第 8 章

SciPy の応用

NumPy が基本的な数値計算と配列操作を担うのに対して，SciPy(Scientific Python) はそれをベースに統計処理・数値積分・線形代数・画像処理などの高度な科学技術計算機能を提供するライブラリです。本章では，SciPy の統計モジュール `scipy.stats` を用いた基本統計量の計算と正規分布のグラフ化，および `scipy.integrate` を用いた数値積分と中心極限定理の応用について学びます。事前に次のコマンドで SciPy と Matplotlib のインストールを確認しておいて下さい。

SciPy, Matplotlib のインストール確認

```
pip install scipy ← SciPy のインストール確認
pip install matplotlib ← Matplotlib のインストール確認
```

8.1 NumPy と SciPy

前述した通り，NumPy と SciPy は，Python で科学技術計算を行うための中心的なライブラリです。それぞれの主な機能を以下にまとめます。

NumPy (`https://numpy.org/`)　基本的な計算機能と高速化を提供する。

- 初等関数（三角関数，指数・対数関数など）
- NDarray と基本線形計算（ベクトル・行列演算）
- 乱数生成，基本統計量（平均・分散・標準偏差など）

SciPy (`https://scipy.org/`)　NumPy をベースに高度な計算機能を提供する。

- 微分 (differentiate) と積分 (integrate)
- 行列の固有値・特異値計算 (`linalg`)
- 画像処理 (`ndimage`)
- 特殊関数 (`special`)
- 統計処理 (`stats`)

8.2 基本統計量の計算

統計処理を行う前に，基本的な用語を整理しておきます。

母集団　すべてのデータの集合（有限または無限）。

確率変数　データが取りうる値を変数として表現したもの。

離散と連続　データがバラバラな値を取る場合を離散，連続的な値を取る場合を連続という。

確率密度関数　確率変数 X が取りうる確率を表現する関数。

確率分布　確率密度関数のグラフ（形状）のこと。

標本　母集団から取り出した離散データ。

標本平均・標本分散・標本標準偏差　標本の平均・分散・標準偏差。

乱数のところで触れたように，離散データ（標本）の平均，分散，標準偏差などの基本統計量は NumPy で計算できます。主な関数を表 8.1 にまとめます。

NumPy 関数	**意味**
`np.average(x)`	平均 (average)
`np.var(x)`	分散 (variance)
`np.std(x)`	標準偏差 (standard deviation)
`np.max(x)`	最大値
`np.min(x)`	最小値
`np.median(x)`	中央値 (median)

表 8.1　NumPy の基本統計量関数

下記の `sample_stat.py` では，2 つの離散データ `x1`，`x2` に対して基本統計量をリストとして計算し，表示しています。

ソースコード 8.1　`sample_stat.py`：平均・分散・標準偏差の計算

```
1  # sample_stat.py: 平均,分散,標準偏差の計算
2  import numpy as np # NumPy
3
4  # 離散データの入力
5  x1 = [3.0, 4.5, 2.0, 3.0, 3.4]
6  x2 = [-3.4, 4.0, 10.0, 4.0, -2]
7
8  x1_stat = [
9      np.average(x1), # 平均値
10     np.var(x1), # 分散
11     np.std(x1), # 標準偏差
12     np.max(x1), # 最大値
13     np.min(x1), # 最小値
14     np.median(x1) # 中央値
15 ]
16 # (1) np.float64 がくっつく
17 # print('x1_stat = ', x1_stat)
18 # (2) np.float64 を消したい場合
19 print('x1_stat␣=␣', end='')
20 for val in x1_stat:
21     print(f'{val:g}', end='␣')
22 print() # 最後は改行
```

また，分散が標準偏差の 2 乗であること，および中央値の確認は次のように行えます。

ソースコード 8.2　sample_stat.py（続き）：分散と中央値の確認

```
40 # 分散 == 標準偏差 ^2
41 print('x1:␣var␣==␣std**2:␣', x1_stat[1], x1_stat[2]**2)
42 print('x2:␣var␣==␣std**2:␣', x2_stat[1], x2_stat[2]**2)
43
44 # 中央値 = sort()
45 x1.sort() # 昇順で並べ替え
46 mid_i = int(len(x1) / 2) # len(x1)=5 -> 5/2=2 -> mid_i = 2
47 if len(x1) % 2 != 0:
48     print('x1.sort:␣', x1, '␣->␣', x1_stat[5], x1[mid_i])
49 else:
50     median = (sorted(x1)[mid_i - 1] + sorted(x1)[mid_i]) / 2
51     print('x1.sort:␣', x1, '␣->␣', x1_stat[5], median)
```

8.3　正規分布とそのグラフ化

正規分布 (normal distribution) $\mathrm{N}(\mu, \sigma^2)$ は，自然界や社会現象に最もよく現れる確率分布であり，平均 μ と分散 σ^2（標準偏差 σ）で決まる釣り鐘型の形状を持ちます。その確率密度関数は次式で与えられます。

$$f(x) = \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{8.1}$$

正規分布の確率密度関数は，NumPy で直接計算する方法と，SciPy の scipy.stats.norm クラスの pdf 関数を使う方法の 2 通りがあります。plot_normal_dist.py では両方の結果を重ね描きして一致を確認します。

ソースコード 8.3　plot_normal_dist.py：正規分布のグラフを描画する

```
1  # plot_normal_dist.py: 正規分布のグラフを描画する
2  import matplotlib.pyplot as plt # Matplotlib
3  import numpy as np # NumPy
4  import scipy.stats as scstats # SciPy.stats
5
6  # 平均mu と標準偏差 sigma を入力
7  str_mu = input('mu␣=␣?␣>␣')
8  str_sigma = input('sigma␣=␣?␣>␣')
9  mu = float(str_mu)
10 sigma = float(str_sigma)
11
12 # x の範囲[a, b]を指定
13 a = mu - sigma * 5
14 b = mu + sigma * 5
15
16 # x 区間の分割数を指定
17 n = 50 # 画面解像度と関数の変化率による
18
19 # x の配列を生成
20 x_h = (b - a) / n # 小区間幅
21 x = [ a + i * x_h for i in range(n + 1)]
22
23 # x ごとに y の値を計算: NumPy 関数を使用
24 y = [(1.0 / (np.sqrt(np.pi * 2) * sigma))
```

```
    * np.exp(-(x[i] - mu)**2 / (2 * sigma**2))
    for i in range(n + 1)]

# x ごとに y の値を計算：SciPy.stats.norm クラスを使用
norm_dist = scstats.norm(loc=mu, scale=sigma)
y_stat = norm_dist.pdf(x)

# 2次元グラフを描画
fig, ax = plt.subplots()
ax.plot(x, y, label='NumPy') # 直線
ax.plot(x, y_stat, 'o', label='SciPy.stat',
        color='red') # ○でプロット
ax.set_title('N(' + str(mu) + ',␣' + str(sigma**2) + ')')
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.grid()
ax.legend()
plt.show()
```

$\mu = 2.0$, $\sigma = 3.0$（すなわち $\sigma^2 = 9.0$）として実行すると，NumPy による折れ線と SciPy による点が完全に重なるグラフ N(2.0, 9.0) が得られ，両計算結果の一致が確認できます（図 8.1）。

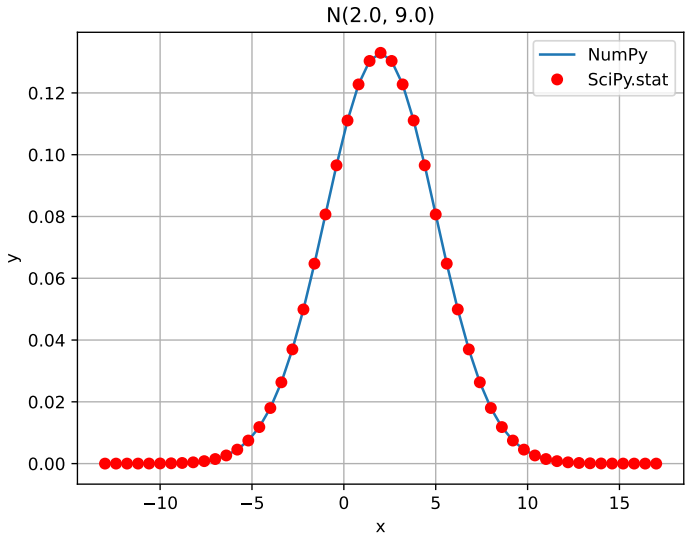

図 8.1　plot_normal_dist.py の実行結果：N(2.0, 9.0)

8.4　定積分の計算：scipy.integrate

8.4.1　基本的な定積分

SciPy の scipy.integrate モジュールには，数値積分のための関数が用意されています。最も基本的な quad 関数を使えば，任意の関数 $f(x)$ の定積分 $\int_a^b f(x)\,dx$ を数値的に計算できます。quad は積分値とともに

推定誤差も返します。

ソースコード 8.4　test_quad.py：数値積分の例

```
# test_quad.py: 数値積分の例
import numpy as np # NumPy
from scipy.integrate import quad # SciPy.integrate

# 積分する関数を定義(例: f(x) = x^2)
def f(x):
    return x**2

# 区間 [0, 1]で定積分を計算
result, error = quad(f, 0, 1)

print("積分結果:", result) # 結果を表示
print("推定誤差:", error) # 推定誤差を表示
```

8.4.2　広義積分（特異点を含む積分）

quad は広義積分（積分区間内に特異点を持つ関数の定積分）も扱えます。ただし，特異点がある場合は収束判定の警告が表示されることがあります。

ソースコード 8.5　test_quad.py（続き）：広義積分の例

```
# 積分する関数を定義(例: f(x) = sin(1/x))
def g(x):
    return np.sin(1 / x)

result, error = quad(g, 0, 1)

print("積分結果:", result) # 結果を表示
print("推定誤差:", error) # 推定誤差を表示
```

$g(x) = \sin(1/x)$ は $x = 0$ に特異点を持つため，実行すると次のような収束警告が表示されますが，積分値自体は得られます。

test_quad.py の実行結果（広義積分）

```
IntegrationWarning: The maximum number of subdivisions (50)
has been achieved.
...
積分結果: 0.5024347471020827
推定誤差: 0.0006174569031721644
```

8.5　中心極限定理

中心極限定理 (Central Limit Theorem) は，区間推定・仮説検定の基礎をなす重要な定理です。

中心極限定理

母集団から n 個の標本 $X_1, X_2, \ldots, X_n$ を無作為集出する。この時，標本平均

$$\overline{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$$

は，母集団の分散が何であれ，近似的に正規分布

$$\mathrm{N}\left(\mu,\ \frac{\sigma^2}{n}\right)$$

に従う。

母集団から n 個のデータ $x_1, x_2, \ldots, x_n$ を無作為抽出する試行を無限回繰り返すと，無限個の取り出し方は n 個の確率変数 $X_1, X_2, \ldots, X_n$ にまとめられます。この n 個の確率変数の平均 $\overline{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$ は，母分散の平均 μ を持ち，分散が σ^2/n である正規分布に「近い」分布を示します。$n \to \infty$ とすれば，$\overline{X}$ は母平均 μ の付近に集中します。このことを中心極限定理は表現しているのです。

次の問題で中心極限定理を適用します。

問題

ある年における日本全国の 20 歳の男子の平均体重が 60kg，標準偏差が 5.4kg であった。このとき，無作為抽出した 49 人の 20 歳男子の平均体重が 62kg 以上になる確率を求めよ。

中心極限定理より，標本平均 $\overline{X}$ は近似的に

$$\mathrm{N}\left(60,\ \frac{5.4^2}{49}\right) = \mathrm{N}\left(60,\ \left(\frac{5.4}{7}\right)^2\right)$$

に従うので，求める確率は

$$\begin{aligned} P(\overline{X} \geq 62) &= 1 - P(\overline{X} < 62) \\ &= 1 - F(62) \\ &= 1 - \Phi\left(\frac{62-60}{5.4/7}\right) \\ &\approx 0.0049 \end{aligned}$$

となります。ここで Φ は標準正規分布の累積分布関数です。Python では，`scipy.stats.norm` クラスの**累積分布関数** `cdf` または**生存関数** `sf`（$= 1 - \mathrm{cdf}(x)$）を用いて計算できます。

ソースコード 8.6　`example_cent_limit.py`：中心極限定理の事例

```
1 # example_cent_limit.py: 中心極限定理の事例
2 # 標本平均がx_bar以上になる確率を導出する
3 import numpy as np # NumPy
4 import scipy.stats as scstats # SciPy.stats
5
6 # 母集団の平均mu と標準偏差 sigma を入力
7 str_mu = input('母集団の平均␣=␣?␣>␣')
8 str_sigma = input('母集団の標準偏差␣=␣?␣>␣')
9 mu = float(str_mu)
```

```
sigma = float(str_sigma)

# 標本平均
str_num_x = input('標本数␣=␣?␣>␣')
num_x = float(str_num_x)
str_x_bar = input('標本の平均␣=␣?␣>␣')
x_bar = float(str_x_bar)

# x ごとに y の値を計算: SciPy.stats.norm クラスを使用
norm_dist = scstats.norm(loc=mu, scale=sigma/np.sqrt(num_x))
p_x_bar0 = 1.0 - norm_dist.cdf(x_bar) # 累積分布関数
p_x_bar1 = norm_dist.sf(x_bar) # 1 - cdf(x)

print('p_x_bar0␣=␣', p_x_bar0)
print('p_x_bar1␣=␣', p_x_bar1)
```

標本数を変えて実行すると，標本数が増えるほど確率が急激に小さくなることが確認できます。

example_cent_limit.py の実行結果（標本数 9）

```
母集団の平均 = ? > 60
母集団の標準偏差 = ? > 5.4
標本数 = ? > 9
標本の平均 = ? > 62
p_x_bar0 =  0.13326026290250537
p_x_bar1 =  0.13326026290250537
```

example_cent_limit.py の実行結果（標本数 49）

```
母集団の平均 = ? > 60
母集団の標準偏差 = ? > 5.4
標本数 = ? > 49
標本の平均 = ? > 62
p_x_bar0 =  0.004762776808383218
p_x_bar1 =  0.004762776808383827
```

example_cent_limit.py の実行結果（標本数 500）

```
母集団の平均 = ? > 60
母集団の標準偏差 = ? > 5.4
標本数 = ? > 500
標本の平均 = ? > 62
p_x_bar0 =  1.1102230246251565e-16
p_x_bar1 =  6.0697638473264710e-17
```

標本数 $n = 9$ では約 13.3% であった確率が，$n = 49$ では約 0.48%，$n = 500$ では事実上 0 になることが確認できます。

演習問題

1. sample_stat.py(p.76) に次の機能を追加せよ。
   (a) x2 の平均，分散，標準偏差，最大値，最小値，中央値の計算と表示
   (b) 分散が標準偏差の 2 乗であることを確認する表示
   (c) 中央値はソート後の中間の値であることを確認する表示
2. Student の t-分布の確率密度関数は scipy.stats.t クラスの pdf 関数で計算できる。これを使い，正規分布と同様に NumPy で計算したグラフと一致することを確認せよ。
3. scipy.integrate.quad を用いて次の定積分を計算し，解析解と比較せよ。
   (a) $\displaystyle\int_0^1 x^3\,dx$
   (b) $\displaystyle\int_0^\pi \sin(x)\,dx$
   (c) $\displaystyle\int_1^\infty \frac{1}{x^2}\,dx$（広義積分）
4. example_cent_limit.py(p.80) を改良し，標本平均が 62kg 以上になる確率が 0.05 以下になる最小の標本数 n を求めるスクリプトにせよ。ループで n を 2 から順に増やし，条件を満たした時点でループを終了する実装例を以下に示す。

ソースコード 8.7　example_cent_limit.py（改良版）

```
12 # 標本平均
13 #str_num_x = input('標本数 = ? > ')
14 #num_x = float(str_num_x)
15 str_x_bar = input('標本の平均␣=␣?␣>␣')
16 x_bar = float(str_x_bar)
17 for num_x in range(2, 1000):
18     # x ごとに y の値を計算: SciPy.stats.norm クラスを使用
19     norm_dist = scstats.norm(loc=mu, scale=sigma/np.sqrt(num_x))
20     p_x_bar0 = 1.0 - norm_dist.cdf(x_bar) # 累積分布関数
21     p_x_bar1 = norm_dist.sf(x_bar) # 1 - cdf(x)
22
23     #print('p_x_bar0 = ', p_x_bar0)
24     #print('p_x_bar1 = ', p_x_bar1)
25     print(num_x, '␣->␣', p_x_bar0, p_x_bar1)
26     if p_x_bar1 <= 0.05: # 0.05以下になったらループストップ
27         break # ループを終了
28
29     # norm_dist を消去
30     del norm_dist
```