The Discrete Fourier Transform

Complex Fourier Series Representation

Recall that a Fourier series has the form

$$a_0 + \sum_{k=1}^{\infty} a_k \cos(kt) + \sum_{k=1}^{\infty} b_k \sin(kt).$$

This representation seems a bit awkward, since it involves two different infinite series. We remedy this by representing a Fourier series with complex numbers.

To begin, recall the identity

$$e^{i\theta} = \cos\theta + i\sin\theta.$$

Moreover, setting $\theta = kt$, we have

$$e^{ikt} = \cos kt + i\sin kt,$$

for any integer k. Thus e^{ikt} has the ability to include both cosines and sines, and thus seems like a good candidate for representing a Fourier series as a single series. Now consider the series

$$\sum_{k=0}^{\infty} c_k e^{ikt}.$$

If we allow for $c_k = a_k - ib_k$ to be a complex number, then the k th term of the series evaluates to

$$(a_k - ib_k)(\cos kt + i\sin kt) = (a_k \cos kt + b_k \sin kt) + i(a_k \sin kt - b_k \cos kt).$$

Notice how the real part $a_k \cos kt + b_k \sin kt$ provides the coefficients of the original Fourier series. However, there is also the imaginary part $(a_k \sin kt - b_k \cos kt)$ which is problematic, since we still want the output of the series to be real-valued, since in practice the signals are real valued. Moreover, since the sine and cosine functions are collectively linearly independent, the imaginary part will only vanish if all of the coefficients are zero, but this would also cause the real part to vanish! For this reason we must also add the **conjugate terms** of the form $c_{-k}e^{-ikt}$, for $k = 1, 2, \ldots$. This gives the series

$$\sum_{k=-\infty}^{\infty} c_k e^{ikt},$$

which is a single series, but now infinite in both directions.

If $c = a + bi$ is a complex number, then $\overline{c} = a - bi$ is called the **conjugate** of c.

Proposition 1. If $c_{-k} = \overline{c_k}$, then, for all $t \in \mathcal{R}$, the Fourier series

$$\sum_{k=-\infty}^{\infty} c_k e^{ikt}$$

is real valued.

Proof of Proposition 1. We have

$$c_k e^{ik} = (a_k - ib_k)(\cos kt + i \sin kt) = (a_k \cos kt + b_k \sin kt) + i(a_k \sin kt - b_k \cos kt),$$

while

$$c_{-k} e^{-ik} = \overline{c_k} e^{-ik} = (a_k + ib_k)(\cos kt - i \sin kt) = (a_k \cos kt + b_k \sin kt) - i(a_k \sin kt - b_k \cos kt).$$

Finally, adding the two equations yields

$$c_k e^{ik} + c_{-k} e^{-ik} = 2(a_k \cos kt + b_k \sin kt).$$

Notice in the proof of Proposition 1 that

$$\overline{c_k e^{ik}} = \overline{c_k} \overline{e^{ikt}} = \overline{c_k} e^{-ik}.$$

In general, if c and d are complex numbers, then $\overline{cd} = \overline{c}\overline{d}$.

From here onward we call the Fourier series of Proposition 1 the **complex-valued Fourier series**, or complex Fourier series for short. Moreover, an n **th degree complex trigonometric polynomial** is one of the form

$$p(t) = \sum_{k=-n}^{n} c_k e^{ikt}.$$

Roots of Unity

Suppose our goal is to obtain a Fourier approximation of a signal $E(t)$. In practice, we do not have access to a rule or equation that describes $E(t)$, and can only come to know it by sampling it at different values of t. From the previous lecture we may assume, by choosing an appropriate time scale, that $E(t)$ is periodic over the interval $[0, 2\pi]$. Moreover, suppose that n samples are taken, and starting at $t = 0$, a sample is taken once every $2\pi/n$ seconds. Then the sampling times are at $t = \frac{2\pi j}{n}$,

for $j = 0, \ldots, n-1$. These sampling times have an interesting mathematical property. Namely that, for each $j = 0, \ldots, n-1$, $e^{\frac{2\pi ij}{n}}$ is a **complex n th root of unity**, meaning that

$$e^{(\frac{2\pi ij}{n})^n} = e^{2\pi ij} = \cos(2\pi j) + i\sin(2\pi j) = 1.$$

Example 1. Determine the a) complex 4th roots of unity, and b) complex 6th roots of unity.

The next proposition shows that $e^{\frac{2\pi ij}{n}}$, $j = 0, \ldots, n-1$, are the only unique powers of $e^{\frac{2\pi i}{n}}$.

Proposition 2. For integers j and k satisfying $j \equiv k \bmod n$, then $e^{\frac{2\pi ij}{n}} = e^{\frac{2\pi ik}{n}}$.

Proof of Proposition 2. Assume $j \equiv k \bmod n$. Then $k = nq + j$, for some integer q. Then

$$e^{\frac{2\pi ik}{n}} = e^{\frac{2\pi i(j+nq)}{n}} = e^{\frac{2\pi ij}{n}} e^{\frac{2\pi inq}{n}} = e^{\frac{2\pi ij}{n}} e^{2\pi iq} = e^{\frac{2\pi ij}{n}} \cdot 1 = e^{\frac{2\pi ij}{n}}.$$

Proposition 2 allows us to define the Abelian group whose elements are the n th roots of unity, with multiplication as addition. Moreover,

$$e^{\frac{2\pi ij}{n}} \cdot e^{\frac{2\pi ik}{n}} = e^{\frac{2\pi i(j+k)}{n}}.$$

Moreover, the addition is associative since multiplying two roots of unity is identical to integer addition (e.g. adding j with k) which is associative. Also, 1 is the identity element, and the (additive) inverse of $e^{\frac{2\pi ij}{n}}$ is $e^{\frac{2\pi i(n-j)}{n}}$. Another way of writing the inverse of $e^{\frac{2\pi ij}{n}}$ is $e^{\frac{-2\pi ij}{n}}$. This is valid, since $n - i \equiv -i \bmod n$.

For simplicity, we denote the j th power of the n th root of unity $e^{\frac{2\pi ij}{n}}$ by ω_n^j. Moreover, ω_n^{-j} denotes the inverse of ω_n^j. In general, for any integer k, ω_n^k is defined as being equal to ω_n^j $j \equiv k \bmod n$.

Example 2. For the 6th roots of unity, determine the inverse of each group element, and verify that $(a+bi)(a+bi)^{-1} = 1$ through direct multiplication.

Polynomial Interpolation

Our goal in this section is to develop a framework for solving the following problem. Suppose we sample our signal $E(t)$ at n different time values $t_0, \ldots, t_{n-1}$, and obtain the n measurements $y_0 = E(t_0), \ldots, y_{n-1} = E(t_{n-1})$. Problem: find the trigonometric polynomial that best fits these measurements. It turns out that if we sample at times $\frac{2\pi j}{n}$, where $j = 0, \ldots, n-1$, and limit the polynomial approximation to having degree $n-1$, then there is a unique $(n-1)$ th degree trigonometric polynomial that fits the n measurements.

Before getting into the details of solving the above problem, consider the related problem: given n points $(x_0, y_0), \ldots, (x_{n-1}, y_{n-1})$ having distinct x values, determine the coefficients $c_0 \ldots, c_{n-1}$ of an $(n-1)$-degree polynomial

$$p(x) = c_0 + c_1 x + \cdots + c_{n-1} x^{n-1},$$

for which $p(x_i) = y_i$ for all $i = 0, \ldots, n-1$. Finding a degree $n-1$ polynomial that agrees with each of the n data point is called **polynomial interpolation**. We now show that there is a unique $n-1$-degree polynomial that fits the data. Indeed, if the above $p(x)$ is the desired polynomial, then $p(x_i) = y_i$ for all $i = 0, \ldots, n-1$, implies the following system of linear equations.

$$c_0 + c_1 x_0 + c_2 x_0^2 + \cdots + c_{n-1} x_0^{n-1} = y_0$$

$$\vdots$$

$$c_0 + c_1 x_{n-1} + c_2 x_{n-1}^2 + \cdots + c_{n-1} x_{n-1}^{n-1} = y_0,$$

which has the coefficient matrix

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^{n-1} \\ & & \vdots & & \\ 1 & x_{n-1} & x_{n-1}^2 & \cdots & x_{n-1}^{n-1} \end{pmatrix}$$

The above matrix is called the n-dimensional **Vandermonde** matrix, in honor of Alexandre-Thophile Vandermonde. The fact that this matrix is always invertible, and hence yields a unique solution, is a special case of the following theorem.

Theorem 1. Let $p_1(x), \ldots, p_{n-1}(x)$ be a sequence of polynomials for which $p_i(x)$ has degree i, $i = 1, \ldots, n-1$. Then if $x_0, \ldots, x_{n-1}$ are distinct complex numbers, then the matrix

$$\begin{pmatrix} 1 & p_1(x_0) & p_2(x_0) & \cdots & p_{n-1}(x_0) \\ & & \vdots & & \\ 1 & p_1(x_{n-1}) & p_2(x_{n-1}) & \cdots & p_{n-1}(x_{n-1}) \end{pmatrix}$$

is invertible.

To prove Theorem 1, we need the following two lemmas.

Lemma 1. Assuming $x \neq y$, The polynomial $x^j - y^j$ is divisible by $x - y$, for all integers $j \geq 1$.

Proof of Lemma 1. One can verify through multiplication that

$$x^j - y^j = (x - y)(x^{j-1} + x^{j-2}y + x^{j-3}y^2 + \cdots + x^2 y^{j-3} + x y^{j-2} + y^{j-1}).$$

Thereofre, $x - y$ is a factor of $x^j - y^j$; i.e., $x^j - y^j$ is divisible by $x - y$.

Lemma 2. Let $p_1(x)$ be a linear polynomial, and $p_j(x)$ be a degree j polynomial, where $j \geq 1$. Then, for any constant x_0, $(p_j(x) - p_j(x_0))/(p_1(x) - p_1(x_0))$ is a degree $j - 1$ polynomial.

Proof of Lemma 2. Suppose

$$p_j(x) = c_0 + c_1 x + \cdots + c_j x^j,$$

where $c_0, \ldots, c_j$ are complex numbers. Then

$$p_j(x) - p_j(x_0) = (c_0 + c_1 x + \cdots + c_j x^j) - (c_0 + c_1 x_0 + \cdots + c_j x_0^j) =$$

$$c_j(x^j - x_0^j) + \cdots + c_1(x - x_0).$$

Thus, since each term of $p_j(x) - p_j(x_0)$ has a factor of the form $(x^i - x_0^i)$, $i = 1, \ldots, j$, it follows by Lemma 1 that $p_j(x) - p_j(x_0)$ is divisible by $(x - x_0)$. Moreover, since $p_1(x) - p_1(x_0) = a(x - x_0)$, for some nonzero complex number a, it follows that $p_j(x) - p_j(x_0)$ is divisible by $p_1(x) - p_1(x_0)$, and, from the proof of Lemma 1, the quotient is a degree $j - 1$ polynomial.

Proof of Theorem 1. The proof uses induction on n. Assume the rows and columns are numbered starting with index 0.

Basis Step: $n = 2$. Then $n - 1 = 1$, and the matrix is

$$\begin{pmatrix} 1 & p_1(x_0) \\ 1 & p_1(x_1) \end{pmatrix}.$$

Then the elementary row operation $r_1 \leftarrow r_1 - r_0$ yields the new matrix

$$\begin{pmatrix} 1 & p_1(x_0) \\ 0 & p_1(x_1) - p_1(x_0) \end{pmatrix}.$$

Since $p_1(x) = ax + b$, where $a \neq 0$, entry a_{01} of the above matrix is $ax_1 - ax_0 = a(x_1 - x_0) \neq 0$, since $x_1 \neq x_0$. Therefore the subsequent row operations $r_1 \leftarrow \frac{1}{a_{01}} r_1$ and $r_0 \leftarrow r_0 - p_1(x_0) r_1$ produces the identity matrix.

Inductive Step: Assume the theorem is true for all values of n up to some value n_0, where $n_0 \geq 2$. We now show it is also true for the value $n_0 + 1$. Let $n = n_0$. Then it must be shown that the matrix

$$\begin{pmatrix} 1 & p_1(x_0) & p_2(x_0) & \cdots & p_{n-1}(x_0) & p_n(x_0) \\ & & \vdots & & & \\ 1 & p_1(x_{n-1}) & p_2(x_{n-1}) & \cdots & p_{n-1}(x_{n-1}) & p_n(x_{n-1}) \\ 1 & p_1(x_n) & p_2(x_n) & \cdots & p_{n-1}(x_n) & p_n(x_n) \end{pmatrix}$$

is invertible, where $x_0, \ldots, x_{n-1}, x_n$ are distinct complex numbers. Using Gauss-Jordan elimination, for each row r_i, $i = 1, \ldots, n$, perform the operation $r_i \leftarrow r_i - r_0$. This has the effect of placing zeros in rows 1 through n of column 0. The matrix is now

$$\begin{pmatrix} 1 & p_1(x_0) & p_2(x_0) & \cdots & p_{n-1}(x_0) & p_n(x_0) \\ & & \vdots & & & \\ 0 & p_1(x_{n-1}) - p_1(x_0) & p_2(x_{n-1}) - p_2(x_0) & \cdots & p_{n-1}(x_{n-1}) - p_{n-1}(x_0) & p_n(x_{n-1}) - p_n(x_0) \\ 0 & p_1(x_n) - p_1(x_0) & p_2(x_n) - p_2(x_0) & \cdots & p_{n-1}(x_n) - p_{n-1}(x_0) & p_n(x_n) - p_n(x_0) \end{pmatrix}.$$

For the moment, ignore row 0 and column 0, since column 0 already has a leading 1 in row 0. It remains to show that the submatrix with rows 1 through n and columns 1 through n can be row reduced to the identity matrix. To show this, first divide row i, $i = 1, \ldots, n$, by $p_1(x_i) - p_1(x_0)$ to obtain the new submatrix

$$\begin{pmatrix} 1 & \frac{p_2(x_1) - p_2(x_0)}{p_1(x_1) - p_1(x_0)} & \cdots & \frac{p_{n-1}(x_1) - p_{n-1}(x_0)}{p_1(x_1) - p_1(x_0)} & \frac{p_n(x_1) - p_n(x_0)}{p_1(x_1) - p_1(x_0)} \\ & & \vdots & & \\ 1 & \frac{p_2(x_{n-1}) - p_2(x_0)}{p_1(x_{n-1}) - p_1(x_0)} & \cdots & \frac{p_{n-1}(x_{n-1}) - p_{n-1}(x_0)}{p_1(x_{n-1}) - p_1(x_0)} & \frac{p_n(x_{n-1}) - p_n(x_0)}{p_1(x_{n-1}) - p_1(x_0)} \\ 1 & \frac{p_2(x_n) - p_2(x_0)}{p_1(x_n) - p_1(x_0)} & \cdots & \frac{p_{n-1}(x_n) - p_{n-1}(x_0)}{p_1(x_n) - p_1(x_0)} & \frac{p_n(x_n) - p_n(x_0)}{p_1(x_n) - p_1(x_0)} \end{pmatrix}.$$

For $i = 2, \ldots, n$, let

$$q_{i-1}(x) = \frac{p_i(x) - p_i(x_0)}{p_1(x) - p_1(x_0)}.$$

Then by Lemma 2, $q_{i-1}(x)$ is a degree $i-1$ polynomial, and the above submatrix may be written as

$$\begin{pmatrix} 1 & q_1(x_1) & q_2(x_1) & \cdots & q_{n-1}(x_1) \\ & & \vdots & & \\ 1 & q_1(x_{n-1}) & q_2(x_{n-1}) & \cdots & q_{n-1}(x_{n-1}) \\ 1 & q_1(x_n) & q_2(x_n) & \cdots & q_{n-1}(x_n) \end{pmatrix}.$$

By the inductive assumption, this matrix is invertible, and can thus be row reduced to the identify matrix. Thereofre, adding back row 0 and column 0 of the original matrix, it follows that the entire matrx can be row reduced to the identity matrix, and is hence invertible.

Collorary 1. The n-dimensional Vandermonde matrix is invertible for all $n \geq 1$.

Proof of Collorary 1. This follows immediately from Theorem 1, by setting $p_i(x) = x^i$, $i = 1, \ldots, n$.

Example 3. Find the equation of the quadratic polynomial whose graph passes through the points $(0, 6)$, $(-1, 9)$, and $(2, 6)$.

The Discrete Fourier Transform

Polynomial interpolation can also be used to find the $n-1$ th degree complex trigonometric polynomial that best fits a set of n data points $(t_0, y_0), \ldots, (t_{n-1}, y_{n-1})$, where $y_j = E(t_j)$ is a sample of the signal $E(t)$ at time t_j, $j = 0, \ldots, n-1$. Assuming a time scale for which $E(t)$ is periodic over $[0, 2\pi]$, let $t_j = 2\pi j/n$. First notice that

$$p(t_j) = \sum_{k=-n+1}^{n-1} c_k e^{2\pi i j k/n} = \sum_{k=-n+1}^{n-1} c_k \omega_n^{jk}.$$

But since $\omega_n^{-kj} = \omega_n^{(n-k)j}$, we see that the terms of $p(t_j)$ with a negative k index may be combined with the terms having positive k index. Thus, assuming $t_j = 2\pi j/n$, we may write $p(t_j)$ as

$$p(t_j) = \sum_{k=0}^{n-1} c_k \omega_n^{jk} = \sum_{k=0}^{n-1} c_k (\omega_n^j)^k.$$

In other words $p(t_j)$ is the evaluation of the complex polynomial

$$p(x) = \sum_{k=0}^{n-1} c_k x^k$$

at the n th root of unity ω_n^j. This brings us to the definition of the discrete Fourier transform.

Discrete Fourier Transform. Given complex coefficients $c_0, \ldots, c_{n-1}$, let $p(x)$ be the polynomial

$$p(x) = \sum_{k=0}^{n-1} c_k x^k.$$

Then the n **th order discrete Fourier transform** is the function

$$\mathrm{DFT}_n(c_0, \ldots, c_{n-1}) = (y_0, \ldots, y_{n-1}),$$

where $y_j = p(\omega_n^j)$, $j = 0, \ldots, n-1$. In words the n th order discrete Fourier transform, takes as input the complex coefficients of a degree $n-1$ polynomial p, and returns the n-dimensional vector whose components are the evaluation of p at each of the n th roots of unity. Another way to write $\mathrm{DFT}_n(c_0, \ldots, c_{n-1})$ is $\mathrm{DFT}_n(p)$, where p is a polynomial of degree $n-1$.

Example 4. Compute $\mathrm{DFT}_4(0, 1, 2, 3)$.

The discrete Fourier transform maps from complex coefficients to data points. However, mapping from data points to coefficients is also important, since their must be some empirical basis for deriving the set of polynomial coefficients. This problem is called the n **th order inverse discrete Fourier transform**, and is represented by the function $\text{DFT}_n^{-1}(y_0, \ldots, y_{n-1})$, which takes as input n samples of a signal, and returns the coefficients of a polynomial $p(x)$ for which $p(\omega_n^j) = y_j$, $j = 0, \ldots, n-1$.

Example 5. Compute $\text{DFT}_4^{-1}(0, 1, -1, 2)$.

The Fast Fourier Transform

To compute the n th order DFT, one must evaluate an $(n-1)$-degree polynomial $p(x)$ at n different values, namely at the n th roots of unity. We now show that each polynomial evaluation can be performed in $\Theta(n)$ steps, which implies that the DFT_n can be computed in no more than $\Theta(n \cdot n) = \Theta(n^2)$ steps.

The method for evaluating an n th-degree polynomial $p_n(x)$ in $\Theta(n)$ steps is called Horner's algorithm. The algorithm is recursive with the following base case and recursive case.

Base case. If $n = 0$, then $p_0(x) = c_0$ is a constant. Thus $p_0(x)$ can be computed in 0 steps.

Recursive case. Suppose

$$p_n(x) = c_n x^n + c_{n-1} x^{n-1} + \cdots + c_1 x + c_0.$$

Then $p_n(x)$ can be written as $p_n(x) = xp_{n-1}(x) + c_0$, where

$$p_{n-1}(x) = c_n x^{n-1} + c_{n-1}x^{n-2} + \cdots + c_2 x + c_1.$$

Thus, $p_n(x)$ can be computed by first calling Horner's algorithm on $p_{n-1}(x)$, and then multiplying the result by x, and adding c_0. Moreover, if T_n is the number of steps needed to compute $p_n(x)$ using Horner's algorithm, then necessarily $T_n = T_{n-1} + 2$, since T_{n-1} steps are needed to compute $p_{n-1}(x)$, followed by 2 arithmetic steps: multiplying by x, and adding c_0.

To see that Horner's algorithm requires $\Theta(n)$ steps, notice that the recurrence $T_n = T_{n-1} + 2$ has an associated recursion tree that consists of a single branch with depth $n+1$, and that each node (except for the leaf) of the tree requires 2 computational steps of work, which yields a total of $2n = \Theta(n)$ steps.

Example 6. Show the sequence of polynomials p_0, p_1, p_2 that are evaluated when using Horner's algorithm to evaluate $p_3(x) = -4x^3 + 6x^2 + 10x - 7$.

Using Horner's algorithm, DFT_n can be computed $\Theta(n^2)$ steps. Unfortunately, for many applications where n is very large (say in the hundreds of thousands or millions), computing DFT_n may require an excessive amount of time. We now provide an algorithm for computing DFT_n in $\Theta(n \log n)$ steps. This algorithm is known as the **fast Fourier transform (FFT)**.

In computing $\text{DFT}_n(p)$, the algorithm starts by assuming that n is a power of 2. Next, assuming

$$p(x) = c_{n-1}x^{n-1} + c_{n-2}x^{n-2} + \cdots + c_1 x + c_0,$$

then $p(x)$ may also be written as $p(x) = p_0(x^2) + xp_1(x^2)$, where

$$p_0(x) = c_{n-2}x^{(n-2)/2} + c_{n-4}x^{(n-4)/2} + \cdots + c_2 x + c_0$$

is a degree $n/2 - 1$ polynomial whose coefficients consist of the even-indexed coefficients of $p(x)$, while

$$p_1(x) = c_{n-1}x^{(n-2)/2} + c_{n-3}x^{(n-4)/2} + \cdots + c_3 x + c_1$$

is a degree $n/2-1$ polynomial whose coefficients consist of the odd-indexed coefficients of $p(x)$

Now, since $p = p_0(x^2) + xp_1(x^2)$, $\mathrm{DFT}_n(p)$ can be computed by first evaluating $p_0(x^2)$ at each of the n th roots of unity, followed by evaluating $xp_0(x^2)$ at the n th roots of unity, and then adding the two output vectors. This is due to the additivity property of polynomials: $(r+s)(x) = r(x) + s(x)$, for any polynomials r and s. In other words

$$\mathrm{DFT}_n(p) = \mathrm{DFT}_n(p_0(x^2)) + \mathrm{DFT}_n(xp_1(x^2)).$$

But, by the Cancellation Rule (see Exercise 5),

$$\mathrm{DFT}_n(p_0(x^2)) = (p_0((\omega_n^0)^2), \ldots, p_0((\omega_n^{n/2-1})^2)p_0((\omega_n^{n/2})^2), \ldots, p_0((\omega_n^{n-1})^2)) =$$

$$(p_0(\omega_{n/2}^0), \ldots, p_0(\omega_{n/2}^{n/2-1})p_0(\omega_{n/2}^{n/2+0}), \ldots, p_0(\omega_{n/2}^{n/2+n/2-1})) =$$

$$(p_0(\omega_{n/2}^0), \ldots, p_0(\omega_{n/2}^{n/2-1})p_0(\omega_{n/2}^0), \ldots, p_0(\omega_{n/2}^{n/2-1})) =$$

$$\mathrm{DFT}_{n/2}(p_0) \cdot \mathrm{DFT}_{n/2}(p_0),$$

where the 2nd to last equality comes from the fact that $n/2+k \equiv k \bmod \mathrm{n}/2$, for each $k = 0, \ldots, n/2-1$. Note also that the last operation $\mathrm{DFT}_{n/2}(p_0) \cdot \mathrm{DFT}_{n/2}(p_0)$ means to concatenate vector $\mathrm{DFT}_{n/2}(p_0)$ with itself. For example,

$$(1,2,3) \cdot (1,2,3) = (1,2,3,1,2,3).$$

Similarly,

$$\mathrm{DFT}_n(p_1(x^2)) = \mathrm{DFT}_{n/2}(p_1) \cdot \mathrm{DFT}_{n/2}(p_1).$$

The above equations yield the following recursive algorithm for computing $\mathrm{DFT}_n(p)$, assuming n is a power of 2.

1. **Base case.** If $n = 2$, then return $\mathrm{DFT}_2(p)$ by using the definition of DFT.
2. Compute $\mathrm{DFT}_{n/2}(p_0)$ and form the vector $Y_0 = \mathrm{DFT}_{n/2}(p_0) \cdot \mathrm{DFT}_{n/2}(p_0)$.
3. Compute $\mathrm{DFT}_{n/2}(p_1)$ and form the vector $Y_1 = \mathrm{DFT}_{n/2}(p_1) \cdot \mathrm{DFT}_{n/2}(p_1)$.
4. For each $j = 0, \ldots, n-1$, replace component j of Y_1 with $\omega_n^j Y_{1j}$. In other words, $Y_{1j} \leftarrow \omega_n^j Y_{1j}$.
5. Return $Y_0 + Y_1$.

Example 7. Compute $\mathrm{DFT}_4(0, 1, 2, 3)$ using the FFT algorithm.

Theorem 2. Let $T(n)$ denote the number of steps needed to compute $\mathrm{DFT}_n(p)$ using the FFT algorithm. Then $T(n)$ satisfies the recurrence

$$T(n) = 2T(n/2) + an,$$

where $a > 0$ is a constant. Moreover, $T(n) = \Theta(n \log n)$.

Proof of Theorem 2. To compute $\mathrm{DFT}_n(p)$, we must first compute both $\mathrm{DFT}_{n/2}(p_0)$ and $\mathrm{DFT}_{n/2}(p_1)$. By definition, both of these computations will require $T(n/2)$ steps. Then vectors Y_0 and Y_1 must be formed. Both can be computed in a constant times n number of steps. For example, to compute Y_1, we need only concatenate $\mathrm{DFT}_{n/2}(p_1)$ with itself which takes n steps, and then multiply each component with the appropriate power of ω_n, which takes another n steps. Finally, computing the vector sum $Y_0 + Y_1$ requires another n steps. Putting this all together, we have

$$T(n) = 2T(n/2) + an,$$

where $a > 0$ is a constant.

To see that $T(n) = \Theta(n \log n)$, note that the algoroithm's recursion tree has a depth of $\log n - 1$, and that the number of nodes at depth i of the tree is 2^i. Moreover, the number of computing steps performed at each depth-i node is $a(n/2^i)$, since each node is performing a $\mathrm{DFT}_{n/2^i}$ computation which requires $a(n/2^i)$ steps. Thus, the total number of steps required at depth i is

$$2^i a(n/2^i) = an.$$

Therefore, since there are $\log n - 1$ depths, this gives the total work as $an(\log n - 1) = \Theta(n \log n)$.

The Inverted Fast Fourier Transform

The FFT algorithm can be modified to compute $\mathrm{DFT}_n^{-1}(y_0, \ldots, y_{n-1})$ in $\Theta(n \log n)$ steps. To begin, recall that we must find coefficients $c_0, \ldots, c_{n-1}$ for which

$$y_j = c_0 + c_1\omega_n^{j1} + c_2\omega_n^{j2} + \cdots + c_{n-1}\omega_n^{j(n-1)},$$

for $j = 0, \ldots, n-1$. This gives the following linear system of n equations expressed in matrix-equation form.

$$\begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega_n & \omega_n^2 & \cdots & \omega_n^{n-1} \\ 1 & \omega_n^2 & \omega_n^4 & \cdots & \omega_n^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \cdots & \omega_n^{(n-1)(n-1)} \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{n-1} \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{n-1} \end{pmatrix}$$

Notice that coefficient matrix is a Vandermonde matrix. Denote it by V_n. Notice that, for $0 \le j, k \le n-1$, entry (j, k) of V_n is ω_n^{jk}. The following theorem now shows how to compute V_n^{-1}.

Theorem 3. For $0 \le j, k \le n-1$, the (j, k) entry of V_n^{-1} is ω_n^{-jk}/n.

Proof of Theorem 3. Let matrix A have coefficients $a_{jk} = \omega_n^{-jk}/n$, for $0 \le j, k \le n-1$. We must show that $V_nA = I_n$, which would then imply that $A = V_n^{-1}$. Now consider entry (j, k) of V_nA.

Case 1: $j = k$. Then

$$(V_nA)_{jj} = \sum_{r=0}^{n-1} \omega_n^{jr}\omega_n^{-rj}/n = \frac{1}{n}\sum_{r=0}^{n-1} 1 = n/n = 1.$$

Case 2: $j \neq k$. Then

$$(V_nA)_{jk} = \sum_{r=0}^{n-1} \omega_n^{jr}\omega_n^{-rk}/n = \frac{1}{n}\sum_{r=0}^{n-1} \omega_n^{r(j-k)}.$$

Now $j - k$ is a nonzero integer that lies in the interval $[-n+1, n-1]$. Thus, $j - k$ is not divisible by n. Therefore, by Exercise 8,

$$\sum_{r=0}^{n-1} \omega_n^{r(j-k)} = 0,$$

which implies $(V_nA)_{jk} = 0$. Therefore, $V_nA = I_n$, and $(V_n^{-1})_{jk} = \omega_n^{-jk}/n$.

Knowing the inverse of V_n allows us to solve the above matrix equation by multiplying both sides by V_n^{-1}. This yields the equation

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{n-1} \end{pmatrix} = \begin{pmatrix} 1/n & 1/n & 1/n & \cdots & 1/n \\ 1/n & \omega_n^{-1}/n & \omega_n^{-2}/n & \cdots & \omega_n^{-(n-1)}/n \\ 1/n & \omega_n^{-2}/n & \omega_n^{-4}/n & \cdots & \omega_n^{-2(n-1)}/n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1/n & \omega_n^{-(n-1)}/n & \omega_n^{-2(n-1)}/n & \cdots & \omega_n^{-(n-1)(n-1)}/n \end{pmatrix} \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{n-1} \end{pmatrix}$$

Therefore, for $j = 0, \ldots, n-1$,

$$c_j = \frac{1}{n}(y_0 + y_1\omega_n^{-j1} + y_2\omega_n^{-j2} + \cdots + y_{n-1}\omega_n^{-j(n-1)}).$$

In words, c_j is obtained by evaluating $\frac{1}{n}p(x) = \frac{1}{n}(y_0 + y_1x + y_2x^2 + \cdots + y_{n-1}x^{n-1})$ at ω_n^{-j}. This leads to the following definition.

Inverted Discrete Fourier Transform. Given complex coefficients $y_0, \ldots, y_{n-1}$, let $p(x)$ be the polynomial

$$p(x) = \sum_{k=0}^{n-1} y_k x^k.$$

Then the n **th order inverted discrete Fourier transform** is the function

$$\mathrm{IDFT}_n(y_0, \ldots, y_{n-1}) = (c_0, \ldots, c_{n-1}),$$

where $c_j = \frac{1}{n}p(\omega_n^{-j})$, $j = 0, \ldots, n-1$. In words the n th order inverted discrete Fourier transform, takes as input the complex coefficients of a degree $n-1$ polynomial p, and returns the n-dimensional vector whose components are the evaluation of $\frac{1}{n} \cdot p$ at each of the inverses of the n th roots of unity. Another way to write $\mathrm{IDFT}_n(c_0, \ldots, c_{n-1})$ is $\mathrm{IDFT}_n(p)$, where p is a polynomial of degree $n-1$.

Notice that IDFT_n can be computed using the FFT algorithm, since it represents the same problem of evaluating a polynomial at each of the n th roots of unity (recall that the inverse of each root of unity is also a root of unity). To understand the recursion, we now have

$$\frac{1}{n}p(x) = \frac{1}{n}p_0(x^2) + \frac{x}{n}p_1(x^2) = \frac{1}{2}(\frac{2}{n}p_0(x^2) + \frac{2x}{n}p_1(x^2)).$$

Notice that evaluating $\frac{2}{n}p_0(x^2)$ at each ω_n^{-j} gives the same result as evaluating $p_0(x)$ at each $\omega_{n/2}^{-j}$, duplicating the resulting vector, and then dividing by $n/2$. Similarly, evaluating $\frac{2}{n}p_1(x^2)$ at each ω_n^{-j} gives the same result as evaluating $p_1(x)$ at each $\omega_{n/2}^{-j}$, duplicating the resulting vector, and then dividing by $n/2$. Thus,

$$\mathrm{IDFT}_n(p) = \frac{1}{2}(\mathrm{IDFT}_{n/2}(p_0) \cdot \mathrm{IDFT}_{n/2}(p_0) + x\mathrm{IDFT}_{n/2}(p_1) \cdot \mathrm{IDFT}_{n/2}(p_1),$$

where $x\mathrm{IDFT}_{n/2}(p_1) \cdot \mathrm{IDFT}_{n/2}(p_1)$ has the effect of multiplying component j of $\mathrm{IDFT}_{n/2}(p_1) \cdot \mathrm{IDFT}_{n/2}(p_1)$ by ω_n^{-j}. This leads us to the following IFFT algorithm.

IFFT Algorithm for computing $\mathbf{IDFT}_n(p) = \mathbf{IDFT}_n(y_0, \ldots, y_{n-1})$

1. **Base case.** If $n = 2$, then return $\mathrm{IDFT}_2(p)$ by using the definition of IDFT.
2. Compute $\mathrm{IDFT}_{n/2}(p_0)$ and form the vector C_0, where $C_0 = \mathrm{IDFT}_{n/2}(p_0) \cdot \mathrm{IDFT}_{n/2}(p_0)$.
3. Compute $\mathrm{IDFT}_{n/2}(p_1)$ and form the vector C_1, where $C_1 = \mathrm{IDFT}_{n/2}(p_1) \cdot \mathrm{IDFT}_{n/2}(p_1)$.
4. For each $j = 0, \ldots, n-1$, replace component j of C_1 with $\omega_n^{-j}C_{1j}$. In other words, $C_{1j} \leftarrow \omega_n^{-j}C_{1j}$.
5. Return $\frac{1}{2}(C_0 + C_1)$ which equals $\mathrm{DFT}_n^{-1}(y_0, \ldots, y_{n-1})$.

Example 8. Compute $\mathrm{DFT}_4^{-1}(0, 1, -1, 2)$ by a) using the definition of $\mathrm{IDFT}_4^{-1}(0, 1, -1, 2)$, and b) using the IFFT algorithm on $\mathrm{IDFT}_4(0, 1, -1, 2)$.

Exercises

1. Prove that for any two complex numbers c and d, $\overline{cd} = \overline{c}\overline{d}$

2. Determine the complex cube roots of unity.

3. Determine the complex 8th roots of unity.

4. For the 8th roots of unity, determine the inverse of each group element, and verify that $(a + bi)(a + bi)^{-1} = 1$ through direct multiplication.

5. Let $n \geq 1$, $d > 0$, and k be integers. Prove that $\omega_{dn}^{dk} = \omega_n^k$. This is called the **cancellation rule**.

6. Let n be an even positive integer. Prove that the square of each of the n th roots of unity yields the $n/2$ roots of unity. Moreover, each $n/2$ root of unity is associated with two different squares of n th roots of unity.

7. Show that $\omega_n^{n/2} = -1$, for all even $n \geq 2$.

8. For positive integer n and for integer j not divisible by n, prove that

$$\sum_{k=0}^{n-1} \omega_n^{jk} = 0.$$

   Hint: use the geometric series formula

$$\sum_{k=0}^{n-1} a^k = \frac{a^n - 1}{a - 1},$$

   which is valid when a is a complex number.

9. Find the equation of the quadratic polynomial whose graph passes through the points $(2, 13)$, $(-1, 10)$, and $(3, 26)$.

10. Find the equation of the cubic polynomial whose graph passes through the points $(0, -1)$, $(1, 0)$, $(-1, -4)$, and $(2, 5)$.

11. Compute $\mathrm{DFT}_4(1, -1, 2, 4)$.

12. Compute $\mathrm{DFT}_4(-1, 3, 4, 10)$.

13. Compute $\mathrm{DFT}_4^{-1}(0, 0, -4, 0)$.

14. Compute $\mathrm{DFT}_4^{-1}(2, 1 - i, 0, 1 + i)$.

15. Show the sequence of polynomials that are evaluated when evaluating $p(x) = x^3 - 3x^2 + 5x - 6$ using Horner's algorithm. Use the algorithm to evaluate $p(-2)$.

16. Show the sequence of polynomials that are evaluated when evaluating $p(x) = 2x^4 - x^3 + 2x^2 + 3x - 5$ using Horner's algorithm. Use the algorithm to evaluate $p(5)$.

17. Use the FFT algorithm to compute $\mathrm{DFT}_4(1, -1, 2, 4)$.
18. Use the FFT algorithm to compute $\mathrm{DFT}_4(-1, 3, 4, 10)$.
19. Compute $\mathrm{IDFT}_4(0, 0, -4, 0)$ using the definition.
20. Compute $\mathrm{IDFT}_4(2, 1-i, 0, 1+i)$ using the definition.
21. Use the IFFT algorithm to compute $\mathrm{IDFT}_4(0, 0, -4, 0)$.
22. Use the IFFT algorithm to compute $\mathrm{IDFT}_4(2, 1-i, 0, 1+i)$.

Exercise Solutions

1. Let $c = a + bi$, and $d = e + fi$. Then

$$\overline{cd} = \overline{(ae - bf) + i(af + be)} = (ad - bf) - i(af + be).$$

   On the other hand,

$$overline\overline{cd} = (a - bi)(e - fi) = (ae - bf) + i(-af - be) = (ae + bf) - i(af + be),$$

   which proves the claim.

2. For $j = 0$,

$$e^{\frac{(2\pi)(0)i}{3}} = 1.$$

   For $j = 1$,

$$e^{\frac{2\pi i}{3}} = -1/2 + \frac{\sqrt{3}i}{2}.$$

   For $j = 2$,

$$e^{\frac{4\pi i}{3}} = -1/2 - \frac{\sqrt{3}i}{2}.$$

3. For $j = 0$,

$$e^{\frac{(2\pi)(0)i}{3}} = 1.$$

   For $j = 1$,

$$e^{\frac{\pi i}{4}} = \frac{\sqrt{2}}{2} + \frac{\sqrt{2}i}{2}.$$

   For $j = 2$,

$$e^{\frac{\pi i}{2}} = i.$$

   For $j = 3$,

$$e^{\frac{3\pi i}{4}} = \frac{-\sqrt{2}}{2} + \frac{\sqrt{2}i}{2}.$$

   For $j = 4$,

$$e^{\pi i} = -1.$$

For $j = 5$,

$$e^{\frac{5\pi i}{4}} = \frac{-\sqrt{2}}{2} + \frac{-\sqrt{2}i}{2}.$$

For $j = 6$,

$$e^{\frac{3\pi i}{2}} = -i.$$

For $j = 7$,

$$e^{\frac{7\pi i}{4}} = \frac{\sqrt{2}}{2} + \frac{-\sqrt{2}i}{2}.$$

4. For example, $\omega_8^2 = i$ while $\omega_8^{-2} = \omega_8^6 = -i$, since $(i)(-i) = 1$. Similarly, $\omega_8^4 = -1$ while $\omega_8^{-4} = \omega_8^4 = -1$, since $(-1)(-1) = 1$.

5. By definition,

$$\omega_{dn}^{dk} = e^{\frac{2\pi i dk}{dn}} = e^{\frac{2\pi i k}{n}} = \omega_n^k.$$

6. For $j = 0, \ldots, n-1$,

$$(\omega_n^j)^2 = \omega_n^j \omega_n^j = \omega_n^{2j} = \omega_{n/2}^j,$$

where the last equality is due to the cancellation rule from Exercise 5. Thus the square of an n th root of unity is indeed an $n/2$ root of unity. Moreover, notice that j ranges from 0 to $n-1$. By definition, when j ranges from 0 to $n/2-1$, we obtain each $n/2$ root of unity. Then, due to the cyclic nature of the roots unity, when j ranges from $n/2$ to $n-1$, we once again obtain each $n/2$ root of unity. Therefore, each $n/2$ root of unity $\omega_{n/2}^j$ is the square of exactly two different n th-roots of unity, namely $(\omega_{n/2}^j)^2$ and $(\omega_{n/2}^{j+n/2})^2$.

7. We have, for even $n \geq 2$,

$$\omega_n^{n/2} = e^{(2\pi i/n)n/2} = e^{\pi i} = \cos\pi + i\sin\pi = -1.$$

8. Using the geometric series formula

$$\sum_{k=0}^{n-1} a^k = \frac{a^n - 1}{a - 1},$$

we have

$$\sum_{k=0}^{n-1} (\omega_n^j)^k = \sum_{k=0}^{n-1} \omega_n^{jk} =$$

$$\frac{\omega_n^{jn} - 1}{\omega_n^j - 1} = \frac{\omega_1^j - 1}{\omega_n^j - 1} = \frac{1 - 1}{\omega_n^j - 1} = 0,$$

where the first equality is due to the cancellation rule, and the 2nd to last equality is due to the fact that $\omega_1^1 = 1$. Notice also that the denominator is not equal to zero, since we assumed j is not divisible by n; i.e. $j \not\equiv 0 \bmod n$.

9. We desire a polynomial of the form $c_0 + c_1 x + c_2 x^2$. The three points imply the following system of equations.

$$c_0 + 2c_1 + 4c_2 = 13$$

$$c_0 - c_1 + c_2 = 10$$

$$c_0 + 3c_1 + 9c_2 = 26$$

Solving this system gives the polynomial $5 - 2x + 3x^2$.

10. We desire a polynomial of the form $c_0 + c_1x + c_2x^2 + c_3x^3$. The four points imply the following system of equations.

$$c_0 = -1$$
$$c_0 + c_1 + c_2 + c_3 = 0$$
$$c_0 - c_1 + c_2 - c_3 = -4$$
$$c_0 + 2c_1 + 4c_2 + 8c_3 = 5$$

Solving this system gives the polynomial $-1 + x - x^2 + x^3$.

11. $\text{DFT}_4(1, -1, 2, 4) = (6, -1 - 5i, 0, -1 - 5i)$

12. $\text{DFT}_4(-1, 3, 4, 10) = (16, -5 - 7i, -10, -5 + 7i)$

13. We desire a polynomial of the form $p(x) = c_0 + c_1x + c_2x^2 + c_3x^3$. Moreover, the four function values $p(1) = 0$, $p(i) = 0$, $p(-1) = -4$, and $p(-i) = 0$ imply the following system of equations.

$$c_0 + c_1 + c_2 + c_3 = 0$$
$$c_0 + ic_1 - c_2 - ic_3 = 0$$
$$c_0 - c_1 + c_2 - c_3 = -4$$
$$c_0 - ic_1 - c_2 + ic_3 = 0$$

Solving this system gives the polynomial $-1 + x - x^2 + x^3$.

14. We desire a polynomial of the form $p(x) = c_0 + c_1x + c_2x^2 + c_3x^3$. Moreover, the four function values $p(1) = 2$, $p(i) = 1 - i$, $p(-1) = 0$, and $p(-i) = 1 + i$ imply the following system of equations.

$$c_0 + c_1 + c_2 + c_3 = 2$$
$$c_0 + ic_1 - c_2 - ic_3 = 1 - i$$
$$c_0 - c_1 + c_2 - c_3 = 0$$
$$c_0 - ic_1 - c_2 + ic_3 = 1 + i$$

Solving this system gives the polynomial $1 + x^3$.

15. $p_0(x) = 1$, $p_1(x) = xp_0(x) - 3 = x - 3$, $p_2(x) = xp_1(x) + 5 = x^2 - 3x + 5$, $p_3(x) = xp_2(x) - 6 = x^3 - 3x^2 + 5x - 6$. $p_0(-2) = 1$, $p_1(-2) = -2(1) - 3 = -5$, $p_2(-2) = -2(-5) + 5 = 15$, $p_3(-2) = -2(15) - 6 = -36$.

16. $p_0(x) = 2$, $p_1(x) = xp_0(x) - 1 = 2x - 1$, $p_2(x) = xp_1(x) + 2 = 2x^2 - x + 2$, $p_3(x) = xp_2(x) + 3 = 2x^3 - x^2 + 2x + 3$, $p_4(x) = xp_3(x) - 5 = 2x^4 - x^3 + 2x^2 + 3x - 5$. $p_0(5) = 2$, $p_1(5) = 5(2) - 1 = 9$, $p_2(5) = 5(9) + 2 = 47$, $p_3(5) = 5(47) + 3 = 238$, $p_4(5) = 5(238) - 5 = 1185$.

17. $p_0(x) = 1 + 2x$, $\text{DFT}_2(1 + 2x) = (3, -1)$. Thus,

$$Y_0 = (3, -1, 3, -1).$$

Also, $p_1(x) = -1 + 4x$, and $\text{DFT}_2(-1 + 4x) = (3, -5)$. Thus,

$$Y_1 = (3, -5, 3, -5).$$

Furthermore, $Y_{1j} \leftarrow \omega_4^j Y_{1j}$ gives

$$Y_1 = (3, -5i, -3, 5i).$$

Finally, $\text{DFT}_4(1, -1, 2, 4) = Y_0 + Y_1 = (6, -1 - 5i, 0, -1 + 5i)$.

18. $p_0(x) = -1 + 4x$, $\mathrm{DFT}_2(-1+4x) = (3,-5)$. Thus,

$$Y_0 = (3,-5,3,-5).$$

Also, $p_1(x) = 3 + 10x$, and $\mathrm{DFT}_2(-1+4x) = (13,-7)$. Thus,

$$Y_1 = (13,-7,13,-7).$$

Furthermore, $Y_{1j} \leftarrow \omega_4^j Y_{1j}$ gives

$$Y_1 = (13,-7i,-13,7i).$$

Finally, $\mathrm{DFT}_4(-1,3,4,10) = Y_0 + Y_1 = (16,-5-7i,-10,-5+7i)$.

19. Input $(0,0,-4,0)$ corresponds with polynomial $p(x) = -4x^2$. Moreover,

$$p(\omega_4^{(-1)(0)}) = p(1) = -4,$$
$$p(\omega_4^{-1}) = p(-i) = 4,$$
$$p(\omega_4^{-2}) = p(-1) = -4,$$

and

$$p(\omega_4^{-3}) = p(i) = 4.$$

Thus,

$$\mathrm{IDFT}_4(0,0,-4,0) = \frac{1}{4}(-4,4,-4,4) = (-1,1,-1,1),$$

and so $\mathrm{DFT}_4^{-1}(0,0,-4,0) = (-1,1,-1,1)$, which corresponds with polynomial $-1+x-x^2+x^3$.

20. Input $(2,1-i,0,1+i)$ corresponds with polynomial $p(x) = 2 + (1-i)x + (1+i)x^3$. Moreover,

$$p(\omega_4^{(-1)(0)}) = p(1) = 4,$$
$$p(\omega_4^{-1}) = p(-i) = 0,$$
$$p(\omega_4^{-2}) = p(-1) = 0,$$

and

$$p(\omega_4^{-3}) = p(i) = 4.$$

Thus,

$$\mathrm{IDFT}_4(0,0,-4,0) = \frac{1}{4}(4,0,0,4) = (1,0,0,1),$$

and so $\mathrm{DFT}_4^{-1}(2,1-i,0,1+i) = (1,0,0,1),$, which corresponds with polynomial $1+x^3$.

21. $p_0(x) = -4x$, $\mathrm{IDFT}_2(-4x) = \frac{1}{2}(-4,4) = (-2,2)$. Thus,

$$C_0 = (-2,2,-2,2).$$

Also, $p_1(x) = 0$, and $\mathrm{IDFT}_2(0) = (0,0)$. Thus,

$$C_1 = (0,0,0,0).$$

Furthermore, $C_{1j} \leftarrow \omega_4^{-j} C_{1j}$ gives

$$C_1 = (0, 0, 0, 0).$$

Finally, $\text{IDFT}_4(0, 0, -4, 0) = \frac{1}{2}(C_0 + C_1) = \frac{1}{2}(-2, 2, -2, 2) = (-1, 1, -1, 1)$. Therefore,

$$\text{DFT}_4^{-1}(0, 0, -4, 0) = (-1, 1, -1, 1),$$

which corresponds with polynomial $-1 + x - x^2 + x^3$.

22. $p_0(x) = 2$, $\text{IDFT}_2(2) = \frac{1}{2}(2, 2) = (1, 1)$. Thus,

$$C_0 = (1, 1, 1, 1).$$

Also, $p_1(x) = (1 - i) + (1 + i)x$, and $\text{IDFT}_2((1 - i) + (1 + i)x) = \frac{1}{2}(2, -2i) = (1, -i)$. Thus,

$$C_1 = (1, -i, 1, -i).$$

Furthermore, $C_{1j} \leftarrow \omega_4^{-j} C_{1j}$ gives

$$C_1 = (1, -1, -1, 1).$$

Finally, $\text{IDFT}_4(2, 1 - i, 0, 1 + i) = \frac{1}{2}(C_0 + C_1) = (1, 0, 0, 1)$. Therefore,

$$\text{DFT}_4^{-1}(2, 1 - i, 0, 1 + i) = (1, 0, 0, 1),$$

which corresponds with polynomial $1 + x^3$.