

The Efficient Discrete Fourier Transform

Zihe Liu zl559

Jerry Liu yhl63

Homerton College, University of Cambridge

October 12, 2026

Outline

1 Discrete Fourier Transform

- Theory
- Complexity

2 Fast Fourier Transform

- Theory
- Properties of W_N
- Derivation of FFT
- Algorithm

3 Experiments

4 Windowing

- Theory

5 Conclusion

Discrete Fourier Transform

Discrete Fourier Transform: Theory

The DFT converts a finite-length time-domain signal into its frequency-domain representation. For an input signal $x[n]$ of length N (with $0 \leq n \leq N-1$), the DFT is defined as:

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \quad 0 \leq k \leq N-1,$$

where the N th principal root of unity is

$$W_N = e^{-j\frac{2\pi}{N}}.$$

DFT as a Matrix Operation

We can rewrite the DFT as an inner product and then in vector form:

$$X[k] = \begin{bmatrix} 1 & W_N^k & W_N^{2k} & \dots & W_N^{(N-1)k} \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ \vdots \\ x[N-1] \end{bmatrix}.$$

Collating for all k , we have:

$$\mathbf{X} = \mathbf{W}\mathbf{x}, \quad \mathbf{W} = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & W_N & W_N^2 & \dots & W_N^{N-1} \\ 1 & W_N^2 & W_N^4 & \dots & W_N^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & W_N^{N-1} & W_N^{2(N-1)} & \dots & W_N^{(N-1)^2} \end{bmatrix}.$$

DFT Pseudocode

Algorithm DFT

```
1: function DFT( $\mathbf{x}$ )
2:    $N \leftarrow \text{length}(\mathbf{x})$ 
3:    $\mathbf{W} \leftarrow \text{zeros}(N, N)$                                 ▷ Initialize DFT matrix
4:   for  $k = 0$  to  $N - 1$  do
5:     for  $n = 0$  to  $N - 1$  do
6:        $\mathbf{W}[k, n] \leftarrow e^{-j2\pi kn/N}$ 
7:     end for
8:   end for
9:    $\mathbf{X} \leftarrow \mathbf{W} \cdot \mathbf{x}$                                 ▷ Matrix-vector multiplication
10:  return  $\mathbf{X}$ 
11: end function
```

Direct computation of $\mathbf{X}$ requires $(N - 1)^2$ complex multiplications and $N(N - 1)$ complex additions. Since multiplication is more compute-intensive than addition, the asymptotic complexity is dominated by multiplication, giving $O(N^2)$.

Fast Fourier Transform

Fast Fourier Transform: Theory

The radix-2 FFT algorithm improves efficiency by dividing the DFT into two DFTs of length $N/2$. For an even-length signal $x[n]$, we split it into even-indexed and odd-indexed components:

$$x_{\text{even}}[n] = x[2n], \quad x_{\text{odd}}[n] = x[2n + 1], \quad 0 \leq n \leq \frac{N}{2} - 1.$$

Their DFTs are combined using:

$$X[k] = X_{\text{even}}[k] + W_N^k X_{\text{odd}}[k], \quad 0 \leq k < \frac{N}{2},$$

$$X\left[k + \frac{N}{2}\right] = X_{\text{even}}[k] - W_N^k X_{\text{odd}}[k].$$

Preliminaries: Properties of W_N

Property 1:

$$W_N^2 = W_{N/2}$$

Proof:

$$W_N^2 = e^{-j\frac{2\pi}{N} \cdot 2} = e^{-j\frac{2\pi}{N/2}} = W_{N/2}.$$

More generally, $W_N^{2nk} = W_{N/2}^{nk}$.

Property 2:

$$W_N^{k+\frac{N}{2}} = -W_N^k.$$

Proof:

$$W_N^{k+\frac{N}{2}} = e^{-j\frac{2\pi}{N}(k+\frac{N}{2})} = e^{-j\frac{2\pi}{N}k} \cdot e^{-j\pi} = -W_N^k.$$

Derivation of FFT

Consider an N -point signal $x[n]$ (with N even). Split the signal into its even and odd indexed parts:

$$x_{\text{even}}[n] = x[2n], \quad x_{\text{odd}}[n] = x[2n + 1], \quad 0 \leq n \leq \frac{N}{2} - 1.$$

Splitting the sum of DFT:

$$X[k] = \sum_{n=0}^{N/2-1} x_{\text{even}}[n] W_N^{2nk} + \sum_{n=0}^{N/2-1} x_{\text{odd}}[n] W_N^{(2n+1)k}.$$

Since $W_N^{2nk} = W_{N/2}^{nk}$, define:

$$X_{\text{even}}[k] = \sum_{n=0}^{N/2-1} x_{\text{even}}[n] W_{N/2}^{nk}, \quad X_{\text{odd}}[k] = \sum_{n=0}^{N/2-1} x_{\text{odd}}[n] W_{N/2}^{nk}.$$

Thus,

$$X[k] = X_{\text{even}}[k] + W_N^k X_{\text{odd}}[k].$$

Derivation of FFT

To obtain the complete N -point DFT, note that the DFTs $X_{\text{even}}[k]$ and $X_{\text{odd}}[k]$ are periodic with period $\frac{N}{2}$:

$$X[k + \frac{N}{2}] = X_{\text{even}}[k] + W_N^{k + \frac{N}{2}} X_{\text{odd}}[k].$$

Using the earlier property $W_N^{k + \frac{N}{2}} = -W_N^k$, we have:

$$X[k + \frac{N}{2}] = X_{\text{even}}[k] - W_N^k X_{\text{odd}}[k].$$

Hence, the FFT is derived via:

$$X[k] = X_{\text{even}}[k] + W_N^k X_{\text{odd}}[k], \quad 0 \leq k < \frac{N}{2},$$

$$X\left[k + \frac{N}{2}\right] = X_{\text{even}}[k] - W_N^k X_{\text{odd}}[k].$$

Radix-2 DIT FFT Implementation

Algorithm FFT

```
1: function FFT( $\mathbf{x}$ )
2:    $N \leftarrow \text{length}(\mathbf{x})$ 
3:   if  $N = 1$  then
4:     return  $\mathbf{x}$                                 ▷ Base case: return the single value
5:   end if
6:    $\mathbf{x}_{\text{even}} \leftarrow \mathbf{x}[0 : 2 : N - 1]$         ▷ Extract even-indexed elements (half of  $\mathbf{x}$ )
7:    $\mathbf{x}_{\text{odd}} \leftarrow \mathbf{x}[1 : 2 : N - 1]$     ▷ Extract odd-indexed elements (other half of  $\mathbf{x}$ )
8:    $\mathbf{X}_{\text{even}} \leftarrow \text{FFT}(\mathbf{x}_{\text{even}})$         ▷ Recursive call on even elements
9:    $\mathbf{X}_{\text{odd}} \leftarrow \text{FFT}(\mathbf{x}_{\text{odd}})$         ▷ Recursive call on odd elements
10:   $\mathbf{X} \leftarrow \text{zeros}(N)$                     ▷ Initialize output vector
11:  for  $k = 0$  to  $N/2 - 1$  do
12:     $W_N^k \leftarrow e^{-j2\pi k/N}$                 ▷ Twiddle factor
13:     $\mathbf{X}[k] \leftarrow \mathbf{X}_{\text{even}}[k] + W_N^k \cdot \mathbf{X}_{\text{odd}}[k]$     ▷ First half of  $\mathbf{X}$ 
14:     $\mathbf{X}[k + N/2] \leftarrow \mathbf{X}_{\text{even}}[k] - W_N^k \cdot \mathbf{X}_{\text{odd}}[k]$   ▷ Second half of  $\mathbf{X}$ 
15:  end for
16:  return  $\mathbf{X}$ 
17: end function
```

Complexity Analysis

Let $A_c(N)$ and $M_c(N)$ denote the number of complex additions and multiplications for an N -point FFT. For $N = 2^k$, we have the recurrence relations:

$$\begin{aligned}A_c(N) &= 2A_c(N/2) + N \\M_c(N) &= 2M_c(N/2) + \frac{N}{2}\end{aligned}$$

For a 2-point DFT ($X[0] = x[0] + x[1]$, $X[1] = x[0] - x[1]$), we have $A_c(2) = 2$ and $M_c(2) = 0$. Solving the recurrences:

$$\begin{aligned}A_c(N) &= N \log_2 N \quad \text{complex additions} \\M_c(N) &= \frac{N}{2} \log_2 N - N + 1 \quad \text{complex multiplications}\end{aligned}$$

The overall computational complexity is $O(N \log N)$.

Experiments

Is our DFT any good?

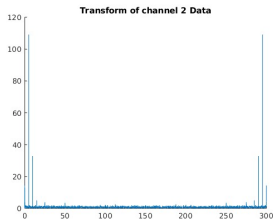


Figure: My DFT

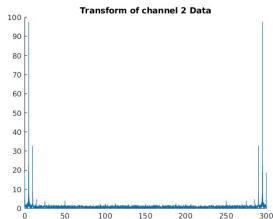


Figure: My FFT

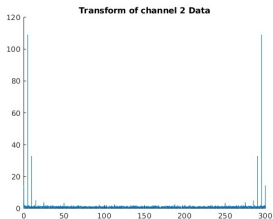


Figure: MATLAB FFT

Experimental Analysis: Complexity

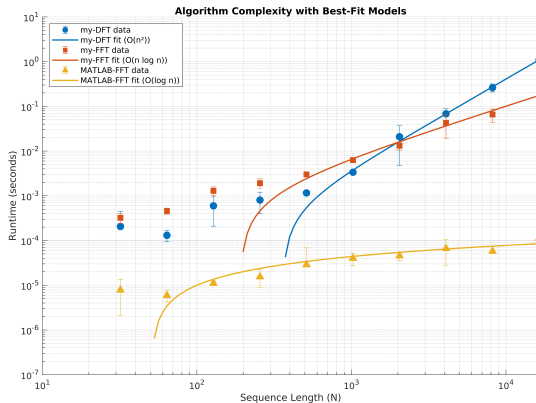


Figure: Execution time versus signal length N for various implementations of FFT

Log-log plots confirm that the FFT scales as $O(N \log N)$ while the DFT scales as $O(N^2)$.

Experimental Analysis: Numerical Error

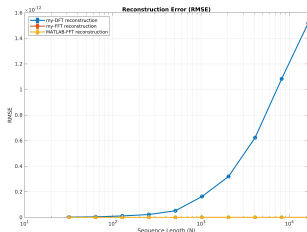


Figure: Reconstruction error versus signal length N for various implementations of FFT

- The reconstruction error was measured using the RMSE, $\sqrt{\frac{1}{N} \sum_{i=1}^N |x_i - \hat{x}_i|^2}$
- Although all errors are very low (around 10^{-12}), the custom DFT shows an exponentially growing error with increasing N .
- In contrast, both FFT implementations maintained minimal error.

Windowing

Why Windowing?

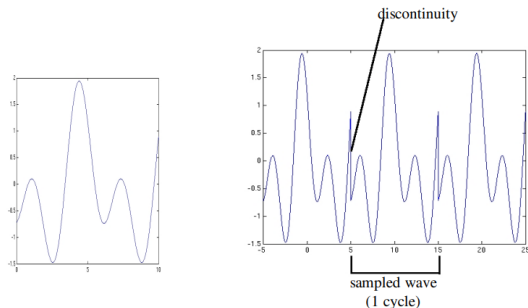


Figure: Discontinuities due to truncation

- When sampling continuous non-periodic signals, we are only able to sample for a finite time period, resulting in a truncated signal causing discontinuities.
- Discontinuities in the time domain become "ripples" in the frequency domain, known as spectral leakage.

Why Windowing?

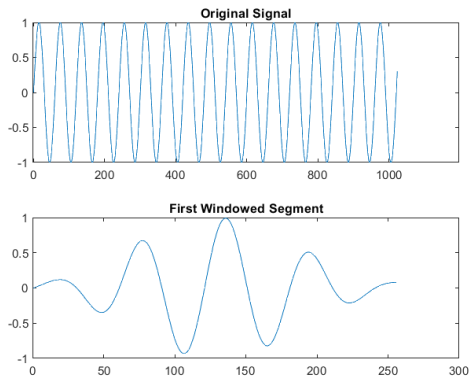


Figure: Hann Windowing on sinusoid segment

- A “window function” which smooths these edges, reducing unwanted artifacts and improving spectral estimation by reducing the values at the edge of the truncated sample.

Windowing Functions: Hann, Hamming, Blackman-Harris

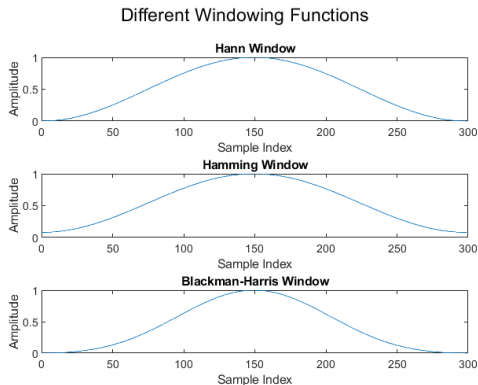


Figure: Discontinuities due to truncation

Windowing Functions: Hann, Hamming, Blackman-Harris

- **Hann Window:** Smoothens the edges using an inverted cosine function.

$$0.5 * (1 - \cos(2\pi n / (N - 1)))$$

- **Hamming Window:** Similar to Hann but retains a small gap at the edges for a narrower main lobe.

$$0.54 - 0.46\cos(2\pi n / (N - 1))$$

- **Blackman-Harris Window:** Stronger tapering for better side lobe suppression at the cost of frequency resolution.

$$0.42 - 0.5\cos(2\pi n / (N - 1)) + 0.8\cos(4\pi n / (N - 1))$$

Measurements and Results

- Implemented the 3 windowing functions in MATLAB.
- Applied to a 5Hz sinusoid to observe the impact on peak amplitudes and side lobe suppression.
- Results illustrate differences in resolution and leakage for each window.

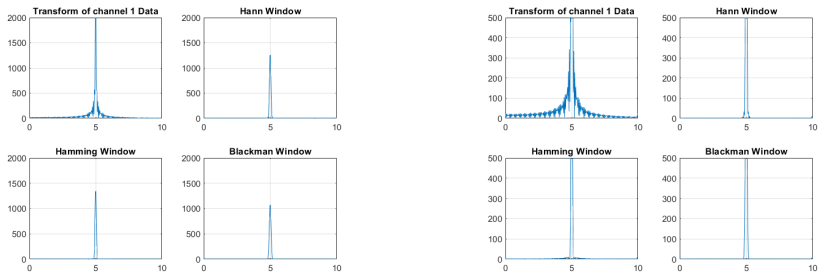


Figure: Effect of windowing on amplitude and side lobe suppression.

Measurements and Results

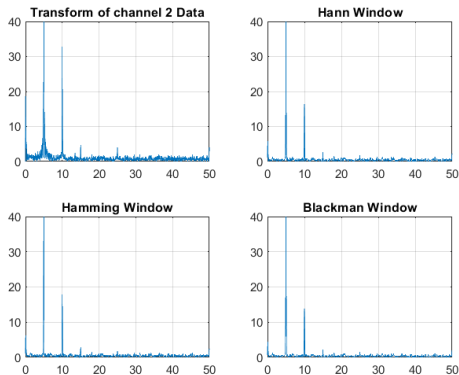


Figure: Windowing function applied to measured ch2 data

Measurements and Results

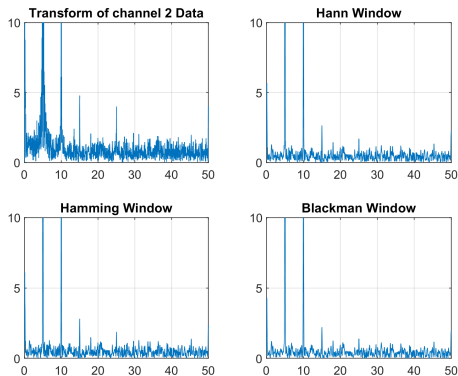


Figure: Windowing function applied to measured ch2 data

Discussion and Conclusions

- The choice of a window function depends on the nature of the signal.
- **Hann/Hamming Window:** Suitable for smooth, continuous signals requiring balanced performance.
- **Blackman Window:** Ideal for transient analysis with minimal leakage.

Application to Earthquake Data

- Earthquakes are characterized by sudden, short-duration energy releases caused by the rupture of faults. These seismic waves result in strong transients that propagate through the Earth's crust.
- Out of the 3 tested cases, the **Blackman-Harris Window** may be the most suitable as it offers the best side lobe suppression capabilities.
- Other dynamic functions such as **Kaiser Window** with adjustable beta can balance resolution and leakage suppression.

Conclusion

Conclusion

- The direct DFT scales as $O(N^2)$, which is computationally expensive for large data sets.
- The FFT reduces the complexity to $O(N \log N)$, providing significant performance gains.
- Experimental results confirm the expected scaling and show that FFT has superior numerical stability compared to the DFT.
- Selecting the appropriate window is crucial for accurate frequency analysis.
- The Hann or Hamming window is recommended for general applications.
- Earthquake data benefits from adaptive windows like Kaiser for transient preservation.