高等数值算法与应用(7)

– 傅里叶分析 (chap8) –

喻文健

大学生“听音”破解周鸿祎手机号

这名学生展示，他首先通过一则采访视频截取了记者拨打周鸿祎手机时的拨号音，然后用软件将拨号音转换成频谱图，再通过软件放大其中拨号音的部分，获取到了手机号码。

而此事得到了周鸿祎的证实，他在自己的微博里感慨地说“这位同学确实能干”。

创新工场董事长李开复也通过微博喊话：“这位同学，来创新工场吧！你学的是信号处理，不知是否读过我的论文？我有合适项目供你考虑。”

Outline

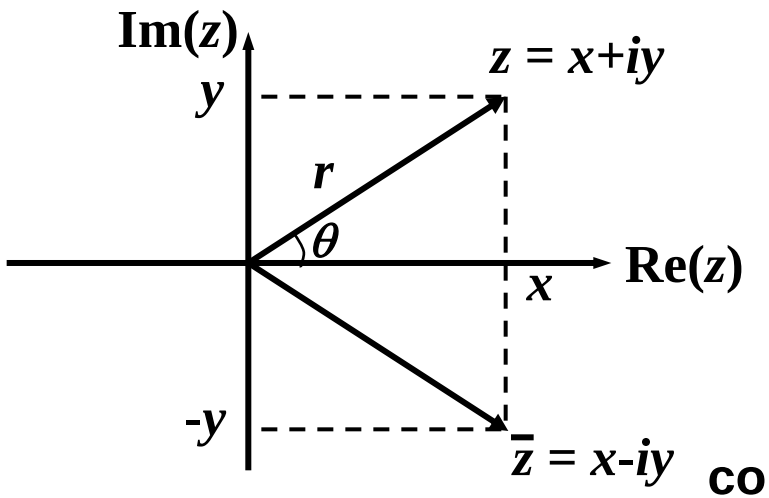
- 离散傅里叶变换 
- FFT的应用—检测周期
- 快速FFT算法

Review of Complex Arithmetic

- 复平面与复数表示

$$z = x + iy$$

imaginary unit $\sqrt{-1}$



complex conjugate

根据欧拉定理： ??

$$e^{i\theta} = \cos\theta + i\sin\theta$$

极坐标形式 $z = re^{i\theta}$

其中 $r = |z|,\ \theta = \arctan(y/x)$

复数的模： $|z| = \sqrt{z\bar{z}} = \sqrt{x^2 + y^2}$ ， $\bar{z} = re^{-i\theta}$

Review of Complex Arithmetic

- **复数的算术运算**

$$z_1 \times z_2 = (x_1 x_2 - y_1 y_2) + i(x_1 y_2 + x_2 y_1)$$

两个复数的+,-,x,/运算需要2到11次实数运算

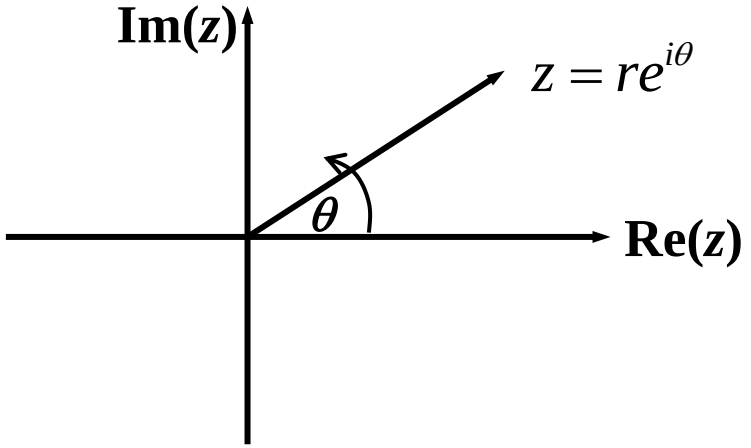
$$z_1 / z_2 = \frac{x_1 x_2 + y_1 y_2}{x_2^{\,2} + y_2^{\,2}} + i\frac{x_2 y_1 - x_1 y_2}{x_2^{\,2} + y_2^{\,2}}$$

极坐标形式 $z = re^{i\theta}$

$$z_1 \times z_2 = r_1 r_2 e^{i(\theta_1 + \theta_2)}$$

$$z_1 / z_2 = (r_1 / r_2) e^{i(\theta_1 - \theta_2)}$$

一个复数乘以 *i*，等价于将其在复平面上逆时针旋转π/2



离散傅里叶变换

- 离散周期函数的三角函数逼近
- 傅里叶变换矩阵
- DFT的性质与演示

离散周期函数的三角函数逼近

- 三角函数适合于逼近有周期性的函数
- 离散周期函数(序列)的三角函数逼近

不妨设复函数 $y(t)$ 的周期为2π, 将[0, 2π]作n等分, 得到序列:

$$y_j = y(2\pi j/n), \quad j = 0,1,...,n-1.$$

2π是其周期

逼近基函数 $\{1, \cos t, \sin t, \cdots, \cos kt, \sin kt, \cdots\}$ 是[0, 2π]上实函数空间的正交函数族,

对于复函数, $\{1, e^{it}, \cdots, e^{i(n-1)t}\}$ 是区间[0, 2π]上的正交函数族. 且

在离散点集 $\{2\pi j/n, \quad j = 0,1,...,n-1\}$ 上, 仍保持正交性.

(复函数I向量内积)

因此, 离散周期函数 $\{y_j\}$ 的最小二乘傅里叶逼近:

离散函数正交基

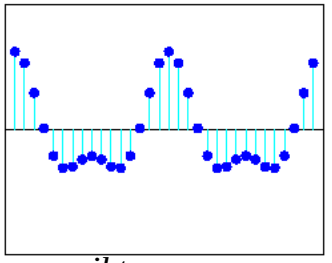
$$y_j \approx S(t_j) = \sum_{k=0}^{n-1} Y_k e^{ikt_j}, \text{ 其中 } Y_k = \frac{< y(t), e^{ikt} >_n}{< e^{ikt}, e^{ikt} >_n} = \frac{1}{n} \sum_{j=0}^{n-1} y_j e^{-ikt_j}, \ k = 0,1,...,n-1$$

离散傅里叶变换(DFT)

对n个点的n阶逼近就是插值, $y_j = S(t_j) = \sum_{k=0}^{n-1} Y_k e^{ikt_j}$

离散傅里叶反变换

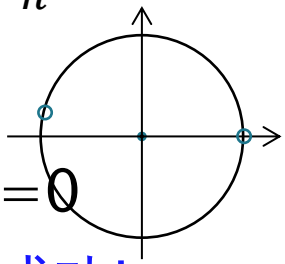
离散周期函数的三角函数逼近



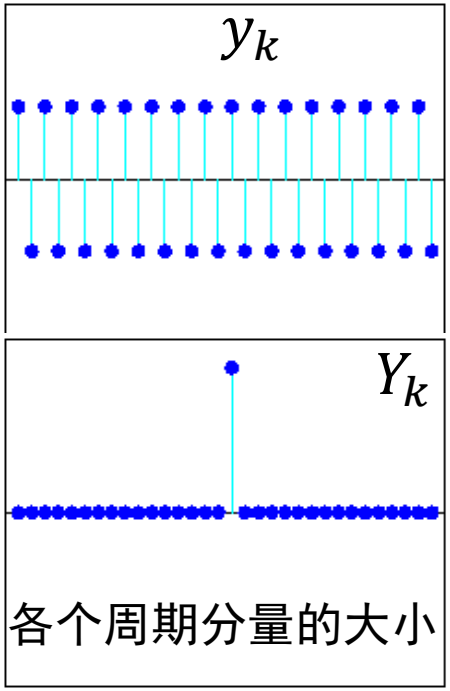
- 一般的周期信号采样
  - 周期为2π的函数用$\{1, e^{it}, \cdots, e^{i(n-1)t}\}$去逼近, $y_j = \sum_{k=0}^{n-1} Y_k e^{ikt_j}$
  - 周期为T的函数, 用$\{1, e^{i\frac{2\pi}{T}t}, \cdots, e^{i\frac{2\pi}{T}(n-1)t}\}$逼近, $y_j = \sum_{k=0}^{n-1} Y_k e^{i\frac{2\pi}{T}kt_j}$
  - 采样n个点: $t_j = jT/n$, $(j=0, \ldots, n-1)$, 采样间隔$\frac{T}{n}$, 采样频率$F_s = \frac{n}{T}$
  - 用DFT算$\{Y_k\}$: $Y_k = \frac{1}{n}\sum_{j=0}^{n-1} y_j e^{-i\frac{2\pi}{T}kt_j} = \frac{1}{n}\sum_{j=0}^{n-1} y_j e^{-i\frac{2\pi}{n}kj}$

例: $y_j = (-1)^j$, $(j=0,\ldots,n-1)$, n为偶数 与T无关!

$\Rightarrow Y_k = \frac{1}{n}\sum_{j=0}^{n-1} (-1)^j e^{-i\frac{2\pi k}{n}j}$ $\quad e^{-i(\frac{2\pi k}{n}-\pi)j}$



- 若k=0, $Y_0=0$; 若k=$n/2$, $Y_{n/2}=1$
- 否则, Y_k是$x^n=1$的n个不同根之和 ,=0
- 验证: $y_j = Y_{n/2} e^{i\frac{2\pi}{n}(n/2)j} = (-1)^j$ 成功!



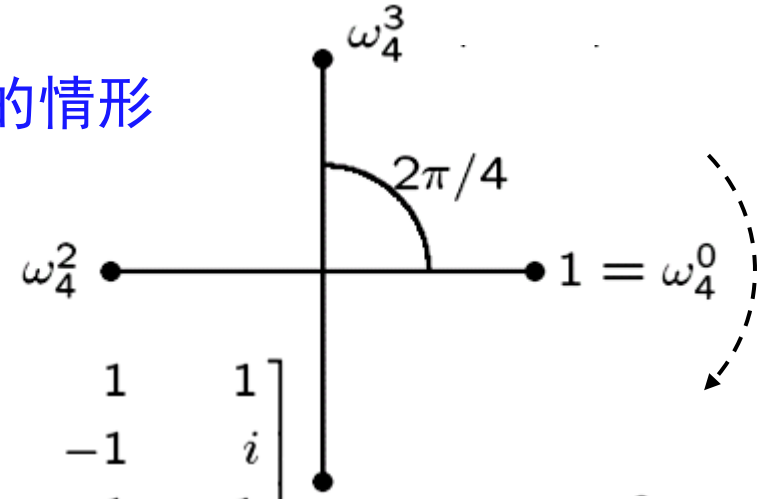
离散傅里叶变换

- 时域信号→ 频域信号 $\{y_k\} \rightarrow \{Y_k\}$ 变换的计算与采样间隔、周期无关

$$Y_k = \frac{1}{n}\sum_{j=0}^{n-1} y_j e^{-i\frac{2\pi}{n}kj}, \quad k = 0, 1, \cdots, n-1$$

  - 舍弃系数$\frac{1}{n}$, 定义离散傅里叶变换DFT $Y_k = \sum_{j=0}^{n-1} y_j \omega^{kj}$
  - $\omega = e^{-i\frac{2\pi}{n}}$ ，是1的n次方根
  - 矩阵公式: $\boxed{Y = Fy}$

n=4的情形



0	0	0	0
0	1	2	3
0	2	4	6
0	3	6	9

$F_{kj} = \omega^{kj}$

$$F_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & \omega^1 & \omega^2 & \omega^3 \\ 1 & \omega^2 & \omega^4 & \omega^6 \\ 1 & \omega^3 & \omega^6 & \omega^9 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{bmatrix}$$

离散傅里叶变换

- 变换矩阵的特点
  - F为对称阵，所有元素模为1
  - 一般地
  - $\boldsymbol{F}^{-1} = \frac{1}{n}\overline{\boldsymbol{F}}$
  - 反变换公式

$\overline{\boldsymbol{F}}$，复共轭

$$\frac{1}{n}\begin{bmatrix}1 & 1 & 1 & 1\\ 1 & \omega^{-1} & \omega^{-2} & \omega^{-3}\\ 1 & \omega^{-2} & \omega^{-4} & \omega^{-6}\\ 1 & \omega^{-3} & \omega^{-6} & \omega^{-9}\end{bmatrix}\begin{bmatrix}1 & 1 & 1 & 1\\ 1 & \omega^{1} & \omega^{2} & \omega^{3}\\ 1 & \omega^{2} & \omega^{4} & \omega^{6}\\ 1 & \omega^{3} & \omega^{6} & \omega^{9}\end{bmatrix}$$

$$Y = Fy \Longrightarrow \boldsymbol{y} = \boldsymbol{F}^{-1}\boldsymbol{Y} = \frac{1}{n}\overline{\boldsymbol{F}}\boldsymbol{Y} = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}$$

$$y_j = \frac{1}{n}\sum_{k=0}^{n-1} Y_k \bar{\omega}^{jk}$$

```
>> omega = exp(-2*pi*i/n);
>> j = 0:n-1;  k = j';
>> F = omega.^(k*j);
```

```
>> F = fft(eye(n))
```

DFT的一些性质

DFT of sequence, even purely real sequence, is in general complex

Components of DFT $\boldsymbol{Y}$ of real sequence $\boldsymbol{y}$ of length n are *conjugate symmetric*: $\boldsymbol{Y}_k$ and $\boldsymbol{Y}_{n-k}$ are complex conjugates for $k = 1, \ldots, (n/2) - 1$

$$\boldsymbol{Y}_k = \sum_{j=0}^{n-1} \boldsymbol{y}_j \boldsymbol{\omega}_n^{kj}$$

$$\boldsymbol{Y}_{n-k} = \sum_{j=0}^{n-1} \boldsymbol{y}_j \boldsymbol{\omega}_n^{(n-k)j}$$

$$\boldsymbol{\omega}_n^{(n-k)j} \cdot \boldsymbol{\omega}_n^{kj} = \boldsymbol{\omega}_n^{nj} = \mathbf{1}$$

共扼, 对任意 $\boldsymbol{k}$

三角函数逼近(频域上)的两个重要分量

- Y_0, 它的值是所有y的和. 也称为直流分量，对应于频率为0的分量(常数基函数)
- n为偶数时, $Y_{n/2}$为实数, 它对应给定采样率下周期达最小值，即T/(n/2),的分量，相应频率最高，为Nyquist频率

$$S(t) = \sum_{k=0}^{n-1} Y_k e^{i2\pi \frac{k}{T} t}$$

$e^{i2\pi \frac{j}{T} t}$与$e^{i2\pi \frac{n-j}{T} t}$在离散点上的值总是共轭的，反映相同的频率特征

看前面那个例子：

Example: DFT

For randomly chosen sequence y

$$F_8 y = F_8 \begin{bmatrix} 4 \\ 0 \\ 3 \\ 6 \\ 2 \\ 9 \\ 6 \\ 5 \end{bmatrix} = \begin{bmatrix} 35 \\ -5.07 + 8.66i \\ -3 + 2i \\ 9.07 + 2.66i \\ -5 \\ 9.07 - 2.66i \\ -3 - 2i \\ -5.07 - 8.66i \end{bmatrix} = Y$$

Transformed sequence is complex, but Y_0 and Y_4 are real, while Y_5, Y_6 and Y_7 are complex conjugates of Y_3, Y_2 and Y_1 respectively

各频率分量的大小不同，但就此例子，它们没有很大的差别

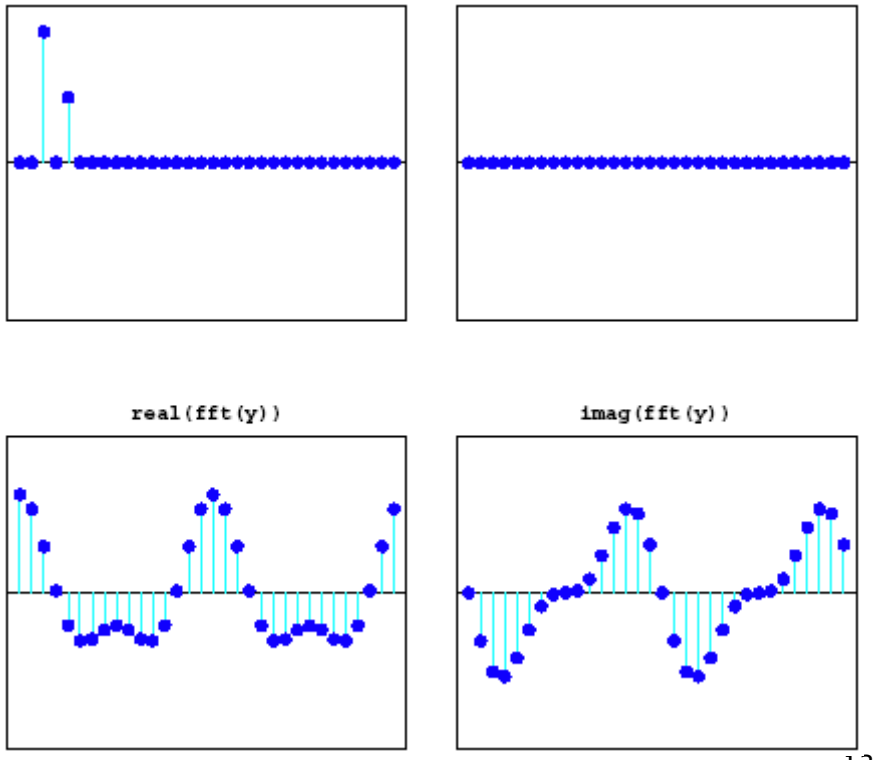
演示程序fftgui

- fftgui, 反映DFT变换前后序列值的对应关系

y_j 换成 Y_j 试试

- $\{y_j\}$仅第1个元素$y_0 \neq 0$
- $\{y_j\}$仅第2个元素$y_1 \neq 0$
- $\{y_j\}$仅第3个元素$y_2 \neq 0$
- $y_2 \neq 0$, $y_4 \neq 0$, 其他均为0
- $\{y_j\}$为实序列，变换结果的对称性
- 中点$y_{n/2} \neq 0$, Nyquist点

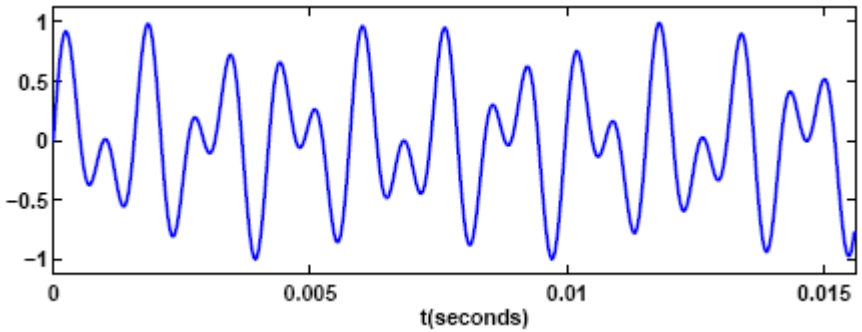
$$Y_k = \sum_{j=0}^{n-1} y_j e^{-2ijk\pi/n}$$



FFT的应用—检测周期

- 电话按键音提取频率
- 演示程序touchtone
- 太阳黑子的周期
- 傅里叶变换矩阵的图

离散傅里叶变换



- 一个音频信号的例子
  - 电话键盘上“1”的波形由两个不同频率sin波形叠加得到
  - 用频率F_s=32768, 即$\frac{n}{T}=\frac{1}{\Delta t}$, 采样波形得到离散序列y
  - 由y的DFT确定这两个频率?

$$y_j = \sum_{k=0}^{n-1} Y_k e^{i\frac{2\pi}{T}kt_j}$$

  - 频谱中的频率间隔为$\frac{1}{T} = \frac{F_s}{n}$
  - 各傅里叶系数对应的频率
  - 另一个例子: $y_j = \sin(2\pi f_1 t_j)$
  - $= \frac{-i}{2}(e^{i2\pi f_1 t_j} - e^{i2\pi(F_s - f_1)t_j})$
  - DFT变换的结果(两个系数$Y_k \neq 0$)

examp1.m 一部分

```
>> pre1sound;
>> plot(t(1:500), y(1:500));
>> sound(y,Fs);
>> p= abs(fft(y));
>> f= (0:n-1)*(Fs/n);
>> plot(f, p, '.-');
>> axis([600 1300 0 1500]);
```

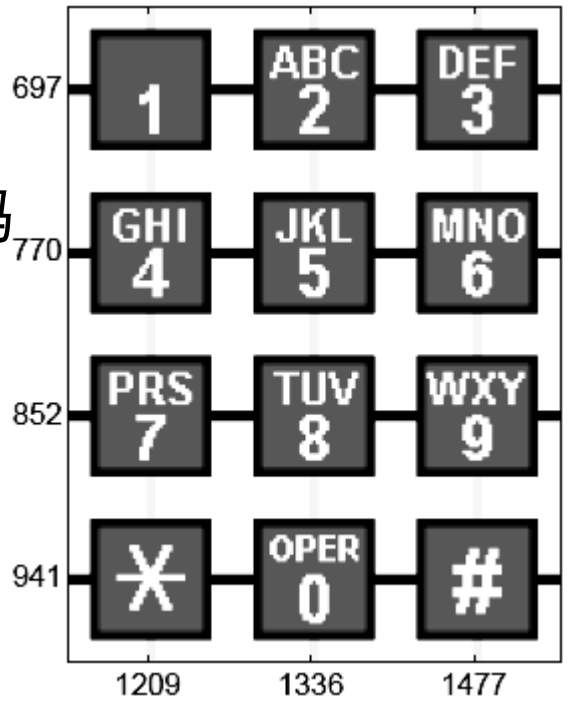
(只看频谱的前一半即可)

得到频率: 695.9, 1207.9

电话按键演示程序touchtone

- 按键式电话——双音多频系统
  - 每个键的声音仅含两个基频
  - 若有按键声音的数据, 可分析出电话号码
  - 分析touchtone.mat中的拨号录音

touchtone演示



demo_ touchtone.m

```
load touchtone          (单字节整数)
yy= double(y.sig)/128;
Fs= y.fs;
n= length(yy);
p= abs(fft(yy(1:n/11)));
n= length(p);
f= (0:n-1)*(Fs/n);
plot(f,p, '.-');
axis([500 1700 0 600])
```

fft需接收double型数据

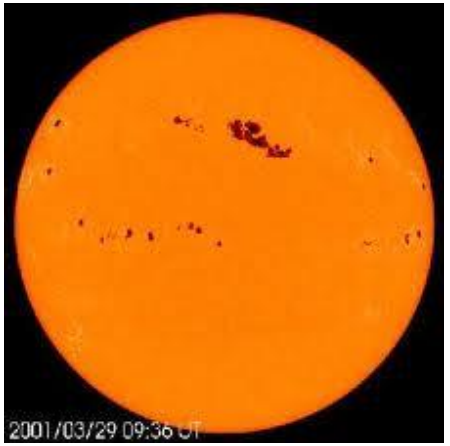
经过试验选取的范围

键1的两个基频: 697, 1209

频谱峰值对应为682, 1210

不吻合与采样率和噪声有关

太阳黑子的周期



- 太阳黑子
  - 太阳表面的深色区域，其活动与天气等有关
  - 天文学家对黑子的数量进行记录(Wolfer黑子数)
  - 根据历史数据，找出每年平均Wolfer黑子数的变化规律
  - sunspot.dat，年份，平均黑子数
  - 去掉增长趋势后用fft作频域分析
  - 频率间隔是1/n
  - 放大后，看到最大频率对应的周期为11.1年
  - 这说明过去300年，黑子爆发的周期稍大于11年

详见sunspotdemo.m

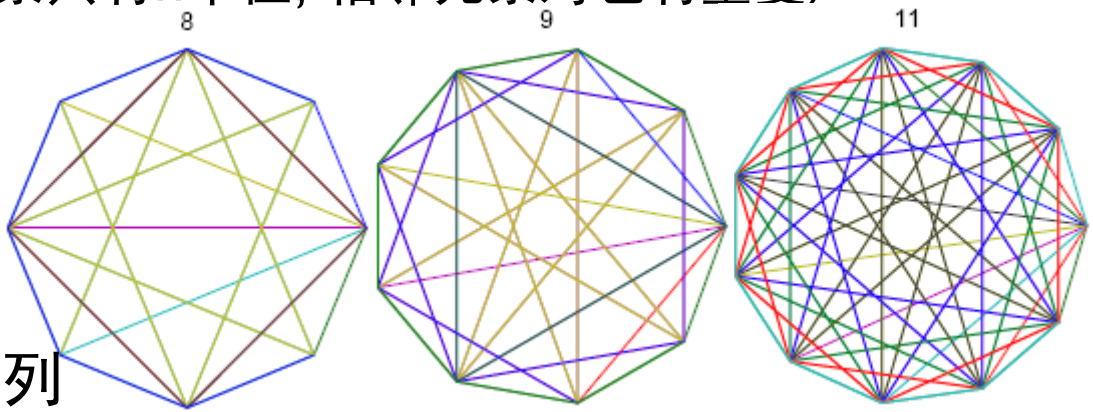
主要命令:

```
[t, wolfer]= sunspot;
n= length(wolfer);
c= polyfit(t, wolfer, 1);
y= wolfer- polyval(c, t);
Y= fft(y);
Fs= 1; f= (0:n/2)*Fs/n;
pow= abs(Y(1:n/2+1));
plot([f; f], [zeros(1, n/2+1); pow']);
```

傅里叶变换矩阵的图

- DFT变换矩阵的图形显示
  - plot(A), 将按矩阵的列逐列绘制曲线，得到n条折线
  - 若数据为复数，则plot自动将其实部做横轴，虚部做纵轴
  - 看看plot(F)的结果
  - n为质数时, 得到完全图；
    否则图是稀疏的, 它预示着快速算法的可能性 (矩阵元素只有n个值, 相邻元素对也有重复)
  - 用演示程序fftmatrix研究这些图
  - fftmatrix(n)画所有列 (改了一个bug)
  - fftmtrix(n,j)画第j+1列

```
>> F= fft(eye(5));
>> plot(F);
>> axis equal
```



快速FFT算法

Idea of Fast DFT (FFT)

$$Y = Fy$$

$$F_{kj} = \omega_n^{kj}$$

ω_n是1的n次方根

设$\alpha = \omega_n^k,\ 0 \le k \le n-1$

$$Y_k = \sum_{j=0}^{n-1} y_j \omega_n^{kj} = y_0 + y_1\alpha + y_2\alpha^2 + \cdots + y_{n-1}\alpha^{n-1}$$

$$= (y_0 + y_2\alpha^2 + y_4\alpha^4 + \cdots) + \alpha(y_1 + y_3\alpha^2 + y_5\alpha^4 + \cdots)$$

$$= DFT_{n/2}(y_{even})_k + \alpha \cdot DFT_{n/2}(y_{odd})_k$$

$DFT_n(y)_k$

设n为偶数，注意到旋转因子ω_n有下述性质：

$$\alpha = \omega_n^k = e^{-2k\pi i/n},\quad \alpha^2 = e^{-2k\pi i/(n/2)} = \omega_{n/2}^k$$

For k=0, …, n/2-1,
转换为n/2阶的DFT计算

当$k \ge n/2$时，怎么办？

注意到$\alpha^2 = \omega_n^{2k} = \omega_n^n \cdot \omega_n^{2(k-n/2)}$ $\xrightarrow{k' = k - \frac{n}{2} < \frac{n}{2}}$ $= \omega_n^{2k'}$

仍用那两个n/2阶DFT变换结果, 即

$$DFT_{n/2}(y_{even})_{k'} + \alpha \cdot DFT_{n/2}(y_{odd})_{k'}$$

可再简化

1

$\forall\ \alpha = \omega_n^k = \omega_n^{n/2} \cdot \omega_n^{k-n/2} = -\omega_n^{k'}$,

FFT Algorithm

```
procedure fft(x, y, n, ω)
    if n = 1 then
        y[0] = x[0]
    else
        for k = 0 to (n/2) − 1
            p[k] = x[2k]
            s[k] = x[2k + 1]
        end
        fft(p, q, n/2, ω^2)
        fft(s, t, n/2, ω^2)
        for k = 0 to n − 1
            y[k] = q[k mod (n/2)] +
                   ω^k t[k mod (n/2)]
        end
    end
```

计算量分析：

$$
\begin{aligned}
C(n) &= 2C(\frac{n}{2}) + \frac{5n}{2} \\
&= 4C(\frac{n}{4}) + 2 \cdot \frac{5n}{2} \\
&= 8C(\frac{n}{8}) + 3 \cdot \frac{5n}{2} \\
&= \cdots \\
&= \frac{5n}{2}\log_2 n \quad < \quad n^2
\end{aligned}
$$

进一步简化

$$
\left.
\begin{aligned}
w &= \left[\omega^0, \cdots, \omega^{n/2-1}\right]^T \\
y &= \begin{bmatrix} q + w.*t \\ q - w.*t \end{bmatrix}
\end{aligned}
\right\}
$$

$\because\ \omega_n^k = \omega_n^{n/2} \cdot \omega_n^{k-n/2} = -\omega_n^{k'}$

FFT算法说明

- 考虑的是理想情况，$n=2^k$
- 事先算出n/2个旋转因子
- 若仔细考虑，可不增加额外存储量 (原地工作)
- 若输入序列x为实数序列，可使计算/存储量减半
- 在某些算法实现中，输出并非按正常的顺序，可有两种处理方法：在进行反FFT变换时把顺序调整回来；将输出再进行排序（也是nlogn的运算量）
- 实际程序中将递归变为循环，见netlib/fftpack

  FFTW: C语言免费程序，曾获1999年Wilkinson数值软件奖 (http://www.fftw.org/)

$\mathcal{O}(n^2)$与 $\mathcal{O}(n\log_2 n)$ 的区别

Use of FFT algorithm reduces work to only $\mathcal{O}(n \log_2 n)$, which makes enormous practical difference in time required to transform large sequences

n	$n \log_2 n$	n^2
64	384	4096
128	896	16384
256	2048	65536
512	4608	262144
1024	10240	1048576

反变换也有类似的FFT算法

$$y_j = \frac{1}{n} \sum_{k=0}^{n-1} Y_k \bar{\omega}^{jk}, \quad j = 0, 1, \ldots, n-1$$

Matlab: ifft

Limitations of FFT

FFT algorithm is not always applicable or maximally efficient

Input sequence assumed to be:

- Equally spaced
- Periodic
- Power of two in length

（频域分析的含义）

First two of these follow from definition of DFT, while third is required for maximal efficiency of FFT algorithm

Care must be taken in applying FFT algorithm to produce most meaningful results as efficiently as possible

For example, transforming sequence that is not really periodic or padding sequence to make its length power of two may introduce spurious noise and complicate interpretation of results

铺垫

伪的

Mixed-Radix FFT

混合基数FFT

It is possible to define "mixed-radix" FFT algorithm that does not require number of points n to be power of two

More general algorithm is still based on divide-and-conquer; sequence is not necessarily split exactly in half at each level, but by smallest prime factor of remaining sequence length

质数因子

Efficiency depends on whether n is product of small primes (ideally power of two)

If not, then much of computational advantage of FFT may be lost

For example, if n itself is prime, then sequence cannot be split at all, and "fast" algorithm becomes standard $\mathcal{O}(n^2)$ matrix-vector multiplication