Discrete Fourier Transform

Problems with star are just for fun: they are not a part of the homework, and their solutions are not used in the text.

We begin by recalling prerequisites: some facts about integers and complex numbers.

1. Arithmetic modulo N. Suppose that a positive integer $N \geq 2$ is given. Then every integer k can be divided by N with remainder, that is we have

$$k = pN + r, \quad \text{where} \quad p, r \quad \text{are integers, and} \quad 0 \leq r \leq N - 1.$$

We call this integer r the *remainder of* k *modulo* N. Two arithmetic operations, addition and multiplication, can be introduced on remainders: the sum of two remainders is the remainder of their sum as integers; multiplication is defined similarly. For example, $1 + 1 = 0 \pmod 2$, $7 + 10 = 5 \pmod{12}$, and $-7 = 5 \pmod{12}$.

1.1 Make the addition and multiplication tables modulo 2, modulo 4 and modulo 5.

1.2* Show that an integer has the same remainder modulo 3 as the sum of its digits in decimal system. Same modulo 9. In particular, an integer is divisible by 3 (or by 9) if and only if the sum of its digits is.

1.3* Find all integers k with the property that 2^k and 2^{k+1} have equal sums of digits.

1.4* Derive the following criterion of divisibility by 11: a number $a_0 + a_1 \times 10 + a_2 \times 10^2 + \ldots$ is divisible by eleven if and only if its alternating sum of digits $a_0 - a_1 + a_2 - \ldots$ is.

1.5 In the arithmetic modulo 4, which remainders have multiplicative inverses? Same question for moduli 2 and 5.

1.6* State the rule for arbitrary modulus. Which remainders have multiplicative inverses? For which moduli all reminders except 0 have multiplicative inverses?

2. Complex numbers. There are several ways to think about complex numbers. One way is algebraic: complex numbers are expressions of the form $a + bi$ with real a and b, which are added as polynomials of first degree with respect to the letter i. To be able to multiply them as polynomials, we set by definition $i^2 = -1$ (more precisely, $i^2 = -1 + 0 \cdot i$). So, for example, $(2 + i)(1 - i) = 3 - i$, $(1 + i)(1 - i) = 2$.

Another way is to identify complex numbers with matrices of the form

$$\begin{pmatrix} a & -b \\ b & a \end{pmatrix}, \quad \text{where} \quad a \text{ and } b \text{ are real.}$$

Then addition is the addition of matrices and the multiplication is multiplication of matrices.

2.1* Show that both definitions agree, that is they lead to the same addition and multiplication rules.

2×2 matrices are in one-to-one correspondence with linear transformations of the plane $\mathbf{R}^2$ (where we choose the standard basis).

2.2* Show that the matrices as above, representing complex numbers, correspond exactly to those linear transformations which preserve angles[1] between vectors.

Another way to characterize these transformations is to say that each of them multiplies the lengths of all vectors by the same factor, and preserves orientation ("clockwise/anticlockwise"). Linear transformations with these properties are called *conformal*.

The *norm* of a complex number $w = a+bi$ is by definition $|w| = \sqrt{a^2+b^2}$ (positive square root). It is also called *modulus* or *absolute value*. Notice that $|a+bi|^2 = \det A$, where A is the matrix above representing $a+bi$.

Complex numbers are in one-to-one correspondence with 2-dimensional vectors, addition corresponds to addition of vectors, and the norm corresponds to the norm of a vector. Using the standard rectangular coordinates in the plane, we also obtain a one-to-one correspondence between complex numbers and points in the plane. The set of all complex numbers of norm 1 is called the *unit circle*. It corresponds to the circle in the plane of radius 1 centered at the origin.

Using the polar coordinates in the plane, we can represent every complex number $w = a+bi$ in the form

$$w = r(\cos\theta + i\sin\theta), \quad \text{where} \quad r = |w|.$$

The polar angle θ in this representation is called an *argument* of w. Each non-zero complex number has infinitely many arguments: w does not change if we add to θ any multiple of 2π. If $w = 0$ its argument is undefined.

[1]The angle between two vectors x and y is defined anticlockwise from x to y. So the angle changes sign when x and y are interchanged. Sometimes this is called an *oriented angle*.

2.3 Write $1+i$, and $1/2+i\sqrt{3}/2$ in polar coordinates.
2.4 Show that the argument of a product of two complex numbers is the sum of their arguments.[2]

This result suggests the following definition of the exponent of a pure imaginary number:

$$\exp(i\theta) = \cos\theta + i\sin\theta, \quad \textbf{Euler's Formula.}$$

For example,

$$\exp(\pi i) = -1, \quad \exp(\pi i/2) = i \quad \text{and so on.}$$

The usual definition of the exponent of *any* complex number w is

$$\exp(w) = \sum_{n=0}^{\infty} \frac{w^n}{n!}.$$

One can show that this series is convergent for all complex w.
2.5* Obtain Euler's formula from this general definition, and using the known Taylor expansions of trigonometric functions.

The most important fact about complex numbers is the
Fundamental Theorem of Algebra. *Every non-constant polynomial with complex coefficients factors into polynomials of first degree. More precisely,*

$$P(w) = a(w-w_1)(w-w_2)\dots(w-w_n),$$

where $a, w_1, \dots, w_n$ *are complex numbers and* $n = \deg P$.

For example, $w^4+1 = (w-1)(w-i)(w+1)(w+i)$.
2.6 Find two solutions of the quadratic equation $w^2+w+1=0$.

3. Roots of Unity. N-th roots of unity are defined as solutions of the equation

$$w^N = 1. \tag{1}$$

There are exactly N distinct N-th roots of unity. For example, 2-nd roots of unity are 1 and -1, and 4-th roots are $1, i, -1, -i$. To find the 3-d roots of unity, first factor

$$w^3 - 1 = (w-1)(w^2+w+1),$$

[2]More precisely, this means that each of the infinitely many arguments of w_1w_2 is the sum of an argument of w_1 and an argument of w_2.

and then use Exercise 2.6.

3.1* Express all 5-th roots of unity in the form $a+bi$ using only arithmetic operations and radicals (square roots of positive numbers)[3], that is exponential and trigonometric functions are prohibited.

3.2* Without using Euler's Formula, exponents or trig, show that for every N, there are exactly N distinct roots of unity of degree N.

Using Euler's Formula, we can solve the equation (1) in the following way. First it follows from (1) that $|w|=1$, so by Euler's Formula, $w=\exp(i\theta)$ for some real θ. Substituting this to (1), we obtain

$$\exp(iN\theta)=1, \quad \text{thus} \quad iN\theta = 2\pi k, \quad k=0,1,2,\dots,$$

so $\theta = 2\pi k/N$. Not all these θ give different solutions of (1). Namely, $\exp(2\pi ik/N)=\exp(2\pi im/N)$ if and only if $k-m$ is divisible by N. So there are exactly N distinct solutions

$$w_k = \exp(2\pi ik/N), \quad k=0,1,2,\dots,N-1. \tag{2}$$

We see that there is a one-to-one correspondence between N-th roots of unity and remainders modulo N, and *product of roots of unity corresponds to the sum of the remainders.*

Notice that $w_k = w_1^k$, that is all N-th roots of unity are powers of one of them. In general, an N-th root of unity w is called *primitive* if all N-th roots of unity are powers of w. Thus the root w_1 (as well as w_{-1}) is always primitive.

3.3* Find all primitive roots of degrees 3, 4 and 12.

3.4* Suppose that N is even, and w is a primitive N-th root of unity. Show that w^2 is a primitive root of degree $N/2$.

Try to state a general rule: which roots of degree N are primitive.

Now we discuss *sums* of the roots of unity.

3.5 Let w be any primitive root of unity, for example $w_1=\exp(2\pi i/N)$. Then

$$\sum_{k=0}^{N-1} w^{mk} = \begin{cases} N & \text{if } N \text{ divides } \quad m, \\ 0 & \text{otherwise.} \end{cases}$$

Hint: use the geometric progression formula to compute the LHS.

[3]The question, for which N this is possible was solved completely by Gauss.

In words: *sum of all roots of unity of fixed degree is* 0. *Sum of* m*-th powers of all* N*-th roots of unity is* N *when* m *is divisible by* N *and zero otherwise.*

4. Discrete Fourier Transform. We fix an integer $N \geq 2$, and a primitive N-th root of unity w, for example we may take $w = w_1 = \exp(2\pi i/N)$. (If you did not care to solve 3.4, just assume that $w = \exp(2\pi i/N)$.) The symbol

$$\sum_k$$

will always mean summation over all remainders k modulo N.

Let $f = (f(0), \ldots, f(N-1))$ be a vector of dimension N, in general, with complex coordinates. The coordinates are indexed by remainders modulo N. We define its (discrete) Fourier Transform as another complex vector $F = (F(0), \ldots, F(N-1))$ of the same dimension N,

$$F(k) = \sum_n w^{kn} f(n). \tag{3}$$

Thus Fourier Transform is a linear operator represented by the $N \times N$ matrix

$$A = (a_{kn}), \quad a_{kn} = w^{kn},$$

which is called the Fourier Matrix of order N.
4.1 Write explicitly the Fourier matrix of order 4 using $w = \exp(2\pi i/4) = i$.
4.2 Find the Fourier Transform of the vector $(2, 1, -2, 1)$ using the matrix from 4.1.
4.3 Using Exercise 3.5, verify that the inverse matrix is

$$A^{-1} = \frac{1}{N}(w^{-kn}).$$

In other words, a vector f can be recovered from its Fourier Transform F by the *Fourier Inversion Formula*:

$$f(n) = \frac{1}{N} \sum_k w^{-nk} F(k).$$

4.4* Prove the *Parseval Indentity*

$$\sum_n |F(n)|^2 = N \sum_n |f(n)|^2.$$

This means that Fourier Transform increases the lengths of vectors by the factor of $\sqrt{N}$.

5. Fast Fourier Transform. Fourier Transform is one of the most basic tools in Mathematics as well as in all kinds of data processing (for science, engineering and communication). One of the important recent applications is to multiplication of large integers needed for coding, for example. Unfortunately there is no time in this course to explain any of these applications. Many achievements of the modern "computer revolution" would be impossible without an algorithm for doing Fourier Transforms fast.

In general, multiplication of an N-vector by an $N \times N$ matrix requires N^2 multiplications of numbers. (For simplicity we don't count additions; their cost is usually less.) So it was an important discovery that Fourier transform can be computed by about $cN \log_2 N$ multiplications, where c is an absolute constant.

5.1 Suppose that your computer screen has 10^6 pixels. So a picture on your screen in represented by a vector of dimension one million. Processing this information is usually done with Fourier Transform. How many multiplications will the usual matrix multiplication require to compute one such transform? Suppose that your computer can perform 50×10^6 multiplications per second. How long will it take to compute Fourier Transform of one picture? How long will it take using the Fast Fourier Transform algorithm which requires, say $4N \log_2 N$ multiplications?

I hope this example explains what I mean in the above sentence on "achievements of computer revolution".

For the Fast Fourier Transform (FFT), it is convenient to choose $N = 2^n$ (always do this if you want to implement this algorithm!) The FFT algorithm is based on the following computation:

$$\begin{aligned} F(k) &= \sum_j w^{kj} f(j) \quad \text{write even and odd } j \text{ separately} \\ &= \sum_{j=0}^{N/2-1} w^{2kj} f(2j) + \sum_{j=0}^{N/2-1} w^{k(2j+1)} f(2j+1) \\ &= \sum_{j=0}^{N/2-1} (w^2)^{kj} f(2j) + w^k \sum_{j=0}^{N/2-1} (w^2)^{kj} f(2j+1). \end{aligned}$$

Now w^2 is a primitive root of degree $N/2$ (by Exercise 3.4, or if you did not do it, and $w = \exp(2\pi i/N)$, then $w^2 = \exp(2\pi i/(N/2))$.) So the last line of the formula above is the sum of two Fourier transforms of dimension $N/2$, first performed on even components of the vector f, second on odd components. Thus doing FT of dimension N is reduced to doing two transforms of dimension $N/2$ each, plus N additional multiplications.

Let $C(N)$ be the number of multiplications needed to compute FT of dimension N. Our argument shows that

$$C(N) = 2C(N/2) + N.$$

If $N = 2^n$, as we assume, the last equation gives a recurrence relation for $C(N)$. A solution of this recurrency is $C^*(N) = N \log_2 N$, indeed

$$2C^*(N/2)+N = 2(N/2)\log_2(N/2)+N = N\log_2 N - N\log_2 2+N = C^*(N).$$

As $C(2) = C^*(2) = 2$, we conclude that $C(N) = C^*(N)$.

It is hard to tell in few sentences who was the first to discover this algorithm. It was published in its present form for the first time by G. Danielson and C. Lanczos at Purdue University in 1942. Danielson was a graduate student in physics; he studied refraction of X-rays in liquids in his PhD thesis. Fourier Transform is the main tool in these questions, and in optics in general. Lanczos was apparently his advisor. (One of the greatest applied mathematicians of this century, he was once Einstein's assistant, then escaped to the US from the Nazi prosecution, and in 1940-s had a temporary job at Purdue).

In the introduction to their paper they wrote:

"If a modern mechanical analyzer[4] is available, the evaluation of Fourier integral presents no difficulty. It is our purpose to show that, for occasional analyses at least, one need not depend on such costly instruments, even when the required number of coefficients is very large."

The paper of Lanczos and Danielson went unnoticed at that time.[5] The algorithm was rediscovered by J. Cooley and J. Tukey in 1965. (Cooley was a computer programmer for IBM, and Tukey, who suggested the algorithm, was a scientific advisor of the US President (JFK). Tukey's concern was the

[4] An analog computer specially designed to evaluate Fourier Transform. You can see a picture of one of these on my web page.

[5] Lanczos' appointment was not renewed after the war. He could not find a tenure position until he grew 60.

possibility of detecting Soviet underground nuclear tests by seismic observations. This also requires Fourier Transform. Later some Soviet scientists said that they are proud to learn that their nuclear tests stimulated the development of the Fast Fourier Transform:-) The paper of Cooley and Tukey was published at exactly the right time, when fast digital computers just became available available.

Later it was discovered that several people had this idea before Cooley, Tukey, Lanczos and Danielson, the oldest reference I know is on the work of Gauss (who was computing the orbits of small planets).

5.2 According to Danielson and Lanczos, their first calculation at Purdue took 10 min to find the FT of a 8-vector (by hand, of course), 25 min for a 16-vector, 60 min for a 32-vector, and 120 min for a 64-vector. Plot these data and discuss, whether they are consistent with the theoretical result, that the time is proportional to $N \log N$. Estimate the coefficient of proportionality (with the least squares or without). Please, do all calculations by hand.

5.3* Suppose that a vector (a, b, c, d, e, f, g, h) has FT (A, B, C, D, E, F, G, H). Find the vector of dimension 4 whose FT is (A, C, E, G).