

Octaveを使ったFFT

Ver. 1.0

2017年3月16日

塚本 貴城

1. フーリエ変換

フーリエ変換を下で定義する.

$$Y(\omega) = \int_0^T x(t)e^{-j\omega t} dt \quad (1)$$

たとえば, 時間信号が

$$x(t) = a_1 \cos(\omega_1 t + \phi_1) \quad (2)$$

$$= \frac{a_1}{2} \left(e^{i(\omega_1 t + \phi_1)} + e^{-i(\omega_1 t + \phi_1)} \right) \quad (3)$$

だったとしたら,

$$Y(\omega) = \frac{a_1}{2} \int_0^T \left[e^{i\phi_1} e^{i(\omega_1 - \omega)t} + e^{-i\phi_1} e^{-i(\omega_1 + \omega)t} \right] dt \quad (4)$$

となる. ここで, $\omega = \omega_1$ を考えると,

$$Y(\omega_1) = \frac{a_1}{2} \left\{ e^{i\phi_1} T + e^{-i\phi_1} \int_0^T e^{-2i\omega_1 t} dt \right\} \quad (5)$$

となる. 第2項は周期変動する成分であるが, T を十分長く取れば (あとでこの式全体を T で割るので) 無視できる. また, $\omega_1 t = \pi$ なら, 消えることになる. なので, 第2項を無視すると (ただし $\omega = 0$ の時は, 第2項は定数となって残るので注意),

$$Y(\omega_1) = \begin{cases} \frac{a_1}{2} T e^{i\phi_1} & (\text{when } \omega_1 \neq 0) \\ a_1 T \cos(\phi_1) & (\text{when } \omega_1 = 0) \end{cases} \quad (6)$$

となる. $\omega_1 = 0$ のときは, ϕ_1 は意味を持たないので (DC なので, 位相は無意味, 式2で $\omega_1 = 0$ としてみればわかるように, この場合は ϕ の変化は DC 成分の大きさの変化と同義), 最終的に

$$Y(\omega_1) = \begin{cases} \frac{a_1}{2} T e^{i\phi_1} & (\text{when } \omega_1 \neq 0) \\ a_1 T & (\text{when } \omega_1 = 0) \end{cases} \quad (7)$$

となる.

2. Octave での FFT 結果

Octave を使って FFT をしてみる. Octave を起動して, 以下のコマンドを打ち込む.

```
N=1000
DT=0.01
Df=1/(N*DT)
```

```

t=linspace(0,(N-1)*DT,N);

a1=1.23
f1=2.5
phi1=90
a2=2.4
f2=5.0
y=a1*sin(2*pi*f1*t+phi1/180.*pi) + a2*sin(2*pi*f2*t);

ft=fft(y); #何も考えず FFT する場合
ft=fft(y)/N*2; #縦軸の単位を, 元信号の単位と同じにする場合
ft(2:N/2) = ft(2:N/2)*2 #omega=0 を除いて, 正負の周波数をマージするため, 2 倍する
f = [0:N-1]*Df;

plot(f(1:N/2),abs(ft(1:N/2)))

```

結果は、図 1 のようになる。FFT の場合は、図のように $n = N/2$ で信号が折り返している。なので、意味があるのは $n = 0 \sim N/2$ までの区間である。

縦軸については、式 7 を参考に考えると、ただ単に

```
ft=fft(y)
```

とただけでは、実際の信号の大きさ（上の例では a_1, a_2 等）が、 $T = N * DT$ 倍されて出力されてしまう。実際図 1 を見ればわかるように、縦軸は非常に大きい値になってしまっている。（ $N=1000$ なので）また、式 7 からわかるように、フーリエ変換された結果は絶対値が $1/2$ になってしまう。

そこで、

```
ft=fft(y)/N*2
```

として、FFT の結果を N で割って（これはフーリエ変換の式（式 7 では T で割ることに相当）、さらに $1/2$ されていることを考慮して $\times 2$ してみる。ところで、この 2 というのは、図 7 等でわかるように、ある周波数 ω_1 の信号によるピークが $N/2$ を対称に 2 つ出てしまうことに起因している。（式 3 のように正負両方の周波数を足しあわせているから）図 2 に結果を示すように、このようなスケーリングを行うことで、信号の大きさ（上の例では a_1, a_2 ）が正しく検出できる。つまり、この処理により、縦軸の単位は元信号のものと一致する（Volt, m など）。

3. パワースペクトル密度 (PSD)

信号が持つパワーは、振幅の 2 乗で与えられる。まずは、上で考えていた正負両方の周波数をたす操作（振幅を 2 倍する）を一旦やめる。よって、パワースペクトルは、

$$P(\omega) = Y^2(\omega_1) \tag{8}$$

となる。

FFT で計算した値は、各周波数における成分の大きさ（振幅）をしめしている。FFT の場合は周波数は離散化されているので、各成分の“守備範囲”は

$$\Delta f = \frac{1}{N\Delta T} \tag{9}$$

となる。

よって、パワースペクトル密度は、

$$D(\omega) = P(\omega)/\Delta f \tag{10}$$

$$= Y^2(\omega)N\Delta T \tag{11}$$

となる。ここで、エネルギーに関しても、正負両方の周波数に分割されているので、1/2 になっている。そこで、この値を 2 倍することで、両方の周波数成分を正の周波数のみで表す。(DC 成分は正も負もなく、1/2 になっていないので、DC 成分は除く) 最終的に、

$$D(\omega) = 2Y(\omega)^2 N \Delta T \quad (12)$$

となる。例えば基の信号の単位が [V] であったなら、PSD の単位は [V²/Hz] になる。

```
#PSD
ft=fft(y)/N;
ps(1:N/2)=abs(ft).^2;
ps(2:N/2)=2*ps(2:N/2);
psd=ps/Df;

plot(f(1:N/2), psd(1:N/2));
loglog(f(1:N/2), psd(1:N/2));
```

4. ホワイトノイズの PSD 計算

試しにホワイトノイズの PSD を計算してみる。平均 0、分散 σ^2 の標準正規分布に従うノイズを考える。ホワイトノイズの自己相関関数は、

$$\phi(\tau) = \begin{cases} \sigma^2 & \tau = 0 \\ 0 & \tau \neq 0 \end{cases} \quad (13)$$

である。つまり、原点にピークがあるデルタ関数になっている。これのフーリエ変換がパワースペクトルになる。(Wiener-Khinchine の定理) よって、パワースペクトルは、

$$P(\omega) = \int_{-\infty}^{\infty} \phi(t) e^{-i\omega t} dt \quad (14)$$

$$= \int_{-\infty}^{\infty} \sigma^2 \delta(t) e^{-i\omega t} dt \quad (15)$$

$$= \sigma^2 e^{i\omega 0} \quad (16)$$

$$= \sigma^2 \quad (17)$$

となり、全周波数で一定値となる。

ところで、実際には上のような離散時間ではなく(帯域無限大)、あるサンプリング周波数でサンプルされたデータを扱う。サンプリング周期を T_s とすると、サンプリング周波数は $f_s = 1/T_s$ である。よって、スペクトル密度値(一定) × 帯域幅 がこのノイズ信号の持つパワーになる。一方、ノイズ信号の振幅 RMS 値は σ である。よって、ノイズ信号のパワーは σ^2 となる。以上より、ノイズの PSD は σ^2/f_s となる。

実際に下のコマンドで計算してみる。

```
N=100000
DT=0.001
Df=1/(N*DT)
t=linspace(0,(N-1)*DT,N);

avg=0
var=1.0
r=randn(1,N)*sqrt(var)+avg;
a1=2.5
```

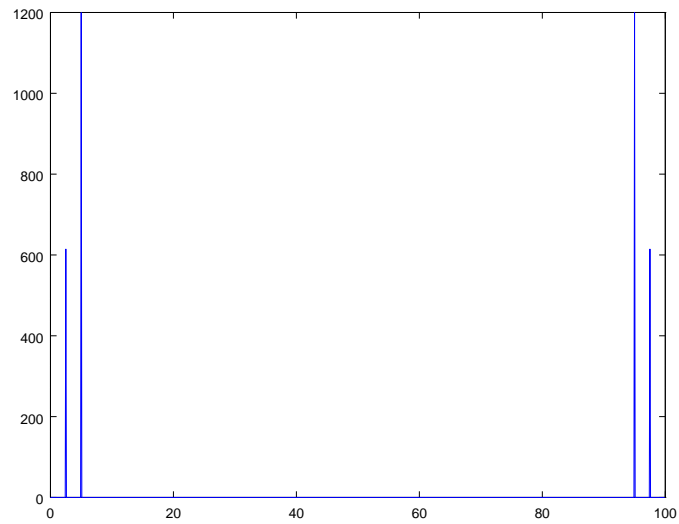


図 1: FFT 結果 . 横軸の単位は Hz . 縦軸は関数 fft() の出力をそのまま

```

f1=100
y=a1*sin(2*pi*f1*t);

#PSD of random noise
ftr=fft( r + y)/N;
f = [0:N-1]*Df;

psr(1:N/2)=abs(ftr(1:N/2)).^2;
psr(2:N/2)=2 *psr(2:N/2);
psdr=psr/Df;

plot(t, y+r);
plot(f(1:N/2),sqrt( psdr(1:N/2)) )

```

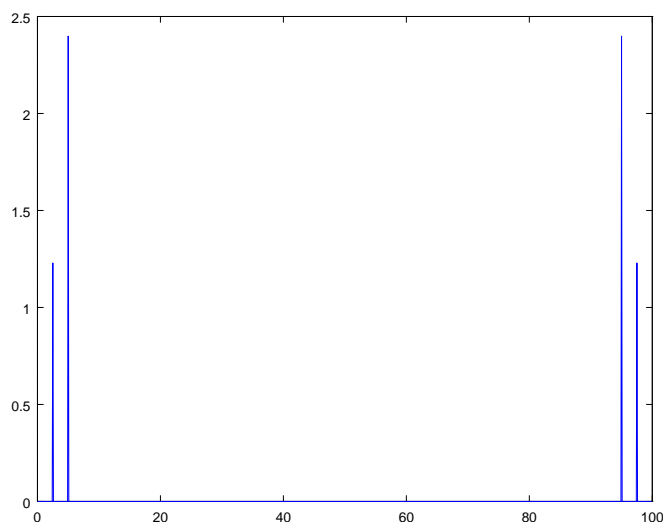


図 2: FFT 結果．横軸の単位は Hz．縦軸は `fft()` の出力をスケールリングして，実単位に合わせたもの．

```

plot(f(1:N/2), psdr(1:N/2) )
plot(f(1:N/2), 10*log10(psdr(1:N/2)) )
loglog(f(2:N/2), psdr(2:N/2))

```

結果を図 3 に示す．ノイズの分散は $\sigma^2 = 1$ であり，サンプリング周波数は $f_s = 1/DT = 1000$ としたので，予想される PSD は

$$\text{PSD}_{\text{noise}} = \sigma^2 / f_s = 1 \times 10^{-3} [\text{V}^2/\text{Hz}] \quad (18)$$

であり，Octave の計算結果と一致する．

また，信号の振幅は $a = 2.5$ としたので，正負周波数を考えると，それぞれの振幅は $a/2 = 1.25$ となる．よって，それぞれのパワーは $a^2/4 = 1.5625$ となる．負の周波数分を正の周波数にマージすると，結局これが 2 倍になるので，信号周波数におけるパワーは $a^2/2 = 3.125$ となる．周波数分解能は， $\Delta f = \frac{1}{N \times DT} = 0.01$ で

あるので，結局 PSD は

$$\text{PSD}_{\text{signal}} = P/\Delta f \quad (19)$$

$$= a^2/2 \times N \times DT \quad (20)$$

$$= 312.5[\text{V}^2/\text{Hz}] \quad (21)$$

となる．

ちなみに，dB 表示にすると

$$\text{PSD}_{\text{noise}} = 10\log_{10} 10^{-3} \quad (22)$$

$$= -30 [\text{dB}/\text{Hz}] \quad (23)$$

$$\text{PSD}_{\text{signal}} = 10\log_{10} 312.5 \quad (24)$$

$$= 24.9 [\text{dB}/\text{Hz}] \quad (25)$$

となる．この結果は，図 4 と一致する．

【メモ】

db を計算するとき，電力の場合は $10\log_{10}(P)$ になることに注意（よく使う，電圧等の db 計算は $20\log_{10} V$ ）．これは，db は基本的に電力比で考えることに由来する．つまり，

$$10\log \frac{P_1}{P_0} = 10\log \frac{V_1^2}{V_0^2} \quad (26)$$

$$= 10\log \frac{V_1}{V_0} \frac{V_1}{V_0} \quad (27)$$

$$= 10\log \left(\frac{V_1}{V_0} \right)^2 \quad (28)$$

$$= 20\log \left(\frac{V_1}{V_0} \right) \quad (29)$$

なので，PSD でデシベル表示するときには $10\log(\dots)$ を使うこと．

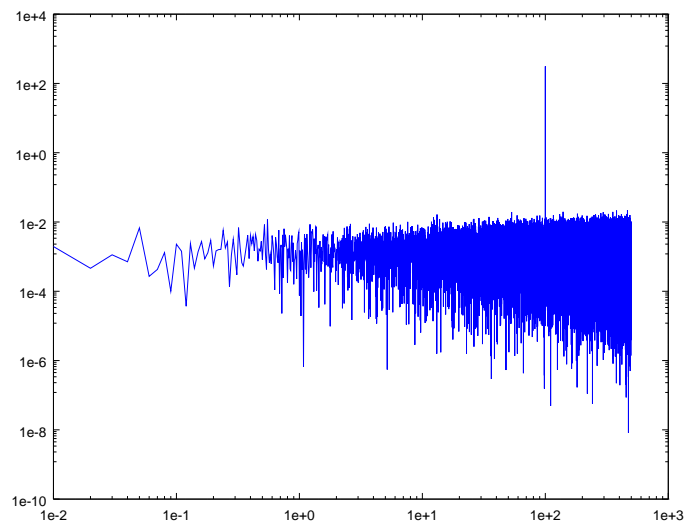


図 3: PSD の計算結果 . ノイズの分散は"1", 信号の振幅は 2.5(@100Hz) .

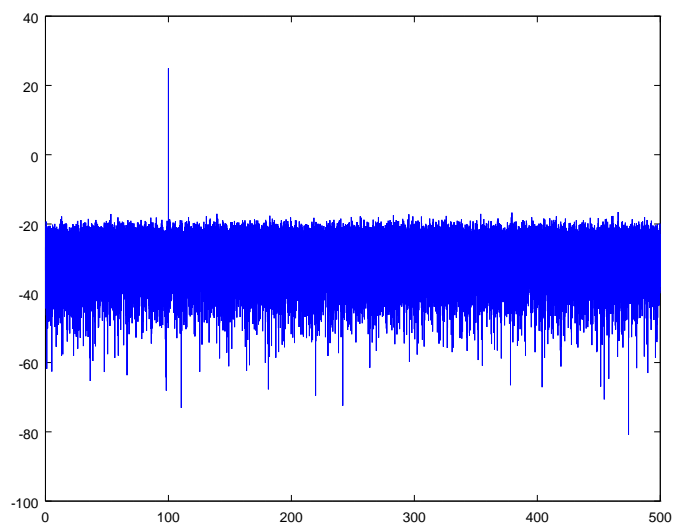


図 4: PSD の計算例 . 縦軸は dB 表示