応用数学 期末レポート

佐々木良輔

1-1

$a(x)$ について

$f(x)$ を

$$f(x) = \begin{cases} 1 & \{-\pi \leq x \leq 0\} \\ \mathrm{e}^{-x} & \{0 \leq x \leq \pi\} \end{cases}$$

で定義する. このとき Fourier 係数は以下で与えられる. まず a_0 について

$$\begin{aligned} a_0 &= \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\mathrm{d}x \\ &= \frac{1}{\pi}\left\{\int_{-\pi}^{0}\mathrm{d}x + \int_{0}^{\pi}\mathrm{e}^{-x}\mathrm{d}x\right\} \\ &= 1 + \frac{1}{\pi}(1-\mathrm{e}^{-\pi}) \end{aligned}$$

次に a_n について

$$\begin{aligned} a_n &= \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\cos nx\mathrm{d}x \\ &= \frac{1}{\pi}\left\{\int_{-\pi}^{0} 1\cdot\cos nx\mathrm{d}x + \int_{0}^{\pi}\mathrm{e}^{-x}\cdot\cos nx\mathrm{d}x\right\} \end{aligned}$$

ここで $I = \int_0^{\pi}\mathrm{e}^{-x}\cdot\cos nx\mathrm{d}x$ とすると, 部分積分を用いて

$$\begin{aligned} I &= \left[-\mathrm{e}^{-x}\cos nx\right]_0^{\pi} - \int_0^{\pi} n\mathrm{e}^{-x}\sin nx\mathrm{d}x \\ &= \left[-\mathrm{e}^{-x}\cos nx\right]_0^{\pi} + n\left[e^{-x}\sin nx\right]_0^{\pi} - n^2 I \\ \therefore \qquad I &= \frac{\mathrm{e}^{-\pi}n\sin n\pi - \mathrm{e}^{-\pi}\cos n\pi + 1}{n^2+1} \end{aligned}$$

であるので以下を得る.

$$a_n = \frac{1}{\pi}\frac{1-\mathrm{e}^{-\pi}\cos n\pi}{n^2+1}$$

同様に b_n について

$$\begin{aligned} b_n &= \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\sin nx\mathrm{d}x \\ &= \frac{1}{\pi}\left\{\int_{-\pi}^{0} 1\cdot\sin nx\mathrm{d}x + \int_{0}^{\pi} \mathrm{e}^{-x}\cdot\sin nx\mathrm{d}x\right\} \end{aligned}$$

ここで $I = \int_0^{\pi} \mathrm{e}^{-x}\cdot\sin nx\mathrm{d}x$ とすると

$$\begin{aligned} I &= \left[-\mathrm{e}^{-x}\sin nx\right]_0^{\pi} + \int_0^{\pi} n\mathrm{e}^{-x}\cos nx\mathrm{d}x \\ &= \left[-\mathrm{e}^{-x}\sin nx\right]_0^{\pi} - n\left[e^{-x}\cos nx\right]_0^{\pi} - n^2 I \\ \therefore \qquad I &= \frac{-\mathrm{e}^{-\pi}\sin n\pi - \mathrm{e}^{-\pi}n\cos n\pi + n}{n^2+1} \end{aligned}$$

であるので以下を得る.

$$b_n = \frac{1}{\pi}\left\{\frac{\cos n\pi - 1}{n} + \frac{n - \mathrm{e}^{-\pi}n\cos n\pi}{n^2+1}\right\}$$

以上から求める Fourier 級数 $a(x)$ は以下のようになる.

$$\begin{aligned} a(x) = \frac{1}{2}\left(1 + \frac{1}{\pi}(1-\mathrm{e}^{-\pi})\right) + \sum_{n=1}^{\infty}\Bigg(&\frac{1}{\pi}\frac{1-\mathrm{e}^{-\pi}\cos n\pi}{n^2+1}\cos nx \\ &+\frac{1}{\pi}\left\{\frac{\cos n\pi - 1}{n} + \frac{n-\mathrm{e}^{-\pi}n\cos n\pi}{n^2+1}\right\}\sin nx\Bigg) \end{aligned}$$

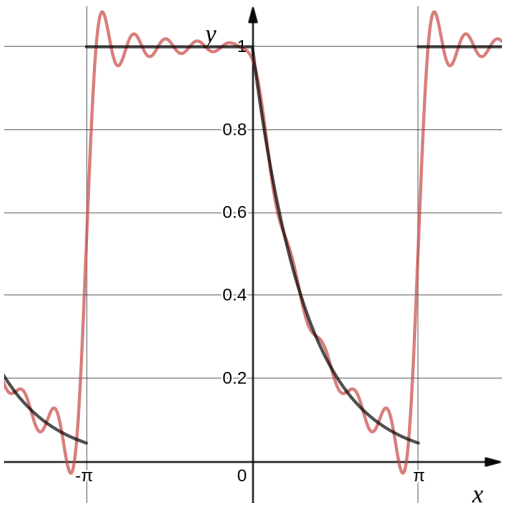

図 1　$y = f(x)$(黒線) と $y = a(x)$(赤線,10 次まで)(by desmos 計算機)

$b(x)$ について

$x \in [0, \pi]$ としたときの Fourier 余弦級数の係数は以下で与えられる. まず a_0 について

$$\begin{aligned} a_0 &= \frac{2}{\pi}\int_0^{\pi} f(x)\mathrm{d}x \\ &= \frac{2}{\pi}\int_0^{\pi} \mathrm{e}^{-x}\mathrm{d}x \\ &= \frac{2}{\pi}(1-\mathrm{e}^{-\pi}) \end{aligned}$$

次に a_n について

$$\begin{aligned} a_n &= \frac{2}{\pi}\int_0^{\pi} f(x)\cos nx\mathrm{d}x \\ &= \frac{2}{\pi}\int_0^{\pi} \mathrm{e}^{-x}\cdot\cos nx\mathrm{d}x \\ &= \frac{2}{\pi}\frac{1-\mathrm{e}^{-\pi}\cos n\pi}{n^2+1} \end{aligned}$$

以上から求める Fourier 余弦級数 $b(x)$ は以下のようになる.

$$b(x) = \frac{1}{\pi}(1-\mathrm{e}^{-\pi}) + \sum_{n=1}^{\infty}\frac{2}{\pi}\frac{1-\mathrm{e}^{-\pi}\cos n\pi}{n^2+1}$$

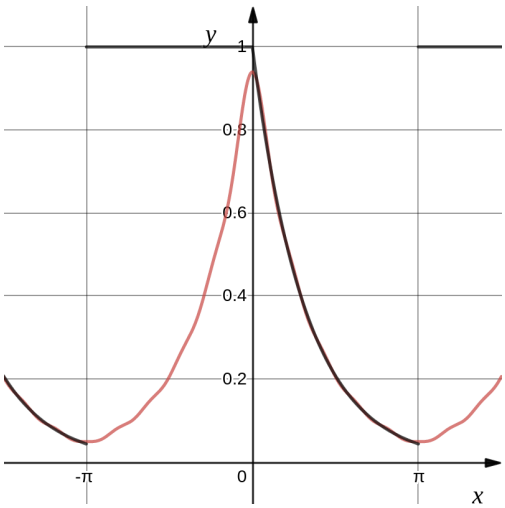

図 2　$y = f(x)$(黒線) と $y = b(x)$(赤線,10 次まで)(by desmos 計算機)

$c(x)$ について

$x \in [0, \pi]$ としたときの Fourier 正弦級数の係数は以下で与えられる.

$$\begin{aligned} b_n &= \frac{2}{\pi} \int_0^{\pi} f(x) \sin nx \mathrm{d}x \\ &= \frac{2}{\pi} \int_0^{\pi} \mathrm{e}^{-x} \cdot \sin nx \mathrm{d}x \\ &= \frac{2}{\pi} \frac{n - \mathrm{e}^{-\pi} n \cos n\pi}{n^2 + 1} \end{aligned}$$

以上から求める Fourier 正弦級数は以下のようになる.

$$c(x) = \sum_{n=1}^{\infty} \frac{2}{\pi} \frac{n - \mathrm{e}^{-\pi} n \cos n\pi}{n^2 + 1}$$

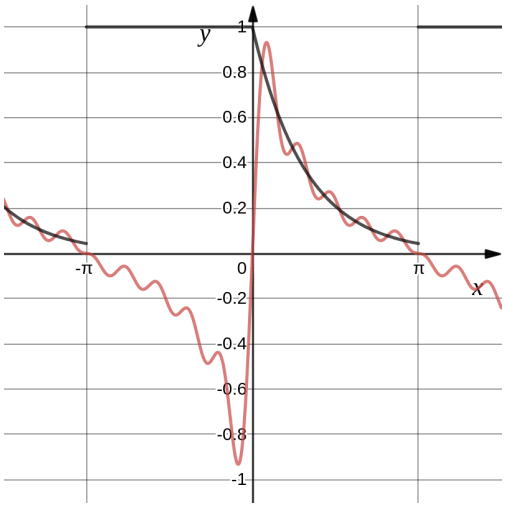

図 3　$y = f(x)$(黒線) と $y = c(x)$(赤線,10 次まで)(by desmos 計算機)

1-2-(a),(b)

以下に各次数の Fourier 係数を示す.

表 1　$a(x)$ の Fourier 係数

次数 n	a_n	b_n
0	$1+(1-\mathrm{e}^{-\pi})/\pi$	0
1	$(1+\mathrm{e}^{-\pi})/2\pi$	$(\mathrm{e}^{-\pi}-3)/2\pi$
2	$(1-\mathrm{e}^{-\pi})/5\pi$	$(2-2\mathrm{e}^{-\pi})/5\pi$
3	$(1+\mathrm{e}^{-\pi})/10\pi$	$(9\mathrm{e}^{-\pi}-11)/30\pi$
4	$(1-\mathrm{e}^{-\pi})/17\pi$	$(4-4\mathrm{e}^{-\pi})/17\pi$
5	$(1+\mathrm{e}^{-\pi})/26\pi$	$(25\mathrm{e}^{-\pi}-27)/130\pi$

表 2　$b(x)$ の Fourier 係数

次数 n	a_n
0	$2(1-\mathrm{e}^{-\pi})/\pi$
1	$(1+\mathrm{e}^{-\pi})/\pi$
2	$(1-\mathrm{e}^{-\pi})/5\pi$
3	$(1+\mathrm{e}^{-\pi})/10\pi$
4	$(1-\mathrm{e}^{-\pi})/17\pi$
5	$(1+\mathrm{e}^{-\pi})/26\pi$

表 3　c(x) について

次数 n	b_n
1	$(1+\mathrm{e}^{-\pi})/\pi$
2	$(4-4\mathrm{e}^{-\pi})/5\pi$
3	$(3+3\mathrm{e}^{-\pi})/5\pi$
4	$(8-8\mathrm{e}^{-\pi})/17\pi$
5	$(5+5\mathrm{e}^{-\pi})/13\pi$

したがって各 Fourier 級数をプロットすると以下のグラフを得る.

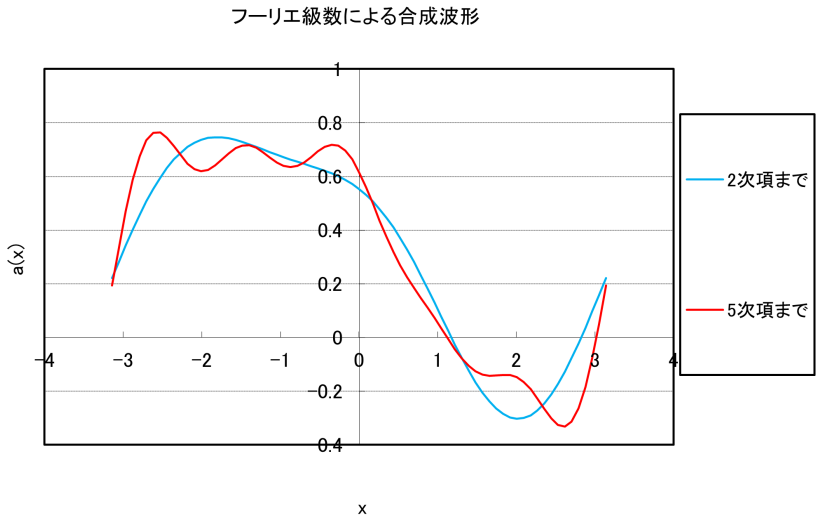

図 4　$a(x)$ のグラフ (水色:2 次まで, 赤:5 次まで)

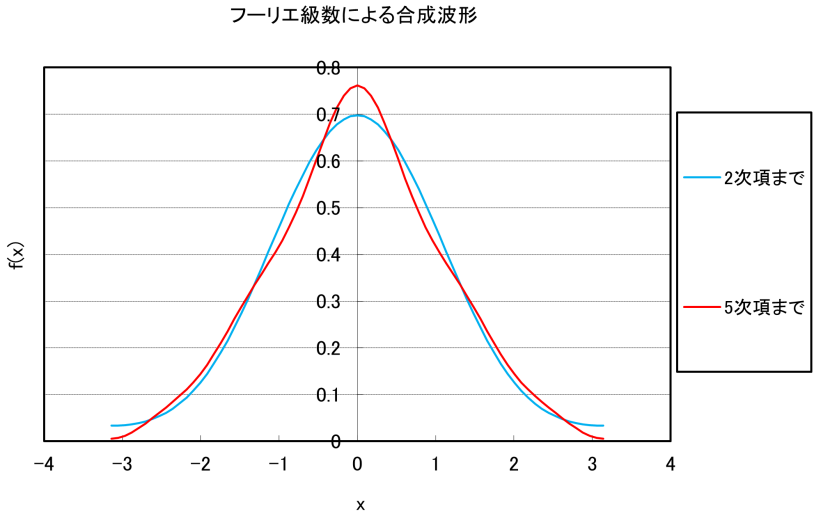

図 5　$b(x)$ のグラフ (水色:2 次まで, 赤:5 次まで)

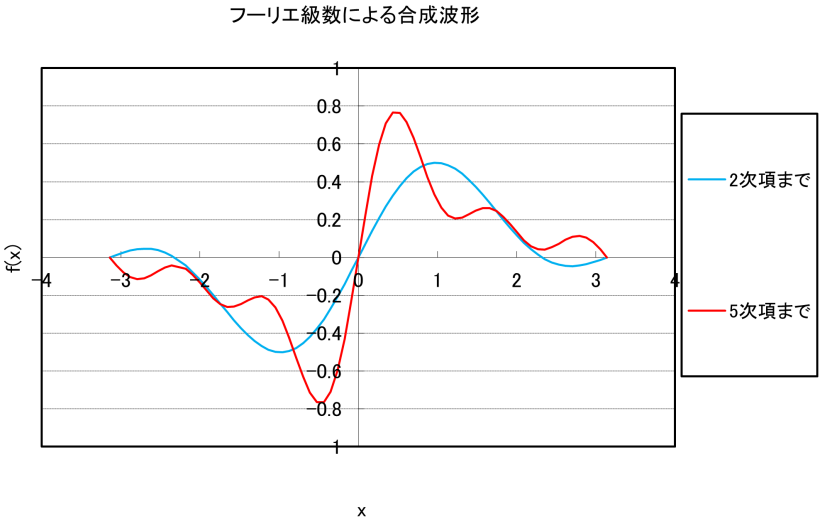

図 6　$c(x)$ のグラフ (水色:2 次まで, 赤:5 次まで)

1-3

以下に $\Delta a(x)$, $\Delta b(x)$, $\Delta c(x)$ のグラフを示す.

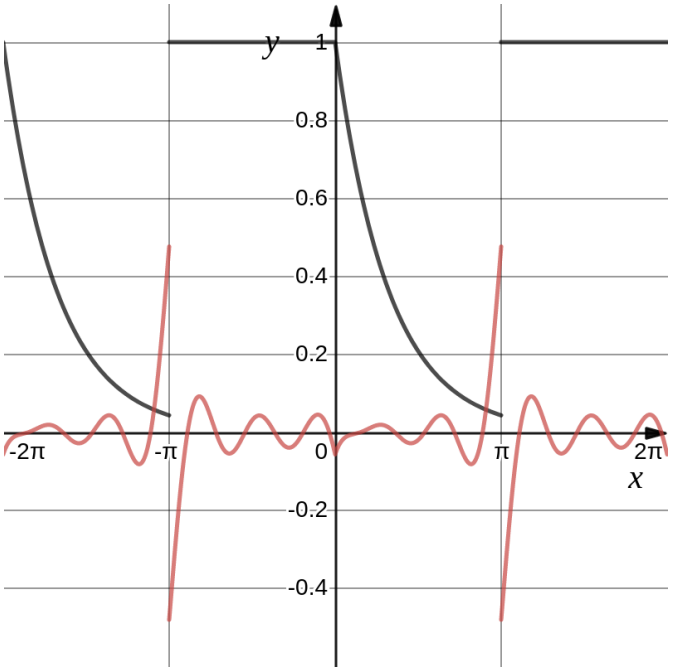

図 7　$y = f(x)$(黒線) と $y = \Delta a(x)$(赤線,5 次まで)(by desmos 計算機)

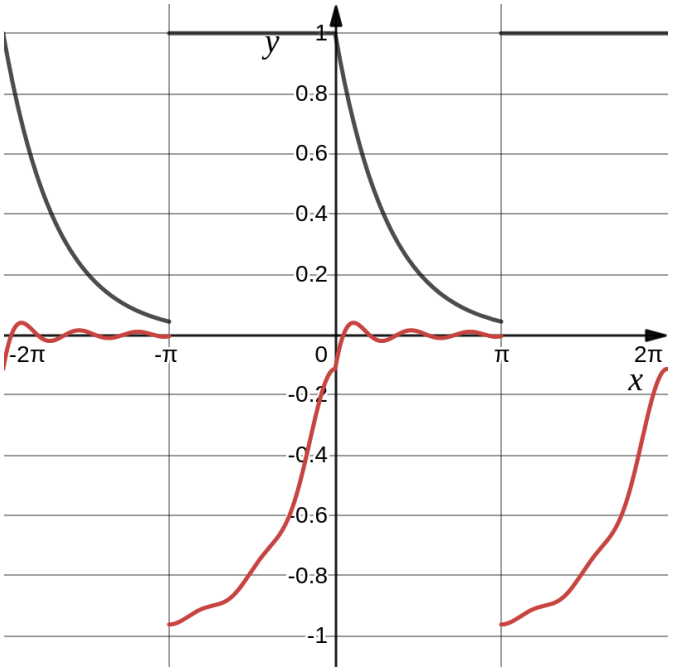

図 8　$y = f(x)$(黒線) と $y = \Delta b(x)$(赤線,5 次まで)(by desmos 計算機)

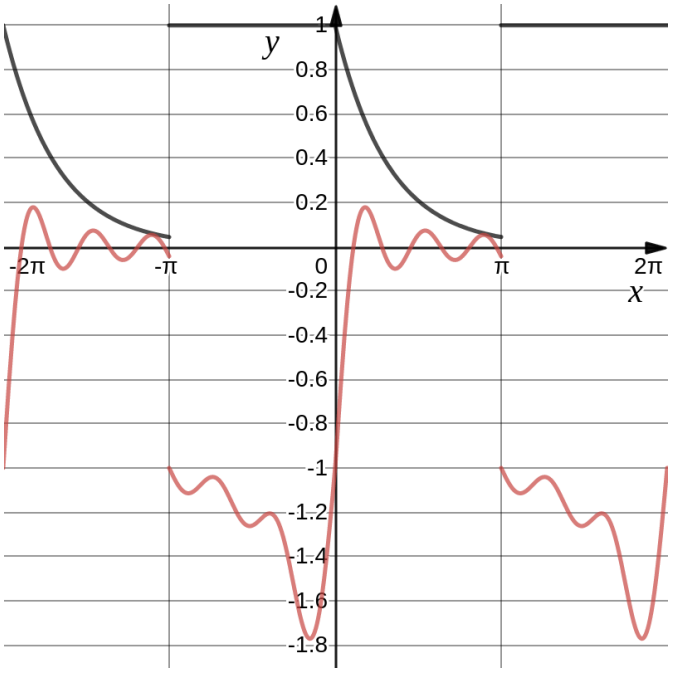

図 9　$y = f(x)$(黒線) と $y = \Delta c(x)$(赤線,5 次まで)(by desmos 計算機)

1-4

(1) 図 1 や図 4 から級数の次数が高いほど元の波形に近づいていることがわかる. そして無限級数となったとき, 級数は元の波形と一致する.(完全性)
(2) 図 6 や図 1 の不連続点をみると関数の不連続点 $x = x_0$ では級数の値は

$$\frac{\lim_{x \to -x_0} f(x) + \lim_{x \to +x_0} f(x)}{2}$$

に収束している.(Dirichlet の定理)
(3) 図 1, 図 7 を見ると, 不連続点では誤差が大きくなっていることがわかる. 不連続点では項数が有限である限り級数は収束しない.(Gibbs の現象)
(4) 表 2, 表 3 を見ると $b(x)$ の方が $c(x)$ に比べて係数がすぐに小さくなっていることがわかる. そのため図 5, 図 6 のように, $b(x)$ の方が $c(x)$ より早く元の関数に近づいていることがわかる.

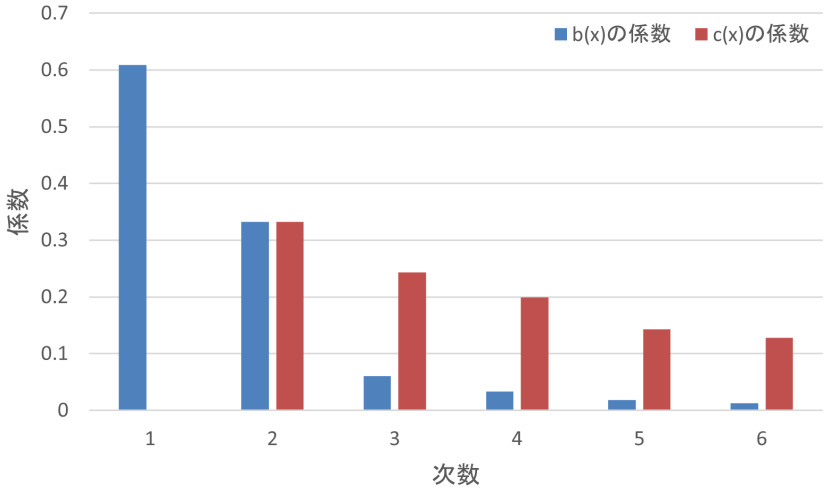

図 10　$b(x)$ と $c(x)$ の係数

(5) 図 5, 図 6 から Fourier 余弦級数は偶関数, Fourier 正弦級数は奇関数となっていることがわかる.

2-1

図 11, 図 12 に元波形と周波数スペクトルを示す. これらは自作の python スクリプト (A.1) で作成した. python スクリプトでは csv データを配列に格納し, これに numpy ライブラリの fft.fft 関数を掛けている. この関数は以下の定義式に基づき離散 Fourier 変換を行う.[1]

$$F_k = \sum_{m=0}^{n-1} f_k \exp\left(-2\pi i \frac{mk}{n}\right)$$

この値は複素数であるので, スペクトルにおいては絶対値を取った. 結果は matplotlib にて出力した.

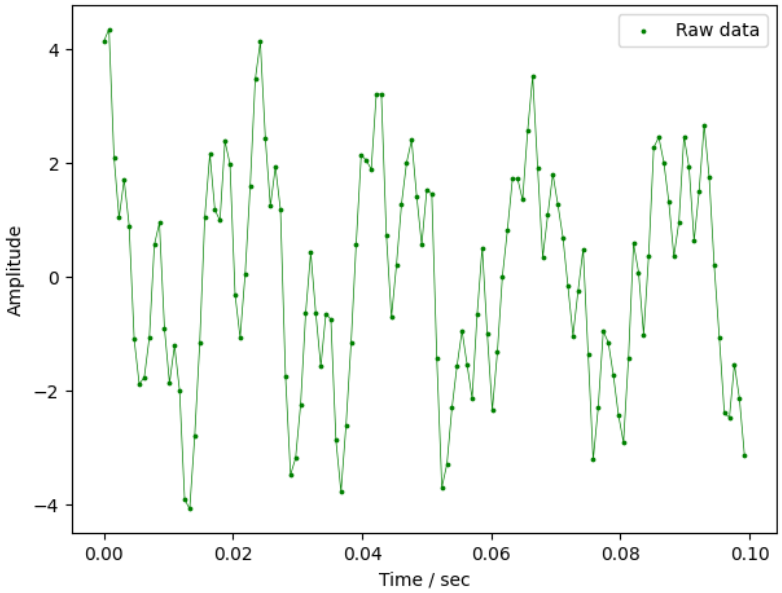

図 11　元波形

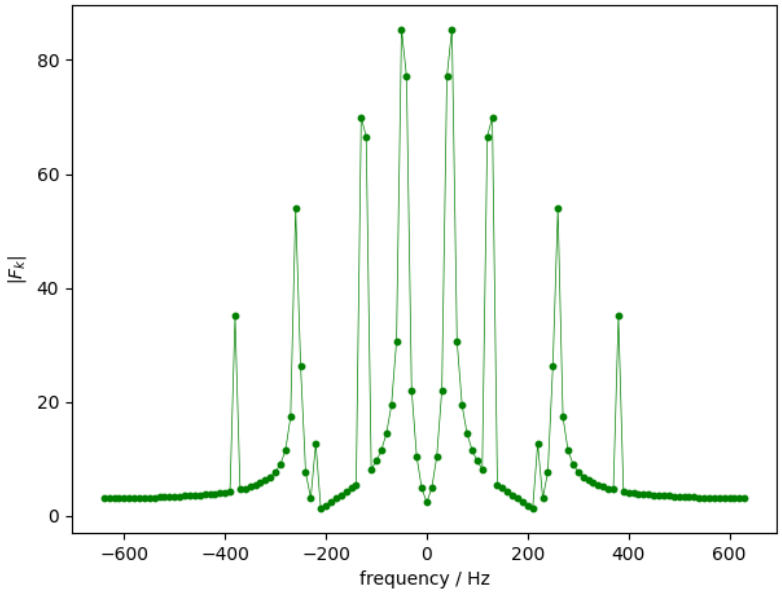

図 12　周波数スペクトル

2-2

元データは幾つかの余弦波の合成であるので, 理想的には周波数スペクトルは図 13 のように δ 関数からなるはずである. しかし実際には, 図 12 のように量子化誤差や離散化によりノイズやサイドローブが乗っていると考えられる. したがって振幅スペクトルのピークを拾うことでノイズを除去し, 元の余弦波を得る. ピークは Scipy ライブラリの signal.find_peaks 関数による極大値, ならびに閾値により検出している.

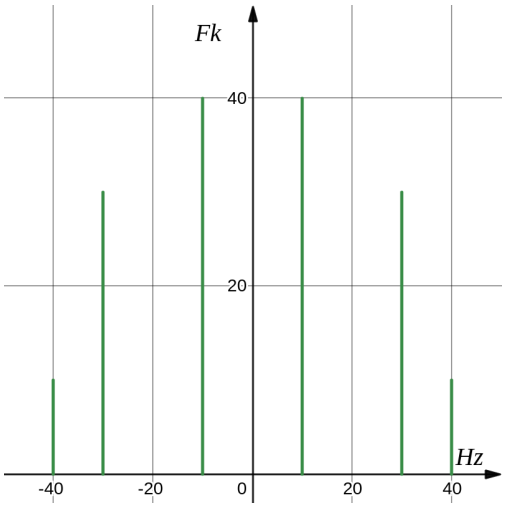

図 13　理想的なスペクトルの例 (by desmos 計算機)

図 14 に振幅スペクトル, 図 15 に位相スペクトルを示す. それぞれのスペクトルには振幅スペクトルのピークを書き込んでいる. なお振幅は以下のように計算される.

$$A_k = \frac{|F_k| \times 2}{N}$$

また表 4 にはノイズ除去により得られたピーク周波数と, そこでの振幅, 位相を示す. これを元に再合成した波形を図 16 に示す. 図 16 のように再合成した波形は元データをある程度再現している. したがって元データは表 4 に示した周波数, 振幅, 位相の余弦波の重ね合わせであると考えられる.

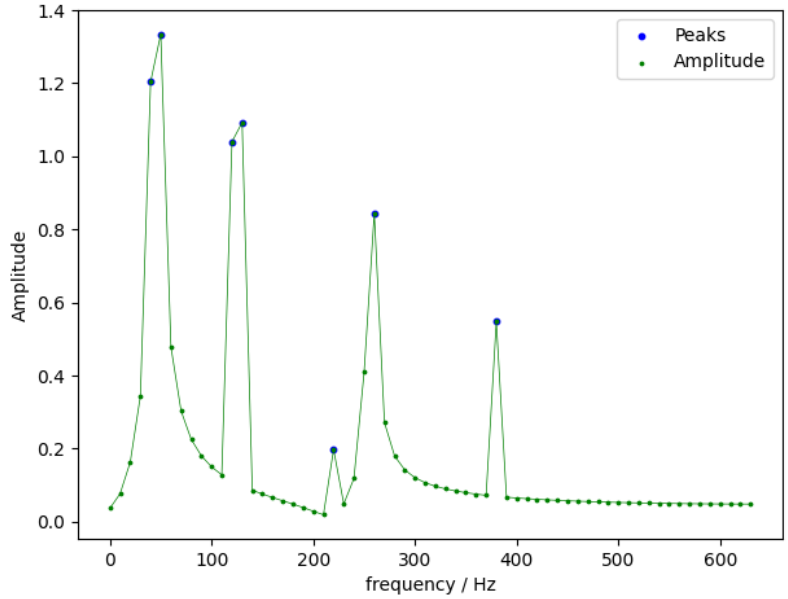

図 14　振幅スペクトルとそのピーク

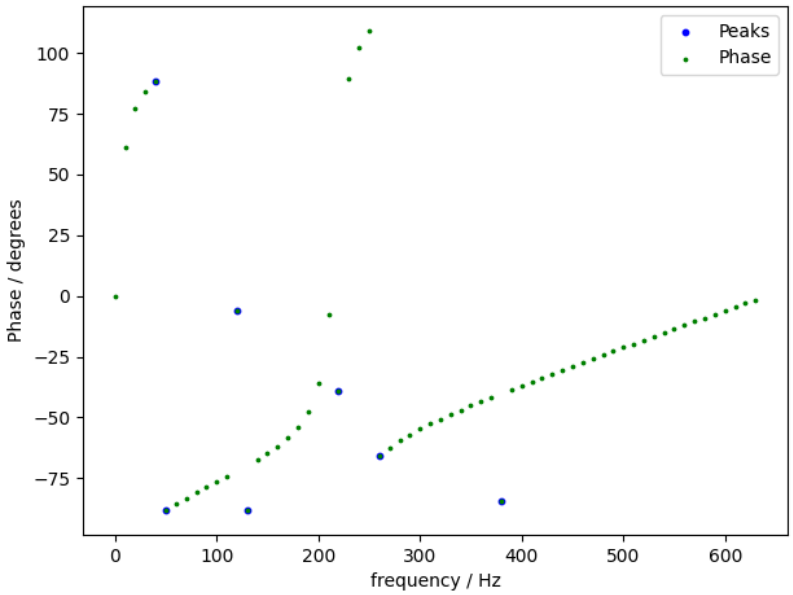

図 15　位相スペクトル

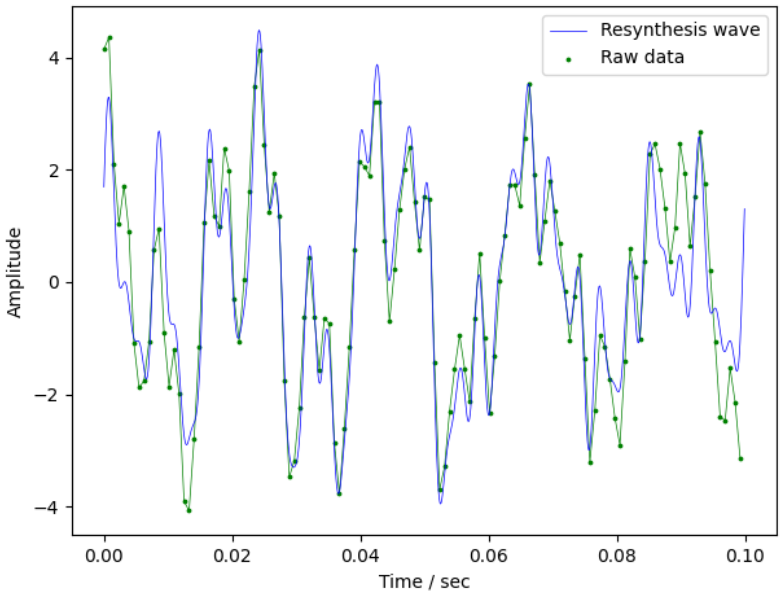

図 16　再合成した波形

表 4　ピーク周波数, 振幅, 位相

周波数 / Hz	振幅	位相 / °
40.0	1.20	88.2
50.0	1.33	-88.4
120	1.04	-5.86
130	1.09	-88.2
220	0.197	-38.9
260	0.843	-65.9
380	0.548	-84.4

3-1

図 17 に元データと再合成した波形, 図 18 に周波数スペクトル, 図 19 に振幅スペクトルを示す. 表 5 に各ピークの周波数, 振幅, 位相を示す. Nyquist 周波数 $f_N = 64.0$ Hz である.

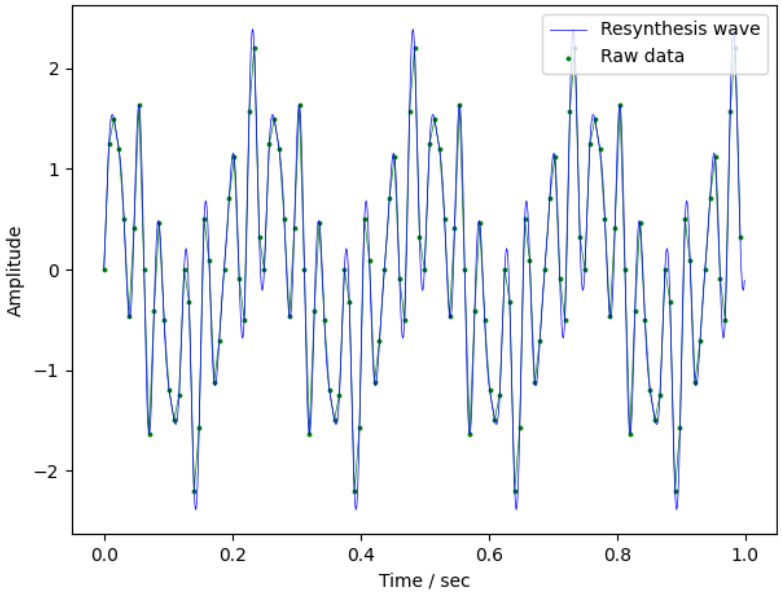

図 17　元データと再合成したデータ

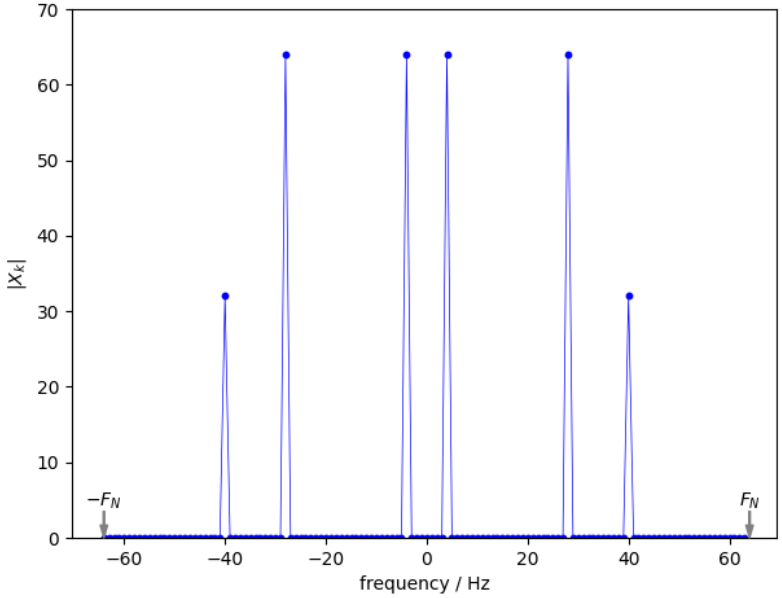

図 18　周波数スペクトル

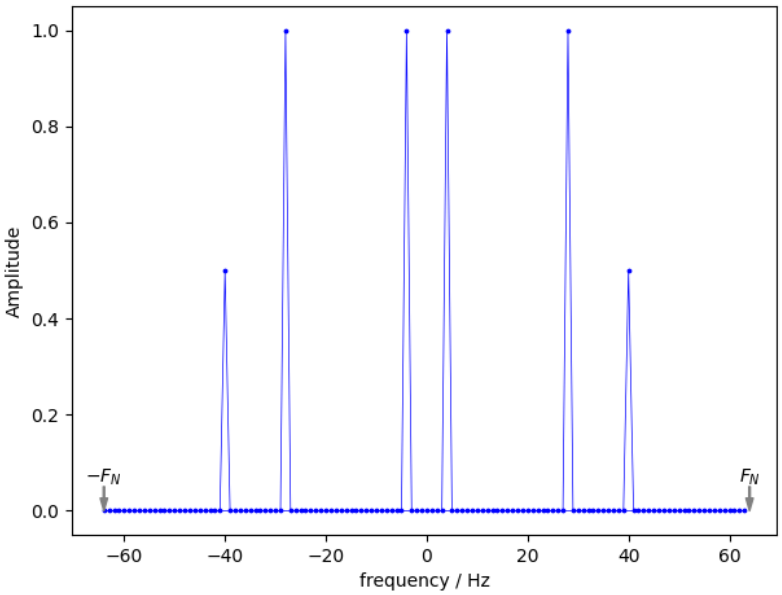

図 19　振幅スペクトル

表 5　各ピークの振幅

周波数 / Hz	振幅	位相 / °
4.00	1.00	2.03×10^{-14}
28.0	1.00	1.80×10^{2}
40.0	0.500	-9.00×10^{1}

3-2

図 20 に元データと再合成した波形, 図 21 に周波数スペクトル, 図 22 に振幅スペクトルを示す. 表 6 に各ピークの周波数, 振幅, 位相を示す. Nyquist 周波数 $f_N = 32.0$ Hz である. また表 A.1 にはダウンサンプリング後のデータを示した. ダウンサンプリング処理は python スクリプトで行っている.

振幅スペクトルから f_N を中心に波形が折り返すエイリアシングが発生していることがわかる. 実際に表 6 より, 振幅 0.5 のピークが $\pm F_N - (\pm 40.0 \mp F_N) = \pm 24.0$ Hz に生じている. また折り返した余弦波は位相が反転している. さらにダウンサンプリングによりデータ数が半分になったため, 周波数スペクトル $|Y_k|$ の値も半分になっている.

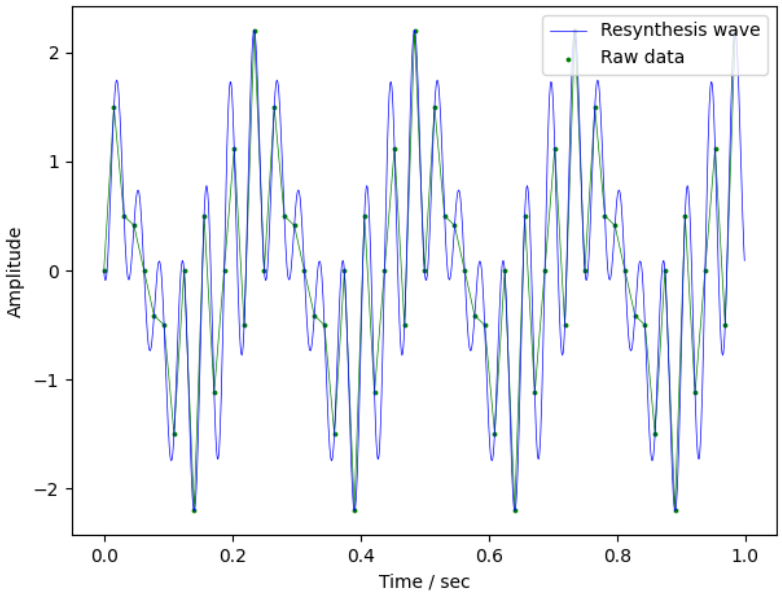

図 20　元データと再合成したデータ

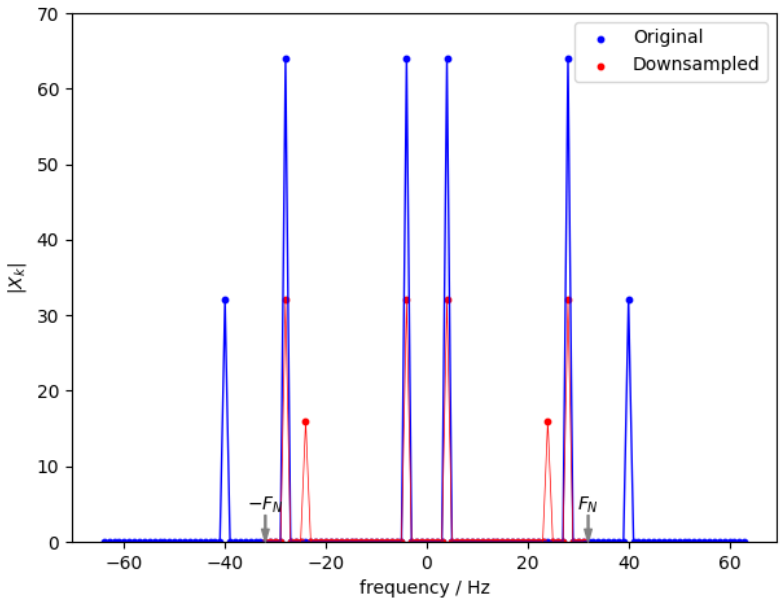

図 21　周波数スペクトル

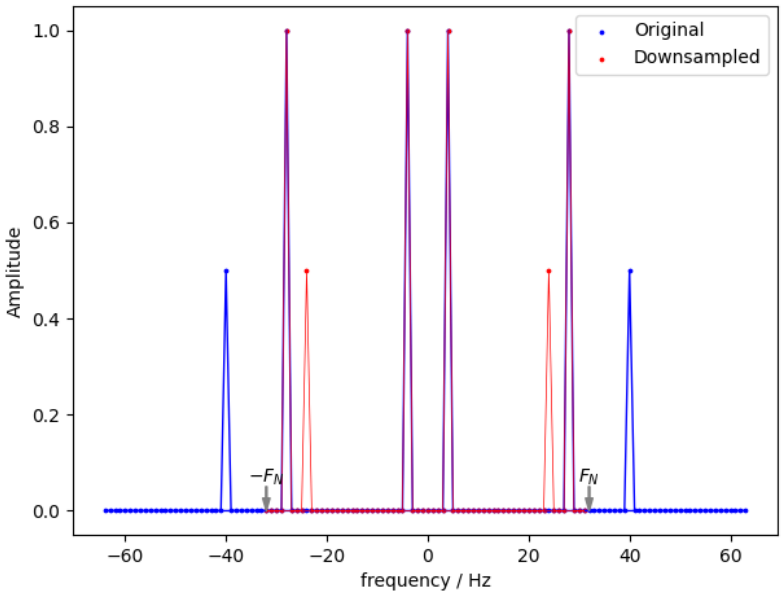

図 22　振幅スペクトル

表 6　各ピークの振幅

周波数 / Hz	振幅	位相 / °
4.00	1.00	2.03×10^{-14}
24.0	0.500	9.00×10^{1}
28.0	1.00	1.80×10^{2}

3-3

図 23 に元データと再合成した波形, 図 24 に周波数スペクトル, 図 25 に振幅スペクトルを示す. 表 7 に各ピークの周波数, 振幅, 位相を示す. Nyquist 周波数 $f_N = 16.0$ Hz である. また表 A.2 にはダウンサンプリング後のデータを示した.

元データでの 4.00 Hz と 28.0 Hz の余弦波は f_N で折り返したときにちょうど重なっている. またそれぞれの余弦波は振幅が等しく逆位相であったので, エイリアシングノイズと 4.00 Hz の余弦波はちょうど打ち消し合うと考えられる. また 40.0 Hz の余弦波が折り返すことで $\pm F_N - (\pm 40 \mp F_N) = \mp 8$ Hz の余弦波が生じると考えられる. 実際にダウンサンプリング後の振幅スペクトルには 8.00 Hz のピークのみが現れ, 他の成分は消えている. さらにダウンサンプリングによりデータ数が 1/4 になったため, 周波数スペクトル $|Z_k|$ の値も 1/4 になっている.

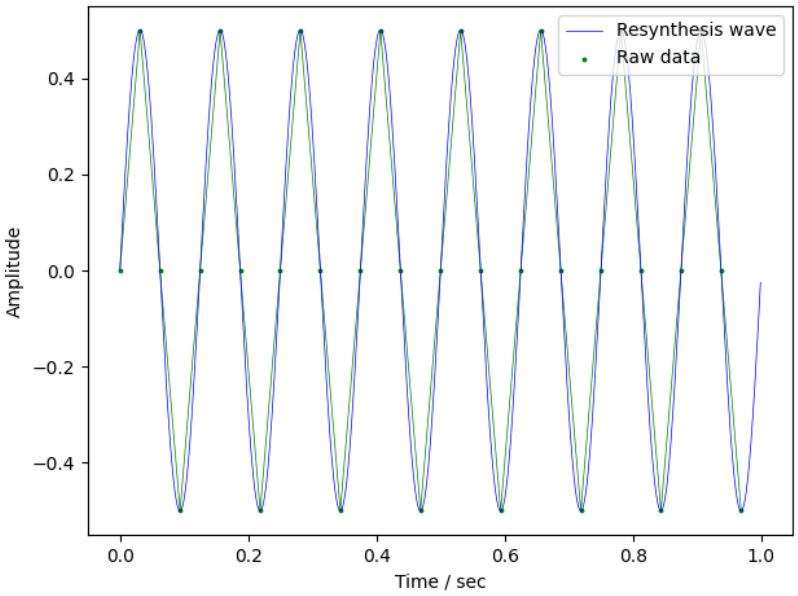

図 23　元データと再合成したデータ

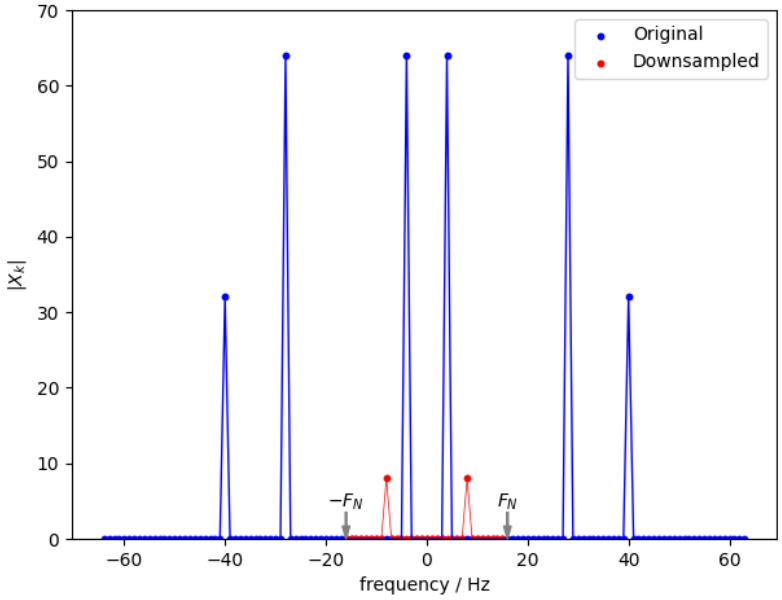

図 24　周波数スペクトル

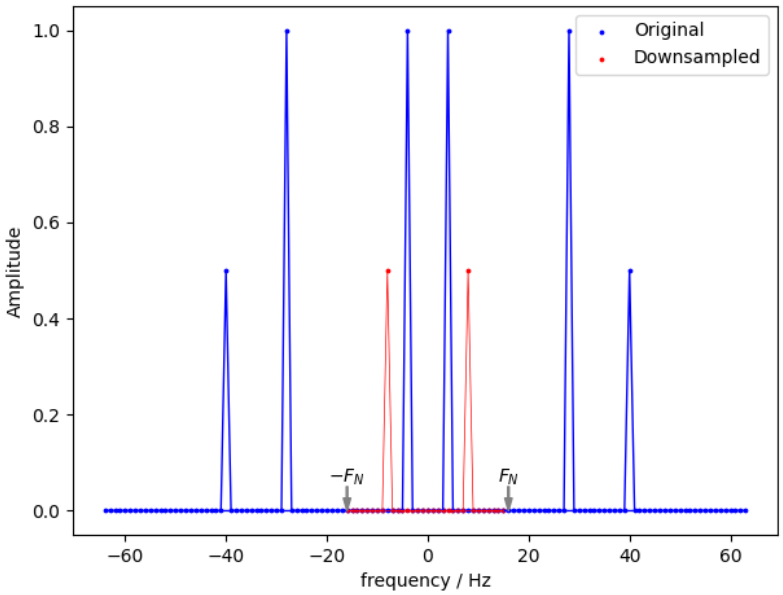

図 25　振幅スペクトル

表 7　各ピークの振幅

周波数 / Hz	振幅	位相 / °
8.00	0.500	-9.00×10^2

参考文献

[1] numpy. Discrete fourier transform (numpy.fft) — numpy v1.19 manual. `https://numpy.org/doc/stable/reference/routines.fft.html#module-numpy.fft`. (Accessed on 01/26/2021).

補遺 A　ソースコード

ソースコード A.1　FFT 用 python スクリプト

```
1  ############################################################
2  #
3  # Author: Ryosuke Sasaki (rsasaki@keio.jp)
4  # Date: 2021/1/26
5  #
6  ############################################################
7
8  import numpy as np
9  from scipy.signal import find_peaks
10 import matplotlib.pyplot as plt
11 import math
12
13 height_threshold = 0.1 # ピーク検出のしきい値
14 lower_threshold = 0.6 # 連続したピークを検出するための下限
15 num_sample = 1000 # 再合成のサンプル数
16 resample = 4 # ダウンサンプリングの回数
17 outdir = "kadai3-3" # 出力ディレクトリ
18 name = "kadai3" # ファイル名
19
20
21 #
22 # データの格納
23 #
24 raw = np.loadtxt('./data_' + name + '.csv', delimiter=',')
25 raw_x = raw[:, 0][::resample]
26 raw_y = raw[:, 1][::resample]
27 dt = raw_x[1] - raw_x[0] # 時間間隔
28 N = len(raw_x) # ダウンサンプル後のデータ数
29 N_raw = len(raw[:, 0]) # 元データ数
30
31 #
32 # FFT
33 #
34 fk = np.fft.fft(raw_y) # を計算 Fk
35 spec = np.vectorize(lambda x: np.abs(x))(fk) # スペクトルを計算
36 phase = np.vectorize(lambda x: np.angle(x))(fk) # 位相を計算
37 amp = np.vectorize(lambda x: np.abs(x)*2/N)(fk) # 振幅を計算
38 freq = np.fft.fftfreq(raw_y.size, dt) # 毎の周波数を計算 index
```

```
F_Nyq = abs(freq[int(N/2)]) # ナイキスト周波数

fk_raw = np.fft.fft(raw[:, 1]) # ダウンサンプル前の Fk
spec_raw = np.vectorize(lambda x: np.abs(x))(fk_raw)
amp_raw = np.vectorize(lambda x: np.abs(x)*2/N_raw)(fk_raw)
freq_raw = np.fft.fftfreq(raw[:, 1].size, raw[1, 0] - raw[0, 0])

# 極大値によるピークの検出
peaks, _ = find_peaks(amp[0:int(N / 2)], height=height_threshold)
# 連続したピークの検出
peaks_ = [int(i) for i in range(int(N/2)) if amp[i]>=0.6 and not i in peaks]
if len(peaks_) > 0:
    peaks = np.append(peaks, peaks_)


#
# 波形再合成
#
def f_resyn(x):
    ret = 0
    for i in peaks:
        ret += amp[i] * math.cos(2 * math.pi * freq[i] * x + phase[i])

    return ret
time = [dt*N/num_sample*i for i in range(num_sample)]
f = [f_resyn(dt*N/num_sample*i) for i in range(num_sample)]

#
# 周波数スペクトルを描画
#
fig0 = plt.figure()
ax = fig0.add_subplot(111)
ax.set_xlabel('frequency / Hz')
ax.set_ylabel('$|F_k|$')
lists = sorted(zip(*[freq, spec]))
freq_sorted, spec_sorted = list(zip(*lists))
fig0.subplots_adjust(left=0.1, right=0.95, bottom=0.1, top=0.95)
ax.plot(freq_sorted, spec_sorted, c="g", linewidth=0.5)
ax.scatter(freq_sorted, spec_sorted, s=10, c="g")
fig0.savefig(outdir+"/Spectrum-"+name+".png")

#
```

```
# 振幅の散布図を描画
#
fig1 = plt.figure()
ax = fig1.add_subplot(111)
ax.set_xlabel('frequency / Hz')
ax.set_ylabel('Amplitude')
ax.scatter(freq[peaks], amp[peaks], s=10, label="Peaks", c="b")
ax.scatter(freq[0:int(N / 2)], amp[0:int(N / 2)], s=3, label="Amplitude", c
    ="g")
ax.plot(freq[0:int(N / 2)], amp[0:int(N / 2)], c="g", linewidth=0.5)
fig1.subplots_adjust(left=0.1, right=0.95, bottom=0.1, top=0.95)
plt.legend(loc='best')
fig1.savefig(outdir+"/Amplitude-"+name+".png")

#
# 位相の散布図を描画
#
fig2 = plt.figure()
ax = fig2.add_subplot(111)
ax.set_xlabel('frequency / Hz')
ax.set_ylabel('Phase / degrees')
ax.scatter(freq[peaks], [np.degrees(x) for x in phase[peaks]], s=10, label="
    Peaks", c="b")
ax.scatter(freq[0:int(N / 2)], [np.degrees(x) for x in phase[0:int(N / 2)]],
     s=3, label="Phase", c="g")
fig2.subplots_adjust(left=0.1, right=0.95, bottom=0.1, top=0.95)
plt.legend(loc='best')
fig2.savefig(outdir+"/Phase-"+name+".png")

#
# 入力データ再合成波形の描画,
#
fig3 = plt.figure()
ax = fig3.add_subplot(111)
ax.set_xlabel("Time / sec")
ax.set_ylabel("Amplitude")
ax.plot(raw_x, raw_y, c="g", linewidth=0.5)
ax.scatter(raw_x, raw_y, c="g", s=3, label="Raw data")
ax.plot(time, f, c="b", linewidth=0.5, label="Resynthesis wave")
fig3.subplots_adjust(left=0.1, right=0.95, bottom=0.1, top=0.95)
#plt.legend(loc='best')
plt.legend(loc='upper right')
```

```
120 fig3.savefig(outdir+"/Wave-"+name+".png")
121
122 #
123 # 入力データの描画
124 #
125 fig4 = plt.figure()
126 ax = fig4.add_subplot(111)
127 ax.set_xlabel("Time / sec")
128 ax.set_ylabel("Amplitude")
129 ax.plot(raw_x, raw_y, c="g", linewidth=0.5)
130 ax.scatter(raw_x, raw_y, c="g", s=3, label="Raw data")
131 fig4.subplots_adjust(left=0.1, right=0.95, bottom=0.1, top=0.95)
132 plt.legend(loc='best')
133 fig4.savefig(outdir+"/WaveOrigin-"+name+".png")
134
135 #
136 # 比較用周波数スペクトル
137 #
138 fig5 = plt.figure()
139 ax = fig5.add_subplot(111)
140 ax.set_xlabel('frequency / Hz')
141 ax.set_ylabel('$|F_k|$')
142 plt.ylim(0, 70)
143 fig5.subplots_adjust(left=0.1, right=0.95, bottom=0.1, top=0.95)
144
145 ax.annotate('', xy=[F_Nyq, 0], xytext=[F_Nyq, 3.5],
146             arrowprops=dict(shrink=0, width=0.7, headwidth=4,
147                             headlength=7, connectionstyle='arc3',
148                             facecolor='gray', edgecolor='gray'))
149 ax.annotate('', xy=[-F_Nyq, 0], xytext=[-F_Nyq, 3.5],
150             arrowprops=dict(shrink=0, width=0.7, headwidth=4,
151                             headlength=7, connectionstyle='arc3',
152                             facecolor='gray', edgecolor='gray'))
153 ax.text(F_Nyq, 4.2, "$F_N$", size=10, horizontalalignment="center")
154 ax.text(-F_Nyq, 4.2, "$-F_N$", size=10, horizontalalignment="center")
155
156 lists = sorted(zip(*[freq_raw, spec_raw]))
157 freq_raw_sorted, spec_raw_sorted = list(zip(*lists))
158 ax.plot(freq_raw_sorted, spec_raw_sorted, c="b", linewidth=0.5)
159 ax.scatter(freq_raw_sorted, spec_raw_sorted, s=10, c="b", label="Original")
160
161 if resample > 1:
```

```
162     lists = sorted(zip(*[freq, spec]))
163     freq_sorted, spec_sorted = list(zip(*lists))
164     ax.plot(freq_sorted, spec_sorted, c="g", linewidth=0.5)
165     ax.scatter(freq_sorted, spec_sorted, s=10, c="g", label="Downsampled")
166     plt.legend(loc='best')
167
168 fig5.savefig(outdir+"/Spectrums-"+name+".png")
169
170 #
171 # 比較用振幅スペクトル
172 #
173 fig6 = plt.figure()
174 ax = fig6.add_subplot(111)
175 ax.set_xlabel('frequency / Hz')
176 ax.set_ylabel('Amplitude')
177
178 ax.annotate('', xy=[F_Nyq, 0], xytext=[F_Nyq, 0.05],
179             arrowprops=dict(shrink=0, width=0.7, headwidth=4,
180                             headlength=7, connectionstyle='arc3',
181                             facecolor='gray', edgecolor='gray'))
182 ax.annotate('', xy=[-F_Nyq, 0], xytext=[-F_Nyq, 0.05],
183             arrowprops=dict(shrink=0, width=0.7, headwidth=4,
184                             headlength=7, connectionstyle='arc3',
185                             facecolor='gray', edgecolor='gray'))
186 ax.text(F_Nyq, 0.06, "$F_N$", size=10, horizontalalignment="center")
187 ax.text(-F_Nyq, 0.06, "$-F_N$", size=10, horizontalalignment="center")
188
189 ax.scatter(freq_raw[0:int(N_raw)], amp_raw[0:int(N_raw)], s=3, label="
        Original", c="b")
190 ax.plot(freq_raw[0:int(N_raw)], amp_raw[0:int(N_raw)], c="b", linewidth=0.5)
191
192 if resample > 1:
193     ax.scatter(freq[0:int(N)], amp[0:int(N)], s=3, label="Downsampled", c="g
            ")
194     ax.plot(freq[0:int(N)], amp[0:int(N)], c="g", linewidth=0.5)
195     plt.legend(loc='best')
196
197 fig6.subplots_adjust(left=0.1, right=0.95, bottom=0.1, top=0.95)
198 fig6.savefig(outdir+"/Amplitudes-"+name+".png")
199
200 #
201 # ダウンサンプリングしたデータを保存
```

```
202 #
203 csv = np.stack([raw_x, raw_y], 1)
204 np.savetxt(outdir + "/" + name + "-" + "downsample" + ".csv", csv, fmt='%.10
        f')
205
206 #
207 # 結果の出力
208 #
209 print("Nyquist frequency:", F_Nyq)
210 print("peaks:")
211 for i in peaks:
212     print("Frequency:", freq[i], " Amplitude:",
213           amp[i], " Phase(degrees):", np.degrees(phase[i]))
214
215 with open(outdir + "/" + name + ".txt", 'w') as f:
216     print("Nyquist frequency:", abs(freq[int(N / 2)]), file=f)
217     print("peaks:", file=f)
218     for i in peaks:
219         print("Frequency:", freq[i], " Amplitude:", amp[i],
220               " Phase(degrees):", np.degrees(phase[i]), file=f)
```

表 A.1: ダウンサンプリングしたデータ y_j

時刻 t	データ y_j
0.0000000000	0.0000000000
0.0156250000	1.4942056740
0.0312500000	0.5000000000
0.0468750000	0.4118134740
0.0625000000	0.0000000000
0.0781250000	-0.4118134740
0.0937500000	-0.5000000000
0.1093750000	-1.4942056740
0.1250000000	-0.0000000000
0.1406250000	-2.2013124560
0.1562500000	0.5000000000
0.1718750000	-1.1189202550
0.1875000000	0.0000000000
0.2031250000	1.1189202550
0.2187500000	-0.5000000000
0.2343750000	2.2013124560
0.2500000000	-0.0000000000
0.2656250000	1.4942056740
0.2812500000	0.5000000000
0.2968750000	0.4118134740
0.3125000000	0.0000000000
0.3281250000	-0.4118134740
0.3437500000	-0.5000000000
0.3593750000	-1.4942056740
0.3750000000	-0.0000000000
0.3906250000	-2.2013124560
0.4062500000	0.5000000000
0.4218750000	-1.1189202550
0.4375000000	-0.0000000000
0.4531250000	1.1189202550
0.4687500000	-0.5000000000
0.4843750000	2.2013124560

表は次ページに続く

前ページからの続き

時刻 t	データ y_j
0.5000000000	0.0000000000
0.5156250000	1.4942056740
0.5312500000	0.5000000000
0.5468750000	0.4118134740
0.5625000000	-0.0000000000
0.5781250000	-0.4118134740
0.5937500000	-0.5000000000
0.6093750000	-1.4942056740
0.6250000000	0.0000000000
0.6406250000	-2.2013124560
0.6562500000	0.5000000000
0.6718750000	-1.1189202550
0.6875000000	-0.0000000000
0.7031250000	1.1189202550
0.7187500000	-0.5000000000
0.7343750000	2.2013124560
0.7500000000	-0.0000000000
0.7656250000	1.4942056740
0.7812500000	0.5000000000
0.7968750000	0.4118134740
0.8125000000	0.0000000000
0.8281250000	-0.4118134740
0.8437500000	-0.5000000000
0.8593750000	-1.4942056740
0.8750000000	-0.0000000000
0.8906250000	-2.2013124560
0.9062500000	0.5000000000
0.9218750000	-1.1189202550
0.9375000000	0.0000000000
0.9531250000	1.1189202550
0.9687500000	-0.5000000000
0.9843750000	2.2013124560

これで終わり

表 A.2　ダウンサンプリングしたデータ z_j

時刻 t	データ y_j
0.0000000000	0.0000000000
0.0312500000	0.5000000000
0.0625000000	0.0000000000
0.0937500000	-0.5000000000
0.1250000000	-0.0000000000
0.1562500000	0.5000000000
0.1875000000	0.0000000000
0.2187500000	-0.5000000000
0.2500000000	-0.0000000000
0.2812500000	0.5000000000
0.3125000000	0.0000000000
0.3437500000	-0.5000000000
0.3750000000	-0.0000000000
0.4062500000	0.5000000000
0.4375000000	-0.0000000000
0.4687500000	-0.5000000000
0.5000000000	0.0000000000
0.5312500000	0.5000000000
0.5625000000	-0.0000000000
0.5937500000	-0.5000000000
0.6250000000	0.0000000000
0.6562500000	0.5000000000
0.6875000000	-0.0000000000
0.7187500000	-0.5000000000
0.7500000000	-0.0000000000
0.7812500000	0.5000000000
0.8125000000	0.0000000000
0.8437500000	-0.5000000000
0.8750000000	-0.0000000000
0.9062500000	0.5000000000
0.9375000000	0.0000000000
0.9687500000	-0.5000000000