知能システム論第 3 回課題

37186305
航空宇宙工学専攻修士一年
荒居秀尚

2018 年 10 月 25 日

1　宿題 1

$$\overline{X} - \overline{Y} \sim N\left(\mu_X - \mu_Y, \frac{\sigma^2}{n_X} + \frac{\sigma^2}{n_Y}\right) \tag{1.1}$$

を示す。

$$\begin{aligned} E[\overline{X} - \overline{Y}] &= E\left[\frac{1}{n_X}\sum_{i=1}^{n_X} X_i - \frac{1}{n_Y}\sum_{i=1}^{n_Y} Y_i\right] && (1.2) \\ &= \frac{1}{n_X}\sum_{i=1}^{n_X} E[X_i] - \frac{1}{n_Y}\sum_{i=1}^{n_Y} E[Y_i] && (1.3) \\ &= \mu_X - \mu_Y && (1.4) \end{aligned}$$

$$\begin{aligned} V\left(\overline{X} - \overline{Y}\right) &= E\left[\left(\overline{X} - \overline{Y}\right)^2\right] - \left\{E\left[\overline{X} - \overline{Y}\right]\right\}^2 && (1.5) \\ &= E\left[\overline{X}^2\right] + E\left[\overline{Y}^2\right] - 2E\left[\overline{XY}\right] - \left\{E\left[\overline{X}\right]\right\}^2 - \left\{E\left[\overline{Y}\right]\right\}^2 + 2E\left[\overline{X}\right]E\left[\overline{Y}\right] && (1.6) \\ &= E\left[\overline{X}^2\right] - \left\{E\left[\overline{X}\right]\right\}^2 + E\left[\overline{Y}^2\right] - \left\{E\left[\overline{Y}\right]\right\}^2 && (1.7) \\ &= V\left(\overline{X}\right) + V\left(\overline{Y}\right) && (1.8) \\ &= \frac{\sigma^2}{n_X} + \frac{\sigma^2}{n_Y} && (1.9) \end{aligned}$$

正規分布の再生性から $\overline{X} - \overline{Y}$ も正規分布に従うため、題意を得る。

2　宿題 2

2.1　A

6 が出る確率の上界は

$$P(|X - 2.2| > 2.8) < \frac{0.7}{2.8^2} = 0.08928 \tag{2.1}$$

である。

2.2 B

1,2,3 のいずれかが出る確率の下界は、4,5,6 がでる確率の上界を 1 から引けば良い。

$$1 - P(|X - 2.2| > 1.2) > 1 - \frac{0.7}{1.2^2} = 0.5139 \tag{2.2}$$

である。

3 宿題 3

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


def calc_chi_square(size):
    X1 = np.random.normal(0.0, 1.0, (size, ))
    X2 = np.random.normal(0.0, 1.0, (size, ))
    return X1**2 + X2**2


def calc_avg(y):
    avgs = []
    for i in range(y.size):
        if i == 0:
            avgs.append(y[i])
        else:
            avgs.append((avgs[i - 1] + y[i]) / (i + 1))
    return np.array(avgs)


if __name__ == "__main__":
    y = calc_chi_square(10000)
    avgs = calc_avg(y)
    x = np.arange(1, 10001)

    plt.plot(x, avgs)
    plt.xlabel("$n$")
    plt.ylabel("$\overline{X}_{n}$")
    plt.grid(True)
    plt.savefig("chi_square_strong_row.eps")
```

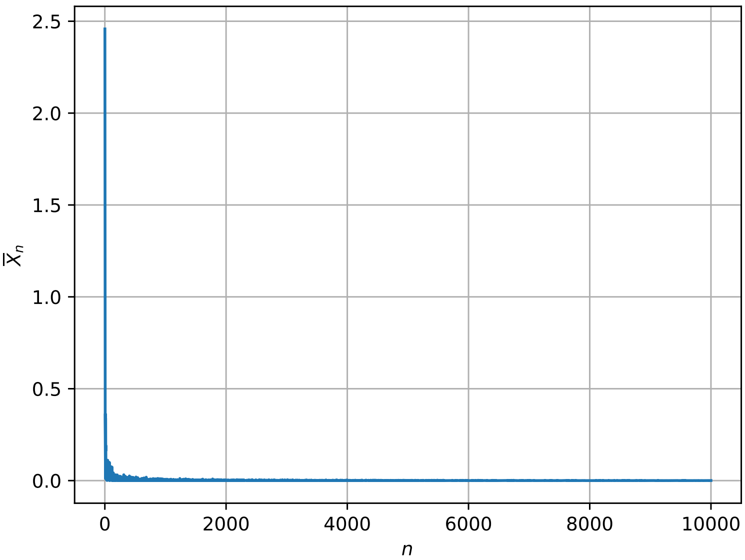

図 3.1: カイ二乗分布と大数の強法則

```python
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns


def chi_square(size):
    X1 = np.random.normal(0.0, 1.0, (size, ))
    X2 = np.random.normal(0.0, 1.0, (size, ))
    return X1**2 + X2**2


def chi_avg(chi):
    return chi.mean()


if __name__ == "__main__":
    color = ["r", "g", "b", "y", "k", "orange"]
    plt.figure()
    plt.ylabel("frequency")
    for i, c in enumerate(color):
        avg = [chi_avg(chi_square(i + 1)) for _ in range(10000)]
        sns.distplot(avg, color=c, label=f"n={i+1}")
    plt.legend()
    plt.savefig("chi_square_ultimate.eps")
```

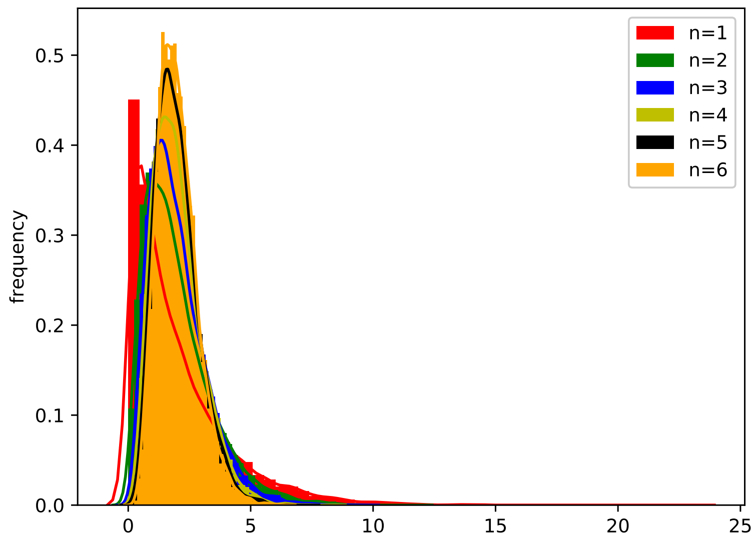

図 3.2: カイ二乗分布と中心極限定理

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


def chi_square(size):
    X1 = np.random.normal(0.0, 1.0, (size, ))
    X2 = np.random.normal(0.0, 1.0, (size, ))
    return X1**2 + X2**2


def t_dist(size):
    X = np.random.normal(0.0, 1.0, (size, ))
    Y = chi_square(size)
    return np.divide(X, np.sqrt(Y / 2 + 1e-7))


def avg(t):
    avgs = []
    for i in range(t.size):
        if i == 0:
            avgs.append(t[i])
        else:
            avgs.append((avgs[i - 1] + t[i]) / (i + 1))
    return np.array(avgs)


if __name__ == "__main__":
    t = t_dist(10000)
    avgs = avg(t)
    x = np.arange(1, 10001)
    plt.plot(x, avgs)
    plt.xlabel("$n$")
    plt.ylabel("$\overline{X}_{n}$")
    plt.grid(True)
    plt.savefig("t_dist_strong_law.eps")
```

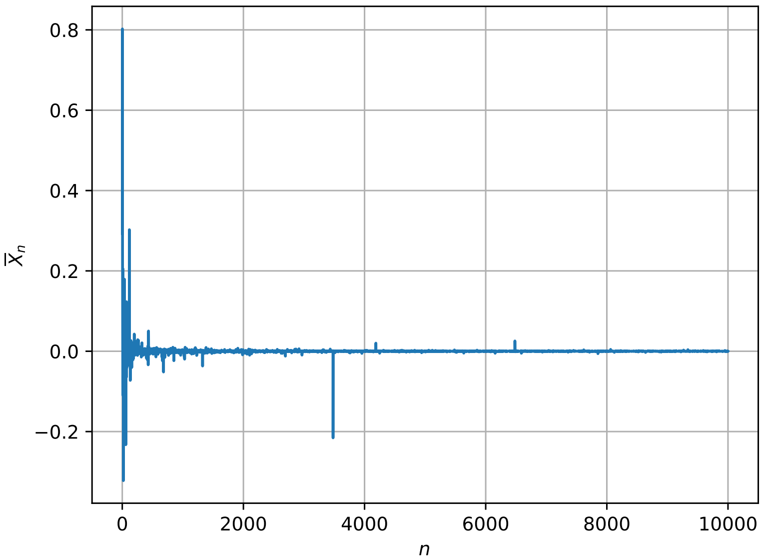

図 3.3: t 分布と大数の強法則

```python
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns


def chi_square(size):
    X1 = np.random.normal(0.0, 1.0, (size, ))
    X2 = np.random.normal(0.0, 1.0, (size, ))
    return X1**2 + X2**2


def t_dist(size):
    X = np.random.normal(0.0, 1.0, (size, ))
    Y = chi_square(size)
    return np.divide(X, np.sqrt(Y / 2 + 1e-7))


def avg(t):
    return t.mean()


if __name__ == "__main__":
    color = ["r", "g", "b", "y", "k", "orange"]
    plt.figure()
    plt.ylabel("frequency")
    for i, c in enumerate(color):
        avgs = np.array([avg(t_dist(i + 1)) for _ in range(10000)])
        mean = avgs.mean()
        normalized = ((avgs - mean) / (1.0 / np.sqrt(i + 1)))
        sns.distplot(normalized, color=c, label=f"n={i+1}")
    plt.legend()
    plt.xlim(-10, 10)
    plt.savefig("t_dist_ultimate.eps")
```

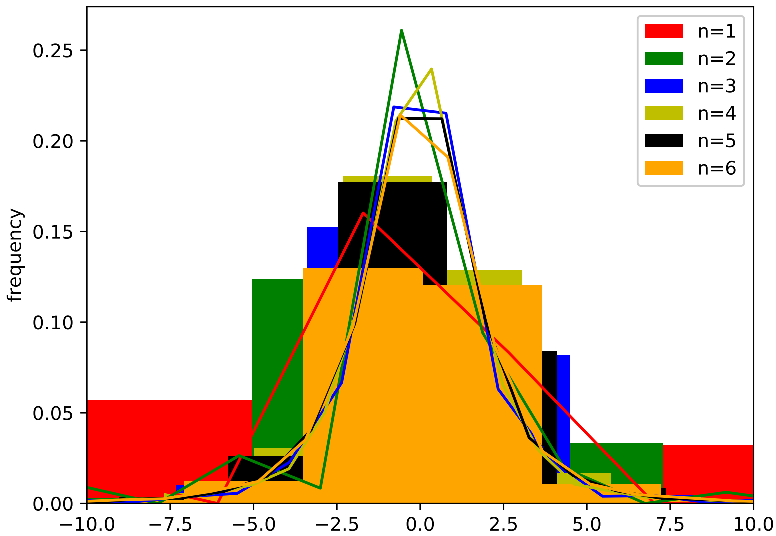

図 3.4: t 分布と中心極限定理