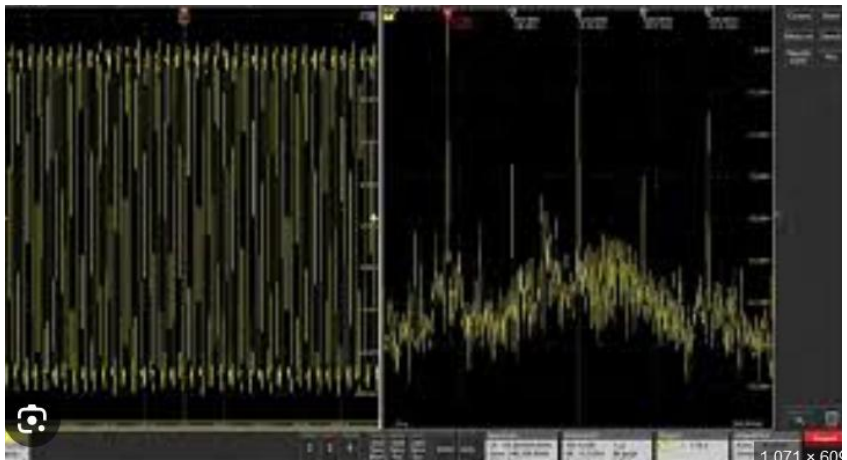
今日の講義の目的

将来現場にでたり, 解析して簡単なようでEXCELではし難いデータ処理や, 制御モデルを, さっとできるようになれば, 強いエンジニアとして重宝されます.
理論を頭でわかっていても実際に使える人は少ない

PID制御, FFTはよく使うので, これを扱います

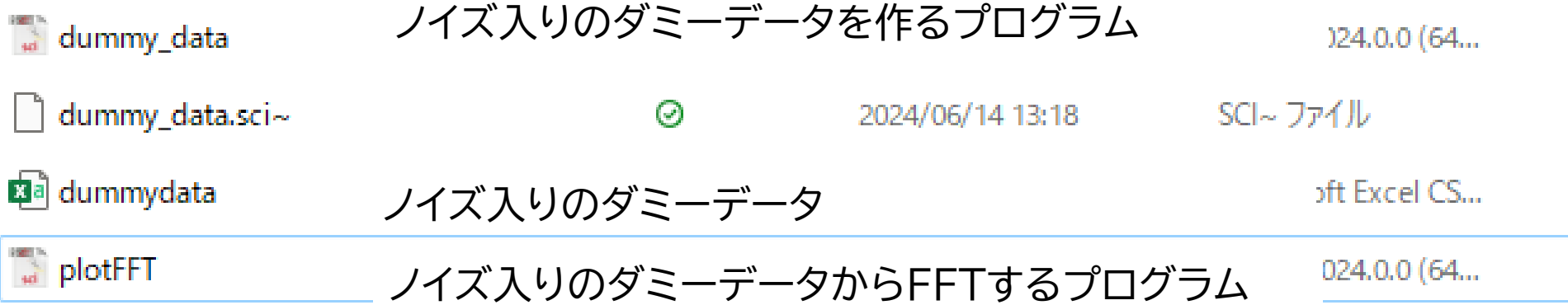
少なくとも人に頼むにおいても, 機器を買うにしても, 何をやっているはわかってほしい



EMDj
Electrical Machines and Drives

FFTをやってみよう

本日配布のFFTのフォルダ



FFTの原理(正確にはDFT)

DFTはデータにCOS, SINをかけて, 足し算をしてるだけです..

$$F(k) = \frac{1}{N}\sum_{n=0}^{N-1} f(n)e^{-i\frac{2\pi k}{N}n}$$

調べたいデータ

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

$$\sin\alpha\cos\beta = \frac{1}{2}\{\sin(\alpha+\beta) + \sin(\alpha-\beta)\}$$

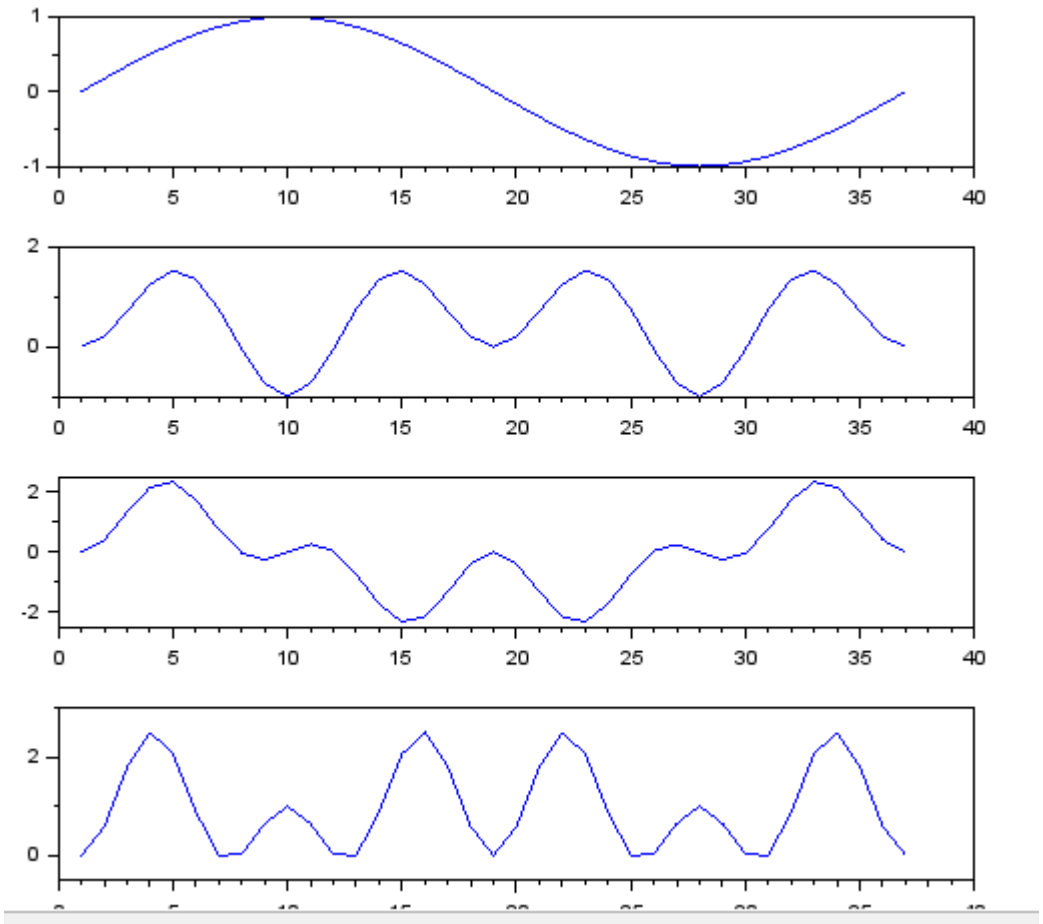
$$\sin\alpha\sin\beta = -\frac{1}{2}\{\cos(\alpha+\beta) - \cos(\alpha-\beta)\}$$

$\alpha = \beta$ のときだけ COS(0)になって値を持つ
それ以外は, 足すとゼロになる
例えば
あるデータ列に, sin(5Θ)を掛けたら
データの5Θの成分だけが, 値を持つ

ともかく, やってみる　just do it !

```
deg=[0:10:360];
thita = deg ./180 *%pi;

data = sin(thita) + 2 *sin(3 * thita);

a= sin(thita);
b= sin(2 * thita);
c= sin(3 * thita);

mean(a .* data)
mean(b .* data)
mean(c .* data)

clf;
subplot(4,1,1)
plot(sin(thita));

subplot(4,1,2)
plot(a .* data);

subplot(4,1,3)
plot(b .* data);

subplot(4,1,4)
plot(c .* data);
```

積分して0になるときと, ならないときが あるよね？



ノイズ入りのデータつくってみます

```
// パラメータの設定
Fs = 1000; // サンプリング周波数 (Hz)
Ts =1 ;//データ全体の実時間（秒）
t = 0:1/Fs:Ts; // Δt （データ点間の示す時間間隔）
f = 50; // 正弦波の周波数 (Hz)
A = 1; // 正弦波の振幅

// 減衰パラメータ
decay_rate = 7; // 減衰率

// 正弦波の生成
x =1...
+ 1.0 *A * exp(-decay_rate*t) .* sin(2*%pi*f*t)...
+ 3.0 * A * exp(-decay_rate*t) .* sin(6*%pi*f*t)...
+ 0.5 * A * exp(-decay_rate*t) .* sin(10*%pi*f*t);

// ノイズの生成
noise_power = 1; // ノイズの強度
noise = sqrt(noise_power) * rand(1, length(t)); // 正規分布に従うノイズ

// 信号とノイズの合成
x_with_noise = x + noise;
```

50Hz.300Hz,500Hz
の混ざった波形になる

データをCSVで吐き出しておきます(準備)

```
//プロット
clf();
subplot(2,1,1)
plot(t,x)
xlabel('Time (s)')
ylabel('Amplitude')
title('Original Signal')

subplot(2,1,2)
plot(t,x_with_noise)
xlabel('Time (s)')
ylabel('Amplitude')
title('Signal with Noise')

X = x_with_noise;
csvWrite(X,"c:¥work¥dummydata.csv")
```

ここは適当なフォルダ作ってデータ格納してください(後で使います)

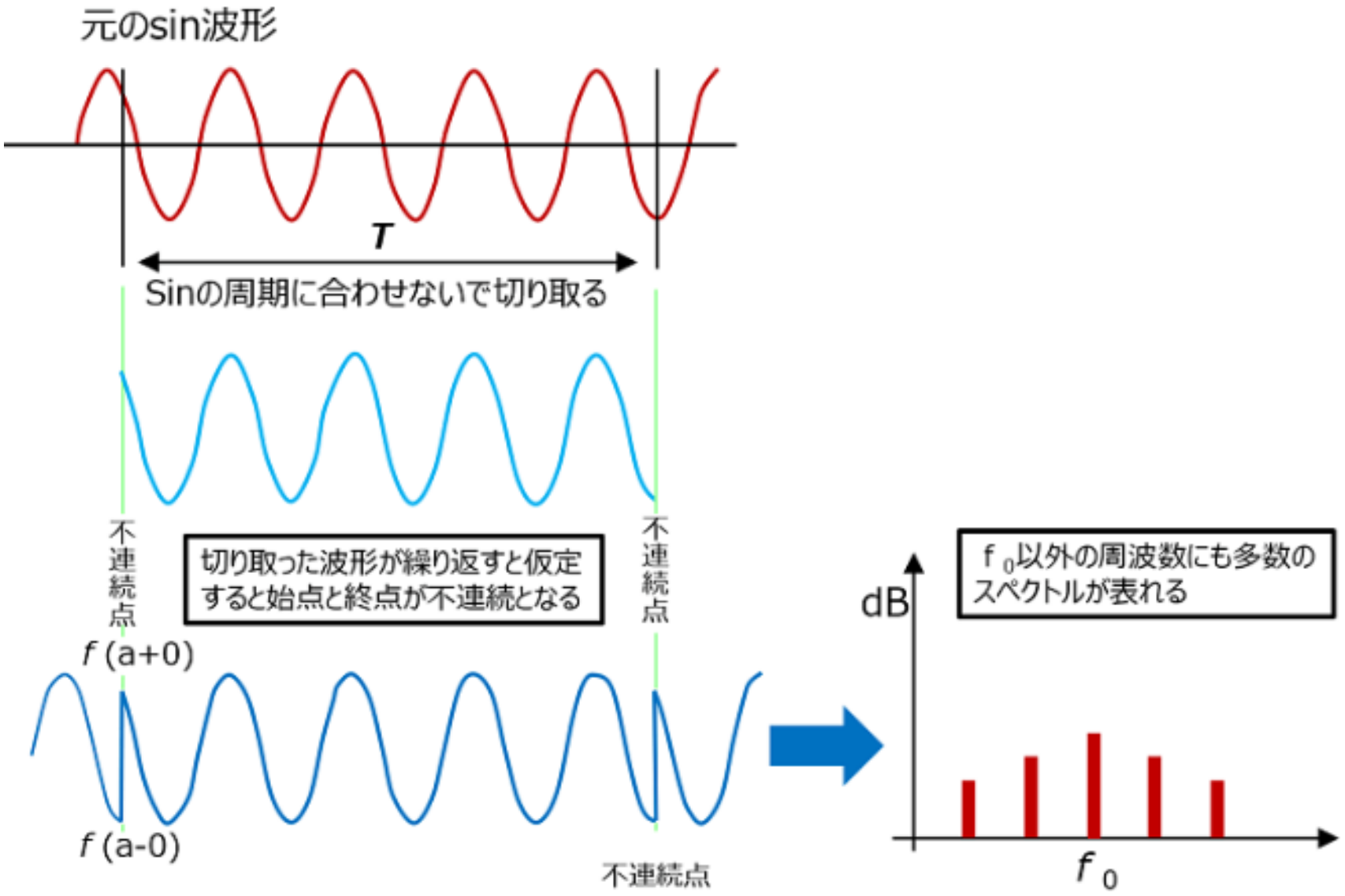
FFTのプログラム

データ格納した
フォルダに書き換え

```
Wkdir="c:¥work¥dummydata.csv" //データ格納するディレクトリここで定義してくだ
Fs = 1000; // サンプリング周波数（仮定）
wav = csvRead(Wkdir);//データ格納するディレクトリここで定義してください//
opt=0; //窓関数 ハミングなら 1
N = size(wav, '*');
    regN = 2/N;
    select opt
    case 1 then
        winf = window('hm',N);
        regF = 2;
    else
        winf = ones(1,N);
        regF = 1;
    end
    y = fft(wav .* winf);
    xf = Fs*(0:(N/2))/N; //associated frequency vector
    nf = size(xf, '*');
    clf();
    plot2d(xf(2:nf), abs(y(2:nf)) * regN * regF);
    xtitle('FFT', '[Hz]');
```

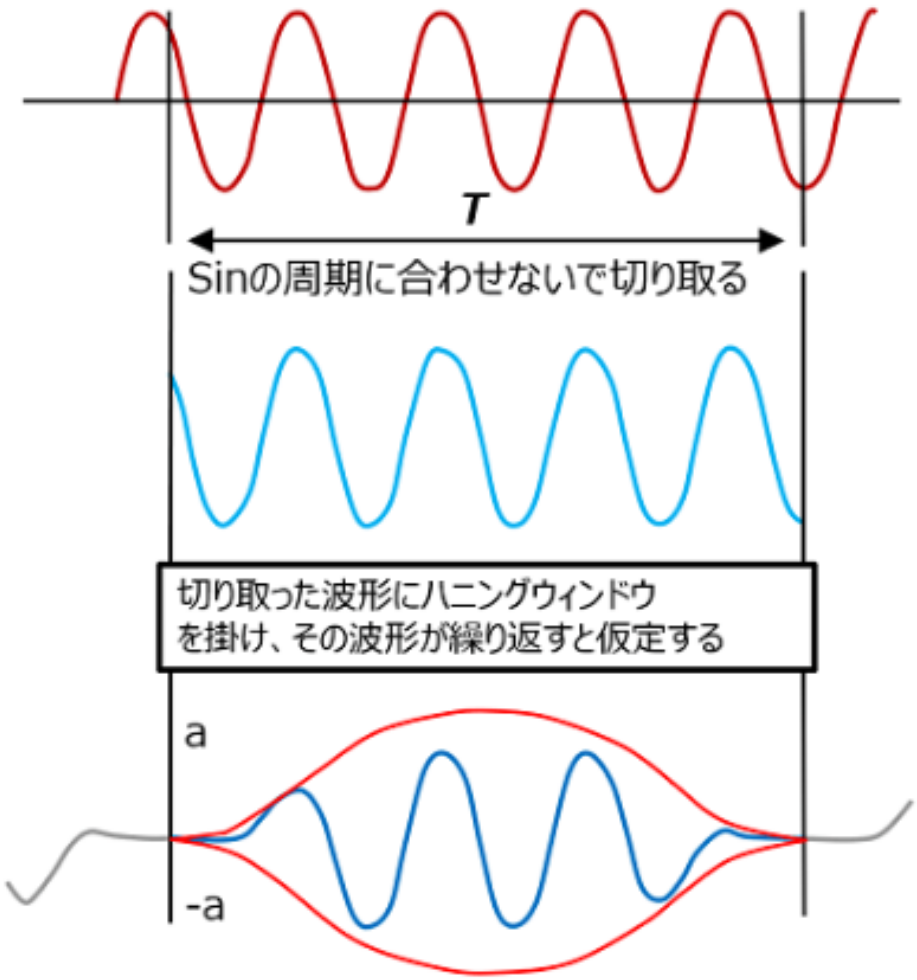
窓関数って何？

窓関数って？

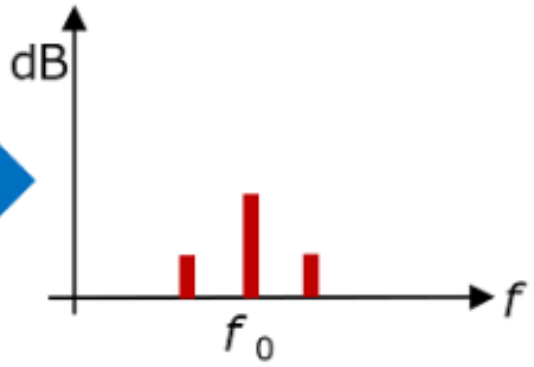

元のsin波形
T
Sinの周期に合わせないで切り取る
不連続点
切り取った波形が繰り返すと仮定すると始点と終点が不連続となる
不連続点
f (a+0)
f (a-0)
不連続点
dB
f 0以外の周波数にも多数のスペクトルが表れる
f 0

窓関数

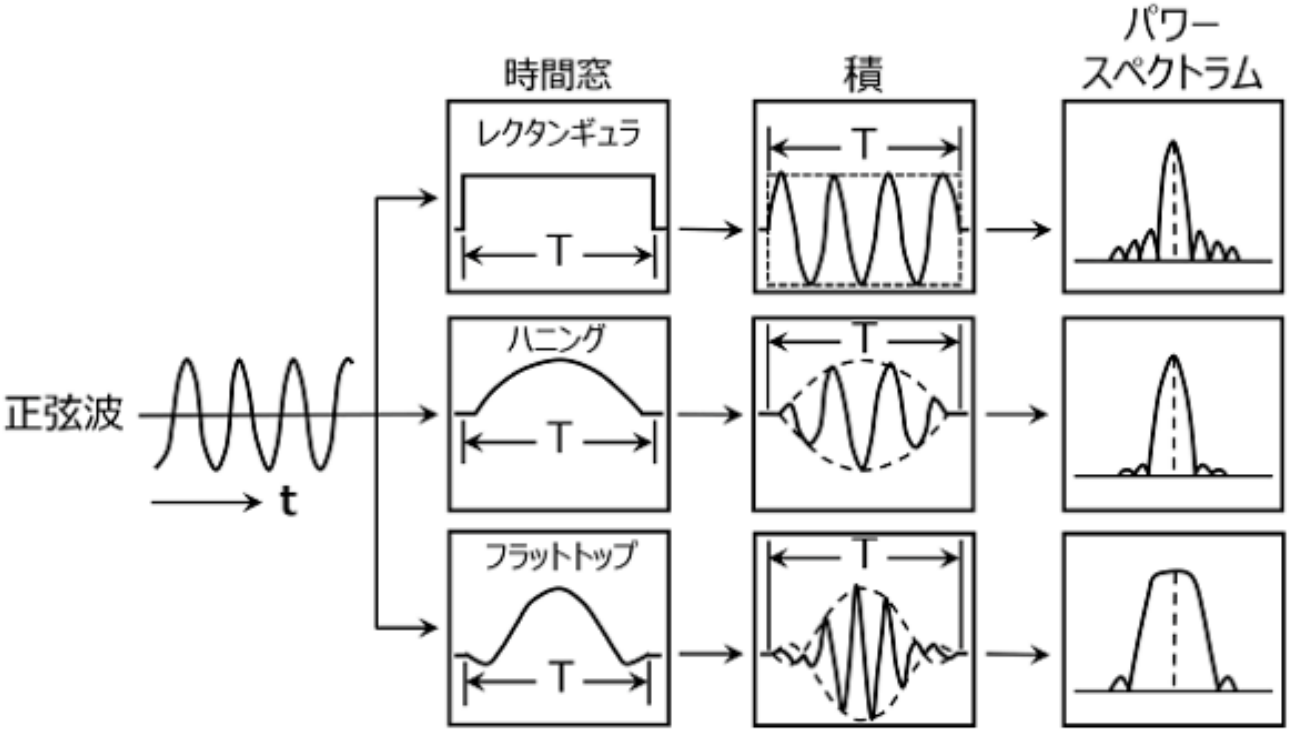
元のsin波形



f_0以外の周波数にもスペクトルが表れる
が、主たるスペクトルはf_0に現れる



主な窓関数



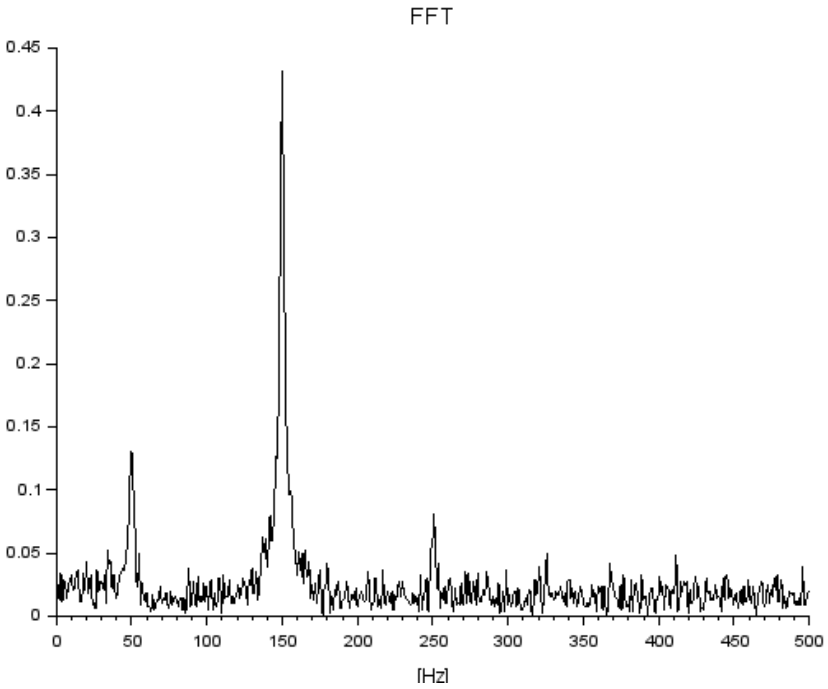
種類	周波数分解能	用途
レクタンギュラ	良い	インパルス波形などの過渡信号
ハニング	普通	一般の連続信号
フラットトップ	悪い	高周波分析などレベルを重視する信号

表1：代表的な窓関数の特長と用途

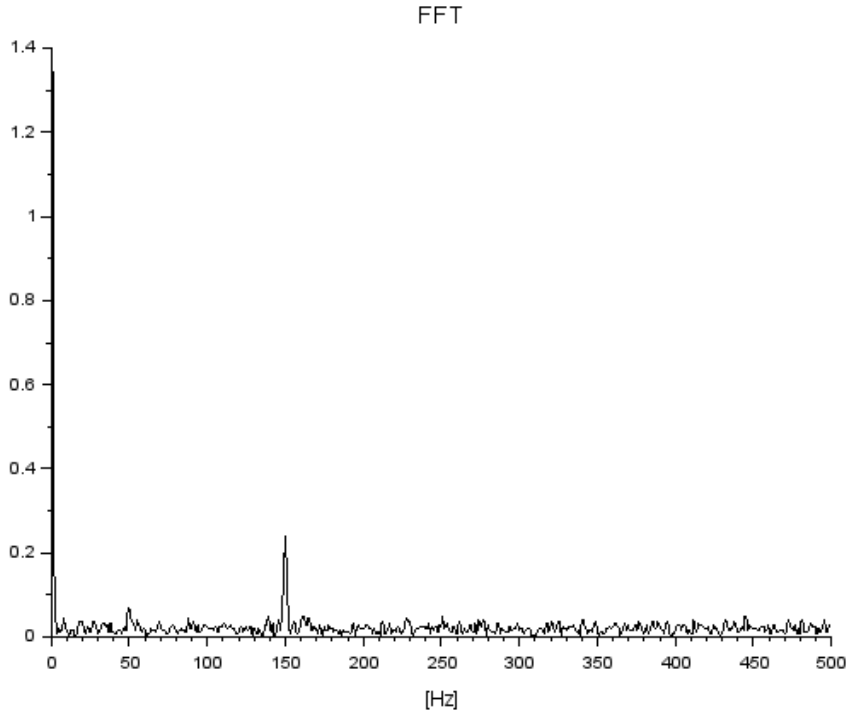
窓関数の効果

50Hz.300Hz,500Hz
の混ざった波形？
500Hzは？

窓関数無し



\ミング 窓関数



課題

- なにか, 1次元の正弦波データ, 実験データ等でよいので
  FFT(DFT)で周波数成分をだしてみましょう
- 元データはEXCELで作ってCSVにしてもかまいません

超お手軽に やるなら, データをカットアンドペーストしてもできます
(電卓替わりで便利です)
DC値だけ2倍になるので, 注意

```
--> a=[0 0 0 0 0 0 0 1 1 1 1 1 1]
 a  =

  0.   0.   0.   0.   0.   0.   0.   1.   1.   1.   1.   1.   1.

--> 2 * abs(fft(a)/length(a))
 ans  =

        column 1 to 12

  0.9230769   0.6381715   0.0792252   0.2169262   0.086874   0.1354125   0.1027682   0.1027682   0.1354125   0.086874   0.2169262   0.0792252

        column 13

  0.6381715
```

2次元DFTはできるの？ GPTに助けて貰えば. 簡単！

EMDj
Electrical Machines and Drives

```
//Scilabでの2D FFTの実行例

//2Dデータの生成
Nx = 128; //x方向のデータ点数
Ny = 128; //y方向のデータ点数
x = linspace(0, 2 * %pi, Nx);
y = linspace(0, 2 * %pi, Ny);
[XX, YY] = ndgrid(x, y);

//時間高調波と空間高調波を含むデータの生成
frequency_x = 2; //x方向の周波数
frequency_y = 4; //y方向の周波数
data = sin(frequency_x * XX) + cos(frequency_y * YY) + 2 * cos(3 * frequency_y * XX);

//2D FFTの実行
fft2_data = fft2(data);

//振幅スペクトルの計算
amplitude_spectrum = 2* abs(fft2_data) / (Nx * Ny);

//1象限のみを取り出す
half_Nx = Nx / 2;
half_Ny = Ny / 2;
amplitude_spectrum_1st_quadrant = amplitude_spectrum(1:half_Nx, 1:half_Ny);

//振幅スペクトルのプロット
figure();
subplot(2,1,1);
surf(amplitude_spectrum_1st_quadrant);
xlabel('Frequency X');
ylabel('Frequency Y');
zlabel('Amplitude');
title('2D FFT Amplitude Spectrum (1st Quadrant)');
```