23MAT112 End-Sem Project Discrete Fourier Analysis And The Fast Fourier Transform

Group 2

Amrita Vishwa Vidyapeetham

23rd May 2024

Overview

The Motivation

The Fourier Transform is an algorithm that has made many parts of modern-day life possible.

Sampling

For n samples, x_j is one such sample point between the interval $[a, b]$, such that,

$$x_j = a + jh, \ j = 0, \ldots, n, \qquad \text{where } h = \frac{b-a}{n}$$

Sampling

For n samples, x_j is one such sample point between the interval $[a, b]$, such that,

$$x_j = a + jh, \ j = 0, \ldots, n, \qquad \text{where } h = \frac{b-a}{n}$$

We will use the "standard" interval $[0, 2\pi]$ which means,

$$x_0 = 0, x_1 = \frac{2\pi}{n}, x_2 = \frac{4\pi}{n}, x_j = \frac{2j\pi}{n}, x_{n-1} = \frac{2(n-1)\pi}{n}$$

Sampling

For n samples, x_j is one such sample point between the interval $[a, b]$, such that,

$$x_j = a + jh, \; j = 0, \ldots, n, \qquad \text{where } h = \frac{b-a}{n}$$

We will use the "standard" interval $[0, 2\pi]$ which means,

$$x_0 = 0, x_1 = \frac{2\pi}{n}, x_2 = \frac{4\pi}{n}, x_j = \frac{2j\pi}{n}, x_{n-1} = \frac{2(n-1)\pi}{n}$$

Remark

Functions defined on other intervals can simply be rescaled to fit the interval $[0, 2\pi]$

An Important Consequence

We will now write the sampled output at a given sample x_j as f_j, in other words $f_j = f(x_j)$.
Take the function,

$$f(x) = e^{inx}$$

Let's take n equally spaced samples, $x_j = \frac{2\pi j}{n}$.

$$f_j = f(\frac{2j\pi}{n})$$

An Important Consequence

We will now write the sampled output at a given sample x_j as f_j, in other words $f_j = f(x_j)$.
Take the function,

$$f(x) = e^{inx}$$

Let's take n equally spaced samples, $x_j = \frac{2\pi j}{n}$.

$$f_j = f(\frac{2j\pi}{n}) = exp(in\frac{2j\pi}{n})$$

An Important Consequence

We will now write the sampled output at a given sample x_j as f_j, in other words $f_j = f(x_j)$.
Take the function,

$$f(x) = e^{inx}$$

Let's take n equally spaced samples, $x_j = \frac{2\pi j}{n}$.

$$f_j = f(\frac{2j\pi}{n}) = exp(in\frac{2j\pi}{n}) = e^{2j\pi i}$$

An Important Consequence

We will now write the sampled output at a given sample x_j as f_j, in other words $f_j = f(x_j)$.
Take the function,

$$f(x) = e^{inx}$$

Let's take n equally spaced samples, $x_j = \frac{2\pi j}{n}$.

$$f_j = f(\frac{2j\pi}{n}) = exp(in\frac{2j\pi}{n}) = e^{2j\pi i} = 1$$

For all values of x_j, $f_j = 1$

An Important Consequence

We will now write the sampled output at a given sample x_j as f_j, in other words $f_j = f(x_j)$.
Take the function,

$$f(x) = e^{inx}$$

Let's take n equally spaced samples, $x_j = \frac{2\pi j}{n}$.

$$f_j = f(\frac{2j\pi}{n}) = exp(in\frac{2j\pi}{n}) = e^{2j\pi i} = 1$$

For all values of x_j, $f_j = 1$
This means that taking n equally spaced samples cannot give back the periodic function of frequency n

An Important Consequence

For n equally spaced samples,

$$e^{i(k+n)x} = e^{ikx}$$

An Important Consequence

For n equally spaced samples,

$$e^{i(k+n)x} = e^{ikx}$$

And more importantly, we can convert negative frequencies to positive frequencies,

$$e^{-ikx} = e^{i(n-k)x}$$

An Important Consequence

For n equally spaced samples,

$$e^{i(k+n)x} = e^{ikx}$$

And more importantly, we can convert negative frequencies to positive frequencies,

$$e^{-ikx} = e^{i(n-k)x}$$

For n samples, the discrete Fourier representation only needs n complex exponentials.(More on that later.)

Discrete Fourier Representation

For n samples, the Fourier representation is,

$$f(x) \sim p(x) = c_0 + c_1 e^{ix} + c_2 e^{2ix} + \cdots + c_{n-1} e^{(n-1)ix}$$

Discrete Fourier Representation

For n samples, the Fourier representation is,

$$f(x) \sim p(x) = c_0 + c_1 e^{ix} + c_2 e^{2ix} + \cdots + c_{n-1} e^{(n-1)ix}$$

Note

If $f(x)$ is real then $p(x)$ is real, at the sampled points, but the function could be complex in between. The imaginary component of the function is removed and $p(x)$ is treated as the interpolating trigonometric polynomial of the function $f(x)$. But, the representation is still retained for convenience.

Complex Vectors

For f_0,

$$p(x_0) = c_0 + c_1 e^{ix_0} + c_2 e^{2ix_0} + \cdots + c_{n-1} e^{(n-1)ix_0}$$

Complex Vectors

For f_0,

$$p(x_0) = c_0 + c_1 e^{ix_0} + c_2 e^{2ix_0} + \cdots + c_{n-1} e^{(n-1)ix_0}$$

For f_j,

$$p(x_j) = c_0 + c_1 e^{ix_j} + c_2 e^{2ix_j} + \cdots + c_{n-1} e^{(n-1)ix_j}$$

Complex Vectors

For f_0,

$$p(x_0) = c_0 + c_1 e^{ix_0} + c_2 e^{2ix_0} + \cdots + c_{n-1} e^{(n-1)ix_0}$$

For f_j,

$$p(x_j) = c_0 + c_1 e^{ix_j} + c_2 e^{2ix_j} + \cdots + c_{n-1} e^{(n-1)ix_j}$$

We define a complex vector $\vec{\omega}$ to find all values of f_j,

$$\vec{\omega_k} = [e^{ikx_0}, e^{ikx_1}, \cdots, e^{ikx_{n-1}}]^T$$

$$\vec{\omega_k} = [1, e^{ik\frac{2\pi}{n}}, e^{ik\frac{4\pi}{n}}, \cdots, e^{ik\frac{2\pi(n-1)}{n}}]^T$$

Complex Vectors

For f_0,

$$p(x_0) = c_0 + c_1 e^{ix_0} + c_2 e^{2ix_0} + \cdots + c_{n-1} e^{(n-1)ix_0}$$

For f_j,

$$p(x_j) = c_0 + c_1 e^{ix_j} + c_2 e^{2ix_j} + \cdots + c_{n-1} e^{(n-1)ix_j}$$

We define a complex vector $\vec{\omega}$ to find all values of f_j,

$$\vec{\omega_k} = [e^{ikx_0}, e^{ikx_1}, \cdots, e^{ikx_{n-1}}]^T$$
$$\vec{\omega_k} = [1, e^{ik\frac{2\pi}{n}}, e^{ik\frac{4\pi}{n}}, \cdots, e^{ik\frac{2\pi(n-1)}{n}}]^T$$

Now we can write for the sample vector $\vec{f}$ with components f_j

$$\vec{f} = c_0\vec{\omega_0} + c_1\vec{\omega_1} + \cdots + c_{n-1}\omega_{n-1}$$

Inner Product For Vectors in $\mathbb{C}^n$

$$\langle f, g \rangle = \frac{1}{n} \sum_{j=0}^{n-1} f_j \overline{g_j} = \frac{1}{n} f(x_j)\overline{g(x_j)}$$

Where $\overline{g(x_j)}$ is the complex conjugate of $g(x_j)$

Remark

This is a rescaled version of the standard Hermitian dot product between complex vectors

Theorem

The sampled exponential vectors $\omega_0, \cdots, \omega_{n-1}$ form an orthogonal basis in $\mathbb{C}^n$ with respect to the inner product,

$$\langle f, g \rangle = \frac{1}{n} \sum_{j=0}^{n-1} f_j \overline{g_j}$$

To prove this, an understanding of nth roots of unity is required.

Nth Roots Of Unity

Definition

A number $z \in \mathbb{C}$ satisfying the equation, where $n \in Z^+$,

$$z^n = 1$$

Nth Roots Of Unity

Definition

A number $z \in \mathbb{C}$ satisfying the equation, where $n \in Z^+$,

$$z^n = 1$$

We define,

$$\zeta_n = e^{2\pi i/n}$$

Where,

$$\zeta_n^n = e^{(2\pi i/n)n} = 1$$

Nth Roots Of Unity

Definition

A number $z \in \mathbb{C}$ satisfying the equation, where $n \in Z^+$,

$$z^n = 1$$

We define,

$$\zeta_n = e^{2\pi i/n}$$

Where,

$$\zeta_n^n = e^{(2\pi i/n)n} = 1$$

Which means that $\zeta_n = \sqrt[n]{1}$.

Nth Roots Of Unity

Any k^{th} power of ζ_n is also an n^{th} root of unity

$$(z^k)^n = (z^n)^k = 1^k = 1$$

Nth Roots Of Unity

Any k^{th} power of ζ_n is also an n^{th} root of unity

$$(z^k)^n = (z^n)^k = 1^k = 1$$

So for the polynomial $z^n - 1$,

$$z^n - 1 = (z-1)(z-\zeta_n)(z-\zeta_n^2)\cdots(z-\zeta_n^{n-1})$$

Proving The Theorem

Take $\zeta_n^k = e^{2\pi ki/n}$, this means that $\zeta_n^n = 1$, and there are n equally spaced such complex numbers, between ζ_1 and ζ_n.

$$z^n - 1 = (z-1)(z-\zeta_n)(z-\zeta_n^2)\cdots(z-\zeta_n^{n-1})$$
$$z^n - 1 = (z-1)(1+z+z^2+\cdots+z^{n-1})$$

Proving The Theorem

Take $\zeta_n^k = e^{2\pi ki/n}$, this means that $\zeta_n^n = 1$, and there are n equally spaced such complex numbers, between ζ_1 and ζ_n.

$$z^n - 1 = (z-1)(z-\zeta_n)(z-\zeta_n^2)\cdots(z-\zeta_n^{n-1})$$
$$z^n - 1 = (z-1)(1+z+z^2+\cdots+z^{n-1})$$

We equate the two to get,

$$(z-1)(1+z+\cdots+z^{n-1}) = (z-1)(z-\zeta_n)\cdots(z-\zeta_n^{n-1})$$
$$1+z+\cdots+z^{n-1} = (z-\zeta_n)\cdots(z-\zeta_n^{n-1})$$
$$1+\zeta_n^k+\zeta_n^{2k}+\cdots+\zeta_n^{(n-1)k} = \{n, k=0 \text{ and } 0, 0<k<n\}$$

Proving The Theorem

Take $\zeta_n^k = e^{2\pi ki/n}$, this means that $\zeta_n^n = 1$, and there are n equally spaced such complex numbers, between ζ_1 and ζ_n.

$$z^n - 1 = (z-1)(z-\zeta_n)(z-\zeta_n^2)\cdots(z-\zeta_n^{n-1})$$
$$z^n - 1 = (z-1)(1+z+z^2+\cdots+z^{n-1})$$

We equate the two to get,

$$(z-1)(1+z+\cdots+z^{n-1}) = (z-1)(z-\zeta_n)\cdots(z-\zeta_n^{n-1})$$
$$1+z+\cdots+z^{n-1} = (z-\zeta_n)\cdots(z-\zeta_n^{n-1})$$
$$1+\zeta_n^k+\zeta_n^{2k}+\cdots+\zeta_n^{(n-1)k} = \{n, k=0 \text{ and } 0, 0<k<n\}$$

This extends to all integers k, If k is a multiple of n, then the sum gives n, while giving 0 otherwise.

Proving The Theorem

Writing the sampled exponential vectors in terms of the n^{th} roots of unity,

$$\omega_k = (1, \zeta_n^k, \zeta_n^{2k}, \zeta_n^{3k}, \cdots, \zeta_n^{(n-1)k})^T$$

Proving The Theorem

Writing the sampled exponential vectors in terms of the n^{th} roots of unity,

$$\omega_k = (1, \zeta_n^k, \zeta_n^{2k}, \zeta_n^{3k}, \cdots, \zeta_n^{(n-1)k})^T$$

We get,

$$\begin{aligned}
\langle \omega_k, \omega_l \rangle &= \frac{1}{n} \sum_{j=0}^{n-1} \zeta_n^{jk} \overline{\zeta_n^{jl}} \\
&= \frac{1}{n} \sum_{j=0}^{n-1} \zeta_n^{j(k-l)} = \{1, k = l \text{ and } 0, k \neq l\}
\end{aligned}$$

Refresher On Orthogonal Bases

The properties of an orthonormal basis allows us to isolate any given component v_i of the vector $\vec{v}$ by simply performing the inner product of $\vec{q_i}$ with the vector $\vec{v}$ ($\vec{q_i} \cdot \vec{q_j} = 0, \vec{q_i} \cdot \vec{q_i} = 1$)

$$\vec{q_i}^T \vec{v} = \vec{q_i}^T v_1 \vec{q_1} + \vec{q_i}^T v_2 \vec{q_2} + \cdots + \vec{q_i}^T v_i \vec{q_i} + \cdots + \vec{q_i}^T v_n \vec{q_n}$$

$$\vec{q_i}^T \vec{v} = v_i$$

Isolating Fourier Coefficients

We apply the inner product of $\vec{f}$ with ω_k to get the coefficient c_k

$$c_k = \langle f, \omega_k \rangle = \frac{1}{n} \sum_{j=0}^{n-1} f_j \overline{e^{ikx_j}}$$

Isolating Fourier Coefficients

We apply the inner product of $\vec{f}$ with ω_k to get the coefficient c_k

$$c_k = \langle f, \omega_k \rangle = \frac{1}{n} \sum_{j=0}^{n-1} f_j \overline{e^{ikx_j}} = \frac{1}{n} \sum_{j=0}^{n-1} \zeta_n^{-jk} f_j$$

Definition

The passage from a signal to its Fourier coefficients is known as The **Discrete Fourier Transform**

Definition

The reconstruction of a signal from its Fourier coefficients is known as the **Inverse Discrete Fourier Transform**

Matrix Form

For a given Fourier coefficient,

$$c_k = \frac{1}{n}\sum_{j=0}^{n-1} f_j \zeta_n^{-jk}$$

So we can construct a Vandermonde matrix F_n where a given term $a_{ij} = \zeta_n^{ij}$, where $i, j = 0, \cdots, n-1$

$$\begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{n-1} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \zeta_n & \zeta_n^2 & \cdots & \zeta_n^{n-1} \\ 1 & \zeta_n^2 & \zeta_n^4 & \cdots & \zeta_n^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \zeta_n^{n-1} & \zeta_n^{2(n-1)} & \cdots & \zeta_n^{(n-1)^2} \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_{n-1} \end{bmatrix}$$

Fast Fourier Transform

1. The Discrete Fourier Transform has a time complexity $O(n^2)$

Fast Fourier Transform

1. The Discrete Fourier Transform has a time complexity $O(n^2)$
2. James Cooley and John Tukey discovered a much more efficient method to compute the DFT, in the 1960s

Fast Fourier Transform

1. The Discrete Fourier Transform has a time complexity $O(n^2)$
2. James Cooley and John Tukey discovered a much more efficient method to compute the DFT, in the 1960s
3. This new algorithm has a time complexity $O(n \log n)$

The Fast Fourier Transform

The key idea is, for n samples, if n is even, There exists an m such that $n = 2m$,

- Split the signal into halves, one set of even samples and one set of odd samples each with m samples

The Fast Fourier Transform

The key idea is, for n samples, if n is even, There exists an m such that $n = 2m$,

- Split the signal into halves, one set of even samples and one set of odd samples each with m samples
- $\zeta_n^2 = \zeta_m$

Clearly, it is best for functions where n is be a power of 2.

The Fast Fourier Transform

The key idea is, for n samples, if n is even, There exists an m such that $n = 2m$,

- Split the signal into halves, one set of even samples and one set of odd samples each with m samples
- $\zeta_n^2 = \zeta_m$
- Split m into halves, one set of even samples and one set of odd samples
- Repeat until number of function samples cannot be halved.

Clearly, it is best for functions where n is be a power of 2.

Fast Fourier Transform

We rearrange the even and odd vectors to get,

$$\hat{f} = F_{2^n}\vec{f} = \begin{bmatrix} I_{2^{n-1}} & -D_{2^{n-1}} \\ I_{2^{n-1}} & -D_{2^{n-1}} \end{bmatrix} \begin{bmatrix} F_{2^{n-1}} & 0 \\ 0 & F_{2^{n-1}} \end{bmatrix} \begin{bmatrix} f_{even} \\ f_{odd} \end{bmatrix}$$

Where I is the identity matrix and D is,

$$D_{2^{n-1}} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & \zeta_{2^{n-1}} & 0 & \cdots & 0 \\ 0 & 0 & \zeta_{2^{n-1}}^2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \zeta_{2^{n-1}}^{2^{n-1}} \end{bmatrix}$$

The Explanation For Efficiency

- The first matrix consists of 4 diagonal matrices, which is not computation intensive.
- The Fourier matrices are broken down further and further until we reach the 2×2 form, if n is a power of 2

Properties

- Linearity
- Periodicity
- Time And Frequency Reversal
- Parseval's Theorem

Properties

- Shift Property
- Complex Conjugate Property
- Convolution Theorem
- Symmetry In The Signal

Applications

1. Image Compression
2. Audio Compression
3. Denoising signals

Image Compression

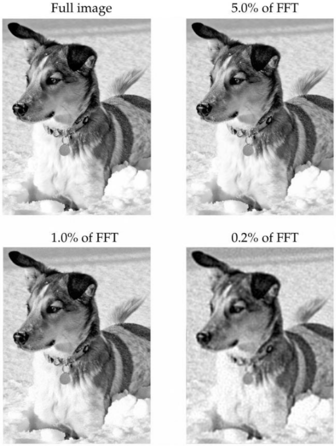

Figure: Compressed images

Audio Compression

- The FFT makes audio compression possible.
- Audio is sampled at 44.1KHz per second.
- The FFT performs significantly faster than the DFT.

Denoising A Signal

- Generating a signal

Denoising A Signal

- Generating a signal
- Adding random noise

Denoising A Signal

- Generating a signal
- Adding random noise
- Computing The Fast Fourier Transform

Denoising A Signal

- Generating a signal
- Adding random noise
- Computing The Fast Fourier Transform
- Finding The Power Spectrum Density

Denoising A Signal

- Generating a signal
- Adding random noise
- Computing The Fast Fourier Transform
- Finding The Power Spectrum Density
- Zero out smaller indices

Denoising A Signal

- Generating a signal
- Adding random noise
- Computing The Fast Fourier Transform
- Finding The Power Spectrum Density
- Zero out smaller indices
- Zero out corresponding Fourier coefficients

Denoising A Signal

- Generating a signal
- Adding random noise
- Computing The Fast Fourier Transform
- Finding The Power Spectrum Density
- Zero out smaller indices
- Zero out corresponding Fourier coefficients
- Plot And Compare Results

The Results

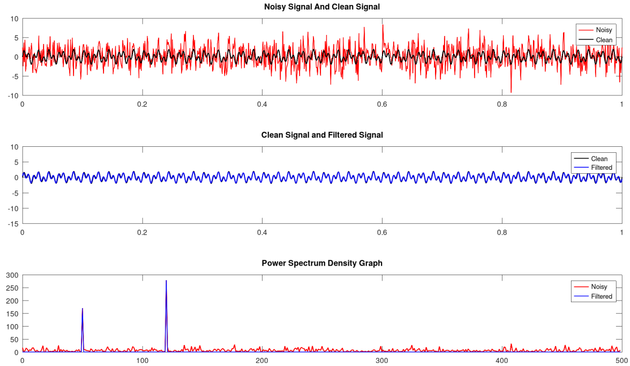

Figure: (From top to bottom): Plot of noisy signal over clean signal, clean signal over final filtered signal and the power spectrum density graph of noisy signal and filtered signal

Code Implementation Of Denoising

```
dt = .001;
t = 0:dt:1;
forig = sin(2*pi*50*t) + sin(2*pi*120*t);
f = forig + 2.5*randn(size(t));
n = length(t);
fhat = fft(f,n);
PSD = fhat.*conj(fhat)/n;
freq = 1/(dt*n)*(0:n);
L = 1:floor(n/2);
%% Use the PSD to filter out noise
indices = PSD>100;
PSDclean = PSD.*indices;
fhat = indices.*fhat;
ffilt = ifft(fhat);
```

Thank You

“Profound study of nature is the most fertile source of mathematical discoveries.” - Joseph Fourier. Noteworthy since, Joseph Fourier discovered the Fourier series while trying tos solve the heat equation