Goals

- Fourier Series.
- Discrete Fourier Transforms.
Complex Numbers.

\( i = \sqrt{-1} \)

\( z = x + iy \)
- \( x \) is the real part of \( z \) and \( y \) is the imaginary part of \( z \)
- \( x = \text{Re}(z) \), \( y = \text{Im}(z) \)

\( \bar{z} = x - iy \) is the complex conjugate of \( z \)

\( |z| = \sqrt{x^2 + y^2} \)

\( e^{i\theta} = \cos \theta + i \sin \theta \) Euler’s formula

\( z = x + iy = |z|e^{i\theta} \) polar representation of a complex number
Fourier Series.

Let $f$ be $2\pi$ periodic functions. The Fourier series of a function $f$ is the series

$$f(x) \sim \sum_{n=-\infty}^{\infty} c_n e^{inx},$$

where

$$c_n = \frac{1}{2\pi} \int_{0}^{2\pi} f(x) e^{-inx} dx.$$

are called Fourier coefficients. Notice that the series may not converge or converge to a different functions. That is why we use $\sim$ sign instead of the equal sign.
Approximating Fourier Coefficients.

To approximate Fourier coefficients $c_k$, $k = 0, 1, \ldots$, We sample $f$ at $N = 2n + 1$ points

$$x_j = j \frac{2\pi}{N}, \quad j = 0, \ldots, N - 1.$$ 

Put

$$f_j = f(x_j), \quad j = 0, \ldots, N - 1.$$ 

Since $f$ is $2\pi$ periodic, we have $f_N = f_0$. Applying the composite trapezoidal rule to approximate $c_k$, we obtain

$$c_k = \frac{1}{2\pi} \int_0^{2\pi} f(x)e^{-ikx} \, dx \approx \frac{1}{N} \sum_{j=0}^{N-1} f_j e^{-\frac{ijk2\pi}{N}}, \quad k = -n, \ldots, 0, \ldots, n.$$
Properties of the Approximating Fourier Coefficients.

For $k = 1, \ldots, n$ the approximate Fourier coefficients satisfy

$$c_{-k} = \frac{1}{N} \sum_{j=0}^{N-1} f_j e^{-\frac{i j (-k) 2\pi}{N}} = \frac{1}{N} \sum_{j=0}^{N-1} f_j e^{\frac{i j 2\pi}{N}} = \frac{1}{N} \sum_{j=0}^{N-1} e^{-\frac{i j k 2\pi}{N}} = \overline{c_k}$$

$$c_{-k} = \frac{1}{N} \sum_{j=0}^{N-1} f_j e^{\frac{i j k 2\pi}{N}} e^{-i j k 2\pi} = \frac{1}{N} \sum_{j=0}^{N-1} f_j e^{\frac{i j (N-k) 2\pi}{N}} = \overline{c_{N-k}}$$
Discrete Fourier Transform.

Given \( N \) real numbers \( f_0, \ldots, f_{N-1} \) we want to compute

\[
c_k = \frac{1}{N} \sum_{j=0}^{N-1} f_j e^{-\frac{ijk2\pi}{N}}, \quad k = 0, \ldots, N - 1.
\]

The map

\[
f_0, \ldots, f_{N-1} \longrightarrow c_k = \frac{1}{N} \sum_{j=0}^{N-1} f_j e^{-\frac{ijk2\pi}{N}}, \quad k = 0, \ldots, N - 1,
\]

is called the Discrete Fourier Transform (DFT).
Matrix Representation of the Discrete Fourier Transform.

Let $\omega_N = e^{i2\pi/N} \in \mathbb{C}$ be the $N$th root of unity. Thus we can rewrite

$$c_k = \frac{1}{N} \sum_{j=0}^{N-1} f_j e^{-\frac{ijk2\pi}{N}} = \frac{1}{N} \sum_{j=0}^{N-1} f_j \omega_N^{-jk}$$

$$= \frac{1}{N} \left(1, \omega_N^{-k}, \omega_N^{-2k}, \ldots, \omega_N^{-(N-1)k}\right) \left(\begin{array}{c} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_{N-1} \end{array}\right),$$

$k = 0, \ldots, N - 1.$
Matrix Representation of the Discrete Fourier Transform.

We can rewrite it using matrix representation

\[
\begin{pmatrix}
  c_0 \\
  c_1 \\
  c_2 \\
  \vdots \\
  c_{N-1}
\end{pmatrix} = \frac{1}{N} \begin{pmatrix}
  1 & 1 & 1 & \ldots & 1 \\
  1 & \omega_N^{-1} & \omega_N^{-2} & \ldots & \omega_N^{-(N-1)} \\
  1 & \omega_N^{-2} & \omega_N^{-4} & \ldots & \omega_N^{-2(N-1)} \\
  \vdots & \vdots & \vdots & \ddots & \vdots \\
  1 & \omega_N^{-(N-1)} & \omega_N^{-2(N-1)} & \ldots & \omega_N^{-(N-1)^2}
\end{pmatrix} \begin{pmatrix}
  f_0 \\
  f_1 \\
  f_2 \\
  \vdots \\
  f_{N-1}
\end{pmatrix}
\]
Matrix Representation of the Discrete Fourier Transform.

The matrix

\[
F_N = \begin{pmatrix}
1 & 1 & 1 & \ldots & 1 \\
1 & \omega_N^{-1} & \omega_N^{-2} & \ldots & \omega_N^{-(N-1)} \\
1 & \omega_N^{-2} & \omega_N^{-4} & \ldots & \omega_N^{-2(N-1)} \\
\ldots & \ldots & \ldots & \ldots & \ldots \\
1 & \omega_N^{-(N-1)} & \omega_N^{-2(N-1)} & \ldots & \omega_N^{-(N-1)^2}
\end{pmatrix}
\]

is called the Fourier matrix. Since

\[
\omega_N^{-l} = e^{-il2\pi/N} = e^{il2\pi/N} = \omega_N^l,
\]

the matrix \(F_N\) is equal to its conjugate \(F_N^\dagger\).
Matrix Representation of the Discrete Fourier Transform.

We can rewrite the Discrete Fourier Transform as

\[
\begin{pmatrix}
c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{N-1}
\end{pmatrix} = \frac{1}{N} \overline{F_N} \begin{pmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_{N-1} \end{pmatrix}.
\]

Thus the application of the DFT to a vector \( f \) is a matrix-vector multiplication

\[
c = \frac{1}{N} \overline{F_N} f.
\]
The Fourier Matrix

Lemma

The Fourier matrix $F_N$ satisfies $F_N F_N = NI$, where $I$ is the identity matrix, i.e.

$$F_N^{-1} = \frac{1}{N} F_N, \quad (F_N)^{-1} = \frac{1}{N} F_N.$$ 

Proof.

For $k, l \in \{0, \ldots, N - 1\}$,

$$(F_N F_N)_{kl} = \sum_{j=0}^{N-1} \omega_N^{-lj} \omega_N^{kj} = \sum_{j=0}^{N-1} e^{-ijl2\pi/N} e^{ijk2\pi/N}$$

$$= \sum_{j=0}^{N-1} e^{ij(k-l)2\pi/N} = \sum_{j=0}^{N-1} \left(e^{i(k-l)2\pi/N}\right)^j$$

$$= \begin{cases} N, & \text{if } k = l; \\ \frac{1-\left(e^{i(k-l)2\pi/N}\right)^N}{1-e^{i(k-l)2\pi/N}} = 0, & \text{if } k \neq l. \end{cases}$$
The inverse Discrete Fourier Transform maps $c$ into $f$ and is given by

$$
\begin{pmatrix}
  f_0 \\
  f_1 \\
  f_2 \\
  \vdots \\
  f_{N-1}
\end{pmatrix}
= F_N
\begin{pmatrix}
  c_0 \\
  c_1 \\
  c_2 \\
  \vdots \\
  c_{N-1}
\end{pmatrix},
$$

or

$$
f_k = \sum_{j=0}^{N-1} c_j e^{i j k 2\pi / N}, \quad k = 0, \ldots, N - 1.
$$
Fast Fourier Transform (FFT).

- Straight forward implementation of matrix-vector products $\bar{F}_N c$ and $\frac{1}{N} F_N f$ require $O(N^2)$ operations.
- Using the special structure of $F_N$, these multiplications can be done more efficiently in $O(N \log N)$ operations using so-called Fast Fourier Transform (FFT).
- Matlab’s `fft(f)` uses FFT to compute $\bar{F}_N f$
- Matlab’s `ifft(c)` uses FFT to compute $\frac{1}{N} F_N c$

Note: there is a difference in scale by $N$: `fft(f)` computes $\bar{F}_N f$, not $\frac{1}{N} \bar{F}_N f$ and `ifft(c)` computes $\frac{1}{N} F_N c$, not $F_N c$. 