# Sensing Techniques and Data Analytics

## Chapter 4: Frequency Domain Analysis

Instructure:
**Mohammad Talebi-Kalaleh**

PhD Candidate of Structural Engineering
University of Alberta

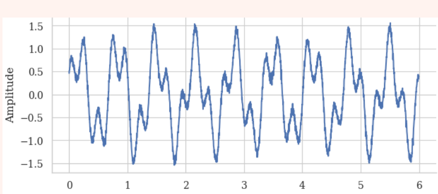
Fall 2025

1

## Why Frequency Domain?

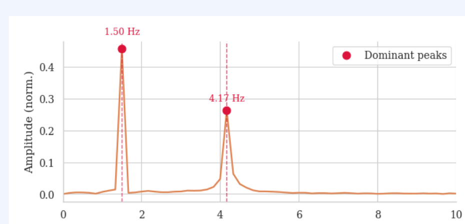
### Time Domain Limitations

- ❌ Mixed frequencies overlap
- ❌ Modal properties hidden
- ❌ Noise affects entire signal
- ❌ Damage signatures unclear




### Frequency Domain Benefits

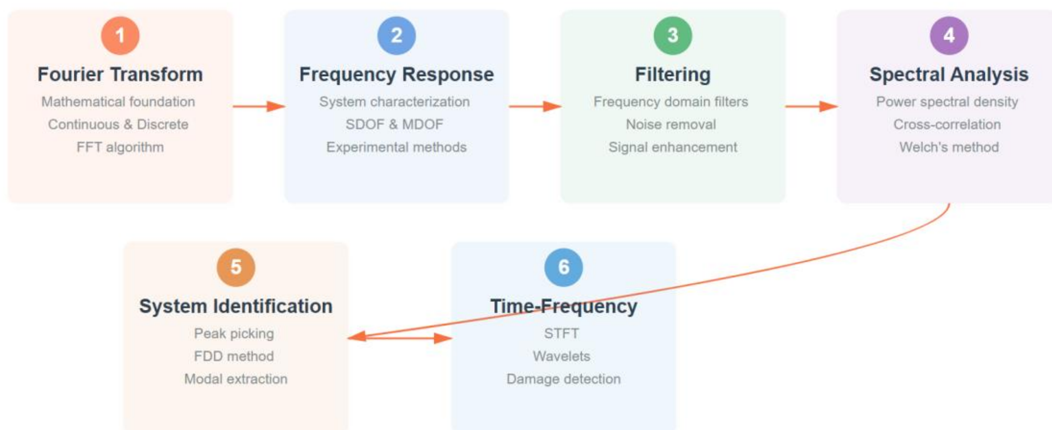
- ✓ Separate frequency components
- ✓ Clear modal identification
- ✓ Target specific frequencies
- ✓ Damage patterns visible






2

# Overview






# Inner Product: Measuring Signal Similarity

## Mathematical Definition

$$\boxed{\langle f, g\rangle = \frac{1}{T}\int f(t) \times g^*(t)dt}$$

- Measures total overlap between signals
  - Large value = high similarity
- Zero value = orthogonal (no similarity)

## Visual Example


## Why This Matters for Fourier Transform

1. Fourier Transform compares your signal with sine/cosine waves at all frequencies
2. High inner product = that frequency is strongly present in your signal
3. This reveals the frequency content of your structural response

**Foundation for understanding frequency decomposition**

# Fourier Transform: Decomposing Signals

## Any Signal = Sum of Sine and Cosine Waves

Fourier Transform finds the amplitude and phase of each frequency component

### Forward Transform

Time → Frequency

$$\boxed{X(f) = \frac{1}{2\pi}\int x(t)e^{-j2\pi ft}dt}$$

- $x(t)$ = time signal
- X(f) = frequency spectrum
- $e^{-j2\pi ft}$ = complex sinusoid

↕

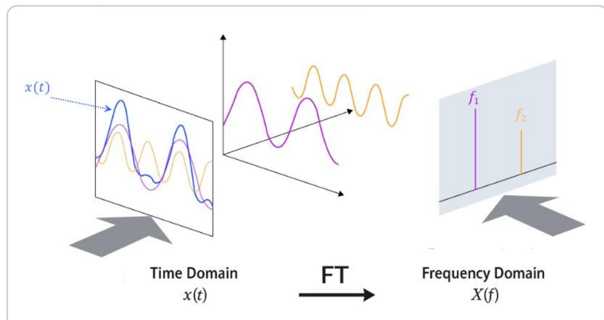
### Inverse Transform

Frequency → Time

$$\boxed{x(t) = \int X(f)e^{j2\pi ft}df}$$

- Sums all frequency components
- Rebuilds signal from spectrum
- Perfect reconstruction

$$e^{j2\pi ft} = \cos(2\pi ft) + j\sin(2\pi ft)$$

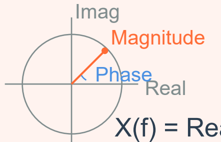







# Magnitude and Phase

### Fourier Transform Output



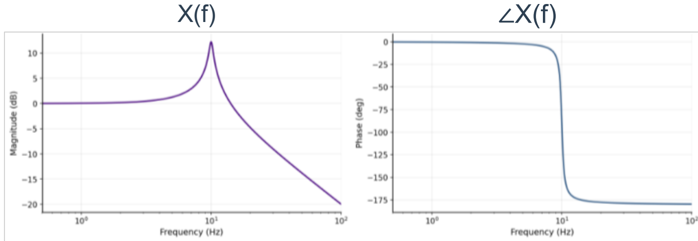

X(f) = Real + j×Imag
∠X(f) = arctan(Imag/Real)

### What They Tell Us

**Magnitude |X(f)|**
How much of each frequency is present

**Phase ∠X(f)**
When each frequency reaches its peak

X(f) ∠X(f)




$$|X(f)|_{dB} = 10\log_{10}(|X(f)|)$$

# Key Fourier Transform Properties

### 1 Linearity

$FT\{ax(t) + by(t)\} = aX(f) + bY(f)$

Combine signals → Combine their spectra

### 2 Time Shifting

$FT\{x(t - \tau)\} = X(f) \times e^{-j2\pi f\tau}$

Time delay → Phase shift in frequency

### 3 Convolution Theorem

$FT\{x(t) * h(t)\} = X(f) \times H(f)$

Convolution in time → Multiplication in frequency

### 4 Differentiation

$FT\{dx/dt\} = j2\pi f \times X(f)$

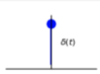
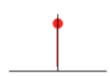
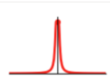
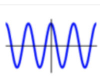
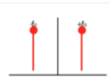
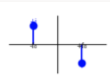
Derivative → Multiply by frequency

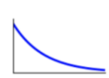
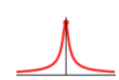
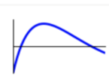
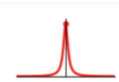
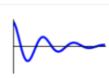
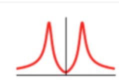
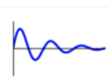
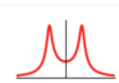
### Why These Properties Matter

- Design filters and analyze system responses efficiently
- Simplify complex calculations (convolution → multiplication)



# Common Fourier Transform Pairs

| Time Domain $x(t)$ | Time Plot | Frequency Domain $X(f)$ | Frequency Plot |
|---|---|---|---|
| $\delta(t)$ |  | $\frac{1}{2\pi}$ |  |
| 1 |  | $\frac{1}{2\pi}\delta(f)$ |  |
| $t$ |  | $\frac{j}{(2\pi f)^2}$ |  |
| $\cos(2\pi f_0 t)$ |  | $\frac{1}{4\pi}[\delta(f - f_0) + \delta(f + f_0)]$ |  |
| $\sin(2\pi f_0 t)$ |  | $\frac{j}{4\pi}[\delta(f - f_0) - \delta(f + f_0)]$ |  |

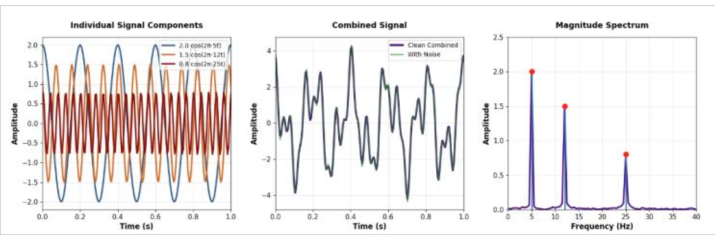
| Time Domain $x(t)$ | Time Plot | Frequency Domain $X(f)$ | Frequency Plot |
|---|---|---|---|
| $e^{-at}u(t)$ |  | $\frac{1}{2\pi(a+j2\pi f)}$ |  |
| $te^{-at}u(t)$ |  | $\frac{1}{2\pi(a+j2\pi f)^2}$ |  |
| $e^{-at}\cos(\omega_0 t)\,u(t)$ |  | $\frac{1}{4\pi}\left[\frac{1}{a+j2\pi(f-f_0)} + \frac{1}{a+j2\pi(f+f_0)}\right]$ |  |
| $e^{-at}\sin(\omega_0 t)\,u(t)$ |  | $\frac{1}{4\pi j}\left[\frac{1}{a+j2\pi(f-f_0)} - \frac{1}{a+j2\pi(f+f_0)}\right]$ |  |

# Storage Efficiency in Frequency Domain

**Example:**

x(t) = 2cos(2π×5t) + 1.5cos(2π×12t) + 0.8cos(2π×25t)

Three modes: 5 Hz, 12 Hz, 25 Hz




### Time Domain Storage

Sampling: 200 Hz
Duration: 10 seconds

**2000 data points**

Store every sample value

### Frequency Domain Storage

Frequencies: [5, 12, 25] Hz
Amplitudes: [2.0, 1.5, 0.8]
Phases: [0, 0, 0]

**9 values only!**

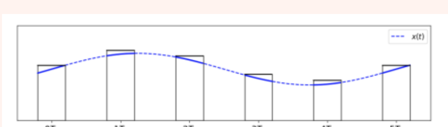
222× reduction



# From Continuous to Discrete

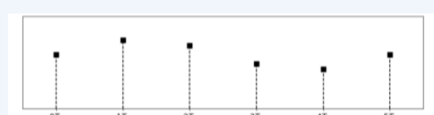
**Digital processing requires discrete samples**

### Continuous Signal



$x(t)$ - Infinite values

### Sampled Signal



$x[n]$ - Discrete samples at $T_s$ intervals

$$x_s(t) = \sum_{n=0}^{N-1} x[n]\,\delta(t - nT_s),$$

### Key Sampling Parameters

**Sampling Rate (fs)**
Samples per second

**Sampling Period (Ts)**
Ts = 1/fs

**N Samples**
Total data points

# Discrete Fourier Transform (DFT)

$$X_s(f) = \frac{1}{2\pi}\int_0^T x_s(t)e^{-j2\pi ft}dt = \frac{1}{N \times T_s}\sum_{n=0}^{N-1} x[n]e^{-j2\pi fnT_s} \times T_s$$

## DFT Mathematical Definition

$$\boxed{X[k] = \sum_{n=0}^{N-1} x[n]e^{-j2\pi kn/N}, \quad k = 0, 1, \ldots, N-1}$$

k = 0, 1, 2, ..., N-1 (frequency bins)

### Input

n: Sample index (time)

N: Number of samples

x[n]: Sampled signal

### Output

Complex values (mag + phase)

k: Frequency bin index

X[k]: Frequency spectrum

### Frequencies

Resolution: Fs/N Hz

Fs: Sampling frequency

f[k] = k × Fs/N

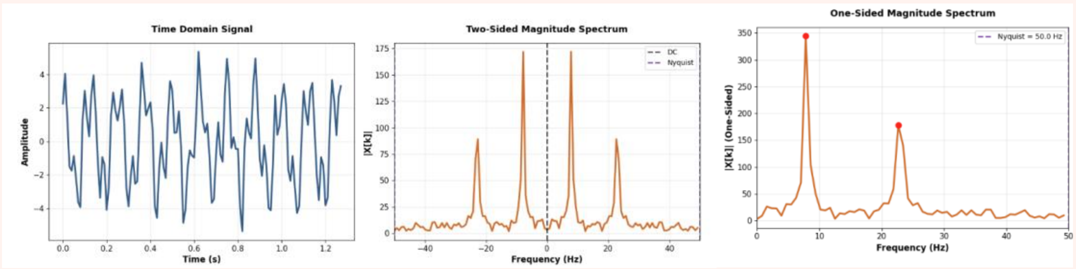


# Python FFT Implementation

```python
import numpy as np

# Forward FFT: Time domain → Frequency domain
X = np.fft.fft(x)

# Create frequency axis
freq = np.fft.fftfreq(len(x), 1/fs)

# Inverse FFT: Frequency domain → Time domain
x_reconstructed = np.fft.ifft(X)

# For real signals, use rfft for efficiency (positive frequencies only)
X_real = np.fft.rfft(x)
freq_real = np.fft.rfftfreq(len(x), 1/fs)

# Inverse of real FFT
x_from_rfft = np.fft.irfft(X_real)
```

# DFT Symmetry and Nyquist Frequency

### For Real Signals: Conjugate Symmetry




**Nyquist Frequency**

f_Nyquist = Fs/2

Maximum detectable frequency
Example: fs = 200 Hz → f_Nyquist = 100 Hz

**What This Means**

✓ Half the FFT output is useful
✓ Must sample > 2× highest frequency
✓ Only plot 0 to Fs/2
✓ Prevents aliasing



# Frequency Response Function (FRF)

### System Characterization in Frequency Domain

$$x(t) = g(t) * u(t) = \int_0^t g(t-\tau)u(\tau)d\tau \rightarrow X(f) = G(f) \cdot U(f)$$

FRF = Output / Input = $X(f) \ / \ U(f) \ = \ G(f)$

**Input U(f)** Force, excitation → **System G(f)** Structure/Bridge → **Output X(f)** Response

**FRF Reveals**
Natural frequencies, damping, mode shapes

**Independent of Input**
System property only

**FRF = Complete system description**