

# Analiza Przeżycia

## Raport 1

Wiktor Niedźwiedzki (258882)      Filip Michewicz (282239)

6 listopada 2025 Anno Domini

### Spis treści

<b>1</b>	<b>Lista 1</b>	<b>3</b>
1.1	Zadanie 1 . . . . .	3
1.1.1	Funkcja gęstości . . . . .	3
1.1.2	Dystrybuanta . . . . .	3
1.1.3	Funkcja kwantylowa (dystrybuanta odwrotna) . . . . .	4
1.1.4	Funkcja hazardu . . . . .	4
1.2	Zadanie 2 . . . . .	5
1.3	Zadanie 3 . . . . .	6
1.4	Zadanie 4 . . . . .	7
1.5	Zadanie 5 . . . . .	8
1.6	Zadanie dodatkowe 1 . . . . .	9
1.6.1	Metoda parabol . . . . .	10
1.6.2	Metoda Monte Carlo . . . . .	11
1.6.3	Wyniki . . . . .	11
<b>2</b>	<b>Lista 2</b>	<b>13</b>
2.1	Zadanie 1 . . . . .	13
2.1.1	Cenzurowanie I typu . . . . .	13
2.1.2	Cenzurowanie II typu . . . . .	14
2.1.3	Cenzurowanie losowe . . . . .	15
2.2	Zadanie 2 . . . . .	16
2.3	Zadanie 3 . . . . .	17
<b>3</b>	<b>Lista 3</b>	<b>19</b>
3.1	Zadanie 1 . . . . .	19
3.1.1	Estymacja punktowa . . . . .	19
3.1.2	Estymacja przedziałowa . . . . .	20

3.2	Zadanie 2 . . . . .	23
3.2.1	Estymacja punktowa . . . . .	23
3.2.2	Estymacja przedziałowa . . . . .	24
3.3	Zadanie 3 . . . . .	26
3.4	Zadanie dodatkowe 1 . . . . .	28
3.5	Zadanie dodatkowe 2 . . . . .	29
3.5.1	Estymacja punktowa . . . . .	30
3.5.2	Estymacja przedziałowa . . . . .	31
<b>4</b>	<b>Lista 4</b>	<b>32</b>
4.1	Zadanie 1 . . . . .	32
4.2	Zadanie 2 . . . . .	34
4.3	Zadanie 3 . . . . .	36
4.4	Zadanie dodatkowe . . . . .	37

## Spis wykresów

1	Funkcja hazardu - rozkład rozszerzony Weibulla - przykładowe parametry	6
2	Histogram i gęstość teoretyczna dla próby z rozkładu $\mathcal{EW}(1, 2, 1)$ - liczność prób $n = 50$ oraz $n = 100$ . . . . .	7
3	Histogram i gęstość teoretyczna dla próby z rozkładu $\mathcal{EW}(\frac{1}{2}, 1, 4)$ - liczność prób $n = 50$ oraz $n = 100$ . . . . .	8

## Spis tabel

1	Podstawowe statystyki opisowe prób z rozkładu $\mathcal{EW}(\alpha, \beta, \gamma)$ . . . . .	9
2	Wartości oczekiwane rozkładu $\mathcal{EW}(\alpha, \beta, \gamma)$ dla wybranych parametrów . . . . .	11
3	Wariancje rozkładu $\mathcal{EW}(\alpha, \beta, \gamma)$ dla wybranych parametrów . . . . .	12
4	Podstawowe statystyki opisowe danych cenzurowanych wygenerowanych z rozkładu $\mathcal{GE}(1, 1.5)$ . . . . .	17
5	Podstawowe statystyki opisowe danych czasu remisji choroby po stosowaniu leku A lub leku B - dane cenzurowane I typu . . . . .	18
6	Przedziały ufności dla danych cenzurowanych I typu - różne poziomy ufności . . . . .	22
7	Realizacje zbiorów ufności dla danych cenzurowanych II typu - różne poziomy ufności . . . . .	25
8	Porównanie estymatorów $\hat{\vartheta}$ oraz $\tilde{\vartheta}$ . . . . .	28
9	Wyniki symulacji mocy testu dwustronnego . . . . .	35
10	Testowanie hipotezy dwustronnej na danych rzeczywistych . . . . .	36

# 1 Lista 1

Lista pierwsza obejmuje analizę wykładniczego rozkładu Weibulla  $\mathcal{EW}(\alpha, \beta, \gamma)$ ; definicje jego funkcji, generowanie danych, wizualizację oraz porównanie statystyk empirycznych i teoretycznych.

## 1.1 Zadanie 1

Zadanie polega na deklaracji:

- funkcji gęstości,
- dystrybuanty,
- funkcji kwantylowej,
- funkcji hazardu,

wykładniczego rozkładu Weibulla  $\mathcal{EW}(\alpha, \beta, \gamma)$ .

### 1.1.1 Funkcja gęstości

Na wykładzie została podana funkcja gęstości rozkładu  $\mathcal{EW}(\alpha, \beta, \gamma)$ , wyrażana wzorem:

$$f(x) = \frac{\alpha\gamma}{\beta} \left(\frac{x}{\beta}\right)^{\alpha-1} \left(1 - \exp\left(-\left(\frac{x}{\beta}\right)^\alpha\right)\right)^{\gamma-1} \exp\left(-\left(\frac{x}{\beta}\right)^\alpha\right) \mathbf{1}_{(0;\infty)}(x)$$

Odpowiadający tej funkcji gęstości kod w języku Python ma postać:

```
def EW_density(x, alpha, beta, gamm):  
    return(alpha * gamm / beta * (x / beta)**(alpha - 1)*  
           (1 - np.exp(-(x / beta)**alpha))**(gamm - 1)*  
           np.exp(-(x / beta)**alpha))
```

### 1.1.2 Dystrybuanta

Na wykładzie została podana dystrybuanta rozkładu  $\mathcal{EW}(\alpha, \beta, \gamma)$ , która wyrażona jest wzorem:

$$F(t) = \left(1 - \exp\left(-\left(\frac{x}{\beta}\right)^\alpha\right)\right)^\gamma \mathbf{1}_{(0;\infty)}(t)$$

Odpowiadający tej dystrybuancie kod w języku Python przedstawiono poniżej.

```
def EW_distribution(t, alpha, beta, gamm):
    return (1 - np.exp(-(t / beta)**alpha))**gamm
```

### 1.1.3 Funkcja kwantylowa (dystrybuanta odwrotna)

Funkcja kwantylowa została wyznaczona z dystrybuanty jako funkcja do niej odwrotna. Oznacza to, że dla danej wartości prawdopodobieństwa  $p \in (0, 1)$  szukamy takiej wartości  $t$ , dla której  $F(t) = p$ .

Punktem wyjścia jest dana dystrybuanta:  $F(t) = \left(1 - \exp\left(-\left(\frac{t}{\beta}\right)^\alpha\right)\right)^\gamma$

Podstawiamy  $F(t) = p$  i rozwiązujemy równanie względem  $t$ :

$$\begin{aligned} p &= \left(1 - \exp\left(-\left(\frac{t}{\beta}\right)^\alpha\right)\right)^\gamma \\ p^{1/\gamma} &= 1 - \exp\left(-\left(\frac{t}{\beta}\right)^\alpha\right) \\ \exp\left(-\left(\frac{t}{\beta}\right)^\alpha\right) &= 1 - p^{1/\gamma} \\ \left(\frac{t}{\beta}\right)^\alpha &= -\ln(1 - p^{1/\gamma}) \\ t^\alpha &= \beta^\alpha(-\ln(1 - p^{1/\gamma})) \\ t &= \beta[-\ln(1 - p^{1/\gamma})]^{1/\alpha} \\ Q(p) &= \beta[-\ln(1 - p^{1/\gamma})]^{1/\alpha} \end{aligned}$$

Otrzymana funkcja jest funkcją kwantylową (odwrotnością dystrybuanty) rozkładu  $\mathcal{EW}(\alpha, \beta, \gamma)$  i pozwala dla zadanego prawdopodobieństwa  $p$  wyznaczyć wartość dystrybuanty  $t$ .

Odpowiadający tej funkcji kandydowi kod w języku Python ma postać:

```
def EW_quantile(p, alpha, beta, gamm):
    return beta * (-np.log(1.0 - p**(1.0 / gamm)))**(1.0 / alpha)
```

### 1.1.4 Funkcja hazardu

Funkcja hazardu rozkładu  $\mathcal{EW}(\alpha, \beta, \gamma)$  wyrażona jest wzorem:

$$h(x) = \frac{f(x)}{1 - F(x)} = \frac{\alpha \gamma \left(\frac{x}{\beta}\right)^{\alpha-1} \left(1 - \exp\left(-\left(\frac{x}{\beta}\right)^\alpha\right)\right)^{\gamma-1} \exp\left(-\left(\frac{x}{\beta}\right)^\alpha\right)}{\beta \left(1 - \left(1 - \exp\left(-\left(\frac{x}{\beta}\right)^\alpha\right)\right)^\gamma\right)} \mathbf{1}_{(0;\infty)}(x)$$

Odpowiadający tej funkcji gęstości kod w języku Python ma postać:

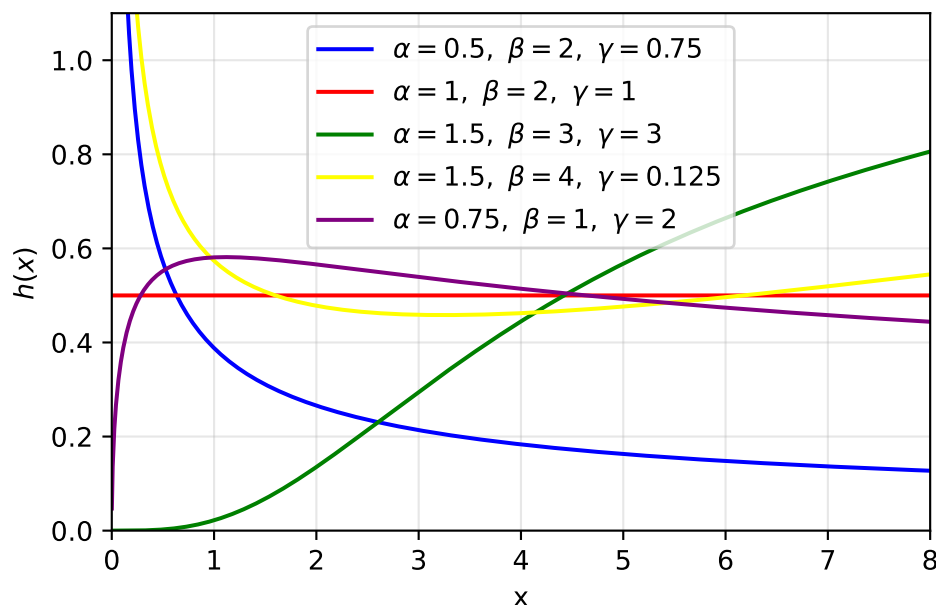
```
def EW_hazard(x, alpha, beta, gamm):
    f = EW_density(x, alpha, beta, gamm)
    F = EW_distribution(x, alpha, beta, gamm)
    if F == 1:
        return np.inf
    return f / (1 - F)
```

## 1.2 Zadanie 2

W tym zadaniu wygenerowano wykresy przykładowych funkcji hazardu rozkładu  $\mathcal{EW}(\alpha, \beta, \gamma)$  dla 5 różnych trójek parametrów.

Użyte parametry są następujące:

- $\alpha = \frac{1}{2}, \beta = 2, \gamma = \frac{3}{4},$
- $\alpha = 1, \beta = 2, \gamma = 1,$
- $\alpha = \frac{3}{2}, \beta = 3, \gamma = 3,$
- $\alpha = \frac{3}{2}, \beta = 4, \gamma = \frac{1}{8},$
- $\alpha = \frac{3}{4}, \beta = 1, \gamma = 2.$



Wykres 1: Funkcja hazardu - rozkład rozszerzony Weibulla - przykładowe parametry

**Wykres 1.** przedstawia funkcję hazardu rozkładu  $\mathcal{EW}(\alpha, \beta, \gamma)$  dla pięciu różnych trójek parametrów. Widać, że w zależności od doboru parametrów funkcja hazardu może być malejąca, rosnąca, stała, unimodalna lub przyjmować kształt „wanny”.

### 1.3 Zadanie 3

W tym zadaniu skonstruowano funkcję generującą zmienne losowe z rozkładu  $\mathcal{EW}(\alpha, \beta, \gamma)$ . W [Zadaniu 1.](#) została wyznaczona funkcja kwantylowa  $F^{-1}(p)$ , dlatego korzystając z metody dystrybuanty odwrotnej można wygenerować zmienne losowe w następujący sposób:

1. Generujemy zmienną losową z rozkładu jednostajnego  $U_i \sim \mathcal{U}(0, 1)$ ,
2. Obliczamy  $X_i = F^{-1}(U_i)$ .

Otrzymana zmienna losowa  $X_i$  ma rozkład jednoznacznie określony przez dystrybuantę  $F(t)$ , czyli w naszym przypadku przez dystrybuantę rozkładu  $\mathcal{EW}(\alpha, \beta, \gamma)$ .

Poniżej przedstawiono kod w Pythonie realizujący ten algorytm.

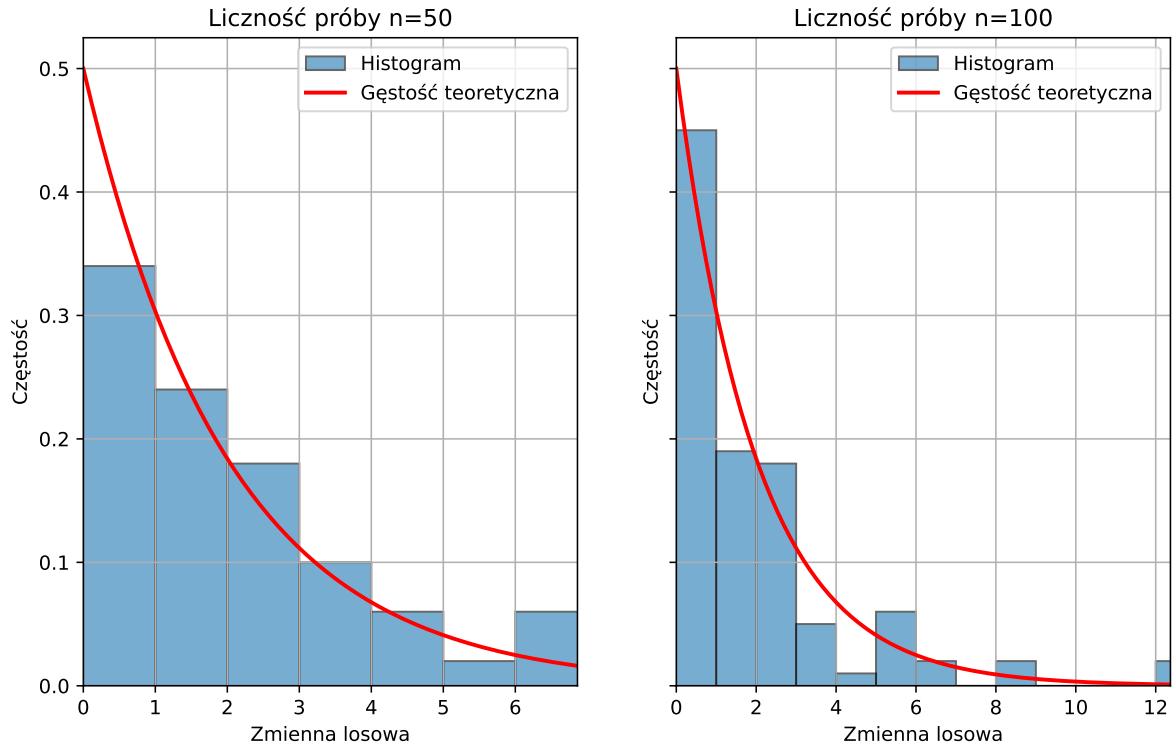
```
def EW_generator(alpha, beta, gamm, size=1):
    u = np.random.rand(size)
    return(EW_quantile(u, alpha, beta, gamm))
```

## 1.4 Zadanie 4

W tym zadaniu wygenerowano realizację próby z rozkładu  $\mathcal{EW}(\alpha, \beta, \gamma)$  o licznosci  $n = 50$  oraz  $n = 100$  dla następujących parametrów:

- $\alpha = 1, \beta = 2, \gamma = 1,$
- $\alpha = \frac{1}{2}, \beta = 1, \gamma = 4.$

Wyniki przedstawiono na poniższych wykresach.

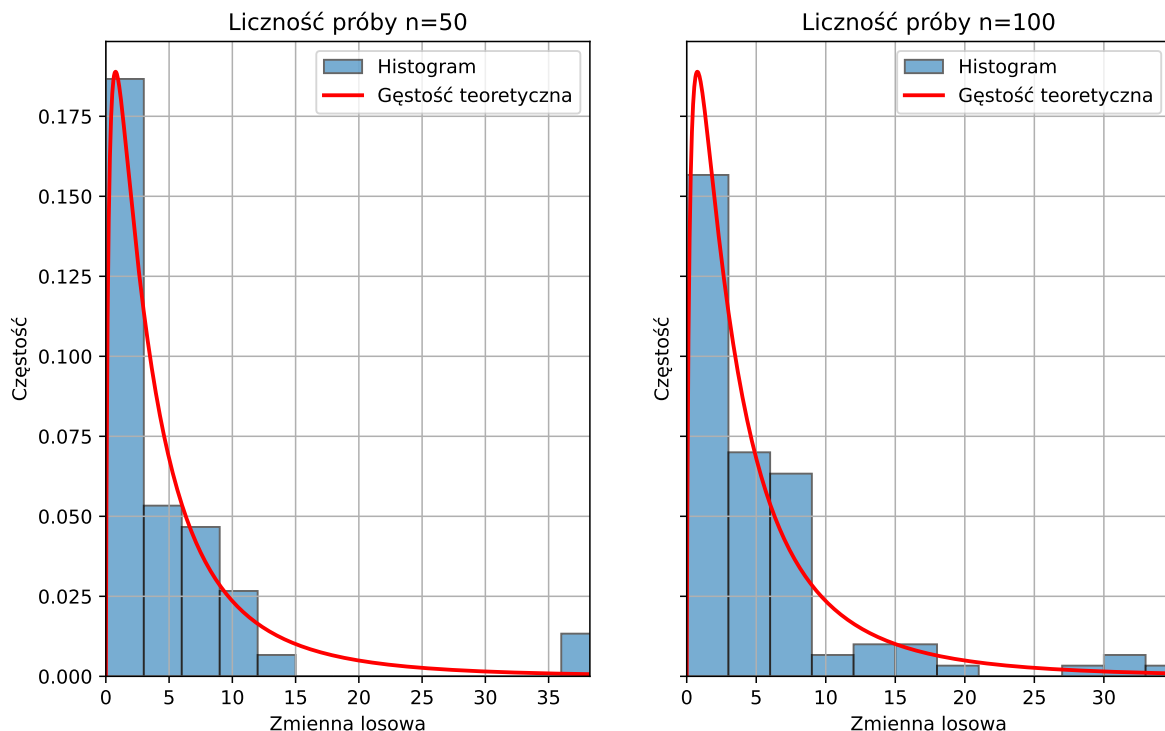


Wykres 2: Histogram i gęstość teoretyczna dla próby z rozkładu  $\mathcal{EW}(1, 2, 1)$  - licznosc prób  $n = 50$  oraz  $n = 100$

Z Wykresu 2. można odczytać, że histogram próby jest bardzo podobny do funkcji gęstości, jednak pewne różnice wynikają z małej licznosci próby  $n = 50$  - w tym przypadku Prawo Wielkich Liczb jeszcze nie zapewnia dokładnego przybliżenia rozkładu prawdopodobieństwa.

Dla większej próby  $n = 100$  dopasowanie histogramu do gęstości wygląda lepiej, choć nadal widać drobne odchylenia.

Mimo to można stwierdzić, że kod poprawnie generuje zmienne losowe z rozkładu  $\mathcal{EW}(\alpha, \beta, \gamma)$ , co potwierdza wizualne porównanie histogramu z krzywą funkcji gęstości.



Wykres 3: Histogram i gęstość teoretyczna dla próby z rozkładu  $\mathcal{EW}(\frac{1}{2}, 1, 4)$  - licznosc prób  $n = 50$  oraz  $n = 100$

Na Wykresie 3., podobnie jak na Wykresie 2., histogram próby jest bardzo podobny do funkcji gęstości, choć nadal widać pewne odchylenia wynikające z ograniczonej licznosci próby  $n = 50$  lub  $n = 100$ .

Tym razem zakres wartości, z którego wylosowały się zmienne losowe, jest większy, jednak licznosc próby pozostała bez zmian.

Mimo zmiany parametrów rozkładu, histogram nadal wizualnie przypomina funkcję gęstości, co stanowi dodatkowe potwierdzenie poprawnego działania algorytmu generującego zmienne losowe z rozkładu  $\mathcal{EW}(\alpha, \beta, \gamma)$ .

## 1.5 Zadanie 5

W tym zadaniu wyznaczono podstawowe statystyki opisowe (zarówno teoretyczne, jak i empiryczne) dla prób wygenerowanych w poprzednim zadaniu.



Tabela 1: Podstawowe statystyki opisowe prób z rozkładu  $\mathcal{EW}(\alpha, \beta, \gamma)$ 

$(\alpha, \beta, \gamma)$	(1, 2, 1)	(1, 2, 1)	(0.5, 1, 4)	(0.5, 1, 4)
Liczność próby	50	100	50	100
Średnia (emp.)	2.05234	2.0419	4.9875	5.50854
Mediana (emp.)	1.53182	1.50667	2.78832	3.3383
Mediana (teor.)	1.38629	1.38629	3.37898	3.37898
Odchylenie standardowe	1.7655	2.3235	7.4733	6.6353
Kwartył dolny (emp.)	0.7261	0.4517	0.8591	1.5337
Kwartył dolny (teor.)	0.5754	0.5754	1.5079	1.5079
Kwartył górny (emp.)	2.6941	2.7576	6.4328	6.5514
Kwartył górny (teor.)	2.7726	2.7726	7.1179	7.1179
Rozstęp	6.8518	12.3463	38.1243	34.7186
Rozstęp międzykwartyłowy (emp.)	1.968	2.3059	5.5737	5.0177
Rozstęp międzykwartyłowy (teor.)	2.1972	2.1972	5.61	5.61
Minimum	0.0116	0.0227	0.139	0.0919
Maksimum	6.8634	12.369	38.2633	34.8105

Z Tabeli 1. można odczytać, że wartości statystyk empirycznych (średnia, mediana, odchylenie standardowe, kwartyły) w przybliżeniu odpowiadają wartościom teoretycznym, choć dla mniejszych prób  $n = 50$  różnice są bardziej widoczne. Wynika z tego, że im większe  $n$ , tym lepsze przybliżenie rozkładu otrzymujemy. Dowód tego faktu uzasadnia twierdzenie Glivenki–Cantellego, które stwierdza, że empiryczna dystrybuanta zbiega jednostajnie do dystrybuanty rzeczywistej, gdy  $n \rightarrow \infty$

## 1.6 Zadanie dodatkowe 1

W tym zadaniu zadeklarowane zostały funkcje do liczenia wartości oczekiwanej oraz wariancji zmiennej losowej o rozkładzie  $\mathcal{EW}(\alpha, \beta, \gamma)$ , który jest absolutnie ciągły. Z tego powodu można zdefiniować ogólny wzór na  $n$ -ty moment oraz wariancję jako:

$$\mathbb{E}[X^n] = \int_{\mathbb{R}} x^n f(x) dx,$$

$$Var[X] = \mathbb{E}[X^2] - (\mathbb{E}[X])^2$$

Dla jak najlepszej dokładności wyników, przedziałem całkowania będzie  $[0; Q(0.99999)]$ , gdzie  $Q(\cdot)$  - funkcja kwantylowa. Całka została policzona na dwa sposoby: metodą parabol (Simpsona) oraz metodą Monte Carlo.

### 1.6.1 Metoda parabol

Poniżej znajduje się kod w języku Python, realizujący metodę Simpsona.

```
def simpson_rule(a, b, function, n):

    if n%2 != 0: raise Exception("The number of intervals must be even.")

    x_points = np.linspace(a, b, n+1)
    y_points = []
    for i in x_points: y_points.append(function(i))

    h = (b-a)/(n)
    integral = 0
    integral += y_points[0]
    even = False

    for i in range(1, n):
        if even:
            integral += 2*y_points[i]
            even = False
        else:
            integral += 4*y_points[i]
            even = True

    integral += y_points[n]
    integral = h*integral/3

    return(integral)

def expected_value_simpson(alpha, beta, gamma, moment = 1):
    quan = EW_quantile(0.99999, alpha, beta, gamma)
    intervals = floor(quan/0.01)
    if intervals % 2 != 0: intervals += 1
    return(simpson_rule(0.001, quan,
        lambda x: x**moment * EW_density(x, alpha, beta, gamma),
        intervals))

def variance_simpson(alpha, beta, gamma):
    return(expected_value_simpson(alpha, beta, gamma, 2) -
        expected_value_simpson(alpha, beta, gamma)**2)
```

### 1.6.2 Metoda Monte Carlo

W metodzie skorzystano z Mocnego Prawa Wielkich Liczb, ponieważ rozkład  $\mathcal{EW}(\alpha, \beta, \gamma)$  jest dodatni i ma skończone momenty pierwszego i drugiego rzędu, dzięki czemu średnia i wariancja próbkowa z niezależnych prób zbiega prawie na pewno do wartości oczekiwanej i wariancji rozkładu, co uzasadnia zastosowanie symulacji Monte Carlo.

Poniżej przedstawiono kod w języku Python, który przeprowadza symulację Monte Carlo w celu przybliżonego oszacowania wartości oczekiwanej oraz wariancji rozkładu.

```
parameters = [(1, 2, 1),
               (1/2, 1, 4)]
n_sim = 100000000
ew_MC = []
var_MC = []

for alpha, beta, gamm in parameters:
    sample = EW_generator(alpha, beta, gamm, n_sim)

    ew_MC.append(np.mean(sample))
    var_MC.append(np.var(sample, ddof=1))
```

### 1.6.3 Wyniki

Tabela przedstawia wyniki uzyskane metodą parabol oraz metodą Monte Carlo. Dla weryfikacji poprawności obliczeń można dobrać parametry tak, aby rozkład przyjmował postać klasycznego wykładniczego lub rozszerzonego wykładniczego, a następnie porównać wartości obliczone numerycznie z wynikami analitycznymi.

Tabela 2: Wartości oczekiwane rozkładu  $\mathcal{EW}(\alpha, \beta, \gamma)$  dla wybranych parametrów

Metoda	$\alpha=1, \beta=2, \gamma=1$	$\alpha=0.5, \beta=1, \gamma=4$
Parabol	1.99975	5.76195
Monte Carlo	2.00022	5.7638

Tabela 3: Wariancje rozkładu  $\mathcal{EW}(\alpha, \beta, \gamma)$  dla wybranych parametrów

Metoda	$\alpha=1, \beta=2, \gamma=1$	$\alpha=0.5, \beta=1, \gamma=4$
Parabol	3.9947	54.5054
Monte Carlo	4.00132	54.8559

W Tabeli 2. i Tabeli 3. przedstawiono empiryczne wartości oczekiwane oraz wariancje rozkładu  $\mathcal{EW}(\alpha, \beta, \gamma)$  dla wybranych parametrów; widać, że wszystkie zastosowane metody numeryczne dają bardzo zbliżone wyniki, co potwierdza ich poprawność i stabilność estymacji. Dodatkowo, poprawność potwierdza również znany nam wynik dla  $\mathcal{EW}(1, 2, 1)$ , który jest równoznaczny z rozkładem wykładniczym  $\mathcal{E}(\beta = 2)$ , którego wartość oczekiwana i wariancja są znane ( $\mathbb{E}[X] = \beta = 2$ ,  $Var[X] = \beta^2 = 4$ ).

## 2 Lista 2

Lista 2 obejmuje generowanie danych cenzurowanych z rozkładu uogólnionego wykładniczego  $\mathcal{GE}(\lambda, \alpha)$  oraz opis statystyk opisowych zarówno dla wygenerowanych danych cenzurowanych, jak i dla podanych danych cenzurowanych.

### 2.1 Zadanie 1

W tym zadaniu zadeklarowano funkcje do prawostronnego cenzurowania danych kompletnych: cenzurowanie typu I, typu II oraz losowe. Chociaż zadanie dotyczy głównie danych z rozkładu  $\mathcal{GE}(\lambda, \alpha)$ , funkcje te można łatwo rozszerzyć na prawostronnie cenzurowane dane z dowolnego rozkładu, podając zamiast parametrów  $\mathcal{GE}(\lambda, \alpha)$  dane kompletne. Pozwoli to potencjalnie na analizę i symulację cenzurowanych danych w szerszym kontekście.

W kontekście zadania łatwo widać że rozkład uogólniony wykładniczy  $\mathcal{GE}(\lambda, \alpha)$  jest szczególnym przypadkiem rozkładu wykładniczego Weibulla  $\mathcal{EW}(\alpha, \beta, \gamma)$  gdzie,

$$\mathcal{GE}(\lambda, \alpha) = \mathcal{EW}(1, \lambda, \alpha)$$

Dlatego też dane z tego rozkładu można generować za pomocą funkcji z [Listy 1. Zadania 3](#).

Podobnie rozkład wykładniczy  $\mathcal{E}(\lambda)$  jest szczególnym przypadkiem uogólnionego rozkładu wykładniczego  $\mathcal{GE}(\lambda, \alpha)$ , a tym samym wykładniczego rozkładu Weibulla  $\mathcal{EW}(\alpha, \beta, \gamma)$  gdzie,

$$\mathcal{E}(\lambda) = \mathcal{GE}(\lambda, 1) = \mathcal{EW}(1, \lambda, 1)$$

Informację na temat danych cenzurowanych zawarto w tzw. indykatorze cenzurowania:

$$\delta_i = \begin{cases} 1, & \text{jeśli dane są kompletne (niecenzurowane),} \\ 0, & \text{jeśli dane są niekompletne (cenzurowane).} \end{cases}$$

#### 2.1.1 Cenzurowanie I typu

Cenzurowanie typu I polega na niewyznaczaniu wartości zmiennej losowej powyżej pewnego czasu  $t_0$ . Oznacza to, że jeśli  $X_i$  przekracza  $t_0$ , to jest ona zastępowana przez  $t_0$ .

Niech  $X_i$  oznacza zaobserwowane czasy życia. Wówczas zmienne losowe po cenzurowaniu definiuje się jako:

$$T_i = \begin{cases} X_i, & X_i \leq t_0 \quad (\delta_i = 1), \\ t_0, & X_i > t_0 \quad (\delta_i = 0), \end{cases} \quad \text{czyli} \quad T_i = \min(X_i, t_0).$$

gdzie  $\delta_i$  jest indykátorem cenzurowania.

Poniżej przedstawiono kod w Pythonie realizujący cenzurowanie I typu dla dowolnych danych:

```
def cenzurowanie_I_typu(t_0, data):
    size = len(data)

    censored_data = []
    deltas = []

    for x in data:
        if x <= t_0:
            censored_data.append(x)
            deltas.append(1)
        else:
            censored_data.append(t_0)
            deltas.append(0)

    return censored_data, deltas
```

Poniżej przedstawiono kod w Pythonie konkretnie do generowania danych cenzurowanych I typu z rozkładu  $\mathcal{GE}(\lambda, \alpha)$ :

```
def GE_cenzurowanie_I_typu(t0, lambd, alpha, n):
    data = EW_generator(1, lambd, alpha, size=n)
    return cenzurowanie_I_typu(t0, data)
```

### 2.1.2 Cenzurowanie II typu

Cenzurowanie typu II polega na obserwacji jedynie  $m$  najmniejszych wartości zmiennych losowych, a pozostałe wartości są zastępowane przez  $X_{(m)}$ . Oznacza to, że zmierzono tylko do uzyskania  $m \leq n$  danych kompletnych.

Niech  $X_{(k)}$  oznacza  $k$ -tą statystykę pozycyjną. Wówczas zmienne losowe cenzurowane definiuje się jako:

$$T_i = \begin{cases} X_{(i)}, & i = 1, \dots, m \quad (\delta_i = 1), \\ X_{(m)}, & i = m + 1, \dots, n \quad (\delta_i = 0), \end{cases} \quad \text{czyli} \quad T_i = \min(X_{(i)}, X_{(m)}).$$

gdzie  $\delta_i$  jest indykátorem cenzurowania.

Poniżej przedstawiono kod w Pythonie realizujący cenzurowanie II typu dla dowolnych danych:

```
def cenzurowanie_II_typu(m, data):
    censored_data = np.sort(data)

    # Wszystkie obserwacje powyżej m-tej zamieniamy na X_(m)
    censored_data[m:] = censored_data[m-1]

    deltas = np.zeros(len(data))
    deltas[:m] = 1

    return censored_data, deltas
```

Poniżej przedstawiono kod w Pythonie konkretnie do generowania danych cenzurowanych I typu z rozkładu  $\mathcal{GE}(\lambda, \alpha)$ :

```
def GE_cenzurowanie_II_typu(m, lambd, alpha, n):
    data = EW_generator(1, lambd, alpha, size=n)
    return cenzurowanie_II_typu(m, data)
```

### 2.1.3 Cenzurowanie losowe

Cenzurowanie losowe jest rozszerzeniem cenzurowania typu I, z tym że  $t_0$  nie jest już wartością stałą, lecz zmienną losową  $C_i$ , różną dla każdej obserwacji. Zmienna  $C_i$  nazywana jest zmienną cenzurującą i ma rozkład  $g$  oraz dystrybuantę  $G$  (nie musi być taka sama jak dla zmiennej cenzurowanej). Zmienna cenzurująca  $C_i$  jest niezależna od  $X_i$  i przyjmuje tę samą rolę dla każdej jednostki.

$$T_i = \begin{cases} X_i, & X_i \leq C_i \quad (\delta_i = 1), \\ C_i, & X_i > C_i \quad (\delta_i = 0), \end{cases} \quad \text{czyli} \quad T_i = \min(X_i, C_i).$$

gdzie  $\delta_i$  jest indykátorem cenzurowania.

Poniżej przedstawiono kod w Pythonie realizujący cenzurowanie losowe dla dowolnych danych:

```
def cenzurowanie_losowe(cen, data):
    deltas = (data <= cen).astype(int)
    censored_data = np.minimum(data, cen)
    return censored_data, deltas
```

Poniżej przedstawiono kod w Pythonie konkretnie do generowania danych cenzurowanych losowo z rozkładu  $\mathcal{GE}(\lambda, \alpha)$ , cenzurowanych zmienną losową z rozkładu wykładniczego  $\mathcal{E}(\lambda)$ :

```
def GE_cenzurowanie_losowe(eta, lamb, alpha, n):  
    data = EW_generator(1, lambd, alpha, size=n)  
    cen = EW_generator(1, eta, 1, size=n)  
    return cenzurowanie_losowe(cen, data)
```

## 2.2 Zadanie 2

W tym zadaniu wygenerowano próbę o rozmiarze  $n = 200$  z rozkładu  $\mathcal{GE}(1, 1.5)$ , a następnie zastosowano cenzurowanie typu I, typu II oraz losowe.

- W cenzurowaniu typu I wybrano parametr  $t_0 = 1$ .
- W cenzurowaniu typu II wybrano parametr  $m = 120$ .
- W przypadku cenzurowania losowego wygenerowano próbę zmiennych cenzurujących z rozkładu  $\mathcal{E}(1)$ .

Poniżej przedstawiono kod w Pythonie generujący wspomniane wyżej dane cenzurowane.

```
lambd = 1.0  
alpha = 1.5  
size = 200  
  
# Parametry cenzurowania  
t0 = 1.0  
m = 120  
eta = 1.0  
  
data = EW_generator(1, lambd, alpha, size)  
cen = EW_generator(1, eta, 1, size)  
  
times_I, deltas_I = cenzurowanie_I_typu(t0, data)  
times_II, deltas_II = cenzurowanie_II_typu(m, data)  
times_R, deltas_R = cenzurowanie_losowe(cen, data)
```

W następnej kolejności wyliczono podstawowe statystyki opisowe wygenerowanych danych. Jednak w przypadku danych cenzurowanych pominięto średnią i odchylenie standardowe, ponieważ klasyczne wzory na te statystyki zakładają pełne obserwacje i nie uwzględniają faktu, że niektóre wartości są ograniczone przez cenzurę. Obliczenie średniej lub odchylenia standardowego z użyciem klasycznych formuł mogłoby prowadzić do obciążonych i niepoprawnych wyników.



Zamiast tego dodano informację o liczbie danych cenzurowanych, co jest kluczowe w analizie takich danych, ponieważ pozwala ocenić stopień niepełności próby i wpływ cenzury na interpretację statystyk opisowych.

Tabela 4: Podstawowe statystyki opisowe danych cenzurowanych wygenerowanych z rozkładu  $\mathcal{GE}(1, 1.5)$

Rodzaj cenzurowania	I typu ( $t_0 = 1.0$ )	II typu ( $m = 120$ )	Losowe ( $\eta = 1.0$ )
Rozmiar próby	200	200	200
Liczba obserwacji kompletnych	103	120	87
Kwantyl dolny (Q1)	0.4493	0.4493	0.2049
Mediana	0.9372	0.9372	0.4321
Kwantyl górny (Q3)	1	1.2715	0.8591
Rozstęp międzykwartylowy (IQR)	0.5507	0.8222	0.6542
Minimum	0.0166	0.0166	0.0029
Maksimum	1	1.2715	3.0092
Rozstęp	0.9834	1.2549	3.0062

Z Tabeli 4. można odczytać m.in., że cenzurowanie wpływa na rozkład danych. Cenzurowanie typu I i typu II są do siebie zbliżone pod względem sposobu ograniczania obserwacji - w cenzurowaniu typu I z góry ustalany jest czas obserwacji  $t_0$ , na podstawie którego wyznaczana jest liczba obserwacji kompletnych, natomiast w cenzurowaniu typu II określa się liczbę obserwacji kompletnych  $m$ , a odpowiadający jej czas obserwacji  $X_{(m)}$  wyznaczany jest na podstawie danych. W przypadku cenzurowania losowego sytuacja wygląda inaczej, ponieważ moment cenzurowania nie jest ustalony z góry, lecz stanowi zmienną losową  $C_i$ , co wprowadza dodatkowy element losowości i powoduje większe zróżnicowanie wśród obserwacji.

## 2.3 Zadanie 3

W tym zadaniu wyznaczono statystyki opisowe danych cenzurowanych I-go typu dotyczących leczenia dwoma różnymi lekami A i B. Grupa 40 pacjentów została podzielona losowo na dwie równoliczne podgrupy. Jednej z grup podano lek A, a drugiej lek B i przez rok obserwowano czas do remisji choroby.

W grupie otrzymującej lek A uzyskano następujące dane. U dziesięciu pacjentów remisja choroby nastąpiła w chwilach: 0.03345514, 0.08656403, 0.08799947, 0.24385821, 0.27755032, 0.40787247, 0.58825664, 0.64125620, 0.90679161, 0.94222208, natomiast u pozostałych dziesięciu pacjentów w ciągu roku nie zaobserwowano remisji.

W grupie otrzymującej lek B uzyskano następujące dane. U dziesięciu pacjentów remisja choroby nastąpiła w chwilach: 0.03788958, 0.12207257, 0.20319983, 0.24474299, 0.30492413,

0.34224462, 0.42950144, 0.44484582, 0.63805066, 0.69119721, natomiast u pozostałych dziesięciu pacjentów w ciągu roku nie zaobserwowano remisji.

Tabela 5: Podstawowe statystyki opisowe danych czasu remisji choroby po stosowaniu leku A lub leku B - dane cenzurowane I typu

Statystyka Opisowa	Lek A	Lek B
Rozmiar próby	20	20
Liczba obserwacji kompletnych	10	10
Liczba danych cenzurowanych	10	10
Minimum	0.0335	0.0379
Kwartyl dolny (Q1)	0.3753	0.3329
Mediana	0.9711	0.8456
Kwartyl górny (Q3)	1	1
Maksimum	1	1
Rozstęp międzykwartyłowy (IQR)	0.6247	0.6671
Rozstęp (max-min)	0.9665	0.9621

Z Tabeli 5. można odczytać, że grupy stosujące lek A oraz lek B nie różnią się znacząco między. Jednak jeżeli mamy już porównywać ich skuteczność, to lek B wypada lepiej. Zarówno mediana, jak i kwartyl dolny posiadają mniejsze wartości. Jest to równoznaczne z tym, że średni czas leczenia lekiem B jest krótszy niż lekiem A. Na korzyść leku A natomiast wypada (choć nieznacznie) minimum: pierwsza wyleczona osoba korzystała właśnie z niego. Natomiast liczba nieskutecznych prób (cenzurowanych) jest równa w obu przypadkach.

### 3 Lista 3

Lista 3 polega na wyznaczaniu estymatorów parametrów dla rozkładów, których obserwacje zostały częściowo cenzurowane, a także na wyznaczaniu przedziałów ufności dla tych estymatorów. Dodatkowo przeprowadzono porównanie różnych estymatorów na danych wygenerowanych.

**UWAGA:** Poniższe fragmenty kodu pozwalają na wyznaczenie estymatorów punktowych oraz przedziałów ufności dla próbek zmiennych losowych pochodzących z rozkładu wykładniczego postaci  $f(x) = \vartheta \exp(-\vartheta x) \mathbb{1}_{(0;\infty)}(x)$ , dla którego wartość oczekiwana wynosi  $\mathbb{E}[X] = \frac{1}{\vartheta}$ . Natomiast w Zadaniach 1. oraz 2. celem jest otrzymanie średniego czasu do remisji choroby, czyli  $\mathbb{E}[X] = \mu$ . Z tego wynika, że  $\vartheta = \frac{1}{\mu}$ . Z tego powodu wyniki zostały podane w postaci:

- $\hat{\mu} = \frac{1}{\hat{\vartheta}}$ , w przypadku estymacji punktowej,
- $\mu \in \left(\frac{1}{T_U}; \frac{1}{T_L}\right)$ , gdzie  $T_L$  oraz  $T_U$  - odpowiednio dolna i górna granica realizacji zbioru ufności dla parametru  $\vartheta$ .

#### 3.1 Zadanie 1

Zadanie polega na oszacowaniu średniego czasu do remisji choroby dla pacjentów leczonych lekami A i B na podstawie danych poddanych cenzurowaniu typu I przyjmując, że dane pochodzą z rozkładu wykładniczego  $\mathcal{E}(1)$ . Obejmuje ono:

- (a) wyznaczenie estymatorów największej wiarygodności średniego czasu do remisji dla obu grup,
- (b) wyznaczenie przedziałów ufności dla średniego czasu do remisji na poziomach ufności 95% ( $\alpha = 0.05$ ) i 99% ( $\alpha = 0.01$ ) dla obu grup.

##### 3.1.1 Estymacja punktowa

Z wykładu wiadomo że estymatorem największej wiarygodności jest estymator  $\hat{\vartheta} = \frac{R}{T_1}$ , gdzie:

$$R = \sum_{i=1}^n \mathbf{1}(X_i \leq t_0), \quad T_1 = \sum_{i=1}^R X_{(i)} + t_0(n - R)$$

Poniżej przedstawiono kod w języku Python służący do wyznaczania estymatora największej wiarygodności dla danych cenzurowanych I typu z rozkładu wykładniczego  $\mathcal{E}(\vartheta)$ . Dodatkowo dokonano estymacji parametrycznej parametru  $\vartheta$  dla danych z [Listy 2. Zadania 3.](#) za pomocą metody największej wiarygodności.

```
def MLE_cenzurowanie_I(times, deltas, t0):
    R = np.sum(deltas)
    n = len(times)
    T1 = np.sum(times[deltas==1]) + t0*(n - R)
    theta_hat = R / T1
    return theta_hat

mu_hat_A = 1/MLE_cenzurowanie_I(times_A, deltas_A, t0)
mu_hat_B = 1/MLE_cenzurowanie_I(times_B, deltas_B, t0)
```

Dla danych dotyczących osób przyjmujących lek A otrzymano estymator  $\hat{\mu}_A = 1.422$ , natomiast dla osób przyjmujących lek B - estymator  $\hat{\mu}_B = 1.346$ .

### 3.1.2 Estymacja przedziałowa

Do estymacji przedziałowej nie można stosować estymatora NW  $\hat{\vartheta} = \frac{R}{T_1}$  do bezpośredniego konstruowania realizacja zbiorów ufności, ponieważ jest on ilorazem zmiennej dyskretnej  $R$  i zmiennej ciągłej  $T_1$ . Taka mieszana natura utrudnia uzyskanie analitycznej funkcji centralnej o znanym rozkładzie w próbie skończonej, więc konstrukcja dokładnych realizacja zbiorów ufności jest trudna (wymaga przybliżeń asymptotycznych lub metod numerycznych).

Z tego względu używa się alternatywnego estymatora  $\tilde{\vartheta} = -\frac{\log(1-\frac{R}{n})}{t_0}$ , który zależy wyłącznie od  $R$  i korzysta z faktu, że  $R \sim \mathcal{B}(n, p)$  z  $p = 1 - \exp(-\vartheta t_0)$ .

Dla  $\tilde{\vartheta}$  realizację zbioru ufności buduje się dwuetapowo: najpierw wyznacza się przedział  $[p_L, p_U]$  dla  $p$  na poziomie ufności  $1 - \alpha$  (np. metodą Cloppera–Pearsona), a następnie transformuje jego końce przez odwrotność  $p = 1 - \exp(-\vartheta t_0)$ , otrzymując

$$\vartheta_L = -\frac{1}{t_0} \ln(1 - p_L), \quad \vartheta_U = -\frac{1}{t_0} \ln(1 - p_U).$$

W praktyce przedział Cloppera–Pearsona dla  $p$  definiuje się jako  $S_{\leq} \cap S_{\geq}$  (równoważnie  $[\inf S_{\geq}; \sup S_{\leq}]$ ), gdzie dla  $k = R$ :

$$S_{\leq} = p : \Pr(\text{Bin}(n, p) \leq k) > \frac{\alpha}{2}, \quad S_{\geq} = p : \Pr(\text{Bin}(n, p) \geq k) > \frac{\alpha}{2}.$$

Po uzyskaniu  $[p_L, p_U]$  powyższa transformacja daje przedział ufności  $[\vartheta_L, \vartheta_U]$  o poziomie ufności  $1 - \alpha$ .

Należy mieć na uwadze, że wynik będzie jedynie przybliżony - co wynika z błędów numerycznych.

Poniżej przedstawiono kod w Pythonie realizujący generowanie realizacji wspomnianego wcześniej zbioru ufności:

```

def binom_cdf(t, n, p):
    s = 0.0
    for i in range(0, t+1):
        s += comb(n, i) * (p**i) * ((1-p)**(n-i))
    return s

def clopper_pearson_binom(R, n, alpha):
    if R == 0:
        p_L = 0
        p_U = 1 - (alpha / 2) ** (1 / n)
        return p_L, p_U
    if R == n:
        p_U = 1.0
        p_L = (alpha / 2.0) ** (1.0 / n)
        return p_L, p_U

    target_low = 1 - alpha / 2
    L, H = 0, 1

    while H - L > 1e-12:
        M = (L + H) / 2
        cdf = binom_cdf(R, n, M)
        if cdf > target_low:
            L = M
        else:
            H = M
    p_L = (L + H) / 2

    target_up = alpha / 2
    L, H = 0, 1

    while H - L > 1e-12:
        M = (L + H) / 2
        cdf = binom_cdf(R, n, M)
        if cdf > target_up:
            L = M
        else:
            H = M
    p_U = (L + H) / 2

    return p_L, p_U

```

```

def p_to_theta(p, t0):
    return -np.log(1 - p) / t0

def Ci_theta_cenzurowanie_I(times, deltas, t0, alpha):
    R = np.sum(deltas)
    n = len(times)

    p_L, p_U = clopper_pearson_binom(R, n, alpha)

    theta_L = p_to_theta(p_L, t0)
    theta_U = p_to_theta(p_U, t0)

    return theta_L, theta_U

```

Poniżej przedstawiono wywołanie kodu dla danych z [Listy 2. Zadanie 3](#). Wyniki przedstawiono w tabeli poniżej.

```

alphas = [0.05, 0.01]
rows = []

for alpha in alphas:
    # Lek A
    thA_L, thA_U = Ci_theta_cenzurowanie_I(times_A, deltas_A, t0, alpha)
    rows.append(["Lek A", alpha, 1/thA_U, 1/thA_L])

    # Lek B
    thB_L, thB_U = Ci_theta_cenzurowanie_I(times_B, deltas_B, t0, alpha)
    rows.append(["Lek B", alpha, 1/thB_U, 1/thB_L])

```

Tabela 6: Przedziały ufności dla danych cenzurowanych I typu - różne poziomy ufności

Grupa	$\alpha$	$\hat{\mu}_L$	$\hat{\mu}_U$
Lek A	0.05	0.768	2.64
Lek B	0.05	0.768	2.64
Lek A	0.01	0.656	3.363
Lek B	0.01	0.656	3.363

W [Tabeli 6](#). widać, że otrzymane realizacje zbioru ufności są identyczne dla obu leków. Wynika to z faktu, że oba zbiory danych zawierają taką samą liczbę obserwacji kompletnych oraz ocenzurowanych, co przy tym samym czasie obserwacji prowadzi do identycznych wartości estymatora i granic przedziału ufności.

## 3.2 Zadanie 2

Zadanie polega na oszacowaniu średniego czasu do remisji choroby dla pacjentów leczonych lekami A i B na podstawie danych poddanych cenzurowaniu typu II, przyjmując, że pochodzą one z rozkładu wykładniczego. Obejmuje ono:

- (a) wyznaczenie estymatorów największej wiarygodności średniego czasu do remisji dla obu grup,
- (b) wyznaczenie przedziałów ufności dla średniego czasu do remisji na poziomach ufności 95% ( $\alpha = 0.05$ ) i 99% ( $\alpha = 0.01$ ) dla obu grup.

**UWAGA:** Formalnie ze względu na zmianę typu danych cenzurowanych z I-go na II-gi powinno się zastąpić dane cenzurowane w postaci  $(t_0, 0)$  na dane  $(X_{(m)}, 0)$ . Jednak estymacja zarówno punktowa, jak i przedziałowa nie korzysta z dokładnych wartości  $X_{(i)}$ ,  $i > m$ , zatem ten krok można pominąć.

### 3.2.1 Estymacja punktowa

Dla danych cenzurowanych II typu pochodzących z rozkładu wykładniczego  $\mathcal{E}(\vartheta)$  estymator największej wiarygodności przyjmuje postać:

$$\hat{\vartheta} = \frac{m}{T_2}, \quad \text{gdzie} \quad T_2 = \sum_{i=1}^m X_{(i)} + (n - m)X_{(m)}.$$

Poniżej przedstawiono kod w języku Python dla danych z [Listy 2. Zadania 3.](#), które tym razem przyjęto jako dane cenzurowane II typu z parametrem cenzurowania  $m = 10$ .

```
m = 10

def MLE_cenzurowanie_II(times, deltas, m):
    n = len(times)
    T2 = np.sum(times[deltas==1]) + (n - m)*np.max(times[deltas==1])
    theta_hat = m / T2
    return theta_hat

mu_hat_A = 1/MLE_cenzurowanie_II(times_A, deltas_A, m)
mu_hat_B = 1/MLE_cenzurowanie_II(times_B, deltas_B, m)
```

Dla danych dotyczących osób przyjmujących lek A otrzymano estymator  $\hat{\mu}_A = 1.364$ , natomiast dla osób przyjmujących lek B - estymator  $\hat{\mu}_B = 1.037$ .

### 3.2.2 Estymacja przedziałowa

Do wyznaczenia przedziałów ufności dla danych cenzurowanych II typu pochodzących z rozkładu wykładniczego  $\mathcal{E}(\vartheta)$  potrzebna będzie funkcja kwantylowa rozkładu gamma  $\Gamma(k, \vartheta)$ . Wynika to stąd, że dla statystyki

$$T_2 = \sum_{i=1}^m X_{(i)} + (n - m)X_{(m)}$$

zachodzi zależność

$$\frac{T_2}{m\vartheta} \sim \text{Gamma}(m, \frac{1}{m})$$

Dystrybuanta rozkładu gamma wyraża się przez niepełną funkcję gamma:

$$F(x; k, \sigma) = \frac{\gamma(k, x/\sigma)}{\Gamma(k)},$$

gdzie

$$\begin{aligned} \gamma(z, x) &= \int_0^x t^{z-1} e^{-t} dt && \text{(niepełna funkcja gamma),} \\ \Gamma(z) &= \int_0^\infty t^{z-1} e^{-t} dt && \text{(funkcja gamma).} \end{aligned}$$

Ponieważ dystrybuanta rozkładu gamma nie ma prostego wzoru elementarnego, zkorzystano z jej funkcji kwantylowej (numerycznie dostępnej w pakietach statystycznych).

Niech  $q_{\alpha/2}$  i  $q_{1-\alpha/2}$  będą kwantylami rozkładu  $\text{Gamma}(m, 1/m)$ . Z faktu, że

$$\Pr\left(q_{\alpha/2} \leq \frac{T_2}{\vartheta} \leq q_{1-\alpha/2}\right) = 1 - \alpha$$

otrzymano realizację zbioru ufności dla  $\vartheta$ :

$$[\vartheta_L, \vartheta_U] = \left[ \frac{m \cdot T_2}{q_{1-\alpha/2}}, \frac{m \cdot T_2}{q_{\alpha/2}} \right].$$

W praktyce kwantyle  $q_{\alpha/2}$  i  $q_{1-\alpha/2}$  oblicza się numerycznie, np. funkcją `scipy.stats.gamma.ppf` w Pythonie z parametrami `a=m` i `scale=1/m`.

Poniżej przedstawiono kod w języku Python realizujący wyżej wspomnianą estymację przedziałową.



```
def Ci_theta_cenzurowanie_II(times, deltas, m, alpha):
    n = len(times)
    T2 = np.sum(times[deltas==1]) + (n - m)*np.max(times[deltas==1])

    q_lower = gamma.ppf(alpha/2, a=m, scale=1/m)
    q_upper = gamma.ppf(1 - alpha/2, a=m, scale=1/m)

    theta_L = m * q_lower / T2
    theta_U = m * q_upper / T2

    return theta_L, theta_U
```

Poniżej przedstawiono wywołanie kodu dla danych z [Listy 2. Zadania 3](#). Wyniki przedstawiono w tabeli poniżej.

```
alphas = [0.05, 0.01]
rows = []

for alpha in alphas:
    # Lek A
    thA_L, thA_U = Ci_theta_cenzurowanie_II(times_A, deltas_A, m, alpha)
    rows.append(["Lek A", alpha, 1/thA_U, 1/thA_L])

    # Lek B
    thB_L, thB_U = Ci_theta_cenzurowanie_II(times_B, deltas_B, m, alpha)
    rows.append(["Lek B", alpha, 1/thB_U, 1/thB_L])
```

Tabela 7: Realizacje zbiorów ufności dla danych cenzurowanych II typu - różne poziomy ufności

Grupa	$\alpha$	$\hat{\mu}_L$	$\hat{\mu}_U$
Lek A	0.05	0.798	2.844
Lek B	0.05	0.607	2.163
Lek A	0.01	0.682	3.669
Lek B	0.01	0.519	2.79

Z [Tabeli 7](#). można odczytać, że realizacje zbioru ufności są węższe dla danych dotyczących leku A.

### 3.3 Zadanie 3

Zadanie polega na symulacyjnym porównaniu dokładności dwóch estymatorów punktowych  $\vartheta$ , zdefiniowanych wzorami:

$$\hat{\vartheta} = \frac{R}{T_1}, \quad \tilde{\vartheta} = -\frac{\log\left(1 - \frac{R}{n}\right)}{t_0}$$

gdzie

$$R = \sum_{i=1}^n \mathbf{1}_{\{X_i \leq t_0\}}, \quad T_1 = \sum_{i=1}^R X_{(i)} + t_0(n - R).$$

Porównanie estymatorów zostanie dokonane na podstawie:

- obciążenia (bias):

$$\text{Bias}(\hat{\vartheta}, \vartheta) = E_{\vartheta}(\hat{\vartheta} - \vartheta), \quad \text{Bias}(\tilde{\vartheta}, \vartheta) = E_{\vartheta}(\tilde{\vartheta} - \vartheta),$$

- średniego błędu kwadratowego (MSE):

$$\text{MSE}(\hat{\vartheta}, \vartheta) = E_{\vartheta}(\hat{\vartheta} - \vartheta)^2, \quad \text{MSE}(\tilde{\vartheta}, \vartheta) = E_{\vartheta}(\tilde{\vartheta} - \vartheta)^2,$$

dla wartości  $\vartheta = 1$ , rozmiarów próby  $n = 10, 30$  oraz parametrów cenzurowania  $t_0 = 0.5, 1, 2$ .

Celem jest ocena, który z estymatorów daje mniejsze obciążenie i mniejszy błąd średniokwadratowy w różnych scenariuszach cenzurowania.

Symulacja będzie polegała na generowaniu prób 100 000 razy. W przypadku, gdy  $R = n$  (czyli wszystkie dane są niecenzurowane), estymatory przyjmują wartości skrajne  $\hat{\vartheta} = 0$ ,  $\tilde{\vartheta} = -\infty$ , co czyni je niepraktycznymi w analizie. W takiej sytuacji iteracja zostanie powtórzona. Wartości  $\hat{\vartheta}$  oraz  $\tilde{\vartheta}$  zawsze będą wyznaczone na podstawie tych samych prób. Takie podejście ogranicza wpływ sytuacji skrajnych i zapewnia poprawność porównania estymatorów.

Poniżej przedstawiono kod w Pythonie dokonujący wyżej wspomnianej symulacji. Wyniki przedstawiono w tabeli poniżej.

```
theta_true = 1
n_values = [10, 30]
t0_values = [0.5, 1, 2]
n_sim = 100000 # liczba replikacji

rows = []
```

```

for n in n_values:
    for t0 in t0_values:
        theta_hat_vals = []
        theta_wave_vals = []
        i = 0
        while i < n_sim:
            # generowanie próbki i cenzurowanie I typu
            sample, deltas = GE_cenzurowanie_I_typu(t0, 1/theta_true, 1, n)
            sample = np.array(sample)
            deltas = np.array(deltas)

            R = int(np.sum(deltas))

            if R == n:
                continue

            T1 = np.sum(sample[deltas==1]) + t0*(n - R)

            theta_hat = R / T1
            theta_wave = -np.log(1 - R/n) / t0

            theta_hat_vals.append(theta_hat)
            theta_wave_vals.append(theta_wave)
            i += 1

        bias_hat = np.mean(np.array(theta_hat_vals) - theta_true)
        bias_wave = np.mean(np.array(theta_wave_vals) - theta_true)
        mse_hat = np.mean((np.array(theta_hat_vals) - theta_true)**2)
        mse_wave = np.mean((np.array(theta_wave_vals) - theta_true)**2)

        rows.append({
            "Liczność próby": n,
            "Czas obserwacji": t0,
            r"$\mathrm{Bias}(\hat{\vartheta})$": bias_hat,
            r"$\mathrm{Bias}(\tilde{\vartheta})$": bias_wave,
            r"$\mathrm{MSE}(\hat{\vartheta})$": mse_hat,
            r"$\mathrm{MSE}(\tilde{\vartheta})$": mse_wave
        })

```

Tabela 8: Porównanie estymatorów  $\hat{\vartheta}$  oraz  $\tilde{\vartheta}$ 

Liczność próby	Czas obserwacji	Bias( $\hat{\vartheta}$ )	Bias( $\tilde{\vartheta}$ )	MSE( $\hat{\vartheta}$ )	MSE( $\tilde{\vartheta}$ )
10	0.5	0.0622	0.0732	0.3036	0.3312
10	1.0	0.0601	0.0848	0.1845	0.2225
10	2.0	-0.0504	-0.0759	0.0791	0.0607
30	0.5	0.0197	0.0222	0.0896	0.0927
30	1.0	0.0212	0.0293	0.0568	0.0660
30	2.0	0.0219	0.0511	0.0405	0.0680

Z Tabeli 8. wynika, że estymator największej wiarygodności  $\hat{\vartheta}$  w każdym przypadku charakteryzuje się mniejszym obciążeniem (Bias) niż estymator  $\tilde{\vartheta}$ , oparty wyłącznie na liczbie obserwacji kompletnych  $R$ . W większości przypadków  $\hat{\vartheta}$  ma również niższą wartość błędu średniokwadratowego (MSE).

Warto zauważyć, że wraz ze wzrostem liczebności próby różnice w obciążeniu i MSE między estymatorami stają się coraz mniejsze. Może to być spowodowane tym, że estymator  $\tilde{p} = \frac{R}{n}$ , na którym opiera się estymator  $\tilde{\vartheta}$ , ma mniejsze wartości dla większej liczności próby. To z kolei powoduje lepsze przybliżenie parametru  $\vartheta$ , a tym samym zmniejszenie różnic z  $\hat{\vartheta}$ .

### 3.4 Zadanie dodatkowe 1

W tym zadaniu udowodniono alternatywny wzór na statystykę  $T_2$  w postaci  $T_2 = \sum_{i=1}^m D_i$ , gdzie  $D_i = (n - i + 1) (X_{(i)} - X_{(i-1)})$ .

$$\begin{aligned}
T_2 &= \sum_{i=1}^m D_i = \sum_{i=1}^m (n - i + 1) (X_{(i)} - X_{(i-1)}) = \\
&\quad \sum_{i=1}^m (n - i + 1) X_{(i)} - \sum_{i=1}^m (n - i + 1) X_{(i-1)} = \\
&\quad \sum_{i=1}^m (n - i + 1) X_{(i)} - \sum_{i=2}^m (n - i + 1) X_{(i-1)} \stackrel{j=i-1}{=} \\
&\quad (n - m + 1) X_{(m)} + \sum_{i=1}^{m-1} (n - i + 1) X_{(i)} - \sum_{j=1}^{m-1} (n - j) X_{(j)} \stackrel{i=j}{=} \\
&\quad (n - m + 1) X_{(m)} + \sum_{i=1}^{m-1} [(n - i + 1) - (n - i)] X_{(i)} = \\
&\quad (n - m + 1) X_{(m)} + \sum_{i=1}^{m-1} X_{(i)} = \sum_{i=1}^m X_{(i)} + (n - m) X_{(m)}
\end{aligned}$$

Dodatkowo wykazane zostało, że jeżeli zmienne  $X_i$  pochodzą z rozkładu  $\mathcal{E}(\vartheta)$  i są niezależne, to  $D_i$  również są niezależne o tym samym rozkładem.

**Fakt:** Rozkład statystyk pozycyjnych ma postać:

$$f_{X_{(1)}, \dots, X_{(m)}}(x) = \frac{n!}{(n-m)!} \prod_{i=1}^m f(x_{(i)}) = \frac{n!}{(n-m)!} \vartheta^m \exp \left( -\vartheta \sum_{i=1}^m (n-i+1) x_{(i)} \right)$$

Niech  $Y_i = X_{(i)} - X_{(i-1)}$ , wtedy:

$$\varphi \left( (X_{(1)}, \dots, X_{(m)}) \right) = (X_{(1)}, X_{(2)} - X_{(1)}, \dots, X_{(m)} - X_{(m-1)}) = (Y_1, \dots, Y_m)$$

$$\varphi^{-1}((Y_1, \dots, Y_m)) = \left( Y_1, Y_1 + Y_2, \dots, \sum_{i=1}^m Y_i \right)$$

$$J_{\varphi^{-1}} = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ 1 & 1 & 0 & \dots & 0 \\ 1 & 1 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & \dots & 1 \end{pmatrix}, \quad |\det J_{\varphi^{-1}}| = 1$$

$$\begin{aligned} f_{Y_1, \dots, Y_m}(y) &= \frac{n!}{(n-m)!} \vartheta^m \exp \left( -\vartheta \sum_{i=1}^m (n-i+1) \sum_{j=1}^i y_j \right) |\det J_{\varphi^{-1}}| = \\ &= \frac{n!}{(n-m)!} \vartheta^m \exp \left( -\vartheta \sum_{j=1}^m \sum_{i=j}^m (n-i+1) y_j \right) \cdot 1 = \\ &= \prod_{i=1}^m (n-i+1) \vartheta \exp \left( -(n-i+1) \vartheta y_i \right) \end{aligned}$$

Rozkład łączny jest iloczynem statystyk brzegowych  $Y_i$  o odpowiednich rozkładach wykładniczych, zatem są one od siebie niezależne.

$$D_i = (n-i+1)Y_i, \quad Y \sim \mathcal{E}((n-i+1)\vartheta), \quad D_i \sim \mathcal{E}(\vartheta)$$

Zatem  $D_i \sim \mathcal{E}(\vartheta)$ . są niezależne i o tym samym rozkładzie.

### 3.5 Zadanie dodatkowe 2

Zadanie polega na wyznaczeniu estymatora NW oraz realizacji zbiorów ufnosci w oparciu o dane cenzurowane I-go typu pochodzące z rozkładu Reyleigha  $\mathcal{Ra}(\sigma)$ :

$$f_{\sigma}(x) = \frac{x}{\sigma^2} \exp \left( -\frac{x^2}{2\sigma^2} \right) \mathbf{1}_{(0;\infty)}(x), \quad F_{\sigma}(t) = 1 - \exp \left( -\frac{t^2}{2\sigma^2} \right).$$

### 3.5.1 Estymacja punktowa

Estymacji punktowej dokonano metodą wiarygodności.

Funkcja wiarygodności oparta o rozkład  $\mathcal{R}a(\sigma)$  ma postać:

$$L(\sigma; t^*) = \frac{n!}{(n-r)!} \frac{\prod_{i=1}^r x_{(i)}}{\sigma^{2r}} \exp \left( -\frac{1}{2\sigma^2} \left( \sum_{i=1}^r x_{(i)}^2 + (n-r)t_0^2 \right) \right)$$

Do szukania maksimum skorzystano z funkcji logarytmicznej (log-wiarygodność), gdyż logarytm zachowuje ekstrema oraz monotoniczność:

$$\ell(\sigma) = \ln \frac{n!}{(n-r)!} + \ln \prod_{i=1}^r x_{(i)} - 2r \ln \sigma - \frac{1}{2\sigma^2} T_3, \quad T_3 = \sum_{i=1}^r x_{(i)}^2 + (n-r)t_0^2$$

Policzono pochodną funkcji  $\ell$  po zmiennej  $\sigma$ :

$$\ell'(\sigma) = \frac{-2r}{\sigma} + \frac{T_3}{\sigma^3} \Rightarrow \hat{\sigma} = \sqrt{\frac{T_3}{2r}}$$

Sprawdono, czy w tym punkcie funkcja jest wklęsła w otoczeniu  $\hat{\sigma}$ :

$$\ell''(\sigma) = \frac{2r}{\sigma^2} - \frac{3T_3}{\sigma^4}, \quad \ell''(\hat{\sigma}) < 0$$

Zatem funkcja  $\ell$  w punkcie  $\hat{\sigma}$  jest maksimum.

Dodatkowo, funkcja osiąga tylko jedno ekstremum w postaci maksimum. Oznacza to, że funkcja rośnie na przedziale  $(0; \hat{\sigma}]$ , oraz maleje na przedziale  $[\hat{\sigma}; \infty)$ . Oznacza to, że w tym punkcie osiąga maksimum globalne.

**Wniosek:**  $L$  jest unimodalna, z maksimum globalnym w  $\hat{\sigma}$ .

### 3.5.2 Estymacja przedziałowa

Estymator NW rozkładu Rayleigha oparty na danych cenzurowanych I-go typu zależy od wartości zmiennych losowych o rozkładach odpowiednio absolutnie ciągłym ( $T_3$ ) oraz dyskretnym ( $R \sim \mathcal{B}(n, p = F_\sigma(t_0))$ ,  $F_\sigma(t_0) = 1 - \exp\left(-\frac{t_0^2}{2\sigma^2}\right)$ ). Z tego powodu, podobnie jak w przypadku rozkładu wykładniczego, wyznaczono przedział ufności dla parametru  $p$  i skorzystano z faktu, że jest on równy dystrybuancie rozkładu  $\mathcal{R}a(\sigma)$  w punkcie  $t_0$ .

Tym samym, mając realizację zbioru ufności  $[p_L; p_U]$  dla parametru  $p$ , po odpowiednich przekształceniach otrzymano realizację zbioru ufności  $[\sigma_L; \sigma_U]$  parametru  $\sigma$ , gdzie:

$$\sigma_L = \frac{t_0}{\sqrt{-2 \ln(1 - p_U)}}, \quad \sigma_U = \frac{t_0}{\sqrt{-2 \ln(1 - p_L)}}.$$

## 4 Lista 4

Lista 4 polega na testowaniu hipotez dotyczących średniego czasu życia na podstawie danych cenzurowanych I typu z rozkładu wykładniczego, poprzez wyznaczanie poziomu krytycznego testu ilorazu wiarygodności oraz ocenę jego mocy i rozmiaru.

### 4.1 Zadanie 1

Zadanie dotyczy deklaracji funkcji do wyznaczania poziomu krytycznego (*eng. p-value*) w teście ilorazu wiarygodności do testowania hipotez lewo-, prawo- i dwustronnych dla danych cenzurowanych I-go typu pochodzących z rozkładu wykładniczego  $\mathcal{E}(\vartheta)$ .

Zaczęto od udowodnienia że funkcja wiarygodności jest funkcją unimodalną (ma jedno maksimum globalne).

Funkcja wiarygodności dla rozkładu wykładniczego  $\mathcal{E}(\vartheta)$  ma postać:

$$L(\vartheta; t^*) = \frac{n!}{(n-r)!} \vartheta^r \exp \left( -\vartheta \left[ \sum_{i=1}^r x_{(i)} + t_0(n-r) \right] \right).$$

Najwygodniej maksimum będzie się szukało przez funkcję logarytmiczną (log-wiarygodność), ponieważ logarytm jest funkcją rosnącą i zachowuje maksimum:

$$\ell(\vartheta) = \ln \frac{n!}{(n-r)!} + r \ln \vartheta - \vartheta S, \quad S = \sum_{i=1}^r x_{(i)} + t_0(n-r).$$

Aby znaleźć maksimum, obliczono pochodną funkcji wiarygodności (log-wiarygodności) po zmiennej  $\vartheta$ .

$$\ell'(\vartheta) = \frac{r}{\vartheta} - S = 0 \quad \Rightarrow \quad \hat{\vartheta} = \frac{r}{S}.$$

Z warunku stacjonarności wynika że ekstremum znajduje się w  $\hat{\vartheta} = \frac{r}{S}$ , który jest naszym estymatorem największej wiarygodności.

Sprawdzono wklęsłość funkcji (co gwarantuje maksimum):

$$\ell''(\vartheta) = -\frac{r}{\vartheta^2} < 0 \quad \forall \vartheta > 0.$$

**Wniosek:**  $\ell$  jest ściśle wklęsła, więc  $L$  jest unimodalna z maksimum globalnym w  $\hat{\vartheta}$ .

Funkcja ilorazu wiarygodności ma postać:

$$\lambda(t^*) = \frac{\sup_{\vartheta \in \Theta_0} L(\vartheta; t^*)}{\sup_{\vartheta \in \Theta} L(\vartheta; t^*)}.$$



Przy testowaniu hipotezy dwustronnej zbiór  $\Theta_0 = \{\vartheta_0\}$  jest jednoelementowy i właśnie w tym punkcie osiąga swoje supremum.

W przypadku hipotezy lewostronnej, tj. testowania na przedziale  $[\vartheta_0, \infty)$ , supremum przyjmuje się w punkcie  $\hat{\vartheta}$ , jeżeli  $\hat{\vartheta} \in [\vartheta_0, \infty]$ , lub w punkcie  $\vartheta_0$  w przeciwnym przypadku.

Analogicznie, dla hipotezy prawostronnej, gdy rozważany przedział to  $(-\infty, \vartheta_0]$ , supremum osiągane jest w  $\hat{\vartheta}$ , jeśli  $\hat{\vartheta} \in (-\infty, \vartheta_0]$ , lub w  $\vartheta_0$  w przeciwnym wypadku.

Zgodnie z twierdzeniem Wilksa, wartość krytyczna ma postać:

$$1 - F_{\chi^2(1)}(-2 \ln \lambda(r, s)),$$

gdzie  $F_{\chi^2(1)}$  oznacza dystrybuantę rozkładu  $\chi^2$  z jednym stopniem swobody.

Poniżej przedstawiono kod deklarujący funkcję do testowania wyżej wspomnianych hipotez danych cenzurowanych z rozkładu wykładniczego  $\mathcal{E}(\vartheta)$ .

```
def IW_cenzurowanie_I(r, s, n, t0, theta0, test_type, alpha):
    if r == 0:
        theta_hat = 1e-9
    else:
        S = s + (n - r) * t0 # Całkowity czas testowania
        theta_hat = r / S # Estymator MLM

    Lambda = (theta0 / theta_hat)**r * np.exp(S * (theta_hat - theta0))

    chi2_statistic = -2 * np.log(Lambda)
    lambda_1 = chi2.ppf(1 - alpha, df=1)

    if test_type == "dwustronna":
        # H0: theta = theta0 vs H1: theta != theta0
        p_value = 1 - chi2.cdf(chi2_statistic, df=1)
    elif test_type == "prawostronna":
        # H0: theta <= theta0 vs H1: theta > theta0
        p_value = ((1 - chi2.cdf(chi2_statistic, df=1))
                    if (theta_hat > theta0) else 1)
    else: # "lewostronna"
        # H0: theta >= theta0 vs H1: theta < theta0
        p_value = ((1 - chi2.cdf(chi2_statistic, df=1))
                    if (theta_hat < theta0) else 1)

    return chi2_statistic, lambda_1, p_value
```

## 4.2 Zadanie 2

Zadanie polega na przeprowadzeniu symulacji, których celem jest oszacowanie mocy (dla 10 wybranych alternatyw) oraz rozmiaru testu z punktu dwustronnego dla wybranej wartości  $\vartheta_0$ ,  $t_0$  oraz  $n \in \{20, 50\}$ .

Moc i rozmiar testu statystycznego można wyznaczyć symulacyjnie metodą Monte Carlo, przy czym poprawność takiego podejścia uzasadnia twierdzenie Glivenki–Cantellego.

Próba została wygenerowana z rozkładu wykładniczego  $\mathcal{E}(2)$ . Dokonano cenzurowania I-typu z parametrem cenzurowania  $t_0 = 1.5$ .

W następnej kolejności zbadano hipotezę zerową  $H_0 : \vartheta = \vartheta_0 = 2$ , przy hipotezie alternatywnej  $H_1 : \vartheta \neq \vartheta_0$  dla 10 wybranych parametrów  $\vartheta_1 \in \{1.1, 1.3, 1.5, 1.7, 1.9, 2.0, 2.1, 2.3, 2.5, 2.7\}$ . Dla każdego przypadku wykonano 10 000 replikacji (które było powtarzane w przypadku  $R = 0$ ). Wyniki przedstawiono w tabeli poniżej.

Poniżej znajduje się kod w języku Python dokonujący symulacji.

```
theta0 = 2
t0 = 1.5
n_vals = [20, 50]
alternatives = [1.1, 1.3, 1.5, 1.7, 1.9, 2, 2.1, 2.3, 2.5, 2.7]
alpha = 0.05
n_sim = 10000 # liczba replikacji

rows = []
rozmiar = []

for n in n_vals:
    for theta in alternatives:
        results = []

        n_cur = 0
        while n_cur != n_sim:
            sample, deltas = GE_cenzurowanie_I_typu(t0, 1/theta0, 1, n)
            sample = np.array(sample)
            deltas = np.array(deltas)

            r = np.sum(deltas)

            if (r != 0):
                n_cur += 1
```

```

else:
    continue

s = np.sum(sample[deltas==1])

_, _, p_value = IW_cenzurowanie_I(r, s, n, t0, theta,
    "dwustronna", alpha)
results.append(int(p_value <= alpha))

moc = np.mean(results)
if theta == theta0:
    rozmiar.append(moc)

rows.append({
    r"\vartheta_1": theta,
    "Liczność próby": n,
    "Moc testu": moc
})

```

Tabela 9: Wyniki symulacji mocy testu dwustronnego

$\vartheta$	Liczność próby	Moc testu	Liczność próby	Moc testu
1.1	20	0.7027	50	0.9820
1.3	20	0.4373	50	0.8272
1.5	20	0.2306	50	0.4938
1.7	20	0.1080	50	0.1954
1.9	20	0.0577	50	0.0616
2.0	20	0.0519	50	0.0548
2.1	20	0.0583	50	0.0635
2.3	20	0.0981	50	0.1667
2.5	20	0.1799	50	0.3460
2.7	20	0.2824	50	0.5529

Z Tabeli 9 można wywnioskować, że moc testu jest poprawnie estymowana w tym sensie, że im dalej  $\vartheta_1$  znajduje się od  $\vartheta_0$ , tym większa jest moc testu. Ponadto, wraz ze wzrostem liczności próby, moc testu również rośnie (z wyjątkiem przypadku  $\vartheta_1 = 2$ , co wynika z faktu, że  $\vartheta_0 = 2$ , czyli gdy hipoteza zerowa jest prawdziwa - moc testu powinna być wtedy niska). Rozmiar testu w przypadku  $\vartheta_1 = \vartheta_0 = 2$  wyniósł 0.052 dla liczności próby  $n = 20$  oraz 0.055 dla liczności próby  $n = 50$ .

### 4.3 Zadanie 3

Zadanie polega na weryfikacji hipotezy, że średni czas do remisji choroby w grupie, która brała lek A, oraz w grupie, która brała lek B (na podstawie danych z [Listy 2. Zadania 3.](#)), można traktować jako realizacje zmiennych losowych z rozkładu wykładniczego  $\mathcal{E}(1)$ . Założono również, że dokonano cenzurowania typu I z parametrem cenzurowania  $t_0 = 1$ . Test przeprowadzono na poziomie istotności  $\alpha = 0.05$ .

Poniżej przedstawiono kod dokonujący testowania tej hipotezy statystycznej. Wyniki przedstawiono w tabelce poniżej.

```
theta0 = 1.0
t0 = 1.0
alpha = 0.05

rows = []

for name, (times, deltas) in datasets.items():
    n = len(times)
    r = int(np.sum(deltas))
    s = float(np.sum(times[deltas==1]))
    chi2_statistic, lambda_1, p_value = IW_cenzurowanie_I(r, s, n, t0, theta0,
        "dwustronna", alpha)
    rows.append({
        "Grupa": name,
        "Liczność próby": n,
        "Liczność danych kompletnych": r,
        "p -wartość": p_value,
    })
```

Tabela 10: Testowanie hipotezy dwustronnej na danych rzeczywistych

Grupa	Liczność próby	Liczność danych kompletnych	p -wartość
Lek A	20	10	0.2374
Lek B	20	10	0.3230

Z [Tabeli 10.](#) odczytano, że wartości p są większe od przyjętego poziomu istotności, zatem nie ma podstaw do odrzucenia hipotezy zerowej  $H_0$ .

#### 4.4 Zadanie dodatkowe

W tym zadaniu zbadano krytyczny poziom testu ilorazu wiarygodności dla hipotezy  $H_0 : \vartheta_A = \vartheta_B$ , przeciwko hipotezie alternatywnej  $H_1 : \vartheta_A \neq \vartheta_B$ . Skorzystano z faktu, że A i B są prowadzone niezależnie, dzięki również niezależność funkcji wiarygodności:

$$L((\vartheta_A, \vartheta_B); (t_A^*, t_B^*)) = L(\vartheta_A; t_A^*) \cdot L(\vartheta_B; t_B^*).$$

Dla hipotezy alternatywnej zbadano przestrzeń:  $\Theta_1 = \{(\vartheta_A, \vartheta_B) : \vartheta_A > 0, \vartheta_B > 0\} = (0; \infty)^2$ , natomiast dla hipotezy zerowej:  $\Theta_0 = \{(\vartheta_A, \vartheta_B) : \vartheta_A = \vartheta_B > 0\} = (0; \infty)$ .

Zarówno w liczniku, jak i w mianowniku zbadano supremum iloczynu dwóch funkcji wiarygodności dla A i B. Z faktu, że funkcja wiarygodności jest jednomodalna wnioskujemy, że:

$$\sup_{(\vartheta_A, \vartheta_B) \in \Theta_1} [L(\vartheta_A; t_A^*) \cdot L(\vartheta_B; t_B^*)] = \sup_{\vartheta_A \in (0; \infty)} L(\vartheta_A; t_A^*) \cdot \sup_{\vartheta_B \in (0; \infty)} L(\vartheta_B; t_B^*) = L(\hat{\vartheta}_A; t_A^*) \cdot L(\hat{\vartheta}_B; t_B^*),$$

W iloczynie natomiast otrzymano iloczyn postaci:

$$L(\vartheta; (t_A^*, t_B^*)) = \frac{n!}{(n-r)!} \vartheta^r \exp\left(-\vartheta \left[\sum_{i=1}^r a_{(i)} + t_0(n-r)\right]\right) \cdot \frac{n!}{(n-r)!} \vartheta^r \exp\left(-\vartheta \left[\sum_{i=1}^r b_{(i)} + t_0(n-r)\right]\right),$$

gdzie  $a_{(i)}, b_{(i)}$  - realizacje statystyk pozycyjnych odpowiednio wektorów A i B. Łatwo zauważyć, że po krótkich przekształceniach test jest postaci:

$$L(\vartheta; (t_A^*, t_B^*)) = \left(\frac{n!}{(n-r)!}\right)^2 \vartheta^{2r} \exp\left(-\vartheta [T_{1_A} + T_{1_B}]\right), \text{ gdzie:}$$

$$T_{1_A} = \sum_{i=1}^r a_{(i)} + (n-r)t_0, \quad T_{1_B} = \sum_{i=1}^r b_{(i)} + (n-r)t_0$$

Z tego natomiast łatwo zauważyć, że otrzymano podobną jak na [Liście 4. Zadaniu 1.](#) funkcję największej wiarygodności, gdzie  $T_1 = T_{1_A} + T_{1_B}$ ,  $R = R_A + R_B$  (w tym przypadku  $R_A = R_B = r$ ). Tym samym otrzymano wzór na wspólny estymator największej wiarygodności  $\hat{\vartheta} = \frac{R_A + R_B}{T_{1_A} + T_{1_B}}$ .

Z wartości supremów licznika oraz mianownika, można wyznaczyć postać funkcji  $\lambda((t_A^*, t_B^*))$ :

$$\lambda((t_A^*, t_B^*)) = \left( \frac{\vartheta^2}{\vartheta_A \vartheta_B} \right) \exp(T_A(\vartheta_A - \vartheta) + T_B(\vartheta_B - \vartheta)).$$

W ten sposób uzyskano wartość *p-value* testu badającego równość parametrów realizacji A i B, wynoszącą 0.292, zatem nie ma podstaw do odrzucenia tej hipotezy zerowej.