

# 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

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

を示す。

$$E[\bar{X} - \bar{Y}] = E\left[\frac{1}{n_X} \sum_{i=1}^{n_X} X_i - \frac{1}{n_Y} \sum_{i=1}^{n_Y} Y_i\right] \quad (1.2)$$

$$= \frac{1}{n_X} \sum_{i=1}^{n_X} E[X_i] - \frac{1}{n_Y} \sum_{i=1}^{n_Y} E[Y_i] \quad (1.3)$$

$$= \mu_X - \mu_Y \quad (1.4)$$

$$V(\bar{X} - \bar{Y}) = E\left[(\bar{X} - \bar{Y})^2\right] - \{E[\bar{X} - \bar{Y}]\}^2 \quad (1.5)$$

$$= E[\bar{X}^2] + E[\bar{Y}^2] - 2E[\bar{X}\bar{Y}] - \{E[\bar{X}]\}^2 - \{E[\bar{Y}]\}^2 + 2E[\bar{X}]E[\bar{Y}] \quad (1.6)$$

$$= E[\bar{X}^2] - \{E[\bar{X}]\}^2 + E[\bar{Y}^2] - \{E[\bar{Y}]\}^2 \quad (1.7)$$

$$= V(\bar{X}) + V(\bar{Y}) \quad (1.8)$$

$$= \frac{\sigma^2}{n_X} + \frac{\sigma^2}{n_Y} \quad (1.9)$$

正規分布の再生性から  $\bar{X} - \bar{Y}$  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

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

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。

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

である。

## 3 宿題 3

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3
4
5 def calc_chi_square(size):
6     X1 = np.random.normal(0.0, 1.0, (size, ))
7     X2 = np.random.normal(0.0, 1.0, (size, ))
8     return X1**2 + X2**2
9
10
11 def calc_avg(y):
12     avgs = []
13     for i in range(y.size):
14         if i == 0:
15             avgs.append(y[i])
16         else:
17             avgs.append((avgs[i - 1] + y[i]) / (i + 1))
18     return np.array(avgs)
19
20
21 if __name__ == "__main__":
22     y = calc_chi_square(10000)
23     avgs = calc_avg(y)
24     x = np.arange(1, 10001)
25
26     plt.plot(x, avgs)
27     plt.xlabel("$n$")
28     plt.ylabel("$\overline{X}_{n}$")
29     plt.grid(True)
30     plt.savefig("chi_square_strong_row.eps")
```

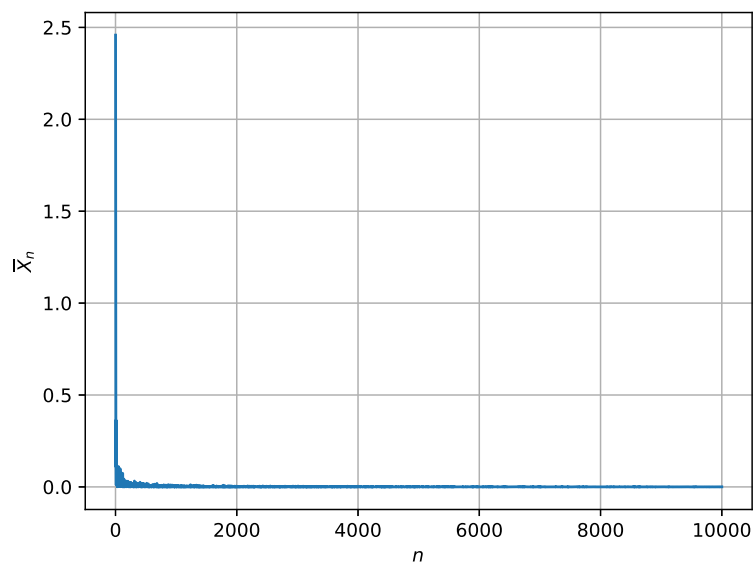


図 3.1: カイ二乗分布と大数の強法則

```

1 import numpy as np
2 import matplotlib.pyplot as plt
3 import seaborn as sns
4
5
6 def chi_square(size):
7     X1 = np.random.normal(0.0, 1.0, (size, ))
8     X2 = np.random.normal(0.0, 1.0, (size, ))
9     return X1**2 + X2**2
10
11
12 def chi_avg(chi):
13     return chi.mean()
14
15
16 if __name__ == "__main__":
17     color = ["r", "g", "b", "y", "k", "orange"]
18     plt.figure()
19     plt.ylabel("frequency")
20     for i, c in enumerate(color):
21         avg = [chi_avg(chi_square(i + 1)) for _ in range(10000)]
22         sns.distplot(avg, color=c, label=f"n={i+1}")
23     plt.legend()
24     plt.savefig("chi_square_ultimate.eps")

```

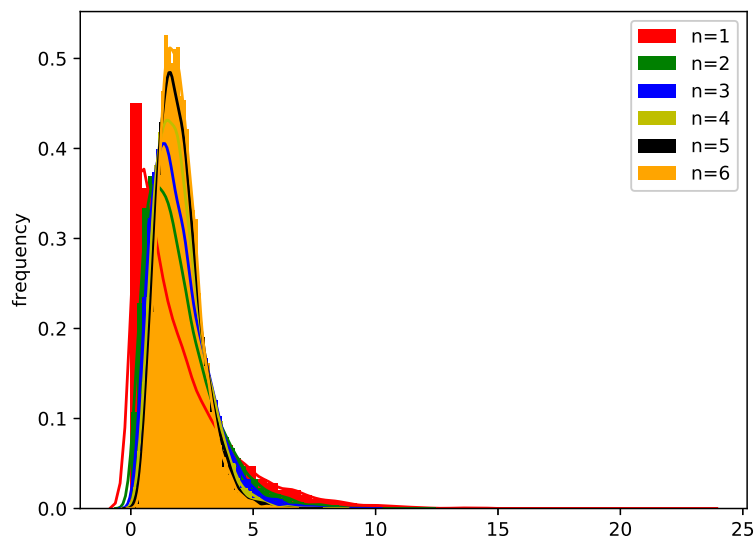


図 3.2: カイ二乗分布と中心極限定理

```

1 import numpy as np
2 import matplotlib.pyplot as plt
3
4
5 def chi_square(size):
6     X1 = np.random.normal(0.0, 1.0, (size, ))
7     X2 = np.random.normal(0.0, 1.0, (size, ))
8     return X1**2 + X2**2
9
10
11 def t_dist(size):
12     X = np.random.normal(0.0, 1.0, (size, ))
13     Y = chi_square(size)
14     return np.divide(X, np.sqrt(Y / 2 + 1e-7))
15
16
17 def avg(t):
18     avgs = []
19     for i in range(t.size):
20         if i == 0:
21             avgs.append(t[i])
22         else:
23             avgs.append((avgs[i - 1] + t[i]) / (i + 1))
24     return np.array(avgs)
25
26
27 if __name__ == "__main__":
28     t = t_dist(10000)
29     avgs = avg(t)
30     x = np.arange(1, 10001)
31     plt.plot(x, avgs)
32     plt.xlabel("$n$")
33     plt.ylabel("$\overline{X}_{n}$")
34     plt.grid(True)
35     plt.savefig("t_dist_strong_law.eps")

```

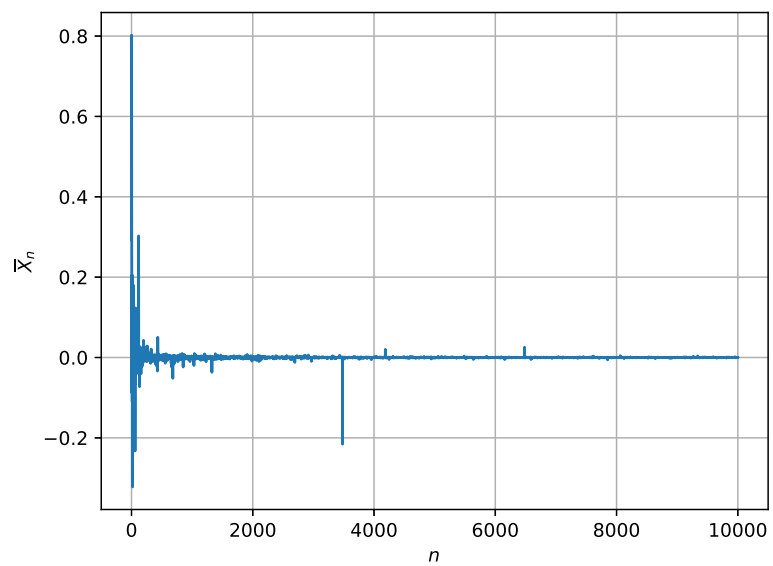


図 3.3: t 分布と大数の強法則

```

1 import numpy as np
2 import matplotlib.pyplot as plt
3 import seaborn as sns
4
5
6 def chi_square(size):
7     X1 = np.random.normal(0.0, 1.0, (size, ))
8     X2 = np.random.normal(0.0, 1.0, (size, ))
9     return X1**2 + X2**2
10
11
12 def t_dist(size):
13     X = np.random.normal(0.0, 1.0, (size, ))
14     Y = chi_square(size)
15     return np.divide(X, np.sqrt(Y / 2 + 1e-7))
16
17
18 def avg(t):
19     return t.mean()
20
21
22 if __name__ == "__main__":
23     color = ["r", "g", "b", "y", "k", "orange"]
24     plt.figure()
25     plt.ylabel("frequency")
26     for i, c in enumerate(color):
27         avgs = np.array([avg(t_dist(i + 1)) for _ in range(10000)])
28         mean = avgs.mean()
29         normalized = ((avgs - mean) / (1.0 / np.sqrt(i + 1)))
30         sns.distplot(normalized, color=c, label=f"n={i+1}")
31     plt.legend()
32     plt.xlim(-10, 10)
33     plt.savefig("t_dist_ultimate.eps")

```

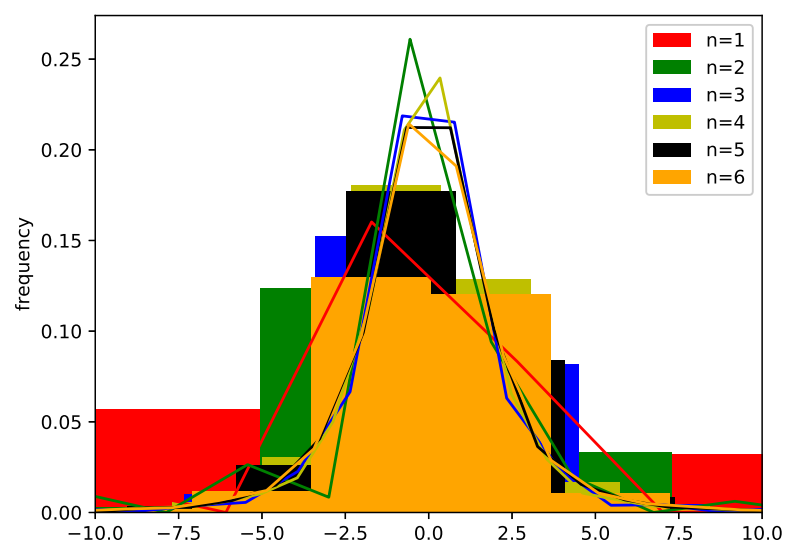


図 3.4: t 分布と中心極限定理