pi}{n}$, being $\mathbf{i}$ the complex unit.

- The Hartley matrix is defined as

$$H_n = (\cos\frac{2\pi ij}{n} + \sin\frac{2\pi ij}{n}) \qquad i,\, j = 0, \ldots, n-1.$$

  Clearly $H_n^T = H_n$. Moreover, $H_n H_n^T = H_n^2 = nI_n$ so that $\frac{1}{\sqrt{n}}H_n$ is orthogonal.

Literature provides various different versions for DST and DCT. In order to show the wide applicability of the approach based on eigenpolynomials to the problem of transform computation we consider three examples. The first two are respectively a DST and a DCT widely used in the numerical solution of certain differential equations [54, 62]. The third is a DCT of interest in image and signal processing [70, 69, 48].

- The Sine matrix is defined as

$$S_n = (\sin\frac{\pi ij}{n+1}) \qquad i,\, j = 1, \ldots, n.$$

  From the definition $S_n = S_n^T$ and easily $S_n S_n^T = S_n^2 = \frac{n+1}{2}I_n$.

- The first version of the Cosine matrix that we consider is defined as

$$C_n = (\cos\frac{\pi ij}{n}) \qquad i,\, j = 0, \ldots, n-1.$$

  Note that C_n is symmetric but not orthogonal.

- The second version of the Cosine matrix that we examine is defined as

$$T_n = (k_i \cos \frac{i(2j+1)\pi}{2n}) \qquad i,\, j = 0, \ldots, n-1,$$

where $k_0 = \frac{1}{\sqrt{2}}$ and $k_i = 1$ otherwise. It turns out that $T_n T_n^T = \frac{n}{2} I$ so $\sqrt{\frac{2}{n}}\, T_n$ is orthogonal. See [2, 34, 69, 70].

3.3 Fourier transform.

Let us begin with a simple observation.

Theorem 3.1 *Let*

$$p_{n,j}(x) = \sum_{k=0}^{n-1} \omega_n^{kj} x^k, \quad j = 0, \ldots, n-1, \tag{3.2}$$

be the eigenpolynomials associated with the columns of F_n, then

$$p_{n,j}(x) = \frac{1-x^n}{1-\omega_n^j x}. \tag{3.3}$$

Therefore the roots of $p_{n,j}(x)$ are the n^{th} roots of 1 without $\bar{\omega}_n^j = \omega_n^{n-j}$.

Proof. The thesis follows from the relation $p_{n,j}(x) = \sum_{k=0}^{n-1} (\omega_n^j x)^k = \frac{1-(\omega_n^j x)^n}{1-\omega_n^j x}$. ∎

The DFT $\mathbf{b} \to F_n \mathbf{b}$ can be reinterpreted in terms of eigenpolynomials as computing the coefficients of the polynomial $c(x) = \sum_{j=0}^{n-1} p_{n,j}(x) b_j$. In view of Theorem 3.1, we have

$$c(x) = (1-x^n) \sum_{j=0}^{n-1} \frac{b_j}{1-\omega_n^j x},$$

so that the numerator of the rational function

$$\sum_{j=0}^{n-1} \frac{b_j}{1-\omega_n^j x} \tag{3.4}$$

yields the coefficients of the sought polynomial. This way, we may get FFT algorithms by using different strategies of grouping the terms in the summation (3.4) in order to reduce the computational cost. For instance, we may compute (3.4) by grouping the terms j and $j + n/2$, $j = 0, \ldots, n/2-1$, that is,

$$c(x) = (1-x^n) \sum_{j=0}^{n/2-1} \left(\frac{b_j}{1-\omega_n^j x} + \frac{b_{n/2+j}}{1+\omega_n^j x}\right)$$

(we used the fact that $\omega_n^{j+n/2} = -\omega_n^j$), which leads to

$$c(x) = (1-x^n) \sum_{j=0}^{n/2-1} \frac{(b_j + b_{n/2+j}) + (b_j - b_{n/2+j})\omega_n^j x}{1-\omega_{n/2}^j x^2}. \tag{3.5}$$

Observe that replacing x^2 with y in (3.5), leads again to a summation of partial fractions where the numerator is a polynomial of degree 1 and the denominator has the form $1-\omega_{n/2}^j y$, $j = 0, \ldots, n/2-1$. This summation can be computed by using the same technique. This way, we arrive at the radix-2 FFT algorithm [26, 53] in its decimation in frequency (DIF) form, described in terms of partial fractions. At the general k-th recursive step of the algorithm $n/2^k$ additions of partial fractions must be performed

$k = 1, \ldots, \log n$. The results are partial fractions having as denominators $1 - \omega_{n/2^k}^j x^{2^k}$, $j = 0, \ldots, n/2^k - 1$, and as numerators polynomials of degree at most $2^k - 1$.

In this section we want to show how algorithms such as radix-4 FFT [53] and split-radix FFT [34] in their DIF form, can be easily described under this unifying framework by using the properties of the eigenpolynomials of the Fourier matrix. Moreover, with this approach we obtain in a very neat way a DIF version of Bruun's algorithm [27, 53]. In the following sections we will perform a similar analysis and we will obtain new fast algorithms for the computation of DHT, DST and DCT.

For radix-4 FFT our strategy will be to group together in (3.4) the four terms j, $n/2 + j$, $n/4 + j$ and $3/4n + j$ with $j = 0, \ldots, n/4 - 1$. Observing that $\omega_n^{n/4+j} = -\mathbf{i}\omega_n^j$ and $\omega_n^{3n/4+j} = \mathbf{i}\omega_n^j$ we obtain

$$c(x) = (1 - x^n) \sum_{j=0}^{n/4-1} \left(\frac{b_j}{1 - \omega_n^j x} + \frac{b_{n/2+j}}{1 + \omega_n^j x} + \frac{b_{n/4+j}}{1 + \mathbf{i}\omega_n^j x} + \frac{b_{3n/4+j}}{1 - \mathbf{i}\omega_n^j x}\right) = (1 - x^n) \sum_{j=0}^{n/4-1} \frac{q(x)}{1 - \omega_{n/4}^j x^4},$$

where

$$\begin{aligned} q(x) \quad = \quad & ((b_j + b_{n/2+j}) + (b_{n/4+j} + b_{3n/4+j})) + ((b_j - b_{n/2+j}) - \mathbf{i}(b_{n/4+j} - b_{3n/4+j}))\omega_n^j x + \\ & ((b_j + b_{n/2+j}) - (b_{n/4+j} + b_{3n/4+j}))\omega_n^{2j} x^2 + ((b_j - b_{n/2+j}) + \mathbf{i}(b_{n/4+j} - b_{3n/4+j}))\omega_n^{3j} x^3. \end{aligned}$$

Replacing x^4 with y we get again a sum of partial fractions and we can recursively proceed in the computation. In this case at the k-th recursive step the denominator of each partial fraction obtained is $1 - \omega_{n/4^k}^j x^{4^k}$ and the numerator is a polynomial of degree at most $4^k - 1$ where $k = 1, \ldots, (1/2)\log n$, and $j = 0, \ldots, n/4^k - 1$.

We turn now to the split-radix FFT algorithm. This algorithm [34] is a blend of radix-2 and radix-4 FFT but nevertheless has an arithmetic cost that is less than that of any radix-2^m FFT algorithm (see the tables at the end of the paragraph and [34]). We start from (3.5) and rewrite it as follows

$$c(x) = (1 - x^n) \sum_{j=0}^{n/2-1} \frac{b_j + b_{n/2+j}}{1 - \omega_{n/2}^j x^2} + (1 - x^n) \sum_{j=0}^{n/2-1} \frac{(b_j - b_{n/2+j})\omega_n^j x}{1 - \omega_{n/2}^j x^2}. \tag{3.6}$$

Now we perform another recursive step of radix-2 FFT algorithm on the second summation in the previous display and we reach a description of the first step of split-radix FFT in terms of sums of partial fractions

$$c(x) = (1 - x^n) \sum_{j=0}^{n/2-1} \frac{b_j + b_{n/2+j}}{1 - \omega_{n/2}^j x^2} + (1 - x^n) \sum_{j=0}^{n/4-1} \frac{p(x)}{1 - \omega_{n/4}^j x^4}, \tag{3.7}$$

where

$$p(x) = ((b_j - b_{n/2+j}) - \mathbf{i}(b_{n/4+j} - b_{3n/4+j}))\omega_n^j x + ((b_j - b_{n/2+j}) + \mathbf{i}(b_{n/4+j} - b_{3n/4+j}))\omega_n^{3j} x^3.$$

At this point we proceed recursively both on the first and on the second summation in (3.7).

We turn now to Bruun's algorithm [27, 53], probably less known among nonspecialists with respect to the algorithms previously discussed. In the original paper [27], Bruun presented only the decimation in time (DIT) version of the algorithm. Here, by using the structure properties of eigenpolynomials we present it and analyze the DIF version.

The key idea is to group in summation (3.4) the terms 0 and $n/2$ and the terms j and $n - j$ for $j = 1, \ldots, n/2 - 1$. Noting that $\omega_n^j = \bar{\omega}_n^{n-j}$ and that $(1 - \omega_n^j x)(1 - \bar{\omega}_n^j x) = (1 - 2x\cos\frac{2\pi j}{n} + x^2)$ the result is

$$c(x) = (1 - x^n) \left(\frac{(a_0 + a_{n/2}) + (a_0 - a_{n/2})x}{1 - x^2} + \sum_{j=1}^{n/2-1} \frac{(a_j + a_{n-j}) - (a_j\bar{\omega}_n^j + a_{n-j}\omega_n^j)x}{1 - 2x\cos\frac{2\pi j}{n} + x^2} \right). \tag{3.8}$$

At this point we perform the second step by grouping terms as we did in the first one, i.e. 0 and $n/4$ and j and $n/2 - j$ for $j = 1, \ldots, n/4 - 1$. Doing so we will obtain as numerators polynomials of

degree at most three and as denominators $1-x^4$ and $(1-\omega_n^j x)(1-\bar{\omega}_n^j x)(1-\omega_n^{n/2-j}x)(1-\bar{\omega}_n^{n/2-j}x)=1-2x^2\cos\frac{2\pi j}{n/2}+x^4$, $j=1,\ldots,n/4-1$ respectively, so that we can at this point substitute x^2 with y and proceed recursively.

Now we want to describe more formally the DIF version of Bruun's algorithm. Let $n=2^d$. The algorithm proceeds throught d steps. In the l-th step, with $l=0,\ldots,d-1$, we will calculate the numerators $\phi_{l,j}$ of 2^{d-l-1} partial fractions. For the sake of clarity we divide the algorithm in two stages. In the first one we perform the first step of the algorithm, in the second one we perform steps from 1 to $d-1$. In both stages we calculate $\phi_{l,0}$ apart from $\phi_{l,j}$ with $j=1,\ldots,2^{d-l-1}-1$, in order to stress that its calculation does not require multiplications but only additions.

Bruun's Algorithm (DIF version)

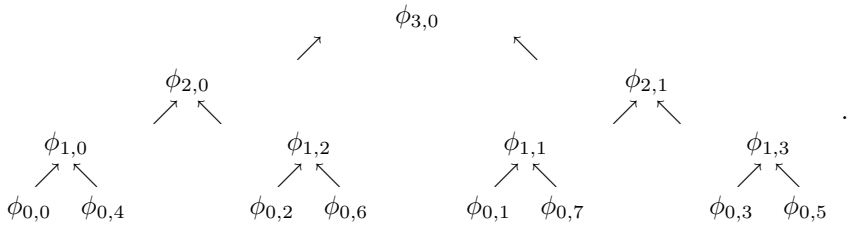
Stage 1) Compute:

$$\begin{aligned}\phi_{0,0} &= a_0(x+1)+a_{n/2}(-x+1)=x(a_0-a_{n/2})+(a_0+a_{n/2})\\ \phi_{0,j} &= a_j(-\bar{\omega}_n^j x+1)+a_{n-j}(-\omega_n^j x+1)=\\ &= -x(a_j\bar{\omega}_n^j+a_{n-j}\omega_n^j)+(a_j+a_{n-j}), \qquad j=1,\ldots,n/2-1.\end{aligned}$$

Stage 2) For $l=1,\ldots,d-1$ compute:

$$\begin{aligned}\phi_{l,0} &= \phi_{l-1,0}(x^{2^l}+1)+\phi_{l-1,2^{d-l-1}}(-x^{2^l}+1)=\\ &= x^{2^l}(\phi_{l-1,0}-\phi_{l-1,2^{d-l-1}})+(\phi_{l-1,0}+\phi_{l-1,2^{d-l-1}}),\\ \phi_{l,j} &= \phi_{l-1,j}(x^{2^l}+2x^{2^{l-1}}\cos\frac{2\pi j}{2^{d-l+1}}+1)+\phi_{l-1,2^{d-l}-j}(x^{2^l}-2x^{2^{l-1}}\cos\frac{2\pi j}{2^{d-l+1}}+1)=\\ &= x^{2^l}(\phi_{l-1,j}+\phi_{l-1,2^{d-l}-j})+x^{2^{l-1}}(\phi_{l-1,j}-\phi_{l-1,2^{d-l}-j})2\cos\frac{2\pi j}{2^{d-l+1}}+(\phi_{l-1,j}+\phi_{l-1,2^{d-l}-j}),\\ &\qquad\qquad j=1,\ldots,2^{d-l-1}-1.\end{aligned}$$

For example for $n=16$ the computation proceeds as follows



The leaves of the preceding tree are evidently not in bit reversed order. However, we can represent the permutation involved using the following recursive relation, where Σ_n denotes the matrix associated with the permutation:

$$\begin{cases}\Sigma_1=(1)\\ \Sigma_n=\Sigma_{\frac{n}{2}}\otimes\begin{pmatrix}1&0\\0&1\end{pmatrix}\left(I_{\frac{n}{2}}\otimes\begin{pmatrix}1\\0\end{pmatrix}\middle|\,K_{\frac{n}{2}}\otimes\begin{pmatrix}0\\1\end{pmatrix}\right)=\left(\Sigma_{\frac{n}{2}}\otimes\begin{pmatrix}1\\0\end{pmatrix}\middle|\,\Sigma_{\frac{n}{2}}K_{\frac{n}{2}}\otimes\begin{pmatrix}0\\1\end{pmatrix}\right).\end{cases}\tag{3.9}$$

Clearly Σ_n is not involutory. Bruun in [27] does not discuss the properties of the permutation Σ_n. In chapter 4 we will study this permutation in detail.

We observe that, by using the above DIF version of Bruun's algorithm together with its original DIT version, it is possible to skip the permutation Σ_n in the important case of computing the convolution of two vectors $\mathbf{c}_1$ and $\mathbf{c}_2$. An analogous feature is well known for DIT and DIF versions of radix-2 FFT [53, 68].

3.3.1 Operation counts.

A precise evaluation of the number of arithmetic operations performed by a transform algorithm is fundamental for theoretical reasons. From a practical viewpoint this evaluation surely gives useful

indications. However, it would be misleading to rely only on operation counts in assessing the value of a given algorithm. Bearing this in mind, we discuss the arithmetic costs of transforming a complex data sequence by means of the four algorithms presented so far.

Usually, complex arithmetic is implemented using real arithmetic. For this reason, operation counts are generally given in terms of number of real additions and multiplications performed.

In the literature on transform algorithms we are faced with two different strategies for the implementation of a complex multiplication. The first one involves the familiar identity

$$(a + \mathbf{i}b)(c + \mathbf{i}d) = ac - bd + \mathbf{i}(ad + bc), \tag{3.10}$$

so that 4 real multiplication and 2 real additions are required. We will refer to this approach as to the $4 + 2$ rule. The second [28] is more sophisticated since it involves the identity

$$(a + \mathbf{i}b)(c + \mathbf{i}d) = a(c + d) - (a + b)d + \mathbf{i}\{a(c + d) + (b - a)c\}, \tag{3.11}$$

in which the factor $(a + \mathbf{i}b)$ has to be thought in the form ω_n^m, for a suitable m (all the complex multiplications performed in radix-2, radix-4 and split-radix FFT are in this form). This means that a and b are known in advance and thus that the two additions $a + b$ and $b - a$ can be precomputed. In this way only 3 real multiplications and 3 real additions are needed, at run time, for the evaluation of a complex multiplication. We will refer to this approach as to the $3+3$ rule. It is worth noting that 6 real operations are performed both with the $4 + 2$ and the $3 + 3$ rule. This means that the $3 + 3$ rule can be useful only in a situation where the cost of a multiplication is notably higher than that of an addition.

With both the $4+2$ and the $3+3$ rule it is possible to refine the operation count taking into account that, when $a + \mathbf{i}b = e^{ik\frac{\pi}{4}}$ with k odd, only two real multiplications and two real additions are required, while no real multiplications nor additions are required if $a + \mathbf{i}b = e^{ik\frac{\pi}{2}}$.

Concerning radix-2, radix-4 and split-radix FFT various authors report precise operation counts for the transform of a general complex sequence, both using $4 + 2$ and $3 + 3$ rule. In the following tables we collect these counts [53, 34, 6].

$4 + 2$ rule	Mults	Adds
Radix-2	$n(2\log n - 7) + 12$	$n(3\log n - 3) + 4$
Radix-4	$n(3/2\log n - 5) + 8$	$n(11/4\log n - 13/6) + 8/3$
Split-Radix	$n(4/3\log n - 38/9) + 6 + (2/9)(-1)^{\log n}$	$n(8/3\log n - 16/9) + 2 - (2/9)(-1)^{\log n}$

$3 + 3$ rule	Mults	Adds
Radix-2	$n(3/2\log n - 5) + 8$	$n(7/2\log n - 5) + 8$
Radix-4	$n(9/8\log n - 43/12) + 16/3$	$n(25/8\log n - 43/12) + 16/3$
Split-Radix	$n(\log n - 3) + 4$	$n(3\log n - 3) + 4$

We stress that the total number of arithmetic operations performed by each algorithm is the same in the two tables and amounts to $5n\log n - 10n + O(1)$, $17/4n\log n - 43/6n + O(1)$ and $4n\log n - 6n + O(1)$ respectively, so that the minimum is reached by the split-radix FFT. It is possible to show that the split-radix FFT performs less arithmetic operations than every radix-2^n FFT algorithm [34].

The distinctive feature of Bruun's algorithm, in DIF version, is that complex arithmetic is performed only in the first stage. Moreover the coefficients of $-x(a_j\bar{\omega}_n^j + a_{n-j}\omega_n^j) + (a_j + a_{n-j})$ can be computed with 4 real multiplications and 6 real additions (to be precise some operations can be saved when $j = n/8$, $j = n/4$ and $j = 3n/8$). In the second stage every multiplication has a real number as one of the factors (for this reason Bruun's algorithm is a member of the family of the *real-factor* algorithms: see the next chapter). A precise operation count follows.

$$\begin{aligned}
\text{Mults} &= \left\{ \begin{array}{ll} \text{Stage 1:} & 4(n/2 - 2) - 4 \\ \text{Stage 2:} & \sum_{l=1}^{d-1}(2^{d-l-1} - 1)2^{l+1} \end{array} \right\} = n(\log n - 1) - 8, \\
\text{Adds} &= \left\{ \begin{array}{ll} \text{Stage 1:} & 6(n/2 - 2) + 8 \\ \text{Stage 2:} & \sum_{l=1}^{d-1}\{2^{l+1}2 + (2^{d-l-1} - 1)2^{l+1}3\} \end{array} \right\} = n(3\log n - 2).
\end{aligned}$$

The total number of operations performed is $4n\log n - 3n + O(1)$, $3n$ operations more than split-radix FFT.

3.3.2 Transforming complex symmetric sequences. Operation counts.

An interesting feature of the DIF version of the four algorithms presented is that if they are applied to a data sequence $\mathbf{b}$ such that $K\mathbf{b} = \bar{\mathbf{b}}$ (the DFT of a real data sequence has this property) then a saving of about one half of the arithmetic cost can be obtained. This feature is dual to that exhibited from the DIT version of the four algorithms when applied to a real data sequence. Observe that in the case of Bruun's algorithm the saving is due to the fact that the $\phi_{0,j}$, $j = 0, \ldots, n/2-1$, have real coefficients, so that only real arithmetic is required in the subsequent steps.

In the following table we display the arithmetic costs of transforming a sequence $\mathbf{b}$ with the property $K\mathbf{b} = \bar{\mathbf{b}}$ with the DIF versions of radix-2 and of split-radix FFT (these costs mach those of transforming a real data sequence using the DIT versions of the two algorithms) [59, 6].

$4+2$ rule	Mults	Adds
Radix-2	$n(\log n - 7/2) + 6$	$n(3/2\log n - 5/2) + 4$
Split-Radix	$n(2/3\log n - 17/9) + 3 + 1/9(-1)^{\log n}$	$n(4/3\log n - 17/9) + 3 - 1/9(-1)^{\log n}$

$3+3$ rule	Mults	Adds
Radix-2	$n(3/4\log n - 5/2) + 4$	$n(7/4\log n - 7/2) + 6$
Split-Radix	$n(1/2\log n - 3/2) + 2$	$n(3/2\log n - 5/2) + 4$

Thus the total number of operations is $2.5n\log n - 6n + O(1)$ and $2n\log n - 4n + O(1)$ respectively.

In the case of Bruun's algorithm we have obtained a count of $n/2(\log n - 1) - 4$ real multiplications and $n(3/2\log n - 2) + 2$ real additions (multiplications by 2 are not included in this count) giving a total of $2n\log n - 2.5n + O(1)$ operations.

In summary, a saving of more that one half of the arithmetic cost is obtained for the three algorithms. Note that, also in this setting, the split-radix FFT has the minimum operation count.

3.4 Structure of the eigenpolynomials.

The results obtained in the previous section by means of a very simple observation such as that embodied in theorem 3.1 prompt us to examine other transform matrices. In this paragraph we determine compact representations for the eigenpolynomials associated with the columns of the Hartley, Sine and Cosine matrices (definitions are in section 3.2).

We start with the Hartley matrix H_n.

Theorem 3.2 *Let*

$$h_{n,j}(x) = \sum_{k=0}^{n-1} \left(\cos\frac{2\pi kj}{n} + \sin\frac{2\pi kj}{n} \right) x^k, \quad j = 0, \ldots, n-1.$$

be the eigenpolynomials associated with the columns of H_n. Then

$$h_{n,j}(x) = (x^n - 1)\frac{\mu_j x - 1}{(x - \bar{\omega}_n^j)(x - \omega_n^j)}, \tag{3.12}$$

where $\mu_j = \cos\frac{2\pi j}{n} - \sin\frac{2\pi j}{n}$. Therefore the roots of $h_{n,j}(x)$ are the n^{th} roots of 1:

1. *without 1 if $j = 0$;*
2. *without -1 if $j = n/2$;*
3. *without ω_n^j and $\bar{\omega}_n^j$ if $j = n/8, 5n/8$;*
4. *without ω_n^j and $\bar{\omega}_n^j$ and with $1/\mu_j$ otherwise ($j \notin \{0,\ n/2,\ n/8,\ 5n/8\}$).*

Proof. From the definition we have

$$h_{n,j}(x) = 1/2(1+\mathbf{i})p_{n,j}(x) + 1/2(1-\mathbf{i})p_{n,n-j}(x),$$

and thus by using (3.3) we find

$$h_{n,j}(x) = \frac{1}{2}(x^n-1)\left(\frac{(1+\mathbf{i})\bar{\omega}_n^j}{x-\bar{\omega}_n^j} + \frac{(1-\mathbf{i})\omega_n^j}{x-\omega_n^j}\right).$$

This yields

$$h_{n,j}(x) = (x^n-1)\frac{\mu_j x - 1}{(x-\bar{\omega}_n^j)(x-\omega_n^j)}, \tag{3.13}$$

where

$$\mu_j = \cos\frac{2\pi j}{n} - \sin\frac{2\pi j}{n}.$$

■

We turn now to the case of the Sine matrix S_n. Let $\rho_n = \omega_{2n} = \exp(-\mathbf{i}\frac{\pi}{n})$, then the following theorem holds.

Theorem 3.3 *Let*

$$s_{n,j}(x) = \sum_{k=1}^{n} x^{k-1}\sin\frac{\pi kj}{n+1}, \quad j = 1,\ldots,n,$$

be the eigenpolynomials associated with the columns of S_n. Then

$$s_{n,j}(x) = \sin(\frac{\pi j}{n+1})\frac{x^{n+1}(-1)^{j+1}+1}{\left(x-\rho_{n+1}^j\right)\left(x-\bar{\rho}_{n+1}^j\right)}. \tag{3.14}$$

It follows that the roots of $s_{n,j}(x)$ are the $(n+1)$-st roots of $(-1)^j$ without ρ_{n+1}^j and $\bar{\rho}_{n+1}^j$.

Proof. Let $q_{n,j}(x) = \sum_{k=1}^{n}\rho_{n+1}^{kj}x^k$ so that

$$q_{n,j}(x) = \frac{x^{n+1}(-1)^j - 1}{x\rho_{n+1}^j - 1} - 1. \tag{3.15}$$

Denoting with $\bar{q}_{n,j}(x)$ the polynomial having conjugate coefficients with respect to $q_{n,j}(x)$ we have

$$s_{n,j}(x) = \frac{1}{2\mathbf{i}x}\left(\bar{q}_{n,j}(x) - q_{n,j}(x)\right), \tag{3.16}$$

and thus, by using (3.15)

$$s_{n,j}(x) = \frac{1}{2\mathbf{i}x}(x^{n+1}(-1)^j - 1)\frac{x\left(\rho_{n+1}^j - \bar{\rho}_{n+1}^j\right)}{\left(x\bar{\rho}_{n+1}^j - 1\right)\left(x\rho_{n+1}^j - 1\right)}.$$

The thesis follows immediately. ■

Concerning DCT we will in turn consider the matrices C_n and T_n. For C_n we have the following result, whose proof is omitted being analogous to that of theorem 3.3.

Theorem 3.4 *Let*

$$c_{n,j}(x) = \sum_{k=0}^{n-1} x^k\cos\frac{\pi kj}{n}, \quad j = 0,\ldots,n-1,$$

be the eigenpolynomials associated with the columns of C_n. Then

$$c_{n,j}(x) = (x\cos(\frac{\pi j}{n}) - 1)\frac{x^n(-1)^j - 1}{\left(x-\rho_n^j\right)\left(x-\bar{\rho}_n^j\right)}. \tag{3.17}$$

Therefore the roots of $c_{n,j}(x)$ are the n^{th} roots of $(-1)^j$:

1. *without* 1 *if* $j = 0$*;*

2. *without* $\mathbf{i}$ *and* $-\mathbf{i}$ *if* $j = n/2$*;*

3. *without* ρ_n^j *and* $\bar{\rho}_n^j$ *and with* $1/\cos\frac{\pi j}{n}$ *otherwise.*

Finally we consider the matrix T_n. In order to simplify our analysis, replace T_n with the scaled matrix D_nT_n where $D_n = \mathrm{Diag}(\sqrt{2}, 1, \ldots, 1)$.

Theorem 3.5 *Let*

$$t_{n,j}(x) = \sum_{k=0}^{n-1} x^k \cos\frac{k(2j+1)\pi}{2n}, \qquad j = 0, \ldots, n-1,$$

be the eigenpolynomials associated with the columns of D_nT_n*. We have*

$$t_{n,j}(x) = \frac{x^{n+1}(-1)^j \sin\frac{(2j+1)\pi}{2n} - x\cos\frac{(2j+1)\pi}{2n} + 1}{(x - \rho_{2n}^{2j+1})(x - \bar{\rho}_{2n}^{2j+1})}. \tag{3.18}$$

Proof. We proceed as in theorem 3.3. We put

$$a_{n,j}(x) = \sum_{k=0}^{n-1} x^k \rho_{2n}^{(2j+1)k} = \frac{x^n(-\mathbf{i})^{2j+1} - 1}{x\rho_{2n}^{2j+1} - 1}.$$

Being

$$t_{n,j}(x) = 1/2(\bar{a}_{n,j}(x) + a_{n,j}(x)),$$

the thesis follows. ∎

The matrix D_nT_n is orthogonal but not symmetric, thus, in order to compute the inverse transform, we have to consider its transposed, i.e. $T_n^T D_n^T$.

Theorem 3.6 *Let*

$$t_{n,k}^{(T)} = \sum_{j=0}^{n-1} x^j \cos\frac{k(2j+1)\pi}{2n}, \qquad k = 0, \ldots, n-1.$$

be the eigenpolynomials associated with the columns of $T_n^T D_n^T$*. Then*

$$t_{n,k}^{(T)}(x) = (x-1)\cos\frac{k\pi}{2n}\frac{x^n(-1)^k - 1}{(x - \rho_n^k)(x - \bar{\rho}_n^k)}. \tag{3.19}$$

Then the roots of $t_{n,k}^{(T)}(x)$ *are the* n^{th} *roots of* $(-1)^k$*:*

1. *without* 1 *if k=0;*

2. *without* ρ_n^k *and* $\bar{\rho}_n^k$ *and with* 1 *if* k *is odd;*

3. *without* ρ_n^k *and* $\bar{\rho}_n^k$ *and with* 1 *having multiplicity* 2 *if* $k \neq 0$ *is even.*

Proof. Put

$$a_{n,k}(x) = \sum_{j=0}^{n-1} x^k \rho_{2n}^{(2j+1)k} = \rho_{2n}^k \frac{x^n(-1)^k - 1}{x\rho_n^k - 1},$$

and observe that

$$t_{n,k}^{(T)}(x) = 1/2(\bar{a}_{n,k}(x) + a_{n,k}(x)).$$

∎

3.5 Trigonometric transforms.

In this paragraph we obtain new algorithms for DHT, DST and DCT. We move from the observation that all the expressions (3.12), (3.14), (3.17), (3.18) and (3.19) have as denominators real polynomials of the form $1 - 2x\cos\alpha + x^2$ for suitable α. They are exactly of the type that we met in paragraph 3.3 in the first step of the DIF version of Bruun's algorithm as denominators of the partial fractions in the summation (3.8). This suggests us to use the same grouping technique used in Bruun's algorithm for the computation of the linear combinations $\sum_{j=0}^{n-1} b_j p_j(x)$ when $p_j = h_{n,j}$, $p_j = s_{n,j}$, $p_j = c_{n,j}$ and so on. In other words, in the first step we compute $(b_0 p_0(x) + b_{n/2} p_{n/2}(x))$ and $(b_j p_j(x) + b_{n-j} p_{n-j}(x))$ for $j = 0, \ldots, n/2 - 1$, and we proceed recursively by using the same grouping strategy in the subsequent steps. Due to the close relation of the algorithms we present with Bruun's algorithm, we call them BFHT, BFST and BFCT respectively.

3.5.1 Hartley transform.

Let $\mathbf{b} = (b_j)$ $j = 0, \ldots, n-1$ with $n = 2^d$. We have to compute the linear combination

$$c(x) = \sum_{j=0}^{n-1} b_j h_{n,j}(x), \tag{3.20}$$

see (3.12). By adding terms 0 and $n/2$ and j and $n - j$ we obtain

$$c(x) = (x^n - 1)\frac{x(b_0 - b_{n/2}) + (b_0 + b_{n/2})}{x^2 - 1} + (x^n - 1)\sum_{j=0}^{n/2-1} \frac{x(b_j\mu_j + b_{n-j}\mu_{n-j}) - (b_j + b_{n-j})}{(x - \omega_n^j)(x - \bar{\omega}_n^j)},$$

in fact, since $(x-\omega_n^j)(x-\bar{\omega}_n^j) = (x-\omega_n^{n-j})(x-\bar{\omega}_n^{n-j})$, the representations (3.12) of $h_{n,j}(x)$ and $h_{n-j,j}(x)$ have the same denominator. From this point BFHT proceeds through $d-1$ steps exactly as Bruun's algorithm.

The formal description of BFHT follows. For the sake of clarity, we divide it in two stages. The first one contains the first step of the algorithm. The second one contains the steps from 1 to $d-1$. As for Bruun's algorithm we denote with $\phi_{l,j}(x)$ with $l = 0, \ldots, d-1$ and $j = 0, \ldots, 2^{d-l-1} - 1$, the numerators of the partial fractions that we construct step after step by grouping and adding terms of the summation (3.20) as described.

BFHT

Stage 1)

$$\begin{cases} \phi_{0,0} = x(b_0 - b_{n/2}) + (b_0 + b_{n/2}). \\ \\ \phi_{0,j} = x\big(b_j\mu_j + b_{n-j}\mu_{n-j}\big) - (b_j + b_{n-j}), \qquad j = 1, \ldots, 2^{d-1} - 1. \end{cases}$$

Stage 2) For $l = 1,\ldots,d-1$ we compute

$$\begin{cases} \phi_{l,0} = \phi_{l-1,0}(x^{2^l} + 1) + \phi_{l-1,2^{d-l-1}}(x^{2^l} - 1) = \\ \qquad = x^{2^l}(\phi_{l-1,0} + \phi_{l-1,2^{d-l-1}}) + (\phi_{l-1,0} - \phi_{l-1,2^{d-l-1}}). \\ \\ \phi_{l,j} = \phi_{l-1,j}(x^{2^l} + 2\cos(2\pi j/2^{d-l+1})x^{2^{l-1}} + 1) + \\ \qquad\qquad + \phi_{l-1,2^{d-l}-j}(x^{2^l} - 2\cos(2\pi j/2^{d-l+1})x^{2^{l-1}} + 1) = \\ \\ \qquad = x^{2^l}(\phi_{l-1,j} + \phi_{l-1,2^{d-l}-j}) + 2\cos(2\pi j/2^{d-l+1})(\phi_{l-1,j} - \phi_{l-1,2^{d-l}-j})x^{2^{l-1}} + \\ \qquad\qquad + (\phi_{l-1,j} + \phi_{l-1,2^{d-l}-j}), \quad j = 1, \ldots, 2^{d-l-1} - 1. \end{cases}$$

The same tree shown in the case of Bruun's algorithm describes as well the computation of BFHT. The operation count for BFHT is given by

$$\text{Mults} \quad = \quad \left\{ \begin{array}{ll} \text{Stage 1:} & 2(n/2-2) - 2 \\ \text{Stage 2:} & \sum_{l=1}^{d-1}(2^{d-l-1} - 1)2^l \end{array} \right\} = \frac{n}{2}(\log n - 1) - 4,$$

$$\text{Adds} \;=\; \left\{ \begin{array}{ll} \text{Stage 1:} & n-2 \\ \text{Stage 2:} & \sum_{l=1}^{d-1}\{2^{l+1}+(2^{d-l-1}-1)2^l 3\} \end{array} \right\} = \frac{3n}{2}(\log n - 1).$$

Comparisons with radix-2 and split-radix fast Hartley transform [58] leads to conclusions analogous to those obtained in paragraph 3.3.2.

3.5.2 Sine transform.

We have to compute $\sum_{j=1}^{n} b_j s_{n,j}(x)$, where formula (3.14) holds for the $s_{n,j}(x)$ with $j \neq 0$. Let in this case $n+1 = 2^d$. For convenience we put $s_{n,0}(x) = 0$ so that we may let j start from 0 in the above summation.

The BFST algorithm we present performs d steps indexed with l ranging from 1 to d. It uses the same grouping technique used by Bruun's algorithm and BFHT. The main difference is that in the present case the eigenpolynomials (3.14) of odd and even index have a slightly different form. However, note that the indexes j and $2^{d-l}-j$ are both even or both odd if $l < d$. This means that the computations of the summations $\sum_{\substack{j=0 \\ j \text{ even}}}^{n} b_j s_{n,j}(x)$ and $\sum_{\substack{j=0 \\ j \text{ odd}}}^{n} b_j s_{n,j}(x)$ are independent until $l = d$. Moreover, since $(x^2-1)\prod_{\substack{j=1 \\ j \text{ even}}}^{n}(x-\rho_{n+1}^j)(x-\bar{\rho}_{n+1}^j) = x^{n+1}-1$ and $\prod_{\substack{j=1 \\ j \text{ odd}}}^{n}(x-\rho_{n+1}^j)(x-\bar{\rho}_{n+1}^j) = x^{n+1}+1$, in the d-th step of the algorithm we have

$$\sum_{\substack{j=0 \\ j \text{ even}}}^{n} b_j s_{n,j}(x) + \sum_{\substack{j=0 \\ j \text{ odd}}}^{n} b_j s_{n,j}(x) = (-x^{n+1}+1)\frac{\phi_{d-1,0}}{x^{n+1}-1} + (x^{n+1}+1)\frac{\phi_{d-1,1}}{x^{n+1}+1} = \phi_{d-1,1} - \phi_{d-1,0},$$

where the symbols $\phi_{l,j}$ have the usual meaning. The formal description of BFST follows.

BFST

Stage 1) Put

$$\left\{ \begin{array}{l} \phi_{1,0} = b_{2^{d-1}}(x^2-1). \\ \\ \phi_{1,j} = x^2 \sin\frac{\pi j}{2^d}(b_j + b_{2^d-j}) + 2\cos\frac{\pi j}{2^d}\sin\frac{\pi j}{2^d}(b_j - b_{2^d-j})x + \sin\frac{\pi j}{2^d}(b_j + b_{2^d-j}). \end{array} \right.$$

Stage 2) For $l = 2, \ldots, d-1$ compute

$$\left\{ \begin{array}{l} \phi_{l,0} = \phi_{l-1,0}(x^{2^l}+1) + \phi_{l-1,2^{d-l}}(x^{2^l}-1) = \\ \qquad = x^{2^l}(\phi_{l-1,0} + \phi_{l-1,2^{d-l}}) + (\phi_{l-1,0} - \phi_{l-1,2^{d-l}}). \\ \\ \phi_{l,j} = \phi_{l-1,j}(x^{2^l} + 2\cos(\pi j/2^{d-l+1})x^{2^{l-1}} + 1) + \\ \qquad\qquad + \phi_{l-1,2^{d-l+1}-j}(x^{2^l} - 2\cos(\pi j/2^{d-l+1})x^{2^{l-1}} + 1) = \\ \\ \qquad = x^{2^l}(\phi_{l-1,j} + \phi_{l-1,2^{d-l+1}-j}) + 2\cos(\pi j/2^{d-l+1})(\phi_{l-1,j} - \phi_{l-1,2^{d-l+1}-j})x^{2^{l-1}} + \\ \qquad\qquad + (\phi_{l-1,j} + \phi_{l-1,2^{d-l+1}-j}), \quad j = 1, \ldots, 2^{d-l}-1. \end{array} \right. \tag{3.21}$$

Stage 3) Compute

$$\phi_{d,0} = \phi_{d-1,1} - \phi_{d-1,0}.$$

We note that $\phi_{l,j}$ has degree $2^{l+1}-2$ for every $j = 0, \ldots, 2^{d-l}-1$. In addition $\phi_{l,j}$ is 1-reciprocal if $j \neq 0$ while is -1-reciprocal if $j = 0$ (an n^{th} degree polynomial $p(x)$ is ϵ-reciprocal if $x^n p(1/x) = \epsilon p(x)$). These properties lead to a reduction of the arithmetic cost by a factor of 1/2. The operation count for BFST follows:

$$\text{Mults} \;=\; \left\{ \begin{array}{ll} \text{Stage 1:} & 2((n+1)/2-1) \\ \text{Stage 2:} & \sum_{l=1}^{d-1}(2^{d-l}-1)2^{l-1} \end{array} \right\} = \frac{n+1}{2}(\log(n+1)-1),$$

$$\text{Adds} \;=\; \left\{ \begin{array}{ll} \text{Stage 1:} & (n+1)-2 \\ \text{Stage 2:} & \sum_{l=1}^{d-1}\{(2^l-2)+(2^{d-l}-1)(2^{l-1}3-1)\} \\ \text{Stage 3:} & (n+1)-2 \end{array} \right\} =$$

$$= (n+1)(\frac{3}{2}\log(n+1)-2) - \log(n+1) + 2.$$

Thus the algorithm performs $2(n+1)\log(n+1) - 5/2(n+1) + O(1)$ arithmetic operations, in line with the best algorithms for the computation of this transform [54].

3.5.3 Cosine transform I.

In this section we consider the cosine matrix $C_n = (\cos \frac{\pi ij}{n})$, whose eigenpolynomials are representable as in (3.17). Here we again let $n = 2^d$.

For the derivation of BFST we exploited the symmetry properties of the eigenpolynomials. In order to do the same for cosine transform let us slightly modify the polynomial $c_{n,j}(x)$ obtained in (3.17). More precisely, we replace $c_{n,j}(x)$ with

$$\tilde{c}_{n,j}(x) = c_{n,j}(x) + \frac{1}{2}(x^n(-1)^j - 1) = \frac{1}{2}(x^2-1)\frac{x^n(-1)^j - 1}{\left(x - \rho_n^j\right)\left(x - \bar{\rho}_n^j\right)}, \tag{3.22}$$

and compute $\sum_{j=0}^{n-1} b_j \tilde{c}_{n,j}$: it is obvious how to recover the correct transform from the resulting polynomial.

The algorithm BFCT-I (in the next section we will consider a different version of the cosine matrix and introduce a BFCT-II) we present performs $d+2$ steps. We have divided the algorithm in four stages. In phase 1 we use the fact that $\tilde{c}_{n,0}(x) = \frac{1}{2}(x+1)^2 \frac{x^n-1}{x^2-1}$. Stage 2 is quite analogous to the same phase in BFST. The comments made for the sum of odd and even indexed eigenpolynomials in BFST apply as well for BFCT-I, with the only observation that in $(d+1)$-st step we compute $\phi_{d-1,0} - \phi_{d-1,1}$. This is done in stage 3. Stage 4 is devoted to the reconstruction of the desired transform $\sum_{j=0}^{n-1} b_j c_{n,j}$ from $\sum_{j=0}^{n-1} b_j \tilde{c}_{n,j}$.

BFCT-I

Stage 1) Put

$$\begin{cases} \phi_{0,0} = b_0/2(x+1)^2. \\ \\ \phi_{0,j} = b_j/2(x^2-1), \qquad j = 1, \ldots, n-1. \end{cases}$$

Stage 2) For $l = 1, \ldots, d-1$ execute (3.21).
Stage 3) Compute

$$\phi_{d,0} = \phi_{d-1,0} - \phi_{d-1,1}.$$

Stage 4) If $\phi_{d,0} = \sum_{j=0}^{n} \varphi_j x_j$ then the desired transform is $(2\varphi_0, \varphi_1, \ldots, \varphi_{n-1})$.

In BFCT-I, for $l = 0, \ldots, d-1$ and $j = 0, \ldots, 2^{d-l} - 1$, we have that $\phi_{l,j}$ has degree 2^{l+1} and is -1-reciprocal if $j \neq 0$ while is 1-reciprocal if $j = 0$ (see the section on BFST).

The operation count for BFCT-I is

$$\begin{aligned} \text{Mults} &= \left\{ \text{ Stage 2: } \sum_{l=1}^{d-1}(2^{d-l}-1)2^{l-1} \right\} = \frac{n}{2}(\log n - 2) + 1, \\ \text{Adds} &= \left\{ \begin{array}{ll} \text{Stage 2:} & \sum_{l=1}^{d-1}\{2^l + (2^{d-l}-1)(2^{l-1}3-1)\} \\ \text{Stage 3:} & n-1 \end{array} \right\} = n(\frac{3}{2}\log n - 2) + \log n + 1, \end{aligned}$$

leading to the same considerations made for BFST.

3.5.4 Cosine transform II.

Here we consider the Cosine matrix T_n defined in section 3.2. In section 3.4 we found compact representations for the eigenpolynomials of $\mathrm{Diag}(\sqrt{2}, 1, \ldots, 1)T_n = D_n T_n$, see (3.18), and of $T_n^T D_n^T$, see (3.19).

We consider first the computation of inverse DCT, i.e. of the linear combination $\frac{b_0}{\sqrt{2}} t_{n,0}^{(T)}(x) + \sum_{k=1}^{n-1} b_k t_{n,k}^{(T)}(x)$. The eigenpolynomials $t_{n,k}^{(T)}(x)$ have a form very close to that of eigenpolynomials

$s_{n,j}(x)$ see (3.14) and $c_{n,j}(x)$ see (3.17). The algorithm that we propose in order to compute their linear combination does not differ in any significant way from BFCT-I. We call it IBFCT-II, for inverse BFCT-II: its formal description follows.

IBFCT-II

Stage 1) Put

$$\begin{cases} \phi_{0,0} = \frac{b_0}{\sqrt{2}}(x+1). \\ \\ \phi_{0,j} = b_j \cos\frac{j\pi}{2n}(x-1), \quad j = 1, \ldots, n-1. \end{cases}$$

Stage 2) For $l = 1, \ldots, d-1$ compute (3.21).
Stage 3) Compute

$$\phi_{d,0} = \phi_{d-1,0} - \phi_{d-1,1}.$$

Remark that (a) the computations of the summations $\sum_{\substack{k=0 \\ j \text{ even}}}^{n} b_k t_{n,k}^{(T)}(x)$ and $\sum_{\substack{k=0 \\ j \text{ odd}}}^{n} b_k t_{n,k}^{(T)}(x)$ are independent until step $d+1$ (b) in IBFCT-II for every $l = 0, \ldots, d-1$ and $j = 0, \ldots, 2^{d-l} - 1$ results that $\phi_{l,j}$ has degree $2^{l+1} - 1$ and is -1-reciprocal if $j \neq 0$ while is 1-reciprocal if $j = 0$.

We turn to the problem of an efficient computation of the linear combination $\sum_{j=0}^{n-1} b_j t_{n,j}(x)$. Unfortunately, expression (3.18) does not exhibit any symmetry property. However, observe that formally

$$\sum_{j=0}^{n-1} b_j t_{n,j} = \frac{p(x)}{x^{2n}+1}, \tag{3.23}$$

being $p(x)$ a polynomial of degree $3n-1$. Clearly $p(x)$ must be of the form $(x^{2n}+1)q(x)$ having $q(x)$ degree $n-1$. This means that we need to compute only the coefficients of the n monomials of lower degree of $p(x)$, but for this is just enough to compute $(x^{2n}+1)\sum_{j=0}^{n-1} b_j \hat{t}_{n,j}(x)$ with

$$\hat{t}_{n,j}(x) = \frac{-x\cos\frac{(2j+1)\pi}{2n} + 1}{(x - \rho_{2n}^{2j+1})(x - \bar{\rho}_{2n}^{2j+1})}.$$

Moreover, we can treat the $\hat{t}_{n,j}(x)$ as we did for the $c_{n,j}(x)$ in (3.22) obtaining

$$\tilde{t}_{n,j}(x) = \hat{t}_{n,j}(x) - \frac{1}{2} = \frac{1}{2}\frac{1-x^2}{\left(x - \rho_{2n}^{2j+1}\right)\left(x - \bar{\rho}_{2n}^{2j+1}\right)}. \tag{3.24}$$

Thus our strategy will be the computation of

$$(x^{2n}+1)\sum_{j=0}^{n-1} b_j \tilde{t}_{n,j}(x). \tag{3.25}$$

Furthermore, we can get the cosine transform of $\mathbf{b}$ from (3.25) just by multiplying the last coefficient of the result by $\sqrt{2}$.

The particular form of the denominators of the eigenpolynomials, i.e. $(x - \rho_{2n}^{2j+1})(x - \bar{\rho}_{2n}^{2j+1})$, $j = 0$, $\ldots$, $n-1$, induces us to use a different grouping technique for computing the linear combination (3.25). More precisely in the first step we add terms of index j and $n-j-1$ for $j = 0, \ldots, n/2-1$ and we proceed uniformly in the successive steps. Proceeding in this way, at the l-th step we are left with $n/2^l$ partial fractions having as denominators $(x^{2^l} - \rho_{2n/2^l}^{2j+1})(x^{2^l} - \bar{\rho}_{2n/2^l}^{2j+1})$, $j = 0, \ldots, n/2^l - 1$, and as numerators -1-reciprocal polynomials of degree 2^{l+1}. As usual, we denote these numerators with $\phi_{l,j}$. The formal description of the algorithm follows.

BFCT-II

Stage 1) For $j = 0, \ldots, n-1$ put

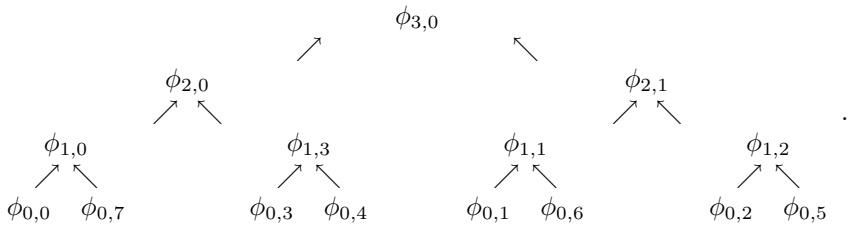
$$\phi_{0,j} = \frac{1}{2} b_j (1 - x^2).$$

Stage 2) For $l = 1, \ldots, d$

$$\begin{aligned}
\phi_{l,j} &= \phi_{l-1,j}(x^{2^l} + 2\cos(\pi(2j+1)/2^{d-l+2})x^{2^{l-1}} + 1) + \\
&\quad + \phi_{l-1,2^{d-l+1}-j-1}(x^{2^l} - 2\cos(\pi(2j+1)/2^{d-l+2})x^{2^{l-1}} + 1) = \\
&= x^{2^l}(\phi_{l-1,j} + \phi_{l-1,2^{d-l+1}-j-1}) + 2\cos(\pi(2j+1)/2^{d-l+2})(\phi_{l-1,j} - \phi_{l-1,2^{d-l+1}-j-1})x^{2^{l-1}} + \\
&\quad + (\phi_{l-1,j} + \phi_{l-1,2^{d-l+1}-j-1}), \quad j = 0, \ldots, 2^{d-l}.
\end{aligned}$$

Stage 3) If $\phi_{d,0} = \sum_{j=0}^{2n} \varphi_j x_j$ then the desired transform is $(\sqrt{2}\varphi_0, \varphi_1, \ldots, \varphi_{n-1})$.

For example for $n = 16$ the computation proceeds as follows



The permutation involved is the following:

$$\begin{cases} \Gamma_1 = (1) \\ \Gamma_n = \Gamma_{\frac{n}{2}} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \left(I_{\frac{n}{2}} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \middle| J_{\frac{n}{2}} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) = \left(\Gamma_{\frac{n}{2}} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \middle| \Gamma_{\frac{n}{2}} J_{\frac{n}{2}} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) . \end{cases} \tag{3.26}$$

The arithmetic costs of IBFCT-II and BFCT-II are the same and precisely $\frac{n}{2}\log n + 1$ multiplications (included the multiplication by $\sqrt{2}$ needed in both cases to reconstruct exactly the DCT), and $n(\frac{3}{2}\log n - 1) + 1$ additions. These costs match those of the best algorithms known so far for the computation of this particular kind of DCT [48, 69].

Chapter 4

Bruun's algorithm: a real factor framework for the fast Fourier transform.

The title and most of the ideas in this chapter have been inspired by the beautiful book of C. Van Loan *Computational Frameworks for the Fast Fourier Transform* [68]. We quote a passage from the Preface that well summarizes the strategy that underlies the book

> [...] different FFT algorithms correspond to different factorizations of the discrete Fourier transform (DFT) matrix. The matrix factorization point of wiew, so successful in other areas of numerical linear algebra, goes a long way toward unifying and simplifying the FFT literature. It closes the gap betweeen the computer implementation of an FFT and the underlying mathematics, because it forces us to think well above the scalar level.

In this chapter we will study the factorization of the Fourier matrix associated with Bruun's algorithm. We think that this will help to clarify the nature of BFHT, BFST and BFCT (see section 3.5), since they are all closely related to Bruun's algorithm.

Bruun's algorithm was published in 1978 [27]. It belongs to a family of algorithms known as *real factor* algorithms. Other members of this family are the Rader-Brenner algorithm [56], published in 1976, and the Preuss algorithm [55], published in 1982. The distinctive feature of these algorithms is that the complex twiddle factors, characteristic of radix-2 FFT, are almost entirely replaced by real numbers. These algorithms obtain a good saving of real multiplications with respect to radix-2^m FFTs which, however, is partially balanced by an increased number of additions. It should be noted that the numerical accuracy of real factor algorithms seemed to be inferior and the indexing required somewhat more complicated. After the publication, in 1984, of the two papers [35, 69] about split-radix FFT the interest in real factor algorithms faded away. The reasons are the better operation count of the split-radix FFT, its flexibility for the treatment of real and real symmetric sequences and its good numerical accuracy [34]. However, some recent results [72, 61] indicate a renewed interest in Bruun's algorithm.

Actually, on most sequential machines split-radix FFT should run faster that every one of the mentioned algorithms. However, we are not aware of any implementation of a real factor algorithm nor of split-radix FFT on a parallel machine such as a vector computer. We hope that our study of Bruun's algorithm can help to evaluate the possible usefulness of the algorithm in a parallel environment.

The chapter is organized as follows. After introducing some notations, in section 4.2 we perform a detailed study of certain permutations that are related with Bruun's algorithm. In section 4.3 we show the factorization of the Fourier matrix that underlies the algorithm. In section 4.4 we will see how the combination of DIT and DIF versions of Bruun's algorithm can be exploited for computing convolutions of real sequences. In section 4.5 we present an autosort variant of the algorithm, particularly suited for an implementation on a vector computer. In section 4.6, using the matrix factorization point of view, we discuss the relation existing between Bruun's algorithm and BFHT, BFST, BFCT. In this way, the

results obtained in this chapter turn out to be useful for a better understanding of all these algorithms. Finally, in section 4.7 we discuss the stability properties of the algorithm.

4.1 Notations.

In this chapter we borrow the notations from [68]. The entries of vectors and matrices are indexed from zero, unless otherwise stated. The range where an index runs is specified by means of *column notation*, i.e., $i = j:k$ is equivalent to $i = j, j+1, \ldots, k$. Nonunit increments are specified with the notation $j:s:k$ so that $i = 0:2:6$ means $i = 0, 2, 4, 6$ and $i = 1:3:5$ means $i = 1, 4$. Given a vector $\mathbf{x} \in \mathsf{C}^n$, with $\mathbf{x}(i:j:k)$ we denote the vector made up by the entries of $\mathbf{x}$ whose idexes belong to the specified range. For example if $\mathbf{x} = (x_i) \in \mathsf{C}^6$ then $\mathbf{x}(1:2:6) = [x_1, x_3, x_5]^T$.

If $\mathbf{x} \in \mathsf{C}^n$ and $n = rc$, then with $\mathbf{x}_{r\times c}$ we denote the $r \times c$ matrix obtained arranging the entries of $\mathbf{x}$ columnwise

$$\mathbf{x}_{r\times c} = [\mathbf{x}(0:r-1)|\mathbf{x}(r:2r-1)|\cdots|\mathbf{x}(n-r:n-1)].$$

We need the following lemma, whose simple proof is left to the reader (see [68]).

Lemma 4.1 *If $A \in \mathsf{C}^{p\times r}$ and $\mathbf{x} \in \mathsf{C}^n$ with $n = rc$ then we have*

(1)

$$\mathbf{y} = (I_c \otimes A)\mathbf{x} \;\Leftrightarrow\; \mathbf{y}_{p\times c} = A\mathbf{x}_{r\times c};$$

(2)

$$\mathbf{y} = (A \otimes I_c)\mathbf{x} \;\Leftrightarrow\; \mathbf{y}_{c\times p} = \mathbf{x}_{c\times r}A^T.$$

Finally, we let $\text{xor} : \{0,1\}^2 \to \{0,1\}$ be the *exlusive or* operator, whose value is 1 if and only if only one of its arguments is 1. The operator xor is commutative and associative, as it is easy to verify.

4.2 The permutations Υ_n, Ψ_n, Γ_n and Σ_n.

It is important to develop a deep understanding of the action of the permutations

$$\begin{cases} \Sigma_1 = (1) \\ \Sigma_n = \Sigma_{n/2} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \left(I_{n/2} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \middle| K_{n/2} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) = \left(\Sigma_{n/2} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \middle| \Sigma_{n/2}K_{n/2} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) \end{cases} \tag{4.1}$$

and

$$\begin{cases} \Gamma_1 = (1) \\ \Gamma_n = \Gamma_{n/2} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \left(I_{n/2} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \middle| J_{n/2} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) = \left(\Gamma_{n/2} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \middle| \Gamma_{n/2}J_{n/2} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) \end{cases} \tag{4.2}$$

that we met in chapter 3, see (3.9) and (3.26). Here a different characterization of these two permutations is given and is proved to be equivalent to (4.2) and (4.1). A "bit level" description of these permutations is also indicated.

Actually, Σ_n and Γ_n are built upon two simpler permutations, denoted by Υ_n and Ψ_n, so that we will start with a detailed study of these ones.

4.2.1 The permutations Υ_n and Ψ_n.

Let $n = 2^d$ with $d \geq 1$. We set $\Upsilon_2 = \Psi_2 = I_2$ and, for $d \geq 2$,

$$\Upsilon_n = \left(\frac{I_{n/2} \otimes (1\ 0)}{I_{n/2} \otimes (0\ 1)} \right) \tag{4.3}$$

and

$$\Psi_n = \left(\begin{array}{c} I_{n/4} \otimes \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ \hline I_{n/4} \otimes \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \end{array} \right). \tag{4.4}$$

As an example let $\mathbf{x} = (x_i)$ with $i = 0:7$. Then

$$\Upsilon_8 \mathbf{x} = (x_0\ x_2\ x_4\ x_6\ x_1\ x_3\ x_5\ x_7)^T,$$

while

$$\Psi_8 \mathbf{x} = (x_0\ x_3\ x_4\ x_7\ x_1\ x_2\ x_5\ x_6)^T.$$

We see that the first (last) four entries of $\Upsilon_8 \mathbf{x}$ are the entries of $\mathbf{x}$ whose index equals 0 (1) modulo 2, i.e., is even (odd). For this reason Υ_n is called *even-odd* permutation. On the other hand, the first (last) four entries of $\Psi_8 \mathbf{x}$ are the entries of $\mathbf{x}$ whose index equals 0 or 3 (1 or 2) modulo 4. The following lemma states these facts formally.

Lemma 4.2 *Let $n = 2^d$ with $d \geq 2$, let $\mathbf{x} = (x_i) \in \mathsf{C}^n$ and let $\mathbf{x}_U$, $\mathbf{x}_L \in \mathsf{C}^{n/2}$.*

1. *We have $\Upsilon_n \mathbf{x} = \begin{pmatrix} \mathbf{x}_U \\ \mathbf{x}_L \end{pmatrix}$, where, for $i = 0:n/2-1$*

$$[\mathbf{x}_U]_i = x_{2i}, \qquad [\mathbf{x}_L]_i = x_{2i+1}.$$

2. *We have $\Psi_n \mathbf{x} = \begin{pmatrix} \mathbf{x}_U \\ \mathbf{x}_L \end{pmatrix}$, where, for $i = 0:n/2-1$*

$$[\mathbf{x}_U]_i = \begin{cases} x_{2i} & \text{if } i \text{ is even} \\ x_{2i+1} & \text{if } i \text{ is odd} \end{cases}, \qquad [\mathbf{x}_L]_i = \begin{cases} x_{2i+1} & \text{if } i \text{ is even} \\ x_{2i} & \text{if } i \text{ is odd} \end{cases}.$$

Proof.

1. Since

$$\Upsilon_n \mathbf{x} = \left(\begin{array}{c} I_{n/2} \otimes (1\,0) \\ \hline I_{n/2} \otimes (0\,1) \end{array} \right) \mathbf{x} = \left(\begin{array}{c} I_{n/2} \otimes (1\,0)\mathbf{x} \\ \hline I_{n/2} \otimes (0\,1)\mathbf{x} \end{array} \right),$$

   we have $\mathbf{x}_U = I_{n/2} \otimes (1\,0)\mathbf{x}$ and $\mathbf{x}_L = I_{n/2} \otimes (0\,1)\mathbf{x}$. Thus, by lemma 4.1 (1),

$$\mathbf{x}_U^T = \mathbf{x}_{U\,1\times n/2} = (1\,0)\mathbf{x}_{2\times n/2} = (1\,0) \left(\begin{array}{c} \mathbf{x}^T(0:2:n-1) \\ \hline \mathbf{x}^T(1:2:n-1) \end{array} \right) = \mathbf{x}^T(0:2:n-1).$$

   Analogously, $\mathbf{x}_L^T = \mathbf{x}^T(1:2:n-1)$. The thesis follows.

2. We have $\mathbf{x}_U = I_{n/4} \otimes \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \mathbf{x}$ and $\mathbf{x}_L = I_{n/4} \otimes \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \mathbf{x}$. Thus, by using again lemma 4.1 (1),

$$\mathbf{x}_{U\,2\times n/4} = \left(\begin{array}{c} \mathbf{x}_U^T(0:2:n/2-1) \\ \hline \mathbf{x}_U^T(1:2:n/2-1) \end{array} \right) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \mathbf{x}_{4\times n/4} = \left(\begin{array}{c} \mathbf{x}^T(0:4:n-1) \\ \hline \mathbf{x}^T(3:4:n-1) \end{array} \right).$$

   This yields $\mathbf{x}_U(0:2:n/2-1) = \mathbf{x}(0:4:n-1)$ and $\mathbf{x}_U(1:2:n/2-1) = \mathbf{x}(3:4:n-1)$. Analogously we find $\mathbf{x}_L(0:2:n/2-1) = \mathbf{x}(1:4:n-1)$ and $\mathbf{x}_L(1:2:n/2-1) = \mathbf{x}(2:4:n-1)$. In both cases the thesis follows.

∎

Thus, if $\mathbf{x} \in \mathsf{C}^n$ the computation of $\mathbf{y} = \Upsilon_n \mathbf{x}$ can be handled as

```
m ← n/2
for j = 0 : m − 1
    y(j) ← x(2j)
    y(j + m) ← x(2j + 1)
end
```

while the computation of $\mathbf{y} = \Psi_n \mathbf{x}$ as

```
m ← n/2
for j = 0 : 2 : m − 1
    y(j) ← x(2j)
    y(j + 1) ← x(2j + 3)
    y(j + m) ← x(2j + 1)
    y(j + m + 1) ← x(2j + 2)
end
```

Now let u_n and p_n the two permutations over $\{0, \cdots, n-1\}$ such that

$$[\Upsilon_n \mathbf{x}]_k = x_{u_n(k)}, \qquad [\Psi_n \mathbf{x}]_k = x_{p_n(k)}.$$

It is interesting to look at the action of u_n and p_n at bit level. To this aim we denote by $(b_d \cdots b_1)_2$, where $b_i \in \{0, 1\}$, the binary representation of the integer $\sum_{i=0}^{d-1} b_{i+1} 2^i$.

Lemma 4.3 *Let $k = (b_d \cdots b_1)_2$. Then*

a) *we have*

$$u_n(k) = u_n((b_d \cdots b_1)_2) = (b_{d-1} \cdots b_1\, b_d)_2;$$

b) *we have*

$$p_n(k) = p_n((b_d \cdots b_1)_2) = (b_{d-1} \cdots b_1\, c)_2,$$

where $c = b_d$ xor b_1.

Proof.

a) By virtue of lemma 4.2 we have

$$[\Upsilon_n \mathbf{x}]_k = \begin{cases} [\mathbf{x}_U]_k & \text{if } 0 \le k < n/2 \\ [\mathbf{x}_L]_{k-n/2} & \text{if } n/2 \le k < n \end{cases} = \begin{cases} x_{2k} & \text{if } 0 \le k < n/2 \\ x_{2k-n+1} & \text{if } n/2 \le k < n \end{cases}.$$

If $0 \le k < n/2$ then $k = (0\, b_{d-1} \cdots b_1)_2$ and

$$u_n(k) = 2k = (b_{d-1} \cdots b_1\, 0)_2 = (b_{d-1} \cdots b_1\, b_d)_2.$$

On the other hand if $n/2 \le k < n$ then $k = (1\, b_{d-1} \cdots b_1)_2$ and

$$u_n(k) = 2k - n + 1 = (b_{d-1} \cdots b_1\, 1)_2 = (b_{d-1} \cdots b_1\, b_d)_2.$$

b) Again, by virtue of lemma 4.2 we have

$$[\Psi_n \mathbf{x}]_k = \begin{cases} [\mathbf{x}_U]_k & \text{if } 0 \le k < n/2 \\ [\mathbf{x}_L]_{k-n/2} & \text{if } n/2 \le k < n \end{cases}.$$

We need to consider four cases.

1. If k is even and such that $0 \le k < n/2$ then $k = (0\, b_{d-1} \cdots b_2\, 0)_2$ and

$$p_n(k) = 2k = (b_{d-1} \cdots b_2\, 0\, 0)_2 = (b_{d-1} \cdots b_1\, c)_2.$$

2. If k is odd and such that $0 \leq k < n/2$ then $k = (0\, b_{d-1} \cdots b_2\, 1)_2$ and

$$p_n(k) = 2k + 1 = (b_{d-1} \cdots b_2\, 1\, 1)_2 = (b_{d-1} \cdots b_1\, c)_2.$$

3. If k is even and such that $n/2 \leq k < n$ then $k = (1\, b_{d-1} \cdots b_2\, 0)_2$ and

$$p_n(k) = 2(k - n/2) + 1 = (b_{d-1} \cdots b_2\, 0\, 1)_2 = (b_{d-1} \cdots b_1\, c)_2.$$

4. If k is odd and such that $n/2 \leq k < n$ then $k = (1\, b_{d-1} \cdots b_2\, 1)_2$ and

$$p_n(k) = 2(k - n/2) = (b_{d-1} \cdots b_2\, 1\, 0)_2 = (b_{d-1} \cdots b_1\, c)_2.$$

■

We end this section with an additional simple lemma.

Lemma 4.4 *Let $n = 2^d$ with $d \geq 2$. Then*

$$\Upsilon_n K_n \Upsilon_n{}^T = \begin{pmatrix} K_{n/2} & O \\ O & J_{n/2} \end{pmatrix} \tag{4.5}$$

and

$$\Psi_n J_n \Psi_n{}^T = I_2 \otimes J_{n/2}. \tag{4.6}$$

Proof. In order to prove equality (4.5) first observe that

$$K_n = K_{n/2} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} + J_{n/2} \otimes \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}.$$

Using this relation we have

$$\begin{aligned} \Upsilon_n K_n \Upsilon_n{}^T &= \left(\frac{I_{n/2} \otimes (1\, 0)}{I_{n/2} \otimes (0\, 1)} \right) (K_{n/2} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} + J_{n/2} \otimes \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}) \left(I_{n/2} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \middle| I_{n/2} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) = \\ &= \begin{pmatrix} K_{n/2} & O \\ O & O \end{pmatrix} + \begin{pmatrix} O & O \\ O & J_{n/2} \end{pmatrix} = \begin{pmatrix} K_{n/2} & O \\ O & J_{n/2} \end{pmatrix}. \end{aligned}$$

The proof of equality (4.6) proceeds along the same line, once noted that $J_n = J_{n/4} \otimes J_4$. For brevity the details are omitted. ■

4.2.2 The permutation Γ_n.

By means of Ψ_n we can now define the permutation Γ_n. Let $n = 2^d$ with $d \geq 1$. We set

$$\begin{cases} \Gamma_1 = (1) \\ \Gamma_n = (I_2 \otimes \Gamma_{n/2}) \Psi_n \end{cases}. \tag{4.7}$$

As an example, if $\mathbf{x} \in \mathsf{C}^8$ we have

$$\Gamma_8 \mathbf{x} = (x_0\, x_7\, x_3\, x_4\, x_1\, x_6\, x_2\, x_5)^T.$$

Let us prove the equivalence of the formulation (4.7) with (4.2).

Theorem 4.5 *Let $n = 2^d$ with $d \geq 1$. Then*

$$\Gamma_n = \left(\Gamma_{n/2} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \middle| \Gamma_{n/2} J_{n/2} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right).$$

Proof. We proceed by induction on n. The cases where $n = 2$ and $n = 4$ are easy. Let $n \geq 8$. Observe that

$$\Psi_n = \left(\begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix} \otimes I_{n/4}\Psi_{n/2} \middle| \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \otimes I_{n/4}\Psi_{n/2} \right).$$

Thus

$$\begin{aligned}
&\left(\Gamma_{n/2} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \middle| \Gamma_{n/2}J_{n/2} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) \Psi_n^T = \\
&= (\Gamma_{n/2}\Psi_{n/2}^T) \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \otimes I_{n/4} + (\Gamma_{n/2}J_{n/2}\Psi_{n/2}^T) \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \otimes I_{n/4} = \\
&= I_2 \otimes (\Gamma_{n/4} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix}) \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \otimes I_{n/4} + I_2 \otimes (\Gamma_{n/4}J_{n/4} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix}) \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \otimes I_{n/4} = \\
&= \begin{pmatrix} \Gamma_{n/4} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} & O & O & O \\ O & O & \Gamma_{n/4} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} & O \end{pmatrix} + \begin{pmatrix} O & \Gamma_{n/4}J_{n/4} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} & O & O \\ O & O & O & \Gamma_{n/4}J_{n/4} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \end{pmatrix} = \\
&= I_2 \otimes \Gamma_{n/2},
\end{aligned}$$

where the first two equalities have been obtained using (4.7) and lemma 4.4 and the last one using the inductive hypothesis. ∎

At this point we are ready for studying the action of Γ_n at bit level.

Theorem 4.6 *Let $n = 2^d$, let $k = (b_d \cdots b_1)_2$ and consider the permutation $g_n(k)$ such that*

$$[\Gamma_n \mathbf{x}]_k = x_{g_n(k)}.$$

Then

$$g_n((b_d \cdots b_1)_2) = (b_1\ (b_1 \text{ xor } b_2) \cdots (b_1 \text{ xor } b_2 \text{ xor } \cdots \text{ xor } b_d))_2.$$

Proof. We proceed by induction. The case where $n = 4$ is easy. Thus, let $n > 4$ set $\mathbf{y} = \Psi_n \mathbf{x}$ and consider the permutation $g'_n(k)$ such that

$$[\begin{pmatrix} \Gamma_{n/2} & O \\ O & \Gamma_{n/2} \end{pmatrix} \mathbf{y}]_k = y_{g'_n(k)}.$$

We have

$$g'_n((b_d \cdots b_1)_2) = (b_d\, b'_1 \cdots b'_{d-1})_2, \tag{4.8}$$

where, by virtue of the inductive hypothesis

$$(b'_1 \cdots b'_{d-1})_2 = g_{n/2}((b_{d-1} \cdots b_1)_2) = (b_1\ (b_1 \text{ xor } b_2) \cdots (b_1 \text{ xor } b_2 \text{ xor } \cdots \text{ xor } b_{d-1}))_2. \tag{4.9}$$

Now, from $y_k = [\Psi_n \mathbf{x}]_k = x_{p_n(k)}$ (see lemma 4.3) we find $y_{g'_n(k)} = x_{p_n(g'_n(k))}$. Thus $g_n(k) = p_n(g'_n(k))$ and the thesis follows from relations (4.8), (4.9) and lemma 4.3. ∎

4.2.3 The permutation Σ_n.

Let $n = 2^d$ with $d \geq 1$. The permutation Σ_n is defined as

$$\begin{cases} \Sigma_1 = (1) \\ \Sigma_n = \begin{pmatrix} \Sigma_{n/2} & O \\ O & \Gamma_{n/2} \end{pmatrix} \Upsilon_n \end{cases}. \tag{4.10}$$

As an example if $\mathbf{x} \in \mathsf{C}^8$ we have

$$\Sigma_8 \mathbf{x} = (x_0\ x_4\ x_2\ x_6\ x_1\ x_7\ x_3\ x_5)^T.$$

We met the permutation Σ_n under another form (4.1). Let us prove the equivalence of the two forms.

Theorem 4.7 *For $n = 2^d$ with $d \geq 1$ we have*

$$\Sigma_n = \left(\Sigma_{\frac{n}{2}} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \middle| \, \Sigma_{\frac{n}{2}} K_{\frac{n}{2}} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right).$$

Proof. We proceed by induction on n. For $n = 2$ the thesis obvious. For $n \geq 4$, we observe that

$$\Upsilon_n = \left(\begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix} \otimes I_{n/4} \Upsilon_{n/2} \middle| \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \otimes I_{n/4} \Upsilon_{n/2} \right).$$

At this point it is possible to proceed as in theorem 4.5. The details are omitted for brevity. ∎

The action of Σ_n at bit level is now easily established.

Theorem 4.8 *Let $n = 2^d$, let $k = (b_d \cdots b_1)_2$ and consider the permutation $s_n(k)$ such that*

$$[\Sigma_n \mathbf{x}]_k = x_{s_n(k)}.$$

Then $s_n(0) = 0$ and if $b_d = b_{d-1} = \cdots = b_{p+1} = 0$ and $b_p = 1$ then $s_n(k) = (c_1 \cdots c_{p-1}\, b_p \cdots b_d)_2$ where $(c_1 \cdots c_{p-1})_2 = g_{2^{p-1}}((b_{p-1} \cdots b_1)_2)$ (see Theorem 4.6), so that

$$s_n(k) = (b_1\ (b_1 \operatorname{xor} b_2) \cdots (b_1 \operatorname{xor} \cdots \operatorname{xor} b_{p-1})\ b_p\ \cdots\ b_d).$$

Proof. We proceed by induction. The case where $n = 2$ is immediate. Let us assume $n \geq 4$. We set $\mathbf{y} = \Upsilon_n \mathbf{x}$ and we consider the permutation $s'_n(k)$ such that

$$[\begin{pmatrix} \Sigma_{n/2} & O \\ O & \Gamma_{n/2} \end{pmatrix} \mathbf{y}]_k = y_{s'_n(k)}.$$

We have

$$s'_n((b_d \cdots b_1)_2) = (b_d\, b'_1 \cdots b'_{d-1})_2,$$

where

$$(b'_1 \cdots b'_{d-1})_2 = \begin{cases} s_{n/2}((b_{d-1} \cdots b_1)_2) & \text{if } b_d = 0 \\ g_{n/2}((b_{d-1} \cdots b_1)_2) & \text{if } b_d = 1 \end{cases}. \tag{4.11}$$

By the inductive hypothesis

$$s_{n/2}((b_{d-1} \cdots b_1)_2) = \begin{cases} (0 \cdots 0)_2 & \text{if } (b_{d-1} \cdots b_1)_2 = 0 \\ (c_1 \cdots c_{p-1}\, b_p \cdots b_{d-1})_2 & \text{if } b_{d-1} = \cdots = b_{p+1} = 0 \text{ and } b_p = 1 \end{cases}, \tag{4.12}$$

where $(c_1 \cdots c_{p-1})_2 = g_{2^{p-1}}((b_{p-1} \cdots b_1)_2)$. Now, being $\mathbf{y} = \Upsilon_n \mathbf{x}$ we have $y_k = [\Upsilon_n \mathbf{x}]_k = x_{u_n(k)}$ (see Lemma 4.3). Hence $y_{s'_n(k)} = x_{u_n(s'_n(k))}$. This yields

$$s_n(k) = u_n(s'_n(k)) = (b'_1 \cdots b'_{d-1}\, b_d)_2. \tag{4.13}$$

From (4.11), (4.12) and (4.13) we get the thesis. ∎

4.3 A factorization of F_n.

Having a precise description of the permutation Σ_n, our aim is now to prove a special factorization of the Fourier matrix F_n that turns out to be intimately connected with Bruun's algorithm, as we will see in the next section. More precisely, we will show that, if $n = 2^d$ then $\Sigma_n F_n = A_1 \cdots A_d$, where the A_k, for $k = 1 : d$, are block diagonal matrices whose diagonal blocks will be called *Bruun butterfly matrices*. We will first define the Bruun butterfly matrices, state our main theorem and illustrate it by an example. Then we will go into the details of the proof.

4.3.1 The Bruun butterfly matrices, the factorization and an example.

The formal definition of the Bruun butterfly matrices follows.

Definition 4.9 Let $n = 2^d$ with $d \geq 2$, $m = 2^k$ with $k = 1\!:\!d$, $r = n/m$, and $p = 0\!:\!r-1$. We call Bruun butterfly matrices the $m \times m$ matrices $B^{(n)}_{m,p}$ defined as

$$B^{(n)}_{m,p} \equiv \begin{cases} F_2 \otimes I_{m/2} & \text{if } p = 0 \\ \begin{pmatrix} 1 & \alpha & 1 & 0 \\ 0 & 1 & \alpha & 1 \\ 1 & -\alpha & 1 & 0 \\ 0 & 1 & -\alpha & 1 \end{pmatrix} \otimes I_{m/4}, \quad \alpha = 2\cos\frac{p\pi}{2r} & \text{if } p \neq 0 \text{ and } m > 2 \\ \begin{pmatrix} 1 & -\bar{\omega}^p_n \\ 1 & -\omega^p_n \end{pmatrix} & \text{if } p \neq 0 \text{ and } m = 2. \end{cases}$$

The announced factorization is as follows.

Theorem 4.10 *With the notations of Definition 4.9 we have*

$$\Sigma_n F_n = A_1 \cdots A_d, \tag{4.14}$$

where for $k = 1\!:\!d$

$$A_k = (\Sigma_r \otimes I_m) \begin{pmatrix} B^{(n)}_{m,0} & & & O \\ & B^{(n)}_{m,1} & & \\ & & \ddots & \\ O & & & B^{(n)}_{m,r-1} \end{pmatrix} (\Sigma_r^T \otimes I_m).$$

As an example it is worth to consider the case where $n = 8$.

Example 4.11 Instead of displaying the whole factorization at once, we take into account that $\Sigma_8 = \begin{pmatrix} \Upsilon_4 & O \\ O & \Psi_4 \end{pmatrix} \Upsilon_8$ (see (4.10)) and we apply the Σ_8 to F_8 in two steps. We have

$$\Upsilon_8 F_8 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -\mathbf{i} & -1 & \mathbf{i} & 1 & -\mathbf{i} & -1 & \mathbf{i} \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & \mathbf{i} & -1 & -\mathbf{i} & 1 & \mathbf{i} & -1 & -\mathbf{i} \\ 1 & \omega & -\mathbf{i} & \omega^3 & -1 & \omega^5 & \mathbf{i} & \omega^7 \\ 1 & \omega^3 & \mathbf{i} & \omega & -1 & \omega^7 & -\mathbf{i} & \omega^5 \\ 1 & \omega^5 & -\mathbf{i} & \omega^7 & -1 & \omega & \mathbf{i} & \omega^3 \\ 1 & \omega^7 & \mathbf{i} & \omega^5 & -1 & \omega^3 & -\mathbf{i} & \omega \end{pmatrix},$$

where $\omega = \omega_8 = e^{-\mathbf{i}\frac{2\pi}{8}}$. Setting

$$W = \begin{pmatrix} 1 & \omega & -\mathbf{i} & \omega^3 \\ 1 & \omega^3 & \mathbf{i} & \omega \\ 1 & \omega^5 & -\mathbf{i} & \omega^7 \\ 1 & \omega^7 & \mathbf{i} & \omega^5 \end{pmatrix},$$

we have

$$\Upsilon_8 F_8 = \begin{pmatrix} F_4 & F_4 \\ W & -W \end{pmatrix} = \begin{pmatrix} F_4 & O \\ O & W \end{pmatrix} F_2 \otimes I_4 = \begin{pmatrix} F_4 & O \\ O & W \end{pmatrix} B^{(8)}_{8,0}.$$

Analogously we have

$$\Upsilon_4 F_4 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & -\mathbf{i} & -1 & \mathbf{i} \\ 1 & \mathbf{i} & -1 & -\mathbf{i} \end{pmatrix} = \begin{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} & O \\ O & \begin{pmatrix} 1 & -\mathbf{i} \\ 1 & \mathbf{i} \end{pmatrix} \end{pmatrix} F_2 \otimes I_2 = \begin{pmatrix} B^{(8)}_{2,0} & O \\ O & B^{(8)}_{2,2} \end{pmatrix} B^{(8)}_{4,0}.$$

On the other hand we calculate $\Psi_4 W$ and we find

$$\Psi_4 W = \begin{pmatrix} 1 & \omega & -\mathbf{i} & \omega^3 \\ 1 & \omega^7 & \mathbf{i} & \omega^5 \\ 1 & \omega^3 & \mathbf{i} & \omega \\ 1 & \omega^5 & -\mathbf{i} & \omega^7 \end{pmatrix} = \begin{pmatrix} \begin{pmatrix} 1 & -\bar{\omega} \\ 1 & -\omega \end{pmatrix} & \begin{pmatrix} 1 & -\bar{\omega} \\ 1 & -\omega \end{pmatrix} \\ \begin{pmatrix} 1 & -\bar{\omega}^3 \\ 1 & -\omega^3 \end{pmatrix} & \begin{pmatrix} 1 & -\bar{\omega}^3 \\ 1 & -\omega^3 \end{pmatrix} \end{pmatrix} + \begin{pmatrix} 0 & \sqrt{2}\begin{pmatrix} 1 & -\bar{\omega} \\ 1 & -\omega \end{pmatrix} & 0 \\ 0 & -\sqrt{2}\begin{pmatrix} 1 & -\bar{\omega}^3 \\ 1 & -\omega^3 \end{pmatrix} & 0 \end{pmatrix}.$$

Putting it another way

$$\Psi_4 W = \begin{pmatrix} B_{2,1}^{(8)} & O \\ O & B_{2,3}^{(8)} \end{pmatrix} \begin{pmatrix} 1 & \alpha & 1 & 0 \\ 0 & 1 & \alpha & 1 \\ 1 & -\alpha & 1 & 0 \\ 0 & 1 & -\alpha & 1 \end{pmatrix},$$

where $\alpha = 2\cos\frac{\pi}{4} = \sqrt{2}$, i.e.,

$$\Psi_4 W = \begin{pmatrix} B_{2,1}^{(8)} & O \\ O & B_{2,3}^{(8)} \end{pmatrix} B_{4,1}^{(8)}.$$

Collecting everything we find what claimed

$$\Sigma_8 F_8 = \begin{pmatrix} B_{2,0}^{(8)} & & & O \\ & B_{2,2}^{(8)} & & \\ & & B_{2,1}^{(8)} & \\ O & & & B_{2,3}^{(8)} \end{pmatrix} \begin{pmatrix} B_{4,0}^{(8)} & O \\ O & B_{4,1}^{(8)} \end{pmatrix} B_{8,0}^{(8)}.$$

4.3.2 Proof of theorem 4.10.

In example 4.11 we used the relation $\Sigma_8 = \begin{pmatrix} \Upsilon_4 & O \\ O & \Psi_4 \end{pmatrix} \Upsilon_8$. Proceding by induction it is easy to show that, if $n = 2^d$ then

$$\Sigma_n = \Sigma_n^{(2)} \cdots \Sigma_n^{(d)} \tag{4.15}$$

where, with $k = 2:d$ and $m = 2^k$

$$\Sigma_n^{(k)} = \begin{pmatrix} \Upsilon_m & & & O \\ & \Psi_m & & \\ & & \ddots & \\ O & & & \Psi_m \end{pmatrix}.$$

Our strategy will be to compute first $\Upsilon_n F_n$, then $\Sigma_n^{(d-1)} \Upsilon_n F_n$ and so on. In order to do this we need a detailed description of certain submatrices of the Fourier matrix that arise during the process.

Definition 4.12 Let $n = 2^d$, $m = 2^k$ with $k = 1:d-1$, $r = n/m$ and $p = 1:2:r-1$. We define the matrix $W_{m,p}^{(n)} \in \mathsf{C}^{m\times m}$ as

$$W_{m,p}^{(n)} \equiv (\omega_n^{v_{m,p}^{(n)}(i) u_m^{(n)}(j)}), \qquad i, j = 0:m-1,$$

where

$$v_{m,p}^{(n)}(i) = \begin{cases} p + ir & \text{if } i \text{ is even} \\ r - p + ir & \text{if } i \text{ is odd} \end{cases}, \qquad u_m^{(n)}(j) = \begin{cases} j & \text{if } j = 0:m/2-1 \\ n/2 - m + j & \text{if } j = m/2:m-1 \end{cases}.$$

Moreover, when $p = 2:2:r-1$ we define

$$W_{m,p}^{(n)} = W_{m,p/q}^{(n/q)},$$

where q is a power of 2 and is such that p/q is an odd integer. Finally, if $p = 0$ we set $W_{m,0}^{(n)} = F_m$.

Example 4.13 From definition 4.12 we have $W^{(n)}_{n/2,1} = (\omega_m^{(2i+1)j})$, i, $j = 0{:}n/2-1$. In fact, $u^{(n)}_{n/2}(j) = j$ for $j = 0\,{:}\,n/2-1$, and $v^{(n)}_{n/2,1}(i) = 2i+1$ for $i = 0\,{:}\,n/2-1$. It is also easy to check that $W^{(n)}_{2,p} = B^{(n)}_{2,p}$, for $p = 0\,{:}\,r-1$.

Concerning $\Upsilon_n F_n$, we have a first simple result.

Lemma 4.14 *With the notations of definition 4.12 we have*

$$\Upsilon_n F_n = \begin{pmatrix} F_{n/2} & F_{n/2} \\ W^{(n)}_{n/2,1} & -W^{(n)}_{n/2,1} \end{pmatrix} = \begin{pmatrix} F_{n/2} & O \\ O & W^{(n)}_{n/2,1} \end{pmatrix} B^{(n)}_{n,0}.$$

Proof. Let A, B, C and $D \in \mathsf{C}^{n/2\times n/2}$ be such that

$$\Upsilon_n F_n = \begin{pmatrix} A & B \\ C & D \end{pmatrix}.$$

On the basis of lemma 4.2 (a), for i, $j = 0{:}n/2-1$ we have

$$\begin{array}{ll} [A]_{ij} = \omega_n^{2ij} = \omega_{n/2}^{ij} & \Rightarrow A = F_{n/2}, \\ [B]_{ij} = \omega_n^{2i(j+n/2)} = \omega_{n/2}^{ij} & \Rightarrow B = F_{n/2}, \\ [C]_{ij} = \omega_n^{(2i+1)j} & \Rightarrow C = W^{(n)}_{n/2,1}, \\ [D]_{ij} = \omega_n^{(2i+1)(j+n/2)} = -\omega_n^{(2i+1)j} & \Rightarrow D = -W^{(n)}_{n/2,1}, \end{array}$$

where, in the last two equalities we have used the fact that $W^{(n)}_{n/2,1} = (\omega_m^{(2i+1)j})$ (see example 4.13). ∎

It is clear that, in order to evaluate $\Sigma_n^{(d-1)}\Upsilon_n F_n$, we need to evaluate $\Psi_{n/2} W^{(n)}_{n/2,1}$. To this respect we have the following general result.

Lemma 4.15 *With the notations of definition 4.12 we have*

$$\Psi_m W^{(n)}_{m,p} = \begin{pmatrix} W^{(n)}_{m/2,p} & W^{(n)}_{m/2,p} \\ W^{(n)}_{m/2,2r-p} & W^{(n)}_{m/2,2r-p} \end{pmatrix} + 2\cos\frac{p\pi}{2r} \begin{pmatrix} \mathbf{O} & W^{(n)}_{m/2,p} & \mathbf{O} \\ \mathbf{O} & -W^{(n)}_{m/2,2r-p} & \mathbf{O} \end{pmatrix},$$

where $\mathbf{O}$ *denotes a* $m/2 \times m/4$ *block of zeros. This implies*

$$\Psi_m W^{(n)}_{m,p} = \begin{pmatrix} W^{(n)}_{m/2,p} & O \\ O & W^{(n)}_{m/2,2r-p} \end{pmatrix} B^{(n)}_{m,p}.$$

Proof. Once the thesis is established in the case where p is odd, the case where p is even follows as a trivial consequence. Thus, let us assume p to be odd. Let us set

$$\Psi_m W^{(n)}_{m,p} = \begin{pmatrix} M_0 \\ M_1 \end{pmatrix},$$

being $M_k \in \mathsf{C}^{m/2\times m}$ for $k = 0\,{:}\,1$. Observe that, by virtue of lemma 4.2 (2), we have

$$M_k = (\omega_n^{v^{(n)}_{m,p}(\sigma_k(i))u^{(n)}_m(j)}),$$

where

$$\sigma_0(i) = \begin{cases} 2i & \text{if } i \text{ is even} \\ 2i+1 & \text{if } i \text{ is odd} \end{cases}, \qquad \sigma_1(i) = \begin{cases} 2i+1 & \text{if } i \text{ is even} \\ 2i & \text{if } i \text{ is odd} \end{cases}.$$

Now let $M_k = (A_k|B_k|C_k|D_k)$ being A_k, B_k, C_k and $D_k \in \mathsf{C}^{m/2\times m/4}$. Observe that

$$\begin{array}{ll} A_k = (\omega_n^{v^{(n)}_{m,p}(\sigma_k(i))j}), & C_k = (\omega_n^{v^{(n)}_{m,p}(\sigma_k(i))(n/2-m/2+j)}), \\ B_k = (\omega_n^{v^{(n)}_{m,p}(\sigma_k(i))(m/4+j)}), & D_k = (\omega_n^{v^{(n)}_{m,p}(\sigma_k(i))(n/2-m/4+j)}). \end{array}$$

Claim: for $k = 0:1$ the following equality holds

$$(A_k|B_k|C_k|D_k) = (A_k|D_k|A_k|D_k) + (-1)^k 2\cos\frac{p\pi}{2r}(\mathbf{O}|A_k|D_k|\mathbf{O}). \tag{4.16}$$

Clearly (4.16) holds if and only if (a) $B_k = D_k + (-1)^k 2\cos\frac{p\pi}{2r}A_k$ and (b) $C_k = A_k + (-1)^k 2\cos\frac{p\pi}{2r}D_k$. The proof of (a) and (b) can be obtained by direct computation. We prove (a) and leave the proof of (b) to the reader. Equating the two matrices entrywise we obtain

$$\omega_n^{v_{m,p}^{(n)}(\sigma_k(i))(m/4+j)} = \omega_n^{v_{m,p}^{(n)}(\sigma_k(i))(n/2-m/4+j)} + (-1)^k 2\cos\frac{p\pi}{2r}\omega_n^{v_{m,p}^{(n)}(\sigma_k(i))j}.$$

Dividing by $\omega_n^{v_{m,p}^{(n)}(\sigma_k(i))j}$ yields

$$\omega_n^{v_{m,p}^{(n)}(\sigma_k(i))m/4} = \omega_n^{v_{m,p}^{(n)}(\sigma_k(i))(n/2-m/4)} + (-1)^k 2\cos\frac{p\pi}{2r}.$$

Now, since p has been assumed to be odd, we have that $v_{m,p}^{(n)}(\sigma_k(i))$ is odd as well, whatever are the values of k and i. Thus the preceding equality becomes

$$\omega_{4r}^{v_{m,p}^{(n)}(\sigma_k(i))} = -\omega_{4r}^{-v_{m,p}^{(n)}(\sigma_k(i))} + (-1)^k 2\cos\frac{p\pi}{2r}. \tag{4.17}$$

Equality (4.17) is identically satisfied. In fact, observe that

$$\omega_{4r}^{v_{m,p}^{(n)}(\sigma_k(i))} + \omega_{4r}^{-v_{m,p}^{(n)}(\sigma_k(i))} = 2\cos\frac{\pi}{2r}v_{m,p}^{(n)}(\sigma_k(i))$$

and

$$v_{m,p}^{(n)}(\sigma_0(i)) = \begin{cases} p+2ri & \text{if } i \text{ is even} \\ 2r-p+2ri & \text{if } i \text{ is odd} \end{cases}, \qquad v_{m,p}^{(n)}(\sigma_1(i)) = \begin{cases} 2r-p+2ri & \text{if } i \text{ is even} \\ p+2ri & \text{if } i \text{ is odd} \end{cases}, \tag{4.18}$$

so that $\cos\frac{\pi}{2r}v_{m,p}^{(n)}(\sigma_k(i)) = (-1)^k\cos\frac{\pi p}{2r}$. Thus we have the claimed identity (4.16). At this point observe that

$$(A_k|D_k) = ((\omega_n^{v_{m,p}^{(n)}(\sigma_k(i))j})|(\omega_n^{v_{m,p}^{(n)}(\sigma_k(i))(n/2-m/4+j)})) = (\omega_n^{v_{m,p}^{(n)}(\sigma_k(i))u_{m/2}^{(n)}(j)})$$

and that, by virtue of relation (4.18) we have $v_{m,p}^{(n)}(\sigma_0(i)) = v_{m/2,p}^{(n)}(i)$ and $v_{m,p}^{(n)}(\sigma_1(i)) = v_{m/2,2r-p}^{(n)}(i)$. Thus $(A_0|D_0) = W_{m/2,p}^{(n)}$ and $(A_0|D_0) = W_{m/2,2r-p}^{(n)}$. This ends the proof of the lemma. ∎

At this point we are able to prove theorem 4.10.

Proof of theorem 4.10. We have to prove that (see (4.14) and (4.15))

$$\Sigma_n^{(2)}\cdots\Sigma_n^{(d)}F_n = A_1\cdots A_d.$$

Our strategy will be to prove by induction on $k = d-1:-1:1$ that

$$\Sigma_n^{(k+1)}\cdots\Sigma_n^{(d)}F_n = (\Sigma_r\otimes I_m)\mathrm{Diag}(W_{m,p}^{(n)})(\Sigma_r^T\otimes I_m)A_{k+1}\cdots A_d, \tag{4.19}$$

where $m = 2^k$, $r = n/m$ and $p = 0:r-1$. Clearly the (4.19) for $k = 1$ is our thesis, since $W_{2,p}^{(n)} = B_{2,p}^{(n)}$ (see example 4.13). Observe that equation (4.19) for $k = d-1$ has been established in lemma 4.14. Now, let us assume the (4.19). Multiplying by $\Sigma_n^{(k)}$ we have

$$\Sigma_n^{(k)}\cdots\Sigma_n^{(d)}F_n = \Sigma_n^{(k)}(\Sigma_r\otimes I_m)\mathrm{Diag}(W_{m,p}^{(n)})(\Sigma_r^T\otimes I_m)A_{k+1}\cdots A_d.$$

Observe that

$$\Sigma_n^{(k)} = \begin{pmatrix} \Upsilon_m & \\ & I_{r-1}\otimes\Psi_m \end{pmatrix}, \qquad \Sigma_r\otimes I_m = \begin{pmatrix} I_m & \\ & P\otimes I_m \end{pmatrix},$$

where P is an appropriate permutation matrix. This implies that $\Sigma_n^{(k)}$ and $\Sigma_r \otimes I_m$ must commute. From lemmma 4.15 follows

$$\Sigma_n^{(k)}\mathrm{Diag}(W_{m,p}^{(n)}) = \mathrm{Diag}(\begin{pmatrix} W_{m/2,p}^{(n)} & O \\ O & W_{m/2,2r-p}^{(n)} \end{pmatrix})\mathrm{Diag}(B_{m,p}^{(n)}).$$

Thus

$$\begin{aligned}(\Sigma_r \otimes I_m)\Sigma_n^{(k)}\mathrm{Diag}(W_{m,p}^{(n)})(\Sigma_r^T \otimes I_m) = \\ = \quad (\Sigma_r \otimes I_m)\mathrm{Diag}(\begin{pmatrix} W_{m/2,p}^{(n)} & O \\ O & W_{m/2,2r-p}^{(n)} \end{pmatrix})(\Sigma_r^T \otimes I_m)A_k.\end{aligned}$$

From the last equality, using the fact that

$$\Sigma_r \otimes I_m = (\Sigma_r \otimes I_2) \otimes I_{m/2} = (\Sigma_{2r} \otimes I_{m/2})(\begin{pmatrix} I_{r/2} \otimes (1\ 0) \\ K_{r/2} \otimes (0\ 1) \end{pmatrix} \otimes I_{m/2}),$$

(see (4.1)) we get

$$(\Sigma_r \otimes I_m)\Sigma_n^{(k)}\mathrm{Diag}(W_{m,p}^{(n)})(\Sigma_r^T \otimes I_m) = (\Sigma_{2r} \otimes I_{m/2})\mathrm{Diag}(W_{m/2,p}^{(n)})(\Sigma_{2r}^T \otimes I_{m/2})A_k.$$

The theorem is proved. ∎

4.4 The Bruun framework.

The results obtained so far allow us to give a matrix description of both DIT and DIF versions of Bruun's algorithm. The key points connected with their implementation will emerge. It turns out that the two algorithms can conveniently be combined so that they provide a method for computing the cyclic convolution of two data sequences. This method works in place both for complex and for real inputs. In addition it does not require the execution of any permutation. Methods of this kind are well known for complex data sequences. However, we stress that to devise FFT-based convolution methods for real sequences working in place and, at the same time, avoiding data reordering is a nontrivial problem studied by various authors [59, 6, 25, 36, 68].

4.4.1 The DIT and DIF Bruun factorizations and algorithms.

The following result is an easy consequence of theorem 4.10.

Theorem 4.16 *Let $n = 2^d$ and let A_k, for $k = 1:d$, be as in theorem 4.10. Then*

$$F_n = \Sigma_n^T A_1 \cdots A_d. \tag{4.20}$$

Moreover

$$F_n = A_d^T \cdots A_1^T \Sigma_n. \tag{4.21}$$

Proof. In theorem 4.10 it is shown that $\Sigma_n F_n = A_1 \cdots A_d$. Multiplying both members by Σ_n^T we obtain equality (4.20). Equality (4.21) is obtained by equality (4.20) by simply transposing both members and taking into account that $F_n = F_n^T$. ∎

The two factorizations of the Fourier matrix (4.20) and (4.21) will be called *DIT and DIF Bruun's factorizations* respectively. These factorizations underlie the DIT and DIF versions of Bruun's algorithm. Consider for example the DIT factorization (4.20) and let $\mathbf{x} \in \mathsf{C}^n$. Then $F_n\mathbf{x} = \Sigma_n^T A_1 \cdots A_d\mathbf{x}$, leading to the following high-level representation of the DIT version of the algorithm.

Algorithm 1 (DIT Bruun) *Given $\mathbf{x} \in \mathsf{C}^n$ the following algorithm computes $F_n\mathbf{x}$. A workspace $\mathbf{y} \in \mathsf{C}^n$ is used.*

for $k = d:-1:1$
  $\mathbf{x} \leftarrow A_k\mathbf{x}$
end
$\mathbf{y} \leftarrow \Sigma_n^T\mathbf{x}$
$\mathbf{x} \leftarrow \mathbf{y}$

An analogous representation can be derived for the DIF version.

Algorithm 2 (DIF Bruun) *Given* $\mathbf{x} \in \mathsf{C}^n$ *the following algorithm computes* $F_n\mathbf{x}$. *A workspace* $\mathbf{y} \in \mathsf{C}^n$ *is used.*

$\mathbf{y} \leftarrow \Sigma_n\mathbf{x}$
$\mathbf{x} \leftarrow \mathbf{y}$
for $k = 1:d$
  $\mathbf{x} \leftarrow A_k^T\mathbf{x}$
end

The implementation of algorithms 1 and 2, has to face two problems.

a) The permutation Σ_n seems not to be efficiently computable in place.

b) The updates $\mathbf{x} \leftarrow A_k\mathbf{x}$ ($\mathbf{x} \leftarrow A_k^T\mathbf{x}$) are computable in place (see section 4.4.2 below) but the arguments of the cosines that enter in the definition of the diagonal blocks of the matrices A_k, are scrambled according to Σ_r (see theorem 4.10), so that their computation at run time implies a nontrivial overhead.

In the view of a) and b) the isolated use of one of the two algorithms is not recommended, particularly in a situation where storage is limited. If extra memory space is available then it is worth considering the use of the *autosort Bruun algorithm* that will be described in section 4.5.

However, there are situations in which the problems pointed out can be avoided. One of these is the computation of the cyclic convolution of two data sequences. Remember that convolution theorem states

$$n\,\mathbf{x} * \mathbf{y} = \bar{F}_n(F_n\mathbf{x} \circ F_n\mathbf{y}),$$

where $*$ denotes cyclic convolution and $\circ$ entrywise vector product. Clearly three transforms have to be computed, so that it makes no sense to compute three times the cosines that appear in the matrices A_k. A better choice is to precompute and store them, in the order in which they are needed, in a suitable vector. In this way the problem raised in b) vanishes. Concerning a) observe that, abbreviating $F_n = \Sigma_n^T A_1 \cdots A_d$ with $F_n = \Sigma_n^T A$ and taking into account that $\bar{F}_n = F_n^H = A^H\Sigma_n$ we have

$$\begin{aligned} n\,\mathbf{x} * \mathbf{y} &= A^H\Sigma_n(\Sigma_n^T A\mathbf{x} \circ \Sigma_n^T A\mathbf{y}) = \\ &= A^H(A\mathbf{x} \circ A\mathbf{y}), \end{aligned}$$

so that the computation of Σ_n can be completely avoided.

The above method for convolution computation can be executed in place, both for complex and for real data sequences. Devising FFT-based methods for computing the cyclic convolution of real data that work in place and do not require data shuffling is not trivial. For an interesting discussion about this point the reader can see [6]. In [59, 25, 68] a good introduction to the problem of transforming real sequences and computing their convolution can be found. In particular in [59] it is pointed out that the DIF radix-2 FFT of a real sequence cannot be computed in place while DIF split-radix FFT can. Thus the split-radix based convolution method works in place and does not require data permutations. Due to the low arithmetic cost of the split-radix algorithm, the split-radix based convolution method is the best actually available on most sequential machines, as long as sequences having lengths being a power of 2 are concerned. Finally, we notice that in [36] an interesting convolution method that involves both Fourier and Hartley transform is presented.

In order to obtain a more refined description of the convolution algorithm based on Bruun's algorithm and to set the background for further developments, in the next section we will discuss in some detail the implementation of the update $\mathbf{x} \leftarrow A_k\mathbf{x}$.

4.4.2 Computing the update $\mathbf{x} \leftarrow A_k\mathbf{x}$.

Remember that (theorem 4.10)

$$A_k = (\Sigma_r \otimes I_m) \begin{pmatrix} B_{m,0}^{(n)} & & & O \\ & B_{m,1}^{(n)} & & \\ & & \ddots & \\ O & & & B_{m,r-1}^{(n)} \end{pmatrix} (\Sigma_r^T \otimes I_m).$$

Thus, the computation of $\mathbf{x} \leftarrow A_k\mathbf{x}$ can be handled as follows

$m \leftarrow 2^k$
$r \leftarrow n/m$
for $p = 0 : r-1$
  $\mathbf{x}(mp : m(p+1)-1) \leftarrow B_{m,s_r(p)}^{(n)}\mathbf{x}(mp : m(p+1)-1)$
end

where s_r is the permutation over $\{0, \cdots, r-1\}$ described in theorem 4.8. Our problem is to specify how to work out an update of the form $\mathbf{x} \leftarrow B_{m,p}^{(n)}\mathbf{x}$. According to definition 4.9 we have to face three cases.

Case where $p = 0$. Let $\mathbf{y}, \mathbf{x} \in \mathsf{C}^m$ such that $\mathbf{y} = B_{m,0}^{(n)}\mathbf{x}$. Partitioning both $\mathbf{x}$ and $\mathbf{y}$ into two subvectors of equal length yields

$$\begin{pmatrix} \mathbf{y}_U \\ \mathbf{y}_L \end{pmatrix} = \begin{pmatrix} I_{m/2} & I_{m/2} \\ I_{m/2} & -I_{m/2} \end{pmatrix} \begin{pmatrix} \mathbf{x}_U \\ \mathbf{x}_L \end{pmatrix} = \begin{pmatrix} \mathbf{x}_U + \mathbf{x}_L \\ \mathbf{x}_U - \mathbf{x}_L \end{pmatrix}.$$

We have to handle 2 additions of vectors in $\mathsf{C}^{m/2}$ that turn into 4 additions of vectors in $\mathsf{R}^{m/2}$. The count halves if $\mathbf{x} \in \mathsf{R}^m$. At the scalar level, since $B_{m,0}^{(n)} = F_2 \otimes I_{m/2}$, we have, by virtue of lemma 4.1 2, $\mathbf{y}_{m/2\times 2} = \mathbf{x}_{m/2\times 2}F_2^T$. This yields the following algorithm

$m_* \leftarrow m/2$
for $j = 0 : m_* - 1$
  $\begin{pmatrix} y_j \\ y_{j+m_*} \end{pmatrix} \leftarrow \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} x_j \\ x_{j+m_*} \end{pmatrix}$
end

It is immediate to obtain an algorithm that performs the computation working in place.

$m_* \leftarrow m/2$
for $j = 0 : m_* - 1$
  $w \leftarrow x_{j+m_*}$
  $x_{j+m_*} \leftarrow x_j - w$
  $x_j \leftarrow x_j + w$
end

Case where $p \neq 0$ and $m \geq 4$. As before let $\mathbf{y} = B_{m,p}^{(n)}\mathbf{x}$. Setting $\mathbf{x}_C = \mathbf{x}(m/4 : 3m/4-1)$ yields

$$\begin{pmatrix} \mathbf{y}_U \\ \mathbf{y}_L \end{pmatrix} = \left(\begin{pmatrix} I_{m/2} & I_{m/2} \\ I_{m/2} & I_{m/2} \end{pmatrix} + \alpha \begin{pmatrix} \mathbf{O} & I_{m/2} & \mathbf{O} \\ \mathbf{O} & -I_{m/2} & \mathbf{O} \end{pmatrix} \right) \mathbf{x} = \begin{pmatrix} \mathbf{x}_U + \mathbf{x}_L + \alpha\mathbf{x}_C \\ \mathbf{x}_U + \mathbf{x}_L - \alpha\mathbf{x}_C \end{pmatrix},$$

where $\alpha = 2\cos\frac{p\pi}{2r}$ and $\mathbf{O}$ denotes a $m/2 \times m/4$ block of zeros. Obviously the computation of $\mathbf{y}$ takes 8 operations on vectors in $\mathsf{R}^{m/2}$. The count halves if $\mathbf{x} \in \mathsf{R}^m$.

At the scalar level since $B_{m,p}^{(n)} = \begin{pmatrix} 1 & \alpha & 1 & 0 \\ 0 & 1 & \alpha & 1 \\ 1 & -\alpha & 1 & 0 \\ 0 & 1 & -\alpha & 1 \end{pmatrix} \otimes I_{m/4}$ we have, by using again lemma 4.1 2, $\mathbf{y}_{m/4\times 4} = \mathbf{x}_{m/4\times 4} \begin{pmatrix} 1 & \alpha & 1 & 0 \\ 0 & 1 & \alpha & 1 \\ 1 & -\alpha & 1 & 0 \\ 0 & 1 & -\alpha & 1 \end{pmatrix}^T$. This yields the algorithm

$m_* \leftarrow m/2$
$m_{**} \leftarrow m/4$
for $j = 0 : m_{**} - 1$

$$\begin{pmatrix} y_j \\ y_{j+m_{**}} \\ y_{j+m_*} \\ y_{j+m_*+m_{**}} \end{pmatrix} \leftarrow \begin{pmatrix} 1 & \alpha & 1 & 0 \\ 0 & 1 & \alpha & 1 \\ 1 & -\alpha & 1 & 0 \\ 0 & 1 & -\alpha & 1 \end{pmatrix} \begin{pmatrix} x_j \\ x_{j+m_{**}} \\ x_{j+m_*} \\ x_{j+m_*+m_{**}} \end{pmatrix}$$

end

At this point it is easy to obtain an algorithm that computes $B^{(n)}_{m,p}\mathbf{x}$ working in place.

$r \leftarrow n/m$
$m_* \leftarrow m/2$
$m_{**} \leftarrow m/4$
$\alpha \leftarrow 2\cos\frac{\pi p}{2r}$
for $j = 0 : m_{**} - 1$
  $w \leftarrow \alpha * x_{j+m_{**}}$
  $z \leftarrow \alpha * x_{j+m_*}$
  $x_j \leftarrow x_j + x_{j+m_*}$
  $x_{j+m_*} \leftarrow x_j - w$
  $x_j \leftarrow x_j + w$
  $x_{j+m_{**}} \leftarrow x_{j+m_{**}} + x_{j+m_*+m_{**}}$
  $x_{j+m_*+m_{**}} \leftarrow x_{j+m_{**}} - z$
  $x_{j+m_{**}} \leftarrow x_{j+m_{**}} + z$
end

Case where $p \neq 0$ and $m = 2$. The computation of $\mathbf{y} = B^{(n)}_{2,p}\mathbf{x}$ should be handled with care. Remember that $B^{(n)}_{2,p}\mathbf{x} = \begin{pmatrix} 1 & -\bar{\omega}^p_n \\ 1 & -\omega^p_n \end{pmatrix}$ and set $\omega^p = \cos\theta_p - \mathbf{i}\sin\theta_p$. Observe that

$$\begin{pmatrix} y_0 \\ y_1 \end{pmatrix} = \begin{pmatrix} 1 & -\bar{\omega}^p_n \\ 1 & -\omega^p_n \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \end{pmatrix} = \begin{pmatrix} x_0 - x_1\cos\theta_p - \mathbf{i}x_1\sin\theta_p \\ x_0 - x_1\cos\theta_p + \mathbf{i}x_1\sin\theta_p \end{pmatrix}.$$

Thus, if $\mathbf{x} \in \mathsf{C}^2$, $\mathbf{y}$ can be computed with 4 real multiplications and 6 real additions. Moreover, if $\mathbf{x} \in \mathsf{R}^2$ then $\mathbf{y}$ has conjugate entries so that only one of them need to be computed. This amounts to two real multiplications and one real addition.

4.5 An autosort Bruun framework.

The problem of the adaptation of FFT algorithms, in particular radix-2 FFT, to vector computers has been discussed by various authors [64, 65, 66, 68]. Due to the high cost of data permutations on vector computers, particular interest have received certain variants of radix-2 FFT known as Stockham autosort algorithms [68]. These algorithms in some sense "scatter" the bit reversing permutation, characteristic of radix-2 FFT, so that it is performed in a certain number of steps as computation proceeds.

The implementation of Bruun's algorithm on a vector computer is attractive for two reasons. The first is that it leads to a certain saving in terms of vector operations with respect to radix-2 FFT. In fact when $m > 2$ we know from section 4.4.2 that it is possible to compute the product $B^{(n)}_{m,p}\mathbf{x}$ with 8 real $m/2$-length vector operations if $\mathbf{x}$ has complex entries and with 4 real $m/2$-length vector operations if $\mathbf{x}$ has real entries. For the radix-2 FFT the counts are 10 and 5 real $m/2$-length vector operations respectively [68]. The second is that, as for radix-2 FFT, we can devise an autosort variant of Bruun's algorithm. In this section we will discuss this variant.

4.5.1 An autosort Bruun factorization and algorithm.

Troughout this section we will use the notations introduced in definition 4.9 and theorem 4.10. Let us set

$$D_k = \begin{pmatrix} B^{(n)}_{m,0} & & & O \\ & B^{(n)}_{m,1} & & \\ & & \ddots & \\ O & & & B^{(n)}_{m,r-1} \end{pmatrix},$$

so that

$$A_k = (\Sigma_r \otimes I_m) D_k (\Sigma_r^T \otimes I_m). \tag{4.22}$$

In addition let us consider the permutation

$$\Phi_n = \left(\frac{I_{n/2} \otimes (1\ 0)}{K_{n/2} \otimes (0\ 1)} \right),$$

so that (see 4.1)

$$\Sigma_n = (\Sigma_{n/2} \otimes I_2) \Phi_n^T. \tag{4.23}$$

Theorem 4.17 *If $n = 2^d$ then*

$$F_n = S_1 \cdots S_d, \tag{4.24}$$

where, for $k = 1:d$,

$$S_k = (\Phi_{2r} \otimes I_{m/2}) D_k.$$

Proof. From theorem 4.10 we have $F_n = \Sigma_n^T A_1 A_2 \cdots A_d$. Using (4.22) we have

$$F_n = \Sigma_n^T (\Sigma_{n/2} \otimes I_2) D_1 ({\Sigma_{n/2}}^T \otimes I_2)(\Sigma_{n/4} \otimes I_4) D_2 ({\Sigma_{n/4}}^T \otimes I_4) \cdots (\Sigma_1 \otimes I_n) D_1 (\Sigma_1^T \otimes I_n).$$

Relation (4.23) yields

$$\begin{aligned} \Sigma_n^T (\Sigma_{n/2} \otimes I_2) &= \Phi_n \\ ({\Sigma_{n/2}}^T \otimes I_2)(\Sigma_{n/4} \otimes I_4) &= \Phi_{n/2} \otimes I_2, \end{aligned}$$

and, in general

$$(\Sigma_r^T \otimes I_m)(\Sigma_{r/2} \otimes I_{2m}) = \Phi_r \otimes I_m.$$

Thus we have

$$F_n = (\Phi_n \otimes I_1) D_1 (\Phi_{n/2} \otimes I_2) D_2 \cdots D_d$$

and the theorem is proved. ∎

We call the (4.24) *autosort Bruun factorization.* This factorization underlies the following

Algorithm 3 (autosort Bruun) *Given $\mathbf{x} \in \mathsf{C}^n$ the following algorithm computes $F_n \mathbf{x}$. A workspace $\mathbf{y} \in \mathsf{C}^n$ is used.*

for $k = d:-1:1$
  $\mathbf{y} \leftarrow S_k \mathbf{x}$
  $\mathbf{x} \leftarrow \mathbf{y}$
end

Notice how the Σ_n has been "scattered" between the marices D_k. This is typical of all autosort algorithms (e.g., Stokham algorithms). As a very kind side effect we have that the Bruun butterfly matrices $B^{(n)}_{m,p}$ appear in along the diagonal of the matrices D_k in natural order.

It is important to point out that the update $\mathbf{y} \leftarrow S_k \mathbf{x}$ can actually be computed in a single step. Let us examine it in some detail.

4.5.2 Computing the update $\mathbf{y} \leftarrow S_k\mathbf{x}$.

Since $S_k = (\Phi_{2r} \otimes I_{m/2})D_k$ let us examine the computations of $\mathbf{x} \leftarrow D_k\mathbf{x}$ and of $\mathbf{y} \leftarrow (\Phi_{2r} \otimes I_{m/2})\mathbf{x}$. Then we will proceed to merge them.

Let us assume $k \geq 2$ (the case where $k = 2$ can be easily worked out) and let us maintain the exposition at the vector level. As usual we set $m = 2^k$ and $r = n/m$. We partition $\mathbf{x}$ and $\mathbf{y}$ into $2r$ subvectors of dimension $m/2$, i.e., $\mathbf{x} = (\mathbf{x}_p)$ and $\mathbf{y} = (\mathbf{y}_p)$ with $p = 0 : 2r - 1$. From what we know from section 4.4.2 the computation of $\mathbf{x} \leftarrow D_k\mathbf{x}$ can be easily worked out

$$
\begin{array}{l}
m \leftarrow 2^k \\
r \leftarrow n/m \\
\begin{pmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \end{pmatrix} \leftarrow \begin{pmatrix} \mathbf{x}_0 + \mathbf{x}_1 \\ \mathbf{x}_0 - \mathbf{x}_1 \end{pmatrix}. \\
\textbf{for } p = 1 : r - 1 \\
\quad \alpha \leftarrow 2\cos\frac{p\pi}{2r} \\
\quad \begin{pmatrix} \mathbf{x}_{2p} \\ \mathbf{x}_{2p+1} \end{pmatrix} \leftarrow \begin{pmatrix} \mathbf{x}_{2p} + \mathbf{x}_{2p+1} + \alpha\mathbf{x}(pm + m/4 : pm + 3/4m) \\ \mathbf{x}_{2p} + \mathbf{x}_{2p+1} - \alpha\mathbf{x}(pm + m/4 : pm + 3/4m) \end{pmatrix} \\
\textbf{end}
\end{array}
$$

For computation of $\mathbf{y} \leftarrow (\Phi_{2r} \otimes I_{m/2})\mathbf{x}$ we have

$$
\begin{array}{l}
m \leftarrow 2^k \\
r \leftarrow n/m \\
\mathbf{y}_0 \leftarrow \mathbf{x}_0 \\
\mathbf{y}_r \leftarrow \mathbf{x}_1 \\
\textbf{for } p = 1 : r - 1 \\
\quad \mathbf{y}_p \leftarrow \mathbf{x}_{2p} \\
\quad \mathbf{y}_{2r-p} \leftarrow \mathbf{x}_{2p+1} \\
\textbf{end}
\end{array}
$$

Merging the two computations is now easy

$$
\begin{array}{l}
m \leftarrow 2^k \\
r \leftarrow n/m \\
\mathbf{y}_0 \leftarrow \mathbf{x}_0 + \mathbf{x}_1 \\
\mathbf{y}_r \leftarrow \mathbf{x}_0 - \mathbf{x}_1 \\
\textbf{for } p = 1 : r - 1 \\
\quad \alpha \leftarrow 2\cos\frac{p\pi}{2r} \\
\quad \mathbf{y}_p \leftarrow \mathbf{x}_{2p} + \mathbf{x}_{2p+1} + \alpha\mathbf{x}(pm + m/4 : pm + 3/4m) \\
\quad \mathbf{y}_{2r-p} \leftarrow \mathbf{x}_{2p} + \mathbf{x}_{2p+1} - \alpha\mathbf{x}(pm + m/4 : pm + 3/4m) \\
\textbf{end}
\end{array}
$$

4.6 Trigonometric transforms.

In section 3.5 we have presented BFHT, BFST and BFCT, i.e., fast algorithms for discrete Hartley, Sine and Cosine transforms (two different versions of DCT have been considered). The matrix factorization point of view is handy in order to make clear the relation existing among Bruun's algorithm and BFHT, BFST, BFCT. Actually, in this section we use DIT and DIF Bruun factorizations (4.20-4.21) in order to derive analogous factorizations for Hartley, Sine and Cosine matrices.

Observe that in factorizations (4.20-4.21) the matrices A_2,...,A_d have real entries. Thus from (4.20) we get

$$\mathrm{Re}(F_n) = \Sigma_n^T \mathrm{Re}(A_1)A_2 \cdots A_d, \tag{4.25}$$

$$\mathrm{Im}(F_n) = \Sigma_n^T \mathrm{Im}(A_1)A_2 \cdots A_d. \tag{4.26}$$

Clearly, analogous relations could be derived from DIF Bruun factorization. Exploiting this observation reduces our job to pick out a suitable submatrix from the real and/or imaginary part of a suitably dimensioned Fourier matrix.

Concerning Hartley matrix (see section 3.2) we have the following simple result.

Theorem 4.18 *We have*

$$H_n = \Sigma_n^T(\mathrm{Re}(A_1) - \mathrm{Im}(A_1))A_2 \cdots A_d, \tag{4.27}$$
$$H_n = A_d^T \cdots A_2^T(\mathrm{Re}(A_1^T) - \mathrm{Im}(A_1^T))\Sigma_n. \tag{4.28}$$

Proof. Since $H_n = \mathrm{Re}(F_n) - \mathrm{Im}(F_n)$, using (4.25-4.26) we get the (4.27). The (4.28) can be obtained by simply transposing both members of the (4.27). ∎

Clearly factorization (4.28) underlies BFHT such as it has been presented in section 3.5.

As a second example, we turn to the cosine matrix C_n. All we will say applies as well, with obvious changes, to the sine matrix S_n (see section 3.2). Observe that C_n is the $n \times n$ leading principal submatrix of F_{2n}, i.e., we have $C_n = (I_n\ O)\mathrm{Re}(F_{2n})\begin{pmatrix} I_n \\ O \end{pmatrix}$. Let us consider the DIF Bruun factorization of F_{2n}

$$F_{2n} = A_{d+1}^T \cdots A_1^T \Sigma_{2n},$$

being the A_k matrices of order $2n$. We recall that $A_{d+1} = F_2 \otimes I_n$, and we set

$$A_k \equiv \begin{pmatrix} U_k & O \\ O & L_k \end{pmatrix}, \qquad k = 1, \ldots, d.$$

Theorem 4.19 *We have*

$$C_n = U_d^T \cdots U_2^T \mathrm{Re}(U_1^T)\Sigma_{n/2} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} + L_d^T \cdots L_2^T \mathrm{Re}(L_1^T)\Gamma_{n/2} \otimes \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}. \tag{4.29}$$

Proof. We have

$$\begin{aligned}
C_n &= (I_n\ O)\mathrm{Re}(F_{2n})\begin{pmatrix} I_n \\ O \end{pmatrix} = \\
&= (I_n\ O)A_{d+1}^T A_d^T \cdots \mathrm{Re}(A_1^T)\Sigma_{2n}\begin{pmatrix} I_n \\ O \end{pmatrix} = \\
&= (I_n\ I_n)A_d^T \cdots \mathrm{Re}(A_1^T)\Sigma_{2n}\begin{pmatrix} I_n \\ O \end{pmatrix} = \\
&= (I_n\ I_n)A_d^T \cdots \mathrm{Re}(A_1^T)\begin{pmatrix} \Sigma_n & O \\ O & \Gamma_n \end{pmatrix}\left(\frac{I_n \otimes (1\,0)}{I_n \otimes (0\,1)}\right)\begin{pmatrix} I_n \\ O \end{pmatrix} = \\
&= \{U_d^T \cdots \mathrm{Re}(U_1^T)\Sigma_n \otimes (1\,0) + L_d^T \cdots \mathrm{Re}(L_1^T)\Gamma_n \otimes (0\,1)\}\begin{pmatrix} I_n \\ O \end{pmatrix}.
\end{aligned}$$

Now, using theorem 4.7, we have

$$\begin{aligned}
&\Sigma_n \otimes (1\,0)\begin{pmatrix} I_n \\ O \end{pmatrix} = \\
&= \left(\Sigma_{\frac{n}{2}} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix}\middle|\, \Sigma_{\frac{n}{2}} K_{\frac{n}{2}} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix}\right) \otimes (1\,0)\begin{pmatrix} I_n \\ O \end{pmatrix} = \\
&= \left(\Sigma_{\frac{n}{2}} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}\middle|\, \Sigma_{\frac{n}{2}} K_{\frac{n}{2}} \otimes \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}\right)\begin{pmatrix} I_n \\ O \end{pmatrix} = \\
&= \Sigma_{\frac{n}{2}} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}.
\end{aligned}$$

Analogously we find

$$\Gamma_n \otimes (0\,1) \begin{pmatrix} I_n \\ O \end{pmatrix} = \Gamma_{\frac{n}{2}} \otimes \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$$

and the thesis follows. ∎

Observe as in relation (4.29) the two matrices that appear at the second member contain respectively the even and odd columns of C_n. Relation (4.29) can be used in order to obtain a matrix description of algorithm BFCT-I. We do not work out the details here.

As a last example we briefly sketch a result concerning the Cosine matrix T_n. We have

$$T_n = (I_n\, O\, O\, O)\mathrm{Re}(F_{4n}) \begin{pmatrix} I_n \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \\ O \end{pmatrix}.$$

As before we consider the DIF Bruun factorization of F_{4n} and, with obvious notations we get the following result

Theorem 4.20 *We have*

$$T_n = (I_n\, O)L_{d+1}^T \cdots \mathrm{Re}(L_1^T)\Gamma_n \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

Proof. Left to the reader. ∎

4.7 On the stability of Bruun's algorithm.

In this section we study the behaviour of Bruun's algorithm with respect to the propagation of the roundoff error arising when the algorithm is carried out in fixed precision using floating point arithmetic. It turns out that the algorithm is stable. Before showing this, let us present some results that will serve both as an introduction and as a term of comparison.

4.7.1 Background.

In [20] a quantitative definition of the numerical stability of an algorithm $\mathcal{A}$ is proposed.

Definition 4.21 Given an algorithm $\mathcal{A}$, let $h = h(g)$ be the minimum number of digits in the floating point arithmetic sufficient to ensure g correct digits in the result. Then $\mathcal{A}$ is *numerically strongly stable* if $h(g) = g + O(\log n)$, where n is the number of input or output values, *numerically stable* if $h(g) = O(g \log n)$, and *numerically unstable* otherwise.

Now, let us recall a famous result due to Gentleman and Sande [40].

Theorem 4.22 *Let d and b be positive integers, let $n = 2^d$ and let $\mathbf{x}$, $\mathbf{y}$ a pair of n-dimensional complex vectors such that $\mathbf{y} = F_n\mathbf{x}$. Let us perform the computation using floating point arithmetic and fixed word length with b bit mantissa, so that $u = 2^{-b}$ is the machine rounding. Then:*

(a) *if $\tilde{\mathbf{y}}$ is the vector computed from $\mathbf{x}$ using the definition of DFT we have*

$$\frac{\|\tilde{\mathbf{y}} - \mathbf{y}\|_2}{\|\mathbf{y}\|_2} \leq u\, 8.5\, n^{1.5}; \tag{4.30}$$

(b) *if $\tilde{\mathbf{y}}$ is the vector computed from $\mathbf{x}$ using the radix-2 FFT we have*

$$\frac{\|\tilde{\mathbf{y}} - \mathbf{y}\|_2}{\|\mathbf{y}\|_2} \leq u\, 8.5\, \log n. \tag{4.31}$$

In the view of definition 4.21, both the $O(n^2)$ implementation of the DFT definition, and the $O(n \log n)$ radix-2 FFT are strongly stable algorithms, and the latter seems to be even more accurate than the former. This is surprising, since often fast algorithms have poor stability properties.

We notice that in [20] a backward rounding error analysis of the radix-2 FFT is performed and it is pointed out that its good stability properties can be exploited in order to compute the product of two polynomials with integer coefficients. This computation can be performed with a relatively low precision (slightly higher than the number of digits needed to represent the output) and, nevertheless, the exact result can still be obtained by rounding off the output to the nearest integer.

4.7.2 Roundoff error analysis for Bruun's algorithm.

We will use a simple model of floating point arithmetic. Specifically, we denote with $fl(\cdot)$ the result obtained by computing the expression between brackets in floating point arithmetic and we assume

$$fl(a \circ b) = (a \circ b)(1 + \epsilon), \qquad |\epsilon| < u = 2^{-b},$$

where $\circ$ stands for any arithmetic operation.

In what follows, we will consider the DIT Bruun's algorithm and we will make an heavy use of the DIT Bruun's factorization (see theorems 4.10 and 4.16). A similar analysis could be performed for the DIF version of the algorithm.

Lemma 4.23 *Let* $\mathbf{x}$ *be an* n*-dimensional complex vector and let* A_k*, for* $k = 1 : d$*, be the matrices defined in theorem 4.10. Then we have*

$$\|fl(A_k\mathbf{x}) - A_k\mathbf{x}\|_2 \dot{\leq} u\, 15\, \|\mathbf{x}\|_2, \tag{4.32}$$

where $\dot{\leq}$ *denotes inequality up to terms of order equal to or greater than* u^2*.*

Proof. Being the A_k block diagonal matrices, it is enough to prove the (4.32) for their diagonal blocks, i.e., the Bruun butterfly matrices $B_{m,p}^{(n)}$ (see definition 4.9). In turn the Bruun butterfly matrices can be arranged, via a simultaneous permutation of rows and columns in block diagonal form, with blocks of the form

$$A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \qquad B = \begin{pmatrix} 1 & \alpha & 1 & 0 \\ 0 & 1 & \alpha & 1 \\ 1 & -\alpha & 1 & 0 \\ 0 & 1 & -\alpha & 1 \end{pmatrix}, \quad \alpha = 2\cos\frac{p\pi}{2r}, \qquad C = \begin{pmatrix} 1 & -\bar{\omega}_n^p \\ 1 & -\omega_n^p \end{pmatrix}.$$

Now, for example, it is easy to prove that

$$|fl(B\mathbf{z}) - B\mathbf{z}| \dot{\leq} u\, 3\, |B||\mathbf{z}|$$

where $\mathbf{z}$ a 4-dimensional vector and the modulus applies entrywise. The preceding inequality can be weakened as follows

$$\|fl(B\mathbf{z}) - B\mathbf{z}\|_2 \dot{\leq} u\, 3\, \|B\|_F \|\mathbf{z}\|_2$$

being $\|\cdot\|_F$ the Frobenius norm. Since $\|B\|_F < \sqrt{24}$ we get the desired result. The other cases can be handled analogously. ∎

At this point we are able to prove the main result of this section. Our proof will follow the lines of the proof given by Gentleman an Sande in their paper [40].

Theorem 4.24 *In the hypothesis and with the notations introduced in theorem 4.22, let* $\tilde{\mathbf{y}}$ *be the vector computed from* $\mathbf{x}$ *using DIT Bruun's algorithm. We have*

$$\frac{\|\tilde{\mathbf{y}} - \mathbf{y}\|_2}{\|\mathbf{y}\|_2} \dot{\leq} u\, 5\, n^{1.2} \log n. \tag{4.33}$$

Proof. We have

$$\begin{aligned}
&\|fl(F_n\mathbf{x}) - F_n\mathbf{x}\|_2 = \|fl(A_1 fl(A_2 \cdots fl(A_d\mathbf{x}))) - A_1A_2\cdots A_d\mathbf{x}\|_2 \leq \\
&\quad \leq \sum_{k=1}^{d} \|A_1\cdots A_{k-1} fl(A_k\cdots fl(A_d\mathbf{x})) - A_1\cdots A_k fl(A_{k+1}\cdots fl(A_d\mathbf{x}))\|_2 \leq \\
&\quad \leq \sum_{k=1}^{d} \|A_1\|_2\cdots\|A_{k-1}\|_2 \|fl(A_k\mathbf{z}_k) - A_k\mathbf{z}_k\|_2 \stackrel{\cdot}{\leq} \\
&\quad \stackrel{\cdot}{\leq} u\,15 \sum_{k=1}^{d} \|A_1\|_2\cdots\|A_{k-1}\|_2\|\mathbf{z}_k\|_2,
\end{aligned}$$

where $\mathbf{z}_k = fl(A_{k+1}\cdots fl(A_d\mathbf{x}))$. Now we have

$$u\,\|\mathbf{z}_k\|_2 \stackrel{\cdot}{\leq} u\,\|A_{k+1}\|_2\cdots\|A_d\|_2\|\mathbf{x}\|_2.$$

It can be easily proved that $\|A_k\|_2 \leq \beta = \sqrt{6+\sqrt{20}}$ for $k = 1:d$. Thus we find

$$\|fl(F_n\mathbf{x}) - F_n\mathbf{x}\|_2 \stackrel{\cdot}{\leq} u\,15\beta^{d-1}\|\mathbf{x}\|_2 \log n.$$

Since $\frac{1}{\sqrt{n}}F_n$ is unitary we have that $\|\mathbf{x}\|_2 = \frac{1}{\sqrt{n}}\|F_n\mathbf{x}\|_2$. Thus

$$15\,\beta^{d-1}\|\mathbf{x}\|_2 = \frac{15}{\beta}\left(\frac{\beta}{\sqrt{2}}\right)^d \|F_n\mathbf{x}\|_2 = \frac{15}{\beta} n^{\log\frac{\beta}{\sqrt{2}}}\|F_n\mathbf{x}\|_2.$$

From the last equality we get the thesis. ∎

By virtue of definition 4.21 we have that Bruun's algorithm is numerically strongly stable.

Bibliography

[1] C. M. Ablow, J. L. Brenner, *Roots and canonical forms for circulant matrices*, Trans. Amer. Math. Soc. 107, 1963, pp. 360–376.

[2] N. Ahmed, T. Natarajan, K. Rao, *Discrete Cosine transform*, IEEE Trans. on Computers C-23, 1974, pp. 90–93.

[3] A. V. Aho, J. E. Hopcroft, J. D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading (Mass.), 1975.

[4] G. Ammar, P. Gader, *A variant of the Gohberg-Semencul formula involving circulant matrices*, SIAM J. Matrix Anal. Appl. 12, 1991, pp. 534–540.

[5] G. Ammar, P. Gader, *New decompositions of the inverse of a Toeplitz matrix*, in Signal Processing, Scattering and Operator Theory and Numerical Methods, Proc. Int. Symp. MTNS-89, Vol. 3, Birkäuser, Boston, 1990, pp. 421–428.

[6] G. Ammar, W. Gragg, *Superfast solution of real positive definite Toeplitz systems*, SIAM J. Matrix Anal. Appl. 9, 1988, pp. 61–76.

[7] R. B. Bapat, V. S. Sunder, *On hypergroups of matrices*, Lin. Multilin. Alg. 29, 1991, pp. 125–140.

[8] J. Barría, P. Halmos, *Vector bases for two commuting matrices*, Lin. Multilin. Alg. 27, 1990, pp. 147–157.

[9] R. Bevilacqua, N. Bonanni, E. Bozzo, *On algebras of Toeplitz plus Hankel matrices*, submitted to Linear Algebra Appl.

[10] R. Bevilacqua, P. Zellini, *Closure, commutativity and minimal complexity of some spaces of matrices*, Lin. Multilin. Alg. 25, 1989, pp. 1–25.

[11] D. Bini, *Relations between exact and approximate bilinear algorithms, applications*, Calcolo 17, 1980, pp. 87–97.

[12] D. Bini, *On a class of matrices related to Toeplitz matrices*, Technical Report B83-5, SUNY, Albany, 1983.

[13] D. Bini, E. Bozzo, *Fast discrete transforms by means of eigenpolynomials*, Computer Math. Applic. 26(9), 1993, pp. 35–52

[14] D. Bini, M. Capovani, *Spectral and computational properties of band symmetric Toeplitz matrices*, Linear Algebra Appl. 52/53, 1983, pp. 99–126.

[15] D. Bini, M. Capovani, *Tensor rank and border rank of band Toeplitz matrices*, SIAM J. Comput. 16, 1987, pp. 252–258.

[16] D. Bini, M. Capovani, G. Lotti, F. Romani, *Complessità Numerica*, Boringhieri, Torino, Italy, 1981.

[17] D. Bini, F. Di Benedetto, *An observation on certain spectral properties of Toeplitz matrices*, Calcolo 28, 1991, pp. 37–43.

[18] D. Bini, F. Di Benedetto, *A new preconditioner for the parallel solution of positive definite Toeplitz systems*, Proc. 2nd ACM Symp. on Parallel Algorithms and Architectures, ACM Press, 1990, pp. 220-223.

[19] D. Bini, P. Favati, *On a matrix algebra related to discrete Hartley transform*, SIAM J. Matrix Analysis Appl. 14, 1993, pp. 500–507.

[20] D. Bini, V. Pan, *Numerical and Algebraic Computations with Matrices and Polynomials*, Birkäuser, Boston (MA), to appear.

[21] D. Bini, V. Pan, *Improved parallel computations with Toeplitz-like and Hankel-like matrices*, Linear Algebra Appl. 188/189, 1993, pp. 3–29.

[22] N. Bonanni, *Propietà Computazionali di Algebre di Matrici*, Tesi di Laurea, Università di Pisa, 1993.

[23] E. Bozzo, *Algebras of higher dimension for displacement decompositions and computations with Toeplitz plus Hankel matrices*, to appear on Linear Algebra Appl.

[24] E. Bozzo, C. Di Fiore, *On the use of certain matrix algebras associated with discrete trigonometric transforms in matrix displacement decomposition*, to appear on SIAM J. Matrix Anal. Appl.

[25] W. L. Briggs, *Further symmetries of in-place FFTs*, SIAM J. Sci. Stat. Comput. 8, 1987, pp. 644–654.

[26] E. O. Brigham, *The Fast Fourier Transform*, Prentice-Hall, Englewood Cliffs (New Jersey), 1974.

[27] G. Bruun, *z-Transform DFT Filters and FFT's*, IEEE Trans. Acoust. Speech Signal Process. ASSP-26, 1978, pp. 56–63.

[28] C. S. Burrus, T. W. Parks, *DFT/FFT and Convolution Algorithms*, Wiley, New York, 1982.

[29] P. J. Davis, *Circulant Matrices*, Wiley, New York, 1979.

[30] P. Delsarte, Y. Genin, *Spectral properties of finite Toeplitz matrices*, Proc. of MTNS Int. Symposium, Beer-Sheva (Israel), 1983.

[31] F. Di Benedetto, *Analysis of preconditioning techniques for ill-conditioned Toeplitz matrices*, to appear on SIAM J. Sci. Stat. Comp.

[32] F. Di Benedetto, *Preconditioning of block Toeplitz matrices by Sine transform*, Technical Report 235, Dipartimento di Matematica, Università di Genova, 1993.

[33] C. Di Fiore, P. Zellini, *Matrix decompositions using displacement rank and classes of commutative matrix algebras*, submitted to Linear Algebra Appl.

[34] P. Duhamel, *Implementation of "Split-Radix" FFT Algorithm for Complex, Real, and Real-Symmetric Data*, IEEE Trans. Acoust. Speech Signal Process. ASSP-34, 1986, pp. 285–295.

[35] P. Duhamel, H. Hollmann, *"Split-Radix" FFT algorithm*, Electron. Lett. 20, 1984, pp. 14–16.

[36] P. Duhamel, M. Vetterli, *Improved Fourier and Hartley algorithms: application to cyclic convolution of real data*, IEEE Trans. Acoust. Speech Signal Process. ASSP-35, 1987, pp. 818–824.

[37] B. Friedlander, M. Morf, T. Kailath, L. Ljung, *New inversion formulas for matrices classified in terms of their distance from Toeplitz matrices*, Linear Algebra Appl. 27, 1979, pp. 31–60.

[38] P. Gader, *Displacement operator based decompositions of matrices using circulants or other group matrices*, Linear Algebra Appl. 139, 1990, pp. 111–131.

[39] F. Gantmacher, *The theory of matrices*, Chelsea, New York, 1959.

[40] W. Gentleman, G. Sande, *Fast Fourier transforms-For fun and profit*, Proc. 1966 Fall Joint Computer Conference AFIPS 29, pp. 563–578.

[41] I. Gohberg, V. Olshevsky, *Circulants, displacements and decompositions of matrices*, Integral Equations and Operator Theory 15, 1992.

[42] I. Gohberg, V. Olshevsky, *Fast algorithms with preprocessing for multiplication of transpose Vandermonde matrix and Cauchy matrix with vectors*, preprint.

[43] I. Gohberg, V. Olshevsky, *Complexity of multiplication with vectors for structured matrices*, preprint.

[44] G. Heinig, K. Rost, *Algebraic Methods for Toeplitz-like Matrices and Operators*, Birkhäuser, Boston (MA), 1984.

[45] R. Horn, C. Johnson, *Matrix Analysis*, Cambridge Univ. Press, New York, 1988.

[46] R. Horn, C. Johnson, *Topics in Matrix Analysis*, Cambridge Univ. Press, New York, 1988.

[47] H. Hou, *A fast recursive algorithm for computing the discrete Cosine transform*, IEEE Trans. Acoust. Speech Signal Process. ASSP-35, 1987, pp. 1455–1461.

[48] H. Hou, *The fast Hartley transform*, IEEE Trans. on Computers C-36, 1987, pp. 147–156.

[49] T. Huckle, *Circulant and skewcirculant matrices for solving Toeplitz matrix problems*, SIAM J. Matrix Anal. Appl. 13, 1992, pp. 767–777.

[50] T. Kailath, S. Kung, M. Morf, *Displacement rank of matrices and linear equations*, J. Math. Anal. Appl., 68 (1979), pp. 395–407.

[51] T. Laffey, *A structure theorem for some matrix algebras*, Linear Algebra Appl. 162/164, 1992, pp. 205–215.

[52] J. Martens, *Recursive cyclotomic factorization. A new algorithm for calculating the discrete Fourier transform*, IEEE Trans. Acoust. Speech Signal Process. ASSP-32, 1984, pp. 750–761.

[53] H. J. Nussbaumer, *Fast Fourier Transform and Convolution Algorithms*, Springer-Verlag, Berlin, 1981.

[54] W. H. Press, B. P. Flannery, S. A. Teukolsky, W. T. Vetterling, *Numerical Recipes. The Art of Scientific Computing*, Cambridge Univ. Press, 1986.

[55] R. D. Preuss, *Very fast computation of the radix-2 discrete Fourier transform*, IEEE Trans. Acoust. Speech Signal Process. ASSP-30, 1982, pp. 595–607.

[56] C. M. Rader, N. M. Brenner, *A new principle for the fast Fourier transform*, IEEE Trans. Acoust. Speech Signal Process. ASSP-24, 1976, pp. 264–266.

[57] D. J. Rose, *Matrix identities for the fast Fourier transform*, Linear Algebra Appl. 29, 1980, pp. 423–443.

[58] H. Sorensen, M. Heideman, C. Burrus, *On computing the split-radix FFT*, IEEE Trans. Acoust. Speech Signal Process. ASSP-34, 1986, pp. 152–156.

[59] H. Sorensen, D. Jones, M. Heideman, C. Burrus, *Real-valued fast Fourier transform algorithms*, IEEE Trans. Acoust. Speech Signal Process. ASSP-35, 1987, pp. 849–863.

[60] H. Sorensen, D. Jones, C. Burrus, M. Heideman, *On computing the discrete Hartley transform*, IEEE Trans. Acoust. Speech Signal Process. ASSP-33, 1985, pp. 1231–1238.

[61] R. Storn, *Some results in fixed point error analysis of the Bruun-FFT algorithm*, IEEE Trans. Acoust. Speech Signal Process. ASSP-41, 1993, pp. 2371–2375.

[62] G. Strang, *Introduction to Applied Mathematics*, Wellesley-Cambridge Press, Wellesley (MA), 1987.

[63] D. Suprunenko, R Tyshkevich, *Commutative matrices*, Academic Press, New York, 1968.

[64] P. N. Swarztrauber, *Vectorizing the FFTs*, in *Parallel Computations*, G. Rodrigue (ed.), Academic Press, New York, 1982.

[65] P. N. Swarztrauber, *FFT algorithms for vector computers*, Parallel Computing 1, 1984, pp. 45–63.

[66] C. Temperton, *Self-sorting in-place fast Fourier Transforms*, SIAM J. Sci. Stat. Comput. 12, 1991, pp. 808–823.

[67] B. Van der Vaerden, *Algebra*, Frederick Ungar Publishing, New York, 1970.

[68] C. Van Loan, *Computational Frameworks for the Fast Fourier Transform*, SIAM, Philadelphia, 1992.

[69] M. Vetterli, H.J. Nussbaumer, *Simple FFT and DCT algorithms with reduced number of operations*, Signal Processing 6, 1984, pp. 267–278.

[70] Z. Wang, *Fast algorithms for the discrete W transforms and the discrete Fourier transform*, IEEE Trans. Acoust. Speech Signal Process. ASSP-32, 1984, pp. 803–816.

[71] W. C. Waterhouse, *Circulant-style matrices closed under multiplication*, Lin. Multilin. Alg. 18, 1985, pp. 197–206.

[72] Y. Wu, *New FFT structures based on the Bruun Algorithm*, IEEE Trans. Acoust. Speech Signal Process. ASSP-38, 1990, pp. 188–191.

[73] P. Zellini, *On the optimal computation of a set of symmetric and persymmetric bilinear forms*, Linear Algebra Appl. 23, 1979, pp. 101–119.