

24.12 The Fast Fourier Transform

Chapter 12 §09 built one unitary matrix, $\mathbf{F}$, that diagonalizes every circulant of its size, and then counted the cost of using it. Forming $\mathbf{F}\mathbf{b}$ as a matrix–vector product takes about n^2 multiplications, so the elegant three-step recipe “transform, divide, transform back” for a circulant system costs $O(n^2)$ — better than elimination, but not dramatically so. The section closed by naming the way out: the powers of ω repeat, splitting one transform of length n into two of length $n/2$ costs almost nothing, and iterating “brings the cost down to about $n \log n$ ”. That algorithm is the **fast Fourier transform**, and Chapter 12 handed it here, “together with the cost accounting that makes ‘quickly’ precise”.

This section pays that debt. There is no new theorem about $\mathbf{F}$ here; there is one matrix identity, proved by computing an entry, and one recurrence, solved exactly.

Throughout, as in Chapter 12 §09, **rows, columns, and vector entries are numbered** $0, 1, \dots, n-1$, and $\omega_n := e^{2\pi i/n}$. We write $\mathbf{F}_n$ for the Fourier matrix of size n of Theorem 12.9.3, so $(\mathbf{F}_n)_{jk} = \omega_n^{jk}/\sqrt{n}$; Chapter 12 wrote it $\mathbf{F}$, the size being fixed there.

24.12.1 Counting the cost of a transform

Chapter 12 §09 defined the **discrete Fourier transform** of $\mathbf{c} \in \mathbb{C}^n$ to be $\hat{\mathbf{c}} = \sqrt{n} \mathbf{F}_n \mathbf{c}$, the vector of eigenvalues $p_{\mathbf{c}}(\omega_n^j)$ of the circulant with first row $\mathbf{c}$. It is the unnormalized transform that the algorithm computes, so it is convenient to give its matrix a name of its own.

Definition 24.12.1. The transform matrix

For $n \geq 1$ let

$$\mathbf{W}_n := \sqrt{n} \mathbf{F}_n \in M_n(\mathbb{C}), \quad (\mathbf{W}_n)_{jk} = \omega_n^{jk} \quad (0 \leq j, k \leq n-1),$$

so that $\hat{\mathbf{c}} = \mathbf{W}_n \mathbf{c}$ for every $\mathbf{c} \in \mathbb{C}^n$.

A **cost** in this section is a count of arithmetic operations, where one complex addition, one complex subtraction, one complex multiplication, and one complex division each count as **one operation**; a multiplication by the constant 1 is free, because it is not performed. What is being counted is the work of applying $\mathbf{W}_n$ to an arbitrary vector, not the work of writing $\mathbf{W}_n$ down.

Applying $\mathbf{W}_n$ as a plain matrix–vector product costs

$$n^2 \text{ multiplications} + n(n-1) \text{ additions} = 2n^2 - n$$

operations: each of the n entries of $\mathbf{W}_n \mathbf{c}$ is a sum of n products. This figure charges every one of those products, including the $2n-1$ whose factor is an entry $\omega_n^0 = 1$ of row 0 or column 0; it is the plain matrix–vector product as it is ordinarily performed, and the comparisons below use it as it stands. For $n = 1024$ that is 2 096 128 operations. The whole of this section is about replacing that number by about 15 000.

24.12.2 Splitting a transform in two

Where can a saving possibly come from? The entries of $\mathbf{W}_n$ are powers of ω_n , and there are only n distinct ones, so the n^2 products $\omega_n^{jk} c_k$ involve enormous repetition. The single algebraic

fact that organizes the repetition is

$$\omega_n^2 = e^{4\pi i/n} = e^{2\pi i/m} = \omega_m, \quad \text{where } n = 2m.$$

Squaring a primitive n -th root of unity gives a primitive m -th root of unity. So if we separate the even-numbered entries of $\mathbf{c}$ from the odd-numbered ones, the even half is transformed by $\mathbf{W}_m$, not by anything new. Two objects give the bookkeeping a name.

Definition 24.12.2. The even-odd permutation, the twiddle matrix, and the butterfly

Let $n = 2m$ with $m \geq 1$.

- (a) The **even-odd permutation matrix** $\mathbf{P}_n \in M_n(\mathbb{C})$ is the permutation matrix determined by

$$(\mathbf{P}_n \mathbf{x})_t = x_{2t}, \quad (\mathbf{P}_n \mathbf{x})_{m+t} = x_{2t+1} \quad (0 \leq t \leq m-1);$$

it lists the even-numbered entries of $\mathbf{x}$ first, then the odd-numbered ones, each in their original order.

- (b) The **twiddle matrix** is $\mathbf{D}_m := \text{diag}(1, \omega_n, \omega_n^2, \dots, \omega_n^{m-1}) \in M_m(\mathbb{C})$. Its entries are powers of ω_n , **not** of ω_m .

- (c) The **butterfly matrix** is

$$\mathbf{B}_n := \begin{pmatrix} \mathbf{I}_m & \mathbf{D}_m \\ \mathbf{I}_m & -\mathbf{D}_m \end{pmatrix} \in M_n(\mathbb{C}).$$

In words: $\mathbf{P}_n$ is pure data movement and does no arithmetic; $\mathbf{D}_m$ supplies the m twiddles that the odd half needs because it starts one place late; and $\mathbf{B}_n$ forms the m sums and the m differences that recombine the two halves. Applied to a vector $(\mathbf{u}, \mathbf{v})$ split into two halves of length m , the butterfly returns $(\mathbf{u} + \mathbf{D}_m \mathbf{v}, \mathbf{u} - \mathbf{D}_m \mathbf{v})$.

Theorem 24.12.1. The Radix-Two Factorization

Let $n = 2m$ with $m \geq 1$. Then

$$\mathbf{W}_n = \mathbf{B}_n \begin{pmatrix} \mathbf{W}_m & \mathbf{O} \\ \mathbf{O} & \mathbf{W}_m \end{pmatrix} \mathbf{P}_n.$$

Equivalently, for the unitary Fourier matrices,

$$\mathbf{F}_n = \frac{1}{\sqrt{2}} \mathbf{B}_n \begin{pmatrix} \mathbf{F}_m & \mathbf{O} \\ \mathbf{O} & \mathbf{F}_m \end{pmatrix} \mathbf{P}_n.$$

Idea. Apply both sides to an arbitrary $\mathbf{x}$ and compute entry j of the left side. Split the defining sum $\sum_k \omega_n^{jk} x_k$ into its even $k = 2t$ and odd $k = 2t + 1$ parts; the relation $\omega_n^2 = \omega_m$ turns each part into a transform of length m , and the odd part carries one spare factor ω_n^j . That is the whole computation, and it already gives the top half of the answer. For the bottom half, write $j = m + i$; the relation $\omega_n^m = 1$ makes the two transforms of length m repeat what they gave at i , while $\omega_n^m = -1$ flips the sign of the spare factor. Sum and difference: the butterfly.

Proof. Let $\mathbf{x} \in \mathbb{C}^n$ and put

$$\mathbf{u} := \mathbf{W}_m(x_0, x_2, \dots, x_{n-2}), \quad \mathbf{v} := \mathbf{W}_m(x_1, x_3, \dots, x_{n-1}),$$

so that $u_i = \sum_{t=0}^{m-1} \omega_m^{it} x_{2t}$ and $v_i = \sum_{t=0}^{m-1} \omega_m^{it} x_{2t+1}$ for $0 \leq i \leq m-1$, by [Definition 24.12.1](#). By [Definition 24.12.2](#) (a), $\mathbf{P}_n \mathbf{x}$ is exactly the pair of these two argument vectors stacked, so the right-hand side applied to $\mathbf{x}$ is $\mathbf{B}_n(\mathbf{u}, \mathbf{v}) = (\mathbf{u} + \mathbf{D}_m \mathbf{v}, \mathbf{u} - \mathbf{D}_m \mathbf{v})$.

Now compute entry j of $\mathbf{W}_n \mathbf{x}$. Splitting the index k into $k = 2t$ and $k = 2t + 1$ with $0 \leq t \leq m-1$, and using $\omega_n^2 = \omega_m$,

$$\begin{aligned} (\mathbf{W}_n \mathbf{x})_j &= \sum_{k=0}^{n-1} \omega_n^{jk} x_k = \sum_{t=0}^{m-1} \omega_n^{2jt} x_{2t} + \sum_{t=0}^{m-1} \omega_n^{j(2t+1)} x_{2t+1} \\ &= \sum_{t=0}^{m-1} \omega_m^{jt} x_{2t} + \omega_n^j \sum_{t=0}^{m-1} \omega_m^{jt} x_{2t+1}. \end{aligned}$$

Top half. For $0 \leq j \leq m-1$ the two sums are u_j and v_j , so

$$(\mathbf{W}_n \mathbf{x})_j = u_j + \omega_n^j v_j = (\mathbf{u} + \mathbf{D}_m \mathbf{v})_j,$$

the last equality by [Definition 24.12.2](#) (b), the j -th diagonal entry of $\mathbf{D}_m$ being ω_n^j .

Bottom half. For $j = m + i$ with $0 \leq i \leq m-1$, we have $\omega_n^{jt} = \omega_m^{mt} \omega_m^{it} = \omega_m^{it}$, since $\omega_m^m = 1$; so both sums repeat their values at i , namely u_i and v_i . The spare factor is

$$\omega_n^{m+i} = \omega_n^m \omega_n^i = e^{\pi i} \omega_n^i = -\omega_n^i,$$

because $\omega_n^m = e^{2\pi i m/n} = e^{\pi i} = -1$. Hence

$$(\mathbf{W}_n \mathbf{x})_{m+i} = u_i - \omega_n^i v_i = (\mathbf{u} - \mathbf{D}_m \mathbf{v})_i.$$

The two halves together say that $\mathbf{W}_n \mathbf{x}$ equals the right-hand side applied to $\mathbf{x}$, for every $\mathbf{x} \in \mathbb{C}^n$; so the two matrices are equal. For the second display, substitute $\mathbf{W}_n = \sqrt{n} \mathbf{F}_n$ and $\mathbf{W}_m = \sqrt{m} \mathbf{F}_m$ and divide by $\sqrt{n}$, using $\sqrt{m}/\sqrt{n} = 1/\sqrt{2}$. ■

The identity is a factorization of one matrix into three, and each of the three is cheap for a different reason: $\mathbf{P}_n$ does no arithmetic, the middle factor is two copies of a **smaller instance of the same problem**, and $\mathbf{B}_n$ costs $O(n)$. That is the entire content of the fast Fourier transform. Everything below is bookkeeping.

Quick check. Write out [Theorem 24.12.1](#) for $n = 2$ and check it directly.

Solution. Here $m = 1$, so $\mathbf{P}_2 = \mathbf{I}_2$ (it lists x_0 then x_1), $\mathbf{W}_1 = (1)$, and $\mathbf{D}_1 = (\omega_2^0) = (1)$. Hence

$$\mathbf{B}_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix},$$

and the right-hand side is $\mathbf{B}_2 \mathbf{I}_2 \mathbf{I}_2 = \mathbf{B}_2$. And $(\mathbf{W}_2)_{jk} = \omega_2^{jk} = (-1)^{jk}$, which is the same matrix.

Example. The factorization for $n = 4$

Write out [Theorem 24.12.1](#) for $n = 4$, and check it against the matrix $\mathbf{F}$ of Example (The Fourier matrix of size four).

Solution. Here $m = 2$ and $\omega_4 = i$, so $\mathbf{D}_2 = \text{diag}(1, i)$ and

$$\mathbf{B}_4 = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & i \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -i \end{pmatrix}, \quad \mathbf{P}_4 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad \mathbf{W}_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

Multiplying, $\begin{pmatrix} \mathbf{W}_2 & \mathbf{O} \\ \mathbf{O} & \mathbf{W}_2 \end{pmatrix} \mathbf{P}_4$ has rows $(1, 0, 1, 0)$, $(1, 0, -1, 0)$, $(0, 1, 0, 1)$, $(0, 1, 0, -1)$. Applying $\mathbf{B}_4$ adds row 3 to row 1, adds i times row 4 to row 2, and so on, giving

$$\mathbf{B}_4 \begin{pmatrix} \mathbf{W}_2 & \mathbf{O} \\ \mathbf{O} & \mathbf{W}_2 \end{pmatrix} \mathbf{P}_4 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{pmatrix} = \mathbf{W}_4,$$

whose (j, k) -entry is i^{jk} . Dividing by $\sqrt{4} = 2$ returns the matrix $\mathbf{F} = \frac{1}{2} \mathbf{W}_4$ written out in Example (The Fourier matrix of size four).

Warning. The twiddles are powers of ω_n , not of ω_m . If $\mathbf{D}_2$ in [Example \(The factorization for \$n = 4\$ \)](#) is taken to be $\text{diag}(1, \omega_2) = \text{diag}(1, -1)$, the product becomes the **real** matrix with rows $(1, 1, 1, 1)$, $(1, -1, -1, 1)$, $(1, -1, 1, -1)$, $(1, 1, -1, -1)$, whereas the second row of $\mathbf{W}_4$ is $(1, i, -1, -i)$. The single factor ω_n^j in the proof is what remembers that the odd-numbered entries start one step later, and it is the only place where the transform of size n differs from two transforms of size m .

24.12.3 The cost, exactly

Read [Theorem 24.12.1](#) from right to left as a procedure.

Algorithm. Input. $\mathbf{x} \in \mathbb{C}^n$, with $n = 2^p$ a power of two.

Output. $\mathbf{W}_n \mathbf{x}$.

1. If $n = 1$, return $\mathbf{x}$.
2. Split $\mathbf{x}$ into its even-numbered and odd-numbered entries (no arithmetic).
3. Compute $\mathbf{u}$ and $\mathbf{v}$, the transforms of the two halves, by applying this procedure to each of them.

4. Multiply: set $v'_0 := v_0$, and $v'_i := \omega_n^i v_i$ for $1 \leq i \leq m-1$, where $m = n/2$.
5. Return $(\mathbf{u} + \mathbf{v}', \mathbf{u} - \mathbf{v}')$.

The procedure returns $\mathbf{W}_n \mathbf{x}$ for every $n = 2^p$, by induction on p : for $p = 0$ step 1 returns $\mathbf{x} = \mathbf{W}_1 \mathbf{x}$, and for $n = 2m$ the inductive hypothesis makes the $\mathbf{u}$ and $\mathbf{v}$ of step 3 the two transforms of length m named in the proof of [Theorem 24.12.1](#), so steps 4 and 5 return $(\mathbf{u} + \mathbf{D}_m \mathbf{v}, \mathbf{u} - \mathbf{D}_m \mathbf{v})$, which that proof identifies with $\mathbf{W}_n \mathbf{x}$. What is left is the cost.

Theorem 24.12.2. The Cost of the Fast Fourier Transform

Let $C(n)$ be the number of operations the procedure above performs on an input of length $n = 2^p$, with $p \geq 0$. Then $C(1) = 0$,

$$C(2m) = 2C(m) + 3m - 1 \quad (m \geq 1),$$

and the solution of this recurrence is exactly

$$C(2^p) = \frac{3}{2} p 2^p - 2^p + 1, \quad \text{that is} \quad C(n) = \frac{3}{2} n \log_2 n - n + 1.$$

In particular $C(n) = O(n \log n)$.

Idea. Step 3 is where the recursion sits and contributes $2C(m)$; steps 4 and 5 are the only arithmetic outside it, and both are linear in n . Count them once, carefully: step 4 is $m-1$ multiplications, the factor $\omega_n^0 = 1$ at $i = 0$ not being applied at all, and step 5 is m additions and m subtractions. Then solve the recurrence by induction on p , which works because the recursion halves the length exactly.

Proof. For $n = 1$ the procedure returns its input and does nothing, so $C(1) = 0$.

Let $n = 2m$ with $m \geq 1$. Step 2 only moves data. Step 3 calls the procedure twice on length m , costing $2C(m)$. Step 4 copies v_0 , which is no arithmetic, and forms $\omega_n^i v_i$ for $i = 1, \dots, m-1$, which is $m-1$ multiplications. Step 5 forms m sums and m differences, which is $2m$ operations. Adding,

$$C(2m) = 2C(m) + (m-1) + 2m = 2C(m) + 3m - 1.$$

Now induct on p . For $p = 0$, the formula gives $\frac{3}{2} \cdot 0 \cdot 1 - 1 + 1 = 0 = C(1)$. Let $p \geq 1$ and suppose $C(2^{p-1}) = \frac{3}{2}(p-1)2^{p-1} - 2^{p-1} + 1$. Taking $m = 2^{p-1}$ in the recurrence,

$$\begin{aligned} C(2^p) &= 2 \left(\frac{3}{2}(p-1)2^{p-1} - 2^{p-1} + 1 \right) + 3 \cdot 2^{p-1} - 1 \\ &= \frac{3}{2}(p-1)2^p - 2^p + 2 + \frac{3}{2} \cdot 2^p - 1 \\ &= \frac{3}{2}p2^p - 2^p + 1, \end{aligned}$$

where the last line collects $\frac{3}{2}(p-1)2^p + \frac{3}{2}2^p = \frac{3}{2}p2^p$. This completes the induction. Substituting $n = 2^p$, so $p = \log_2 n$, gives the stated formula, and $\frac{3}{2}n \log_2 n - n + 1 \leq \frac{3}{2}n \log_2 n$ for $n \geq 1$, so $C(n) = O(n \log n)$. ■

The exact formula is worth tabulating against the $2n^2 - n$ of a plain matrix–vector product.

n	$C(n)$	$2n^2 - n$
2	2	6
8	29	120
32	209	2016
1024	14 337	2 096 128

At $n = 1024$ the ratio is about 146, and it grows without bound, because $(2n^2 - n)/C(n)$ behaves like $\frac{4}{3}n/\log_2 n$. This is the sense in which the transform can be applied “quickly”: the exponent drops from 2 to 1, up to a logarithm, and the constant $\frac{3}{2}$ in front is small.

Warning. The recursion needs n even at every level, so $n = 2^p$ is the clean case. For n odd and greater than 1 the factorization of [Theorem 24.12.1](#) does not apply at all, and the procedure above cannot start. For $n = 2^a q$ with q odd, it descends a levels and then must transform vectors of length q some other way; if q is large, nothing has been gained. The remedy is either to split by other factors of n , as in [Exercise \(C1: Splitting into three\)](#), or, for the convolution use below, to pad with zeros up to the next power of two — which at most doubles n and so changes the cost by a bounded factor.

Example. A transform of length eight

Compute $\mathbf{W}_8 \mathbf{c}$ for $\mathbf{c} = (1, 2, 3, 4, 1, 2, 3, 4)$ by the procedure, showing every level, and check the result against the direct definition at one entry.

Solution. Level 1. The even-numbered entries are $(1, 3, 1, 3)$ and the odd-numbered ones are $(2, 4, 2, 4)$.

Level 2, applied to $(1, 3, 1, 3)$. Its even and odd halves are $(1, 1)$ and $(3, 3)$, whose transforms are $\mathbf{W}_2(1, 1) = (2, 0)$ and $\mathbf{W}_2(3, 3) = (6, 0)$. Here $n = 4$, so $\mathbf{D}_2 = \text{diag}(1, i)$ and $\mathbf{D}_2(6, 0) = (6, 0)$. The butterfly gives $(2 + 6, 0 + 0, 2 - 6, 0 - 0) = (8, 0, -4, 0)$. So $\mathbf{u} = (8, 0, -4, 0)$.

Level 2, applied to $(2, 4, 2, 4)$. Its even and odd halves are $(2, 2)$ and $(4, 4)$, whose transforms are $\mathbf{W}_2(2, 2) = (4, 0)$ and $\mathbf{W}_2(4, 4) = (8, 0)$. Again $n = 4$ and $\mathbf{D}_2(8, 0) = (8, 0)$, so the butterfly gives $(4 + 8, 0, 4 - 8, 0)$. So $\mathbf{v} = (12, 0, -4, 0)$.

Level 1, recombination. Now $n = 8$ and $\mathbf{D}_4 = \text{diag}(1, \omega_8, i, \omega_8^3)$ with $\omega_8 = e^{\pi i/4}$, so

$$\mathbf{D}_4 \mathbf{v} = (12, 0, -4i, 0),$$

the irrational entries of $\mathbf{D}_4$ never appearing because $v_1 = v_3 = 0$. The butterfly gives

$$\mathbf{W}_8 \mathbf{c} = (20, 0, -4 - 4i, 0, -4, 0, -4 + 4i, 0).$$

Check at $j = 2$. By [Definition 24.12.1](#), $(\mathbf{W}_8 \mathbf{c})_2 = \sum_k \omega_8^{2k} c_k = \sum_k i^k c_k$ since $\omega_8^2 = i$. This is

$$1 + 2i - 3 - 4i + 1 + 2i - 3 - 4i = -4 - 4i,$$

matching. The zeros in the odd positions are not an accident. Since $\mathbf{c}$ repeats with period

4, that is $c_{k+4} = c_k$, splitting the defining sum at $k = 4$ gives

$$(\mathbf{W}_8 \mathbf{c})_j = \sum_{k=0}^3 c_k (\omega_8^{jk} + \omega_8^{j(k+4)}) = (1 + (-1)^j) \sum_{k=0}^3 \omega_8^{jk} c_k,$$

using $\omega_8^{4j} = (e^{\pi i})^j = (-1)^j$. The factor $1 + (-1)^j$ is 0 at every odd j .

24.12.4 What the speed buys

Two corollaries, both immediate now, and both paying off Chapter 12 §09.

Chapter 12 §09 proved the Convolution Theorem: $\widehat{\mathbf{c} * \mathbf{d}} = \widehat{\mathbf{c}} \odot \widehat{\mathbf{d}}$ entrywise (Theorem 12.9.6 (b)). Before this section that identity converted an $O(n^2)$ convolution into three $O(n^2)$ transforms, which is no gain at all. With Theorem 24.12.2 it becomes a genuine algorithm.

Corollary 24.12.3. Convolution in $O(n \log n)$

Let $n = 2^p$. The cyclic convolution $\mathbf{c} * \mathbf{d}$ of two vectors in $\mathbb{C}^n$ can be computed in

$$3C(n) + 2n = \frac{9}{2} n \log_2 n - n + 3$$

operations, where C is as in Theorem 24.12.2. Consequently, the product of two polynomials in $\mathbb{C}[x]$ of degree at most d can be computed in $O(d \log d)$ operations.

Proof. **The inverse transform costs the same.** By Theorem 12.9.3, $\mathbf{F}_n^{-1} = \mathbf{F}_n^*$, and $\mathbf{F}_n$ is symmetric because $(\mathbf{F}_n)_{jk} = \omega_n^{jk} / \sqrt{n}$ is unchanged by swapping j and k ; hence $\mathbf{F}_n^* = \overline{\mathbf{F}_n}$ and

$$\mathbf{W}_n^{-1} = \frac{1}{\sqrt{n}} \mathbf{F}_n^{-1} = \frac{1}{\sqrt{n}} \overline{\mathbf{F}_n} = \frac{1}{n} \overline{\mathbf{W}_n}.$$

Conjugating the identity of Theorem 24.12.1 entry by entry, and noting that $\mathbf{P}_n$ is real, gives

$$\overline{\mathbf{W}_n} = \overline{\mathbf{B}_n} \begin{pmatrix} \overline{\mathbf{W}_m} & \mathbf{O} \\ \mathbf{O} & \overline{\mathbf{W}_m} \end{pmatrix} \mathbf{P}_n,$$

which is the same factorization with every ω_n replaced by $\overline{\omega_n} = \omega_n^{-1}$. The procedure therefore runs verbatim with ω_n^{-1} in place of ω_n and computes $\overline{\mathbf{W}_n} \mathbf{y}$ in $C(n)$ operations.

The algorithm. By Theorem 12.9.6 (b) and the definition $\widehat{\mathbf{c}} = \mathbf{W}_n \mathbf{c}$,

$$\mathbf{W}_n(\mathbf{c} * \mathbf{d}) = (\mathbf{W}_n \mathbf{c}) \odot (\mathbf{W}_n \mathbf{d}), \quad \text{so} \quad \mathbf{c} * \mathbf{d} = \frac{1}{n} \overline{\mathbf{W}_n} ((\mathbf{W}_n \mathbf{c}) \odot (\mathbf{W}_n \mathbf{d})).$$

Computing $\mathbf{W}_n \mathbf{c}$ and $\mathbf{W}_n \mathbf{d}$ costs $2C(n)$; the n entrywise products cost n ; the inverse transform costs $C(n)$; the n divisions by n cost n . The total is $3C(n) + 2n$, and substituting the formula of Theorem 24.12.2 gives $\frac{9}{2} n \log_2 n - 3n + 3 + 2n$, as stated.

Polynomials. Let $p = \sum_{l=0}^d p_l x^l$ and $q = \sum_{l=0}^d q_l x^l$, and choose $n = 2^p$ with $n \geq 2d + 1$ and $n < 4d + 2$, which is possible since the powers of two are spaced by a factor of two. Let $\mathbf{c}, \mathbf{d} \in \mathbb{C}^n$ be the coefficient vectors padded with zeros. The coefficient of x^r in pq , for $0 \leq r \leq 2d$, is $\sum_l c_l d_{r-l}$ summed over $0 \leq l \leq r$, while $(\mathbf{c} * \mathbf{d})_r = \sum_{l=0}^{n-1} c_l d_{r-l}$ with the subscript read modulo n . The two agree: a term of the second sum with $l > r$ has d -subscript $r - l + n$, and it can be nonzero only if $l \leq d$ and $r - l + n \leq d$, which forces

$n \leq d + l - r \leq 2d$, contradicting $n \geq 2d + 1$. So the coefficients of pq are the first $2d + 1$ entries of $\mathbf{c} * \mathbf{d}$, computable in $3C(n) + 2n$ operations with $n < 4d + 2$, which is $O(d \log d)$. ■

Warning. **Convolution wraps around; polynomial multiplication does not.** Padding to length $n \geq 2d + 1$ is not a convenience, it is the hypothesis. Take $p = q = 1 + x$, so $d = 1$, and convolve the unpadding vectors $\mathbf{c} = \mathbf{d} = (1, 1)$ in $\mathbb{C}^2$: the answer is $(c_0d_0 + c_1d_1, c_0d_1 + c_1d_0) = (2, 2)$, the coefficient vector of $2 + 2x$, whereas $(1 + x)^2 = 1 + 2x + x^2$. The x^2 term has wrapped around onto the constant term, because in $\mathbb{C}^n$ the exponents are read modulo n and $x^2 \equiv 1$. Padding to $\mathbf{c} = \mathbf{d} = (1, 1, 0, 0)$ in $\mathbb{C}^4$ gives $(1, 2, 1, 0)$, which is right.

Corollary 24.12.4. Solving a Circulant System in $O(n \log n)$

Let $n = 2^p$, let $\mathbf{C} \in M_n(\mathbb{C})$ be an invertible circulant with first row $\mathbf{c}$, and let $\mathbf{b} \in \mathbb{C}^n$. Then the solution of $\mathbf{C}\mathbf{x} = \mathbf{b}$ can be computed in $3C(n) + 2n$ operations.

Proof. By Theorem 12.9.4, $\mathbf{C} = \mathbf{F}_n \mathbf{\Lambda} \mathbf{F}_n^*$ with $\mathbf{\Lambda} = \text{diag}(p_{\mathbf{c}}(1), \dots, p_{\mathbf{c}}(\omega_n^{n-1}))$, and the diagonal of $\mathbf{\Lambda}$ is exactly $\hat{\mathbf{c}} = \mathbf{W}_n \mathbf{c}$; the letter $\mathbf{D}$ is reserved here for the twiddle matrix of Definition 24.12.2. Since $\mathbf{C}$ is invertible, every $\hat{c}_j \neq 0$ by Corollary 12.9.5 (d), so $\mathbf{\Lambda}^{-1}$ exists and

$$\mathbf{x} = \mathbf{C}^{-1} \mathbf{b} = \mathbf{F}_n \mathbf{\Lambda}^{-1} \mathbf{F}_n^* \mathbf{b} = \frac{1}{n} \mathbf{W}_n \left((\overline{\mathbf{W}_n \mathbf{b}}) \oslash \hat{\mathbf{c}} \right),$$

where $\oslash$ is entrywise division and the last equality uses $\mathbf{F}_n = n^{-1/2} \mathbf{W}_n$ and $\mathbf{F}_n^* = n^{-1/2} \overline{\mathbf{W}_n}$, as in the proof of Corollary 24.12.3. The three transforms cost $C(n)$ each, by Theorem 24.12.2 and the first part of that proof; the n divisions and the n final scalings by $1/n$ cost $2n$. Sanity check at $\mathbf{C} = \mathbf{I}_n$: then $\mathbf{c} = \mathbf{e}_0$, $\hat{\mathbf{c}} = (1, \dots, 1)$, and the formula returns $\frac{1}{n} \mathbf{W}_n \overline{\mathbf{W}_n} \mathbf{b} = \frac{1}{n} (n \mathbf{I}_n) \mathbf{b} = \mathbf{b}$, using $\mathbf{W}_n \overline{\mathbf{W}_n} = n \mathbf{F}_n \mathbf{F}_n^* = n \mathbf{I}_n$. ■

Remark. None of this is a new theorem about the discrete Fourier transform. The matrix $\mathbf{F}_n$, the factorization $\mathbf{C} = \mathbf{F}_n \mathbf{\Lambda} \mathbf{F}_n^*$ of Theorem 12.9.4, and the Convolution Theorem (Theorem 12.9.6) are exactly as Chapter 12 left them; Theorem 24.12.1 is an identity between matrices all of which were already known, and Theorem 24.12.2 counts operations. The transform is a change of basis, and the fast Fourier transform is a way of applying that change of basis in $O(n \log n)$ operations instead of $O(n^2)$. What changes is not what can be computed but what can be computed at the sizes that occur.

24.12.5 Exercises

24.12.5.1 A. Check your understanding

Exercise. A1

- (a) Write down the matrix $\mathbf{W}_n$, and say how it is related to the Fourier matrix $\mathbf{F}_n$ of Chapter 12.
- (b) State [Theorem 24.12.1](#), and say what each of the three factors on the right costs to apply.
- (c) How many operations does $\mathbf{W}_n \mathbf{x}$ cost as a plain matrix–vector product, and how many by the fast algorithm when $n = 2^p$?
- (d) True or false: the fast Fourier transform computes a different transform from the one in Chapter 12, which is why it is faster. Justify your answer.
- (e) Why does the algorithm require n to be a power of two, and what is done when it is not?

Solution.

- (a) $\mathbf{W}_n = \sqrt{n} \mathbf{F}_n$, with $(\mathbf{W}_n)_{jk} = \omega_n^{jk}$ for $0 \leq j, k \leq n-1$ and $\omega_n = e^{2\pi i/n}$ ([Definition 24.12.1](#)). It is the matrix of the discrete Fourier transform $\mathbf{c} \mapsto \hat{\mathbf{c}}$ of Chapter 12 §09.
- (b) $\mathbf{W}_n = \mathbf{B}_n \begin{pmatrix} \mathbf{W}_m & \mathbf{O} \\ \mathbf{O} & \mathbf{W}_m \end{pmatrix} \mathbf{P}_n$ for $n = 2m$. Applying $\mathbf{P}_n$ costs nothing, since it only moves entries; the middle factor costs two transforms of length m ; and $\mathbf{B}_n$ costs $m-1$ multiplications and $2m$ additions and subtractions.
- (c) Directly: n^2 multiplications and $n(n-1)$ additions, that is $2n^2 - n$. By the algorithm: $C(n) = \frac{3}{2}n \log_2 n - n + 1$ ([Theorem 24.12.2](#)).
- (d) False. [Theorem 24.12.1](#) is an identity between matrices, so the algorithm returns $\mathbf{W}_n \mathbf{x}$ exactly, in exact arithmetic. It is faster because it applies the same matrix by a cheaper route, not because it computes something else.
- (e) The factorization needs $n = 2m$, and the recursion needs that at every level, so it needs $n = 2^p$. For other n one either pads with zeros to the next power of two, which is legitimate for convolution provided the padded length still exceeds the degree of the product (see the warning after [Corollary 24.12.3](#)), or splits by a different factor of n , as in [Exercise \(C1: Splitting into three\)](#).

24.12.5.2 B. Practice

Exercise. B1: A transform of length four

Let $\mathbf{c} = (1, 1, -1, -1) \in \mathbb{C}^4$.

- (a) Compute $\mathbf{W}_4 \mathbf{c}$ by the procedure of this section, showing the two halves and the twiddles.
- (b) Check the answer by evaluating the symbol $p_{\mathbf{c}}$ at the fourth roots of unity.
- (c) Count the operations your computation in (a) used, and compare with $C(4)$.

Solution.

- (a) The even-numbered entries are $(1, -1)$ and the odd-numbered ones are $(1, -1)$. Both transform to $\mathbf{W}_2(1, -1) = (1-1, 1+1) = (0, 2)$, so $\mathbf{u} = \mathbf{v} = (0, 2)$. With $n = 4$,

$\mathbf{D}_2 = \text{diag}(1, i)$, so $\mathbf{D}_2 \mathbf{v} = (0, 2i)$. The butterfly gives

$$\mathbf{W}_4 \mathbf{c} = (0 + 0, 2 + 2i, 0 - 0, 2 - 2i) = (0, 2 + 2i, 0, 2 - 2i).$$

- (b) The symbol is $p_{\mathbf{c}}(x) = 1 + x - x^2 - x^3$ (Definition 12.9.1), and $\hat{c}_j = p_{\mathbf{c}}(i^j)$. Then $p_{\mathbf{c}}(1) = 1 + 1 - 1 - 1 = 0$; $p_{\mathbf{c}}(i) = 1 + i + 1 + i = 2 + 2i$, using $-i^2 = 1$ and $-i^3 = i$; $p_{\mathbf{c}}(-1) = 1 - 1 - 1 + 1 = 0$; and $p_{\mathbf{c}}(-i) = 1 - i + 1 - i = 2 - 2i$. The four values agree with (a).
- (c) Each $\mathbf{W}_2$ costs 2 operations (one sum, one difference), so the two halves cost 4; the twiddles cost 1, since the factor 1 is free and only $i \cdot 2$ is a multiplication; the final butterfly costs 4. The total is 9, and $C(4) = \frac{3}{2} \cdot 4 \cdot 2 - 4 + 1 = 9$. A direct matrix–vector product would cost $2 \cdot 16 - 4 = 28$.

Exercise. B2: A transform of length eight

Let $\mathbf{c} = (2, 1, 0, 3, 2, 1, 0, 3) \in \mathbb{C}^8$. Compute $\mathbf{W}_8 \mathbf{c}$ by the procedure, showing both levels, and verify entry $j = 2$ directly from Definition 24.12.1.

Solution. Level 1. Even-numbered entries $(2, 0, 2, 0)$; odd-numbered entries $(1, 3, 1, 3)$.

Level 2 on $(2, 0, 2, 0)$. Halves $(2, 2)$ and $(0, 0)$, with transforms $(4, 0)$ and $(0, 0)$. Here the inner n is 4, so $\mathbf{D}_2 = \text{diag}(1, i)$ and $\mathbf{D}_2(0, 0) = (0, 0)$. The butterfly gives $\mathbf{u} = (4, 0, 4, 0)$.

Level 2 on $(1, 3, 1, 3)$. Halves $(1, 1)$ and $(3, 3)$, with transforms $(2, 0)$ and $(6, 0)$; $\mathbf{D}_2(6, 0) = (6, 0)$; the butterfly gives $\mathbf{v} = (8, 0, -4, 0)$.

Level 1, recombination. Now $n = 8$ and $\mathbf{D}_4 = \text{diag}(1, \omega_8, i, \omega_8^3)$, so $\mathbf{D}_4 \mathbf{v} = (8, 0, -4i, 0)$. Hence

$$\begin{aligned} \mathbf{W}_8 \mathbf{c} &= (4 + 8, 0, 4 - 4i, 0, 4 - 8, 0, 4 + 4i, 0) \\ &= (12, 0, 4 - 4i, 0, -4, 0, 4 + 4i, 0). \end{aligned}$$

Check at $j = 2$. Since $\omega_8^2 = i$, Definition 24.12.1 gives $(\mathbf{W}_8 \mathbf{c})_2 = \sum_k i^k c_k = 2 + i + 0 - 3i + 2 + i + 0 - 3i = 4 - 4i$, as computed.

Exercise. B3: A product of polynomials by convolution

Let $p = 1 + 2x + 3x^2$ and $q = 2 + x$ in $\mathbb{C}[x]$.

- (a) What is the smallest power of two n for which the recipe in the proof of Corollary 24.12.3 is valid? Write down the padded coefficient vectors $\mathbf{c}$ and $\mathbf{d}$.
- (b) Compute $\mathbf{c} * \mathbf{d}$ directly from the definition of cyclic convolution, and read off pq .
- (c) Show that using $n = 3$ instead would give the wrong answer, and identify which coefficient wraps onto which.

Solution.

- (a) The product pq has degree 3, so 4 coefficients are needed and any $n \geq 4$ works; the smallest power of two is $n = 4$. The padded vectors are $\mathbf{c} = (1, 2, 3, 0)$ and

$\mathbf{d} = (2, 1, 0, 0)$.

- (b) With subscripts modulo 4 and $d_0 = 2, d_1 = 1, d_2 = d_3 = 0$, the sum $(\mathbf{c} * \mathbf{d})_r = \sum_l c_l d_{r-l}$ keeps only $l = r$ and $l = r - 1$, so $(\mathbf{c} * \mathbf{d})_r = 2c_r + c_{r-1}$:

$$\mathbf{c} * \mathbf{d} = (2 + 0, 4 + 1, 6 + 2, 0 + 3) = (2, 5, 8, 3).$$

Hence $pq = 2 + 5x + 8x^2 + 3x^3$, which one checks by expanding: $(1 + 2x + 3x^2)(2 + x) = 2 + x + 4x + 2x^2 + 6x^2 + 3x^3$.

- (c) Length 3 holds p and q but not the four coefficients of pq , and although 3 is not a power of two it shows the wrap plainly: with $\mathbf{c} = (1, 2, 3)$, $\mathbf{d} = (2, 1, 0)$, and subscripts modulo 3,

$$\mathbf{c} * \mathbf{d} = (2 + 3, 4 + 1, 6 + 2) = (5, 5, 8),$$

which is pq with the coefficient 3 of x^3 added onto the constant term, since $x^3 \equiv 1$ modulo $x^3 - 1$. The requirement $n \geq 2d + 1$ in [Corollary 24.12.3](#) is exactly what rules this out.

24.12.5.3 C. Going deeper

Exercise. C1: Splitting into three

Let $n = 3m$ with $m \geq 1$, let $\omega_n = e^{2\pi i/n}$, and for $s = 0, 1, 2$ let

$$\mathbf{u}^{(s)} := \mathbf{W}_m(x_s, x_{3+s}, \dots, x_{3(m-1)+s}) \in \mathbb{C}^m.$$

- (a) Prove that for $0 \leq q \leq 2$ and $0 \leq i \leq m - 1$,

$$(\mathbf{W}_n \mathbf{x})_{qm+i} = \sum_{s=0}^2 \omega_3^{qs} \omega_n^{is} u_i^{(s)},$$

where $\omega_3 = e^{2\pi i/3}$.

- (b) Deduce a recursive procedure for $n = 3^p$, and show that its cost C_3 satisfies $C_3(3m) \leq 3C_3(m) + 12m$ and hence $C_3(3^p) \leq 4p 3^p = 4n \log_3 n$.

Hint for (a): imitate the proof of [Theorem 24.12.1](#), splitting the index k by its remainder modulo 3 and using $\omega_n^3 = \omega_m$ and $\omega_n^m = \omega_3$.

Solution.

- (a) Write $k = 3t + s$ with $0 \leq s \leq 2$ and $0 \leq t \leq m - 1$, which lists each $k \in \{0, \dots, n - 1\}$ exactly once. Then, using $\omega_n^3 = e^{6\pi i/n} = e^{2\pi i/m} = \omega_m$,

$$(\mathbf{W}_n \mathbf{x})_j = \sum_{s=0}^2 \sum_{t=0}^{m-1} \omega_n^{j(3t+s)} x_{3t+s} = \sum_{s=0}^2 \omega_n^{js} \sum_{t=0}^{m-1} \omega_m^{jt} x_{3t+s}.$$

Put $j = qm + i$. Since $\omega_m^m = 1$, we get $\omega_m^{jt} = \omega_m^{it}$, so the inner sum is $u_i^{(s)}$ by [Definition 24.12.1](#). And $\omega_n^m = e^{2\pi im/n} = e^{2\pi i/3} = \omega_3$, so

$$\omega_n^{js} = \omega_n^{(qm+i)s} = (\omega_n^m)^{qs} \omega_n^{is} = \omega_3^{qs} \omega_n^{is}.$$

Substituting gives the stated formula.

- (b) The procedure is: split $\mathbf{x}$ by the remainder of the index modulo 3 (no arithmetic); transform the three parts recursively, getting $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \mathbf{u}^{(2)}$; multiply $u_i^{(s)}$ by ω_n^{is} for $s = 1, 2$ and all i ; and for each i apply $\mathbf{W}_3$ to the resulting triple, placing the three outputs at positions $i, m + i$, and $2m + i$. Correctness is (a).

Cost. The recursive calls cost $3C_3(m)$. The twiddles are $2m$ multiplications, the factors for $s = 0$ being 1. Each three-point transform $(a, b, c) \mapsto (a + b + c, a + \omega_3 b + \omega_3^2 c, a + \omega_3^2 b + \omega_3 c)$ costs at most 4 multiplications and 6 additions, that is 10 operations, and there are m of them. Hence $C_3(3m) \leq 3C_3(m) + 2m + 10m = 3C_3(m) + 12m$.

Now put $D_p = C_3(3^p)/3^p$. The inequality with $m = 3^{p-1}$ reads $3^p D_p \leq 3 \cdot 3^{p-1} D_{p-1} + 12 \cdot 3^{p-1}$, that is $D_p \leq D_{p-1} + 4$. Since $D_0 = C_3(1) = 0$, induction gives $D_p \leq 4p$, so $C_3(3^p) \leq 4p 3^p$. With $n = 3^p$, that is $4n \log_3 n = O(n \log n)$.

Exercise. C2: The fourth power of the transform

Let $n \geq 1$ and let $\mathbf{R}_n \in M_n(\mathbb{C})$ be the permutation matrix with $(\mathbf{R}_n)_{jl} = 1$ when $j + l \equiv 0 \pmod{n}$ and 0 otherwise, so that $(\mathbf{R}_n \mathbf{x})_j = x_{n-j}$ with the subscript read modulo n .

- Prove that $\mathbf{W}_n^2 = n \mathbf{R}_n$.
- Deduce that $\mathbf{W}_n^4 = n^2 \mathbf{I}_n$ and $\mathbf{F}_n^4 = \mathbf{I}_n$, and that every eigenvalue of $\mathbf{F}_n$ lies in $\{1, i, -1, -i\}$.
- Deduce a second proof that $\mathbf{W}_n^{-1} = \frac{1}{n} \overline{\mathbf{W}_n}$.

Solution.

- (a) By [Definition 24.12.1](#) and the definition of the matrix product,

$$(\mathbf{W}_n^2)_{jl} = \sum_{k=0}^{n-1} \omega_n^{jk} \omega_n^{kl} = \sum_{k=0}^{n-1} (\omega_n^{j+l})^k.$$

Put $z = \omega_n^{j+l}$, so $z^n = 1$. If n divides $j + l$, then $z = 1$ and the sum is n . Otherwise $z \neq 1$ by Example (Roots of Unity), and the geometric-series identity $(z - 1)(1 + z + \cdots + z^{n-1}) = z^n - 1 = 0$ used in the proof of Theorem 12.9.3 gives $\sum_k z^k = 0$, since $z - 1 \neq 0$. So $(\mathbf{W}_n^2)_{jl} = n$ exactly when $j + l \equiv 0 \pmod{n}$ and 0 otherwise, which is $n \mathbf{R}_n$.

- The map $j \mapsto -j$ modulo n is its own inverse, so $\mathbf{R}_n^2 = \mathbf{I}_n$ and $\mathbf{W}_n^4 = (n \mathbf{R}_n)^2 = n^2 \mathbf{I}_n$. Since $\mathbf{W}_n = \sqrt{n} \mathbf{F}_n$, we get $n^2 \mathbf{F}_n^4 = n^2 \mathbf{I}_n$, so $\mathbf{F}_n^4 = \mathbf{I}_n$. Hence $x^4 - 1$ annihilates $\mathbf{F}_n$; if $\mathbf{F}_n \mathbf{v} = \lambda \mathbf{v}$ with $\mathbf{v} \neq \mathbf{0}$, then $\mathbf{v} = \mathbf{F}_n^4 \mathbf{v} = \lambda^4 \mathbf{v}$, so $\lambda^4 = 1$ and $\lambda \in \{1, i, -1, -i\}$ by Example (Roots of Unity).
- From (a), $\mathbf{W}_n \cdot \frac{1}{n} \mathbf{W}_n \mathbf{R}_n = \mathbf{R}_n \mathbf{R}_n = \mathbf{I}_n$, so $\mathbf{W}_n^{-1} = \frac{1}{n} \mathbf{W}_n \mathbf{R}_n$. Its (j, l) -entry is $\frac{1}{n}$ times entry $(j, n - l)$ of $\mathbf{W}_n$, namely $\frac{1}{n} \omega_n^{j(n-l)} = \frac{1}{n} \omega_n^{-jl} = \frac{1}{n} \overline{\omega_n^{jl}}$, using $\omega_n^n = 1$ and $|\omega_n| = 1$. That is the (j, l) -entry of $\frac{1}{n} \overline{\mathbf{W}_n}$.