# 統計的機械学習
# 第五回　レポート

37-196360　森田涼介

2019 年 5 月 15 日

# 宿題 1

原信号 $\{\boldsymbol{s}_i\}_{i=1}^n$ $\left(\boldsymbol{s}_i \in \mathbb{R}^d\right)$，観測信号 $\{\boldsymbol{x}_i\}_{i=1}^n$ $\left(\boldsymbol{x}_i \in \mathbb{R}^d\right)$ について，混合行列 $\boldsymbol{M}$ $(\in \mathbb{R}^{d\times d})$ を用いて，

$$\boldsymbol{x}_i = \boldsymbol{M}\boldsymbol{s}_i \tag{1}$$

が成立するときを考える。ここで，原信号 $\{\boldsymbol{s}_i\}_{i=1}^n$ は i.i.d，期待値 $\boldsymbol{\mu}_s = \boldsymbol{0}$ で，共分散行列が単位行列 $\boldsymbol{\Sigma}_s = \boldsymbol{I}_d$ であり，また，$\boldsymbol{M}$ は逆行列を持つと仮定する（独立成分分析）。

観測信号を球状化（白色化）することを考える。観測信号の期待値と共分散行列は次のようになる。

$$\begin{aligned}
\boldsymbol{\mu}_x &= \frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i && (2)\\
&= \frac{1}{n}\sum_{i=1}^n \boldsymbol{M}\boldsymbol{s}_i && (3)\\
&= \boldsymbol{M}\frac{1}{n}\sum_{i=1}^n \boldsymbol{s}_i && (4)\\
&= \boldsymbol{M}\boldsymbol{\mu}_s && (5)\\
&= \boldsymbol{0} && (6)
\end{aligned}$$

$$\begin{aligned}
\boldsymbol{\Sigma}_x &= \frac{1}{n}\sum_{i=1}^n (\boldsymbol{x}_i - \boldsymbol{\mu}_x)^{\mathrm{T}}(\boldsymbol{x}_i - \boldsymbol{\mu}_x) && (7)\\
&= \frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_i && (8)\\
&\equiv \boldsymbol{C} && (9)
\end{aligned}$$

これより，$\boldsymbol{\Sigma}_x = \boldsymbol{C}$ が逆行列を持つことを仮定し，$\boldsymbol{x}_i$ を白色化すると，

$$\begin{aligned}
\tilde{\boldsymbol{x}}_i &= \boldsymbol{\Sigma}^{-\frac{1}{2}}(\boldsymbol{x}_i - \boldsymbol{\mu}_x) && (10)\\
&= \boldsymbol{C}^{-\frac{1}{2}}\boldsymbol{x}_i && (11)
\end{aligned}$$

また，式 (1) の両辺に左側から $\boldsymbol{C}^{-1/2}$ をかけることで，

$$\begin{aligned}
\tilde{\boldsymbol{x}}_i &= \tilde{\boldsymbol{M}}\boldsymbol{s}_i && (12)\\
\tilde{\boldsymbol{M}} &= \boldsymbol{C}^{-\frac{1}{2}}\boldsymbol{M} && (13)
\end{aligned}$$

となる。

白色化された $\tilde{\boldsymbol{x}}_i$（平均 $\boldsymbol{0}$）の共分散行列を考える。$\boldsymbol{s}_i$ は平均 $\boldsymbol{0}$，共分散行列 $\boldsymbol{I}_d$ であることに注意しつつ，式

(12) を用いれば，

$$\begin{aligned}
\boldsymbol{\Sigma}_{\tilde{x}} &= \frac{1}{n}\sum_{i=1}^{n}\tilde{\boldsymbol{x}}_i\tilde{\boldsymbol{x}}_i^{\mathrm{T}} && (14)\\
&= \frac{1}{n}\sum_{i=1}^{n}(\tilde{\boldsymbol{M}}\boldsymbol{s}_i)(\boldsymbol{s}_i^{\mathrm{T}}\tilde{\boldsymbol{M}}^{\mathrm{T}}) && (15)\\
&= \tilde{\boldsymbol{M}}\left(\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{s}_i\boldsymbol{s}_i^{\mathrm{T}}\right)\tilde{\boldsymbol{M}}^{\mathrm{T}} && (16)\\
&= \tilde{\boldsymbol{M}}\boldsymbol{\Sigma}_s\tilde{\boldsymbol{M}}^{\mathrm{T}} && (17)\\
&= \tilde{\boldsymbol{M}}\boldsymbol{I}_d\tilde{\boldsymbol{M}}^{\mathrm{T}} && (18)\\
&= \tilde{\boldsymbol{M}}\tilde{\boldsymbol{M}}^{\mathrm{T}} && (19)
\end{aligned}$$

白色化より，当然これは単位行列 $\boldsymbol{I}_d$ となる。結局，

$$\tilde{\boldsymbol{M}}\tilde{\boldsymbol{M}}^{\mathrm{T}} = \boldsymbol{I}_d \tag{20}$$

を得る。式 (20) は $\tilde{\boldsymbol{M}}$ が直交行列であることを示している。つまり，

$$\tilde{\boldsymbol{M}}^{-1} = \tilde{\boldsymbol{M}}^{\mathrm{T}} \tag{21}$$

が成立する。これより，次式が成立する。

$$\tilde{\boldsymbol{M}}\tilde{\boldsymbol{M}}^{\mathrm{T}} = \tilde{\boldsymbol{M}}^{\mathrm{T}}\tilde{\boldsymbol{M}} = \boldsymbol{I}_d \tag{22}$$

# 宿題 2

射影追跡の近似ニュートンアルゴリズムを実装する。

## 理論

i.i.d な標本 $\{\boldsymbol{x}_i\}_{i=1}^n$, $(\boldsymbol{x}_i \in \mathbb{R}^d)$ に対して射影追跡を行うことを考える。標本を中心化・球状化したものを $\{\tilde{\boldsymbol{x}}_i\}_{i=1}^n$ とし，以降はこれについて考える。$\boldsymbol{b}$ $(\in \mathbb{R}^d)$ によって射影するとすれば，射影方向 $\boldsymbol{b}$ の非ガウス性は

$$s = \langle \boldsymbol{b}, \tilde{\boldsymbol{x}} \rangle \qquad (||\boldsymbol{b}|| = 1) \tag{23}$$

によって測ることができる。また，非ガウス性尺度を $G(s)$ の期待値 $E[G(s)]$ とする。これを最大化するような $\boldsymbol{b}$ が求まればよい。なお，$G(s)$ としては様々な関数が考えられるが，例えば $G(s) = s^4$ とすれば，これは尖度を表す。

いま，標本を用いて $G(s)$ の期待値を表すと，

$$E[G(s)] = \frac{1}{n}\sum_{i=1}^n G(s_i) \tag{24}$$

$$s_i = \langle \boldsymbol{b}, \tilde{\boldsymbol{x}}_i \rangle, \quad ||\boldsymbol{b}|| = 1 \tag{25}$$

となる。これをラグランジュの未定乗数法で最大化することを考える。

$$L[\boldsymbol{b}, \lambda] = \frac{1}{n}\sum_{i=1}^n G(s_i) + \lambda(||\boldsymbol{b}|| - 1) \tag{26}$$

$\partial s_i / \partial \boldsymbol{b} = \tilde{\boldsymbol{x}}_i$ に注意すると，ラグランジアンの $\boldsymbol{b}$ による偏微分は，

$$\frac{\partial L}{\partial \boldsymbol{b}} = \frac{1}{n}\sum_{i=1}^n \tilde{\boldsymbol{x}}_i G'(s_i) + 2\lambda \boldsymbol{b} \tag{27}$$

近似ニュートン法により，$\partial L / \partial \boldsymbol{b} = \boldsymbol{0}$ となるような $\boldsymbol{b}$ を探す。

$$f(\boldsymbol{b}) = \frac{\partial L}{\partial \boldsymbol{b}} = \frac{1}{n}\sum_{i=1}^n \tilde{\boldsymbol{x}}_i G'(s_i) + 2\lambda \boldsymbol{b} \tag{28}$$

とおくと，$\boldsymbol{b}$ の更新式は次で与えられる。

$$\boldsymbol{b} \leftarrow \boldsymbol{b} - \left(\frac{\partial f}{\partial \boldsymbol{b}}\right)^{-1} f(\boldsymbol{b}) \tag{29}$$

ここで，

$$\frac{\partial f}{\partial \boldsymbol{b}} = \frac{\partial^2 L}{\partial \boldsymbol{b} \partial \boldsymbol{b}^{\mathrm{T}}} = \frac{1}{n}\sum_{i=1}^n \tilde{\boldsymbol{x}}_i \tilde{\boldsymbol{x}}_i^{\mathrm{T}} G''(s_i) + 2\lambda \boldsymbol{I}_d \tag{30}$$

であり，

$$\frac{1}{n}\sum_{i=1}^n \tilde{\boldsymbol{x}}_i \tilde{\boldsymbol{x}}_i^{\mathrm{T}} G''(s_i) \sim \frac{1}{n}\sum_{i=1}^n G''(s_i) \boldsymbol{I}_d \tag{31}$$

と近似すると，結局

$$\frac{\partial f}{\partial \boldsymbol{b}} \sim \frac{1}{n}\sum_{i=1}^{n} G''(s_i)\boldsymbol{I}_d + 2\lambda \boldsymbol{I}_d \tag{32}$$

$$= \left(\frac{1}{n}\sum_{i=1}^{n} G''(s_i) + 2\lambda\right)\boldsymbol{I}_d \tag{33}$$

$$\tag{34}$$

となり，これより，

$$\left(\frac{\partial f}{\partial \boldsymbol{b}}\right)^{-1} = \frac{1}{\frac{1}{n}\sum_{i=1}^{n} G''(s_i) + 2\lambda}\boldsymbol{I}_d \tag{35}$$

を得る。以上より，$\boldsymbol{b}$ の更新式は次のようになる。

$$\boldsymbol{b} \leftarrow \boldsymbol{b} - \left(\frac{\partial f}{\partial \boldsymbol{b}}\right)^{-1} f(\boldsymbol{b}) \tag{36}$$

$$= \boldsymbol{b} - \frac{1}{\frac{1}{n}\sum_{i=1}^{n} G''(s_i) + 2\lambda}\boldsymbol{I}_d \left(\frac{1}{n}\sum_{i=1}^{n} \tilde{\boldsymbol{x}}_i G'(s_i) + 2\lambda \boldsymbol{b}\right) \tag{37}$$

$$= \frac{1}{\frac{1}{n}\sum_{i=1}^{n} G''(s_i) + 2\lambda}\left\{\left(\frac{1}{n}\sum_{i=1}^{n} G''(s_i)\right)\boldsymbol{b} - \frac{1}{n}\sum_{i=1}^{n} \tilde{\boldsymbol{x}}_i G'(s_i)\right\} \tag{38}$$

$\boldsymbol{b}$ は更新の後正規化されるため，結局，

$$\boldsymbol{b} \leftarrow \left(\frac{1}{n}\sum_{i=1}^{n} G''(s_i)\right)\boldsymbol{b} - \frac{1}{n}\sum_{i=1}^{n} \tilde{\boldsymbol{x}}_i G'(s_i) \tag{39}$$

とすればよい。

# 問題設定

非ガウス性尺度を，次の 3 つの関数とした場合について考える。

$$G(s) = s^4 \tag{40}$$

$$G(s) = \log(\cosh(s)) \tag{41}$$

$$G(s) = -\exp\left(-\frac{s^2}{2}\right) \tag{42}$$

サンプルは，$x$ 軸を一様分布，$y$ 軸を標準正規分布からサンプリングしたのち，適当な変換をかけて白色化したものとする。また，これは 1 点だけ外れ値を含む。サンプル数は $n = 1{,}000,\ 1{,}500,\ 10{,}000$ の 3 段階について考えた。

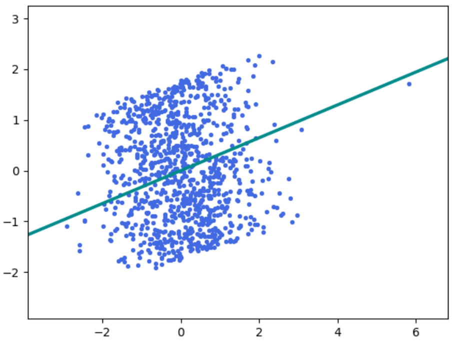
また，収束の条件は，現在の $\boldsymbol{b}$ 直近 5 イタレーションにおける $\boldsymbol{b}$ のそれぞれについて Euclid 距離を計算し，そのうちの最大値が $1 \times 10^{-4}$ 以下になったときとした。

なお，プログラムは 12 ページの Listing 1 に示した。

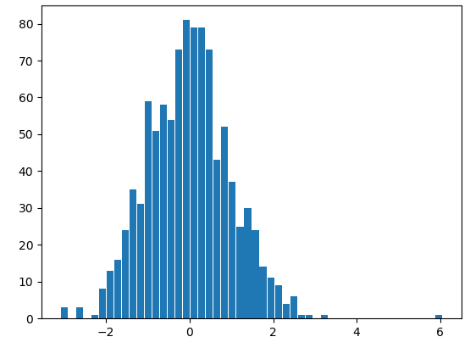
## 結果

各設定について，$\boldsymbol{b}$ の描かれた散布図，及びそこにサンプルを射影したときのヒストグラムを示す。図 2–6 に $G(s) = s^4$，図 8–12 に $G(s) = s^4$，図 14–18 に $G(s) = s^4$ の結果を示した。

結果を見ると，サンプル数 1,000 ではどのモデルも外れ値に引っ張られてしまい，いい射影を得られていないことがわかる。また，サンプル数を 10,000 まで増やせば，どのモデルでも外れ値の影響はなくなっている。違いが出ているのはサンプル数 1,500 のときで，このとき，$G(s) = s^4$ のモデルのみ外れ値の影響を大きく受けてしまっている。このことから，非ガウス性を評価する際に外れ値の存在が予想される場合は，尖度以外の指標を用いた方がよりよい射影を得られる可能性が高そうであるということが分かった。

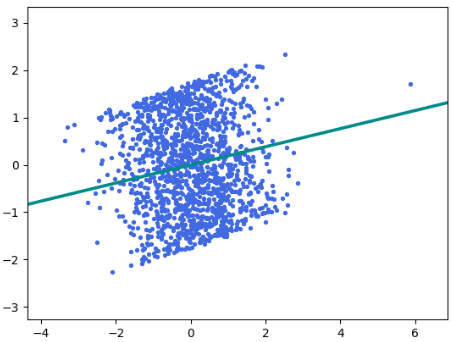



(a) 散布図

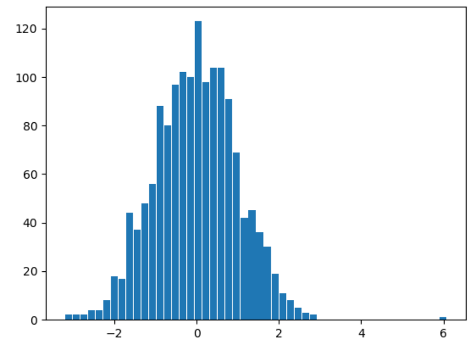



(a) ヒストグラム

図 2: 非ガウス性尺度 $G(s) = s^4$，標本数 1,000 のときの散布図と射影方向，及びヒストグラム

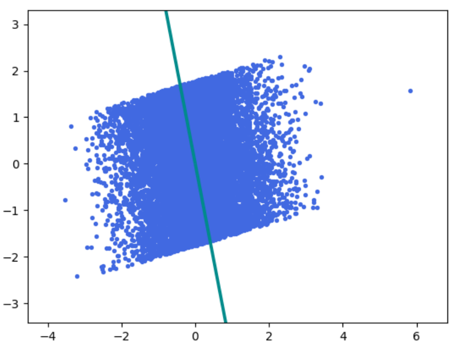



(a) 散布図

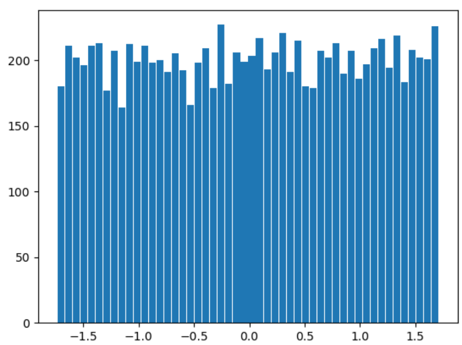



(a) ヒストグラム

図 4: 非ガウス性尺度 $G(s) = s^4$，標本数 1,500 のときの散布図と射影方向，及びヒストグラム

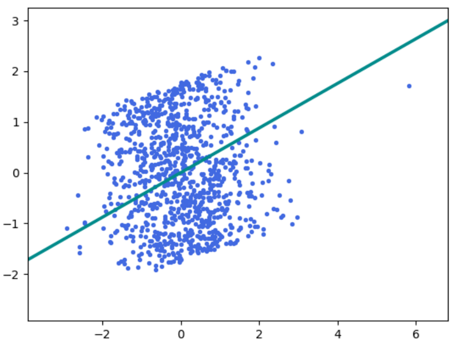



(a) 散布図

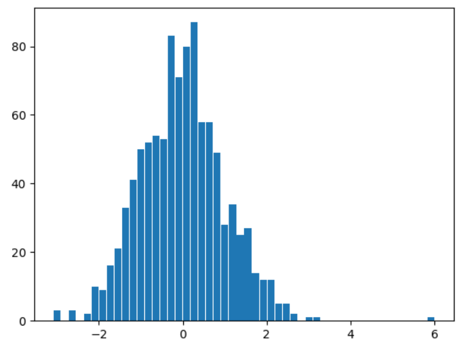



(a) ヒストグラム

図 6: 非ガウス性尺度 $G(s) = s^4$，標本数 10,000 のときの散布図と射影方向，及びヒストグラム

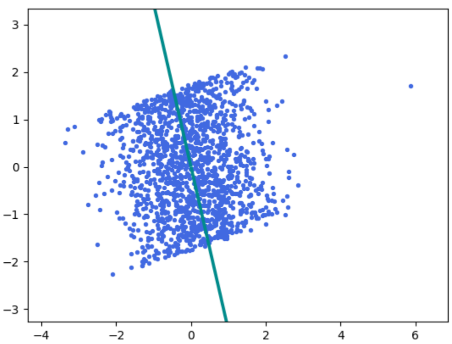


(a) 散布図

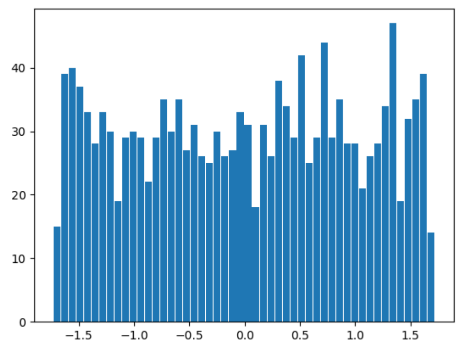


(a) ヒストグラム

図 8: 非ガウス性尺度 $G(s) = \log\cosh(s)$，標本数 1,000 のときの散布図と射影方向，及びヒストグラム

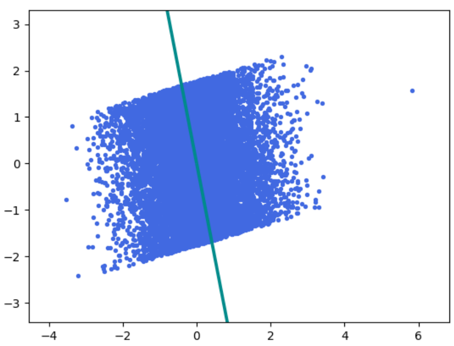


(a) 散布図

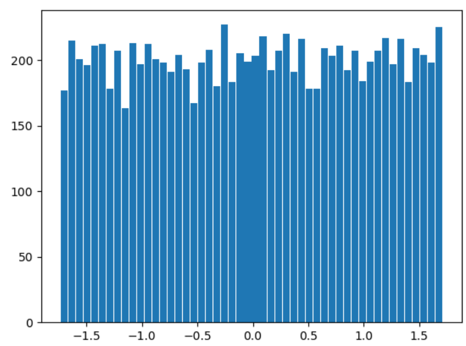


(a) ヒストグラム

図 10: 非ガウス性尺度 $G(s) = \log\cosh(s)$，標本数 1,500 のときの散布図と射影方向，及びヒストグラム

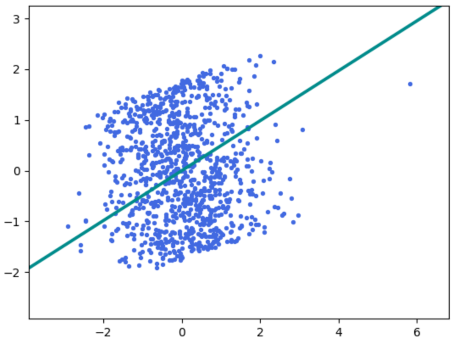



(a) 散布図

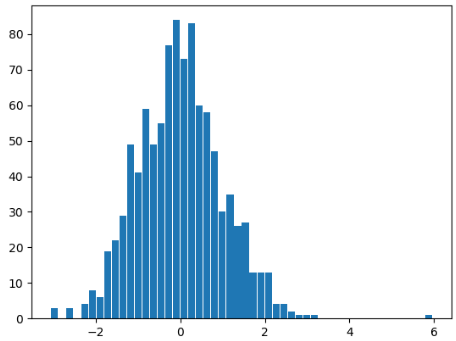



(a) ヒストグラム

図 12: 非ガウス性尺度 $G(s) = \log\cosh(s)$，標本数 10,000 のときの散布図と射影方向，及びヒストグラム

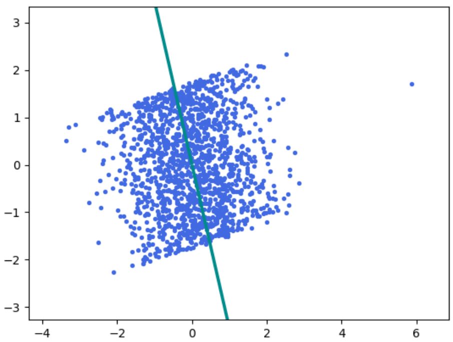



(a) 散布図

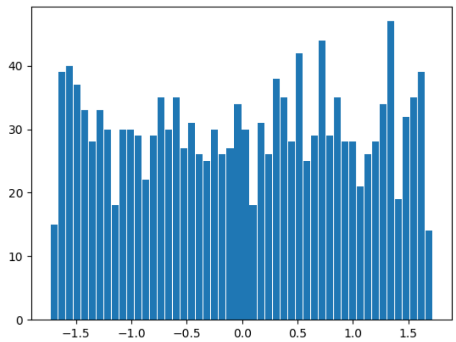



(a) ヒストグラム

図 14: 非ガウス性尺度 $G(s) = -\exp\left(-s^2/2\right)$，標本数 1,000 のときの散布図と射影方向，及びヒストグラム

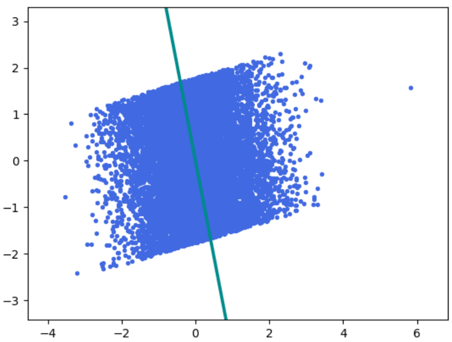



(a) 散布図

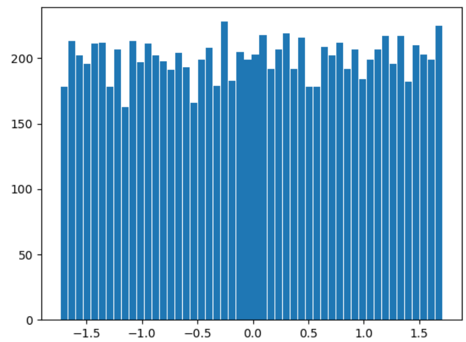



(a) ヒストグラム

図 16: 非ガウス性尺度 $G(s) = -\exp\left(-s^2/2\right)$，標本数 1,500 のときの散布図と射影方向，及びヒストグラム




(a) 散布図




(a) ヒストグラム

図 18: 非ガウス性尺度 $G(s) = -\exp\left(-s^2/2\right)$，標本数 10,000 のときの散布図と射影方向，及びヒストグラム

# プログラム

実行環境と用いた言語・ライブラリを以下の表 1 に示す。

表 1: プログラムの実行環境

| | | |
|---|---|---|
| OS | : | Microsoft Windows 10 Pro (64bit) |
| CPU | : | Intel(R) Core(TM) i5-4300U |
| RAM | : | 4.00 GB |
| 使用言語 | : | Python3.6 |
| 可視化 | : | matplotlib ライブラリ |

Listings 1: `assignment2.py`

```python
# -*- coding: utf-8 -*-


import numpy as np
import matplotlib.pyplot as plt
from scipy.linalg import sqrtm


def generate_data(n=1000):
    x = np.concatenate([
        np.random.rand(n, 1),
        np.random.randn(n, 1)
        ], axis=1)
    x[0, 1] = 6   # outlier

    # Standardization
    x = (x - np.mean(x, axis=0)) / np.std(x, axis=0)

    M = np.array([[1, 3], [5, 3]])
    x = x.dot(M.T)
    x = np.linalg.inv(sqrtm(np.cov(x, rowvar=False))).dot(x.T).T
    return x


def metric_s4(s, derivative=0):
    if derivative == 0:
        met = s**4
    elif derivative == 1:
        met = 4*s**3
    elif derivative == 2:
        met = 12*s**2
    else:
```

```
        raise ValueError('Derivatives more than second are not defined. But the input was: {}'.format(derivative))
    return met


def metric_logcosh(s, derivative=0):
    if derivative == 0:
        met = np.log(np.cosh(s))
    elif derivative == 1:
        met = np.tanh(s)
    elif derivative == 2:
        met = 1 - np.tanh(s)**2
    else:
        raise ValueError('Derivatives more than second are not defined. But the input was: {}'.format(derivative))
    return met


def metric_exp(s, derivative=0):
    if derivative == 0:
        met = - np.exp(-(s**2)/2)
    elif derivative == 1:
        met = s*np.exp(-(s**2)/2)
    elif derivative == 2:
        met = (1 - s**2) * np.exp(-(s**2)/2)
    else:
        raise ValueError('Derivatives more than second are not defined. But the input was: {}'.format(derivative))
    return met


def normalize(b):
    b = b / np.linalg.norm(b)
    if b[0] < 0:
        b *= -1
    return b


def update(b, x_whitened, metric):
    n = len(x_whitened)
    s = x_whitened.dot(b)
    b_new = (
        (np.mean(metric(s, 2))) * b
        - (1/n) * np.sum(x_whitened * metric(s, 1)[:, np.newaxis], axis=0)
        )
    return b_new

```

```

def train(x_whitened, metric, max_iter=100, eps=1e-4, n=5):
    d = x_whitened.shape[1]

    # initialize b
    b = np.random.randn(d)
    b = normalize(b)

    b_olds = []
    for i in range(max_iter):
        b_old = b.copy()
        b = update(b, x_whitened, metric)
        b = normalize(b)

        # if converge, break
        if len(b_olds) < n:
            b_olds.append(b_old)
        else:
            b_olds[:-1] = b_olds[1:]
            b_olds[-1] = b_old
            b_olds = np.array(b_olds)
            diffs = np.sqrt(np.sum((b_olds - b)**2, axis=1))
            if diffs.max() < eps:
                break
    n_iter = i + 1
    return b, n_iter


def main():
    # settnigs
    n_sample = 1000
    metric, metric_name = metric_s4, 's4'
    #metric, metric_name = metric_logcosh, 'logcosh'
    #metric, metric_name = metric_exp, 'exp'

    offset = 1.0
    np.random.seed(0)
    scatter_path =
    f'../figures/assignment2_result_{metric_name}_n{n_sample}_scatter.png'
    hist_path =
    f'../figures/assignment2_result_{metric_name}_n{n_sample}_hist.png'


    # load data
    x = generate_data(n_sample)
    #x_whitened = (x - np.mean(x, axis=0)) / np.std(x, axis=0)

```

```

    # train
    b, n_iter = train(x, metric, max_iter=1000, eps=1e-4, n=5)


    # result
    print(f'Metric: {metric_name}')
    print(f'#Data: {n_sample} \t #Iter: {n_iter}')
    print('b: {} (norm = {})'.format(b, np.linalg.norm(b)))


    # plot scatter
    scale = 1e3
    x0_max, x0_min = x[:, 0].max(), x[:, 0].min()
    x1_max, x1_min = x[:, 1].max(), x[:, 1].min()


    plt.scatter(x[:, 0], x[:, 1], color='royalblue', s=8)
    plt.quiver(
        0, 0, b[0]*scale, b[1]*scale,
        color='darkcyan', angles='xy', scale_units='xy', scale=6.5,
        )
    plt.quiver(
        0, 0, -b[0]*scale, -b[1]*scale,
        color='darkcyan', angles='xy', scale_units='xy', scale=6.5,
        )
    plt.xlim([x0_min-offset, x0_max+offset])
    plt.ylim([x1_min-offset, x1_max+offset])
    plt.savefig(scatter_path)
    plt.show()


    # plot hist
    x_casted = x.dot(b)
    plt.hist(x_casted, bins=50, rwidth=0.9)
    plt.savefig(hist_path)
    plt.show()


if __name__ == '__main__':
    main()
```