Conference Applications of Mathematics 2015, in honor of the birthday anniversaries of Ivo Babuška (90), Milan Práger (85), and Emil Vitásek (85)
J. Brandts, S. Korotov, M. Křížek, K. Segeth, J. Šístek, T. Vejchodský (Eds.)
Institute of Mathematics CAS, Prague 2015

FACTORIZATION MAKES FAST WALSH, PONS AND OTHER HADAMARD-LIKE TRANSFORMS EASY

Jaroslav Kautsky

Flinders University
CSEM, GPO Box 2100, Adelaide 5001, Australia
and
Prometheus, Inc., Newport, RI USA
jardakau@internode.on.net

Abstract: A simple device, based on the factorization of invertible matrix polynomials, enabling to identify the possibility of fast implementation of linear transforms is presented. Its applicability is demonstrated in the case of Hadamard matrices and their generalization, Hadamard matrix polynomials.

Keywords: Hadamard matrices, matrix polynomials, fast implementation, in-place implementation

MSC: 11C08, 15A23, 15B34, 65Y20

1. Introduction

Performing a linear transformation $\boldsymbol{y} = A\boldsymbol{x}$ with an $n \times n$ matrix A requires n^2 operations. We talk about a *fast* implementation of such a transformation if we can lower the number of operations, such as by using the Fast Fourier Transform (FFT) [3], which caused a revolution in signal processing by bringing the cost down to $n \log n$.

Hand-in-hand with the operation cost go memory requirements. In a straightforward implementation we need additional n memory locations. It may be desirable to perform the transformation *in place*, that is, the output $\boldsymbol{y}$ is stored directly into the locations occupied by the input $\boldsymbol{x}$, requiring possibly some small number, independent of n, of memory locations. Fast implementations usually allow such memory savings.

Not long after the FFT technique for reducing the cost similar results appeared for Walsh-Hadamard transforms [5, 12]; this development has continued to the present [13] and now includes Hadamard transforms other than those based on Walsh matrices [1]. Surprisingly, all of these results appear to be based on considerations following those in the development of the FFT.

In this paper we offer a different way to derive fast and in-place algorithms, not only for Hadamard matrices but also for their generalization, *Hadamard matrix polynomials* introduced in Section 3. Our approach is based on the factorization of invertible matrix polynomials discussed in Section 2 and allows not only the derivation of theoretical results but also, being based on a simple deterministic algorithm, the capacity to determine by computational means if a fast implementation exists in particular cases. That is presented in Section 4 and Section 5 for fast and in-place implementations, respectively. We conclude in Section 6 by factorizing a degree three 8×8 Hadamard matrix polynomial into five sparse factors.

The concept, properties and the construction of Hadamard matrix polynomials have been developed by the author and Radka Tezaur/Turcajova. They are documented in unpublished reports for Prometheus, Inc. (some related work can be found in `http://www.prometheus-us.com/PONS-papers/`) and have been used in signal processing applications such as [11]. A special case (size 2×2) has been presented in a Departmental seminar for which an abstract is available [10]. They are introduced here only for the purpose of demonstrating the applicability of the idea of finding fast implementations by factorization of matrix polynomials.

2. IMP — invertible matrix polynomials and their factorization

A matrix polynomial of size m and of order p (or degree $p-1$), $m > 1$, $p > 0$, is given by

$$A(z) = \sum_{k=1}^{p} A_k z^{k-1}$$

where A_k are real $m \times m$ matrices. The product of matrix polynomials is again a matrix polynomial which, unlike scalar polynomials ($m = 1$), may have degree less than the sum of the exact degrees of the multiplicands. This leads to the concepts of invertible matrix polynomials (IMPs) and, as a special case, orthogonal (or unitary) matrix polynomials.

We state these concepts formally as follows:

Definition 2.1 *1. A matrix polynomial $A(z)$ of order p is called invertible if there exists a polynomial $B(z)$ of order q such that $B(0) \neq 0$ and*

$$A(z)B(z) = \beta z^s I$$

(I is the identity matrix) for some scalar $\beta > 0$ and some s, $0 \le s \le p+q-2$.

2. An invertible matrix polynomial $A(z)$ of order p is called orthogonal if

$$B(z) = \sum_{k=1}^{p} A^T_{p-k+1} z^{k-1} = z^{p-1} A^T\left(\frac{1}{z}\right) , \tag{1}$$

$\beta = \|\boldsymbol{e}_1^T A(1)\|_2^2$ ($\boldsymbol{e}_1$ is the first column of the identity matrix) and $s = p - 1$.

We have introduced the parameter β into the concept of inverse for later convenience and also include the parameter s for larger applicability in some situations.

Our results depend on the factorization of matrix polynomials. An orthogonal matrix polynomial of degree p can always be factorized into a product of exactly p linear factors [7, 9], a strong result a weaker form of which has not yet been established for IMPs. It has been shown in [6, p. 112], though, that matrix polynomials can not be generally factorized as demonstrated by the following example:

Example 2.2

$$A(z) = \begin{pmatrix} 0 & -1 \\ 0 & 0 \end{pmatrix} + z^2 \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

is an IMP of size 2 *and degree* 2 *and its inverse is*

$$B(z) = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} + z^2 \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

with $s = 4$ *and* $\beta = 1$.

What has actually been proved in [6] is that this IMP can not be expressed as a product of *two* IMPs, *both* of degree less than two (it is, in fact, factorizable into a product of three IMPs of degree one). The phrase "can not be factorized" here has a very restricted meaning — the product of factors of lower degrees and not more of them than the current degree. The word "factorization" in this paper (and in its title) is taken to mean simply "expressing as a product of factors".

Due to their invertibility IMPs can be factorized in many ways; we can actually prescribe all the partial products in the factorization.

Theorem 2.3 *Let* $B_j(z)$, $j = 0, 1, \ldots, n$, *be invertible matrix polynomials of the same size. Then there exist IMPs* $F_j(z)$, $j = 0, 1, \ldots, n$, *such that*

$$B_j(z) = F_j(z) F_{j-1}(z) \ldots F_0(z) \ , \qquad j = 0, 1, \ldots, n \ .$$

Proof. Take $F_0(z) = B_0(z)$ and $F_j(z) = B_j(z) B_{j-1}^{-1}(z)$, $j = 1, 2, \ldots, n$. □

The degrees of the factors depend, of course, on the chosen partial products. Regardless of how trivial this observation is, it is in fact a basis for what follows because some factorings may have certain desirable properties that others do not, such as particularly sparseness. We can demonstrate an application of this idea—using factorization into *sparse* factors to reduce complexity—in the case of Walsh--Hadamard matrices, which are given by the recurrence

$$W_0 = 1 \ , \qquad W_{k+1} = \begin{pmatrix} W_k & W_k \\ W_k & -W_k \end{pmatrix} \ , \qquad k = 0, 1, \ldots \quad .$$

The steps in this construction are the source of partial products; the only difficulty in applying Theorem 2.3 is the "same size" requirement because W_{k+1} is a square matrix of size 2^{k+1}, double that of W_k. That is overcome by doubling the size of W_k; if we choose

$$B_0 = \begin{pmatrix} W_k & 0 \\ 0 & W_k \end{pmatrix} \quad \text{and} \quad B_1 = W_{k+1}$$

we get

$$B_1 = FB_0 \quad \text{where} \quad F = \begin{pmatrix} W_k & W_k \\ W_k & -W_k \end{pmatrix} \begin{pmatrix} W_k^{-1} & 0 \\ 0 & W_k^{-1} \end{pmatrix} = \begin{pmatrix} I & I \\ I & -I \end{pmatrix}$$

leading immediately to the fast application of W_n costing $N \log_2 N$ operations, $N = 2^n$. A suitable same permutation of both rows and columns changes this factor F into a block diagonal matrix with 2×2 blocks W_1 in the diagonal, leading to an in-place implementation needing only one temporary memory location. Alternatively, this structure can be exploited for parallel processing.

We conclude this section by a comment on the apparently obvious statement that if $A(z) = F(z)B(z)$ then the application of $A(z)$ is equivalent to the application of $B(z)$ followed by that of $F(z)$. While this is straightforward for order $p = 1$ (multiplication of matrices and vectors), for $p > 1$ a rigorous definition of the meaning of “application” involves a lengthy exposition. The difference between applying matrix polynomials of order $p = 1$ and orders $p > 1$ is similar to moving from discrete Fourier transform to discrete wavelet transforms (originally called *lapped* transforms because of the need to use input data from the adjacent blocks). One approach is to express this application using block Toeplitz matrices (made circulant for invertibility) as in [8].

However, for the purpose of this paper it is sufficient to state that by application of an $N \times N$ matrix polynomial $A(z)$ of order p to a vector $\boldsymbol{x} = (\, \boldsymbol{x}_1^T \quad \boldsymbol{x}_2^T \quad \ldots \quad \boldsymbol{x}_{K+p}^T \,)^T$ with $N \times 1$ components $\boldsymbol{x}_j$ we mean the evaluation of output

$$\boldsymbol{y}_j = \sum_{k=1}^{p} A_k \boldsymbol{x}_{j+k-1} \,, \quad j = 1, 2, \ldots, K \,, \tag{2}$$

from which the statement about application of factorized matrix polynomials is easily justified.

It is important to note that when $\boldsymbol{y}_j$, for some j, has been evaluated in (2) then the vector $\boldsymbol{x}_j$ will not be needed to evaluate $\boldsymbol{y}_k$ for $k > j$.

3. HMP — Hadamard matrix polynomials and some constructions

Hadamard matrices are, up to a scalar multiple, orthogonal matrices the elements of which are restricted to have values 1 and -1. We extend this notion to matrix polynomials as follows:

Definition 3.1 *An orthogonal matrix polynomial* $A(z) = \sum_{j=1}^{p} A_j z^{j-1}$ *with elements* 1 *and* -1 *in all matrices* A_j *will be called a Hadamard matrix polynomial (HMP).*

We have introduced the parameter β into the definition of inverse so that we can say that the inverse of an HMP is again an HMP. For an HMP of size m and order p, $\beta = mp$.

Example 3.2

$$A(z) = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} + z \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

is an HMP of size 2 *and order* 2 *and its inverse is*

$$B(z) = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} + z \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

with $s = 1$ *and* $\beta = 4$.

In principle, construction of HMPs can be done by exhaustive search. However, this approach quickly becomes infeasible as the dimension of the problem grows.

As for Walsh-Hadamard matrices, there are constructions which allow doubling either the size or the order of an existing HMP. Starting with the simplest HMP (size 2, order 1)

$$H_0(z) = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

we can thus construct HMPs for which both size and order are powers of 2.

In this paper we will consider only three types of constructions: two which double the size of an HMP and one which doubles the order. Let us define them formally.

Denote by E the per-identity, that is the matrix such that XE reverses the order of columns of matrix X while EX does the same for rows. Also, let D_S be a diagonal matrix with $(\,1 \quad -1 \quad 1 \quad -1 \quad \ldots\,)$ in the diagonal, the matrix post-multiplication by which will change the sign of every second column.

Definition 3.3 *Let* $A(z) = \sum_{k=1}^{p} A_k z^{k-1}$ *be an HMP of even size.*

1. *We define the Walsh-Sylvester size extension* $S_W(A, z)$ *by*

$$S_W(A, z) = \begin{pmatrix} A(z) & A(z) \\ A(z) & -A(z) \end{pmatrix} .$$

2. *We define the PONS-like size extension* $S_P(A, z)$ *by*

$$S_P(A, z) = \begin{pmatrix} A(z) & D_S E A(z) \\ D_S E A(z) & A(z) \end{pmatrix} .$$

3. Block the matrices A_k into equal parts

$$A_k = \begin{pmatrix} U_k^T \\ V_k^T \end{pmatrix} .$$

We define the order extension $O_p(A, z)$ by

$$O_p(A, z) = \sum_{k=1}^{p} \begin{pmatrix} U_k^T \\ U_k^T \end{pmatrix} z^{2k-2} + \begin{pmatrix} V_k^T \\ -V_k^T \end{pmatrix} z^{2k-1} .$$

Proposition 3.4 *The extended matrix polynomials are again HMPs.*

Proof. Straightforward using the second characterization of the orthogonal matrix polynomial in (1). □

Note that the HMP in Example 3.2 is obtained by $O_p(H_0, z)$.

There are other constructions which double either the size or the order of an HMP and also some which preserve both size and order: these include transposing the coefficient matrices, changing sign or permuting rows and columns or reversing the polynomials. It is thus possible to create a large variety of HMPs of increasing size and order. The type S_P is one of similar constructions resulting in so called PONS Hadamard matrices with many properties which are desirable in signal processing [2].

4. HMP — Fast implementation

The constructions of HMPs introduced in the previous section are the basis for applying Theorem 2.3, as was already demonstrated for Walsh-Hadamard matrices in Section 2 (order $p = 1$). We now show the extension to a degree one HMP (order $p = 2$).

Proposition 4.1 *Let W_n be a Walsh-Hadamard matrix of size $N = 2^n$. Then*

$$O_p(W_n, z) = F(z) W_n \quad \textit{where} \quad F(z) = \begin{pmatrix} I & 0 \\ I & 0 \end{pmatrix} + z \begin{pmatrix} 0 & I \\ 0 & -I \end{pmatrix} ,$$

here the identity matrices (as well as the zero blocks) are of size $N/2$. The factor's first coefficient $F_1 = \begin{pmatrix} I & 0 \\ I & 0 \end{pmatrix}$ is permutable to a block diagonal matrix with blocks $\begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix}$ of size 2.

Proof. Denote, for brevity, $W = W_{n-1}$. Calculate

$$\left(\begin{pmatrix} I & 0 \\ I & 0 \end{pmatrix} + z \begin{pmatrix} 0 & I \\ 0 & -I \end{pmatrix} \right) \begin{pmatrix} W & W \\ W & -W \end{pmatrix} = \begin{pmatrix} W & W \\ W & W \end{pmatrix} + z \begin{pmatrix} W & -W \\ -W & W \end{pmatrix} = O_p(W_n, z) .$$

The permutation $(\, 1 \quad N/2+1 \quad 2 \quad N/2+2 \quad \ldots \quad N/2 \quad N \,)$ applied to both rows and columns of F_1 achieves the required diagonalization. □

Further doubling of the order leads to factors of the same structure but of correspondingly higher degrees, with only the first and last coefficient matrices non-zero. The computational complexity is two per item of the output (i. e., two nonzero elements in each row of the linear transformation) in each step.

A more complicated factor is obtained when we reverse the order of extensions (results of this kind are best obtained by computer software and can be checked by calculations similar to those proving the Proposition 4.1).

Proposition 4.2 *If we apply the Walsh extension S_W to the linear HMP $O_p(W_n, z))$ (using the notation of Proposition 4.1) the resulting factor is*

$$F(z) = \begin{pmatrix} I & I & 2I & 0 \\ I & I & 0 & 2I \\ I & I & 0 & -2I \\ I & I & -2I & 0 \end{pmatrix} + z \begin{pmatrix} I & -I & 0 & 0 \\ -I & I & 0 & 0 \\ -I & I & 0 & 0 \\ I & -I & 0 & 0 \end{pmatrix} .$$

The minimal diagonal blocks after permutations have size 4.

The complexity is now much worse—five operations per item (six, actually, if we count only additions and implement the multiplier 2 as two additions). Interestingly, further extensions of size have similarly shaped factors while doubling the order leads to factors with complexity two per item as in Proposition 4.1.

Similar observations can be pursued for HMPs based on the S_P size extensions. These lead to transforms complementary to Walsh-Hadamard transforms with important applications in signal processing [1].

5. HMP — in-place implementation

It is obvious that if we apply a linear transform with an upper triangular matrix, then when we have calculated the first k elements of the result the first k elements of the input will not be needed any more and can be replaced by the output. That observation is the basis of the "in-place" implementation. Similarly, in the application of a matrix polynomial, in (2), we note that, once $\boldsymbol{y}_j$ is evaluated, $\boldsymbol{x}_j$ will not be needed to evaluate $\boldsymbol{y}_k$ for $k > j$.

So memory requirements depend upon how we can implement the evaluation of $A_1\boldsymbol{x}_j$, that is $F_1\boldsymbol{x}_j$, as we are now discussing the application of a factor. As already pointed out in particular cases, we need to be able to permute rows and columns of F_1 into a block upper triangular form; we can then proceed as suggested above and the maximal size of the diagonal blocks gives the number of additional memory locations needed to calculate the transform in place.

It is important to realize that the rows and columns must be permuted in the same way, otherwise we would be storing the results corresponding to the diagonal blocks in the wrong places and overwriting what is still needed in using the next block.

The problem of finding the block triangular form by a symmetric permutation is equivalent to that of determining the strongly connected components of a graph. It can be solved, for example, by Tarjan's algorithm [4].

6. An example

An 8×8 HMP of order 4 (that is, degree 3) constructed by two size extensions S_P followed by two order doubling O_p is visualized as:

```
.......... .......... .......... ..........
.+++---+-. .--+-+++-. .+++---+-. .++-+---+.
.+----+--. .-+--+---. .+----+--. .+-++-+++.
.+-+++---. .+---+-++. .+-+++---. .-+++-+--.
.--+-----+. .---+--+-. .--+-----+. .+++-++-+.
.+++---+-. .--+-+++-. .---+++-+. .--+-+++-.
.+----+--. .-+--+---. .-++++-++. .-+--+---.
.+-+++---. .+---+-++. .-+---+++. .+---+-++.
.--+-----+. .---+--+-. .++-++++-. .---+--+-.
.......... .......... .......... ..........
```

where + and - denote 1 and -1, respectively, and we have surrounded the four coefficient matrices $A_1, \ldots, A_4$ by dots. Note that unlike in the Walsh extensions the pattern of ±1 looks almost random in this PONS-like HMP. Nevertheless, the fast/in-place implementation is still achievable by the following five factors

```
..........     ..........     ..........     .......... ..........
.++      .     .+  +    .     .+      +.     .+       . .    +   .
.+-      .     . +-     .     . +    - .     . +      . .     +  .
.  ++    .     . ++     .     .  +  +  .     .  +     . .      + .
.  +-    .     .-  +    .     .   +-   .     .   +    . .       +.
.    ++  . ,   .    +  +. ,   .   ++   . ,   .+       . .    -   .
.    +-  .     .     +- .     .  -  +  .     . +      . .     -  .
.      ++.     .     ++ .     . +    + .     .  +     . .      - .
.      +-.     .    -  +.     .-      +.     .   +    . .       -.
..........     ..........     ..........     .......... ..........
```

and

```
.......... .......... ..........
.+       . .        . .    +   .
. +      . .        . .     +  .
.  +     . .        . .      + .
.   +    . .        . .       +.
.+       . .        . .    −   .
. +      . .        . .     −  .
.  +     . .        . .      − .
.   +    . .        . .       −.



.......... .......... ..........
```

Notice that application of each row of the original HMP involving 31 additions and subtractions is replaced by just 5 such operations. This was achieved by a factorization of a degree 3 matrix polynomial into the product of five matrix polynomials of degrees 0, 0, 0, 1 and 2, respectively.

One other observation important for implementation is the regular pattern in the structure of the factors which can be realized directly by the computer program and thus avoiding the need to store the original HMP or its factors.

7. Conclusion

In this paper we show that there is a large class of Hadamard matrices and Hadamard matrix polynomials which are constructed in such a way that using the new result of Theorem 2.3 can easily determine whether or not transforms can be implemented fast and in-place. Our approach suggests a simple algorithm which determines the polynomial matrix factors from which it is possible to see, in each case, how good the fast and in-place implementation will be. Theorem 2.3 resolves an important need because the savings available from fast and in-place transforms differ significantly from one case to another, as shown by two examples in Section 3. Indeed, the question what savings can be achieved by fast implementation is thus far from trivial. The possible savings result from two properties: the sparseness of the factors and the small integer sizes of their non-zero elements. An alternative factorization for orthogonal matrix polynomials mentioned in Section 2 ([9]) is not applicable here because the factors would neither be sparse nor have integer values.

Acknowledgements

The author thanks the anonymous referee for useful comments improving the presentation of the material.

This article is dedicated to my theses advisers and colleagues at the occasions of their birthdays. I have worked with them in the years 1956-1968 so it is not surprising that the topic has probably drifted far away from their interests. Nevertheless, I am grateful for their guidance and influence.

References

[1] Byrnes, J., Gertner, I., Ostheimer, G., and Ramalho, M.: Discrete one dimensional signal processing apparatus and method using energy spreading coding. U.S. patent number 5,913,186 (1999).

[2] Byrnes, J.S., Saffari, B., and Shapiro, H.: Energy spreading and data compression using the Prometheus orthonormal set. Proceedings of the IEEE Digital Signal Processing Workshop, Norway (1996), 9–12.

[3] Cooley, J. and Tukey, J.: An algorithm for the machine calculation of complex Fourier series. Math. Comp. **19** (1965), 297–301.

[4] Duff, I. and Reid, J.: An implementation of Tarjan's algorithm for the block triangularization of a matrix. ACM Transactions on Mathematical Software **4** (1978), 137–147.

[5] Fino, B. and Algazi, V.: Unified matrix treatment of the fast Walsh-Hadamard transform. IEEE Trans. on Computers **C-25** (1976), 1142–1146.

[6] Gohberg, L., Lancaster, P., and Rodman, L.: *Matrix polynomials.* Academic Press, New York, 1982.

[7] Kautsky, J. and Turcajová, R.: A matrix approach to discrete wavelets. In: C.K. Chui, L. Montefusco, and L. Puccio (Eds.), *Wavelets: Theory, Algorithms, and Applications*, *Wavelet Analysis and Its Applications*, vol. 5, pp. 117–136. Academic Press, 1994.

[8] Kautsky, J. and Turcajová, R.: Discrete biorthogonal wavelet transforms as block circulant matrices. Linear Algebra Appl. **223/224** (1995), 393–413.

[9] Kautsky, J. and Turcajová, R.: Pollen product factorization and construction of higher multiplicity wavelets. Linear Algebra Appl. **222** (1995), 241–260.

[10] Turcajova, R.: Construction of Hadamard Matrix Polynomials. URL: `http://www2.cs.cas.cz/semincm/abstracts/2004-06-17.Turcajova.html` (2004).

[11] Ultrafast efficient data compression. URL: `http://www.prometheus-us.com/Projects/UltrafastEfficientDataCompression.html` (1999).

[12] Vlasenko, V. and Rao, K.: Unified matrix treatment of discrete transforms. IEEE Trans. on Computers **C-28** (1979), 934–938.

[13] Yusuf, Z., Abbasi, S., and Alamoud, A.: A novel complete set of Walsh and inverse Walsh transforms for signal processing. *International Conference on Communication Systems and Network Technologies (CSNT)*, 2011, 504–509.