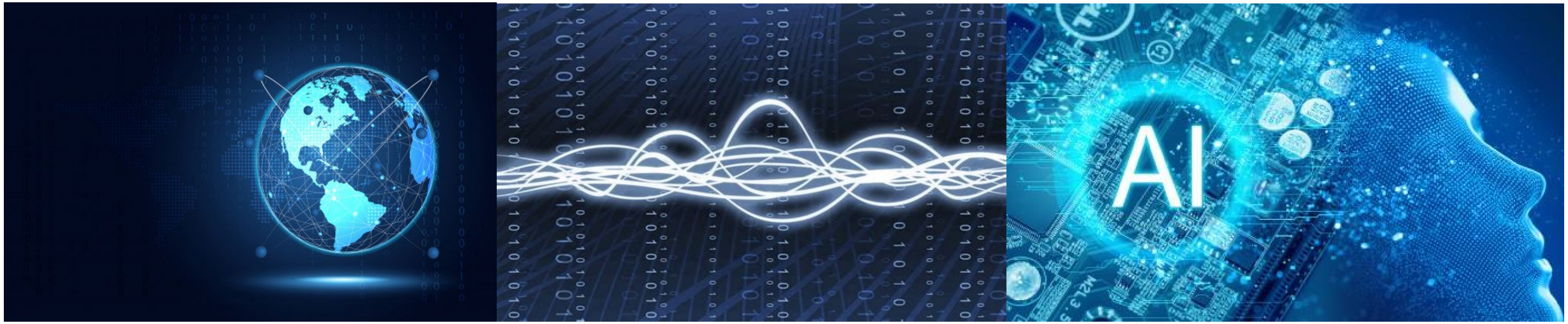


《数字信号处理》

授课教师：王丰
fengwang13@gdut.edu.cn
广东工业大学信息工程学院电子系

第四章 快速傅里叶变换（FFT）

fengwang13@gdut.edu.cn

主要内容：

4.1 直接计算 DFT 的问题及改进途径

4.2&4.3 FFT 算法

按时间抽选 (DIT) 的基-2 FFT 算法

按频率抽选 (DIF) 的基-2 FFT 算法

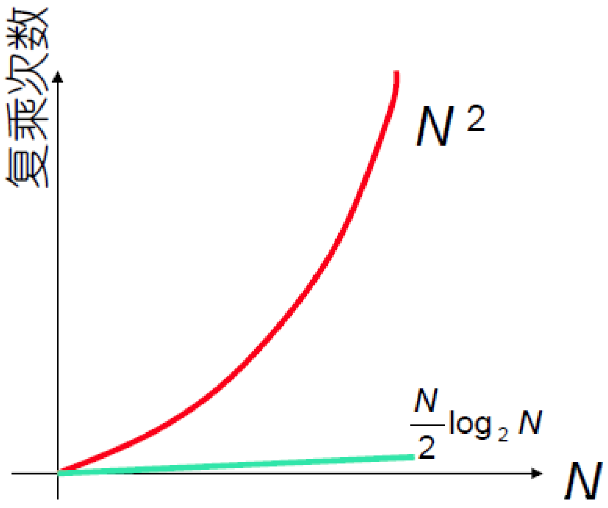
4.4 DIT-FFT与DIF-FFT的异同

fengwang13@gdut.edu.cn

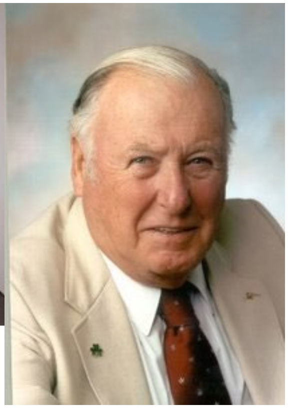
*DSP 发展的里程碑

*FFT是DFT 的一种快速高效计算方法，而不是一种新的变换，可以在数量级的意义上提高运算速度。

1965，Cooley和Tukey在《计算数学》(Mathematics of Computation)上发表“用机器计算复序列傅里叶级数的一种算法”为数字信号处理学科的开始。




James Cooley


John Tukey

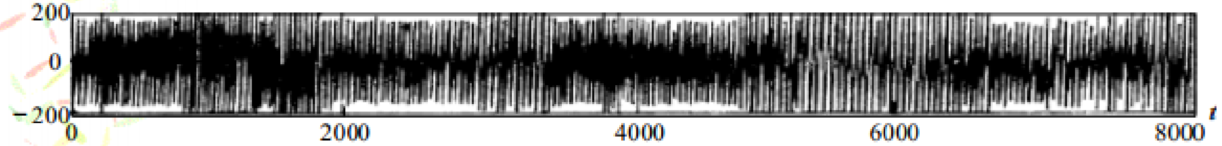
Example (1969)：N=2048

- DFT time = 13.5 hours
- FFT time = 2.4 seconds

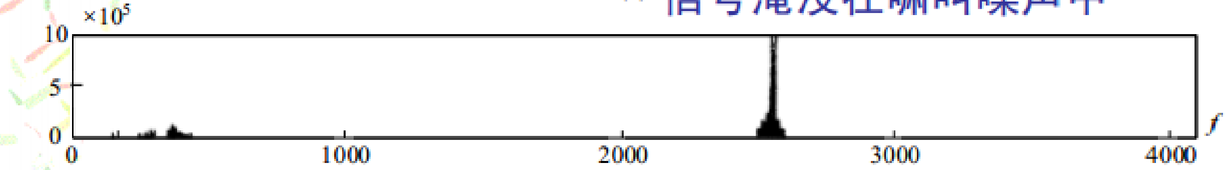
Cooley, J.W. and Tukey, J.W. (1965) An Algorithm for the Machine Calculation of Complex Fourier Series. Mathematics of Computation, 19, 297-301.

fengwang13@gdut.edu.cn

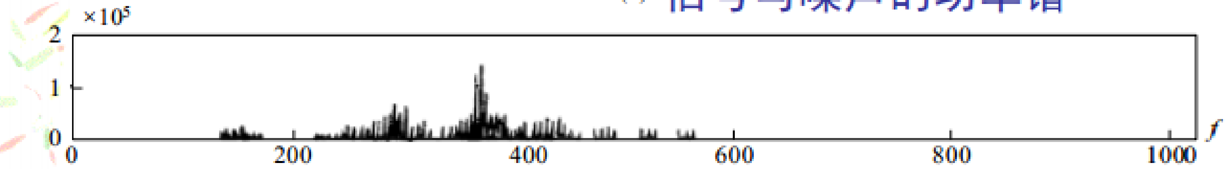
FFT 应用：语音信号消噪过程



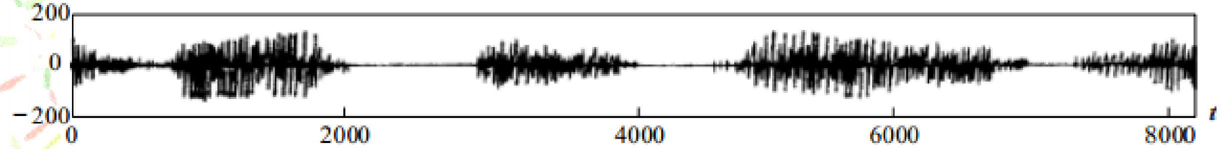
(a) 信号淹没在啸叫噪声中



(b) 信号与噪声的功率谱



(c) 去噪后的功率谱



(d) 重构原语音信号

4.1 直接计算DFT的问题及改进途径

fengwang13@gdut.edu.cn

DFT: $X(k)=\sum_{n=0}^{N-1}x(n)W_N^{kn},\qquad k=0,1,\ldots,N-1$

IDFT: $x(n)=\frac{1}{N}\sum_{k=0}^{N-1}X(k)W_N^{-kn},\qquad n=0,1,\ldots,N-1$

注释:
DFT与IDFT的计算量基本相同，只差一个因子 $1/N$

思考：对长度为 N 的有限长序列 $x(n)$， 计算其 DFT需多大的运算量?

	复数乘法	复数加法
一个$X(k)$	N	$N-1$
N个$X(k)$ (***N*点DFT**)	N^2	$N(N-1)$

fengwang13@gdut.edu.cn

例：石油勘探， 24 道记录，每道波形记录长度 5 秒

1、若每秒抽样 500 点 ， 则
- ✓ 每道总抽样点数： 500*5=2500 点
- ✓ 24 道总抽样点数： 24*2500=60000 点

2、在1的条件下，若一台计算机的速度是每次复数乘法20ns，每次复数加法5ns，则
- ✓ 直接DFT 复数乘法运算量： $N^2=(60000)^2=3.6*10^9$
- ✓ 直接DFT 复数加法运算量约为： $N^2=(60000)^2=3.6*10^9$
- ✓ 总计算时间：T=(20+5)*3.6=90s=1.5min

fengwang13@gdut.edu.cn

FFT算法基本思想

FFT算法就是不断地把长序列的DFT分解成几个短序列的DFT，并利用旋转因子W_N^{nk}的**周期性和对称性**，减少DFT的运算次数。

$$W_N^{m+lN}=e^{-j\frac{2\pi}{N}(m+lN)}=e^{-j\frac{2\pi}{N}m}=W_N^m$$

$$W_N^{-m}=W_N^{N-m} \qquad [W_N^{N-m}]^*=W_N^m$$

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

fengwang13@gdut.edu.cn

FFT 算法分类

◆ 按抽取方法

1. 时间抽选法 DIT (Decimation-In-Time)
2. 频率抽选法 DIF (Decimation-In-Frequency)

◆ 按“基数”

1. 基-2 FFT 算法： N 为 2 的整数幂的 FFT 算法
2. 基-4 FFT 算法
3. 混合基 FFT 算法
4. 分裂基 FFT 算法

4.2 按时间抽选 (DIT) 的基-2 FFT 算法 (Cooley-Tukey 算法)

fengwang13@gdut.edu.cn

1、算法原理

设序列点数 $N=2^L$， L 为正整数；**若不满足，则补零。**

将序列 $x(n)$ 按 n 的奇偶分成两组：

$$\begin{cases} x_1(r)=x(2r) \\ x_2(r)=x(2r+1) \end{cases}, \quad r=0,1,\ldots,\frac{N}{2}-1$$

$$k=0,1,\ldots,\frac{N}{2}-1$$

$$\begin{aligned}
X(k) &= \sum_{n=even} x(n)W_N^{kn} + \sum_{n=odd} x(n)W_N^{kn} \\
&= \sum_{r=0}^{N/2-1} x(2r)W_N^{2kr} + \sum_{r=0}^{N/2-1} x(2r+1)W_N^{k(2r+1)} \\
&= \sum_{r=0}^{N/2-1} x_1(r)W_N^{2kr} + W_N^k \sum_{r=0}^{N/2-1} x_2(r)W_N^{2kr} \\
&= \underbrace{\sum_{r=0}^{N/2-1} x_1(r)W_{N/2}^{kr}}_{X_1(k)=DFT[x_1(r)]} + W_N^k \underbrace{\sum_{r=0}^{N/2-1} x_2(r)W_{N/2}^{kr}}_{X_2(k)=DFT[x_2(r)]} \\
&= X_1(k)+W_N^k X_2(k)
\end{aligned}$$

4.2 按时间抽选 (DIT) 的基-2 FFT 算法 (Cooley-Tukey 算法)

fengwang13@gdut.edu.cn

再利用周期性求 $X(k)$ 的后半部分

$$\because\ W_{N/2}^{r(N/2+k)} = W_{N/2}^{rk}$$

$$X_1(\frac{N}{2}+k) = \sum_{r=0}^{N/2-1} x_1(r) W_{N/2}^{r(N/2+k)} = \sum_{r=0}^{N/2-1} x_1(r) W_{N/2}^{rk} = X_1(k)$$

$$X_2(\frac{N}{2}+k) = X_2(k)$$

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

$$\begin{cases} X(k) = X_1(k) + W_N^k X_2(k), \quad k = 0,1,\ldots,N/2-1 \\ X(\frac{N}{2}+k) = X_1(\frac{N}{2}+k) + W_N^{(N/2+k)} X_2(\frac{N}{2}+k) \end{cases}$$

$$= X_1(k) - W_N^k X_2(k), \quad k = 0,1,\ldots N/2-1$$

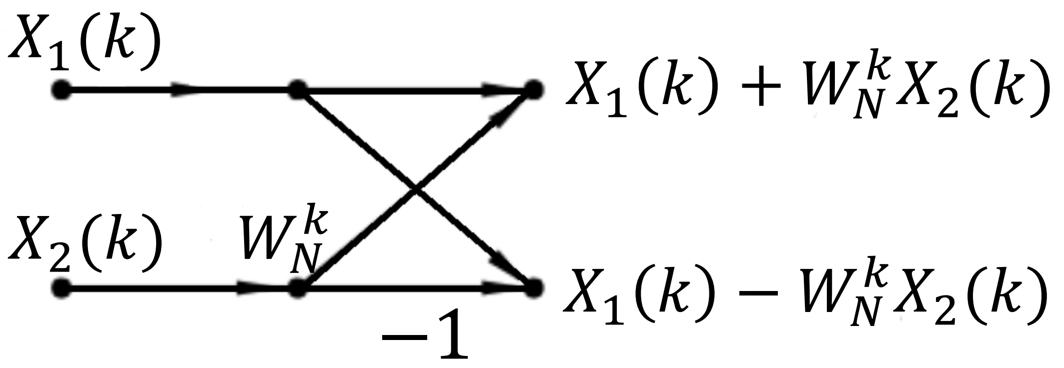
4.2 按时间抽选 (DIT) 的基-2 FFT 算法 (Cooley-Tukey 算法)

fengwang13@gdut.edu.cn

小结

$$\therefore \begin{cases} X(k) = X_1(k) + W_N^k X_2(k) \\ X(k+\frac{N}{2}) = X_1(k) - W_N^k X_2(k) \end{cases} \qquad k = 0,1,\ldots,N/2-1$$

□ 将该运算用“蝶形”信号流图表示：



Butterfly：“会飞的花朵”

注释：

① 上支路为加法，下支路为减法

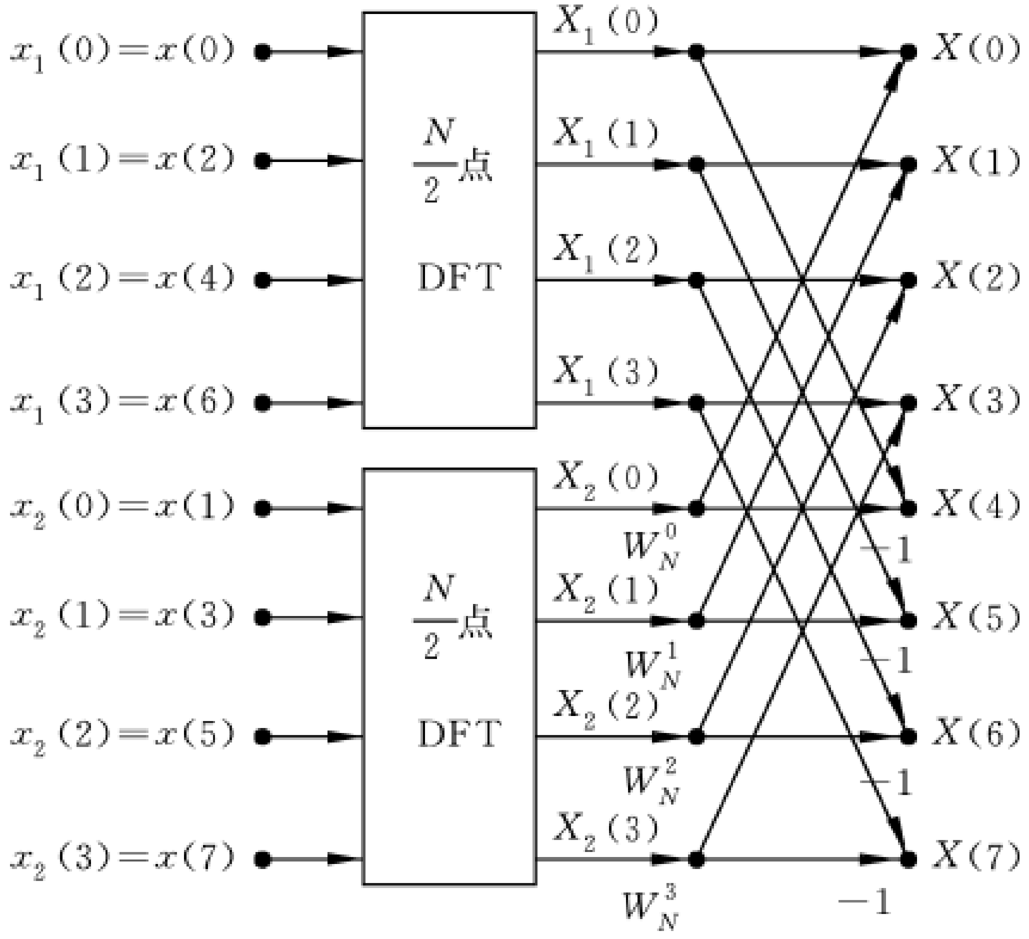
② 乘法运算的支路标“箭头”和“系数”

③ 若支路上没有标出系数，则该支路的传输系数为1

4.2 按时间抽选 (DIT) 的基-2 FFT 算法 (Cooley-Tukey 算法)

fengwang13@gdut.edu.cn

将一个 N 点 DFT 分解为两个 $N/2$ 点 DFT： $N=8=2^3$



	复数乘法	复数加法
一个 $N/2$ 点 DFT	$(N/2)^2$	$N/2\ (N/2-1)$
两个 $N/2$ 点 DFT	$N^2/2$	$N\ (N/2-1)$
一个蝶形	1	2
$N/2$ 个蝶形	$N/2$	N
总计	$N^2/2+N/2$ $\approx N^2/2$	$N(N/2-1)+N$ $\approx N^2/2$

□ 运算量减少了近一半！

4.2 按时间抽选 (DIT) 的基-2 FFT 算法 (Cooley-Tukey 算法)

fengwang13@gdut.edu.cn

- 由于 $N=2^L$， $N/2=2^{L-1}$ 仍为偶数，因此，2个 $N/2$ 点DFT 又可同样进一步分解为 4 个 $N/4$ 点的 DFT

$$\begin{cases} x_1(2l)=x_3(l) \\ x_1(2l+1)=x_4(l) \end{cases} \quad l=0,1,\ldots,\frac{N}{4}-1$$

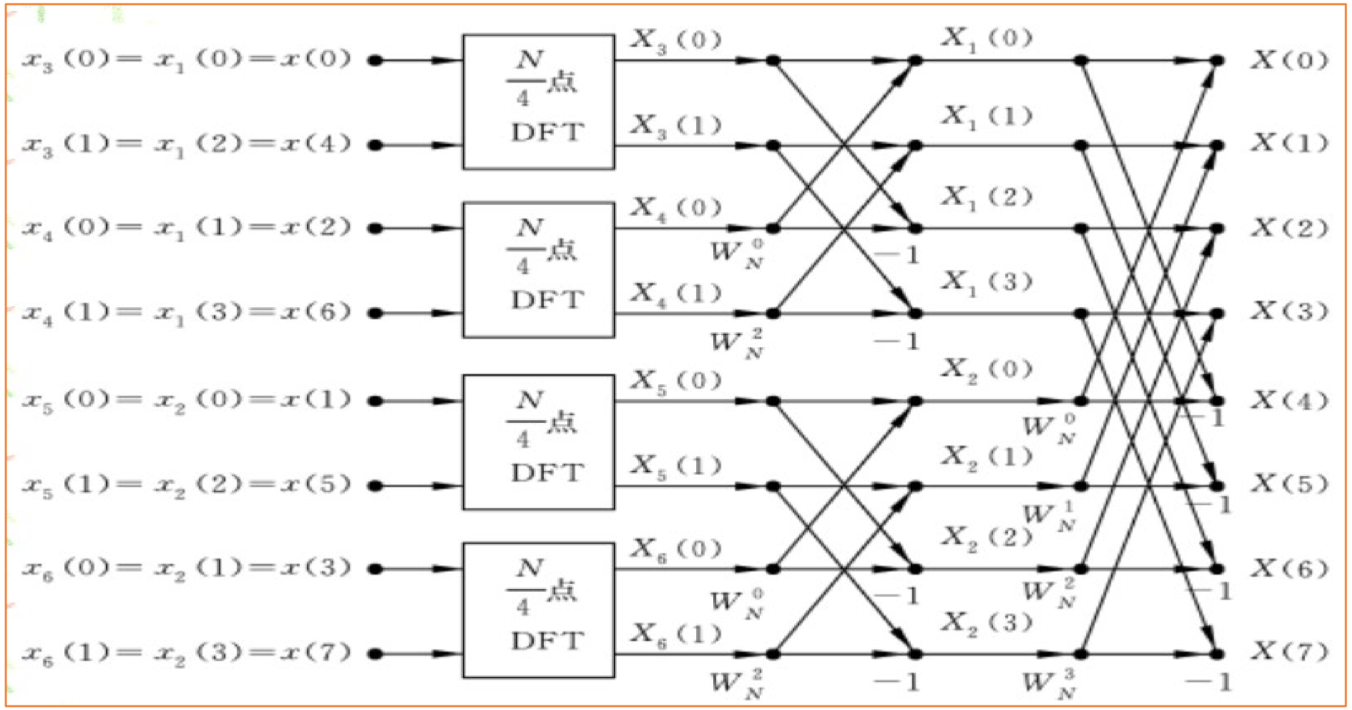
$$\begin{cases} X_1(k)=X_3(k)+W_{N/2}^{k}X_4(k) \\ X_1(\frac{N}{4}+k)=X_3(k)-W_{N/2}^{k}X_4(k) \end{cases} \quad k=0,1,\ldots,\frac{N}{4}-1$$

$x_2(r)$ 也可进行同样的分解，得到：

$$\begin{cases} X_2(k)=X_5(k)+W_{N/2}^{k}X_6(k) \\ X_2(\frac{N}{4}+k)=X_5(k)-W_{N/2}^{k}X_6(k) \end{cases} \quad k=0,1,\ldots,\frac{N}{4}-1$$

由2个 $N/4$ 点 DFT 组合成1个 $N/2$ 点 DFT

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



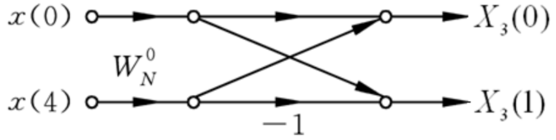
4.2 按时间抽选 (DIT) 的基-2 FFT 算法 (Cooley-Tukey 算法)

fengwang13@gdut.edu.cn

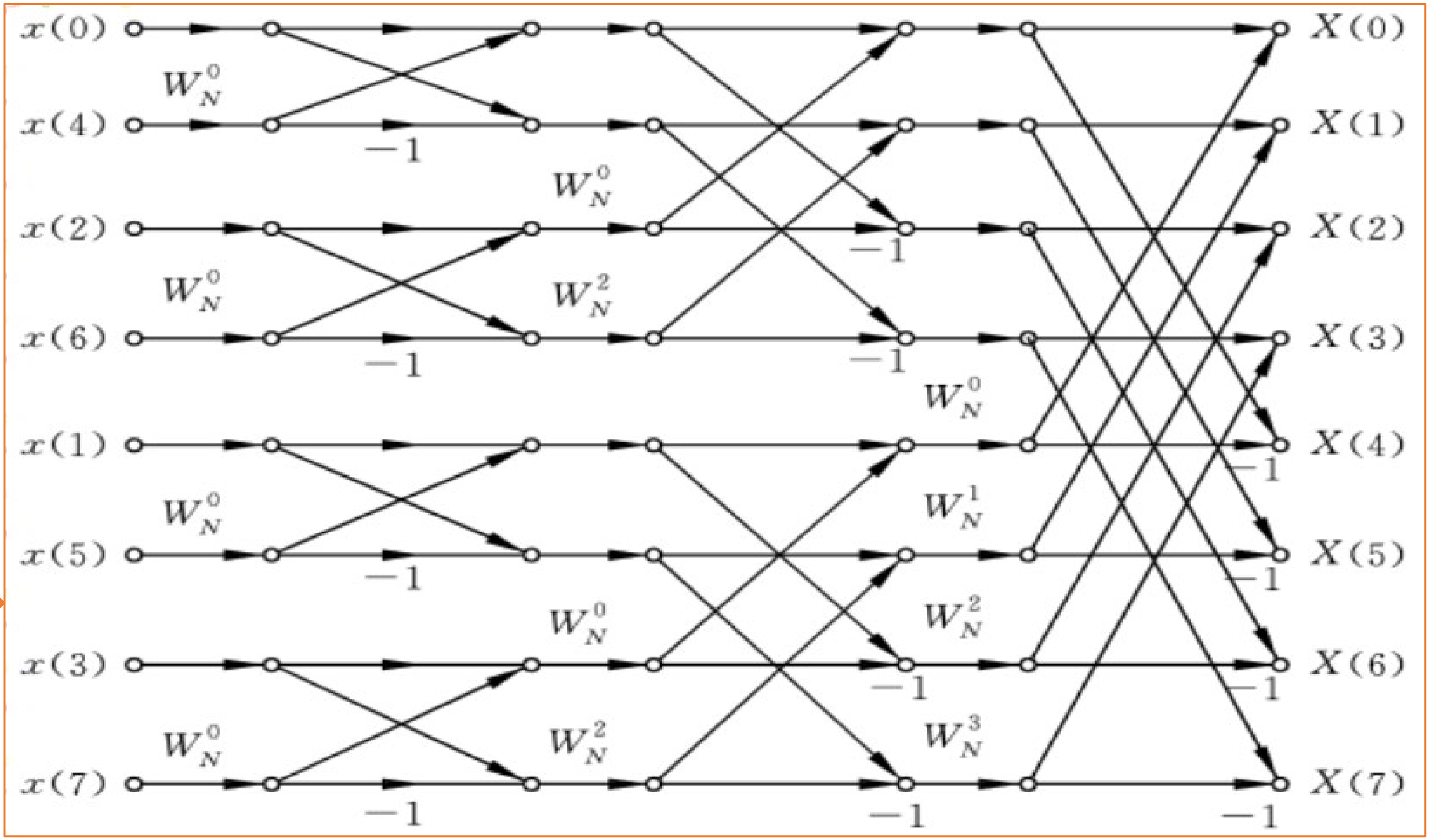
- 如此不断分解，直到分解为 2 点 DFT。

$$
\begin{cases}
X_3(0) = x(0) + W_2^0 x(4) \\
\qquad\quad = x(0) + W_N^0 x(4) \\
\qquad\quad = x(0) + x(4) \\
X_3(1) = x(0) - x(4)
\end{cases}
$$

两点 DFT实际上只是加减运算



第1级　　第2级　　第3级



每一步分解都是按输入序列在时间上的次序是属于偶数还是奇数来分解为两个更短的序列，所以称为“按时间抽选法”

4.2 按时间抽选 (DIT) 的基-2 FFT 算法 (Cooley-Tukey 算法)

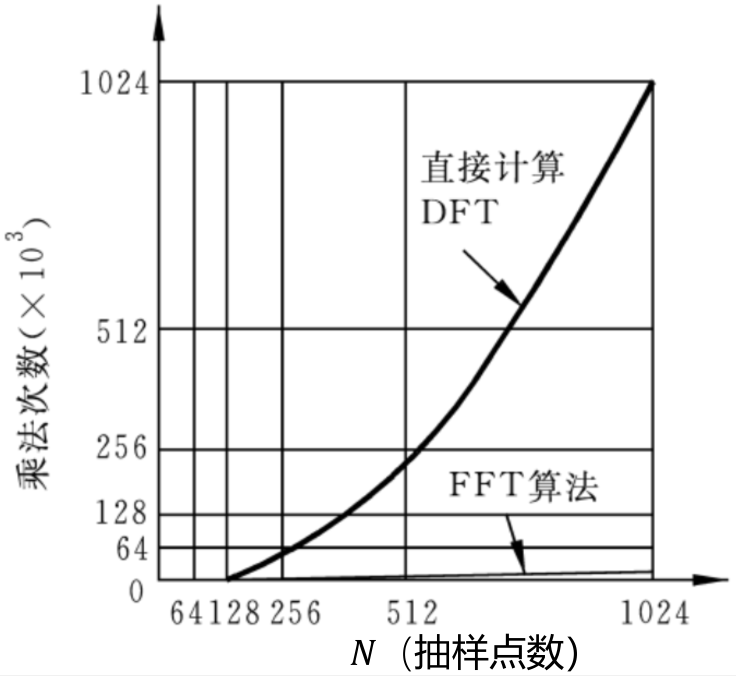
fengwang13@gdut.edu.cn

2、运算量

- **$N=2^L$时，共有 $L=\log_2 N$ 级蝶形**
- 每级有 $N/2$ 个蝶形运算
- 每级 $N/2$ 次复乘法，N 次复加（每个蝶形有1次复乘，2次复加）。
- 因此，共有复乘法$L*(N/2)=(N/2)\log_2 N$ 次、复加法 $L*N=N\log_2 N$ 次。

比较 DFT：
$$\frac{m_F(\text{DFT})}{m_F(\text{FFT})}=\frac{N^2}{\frac{N}{2}\log_2 N}=\frac{2N}{\log_2 N}$$

◆ 当点数 N 越大时， FFT 的优点越突出

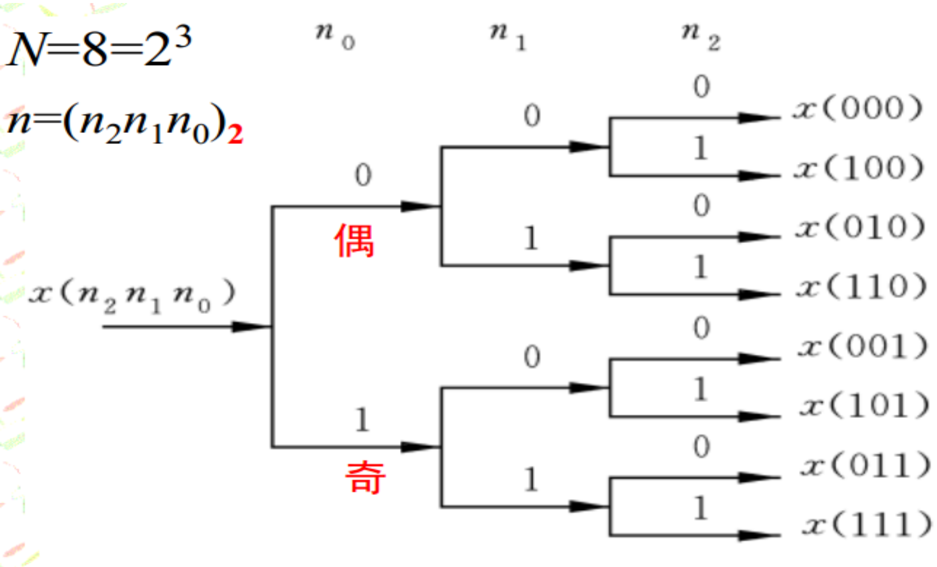


4.2 按时间抽选 (DIT) 的基-2 FFT 算法 (Cooley-Tukey 算法)

fengwang13@gdut.edu.cn

3、倒位序规律

输入 *x*(*n*) 按标号 *n* 的奇偶分组造成倒位序



$$n=(n_2n_1n_0)_2 \quad ,\text{则}\hat{n}=(n_0n_1n_2)_2$$

自然顺序	二进制数	倒位序二进制数	倒位序
0	000	000	0
1	001	100	4
2	010	010	2
3	011	110	6
4	100	001	1
5	101	101	5
6	110	011	3
7	111	111	7

4.2 按时间抽选 (DIT) 的基-2 FFT 算法 (Cooley-Tukey 算法)

fengwang13@gdut.edu.cn

例 5：试求出 N=64 时用 DIT 共有多少级，每级有多少个蝶形单元，并写出每一级的旋转因子。

解：$N = 64 = 2^6$

$\therefore$ 蝶形图共有 6 级，每级 32 个蝶形单元。

$m=1$，旋转因子为 W_{64}^{0}，为 1 个

$m=2$，旋转因子为 W_{64}^{0}，W_{64}^{16}，共 2 个

$m=3$，旋转因子为 W_{64}^{0}，W_{64}^{8}，W_{64}^{16}，W_{64}^{24}，共 4 个

$m=4$，旋转因子为 W_{64}^{0}，W_{64}^{4}，$W_{64}^{8},\cdots$，W_{64}^{28}，共 8 个

$m=5$，旋转因子为 W_{64}^{0}，W_{64}^{2}，$W_{64}^{4},\cdots$，W_{64}^{30}，共 16 个

$m=6$，旋转因子为 W_{64}^{0}，W_{64}^{1}，$W_{64}^{2},\cdots$，W_{64}^{31}，共 32 个

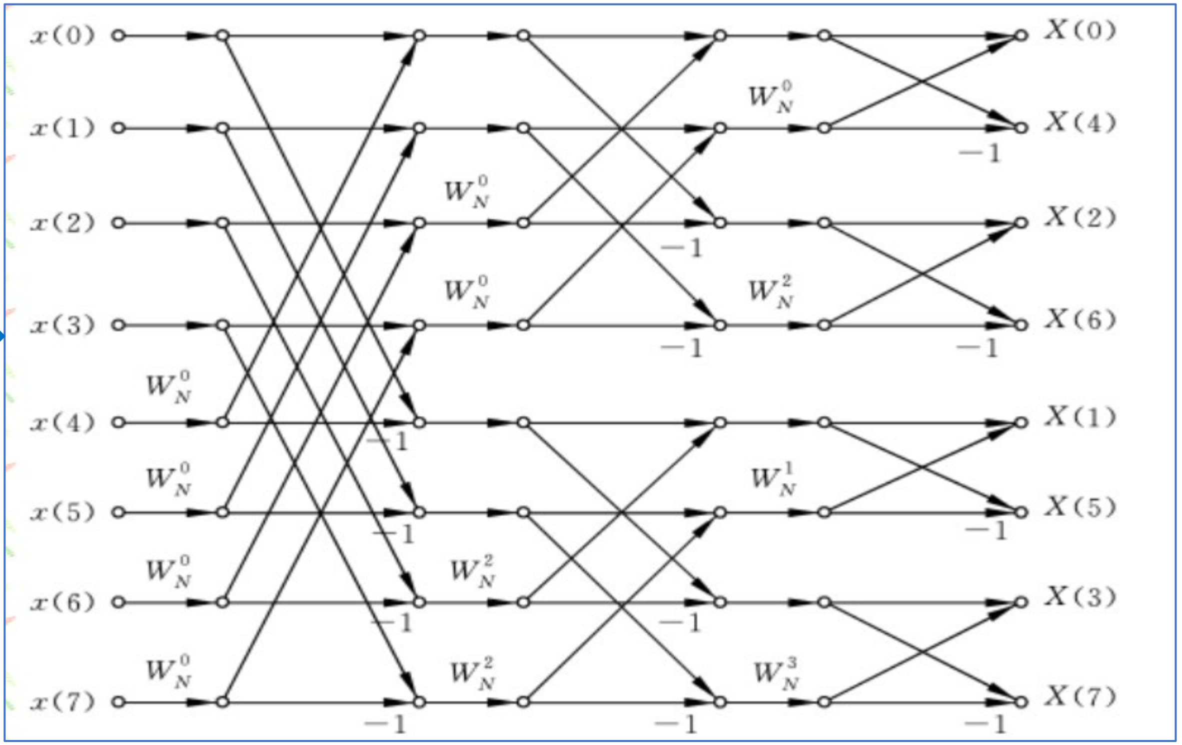
4.2 按时间抽选 (DIT) 的基-2 FFT 算法 (Cooley-Tukey 算法)

fengwang13@gdut.edu.cn

4、 DIT-FFT 算法的其他形式流图

- 只要保持各节点所连的支路及其传输系数不变，则不论节点位置在同一列中如何排列，所得流图都是等效的，只是数据的提取和存放的次序不同

按时间抽选，输入自然顺序、输出倒位序的 FFT 流图



4.3 按频率抽选 (DIF) 的基-2 FFT 算法 (桑德-图基算法)

fengwang13@gdut.edu.cn

1、算法原理

□ 与 DIT-FFT 算法类似分解，但是抽选的是 *X*(*k*)，即 *X*(*k*) 分解为奇数与偶数序号的两个序列

$$X(k)=\sum_{n=0}^{N/2-1}x(n)W_N^{nk}+\sum_{n=N/2}^{N-1}x(n)W_N^{nk}$$

$$=\sum_{n=0}^{N/2-1}x(n)W_N^{nk}+\sum_{n=0}^{N/2-1}x(n+\frac{N}{2})W_N^{(n+\frac{N}{2})k}$$

$$=\sum_{n=0}^{N/2-1}\left[x(n)+W_N^{Nk/2}x(n+\frac{N}{2})\right]W_N^{nk}$$

$$W_N^{Nk/2}=(-1)^k$$

$$=\sum_{n=0}^{N/2-1}\left[x(n)+(-1)^k x\left(n+\frac{N}{2}\right)\right]W_N^{nk}$$

$$k=0,1,\ldots,N-1$$

按 k 的奇偶将 $X(k)$ 分成两部分

$$X(2r)=\sum_{n=0}^{N/2-1}\left[x(n)+x(n+\frac{N}{2})\right]W_N^{2nr}$$

$$=\sum_{n=0}^{N/2-1}\left[x(n)+x(n+\frac{N}{2})\right]W_{N/2}^{nr}$$

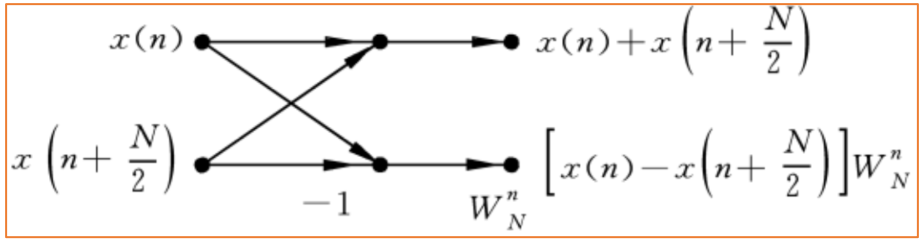
$$r=0,1,\ldots,\frac{N}{2}-1$$

$$X(2r+1)=\sum_{n=0}^{N/2-1}\left[x(n)-x(n+\frac{N}{2})\right]W_N^{(2r+1)n}$$

$$=\sum_{n=0}^{N/2-1}\left\{\left[x(n)-x(n+\frac{N}{2})\right]W_N^{n}\right\}W_{N/2}^{nr}$$

$X(2r)$, $X(2r+1)$ 对应两个 $N/2$ 点的 DFT。

按频率抽选蝶形运算流图

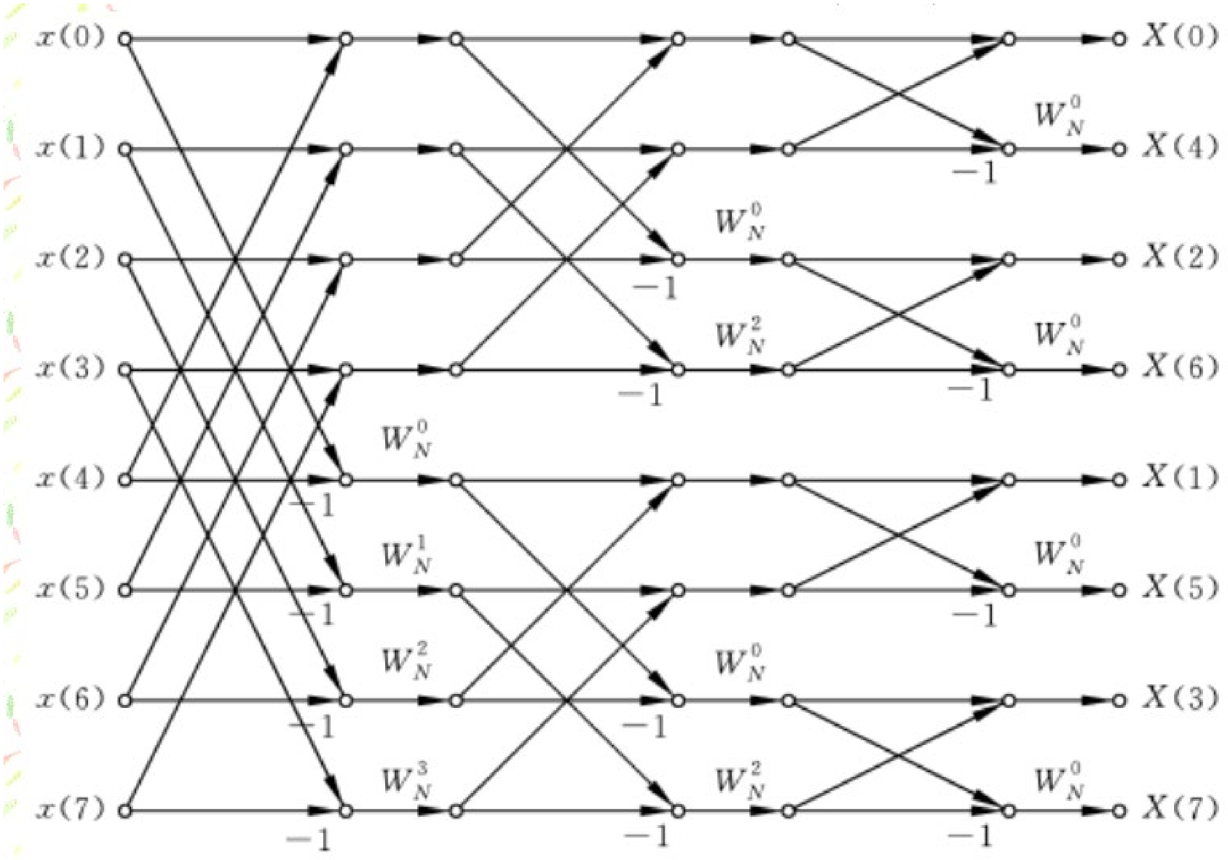


4.3 按频率抽选 (DIF) 的基-2 FFT 算法 (桑德-图基算法)

fengwang13@gdut.edu.cn

- 按照分解思路继续分解，直到只计算 2 点的 DFT

- $N=8=2^3$ **基-2 DIF-FFT 流图 (L=3)**



4.4 DIT-FFT 与 DIF-FFT 的异同

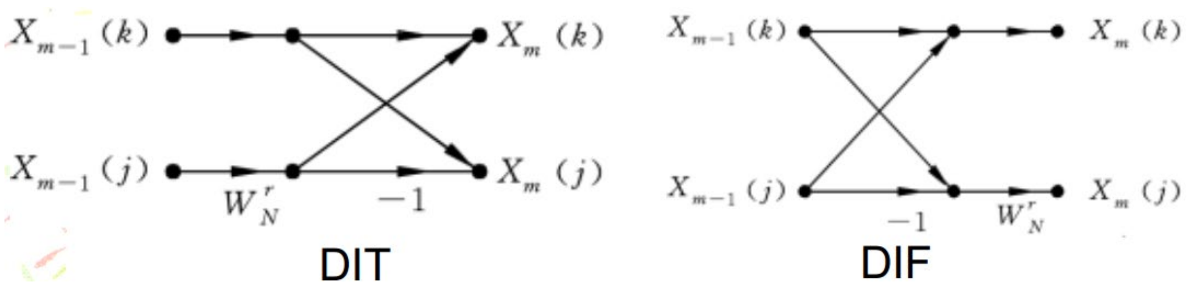
fengwang13@gdut.edu.cn

注释：
“转置”就是将原流图的所有支路方向都反向，并交换输入与输出变量，但节点变量值不交换

1、基本蝶形不同

- DIT：先作复乘，后作加减法
- DIF：先作减法，后作复乘

2、DIT 和DIF 的基本蝶形互为转置



3、运算量相同

- *L* 级（列）运算，每级 *N*/2 个蝶形

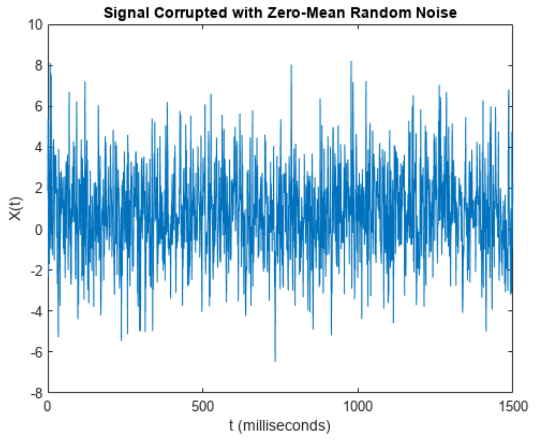
复乘：$m_F = \frac{N}{2}\log_2 N$

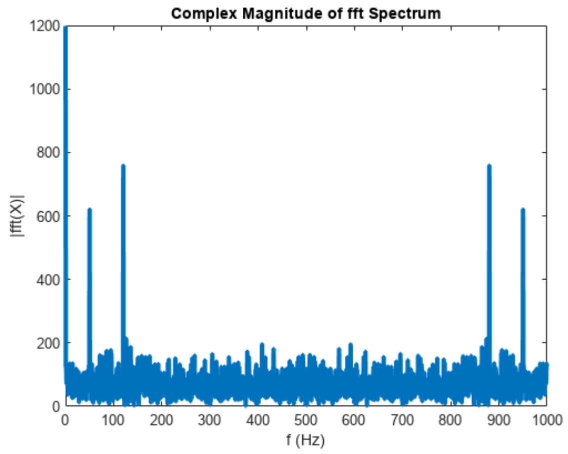
复加：$a_F = N\log_2 N$

Matlab code: FFT编程

fengwang13@gdut.edu.cn

- Y= fft(x) computes DFT of X using a FFT algorithm. Y is the same size as X.
- Y= fft(x,n) returns the n-point DFT





例1：混有噪声的信号

```
fs = 1000; % sampling frequency
T = 1/fs;   % sampling period
L = 1500; % length of signal
t = (0:L-1)*T;
S = 0.8 + 0.7*sin(2*pi*50*t) + sin(2*pi*120*t);
% signal of interest
X = S + 2*randn(size(t)); % noisy signal
figure;
plot(1000*t, X);
xlabel('t (ms)');
ylabel('X(t)');
```

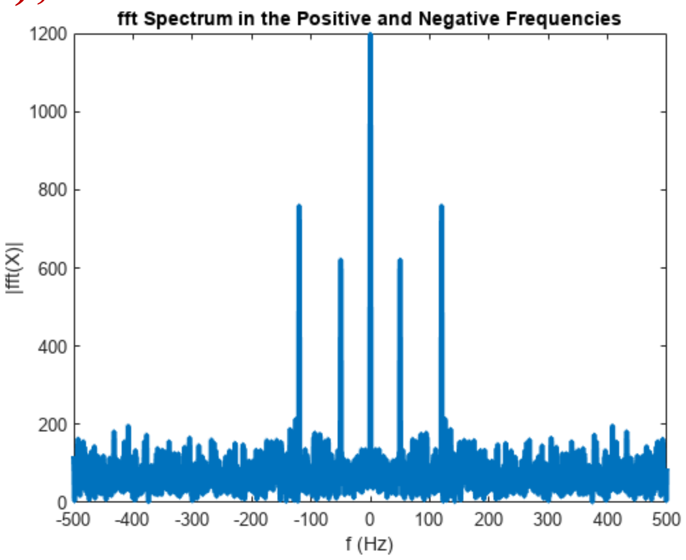
```
Y = fft(X); %
figure;
plot((0:L-1)*fs/L, abs(Y), 'LineWidth',2);
xlabel('f (Hz)');
ylabel('|fft(X)|');
```

Matlab code: FFT编程

fengwang13@gdut.edu.cn

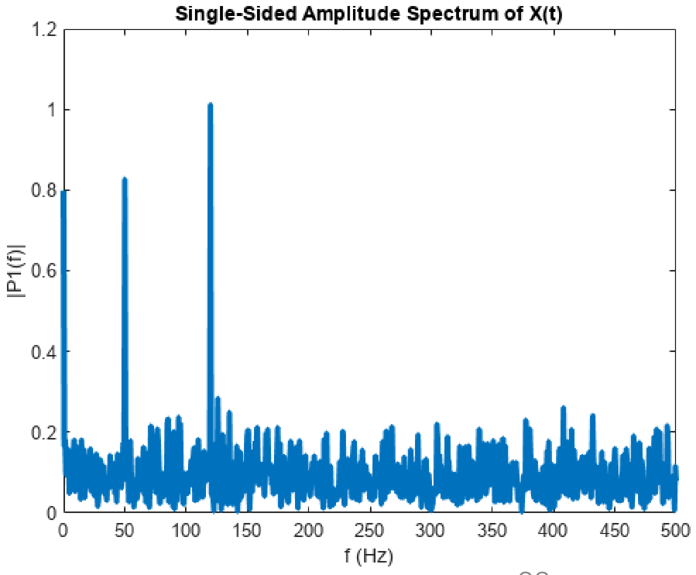
- For real signals, the fft spectrum is a two-sided spectrum.
- To show the fft spectrum in the positive and negative frequencies, use fftshift

```
figure;
plot((-L/2:L/2-1)*fs/L,abs(fftshift(Y)), ‘LineWidth’,2);
xlabel(‘f (Hz)’);
ylabel(‘|fft(X)|’);
```



- To find the amplitudes of the three frequency peaks, convert the fft spectrum in Y to the single-sided amplitude spectrum

```
P2 = abs(Y/L); P1 = P2(1:L/2+1);
P1(2:end-1) = 2*P2(2:end-1);
figure;
plot((0:L/2)*fs/L,P1, ‘LineWidth’,2);
xlabel(‘f (Hz)’);
ylabel(‘|P1(f)|’);
```

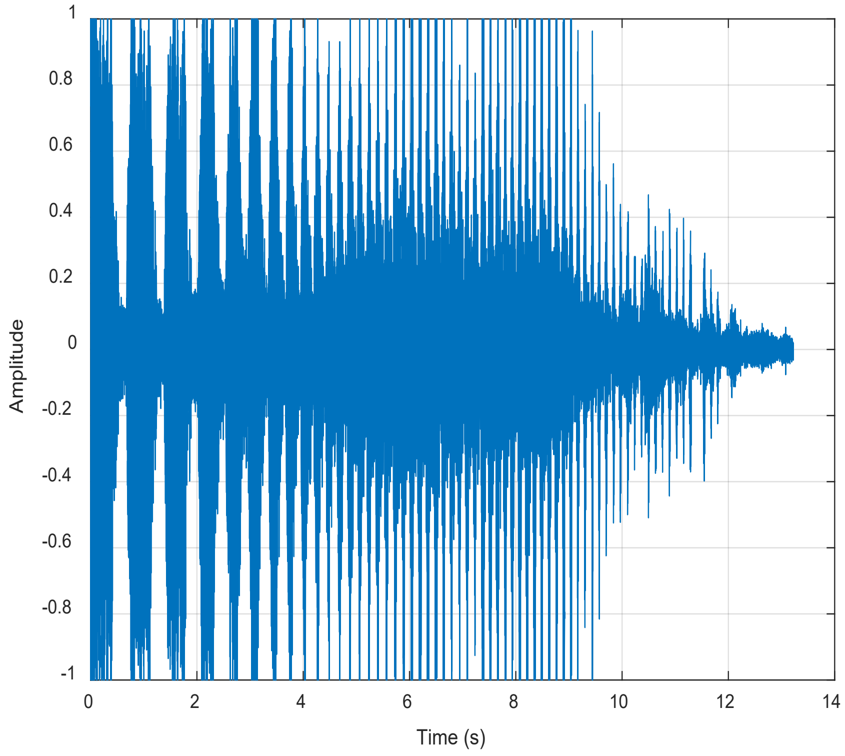


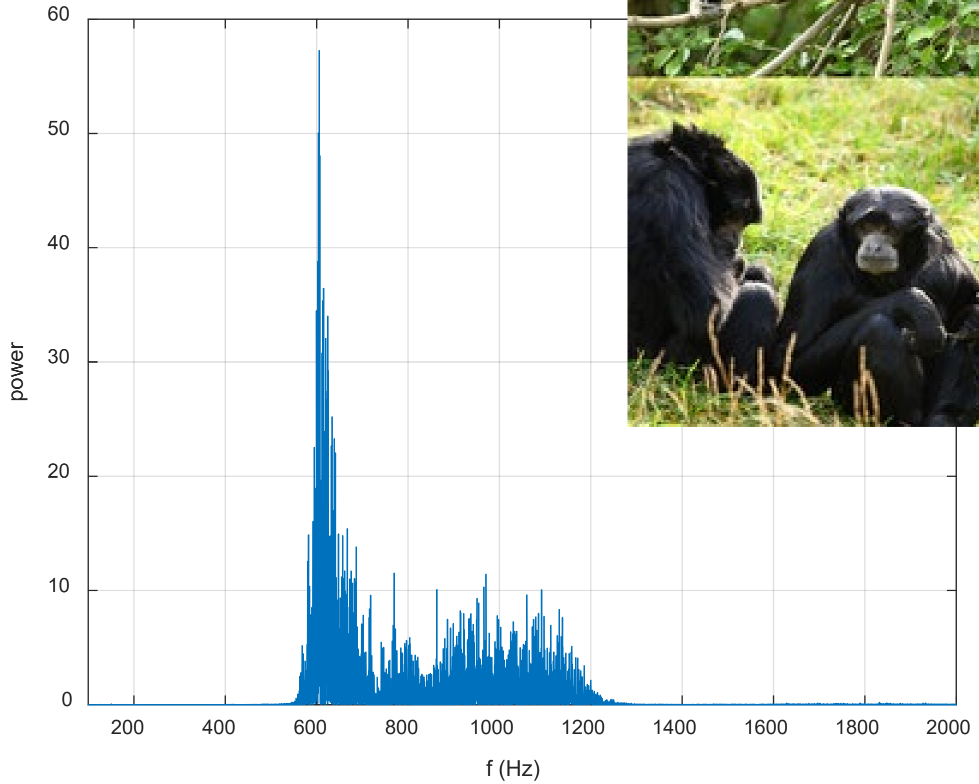
Matlab code: 读取音频文件1，FFT编程

fengwang13@gdut.edu.cn

- abcFile = 'Gibbon-DK_01_clipped.wav';
- [x,fs] = audioread(abcFile);
- sound(x,fs); % listen the audio file

长臂猿(Gibbon)



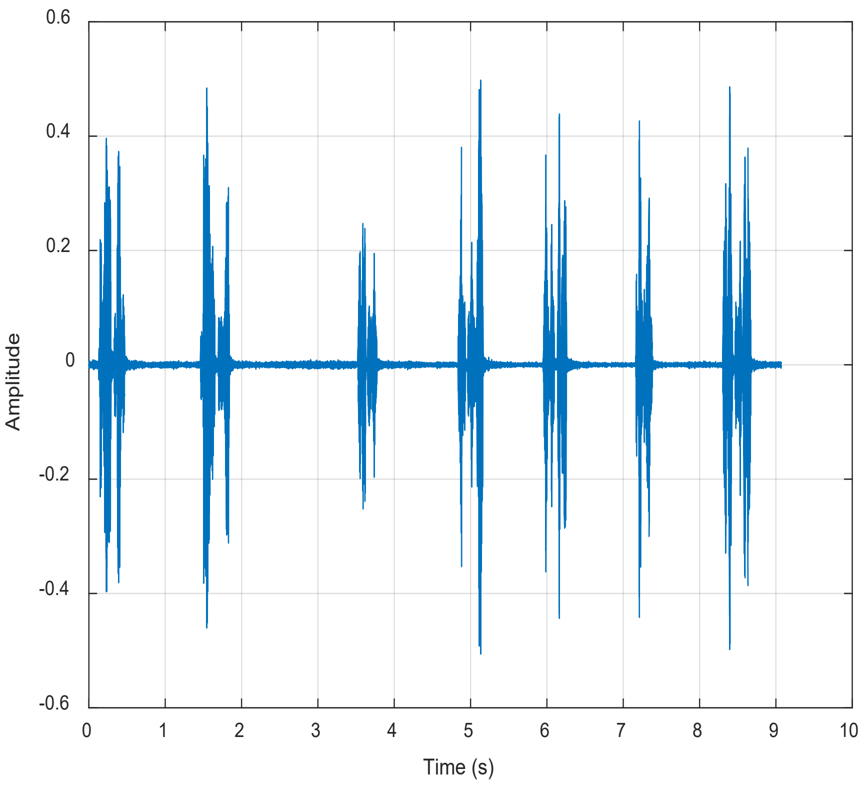


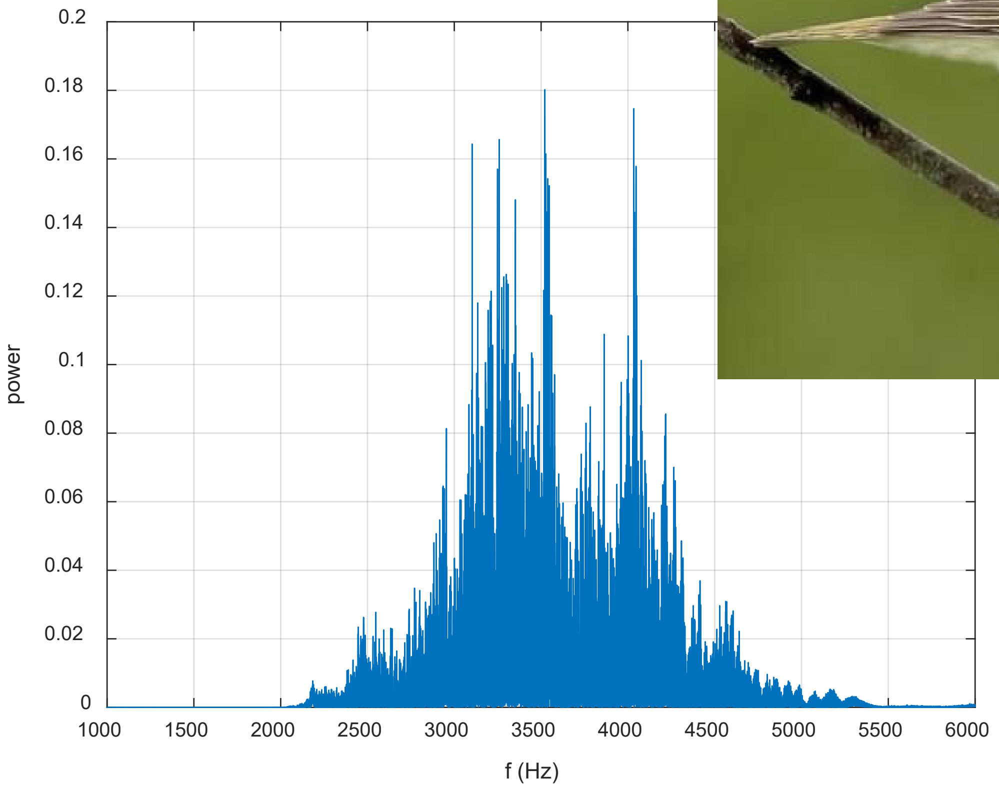
Matlab code: 读取音频文件2，FFT编程

fengwang13@gdut.edu.cn

- abcFile = ‘RedeyedVireo-ML176120.wav’;
- [x,fs] = audioread(VireoFile);
- vireoSing = x(1.46e6:1.66e6);

红眼莺雀
(Redeyed Vireo)





作业

fengwang13@gdut.edu.cn

1. 画出输入自然顺序、输出倒位序的N=8基-2 DIT-FFT流图
2. 画出输入倒位序、输出自然顺序的N=8基-2 DIT-FFT流图
3. 画出输入自然顺序、输出倒位序的N=8基-2 DIF-FFT流图
4. Matlab编程（按照“红眼莺雀”例子，使用FFT）：

（1）读取完整音频文件2(RedeyedVireo-ML176120.wav)，用Matlab编程画出时域波形，画出功率谱图

（2）用手机或电脑自带软件录制一段自己说话的音频文件（可以说一段美食解说词：“阳光和温度，造就美味，更带来多彩的世界。冰消水融，万物复苏，生生不息，光合作用促成植物发育、成熟，不同的积温，滋养初种类繁多的作物”），保存成wav格式，用Matlab编程画出时域波形、画出功率谱。