画像処理技術における信号処理技術の応用

確率解析信号特論 課題レポート

創成科学研究科 基盤科学系専攻 情報コース
20-8801-037-4
寄元康平

第 1 章

はじめに

信号処理技術とは，信号データの計測や解析，また解析するための技術を指す．具体的には，音声や回路の電流値といったような計測器から計測されたアナログデータをデジタルデータへサンプリング処理，データに含まれている周波数成分の解析やデータの特徴となる周波数領域の抽出，また計測時に生じたノイズを除去するためのフィルタ処理が挙げられ，デジタルデータの前処理や音声の合成などに応用されている．一方で，画像処理技術とは，画像における情報の解析，抽出や必要に応じた加工などのような画像データに対して行われる処理技術の総称である．具体的な例としては画像内のノイズ除去，輪郭の抽出，拡大や縮小，パターンマッチングなどが挙げられる．これらの技術はレントゲン写真，CT スキャンの取得や解析といった医療現場，工場での製品の異常検知，また撮影した画像に対しての編集といったような我々の身の回りの画像に対して幅広く活用されている．最近では，コンピュータの発展に伴い機械学習技術を導入したより高度でかつ高精度な画像処理技術が研究，開発されている．信号処理技術には，データ内のノイズ除去や必要となる特徴の抽出などのように画像処理技術と共通する技術がいくつか存在する．このことから信号処理技術に用いられているフィルタ処理を画像処理に活用できるではないかと考えた．

本レポートでは，信号処理技術における一部のフィルタ処理を画像データに対して活用する実験，検証を行った．

第 2 章

信号処理技術

本章では時間領域の離散データを周波数領域の離散データに変換する離散フーリエ変換，また時間領域のデータから特定の周波数領域を抽出するフィルタ処理であるバンドパスフィルタについて紹介する．

2.1　離散フーリエ変換

音声や電流といった信号を計測したとき，それらの値は時間領域における電気信号の連続データ (アナログデータ) として出力される．連続データはコンピュータ上で扱うことができないので，連続データをある一定間隔でサンプリングすることでコンピュータ上で扱える離散データ (デジタルデータ) へ変換することでコンピュータ上でのデータの取得，解析が可能となる (図 2.1).

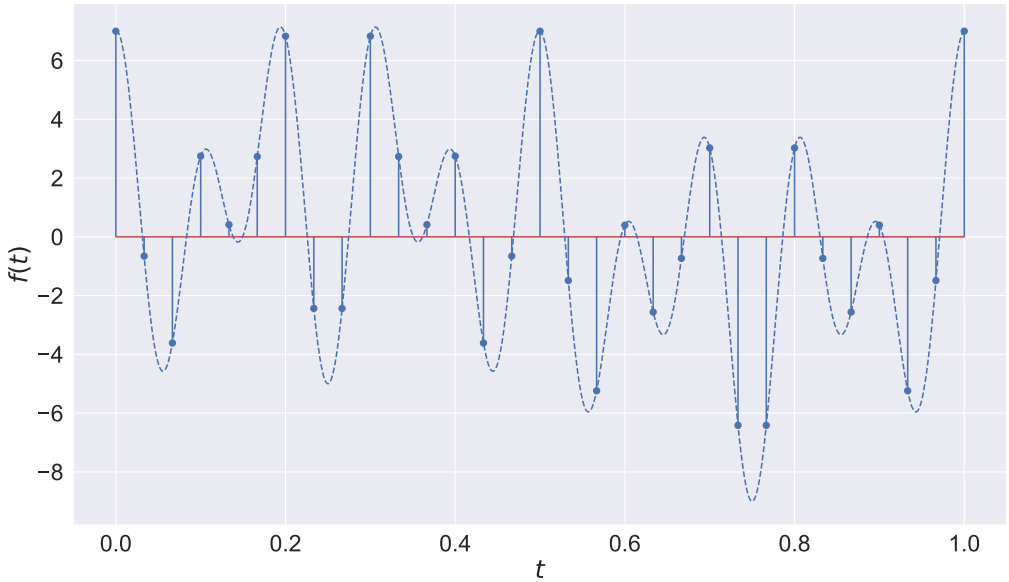

図 2.1　離散データの一例

この離散データに含まれている周波数成分を算出する手法として離散フーリエ変換が存在する．離散フーリ

エ変換とは，式 (2.1) で定義される手法であり，時間領域の離散データ $f(t)$ を周波数領域の離散データ $F(\omega)$ へ変換することができる．

$$F(\omega) = \sum_{t=0}^{N-1} f(t) e^{-j\frac{2\pi\omega}{N}t} \tag{2.1}$$

このとき，式 (2.1) における N は $f(t)$ におけるデータ数である．式 (2.1) では，時間領域の実空間データを複素数空間における三角関数の総和の形へ変換することにより，時間領域のデータに含まれている周波数成分を算出できる (図 2.2)． 図 2.2は 図 2.1を離散フーリエ変換したときの周波数領域データを示している．これより， 図 2.1のデータは，1Hz, 6Hz, 10Hz の周波数を含むことがわかる．

また，周波数領域 $F(\omega)$ は離散フーリエ逆変換 (式 (2.2)) を用いることで時間領域の離散データ $f(t)$ へ復元することができる (図 2.3)．

$$f(t) = \frac{1}{N} \sum_{\omega=0}^{N-1} F(\omega) e^{j\frac{2\pi t}{N}\omega} \tag{2.2}$$

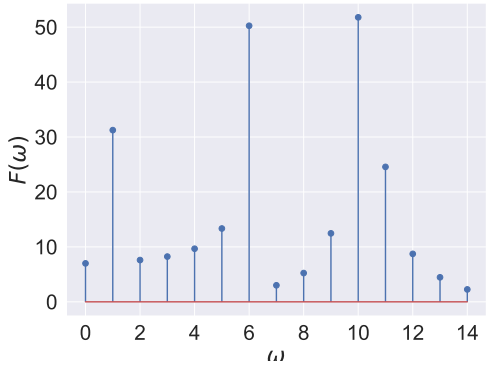

図 2.2　図 2.1を式 (2.1) を用いて周波数領域の離散データに変換

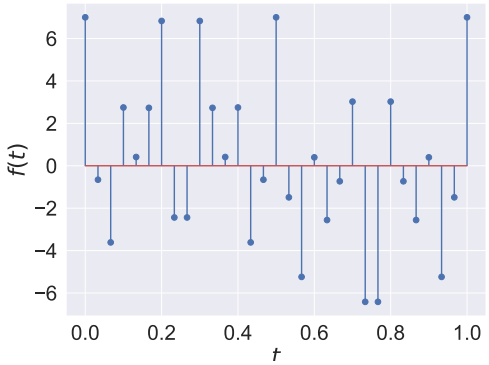

図 2.3　図 2.2を式 (2.2) を用いて時間領域の離散データに逆変換

2.2　バンドパスフィルタ

離散フーリエ変換によって算出された周波数領域のデータにおいて，ある範囲の周波数成分のみを抽出し時間領域のデータに逆変換するフィルタ処理のことをバンドパスフィルタという．このとき，抽出する周波数成分の閾値のことをカットオフ周波数という．バンドパスフィルタは，計測したデータにノイズが含まれている場合や任意の周波数を含む信号のみを計測する場合などに活用される． 図 2.4，図 2.5にバンドパスフィルタを 図 2.1に対して行った例を示す． 図 2.4では，バンドパスフィルタを用いて $3 \leq \omega \leq 7$ の周波数を抽出しており， 図 2.5において 6Hz のみのデータが抽出できていることがわかる．

バンドパスフィルタにおいて，単一のカットオフ周波数以上の範囲のみを抽出し高周波領域のデータを抽出するフィルタ処理をハイパスフィルタという． 図 2.4，図 2.5にハイパスフィルタを 図 2.1に対して行った例を示す． 図 2.4では，ハイパスフィルタを用いて $7 \leq \omega \leq 14$ の周波数を抽出しており， 図 2.5において 10Hz のみのデータが抽出できていることがわかる．

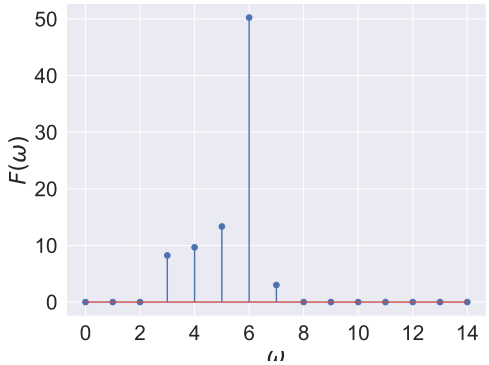

図 2.4　図 2.2に対してバンドパスフィルタを行った場合 (周波数領域)

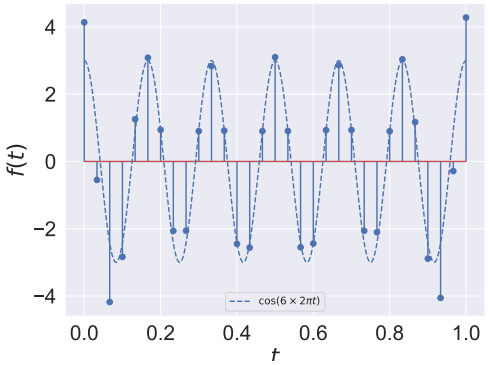

図 2.5　図 2.2に対してバンドパスフィルタを行った場合 (時間領域)

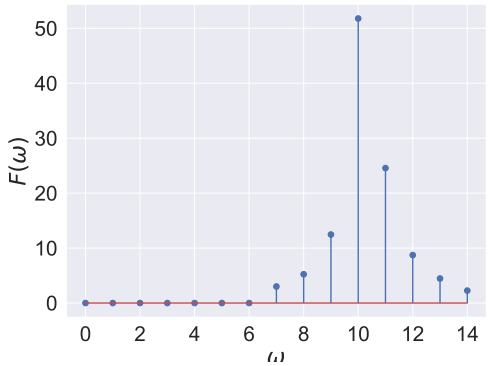

図 2.6　図 2.2に対してハイパスフィルタを行った場合 (周波数領域)

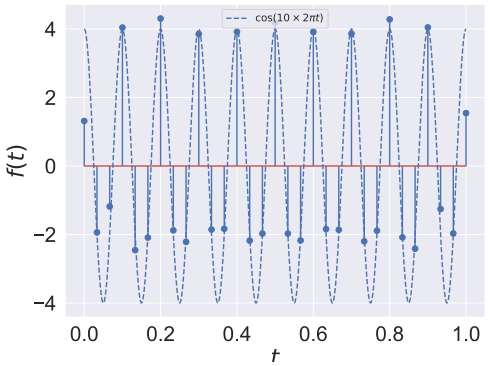

図 2.7　図 2.2に対してハイパスフィルタを行った場合 (時間領域)

また，カットオフ周波数以下の範囲のみを抽出し高周波領域のデータを抽出するフィルタ処理をローパスフィルタという． 図 2.8，図 2.9にローパスフィルタを 図 2.1に対して行った例を示す． 図 2.8では，バンドパスフィルタを用いて $0 \leq \omega \leq 3$ の周波数を抽出しており， 図 2.9において 1Hz のみのデータが抽出できていることがわかる．

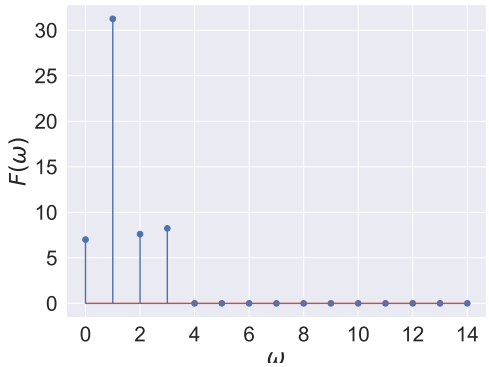

図 2.8　図 2.2に対してローパスフィルタを行った場合 (周波数領域)

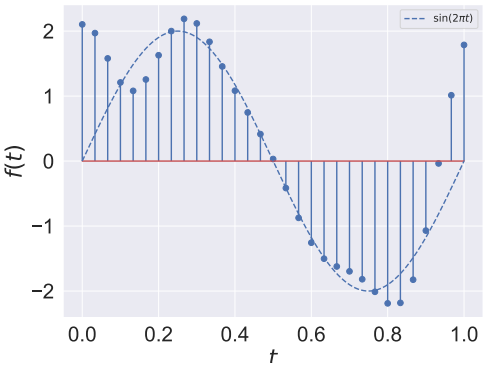

図 2.9　図 2.2に対してローパスフィルタを行った場合 (時間領域)

第 3 章

信号処理技術の画像データへの応用

本章では，離散フーリエ変換の画像データへの応用について紹介し，また信号処理におけるフィルタ処理の一部を画像データへ活用する．なお，実装する環境を以下に示す．

- OS: Windows10
- プログラム言語: Python 3.6
- ライブラリ
  - NumPy: 数値計算ライブラリ
  - OpenCV: 画像処理ライブラリ
  - MatPlotLib: 画像出力ライブラリ

3.1 離散フーリエ変換の画像データへの応用

これまでの離散フーリエ変換 (式 (2.1)) は時間空間の 1 次元離散データにおける周波数成分を抽出し，離散逆フーリエ変換 (式 (2.2)) を用いて時間領域のデータへ復元していた．この 1 次元データ上での離散フーリエ変換を画像データへ応用するには，画像を 2 次元の時間空間における離散データ $f(x, y)$ とみなし，x, y それぞれの方向に対して離散フーリエ変換 (式 (3.1)) することで，2 次元の周波数成分 $F(u, v)$ を算出できる．

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)j} \tag{3.1}$$

図 3.1， 図 3.2 に画像データへ離散フーリエ変換を行った例を示す． 図 3.1は入力画像であり， 図 3.2は入力画像の周波数領域である．この図では，図の中心に近い領域は低周波成分を示し，中心から離れていくほど高周波成分を示している．**??**に対して信号処理にて用いられているバンドパスフィルタを用いることで信号処理技術でのフィルタ処理を画像データに応用できる．

離散フーリエ逆変換も式 (3.1) と同様に (u, v) それぞれの方向に対して逆変換を行うことで 図 3.2から元の画像へ復元できる (式 (3.2))．

$$f(x,y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) e^{2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)j} \tag{3.2}$$

図 3.1　入力画像

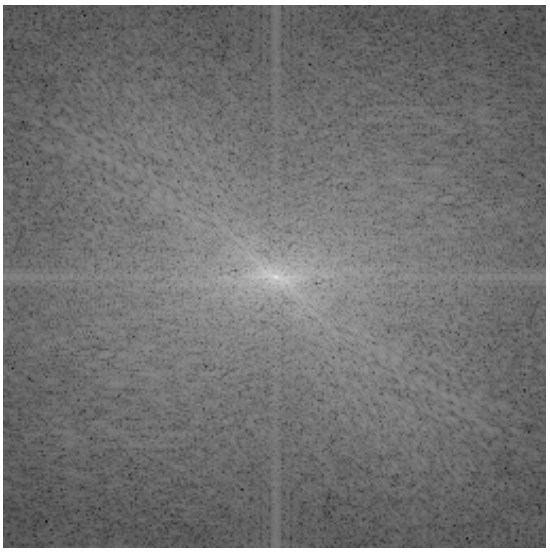

図 3.2　入力画像の周波数成分

3.2　ハイパスフィルタによる画像処理

Listing 3.1に，画像に対してハイパスフィルタを行う high_pass_img() 関数を示す．Listing 3.1では，入力された画像から離散フーリエ変換を用いて周波数成分を抽出し，x, y それぞれのカットオフ周波数より低い周波数領域に 0 を代入したのちに離散フーリエ逆変換を用いて画像へと復元する．ただし，処理時間の都合上，離散フーリエ変換，逆変換は NumPy ライブラリにて実装されている高速フーリエ変換を用いる．

Listing 3.1　画像へのハイパスフィルタを行う high_pass_img() 関数

```
1  # coding: utf-8
2
3
4  import cv2
5  import numpy as np
6  import matplotlib.pyplot as plt
7
8
9  def high_pass_img(img, cut_x, cut_y):
10     '''画像へのハイパスフィルタの実装
11
12
13
14     Parameters
15     ----------
16     img: 入力する画像
17     cut_x : 座標のカットオフ周波数 x
18     cut_y : 座標のカットオフ周波数 y
```

```
19
20      Returns
21      -------
22      idft_img: ハイパスフィルタを行った画像
23      '''
24
25      img = np.array(img, np.float32)
26
27      spectral = np.fft.fft2(img) # 高速フーリエ変換による周波数領域への変換
28      spectral = np.fft.fftshift(spectral) # 低周波成分を画像中心になるようにシフト移動
29      spectral[cut_x: -cut_x] = 0 # ハイパスフィルタによる処理（方向 x）
30      spectral[:, cut_y: -cut_y] = 0 # ハイパスフィルタによる処理（方向 y）
31
32      spectral = np.fft.ifftshift(spectral) # 周波数成分の並びを戻す
33      idft_img = np.fft.ifft2(spectral) # 高速フーリエ逆変換
34      idft_img = idft_img.real # 出力する実部
35
36      return idft_img
```

図 3.3に入力画像，図 3.4に**??**に対してハイパスフィルタで処理した後の周波数成分，図 3.5に復元した出力画像を示す．図 3.5より，画像内の顔の輪郭や髪の毛などといったよなピクセル間の温度差の激しい部分，すなわちエッジが抽出できており，逆に帽子の模様や鏡の縁のようなピクセル間の温度差が類似している部分が除去されている．このことから，画像データに対してハイパスフィルタを用いることにより画像のエッジ抽出として活用することができる．

図 3.3　入力画像

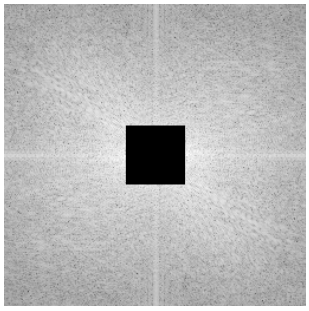

図 3.4　ハイパスフィルタによる抽出

図 3.5　処理後の画像

3.3　ローパスフィルタのによる画像処理

Listing 3.2に，画像に対してローパスフィルタを行う low_pass_img() 関数を示す．Listing 3.2では，Listing 3.2と同様に入力された画像の周波数成分を抽出し，x, y それぞれのカットオフ周波数より高い周波数領域に 0 を代入したのちに逆変換を用いて画像へと復元する．

Listing 3.2　画像へのローパスフィルタを行う low_pass_img() 関数

```python
# coding: utf-8


import cv2
import numpy as np
import matplotlib.pyplot as plt


def low_pass_img(img, cut_x, cut_y):
    '''画像へのローパスフィルタの実装


    Parameters
    ----------
    img: 入力する画像
    cut_x : 座標のカットオフ周波数 x
    cut_y : 座標のカットオフ周波数 y

    Returns
    -------
    idft_img: ローパスフィルタを行った画像
    '''

    img = np.array(img, np.float32)

    spectral = np.fft.fft2(img) # 高速フーリエ変換による周波数領域への変換
    spectral = np.fft.fftshift(spectral) # 低周波成分を画像中心になるようにシフト移動

    spectral[:cut_x] = 0 # ローパスフィルタによる処理（方向 x）
    spectral[-cut_x:] = 0
    spectral[:, :cut_y] = 0 # ローパスフィルタによる処理（方向 x）
    spectral[:, -cut_y:] = 0

    spectral = np.fft.ifftshift(spectral) # 周波数成分の並びを戻す
    idft_img = np.fft.ifft2(spectral) # 高速フーリエ逆変換
    idft_img = idft_img.real # 出力する実部
    return idft_img
```

図 3.6に入力画像， 図 3.7に??に対してローパスフィルタで処理した後の周波数成分， 図 3.8に復元した出力画像を示す． 図 3.8と 図 3.6を比較すると，画像内の輪郭がぼやけており鮮明さが失われていることがわかる．このことから，画像データに対してローパスフィルタを用いると画像に対してぼかしをかける処理として活用することができる．

図 3.6　入力画像

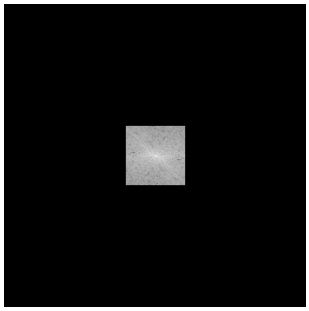

図 3.7　ローパスフィルタによる抽出

図 3.8　処理後の画像

第 4 章

結論

本レポートでは，信号処理技術に用いられている離散フーリエ変換，フィルタ処理の一部を画像データに活用する実験を行った．結果として，データの高周波成分を抽出するハイパスフィルタは画像処理においてはエッジ抽出として活用でき，データの低周波成分を抽出するローパスフィルタは画像処理におけるぼかしをかける処理として活用できることがわかった．