

## 6.1 FFT implementations

We noted in Section ii3.3 that the discrete Fourier transform (DFT), given by equations (ii.3-3) and (ii.3-4) could be written in matrix form as (ii.3-5) and (ii.3-6) resp. We will soon see that it suffices to have a code for one of the cases, e.g.

$$\begin{bmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \cdots & \omega^{N-1} \\ 1 & \omega^2 & \omega^4 & \omega^6 & \cdots & \omega^{2N-1} \\ \vdots & \vdots & & & & \\ \vdots & \vdots & & & & \\ 1 & \omega^{N-1} & \cdots & \cdots & \cdots & \omega^{(N-1)^2} \end{bmatrix} \begin{bmatrix} \hat{u}_0 \\ \hat{u}_1 \\ \hat{u}_2 \\ \vdots \\ \hat{u}_{N-1} \end{bmatrix} = \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ \vdots \\ u_{N-1} \end{bmatrix}$$

where  $\omega = e^{2\pi i/N}$ . Figure 1 shows this matrix in the case of  $N = 8$ , and also illustrates a few simple steps, which splits the matrix into a product of three matrices - in order from left to right:

- a very sparse matrix (three diagonals only),
- a matrix which contains two DFTs of half the size (i.e. the process of splitting can be continued), and
- a permutation matrix.

Repeating this splitting idea on the two  $4 \times 4$  blocks, and then on the resulting four  $2 \times 2$  blocks will generate (to the left in the three steps) the first three matrices seen in the top section of Figure 2. The three permutation matrices that emerged to the right in each of the factorizations can be combined into a single permutation matrix. This leads to the Cooley-Tukey factorization. The generalization to a matrix size of  $N = 2^m$  is immediate -  $m$  sparse factors emerge, preceding the permutation matrix. We can note:

- i. The permutation matrix can be described explicitly through *bit reversal*. The following pattern for the matrix in the  $8 \times 8$ -case generalizes in the obvious way:

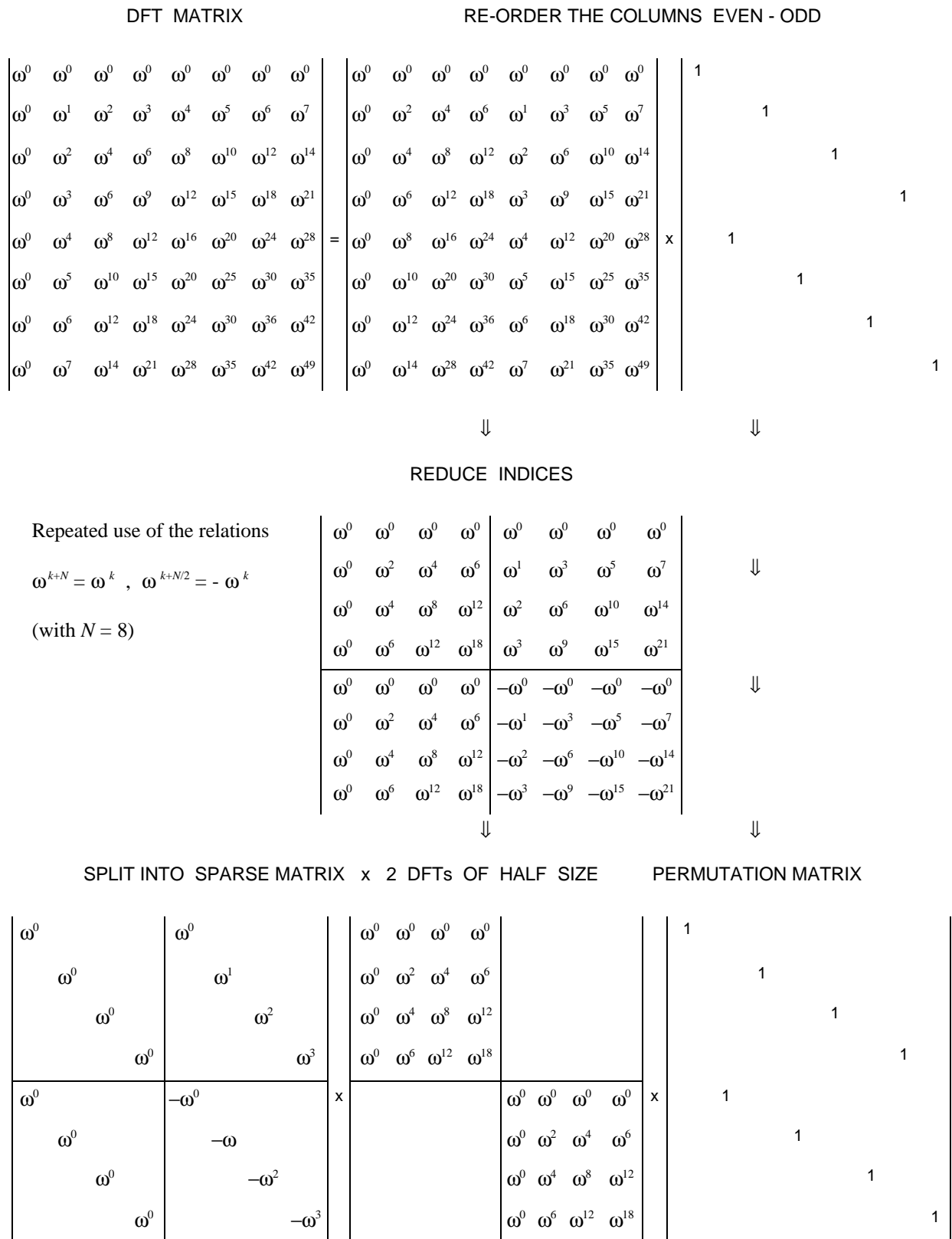


Figure 1. Illustration of DFT matrix factorization in the case of  $N = 8$ .

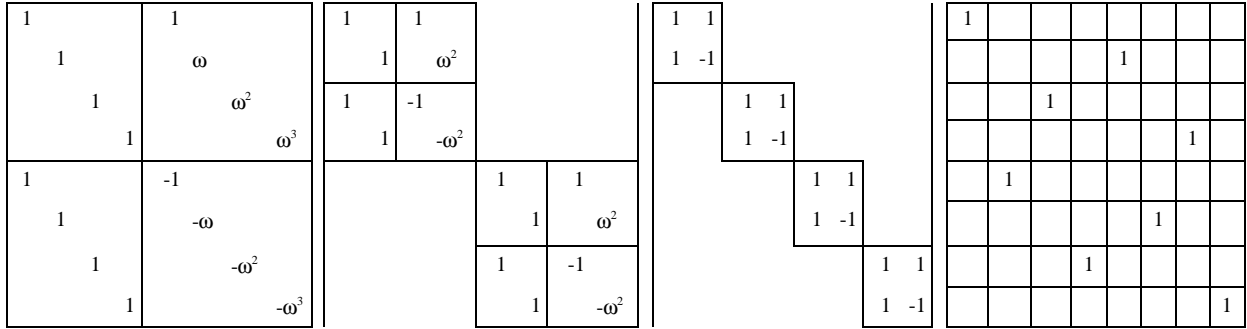


Figure 2 a. Cooley-Tukey factorization.

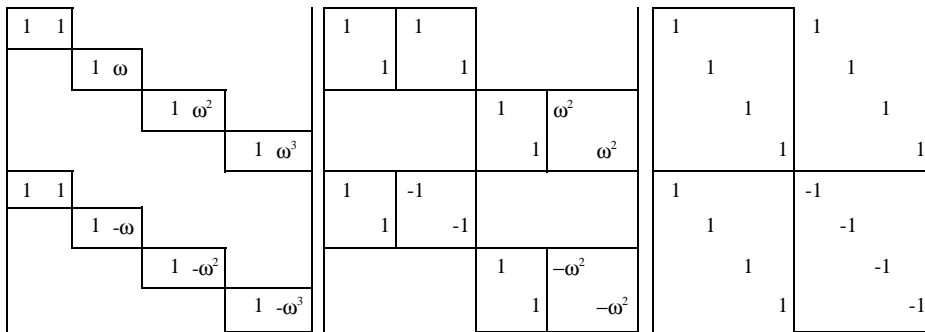


Figure 2 b. Glassman factorization.

Figure 2. Cooley-Tukey and Glassman factorizations of the DFT matrix in case of  $N = 8$ .

*Illustration of bit reversal in case of  $N = 8$*

column	position of entry		expressed in binary	binary digits reversed		becomes equal to
0	0	=	000	000	=	0
1	4	=	100	001	=	1
2	2	=	010	010	=	2
3	6	=	110	011	=	3
4	1	=	001	100	=	4
5	5	=	101	101	=	5
6	3	=	011	110	=	6
7	7	=	111	111	=	7

From the structure of the permutation matrix in Figure 1, the bit reversal principle can be shown for ex. by induction over the order of the DFT matrix size (here assumed to be powers of two). In FFT codes, the permutations can be implemented in several ways:

- by explicit computation of the bit reversal for each case,
  - by noting that the bit reversal in effect only amounts to a few pairwise exchanges of elements, or
  - by storing the 'position of entry' vector for the largest case  $N$  that is of interest. For transforms over  $N/2$  points, we pick every second element from this pointer vector, for  $N/4$  points every fourth etc.
- ii. Every matrix  $\times$  vector multiplication amounts to a series of replacements of pairs of vector elements - these can all be performed with overwriting, i.e. like for the permutations, no extra storage vector is needed.

An alternative to the Cooley-Tukey algorithm was proposed by Glassman (bottom part of Figure 2); the permutations are now incorporated into the factors, changing their structure somewhat. In this case, an additional storage vector is needed for intermediate results during the matrix  $\times$  vector multiplications. This factorization has proven to be particularly effective on certain high-performance vector computers.