Z.t 前记：虽然我是通信工程（“通信”不是“通讯”……）毕业的，信号与系统是必修的主干课程，但是我一直无法理解 Fourier Transform，更不用说 FFT 了：（ 毕业快 10 年了，偶然间又想起来，所以翻箱倒柜的看，希望在有生之年能够弄清楚这些东西。
就是这样。

1702 - Using FFT to Obtain Simple Spectral Analysis Plots

使用 FFT 取得频谱分析图

提问:

如何使用 FFT 函数绘制频率-功率关系图？

回答:

我们使用 200Hz 的正弦曲线作为例子。假定你要处理的数据在数列 x 中。

```
% 产生一个频率，共有 1024 个采样点
Fs = 1024;


% 持续时间是 1 秒
t = 0:1/Fs:1;


%创建 200Hz 的正弦曲线
x =sin(2*pi*t*200);
```

第一步，你需要调用 FFT 函数。为了让 FFT 达到最快，需要在数据序列中插入足够多的 0，使得数据长度等于 2 的幂次。你可以在 FFT 的第二个参数指定参加 FFT 运算的序列长度，如果实际长度不足内置的 FFT 函数会自动使用 0 填充，具体做法如下：

```
% nextpow2（n）是求大于等于 n 的 2 的最小幂次. 数学语言描述就是：求最小的 p,满足 2^p >= abs(A)比如：nextpow2(1000)=10，因为靠近 1024。从 513 到 1024 的整数,，nextpow2(n) 都是 10.
```

```
nfft = 2^(nextpow2(length(x)));
```

%调用 fft 函数，函数会自动填充 x 序列，使它的长度等于 nnf

```
fftx = fft(x,nfft);
```

如果 nfft 是偶数（看过上面的操作你会知道一定是偶数），通过 fft 的对称性我们可以知道运算结果中最开始的(1+nfft/2)个点是独立的，序列其余值是冗余的（会出现镜像现象，所以需要 cut 一部分）。x 序列的直流分量是 fftx(1)，同时 fftx(1+nfft/2)>1 是 x 序列的奈奎斯特频率分量。如果 nfft 是奇数，奈奎斯特频率分量没有计算，有意义的点是（nfft+1）/2。对于上面 2 种情况，都可以用 ceil((nfft+1)/2)进行处理。

```
% 计算独立点

NumUniquePts = ceil((nfft+1)/2);

% FFT 是对称的，抛弃一半

fftx = fftx(1:NumUniquePts);
```

下一步，计算 fft 的幅度:

```
% 计算 x fft 结果的幅度

mx = abs(fftx);
```

需要注意，MATLAB 不会根据输入的长度剪裁输出的 fft 序列的长度:

```
% 处理 fft 结果，使得它和 x 长度无关（Z.t:我不懂）
mx = mx/length(x);


% 将序列 x 经过 fft 运算之后的结果取平方
mx = mx.^2;


% 因为前面去掉了 FFT 的一半数值，所以我们需要将 mx 乘以 2 来保证能量不变
% 如果存在直流分量和奈奎斯特分量，它们是独立的，不能乘以 2

if rem(nfft, 2) % 偶数的 nfft 不包括奈奎斯特点
  mx(2:end) = mx(2:end)*2;
else
```

```
  mx(2:end -1) = mx(2:end -1)*2;
end
```

现在可以创建频率向量了：

```
% This is an evenly spaced frequency vector with NumUniquePts points.


f = (0:NumUniquePts-1)*Fs/nfft;
```

最后，绘制标题和坐标名称

```
% Generate the plot, title and labels.
plot(f,mx);
title('Power Spectrum of a 200Hz Sine Wave');
xlabel('Frequency (Hz)');
ylabel('Power');
```

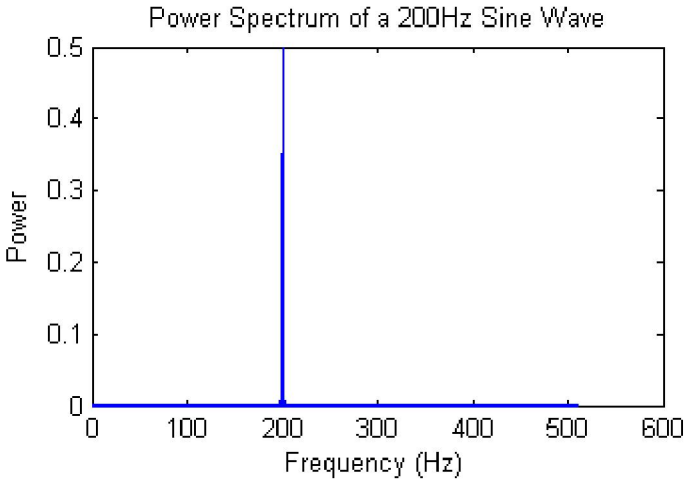
上面的动作集成在一起，就是下面的 MATLAB 文件：

```
% Sampling frequency

Fs = 1024;


% Time vector of 1 second

t = 0:1/Fs:1;


% Create a sine wave of 200 Hz.

x = sin(2*pi*t*200);


% Use next highest power of 2 greater than or equal to length(x) to
calculate FFT.
nfft= 2^(nextpow2(length(x)));


% Take fft, padding with zeros so that length(fftx) is equal to nfft

fftx = fft(x,nfft);
```

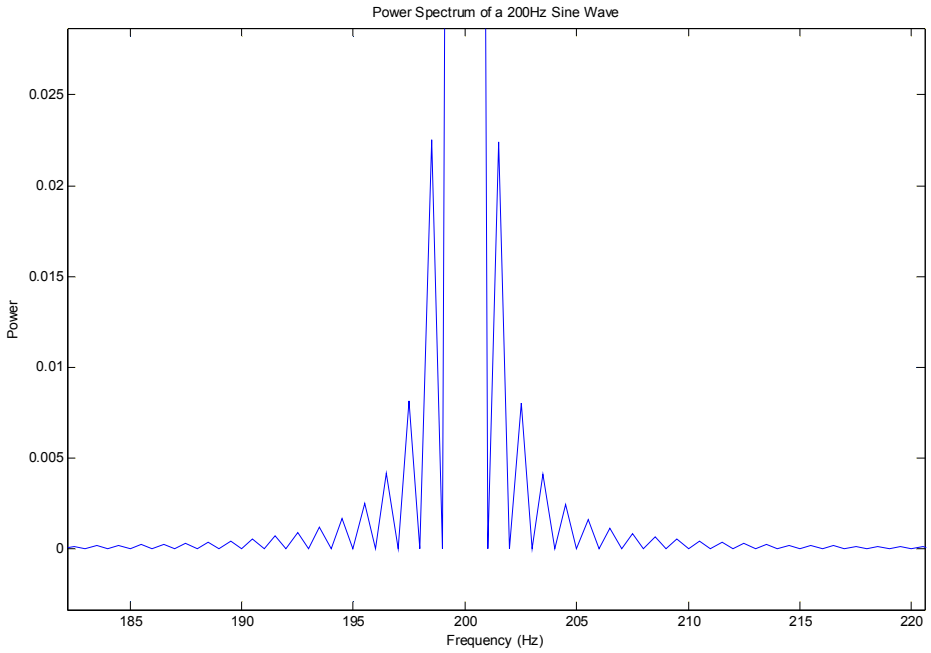
```
% Calculate the numberof unique points
NumUniquePts = ceil((nfft+1)/2);


% FFT is symmetric, throw away second half

fftx = fftx(1:NumUniquePts);


% Take the magnitude of fft of x and scale the fft so that it is not a
function of the length of x

mx = abs(fftx)/length(x);


% Take the square of the magnitude of fft of x.

mx = mx.^2;


% Since we dropped half the FFT, we multiply mx by 2 to keep the same energy.
% The DC component and Nyquist component, if it exists, are unique and
should not be multiplied by 2.


if rem(nfft, 2) % odd nfft excludes Nyquist point
  mx(2:end) = mx(2:end)*2;
else
  mx(2:end -1) = mx(2:end -1)*2;
end


% This is an evenly spaced frequency vector with NumUniquePts points.

f = (0:NumUniquePts-1)*Fs/nfft;


% Generate the plot, title and labels.

plot(f,mx);
title('Power Spectrum of a 200Hz Sine Wave');
xlabel('Frequency (Hz)');
```

ylabel('Power');

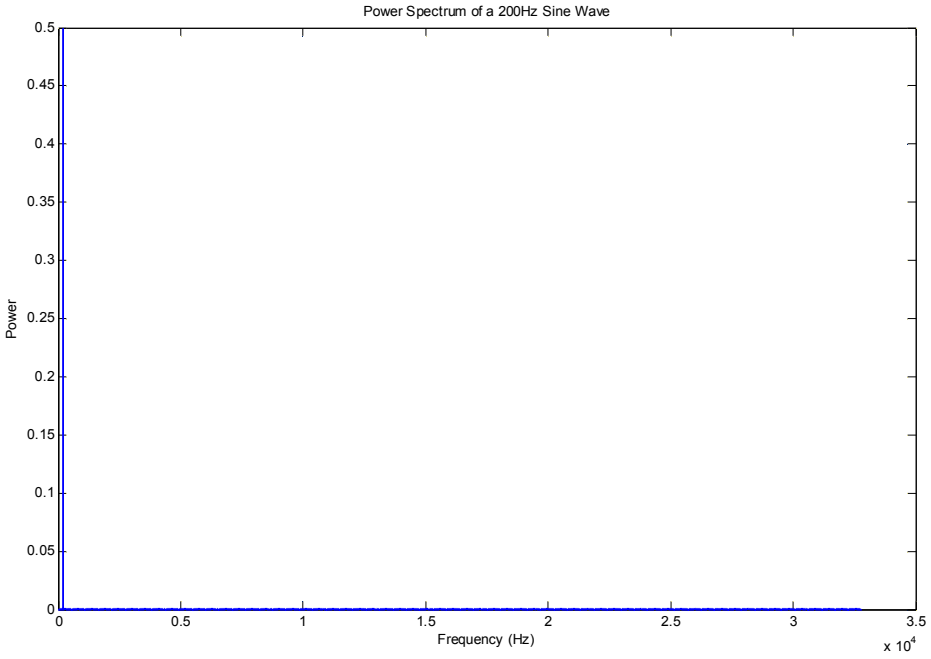
The resulting plot looks like the following:



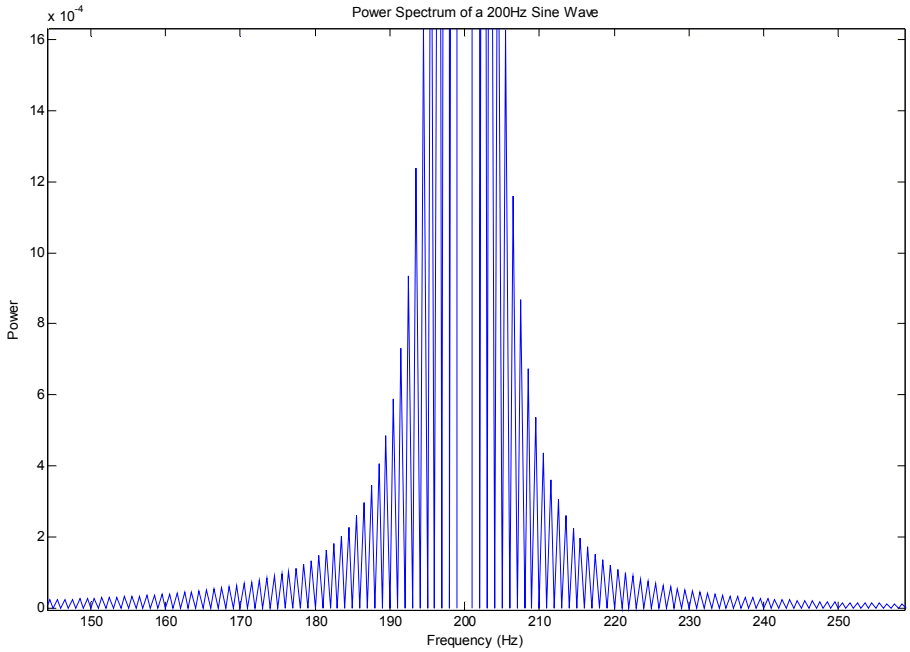
放大一些观察，发现 200 附近很不平滑……不知道这又是那个定理？



尝试改变 Fs = 65536； 增加采样点，运行结果如下：



再放大 200Hz 处



原文在：

http://www.mathworks.com/support/tech-notes/1700/1702.html 内容如下：

1702 - Using FFT to Obtain Simple Spectral Analysis Plots

Question:

How can you correctly scale the output of the FFT function to obtain a meaningful power versus frequency plot?

Answer:

Assume x is a vector containing your data. A sample vector used with this technical note is a 200 Hz sinusoid signal.

```
% Sampling frequency
Fs = 1024;


% Time vector of 1 second
t = 0:1/Fs:1;


% Create a sine wave of 200 Hz.
x =sin(2*pi*t*200);
```

First, you need to call the FFT function. For the fastest possible ffts, you will want to pad your data with enough zeros to make its length a power of 2. The built-in FFT function does this for you automatically, if you give a second argument specifying the overall length of the fft, as demonstrated below:

```
% Use next highest power of 2 greater than or equal to length(x) to calculate fft

nfft = 2^(nextpow2(length(x)));


% Take fft, padding with zeros so that length(fftx) is equal to nfft

fftx = fft(x,nfft);
```

If nfft is even (which it will be, if you use the above two commands above), then the magnitude of the fft will be symmetric, such that the first (1+nfft/2) points are unique, and the rest are symmetrically redundant. The DC component of x is fftx(1) , and fftx(1+nfft/2)> is the Nyquist frequency component of x. If nfft is odd, however, the Nyquist frequency component is not evaluated, and the number of unique points is (nfft+1)/2 . This can be generalized for both cases to ceil((nfft+1)/2) .

```
% Calculate the number of unique points

NumUniquePts = ceil((nfft+1)/2);

% FFT is symmetric, throw away second half

fftx = fftx(1:NumUniquePts);
```

Next, calculate the magnitude of the fft:

```
% Take the magnitude of fft of x

mx = abs(fftx);
```

Consider the fact that MATLAB does not scale the output of fft by the length of the input:

```
% Scale the fft so that it is not a function of the length of x

mx = mx/length(x);


% Now, take the square of the magnitude of fft of x which has been scaled properly.
mx = mx.^2;


% Since we dropped half the FFT, we multiply mx by 2 to keep the same energy.
% The DC component and Nyquist component, if it exists, are unique and should not be multiplied by 2.

if rem(nfft, 2) % odd nfft excludes Nyquist point
  mx(2:end) = mx(2:end)*2;
else
  mx(2:end -1) = mx(2:end -1)*2;
end
```

Now, create a frequency vector:

```
% This is an evenly spaced frequency vector with NumUniquePts points.


f = (0:NumUniquePts-1)*Fs/nfft;
```

Finally, generate the plot with a title and axis labels.

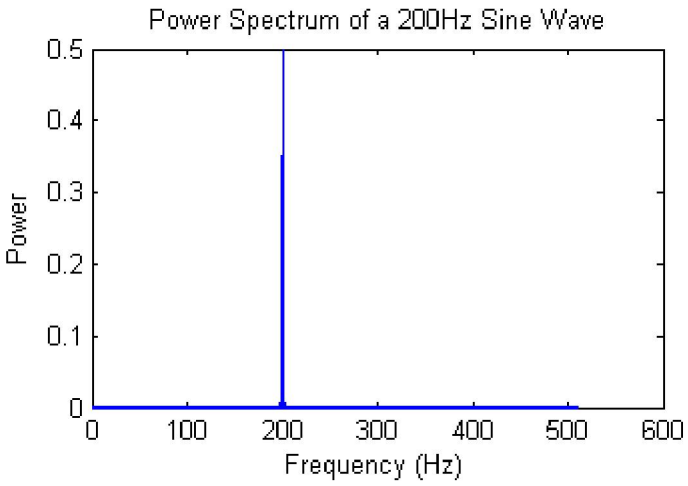
```
% Generate the plot, title and labels.
plot(f,mx);
title('Power Spectrum of a 200Hz Sine Wave');
xlabel('Frequency (Hz)');
ylabel('Power');
```

Bringing this all together, you get the following MATLAB file:

```
% Sampling frequency

Fs = 1024;


% Time vector of 1 second

t = 0:1/Fs:1;


% Create a sine wave of 200 Hz.

x = sin(2*pi*t*200);


% Use next highest power of 2 greater than or equal to length(x) to
calculate FFT.
nfft= 2^(nextpow2(length(x)));


% Take fft, padding with zeros so that length(fftx) is equal to nfft

fftx = fft(x,nfft);


% Calculate the numberof unique points
```

```
NumUniquePts = ceil((nfft+1)/2);


% FFT is symmetric, throw away second half

fftx = fftx(1:NumUniquePts);


% Take the magnitude of fft of x and scale the fft so that it is not a
function of the length of x

mx = abs(fftx)/length(x);


% Take the square of the magnitude of fft of x.

mx = mx.^2;


% Since we dropped half the FFT, we multiply mx by 2 to keep the same energy.
% The DC component and Nyquist component, if it exists, are unique and
should not be multiplied by 2.


if rem(nfft, 2) % odd nfft excludes Nyquist point
  mx(2:end) = mx(2:end)*2;
else
  mx(2:end -1) = mx(2:end -1)*2;
end


% This is an evenly spaced frequency vector with NumUniquePts points.

f = (0:NumUniquePts-1)*Fs/nfft;


% Generate the plot, title and labels.

plot(f,mx);
title('Power Spectrum of a 200Hz Sine Wave');
xlabel('Frequency (Hz)');
ylabel('Power');
```

The resulting plot looks like the following:



By Zoologist@ www.lab-z.com

2011-3-6