演習 2

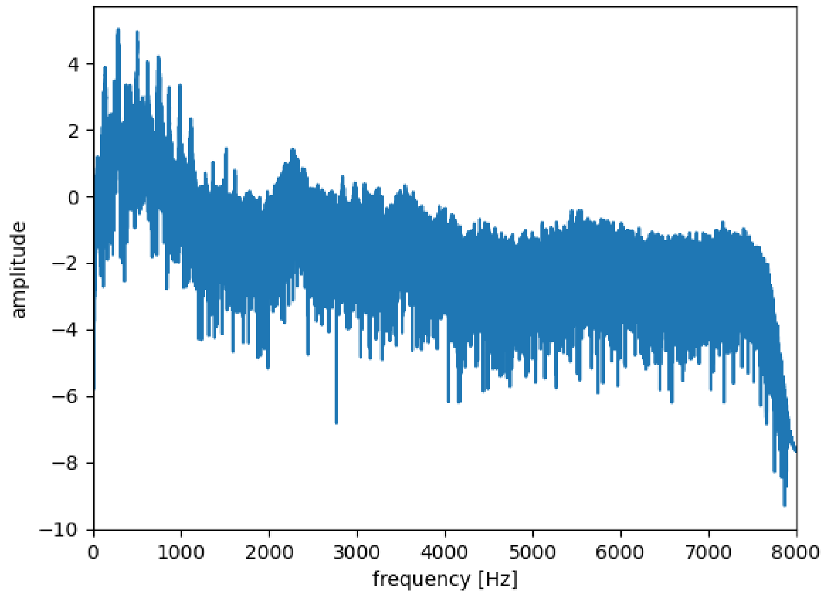

図 1: aiueo.wav の波形とスペクトル

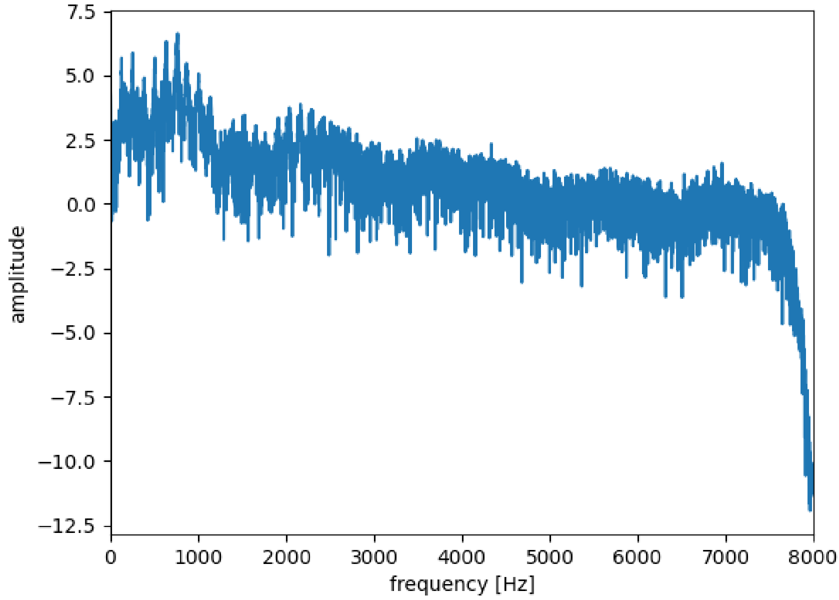

図 2: a.wav の波形とスペクトル

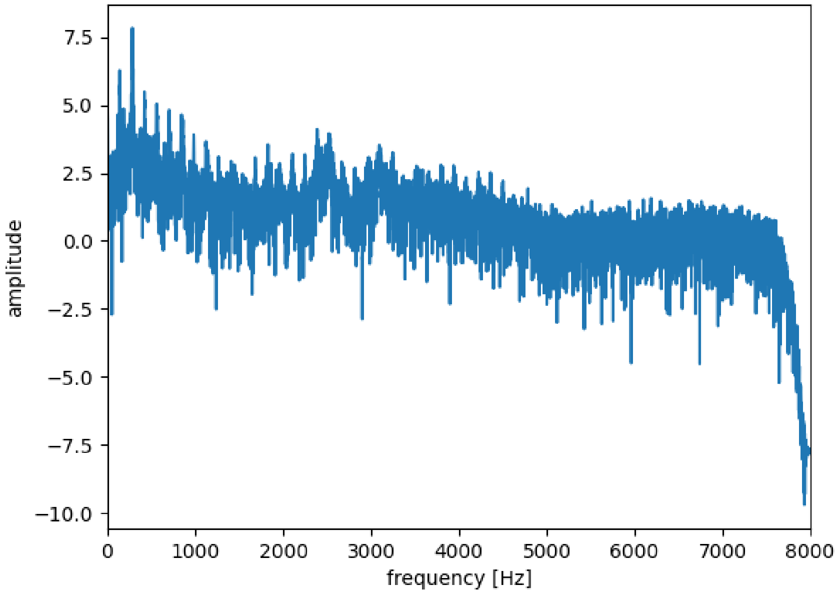

図 3: i.wav の波形とスペクトル

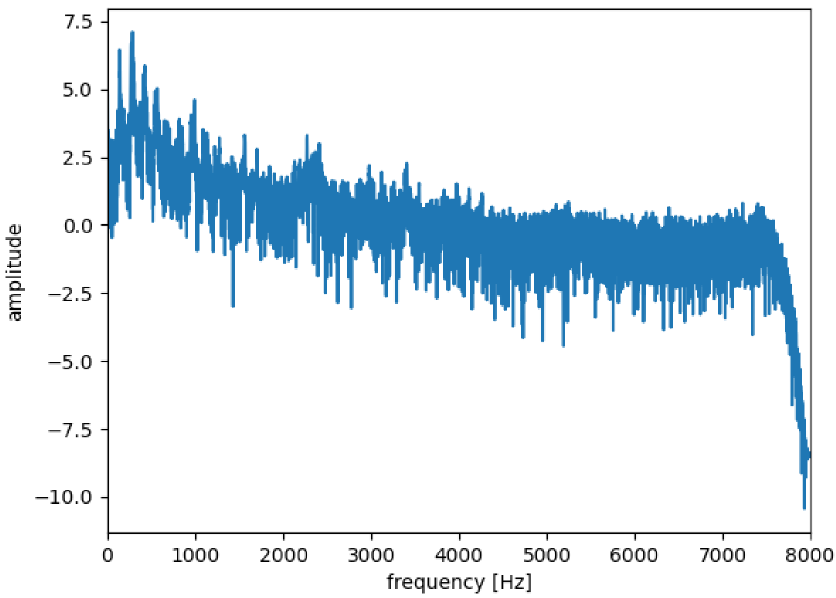

図 4: u.wav の波形とスペクトル

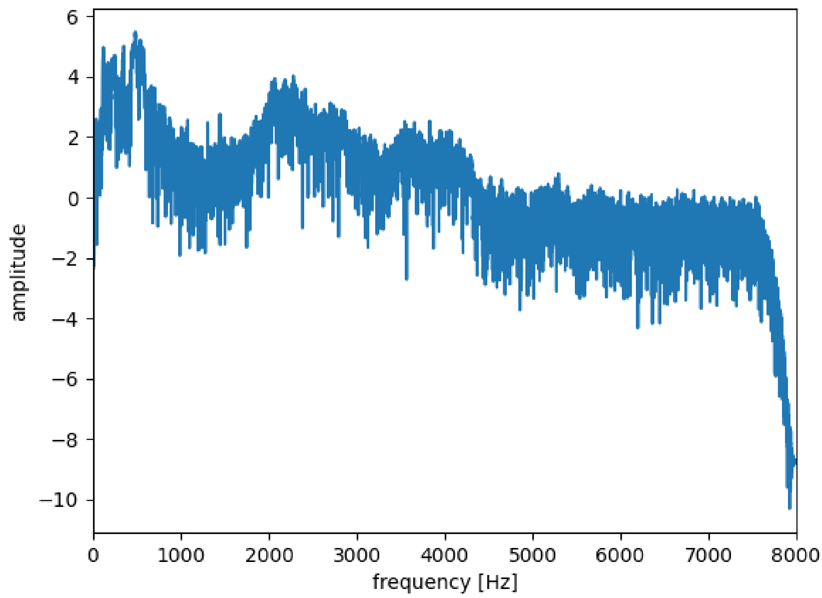

図 5: e.wav の波形とスペクトル

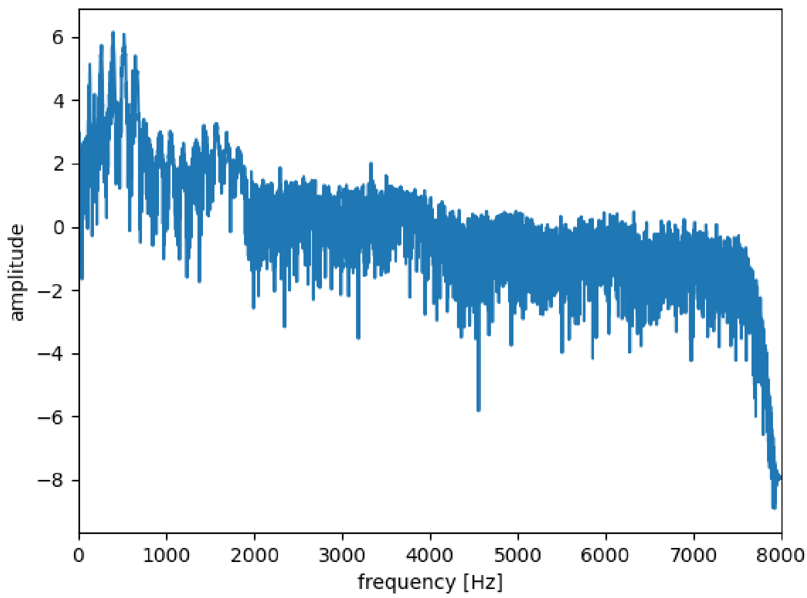

図 6: o.wav の波形とスペクトル

演習 3

実数入力に対する 1 次元離散フーリエ変換を計算するするライブラリ．高速フーリエ変換（FFT）によって、実数値配列の 1 次元 n 点離散フーリエ変換（DFT）を計算する．入力は実数値配列で、出力は複素数値配列である．入力には他にも n(使用する入力内の変換軸に沿ったポイントの数, optional), axis(変換を行う軸, optional) を指定できる．

※純粋な実数入力に対して DFT を計算すると、出力はエルミート対称 (その成分は任意の添字 i, j について (i, j) 成分は (j, i) 成分の複素共役と等しい.) になる．つまり、負の周波数項は対応する正の周波数項の複素共役にすぎない．したがって負の周波数項は冗長になるので RFFT では負の周波数項を計算

しない．その結果、出力の軸の長さは $\lfloor n/2 \rfloor + 1$ になる．

演習 4

バタフライ演算

DFT について　そもそも、DFT では N 点の実変数 $f(0), f(1), ..., f(N-1)$ を離散フーリエ変数 $F(0), F(1), ..., F(N-1)$ に変換するために N 次正方行列である DFT 行列を掛け合わせているのだった．この計算では，$O(N^2)$ となる．この計算量を軽減するために、高速フーリエ変換（FFT）が考案された．

FFT の計算量　上述の通り DFT では計算量が多いので、バタフライ演算を用いて計算量を減らしている．

FFT の実践 (手計算)

$$\text{input} = (1,0,3,2,4,0,2,0)^T$$

$$(\text{略}) \tag{1}$$

以下で計算が正しいことを検証する．

Listing 1: FFT の実践 (numpy 篇)

```
>>> import numpy as np
>>> np.fft.fft([1,0,3,2,4,0,2,0])
array([12. +0.j , -4.41421356-2.41421356j,
      0. +2.j , -1.58578644-0.41421356j,
      8. +0.j , -1.58578644+0.41421356j,
      0. -2.j , -4.41421356+2.41421356j])
```