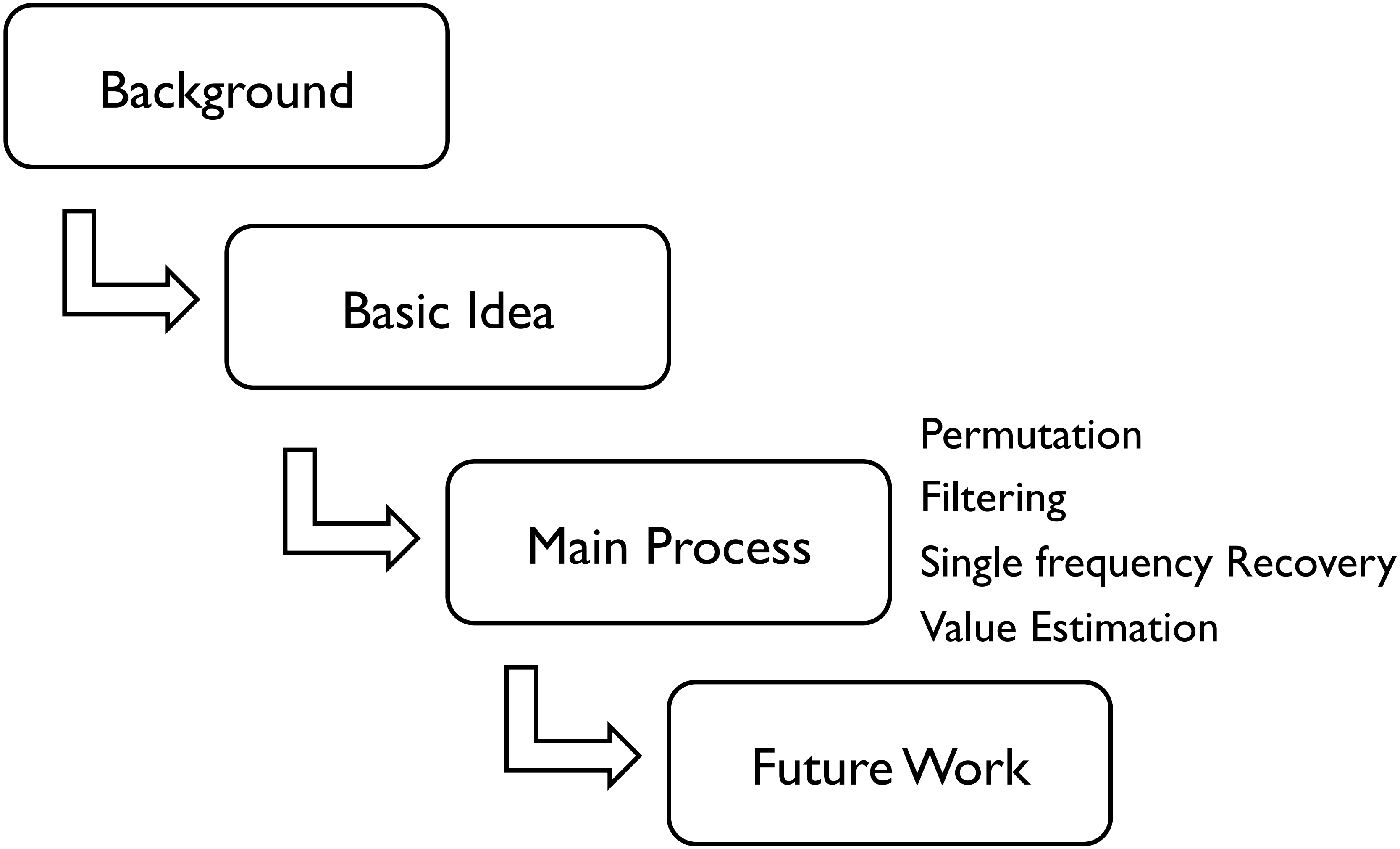
Sparse Fourier Transform

-- Simple Implement and Performance

ZhiXiong Yang
He Zhang
Xi Zhang

Instructor: A.Petropulu


Background
Basic Idea
Main Process
Permutation
Filtering
Single frequency Recovery
Value Estimation
Future Work

Background

-- Why need Sparse Fourier Transform ?

- Calculate DFT of a Signal
- The spectrum contains only a few dominating frequencies
- Don't want to waste time on the zeroes in the spectrum
- Very easy to calculate if the signal has only one frequency

Basic Idea

- Single frequency can be calculated fast

$$\frac{x[n+1]}{x[n]} = e^{j\frac{2\pi}{N}\omega}$$

- Several ways to search it. (Binary, CRT)
- What SFT does is to isolate each dominating frequency from the original signal and calculate each frequency separately

Process

- Permutation

  -- To separate close frequencies

- Filtering(Binning)

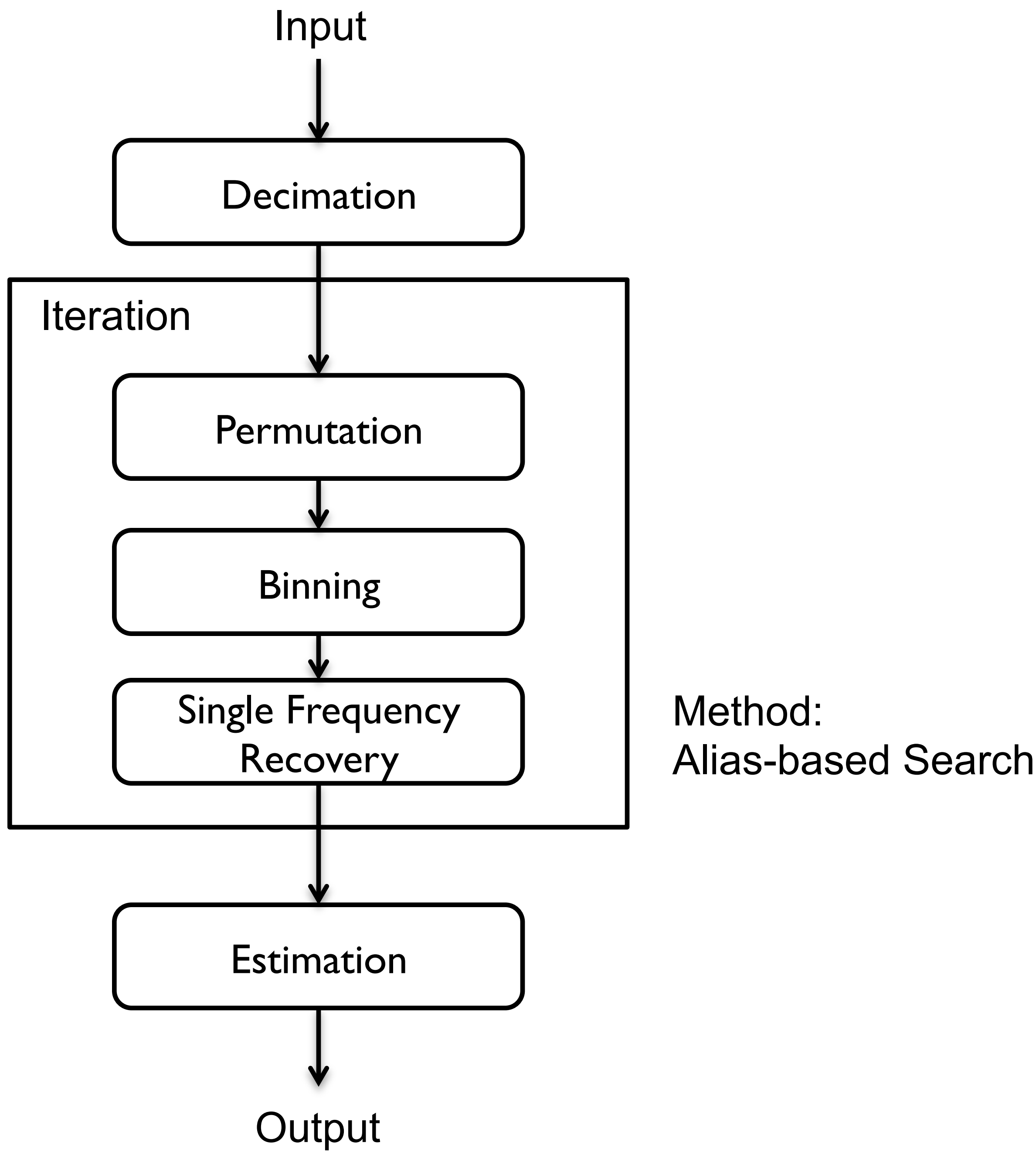
  -- To extract single frequency from the signal

- Single Frequency Recovery

  -- To locate the frequency

- Value Estimation

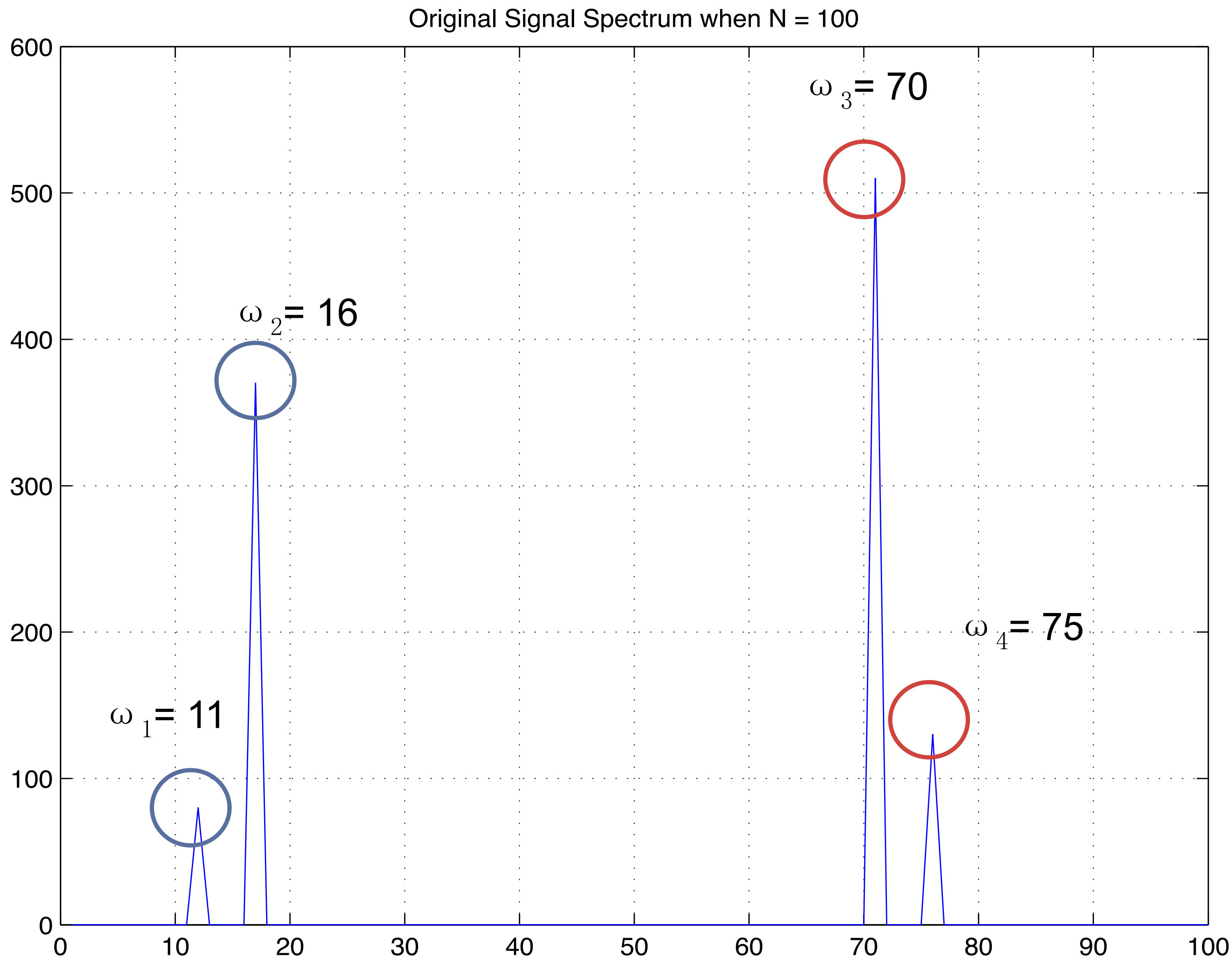
  -- To calculate the energy for each frequency

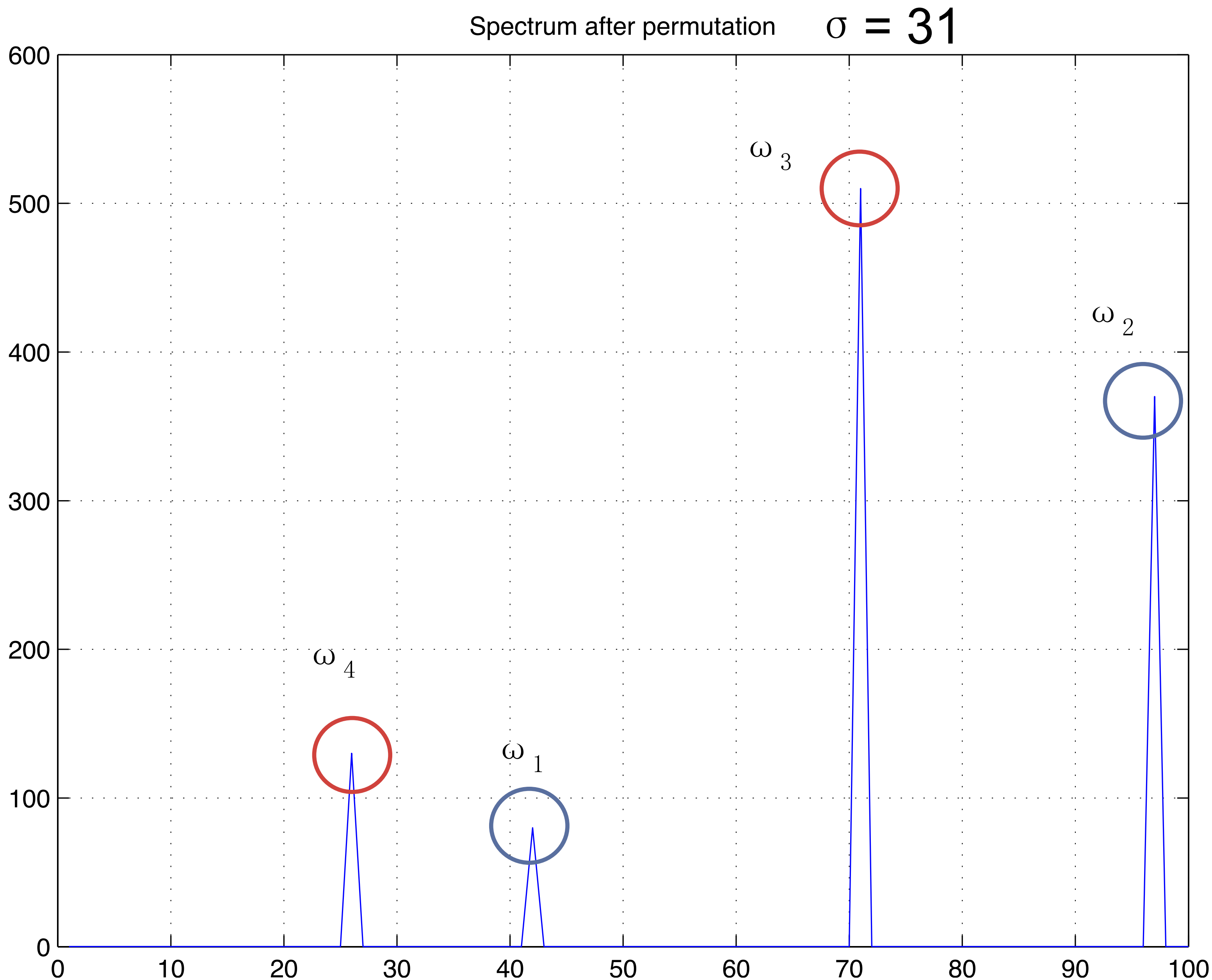

Input
Decimation
Iteration
Permutation
Binning
Single Frequency Recovery
Estimation
Output
Method:
Alias-based Search

Permutation

- Motivation
  - Two frequencies could be very close to each other.
- Method
  - By creating a random prime parameter we want to make the frequencies "randomly" distributed.
  - Repeat several times then every frequency should have its chance to be alone.
  - After binning and frequency search, ω can be derived by mapping back.

$$y(n) \rightarrow x(\sigma n) \Leftrightarrow \hat{Y}(\omega) \rightarrow \hat{X}(\sigma^{-1}\omega)$$


Original Signal Spectrum when N = 100
ω_3= 70
ω_2= 16
ω_4= 75
ω_1= 11
600
500
400
300
200
100
0
0
10
20
30
40
50
60
70
80
90
100



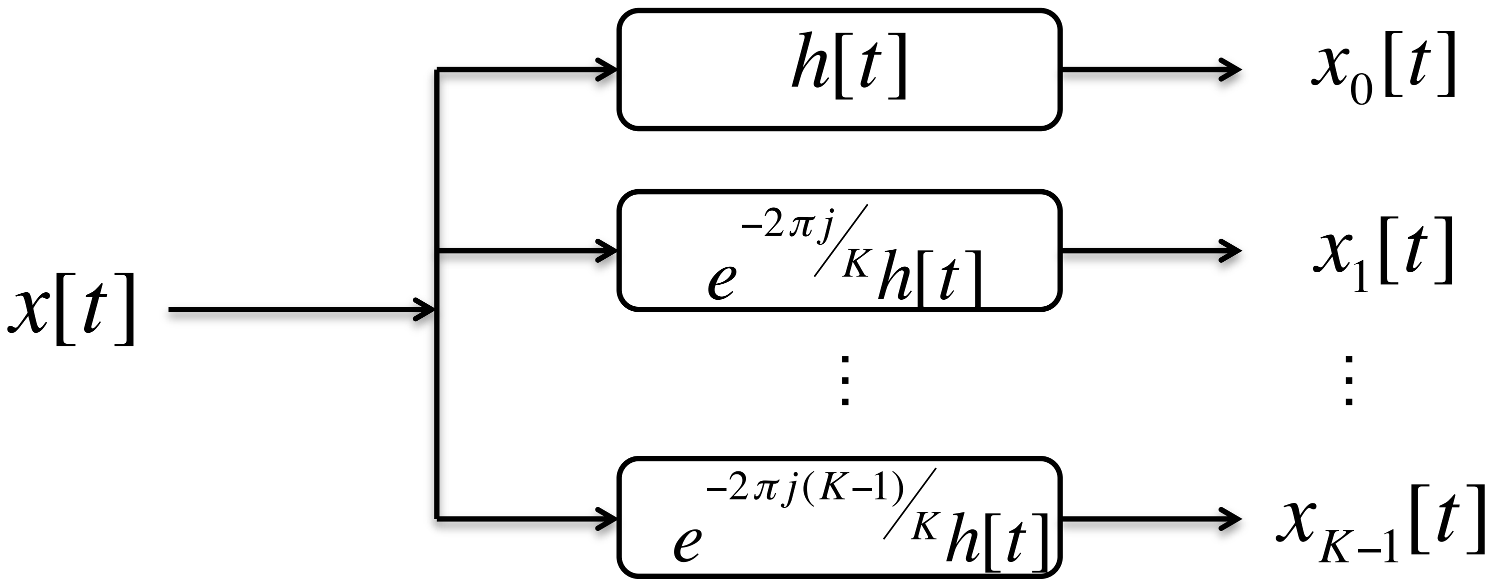
Spectrum after permutation
σ = 31
ω 3
ω 2
ω 4
ω 1
600
500
400
300
200
100
0
0
10
20
30
40
50
60
70
80
90
100

Filtering

- Goal is to separate the spectrum into several parts. Each bin will contain only one frequency in ideal situation.
- Pass the signal through a band-pass filter.
- Each bin will have a largest value frequency -- could be noise or nothing at all.
- Do single frequency recovery for each bin.
- We are democratic, we vote.

Filtering Bank

h[t]: band-pass filter



Single Frequency Recovery

[Alias-based Search]

- Pick small number $m_1, m_2, \ldots, m_k$ equally spaced points to do FFT.
- Find the largest value ω_k in FFT.
- Then $\omega_i \equiv m_i \pmod{N}$
- Use *CRT algorithm* to solve the index.
- Map back index to original frequencies, which are presented before permutation.

Single Frequency Recovery

- *Chinese Remainder Theorem (CRT)*

– Ex:

N=30=2x3x5,

A mod 2=1, A mod 3=1, A mod 5=1. A=?

–Answer: A=21

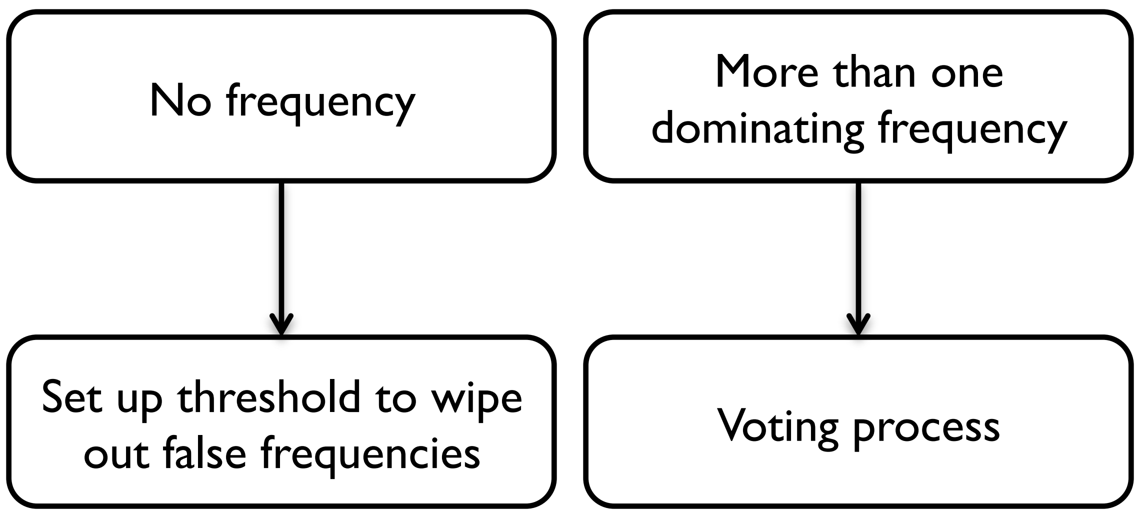
It can be uniquely determined as long as

2x3x5 ≥ N

Value Estimation

-- Unbiased estimation

- Use the value obtained during single frequency recovery.
- Consider following situations in recovery:



Result Analysis

- Robustness to noise
  - The input contains 5kHz, 6kHz, 6.5kHz, 6.7kHz, 11.45kHz and 11.668kHz.
  - With six inputs, set up k=6.
  - And the input signal is combined with the white Gaussian noise, depends on the SNR.

- Test in no-noise, high-SNR and low-SNR:

No-noise

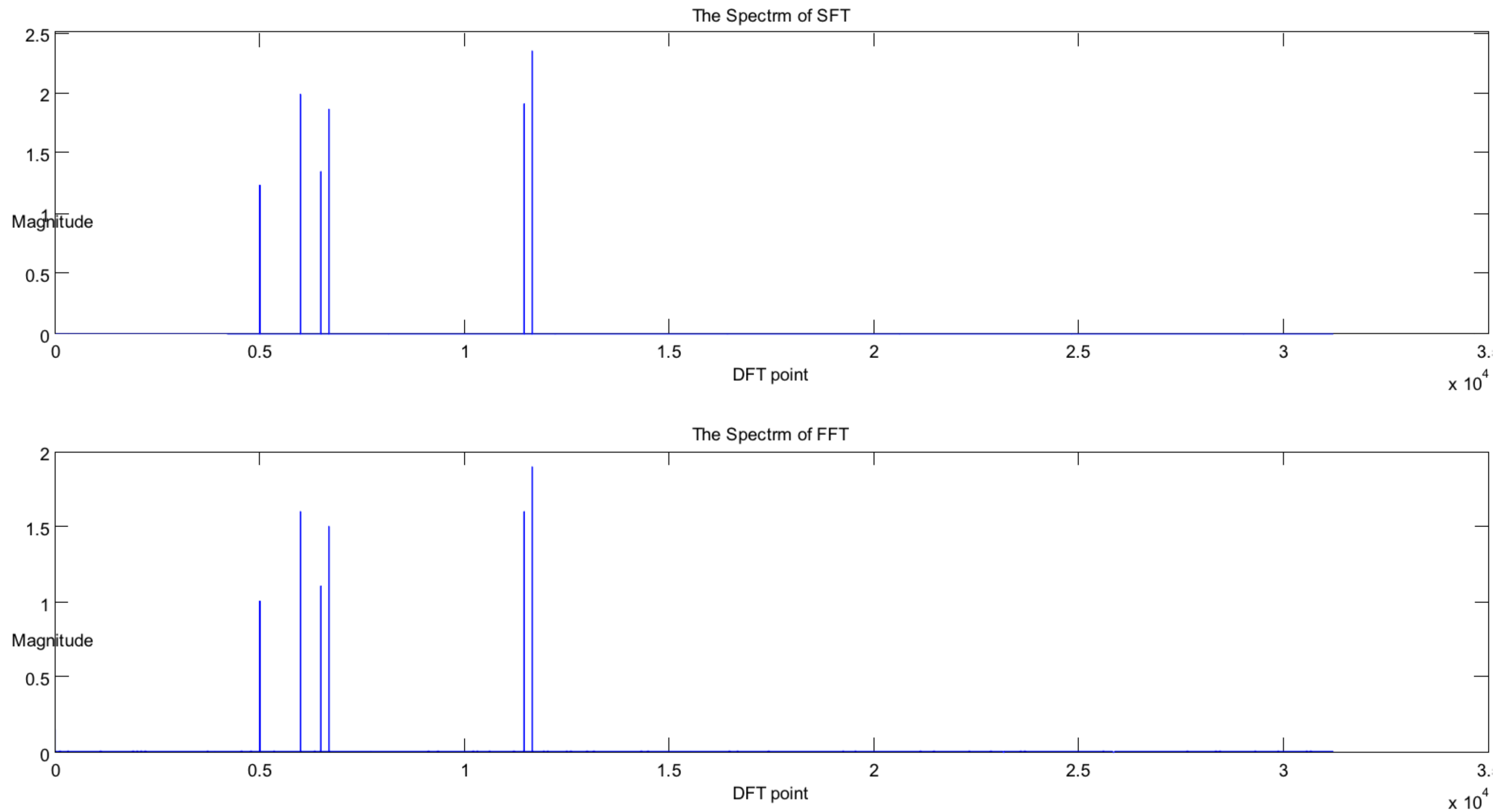


Figure 5: The SFT and FFT of the input signal(with no noise)

High-SNR

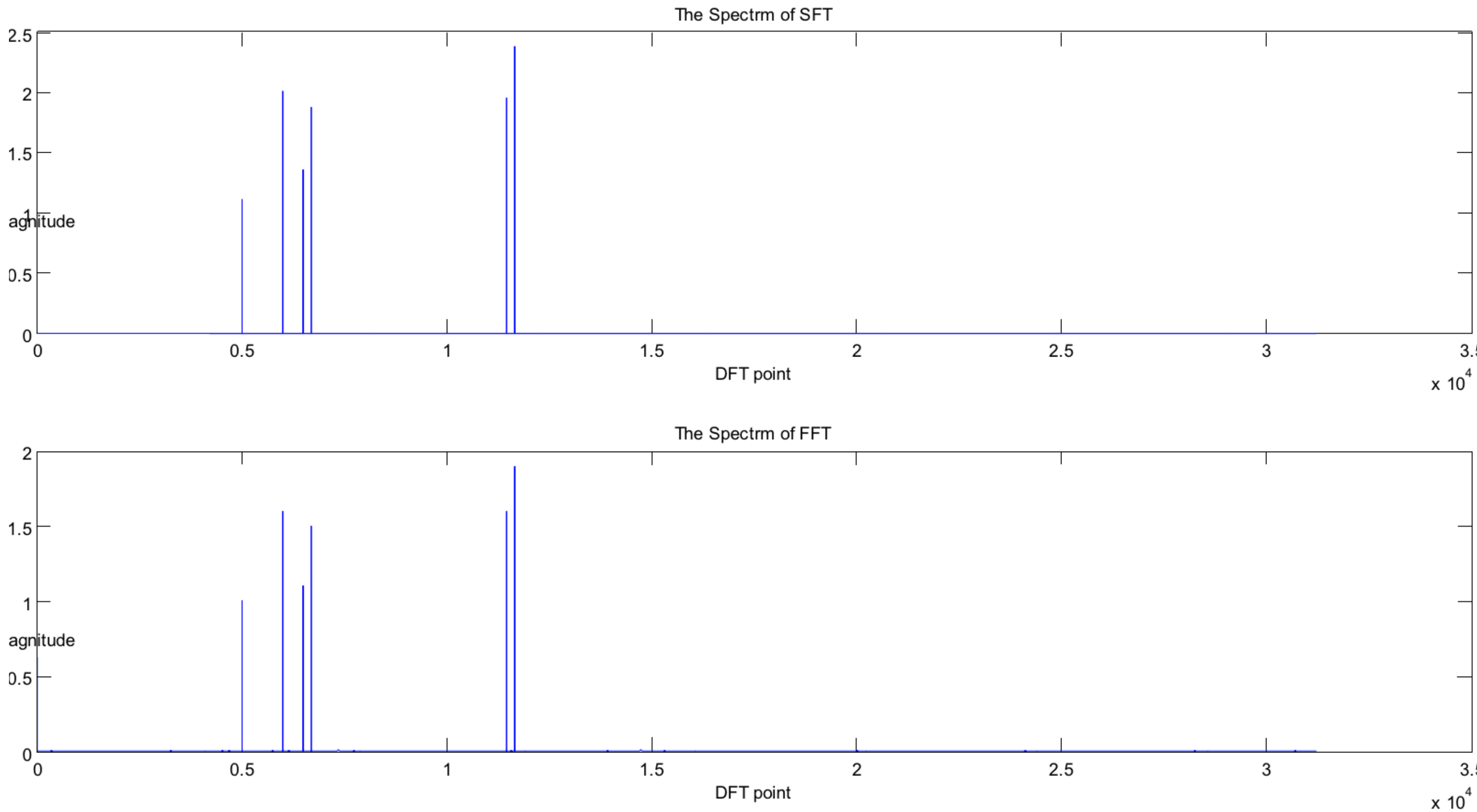


Figure 6 The SFT and FFT of the input signal(with SNR=5)

Low-SNR

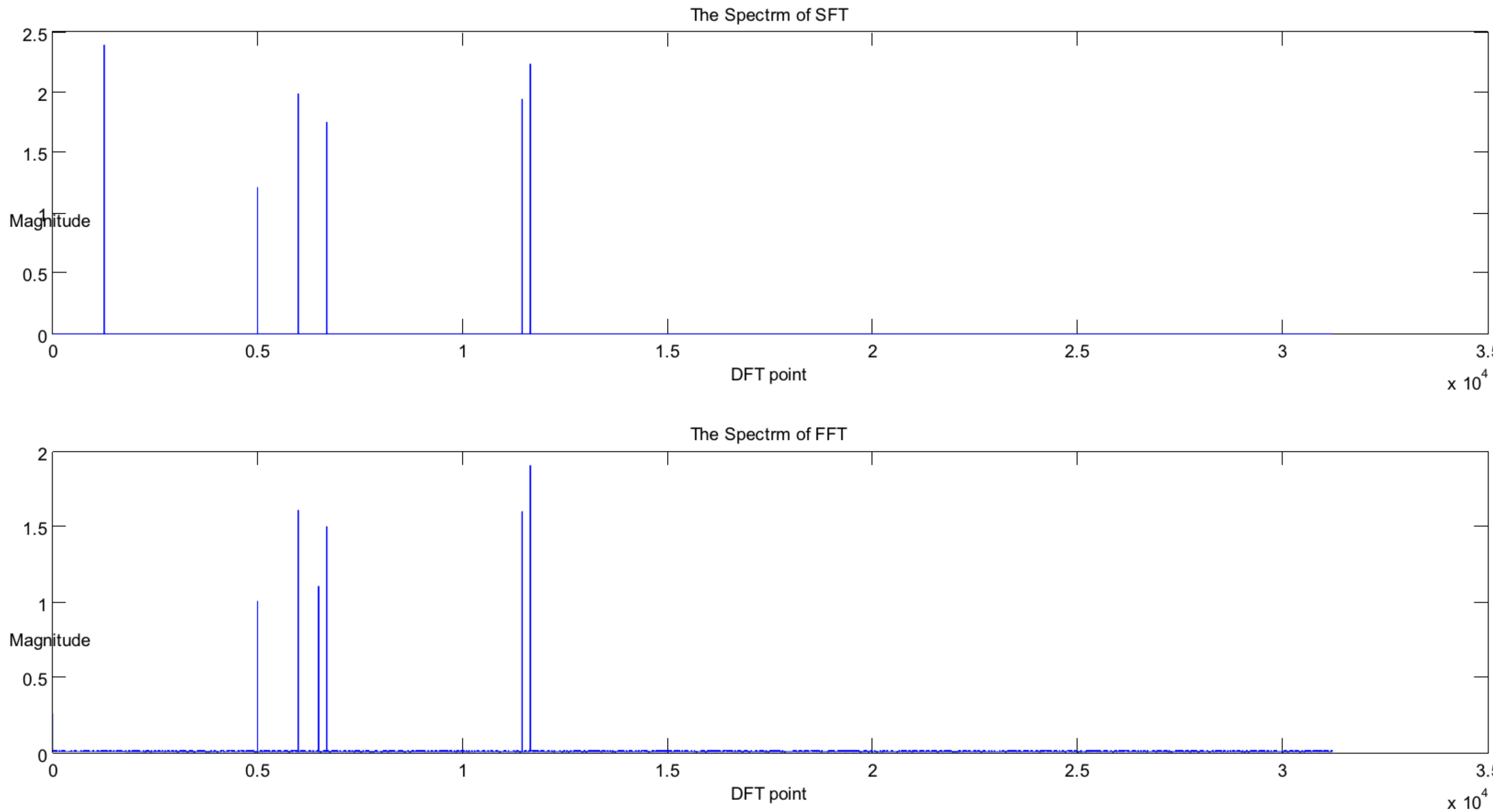
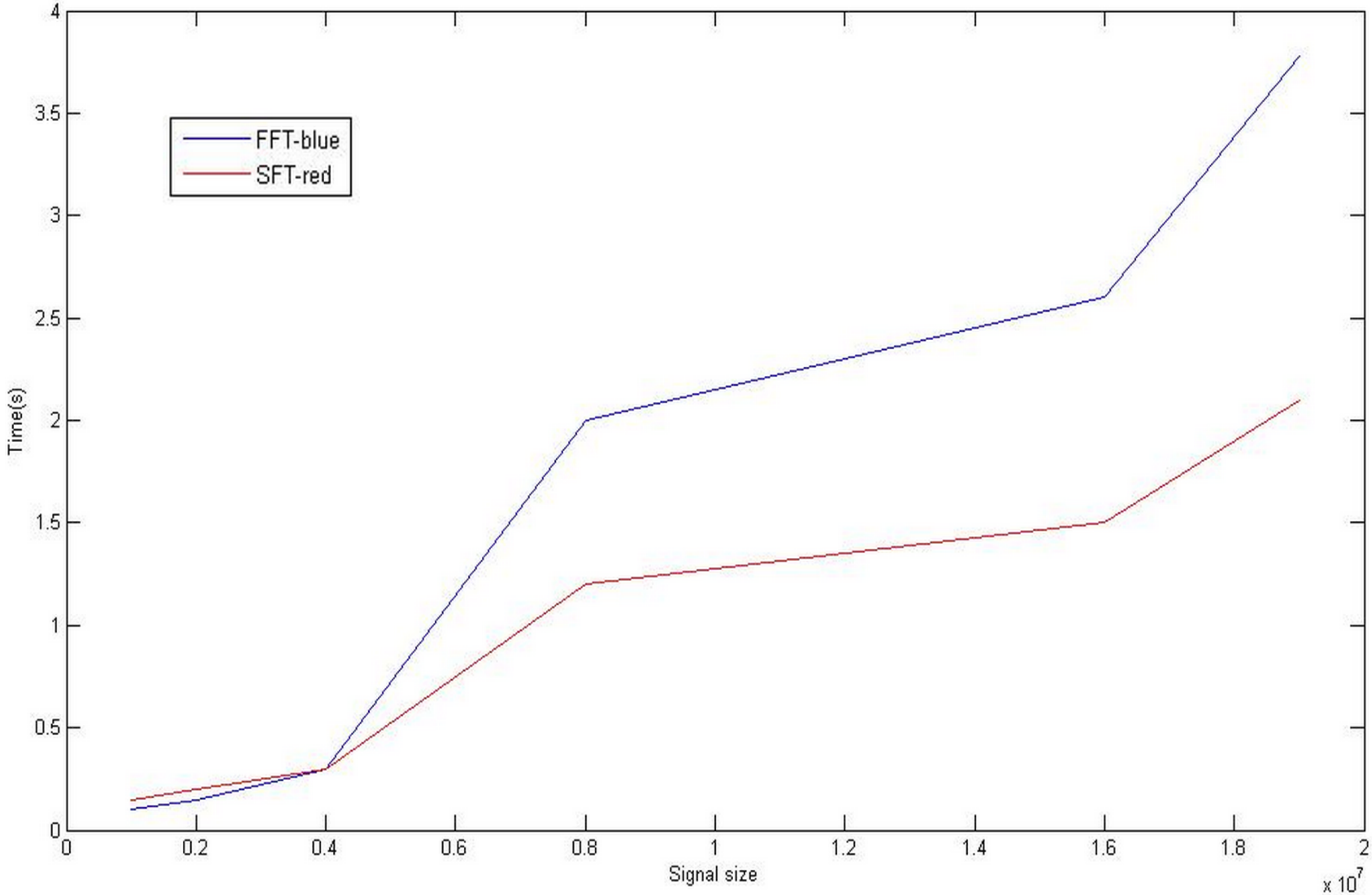


Figure 9: The SFT and FFT of the input signal(with SNR=0.5)

Low-SNR

- Noise may occupy some ‘bin’ during binning process.
- Increase the times of iteration will guarantee since noise appears randomly in spectrum.

Running Time Analysis



Future Work

- Permutation algorithm
- Filter performance
- Flexibility of the input length
- Voting process
- Stability
- Accuracy