Analiza Przeżycia

Raport 1

Wiktor Niedźwiedzki (258882) Filip Michewicz (258882)

Invalid Date

Spis treści

Spis rysunków

Spis tabel


```
# Import necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.stats import expon, gamma

from IPython.display import Markdown  # formatowanie zmiennych do markdown
from tabulate import tabulate  # formatowanie tabelek
```

1 Przykłady użycia pythona

1.1 Losowanie próby z danego rozkładu

Funkcje dotyczące poszczególnych rozkładów możemy uzyskać korzystając z biblioteki *scipy*. Rozkład wykładniczy znajdziemy w klasie *expon*. Liczby losowe (pseudolosowe) generujemy za pomocą metody *rvs*.

```
# Create x values from Exp(2)
x = expon.rvs(loc=0, scale=2, size=10)
# https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.expon.html
```

Inną możliwością jest skorzystanie z biblioteki *numpy*.

```
# Create x values from Exp(2)
x = np.random.exponential(scale=2, size=10)
# https://numpy.org/doc/2.3/reference/random/generated/numpy.random.exponential.html
```

Do uzyskania dystrybuanty możemy użyć metody *cdf*. Analogicznie *pdf* to gęstość, *sf* to funkcja przeżycia, a *ppf* to funkcja kwantylowa.

```
# Create x values
x = np.arange(0, 10.1, 0.1)
n = len(x)

# Initialize list to store data
data_list = []

# Exponential distribution with rate=4 (mean=1/4)
y1 = expon.cdf(x, scale=1/4)
data_list.append(pd.DataFrame({
    'axis_x': x,
    'axis_y': y1,
    'podtytul': 'Dystrybuanta rozkładu wykładniczego',
    'rozklad': 'Exp(4)'
}))

# Exponential distribution with rate=1/4 (mean=4)
y2 = expon.cdf(x, scale=4)
data_list.append(pd.DataFrame({
    'axis_x': x,
```

```
    'axis_y': y2,
    'podtytul': 'Dystrybuanta rozkładu wykładniczego',
    'rozklad': 'Exp(1/4)'
}))

# Gamma distribution with shape=4, rate=2 (scale=1/2)
y3 = gamma.cdf(x, a=4, scale=1/2)
data_list.append(pd.DataFrame({
    'axis_x': x,
    'axis_y': y3,
    'podtytul': 'Dystrybuanta rozkładu gamma',
    'rozklad': 'G(4,2)'
}))

# Gamma distribution with shape=1/4, rate=2 (scale=1/2)
y4 = gamma.cdf(x, a=1/4, scale=1/2)
data_list.append(pd.DataFrame({
    'axis_x': x,
    'axis_y': y4,
    'podtytul': 'Dystrybuanta rozkładu gamma',
    'rozklad': 'G(1/4,2)'
}))

# Combine all data
data = pd.concat(data_list, ignore_index=True)
```

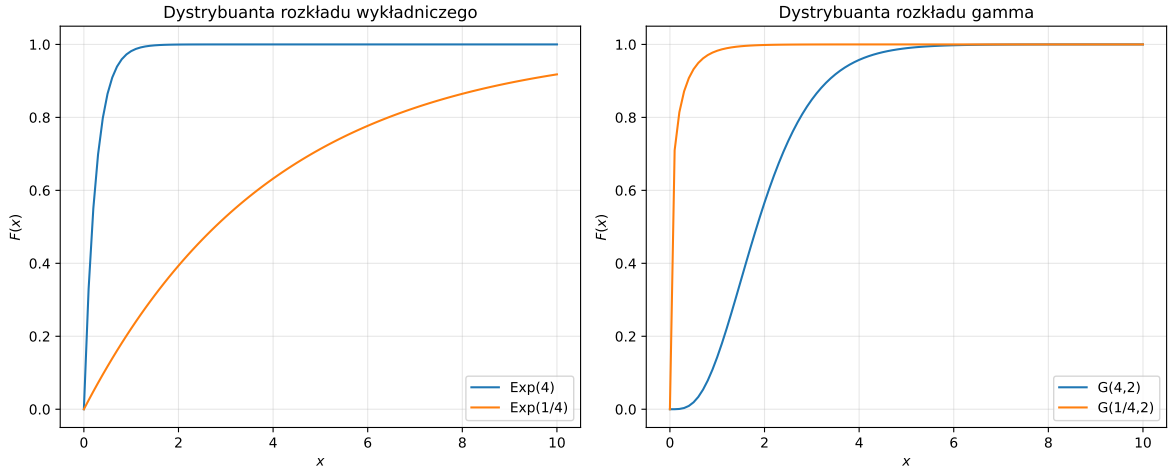
1.2 Generowanie wykresu

Wykresy możemy wygenerować korzystając z biblioteki *matplotlib*.

```
# Create the plot with facets
fig, axes = plt.subplots(1, 2, figsize=(12, 5))

# Plot for exponential distribution
exp_data = data[data['podtytul'] == 'Dystrybuanta rozkładu wykładniczego']
for rozklad in exp_data['rozklad'].unique():
    subset = exp_data[exp_data['rozklad'] == rozklad]
    axes[0].plot(subset['axis_x'], subset['axis_y'], label=rozklad)
axes[0].set_xlabel('$x$')
axes[0].set_ylabel('$F(x)$')
# stosujemy dolary robiac podpisy osi, etykiety w legendzie itp.!
```

```
axes[0].set_title('Dystrybuanta rozkładu wykładniczego')
axes[0].legend()
axes[0].grid(True, alpha=0.3)

# Plot for gamma distribution
gamma_data = data[data['podtytul'] == 'Dystrybuanta rozkładu gamma']
for rozklad in gamma_data['rozklad'].unique():
    subset = gamma_data[gamma_data['rozklad'] == rozklad]
    axes[1].plot(subset['axis_x'], subset['axis_y'], label=rozklad)
axes[1].set_xlabel('$x$')
axes[1].set_ylabel('$F(x)$')
axes[1].set_title('Dystrybuanta rozkładu gamma')
axes[1].legend()
axes[1].grid(True, alpha=0.3)

plt.tight_layout()
plt.savefig('example_plot.pdf')
plt.close()
```

Kod napisany powyżej służy do wygenerowania wykresu 1 przedstawiającego wykresy dystrybuant rozkładu wykładniczego i rozkładu gamma z różnymi parametrami. W kodzie powyższego chunka jest pokazane jak zmieniać opcję poszczególnych chunków - w tym przypadku *echo: true* oznacza, że wyświetlamy kod w pliku pdf, a *echo: false* oznacza, że jest on ukryty.



Rysunek 1: Przykładowy wykres

1.3 Przykładowa tabelka

```
# Set random seed for reproducibility
np.random.seed(123456)
# Create x values from Exp(1)
x = expon.rvs(loc=0, scale=1, size=100)
# Create y values from Exp(5)
y = expon.rvs(loc=0, scale=1/5, size=100)

# Calculate summary statistics for x
stat_opis_x = [
    np.min(x),                    # Min
    np.percentile(x, 25),         # 1st Quartile
    np.median(x),                 # Median
    np.mean(x),                   # Mean
    np.percentile(x, 75),         # 3rd Quartile
    np.max(x),                    # Max
    np.std(x, ddof=1)             # Standard Deviation (sample std)
]

stat_opis_y = [
    np.min(y),                    # Min
    np.percentile(y, 25),         # 1st Quartile
    np.median(y),                 # Median
    np.mean(y),                   # Mean
    np.percentile(y, 75),         # 3rd Quartile
    np.max(y),                    # Max
    np.std(y, ddof=1)             # Standard Deviation (sample std)
]

# Create a DataFrame with row and column names
stat_opis_df = pd.DataFrame(
    [stat_opis_x, stat_opis_y],
    columns=["Min", "Pierwszy Kw.", "Mediana", "Średnia", "Trzeci Kw.", "Max", "Odch.Stand."]
    index=["$x$", "$y$"]
)

Markdown(
  tabulate(stat_opis_df, showindex=True,
           headers=stat_opis_df.columns)
)
```

```
# more about tables:
# https://quarto.org/docs/authoring/tables.html

# wiecej o odnoszeniu sie do tabel i wykresow:
# https://quarto.org/docs/authoring/cross-references.html

# Type          Syntax              Output
# Default         @fig-elephant       Figure 1
# Capitalized     @Fig-elephant       Figure 1
# Custom Prefix [Fig @fig-elephant] Fig 1
# No Prefix     [-@fig-elephant]      1

# wiec jesli chcemy w tekscie sam numerek 1, a nie Table 1, to robimy [-@tbl-stats]
```

Tabela 1: Podstawowe statystyki z dwóch wygenerowanych próbek

	Min	Pierwszy Kw.	Mediana	Średnia	Trzeci Kw.	Max	Odch.Stand.
x	0.0166829	0.278261	0.629113	0.948054	1.2382	4.50527	0.959858
y	0.00389279	0.06599	0.153721	0.215144	0.25391	0.995943	0.209821

W Tabeli 1 widzimy podsumowanie wygenerowanych próbek.

---

2 Podsumowanie

Tu można umieścić podsumowanie raportu.