INTRODUCTION TO FOURIER ANALYSIS OVER FINITE ABELIAN GROUPS

WILLIAM DEMEO

ABSTRACT. These are lecture notes to accompany two lectures, intended for linear algebra students, covering just a few basic definitions and facts that are useful in applications such as **d**igital (or **d**iscrete-time) **s**ignal **p**rocessing (DSP).[1] The objective is to acquaint students with yet another area in which vector spaces and linear transformations naturally arise.

The work of Richard Tolimieri and Myoung An is the main source of inspiration for these lectures. (See [1], for example.)

Part 1. Translations and Convolutions (Monday)

1. Definitions and Notations

We view a "digital signal" as simply a complex-valued function defined on a finite set. For example, if we sampled an analog signal at n points, and if we let $f(k)$ denote the value we observed at point k, then the result is a function $f : \{0, 1, \dots, n-1\} \to \mathbb{C}$, which we may identify with the n-tuple (or vector) of values, $\mathbf{f} = (f(0), f(1), \dots, f(n-1))$. Thus, we may equate the set of all such "length-n" signals with the (complex) vector space $\mathbb{C}^n$.[2]

Throughout this lecture, we assume that our signals are always defined on the "index set"

$$\underline{n} := \{0, 1, \dots, n-1\},$$

We need to endow this set with some additional structure. In particular, we want to define a binary "addition" operation on the index set, so that we can give meaning to "translations" of signals and evaluate expressions such as $f(x + y)$. We have to be a little bit careful when doing this, because if the signal is only defined on the set $\underline{n}$, then $f(x + y)$ only has meaning if the binary addition operation $(x, y) \mapsto x + y$ returns a number in $\underline{n}$ for every pair $(x, y) \in \underline{n} \times \underline{n}$. To make this work, we use the so called *cyclic group* $\mathbb{Z}/n\mathbb{Z}$ of integers modulo n for the index set, which we define in just a moment. But first let me note that, although it's possible the exploit the group structure of $\mathbb{Z}/n\mathbb{Z}$ to simplify the presentation of DSP, we will hardly use any group theory in these notes. Having said that, we do need to know a few very basic facts about $\mathbb{Z}/n\mathbb{Z}$.[3] But first, here is the definition we will use. For simplicity, we will identify $\mathbb{Z}/n\mathbb{Z}$ with the set $\{0, 1, \dots, n-1\}$, along with the following operations defined on this set:

- (binary addition) $(x, y) \mapsto x + y$ is addition modulo n

Date: December 8, 2015

The latest draft is available at github.com/williamdemeo .

[2]The vector space $\mathbb{C}^n$ is described in detail in the first section of Chapter 7 of our textbook.

[3]If you are curious to know more about the cyclic group $\mathbb{Z}/n\mathbb{Z}$, see Wikipedia: Abelian group.

- (unary negation) $x \mapsto -x$ is $n - x$
- (nullary identity) 0 satisfies $x + 0 = x = 0 + x$.

Note that addition modulo n is commutative ($x+y = y+x$, for all x, y) and associative ($(x+y)+z = x + (y + z)$ for all x, y, z). Although it is not a standard convention, it will greatly simplify our notation to let the symbol $\underline{n}$ stand for both the set $\{0, 1, \dots, n-1\}$ as well as the cyclic group $\mathbb{Z}/n\mathbb{Z}$ with operations defined as above. So, $\underline{n}$ is the *algebraic structure* $\langle\{0, 1, \dots, n-1\}, +, -, 0\rangle$, where $+$, $-$, and 0 are taken to be the binary, unary, and nullary (resp.) operations defined above.

One fact about the group $\underline{n}$ that is easily deduced from the definition is that $nx = 0$ holds for all $x \in \underline{n}$.

Exercise 1. Prove that $nx = 0$ holds for all $x \in \underline{n}$.

Denote by $\mathcal{L}(\underline{n})$ the set of all complex-valued functions defined on $\underline{n}$. Define a (binary) addition operation on $\mathcal{L}(\underline{n})$ as follows: $(f, g) \mapsto f + g$ where $f + g$ is the function in $\mathcal{L}(\underline{n})$ defined for each $x \in \underline{n}$ by

$$(f + g)(x) = f(x) + g(x). \tag{1.1}$$

(On the right-hand side of (1.1) is the usual addition of two complex numbers, $f(x)$ and $g(x)$.) For each $c \in \mathbb{C}$, define a (unary) scalar multiplication operation on $\mathcal{L}(\underline{n})$ by $f \mapsto cf$, where cf is the function in $\mathcal{L}(\underline{n})$ defined for each $x \in \underline{n}$ by

$$(cf)(x) = cf(x). \tag{1.2}$$

(On the right is the usual multiplication of two complex numbers, c and $f(x)$.)

Exercise 2. Prove that set $\mathcal{L}(\underline{n})$, along with vector addition and scalar multiplication defined in the previous paragraph, is a vector space. What is the zero vector?

The vector space $\mathcal{L}(\underline{n})$ is the *space of complex-valued functions* on $\underline{n}$. The elements of $\mathcal{L}(\underline{n})$ are functions $f : \{0, 1, \dots, n-1\} \to \mathbb{C}$, and we can identify each such function with the n-tuple (or vector) of its values, $\mathbf{f} = (f(0), f(1), \dots, f(n-1))$. Thus, as mentioned above, we may identify the space $\mathcal{L}(\underline{n})$ of complex-valued functions with the n-dimensional complex vector space $\mathbb{C}^n$, which is described in the first section of Chapter 7 of our textbook.

1.1. **Convolution and Translation.**

An important operation in signal processing and elsewhere is called convolution which can be viewed as a sort of "mixing" of two signals. Given two signals, f and g in $\mathcal{L}(\underline{n})$, we define the *convolution of g by f* to be the function $f * g \in \mathcal{L}(\underline{n})$ defined for each $x \in \underline{n}$ as follows:

$$(f * g)(x) = \sum_{y \in \underline{n}} f(y)g(x - y). \tag{1.3}$$

Note that the expression on the right hand side of (1.3) makes sense because of the way we defined addition and negation as (group) operations on $\underline{n}$. In particular, for all $x, y \in \underline{n}$, we have $x - y \in \underline{n}$.

Warning: Many authors call the operation defined in (1.3) "cyclic convolution" and reserve the unqualified term "convolution" for the same operation defined on $\mathcal{L}(\mathbb{Z})$.[4]

The best way to view the expression in (1.3) is as a linear combination of translations of the function g. Indeed, for each $y \in \underline{n}$, define the *translation by* y as the mapping $\mathsf{T}(y) : \mathcal{L}(\underline{n}) \to \mathcal{L}(\underline{n})$ that takes a function $g \in \mathcal{L}(\underline{n})$ to the ("y-translated") function $\mathsf{T}(y)g$, which is defined for each $x \in \underline{n}$ as follows:

$$(\mathsf{T}(y)g)(x) = g(x - y).$$

Example 1.1. Suppose $g \in \mathcal{L}(\underline{4})$ is $(g(0), g(1), g(2), g(3)) = (5, 6, 7, 8)$. Then $\mathsf{T}(1)g$ is

$$(g(0-1), g(1-1), g(2-1), g(3-1)) = (g(3), g(0), g(1), g(2)) = (8, 5, 6, 7).$$

So we see that $\mathsf{T}(1)$ performs a (cyclic) shift on its operand g; it moves all the values of g one place to the right, and wraps the end of the vector around to the beginning. Similarly, $\mathsf{T}(3)g$ is $(g(1), g(2), g(3), g(0)) = (6, 7, 8, 5)$.

Exercise 3. Prove that for each $y \in \underline{n}$ the translation by y operator $\mathsf{T}(y) : \mathcal{L}(\underline{n}) \to \mathcal{L}(\underline{n})$ is a linear transformation.

For each $x \in \underline{n}$ denote by $e_x \in \mathcal{L}(\underline{n})$ the function that is 1 at x and 0 otherwise. That is,

$$e_x(k) = \begin{cases} 1, & k = x, \\ 0, & k \neq x. \end{cases}$$

The set $\{e_x : x \in \underline{n}\}$ is called the *canonical basis* for $\mathcal{L}(\underline{n})$. The next exercise asks you to give partial justification for this terminology.

Exercise 4. Prove that the canonical basis is a basis for $\mathcal{L}(\underline{n})$.

Translations permute the elements of the canonical basis,

$$\mathsf{T}(y)e_x = e_{x+y}, \quad x, y \in \underline{n}.$$

For each $y \in \underline{n}$, $\mathsf{T}(y)$ is a linear transformation of $\mathcal{L}(\underline{n})$ so we can represent $\mathsf{T}(y)$ as a matrix with respect to the given basis. For example, relative to the canonical basis $\mathcal{E} = \{e_0, e_1, \ldots, e_{n-1}\}$, the linear transformation $\mathsf{T}(1)$ is represented by the "cyclic shift" matrix

$$[\mathsf{T}(1)]_{\mathcal{E}} = \begin{bmatrix} 0 & 1 & & \cdots & 0 \\ & 0 & 1 & & \vdots \\ \vdots & & \ddots & \ddots & \\ 0 & & & 0 & 1 \\ 1 & 0 & \cdots & & 0 \end{bmatrix}.$$

If we denote by $\mathcal{T}(\underline{n})$ the collection of all translations on $\mathcal{L}(\underline{n})$, that is,

$$\mathcal{T}(\underline{n}) := \{\mathsf{T}(y) : y \in \underline{n}\},$$

[4] $\mathcal{L}(\mathbb{Z})$ denotes the space of complex-valued functions defined on $\mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$; the convolution of two such functions is $(f * g)(x) = \sum_{y \in \mathbb{Z}} f(y)g(x - y)$.

then the mapping $\mathsf{T} : \underline{n} \to \mathcal{T}(\underline{n})$ satisfies, for all $x, y \in \underline{n}$,

$$\mathsf{T}(x+y) = \mathsf{T}(x)\mathsf{T}(y) \quad \text{and} \quad \mathsf{T}(y)^{-1} = \mathsf{T}(-y). \tag{1.4}$$

Thus $\mathcal{T}(\underline{n})$ is closed under operator composition and operator inverse. Moreover, $\mathcal{T}(\underline{n})$ is a *commuting family* of linear operators, that is $\mathsf{T}(x)\mathsf{T}(y) = \mathsf{T}(y)\mathsf{T}(x)$ holds for all $x, y \in \underline{n}$. Indeed, since addition modulo n is commutative, we have by (1.4)

$$\mathsf{T}(x)\mathsf{T}(y) = \mathsf{T}(x+y) = \mathsf{T}(y+x) = \mathsf{T}(y)\mathsf{T}(x).$$

Also, by Exercise 1, we have

$$\mathsf{T}(y)^n = \mathsf{T}(ny) = \mathsf{T}(0) = \mathrm{id}_{\mathcal{L}(\underline{n})},$$

where $\mathrm{id}_{\mathcal{L}(\underline{n})}$ denotes the identity operator (that is, $\mathrm{id}_{\mathcal{L}(\underline{n})}(f) = f$). For $f \in \mathcal{L}(\underline{n})$, define the function $\mathsf{C}(f) : \mathcal{L}(\underline{n}) \to \mathcal{L}(\underline{n})$ as the following linear combination of translations:

$$\mathsf{C}(f) = \sum_{y \in \underline{n}} f(y)\mathsf{T}(y).$$

We call $\mathsf{C}(f)$ *convolution by* f. As a linear combination of linear transformations, convolution by f is itself linear. We can now define for all $f, g \in \mathcal{L}(\underline{n})$ the *convolution of* g *by* f by applying the "convolution by f" operator to g, resulting in the function

$$\mathsf{C}(f)g = \sum_{y \in \underline{n}} f(y)\mathsf{T}(y)g.$$

This is the function defined for each $x \in \underline{n}$ by

$$(\mathsf{C}(f)g)(x) = \sum_{y \in \underline{n}} f(y)\mathsf{T}(y)g(x) = \sum_{y \in \underline{n}} f(y)g(x-y),$$

which agrees with the function $f * g$ defined above, as expected.

Note that, by the change of variables $z = x - y$, we have

$$(\mathsf{C}(f)g)(x) = \sum_{y \in \underline{n}} f(y)g(x-y) = \sum_{x-z \in \underline{n}} f(x-z)g(z) = \sum_{z \in \underline{n}} g(z)f(x-z) = (\mathsf{C}(g)f)(x),$$

so that $\mathsf{C}(f)g = \mathsf{C}(g)f$.

Exercise 5.

a (for programmers). In your favorite programming language, try to implement the cyclic convolution described in this section.[5] Your program should accept as input two functions (lists? vectors? ...your choice) f and g of type $\{0, 1, \ldots, n-1\} \to \mathbb{C}$ and return the function $\mathsf{C}(f)g$ (the convolution of g by f). (To make your life easier, just get it to work for the special case of real-valued functions, and don't worry about handling complex numbers at this point.)

b (for functional programmers). See if you can write your convolution method so that it can be partially applied. That is, it can take a single function f as input and return the operator $\mathsf{C}(f)$, which is a function of type $\mathcal{L}(\underline{n}) \to \mathcal{L}(\underline{n})$. (That is, if you apply your function to a single $f \in \mathcal{L}(\underline{n})$, then the output is the function $\mathsf{C}(f)$ that takes another function $g \in \mathcal{L}(\underline{n})$ as input and returns $\mathsf{C}(f)g$.) (*Hint:* If your favorite programming language has `map` and `reduce`, use them!)

[5]This can be done quite easily in Scala or Python, for example.

The DSP of functions defined on $\underline{n}$ is mainly about *translation-invariant subspaces* and *translation-invariant linear operators* on $\mathcal{L}(\underline{n})$. A subspace $V \leq \mathcal{L}(\underline{n})$ is called *translation-invariant* if for all $f \in V$ and $y \in \underline{n}$, we have $\mathsf{T}(y)f \in V$. A linear operator $\mathsf{M} : \mathcal{L}(\underline{n}) \to \mathcal{L}(\underline{n})$ is called *translation-invariant* if for all $y \in \underline{n}$ we have $\mathsf{M}\,\mathsf{T}(y) = \mathsf{T}(y)\,\mathsf{M}$.

Exercise 6.

a. Suppose V is a translation-invariant subspace. Prove that V is also "convolution-invariant." That is, for $g \in V$ and $f \in \mathcal{L}(\underline{n})$, we have $\mathsf{C}(f)g \in V$.

b. Suppose M is a translation-invariant linear operator on $\mathcal{L}(\underline{n})$. Prove that M is also "convolution-invariant." That is, for $f \in \mathcal{L}(\underline{n})$, we have $\mathsf{C}(f)\,\mathsf{M} = \mathsf{M}\,\mathsf{C}(f)$.

c. Suppose λ is an eigenvalue of the translation operator $\mathsf{T}(y)$. Say precisely what this means and then describe the eigenspace $\mathrm{E}(\lambda)$ of $\mathsf{T}(y)$ associated with λ. Is $\mathrm{E}(\lambda)$ a translation-invariant subspace of $\mathcal{L}(\underline{n})$?

Part 2. Fourier Analysis (Tuesday)

Before proceeding, students who are not already familiar with complex numbers should take a moment to familiarize themselves with the basic definitions and facts.[6] Please consult either the breif appendix section below, or read Wikipedia: Complex number or type something like "complex numbers primer" into google.

Fourier analysis provides the tools for the construction and analysis of translation-invariant subspaces and operators of $\mathcal{L}(\underline{n})$. These tools begin with the construction of a special basis of $\mathcal{L}(\underline{n})$ as follows: For $y \in \underline{n}$, let $\varepsilon_y \in \mathcal{L}(\underline{n})$ to be the complex-valued function whose value at x is $e^{2\pi i \frac{yx}{n}}$, where yx is the product modulo n. Thus, for each $x \in \underline{n}$,

$$\varepsilon_y(x) = e^{2\pi i \frac{yx}{n}}.$$

The collection $\{\varepsilon_y : y \in \underline{n}\}$ is a basis for $\mathcal{L}(\underline{n})$ called the *exponential basis*. To prove that this set really does form a basis, we need the following important lemma (which is proved in Appendix Section A):

Lemma 1.2.

$$\sum_{j \in \underline{n}} e^{2\pi i \frac{kj}{n}} = \begin{cases} n, & k \equiv 0 \mod n, \\ 0, & \text{otherwise.} \end{cases} \tag{1.5}$$

The formula (1.5) is at the heart of some of the most important DSP theorems and algorithms, including the Fast Fourier Transform and Shannon sampling.

1.2. **Translation eigenvector basis.** Let $T : V \to V$ be a linear transformation. Recall that an *eigenvalue* of T is a scalar λ for which there exists a nonzero vector $v \in V$ satisfying $Tv = \lambda v$. Let us call such a vector v a *T-eigenvector*. Now suppose that $\mathcal{T}$ is a collection of linear transformations on V and suppose there is a single vector v that is a T-eigenvector for every $T \in \mathcal{T}$. We call such

[6] In lecture last week we covered almost everything we need to know about complex numbers except for Euler's forumula and roots of unity. Appendix Section A covers these topics.

v a $\mathcal{T}$-eigenvector. Of course, the eigenvalue associated with each T may vary, but we can define a function $\lambda : \mathcal{T} \to \mathbb{C}$ that denotes the corresponding eigenvalue; that is, for each $T \in \mathcal{T}$, we have

$$Tv = \lambda(T)v.$$

For example, above we considered the set $\mathcal{T}(\underline{n}) = \{\mathsf{T}(y) : y \in \underline{n}\}$ of translations defined on $\mathcal{L}(\underline{n})$. In this section, we will prove that every exponential function ε_y is a $\mathcal{T}(\underline{n})$-eigenvector, and that the so called "exponential basis," defined above by $\{\varepsilon_y : y \in \underline{n}\}$, really is a basis for $\mathcal{L}(\underline{n})$. The first of these facts is easy to prove, so we dispense with it immediately.

Theorem 1.3. *Each exponential function $\varepsilon_z(x) = e^{2\pi i \frac{zx}{n}}$ is a $\mathcal{T}(\underline{n})$-eigenvector with eigenvalues $\lambda(\mathsf{T}(y)) = \varepsilon_z(-y)$. That is,*

$$\mathsf{T}(y)\varepsilon_z = \varepsilon_z(-y)\varepsilon_z.$$

Proof. Fix $x \in \underline{n}$. Then,

$$\mathsf{T}(y)\varepsilon_z(x) = \varepsilon_z(x-y) = e^{2\pi i \frac{z(x-y)}{n}} = e^{2\pi i \frac{z(-y)}{n}} e^{2\pi i \frac{zx}{n}} = \varepsilon_z(-y)\varepsilon_z(x).$$

In other terms, $\mathsf{T}(y)\varepsilon_z = \varepsilon_z(-y)\varepsilon_z$, as desired. □

The other main result we want to prove in this section is

Theorem 1.4. *The set $\{\varepsilon_y : y \in \underline{n}\}$ is a basis for $\mathcal{L}(\underline{n})$.*

This takes more work...

(to be continued)

Appendix A. Brief introduction to complex numbers

We will review just a couple of basic facts about complex numbers and complex exponential functions that we use repeatedly in these lectures.

Hopefully we all learned in high school that there is no real number whose square is -1. In other words, the field of real numbers contains no element x such that $x^2 = -1$. Nonetheless, it can be very helpful to have a mathematical formalism that can deal with the object $\sqrt{-1}$ and even put it to work for us. This is where "complex numbers" come from. We extend the real numbers and give a special name to the object $\sqrt{-1}$. Most authors use i to refer to $\sqrt{-1}$, but some authors (expecially engineers and physicists) like to use j. We will adopt the convention $i = \sqrt{-1}$. So, for example, we have $i^2 = -1$ and $\sqrt{-9} = \sqrt{(-1)(9)} = \sqrt{-1}\sqrt{9} = i3$ and $\sqrt{-2} = i\sqrt{2}$.

More generally, a *complex number* is a number of the form $z = a + ib$ for some real numbers a and b. The first component $a = \Re(z)$ is called the *real part* of z, and the second component $b = \Im(z)$ is called the imaginary part. The set of all complex numbers is denoted by $\mathbb{C}$. That is,

$$\mathbb{C} = \{a + ib : a, b \in \mathbb{R}\}.$$

Note that the complex numbers include the real numbers (as complex numbers having imaginary part 0). It is sometimes helpful to view the complex number $z = a + ib$ as an ordered pair (a, b) and in that sense we see z as an element of the so called *complex plane* (also denoted by $\mathbb{C}$), which looks

a lot like $\mathbb{R}^2$ (the real plane), with one major difference: in the complex plane, the pair $(a, b) \in \mathbb{C}$ gives the coordinates of a point with respect to the basis $\{1, i\}$ (instead of the basis $\{(1, 0), (0, 1)\}$).

Thus, we can depict the complex number $z = a + ib$ as the point in the complex plane with coordinates (a, b), where numbers on the vertical axis are scalar multiples of i. In the complex plane, the vertical axis is sometimes called the *imaginary axis*, which the horizontal axis is sometimes called the *real axis*.

The *complex conjugate* of $z = a + ib$ is $\bar{z} = a - ib$. The *modulus* of $z = a + ib$ is $|z| = \sqrt{z\bar{z}} = \sqrt{a^2 + b^2}$. By plotting the complex number $a + ib$ in the plane with horizontal coordinate a and vertical coordinate b, one can easily see by the Pythagorean Theorem that the modulus of $a + ib$ is the length of the line extending from $(0, 0)$ to (a, b).

For our purposes, probably the most important fact about complex numbers is that they can be represented in polar coordinates and using "complex exponentials." Indeed, a complex number $z = (a, b)$ can be specified by giving, instead of a and b, the modulus $|z|$ and the angle θ between the positive real axis and the line extending from $(0, 0)$ to (a, b) in the plane. It follows from basic trigonometry[7] that $a = |z| \cos\theta$ and $b = |z| \sin\theta$, so we see that a complex number can be written using the following "trigonometric representation:"

$$z = a + ib = |z| \cos\theta + i|z| \sin\theta = |z|(\cos\theta + i \sin\theta). \tag{A.1}$$

An important identity related to the trigonometric representation (A.1) is *Euler's Formula*:

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

This provides yet another way to represent the complex number z, namely,

$$z = a + ib = |z|(\cos\theta + i \sin\theta) = |z|e^{i\theta}.$$

Notice in particular that

$$e^{2\pi i 0} = e^{2\pi i} = 1, \quad e^{2\pi i \frac{1}{4}} = i, \quad e^{2\pi i \frac{1}{2}} = -1, \quad e^{-2\pi i \frac{1}{4}} = -i.$$

If $z = a + ib$ and if $|z| = a^2 + b^2 = 1$, then z lies on the unit circle. A particularly important sets of complex numbers is $\{e^{2\pi i \frac{k}{n}} : k = 0, 1, \ldots, n - 1\}$, which consists of n points evenly spaced around the unit circle. This is called the set of *n-th roots of unity*. For example, there are six 6-th roots of unity. They are

$$\{1, e^{2\pi i \frac{1}{6}}, e^{2\pi i \frac{2}{6}}, e^{2\pi i \frac{3}{6}}, e^{2\pi i \frac{4}{6}}, e^{2\pi i \frac{5}{6}}\} = \{1, \omega, \omega^2, \omega^3, \omega^4, \omega^5\}, \quad \text{where } \omega = e^{2\pi i \frac{1}{6}}.$$

We conclude this brief introduction to complex numbers with an important formula for the sum of all the complex roots of unity.

Lemma A.1.

$$\sum_{j \in \underline{n}} e^{2\pi i \frac{kj}{n}} = \begin{cases} n, & k \equiv 0 \mod n, \\ 0, & \text{otherwise.} \end{cases} \tag{A.2}$$

[7] Recall the "SOH-CAH-TOA" identities from high school.

Proof. Suppose $k \equiv 0 \mod n$.[8] Then $e^{2\pi i \frac{kj}{n}} = e^0 = 1$ for all $j \in \underline{n}$, so $\sum_{j\in\underline{n}} e^{2\pi i \frac{kj}{n}} = \sum_{j\in\underline{n}} 1 = n$. Suppose, on the other hand, that k is not congruent to 0 modulo n, and consider the following derivation:

$$\sum_{j\in\underline{n}} e^{2\pi i \frac{kj}{n}} = \sum_{j\in\underline{n}} e^{2\pi i \frac{k(j-1+1)}{n}} \tag{A.3}$$

$$= \sum_{j\in\underline{n}} e^{2\pi i \frac{k}{n}} e^{2\pi i \frac{k(j-1)}{n}} \tag{A.4}$$

$$= e^{2\pi i \frac{k}{n}} \sum_{j\in\underline{n}} e^{2\pi i \frac{k(j-1)}{n}} \tag{A.5}$$

$$= e^{2\pi i \frac{k}{n}} \sum_{\ell\in\underline{n}} e^{2\pi i \frac{k\ell}{n}} \tag{A.6}$$

Equality (A.3) holds since adding and subtracting 1 changes nothing. Equality (A.4) holds by laws of exponents. Equality (A.5) holds because the factor $e^{2\pi i \frac{k}{n}}$ does not depend on the index of summation j. The last equality holds by the change of variable $\ell = j - 1$. (Notice that summing over all j in $\underline{n}$ is the same as summing over all $\ell = j-1$ for which $j \in \underline{n}$.) Finally, since we assumed k is not congruent to 0 modulo n, we have $e^{2\pi i \frac{k}{n}} \neq 1$, so the equality

$$\sum_{j\in\underline{n}} e^{2\pi i \frac{kj}{n}} = e^{2\pi i \frac{k}{n}} \sum_{\ell\in\underline{n}} e^{2\pi i \frac{k\ell}{n}}$$

holds if and only if $\sum_{j\in\underline{n}} e^{2\pi i \frac{kj}{n}} = 0$. □

The proof reveals the important role played by the group structure of the index set $\underline{n}$. Evidently, the formula (A.2) is true because of the way modular arithmetic works. More generally, the formula holds when we replace $\underline{n}$ with any group G, since the set $\{x : x \in G\}$ of all elements of the group is no different from the set $\{x - y : x \in G\}$ of all "translated" group elements. Therefore, summing over all elements $x \in G$ is the same as summing over all elements $x - y$ for $x, y \in G$. (To take a concrete example, the set $\mathbb{Z}$ is the same as the set $\{n - 1 : n \in \mathbb{Z}\}$.)

References

[1] R. Tolimieri and M. An, "Group filters and image processing," in *Computational noncommutative algebra and applications*, ser. NATO Sci. Ser. II Math. Phys. Chem. Vol. 136, Kluwer Acad. Publ., Dordrecht, 2004, pp. 255–308. DOI: 10.1007/1-4020-2307-3_10. [Online]. Available: http://dx.doi.org/10.1007/1-4020-2307-3_10.

[8] Recall that $k \equiv j \mod n$ means $k - j$ is a multiple of n, that is, $k - j \in \{nd : d \in \mathbb{Z}\}$. Therefore, $k \equiv 0 \mod n$ simply means that k is a multiple of n, say, dn, so $e^{2\pi i \frac{kj}{n}} = e^{2\pi i \frac{dnj}{n}} = e^{2\pi i dj} = e^{2\pi i} = 1$. See appendix Section A.