Politechnika Wrocławska

Wydział Informatyki i Telekomunikacji

**Modele systemów dynamicznych**

Sprawozdanie z laboratorium 4

**Igor Lis**

Nr albumu: **284053**

Kierunek: **Inżynieria systemów**

WROCŁAW 2025

**MODEL WZROSTU LOGISTYCZNEGO**

# Wstęp teoretyczny

Celem niniejszego sprawozdania jest implementacja i analiza modelu wzrostu logistycznego populacji. Model ten stanowi klasyczne zagadnienie w ekologii matematycznej i opisuje, jak wielkość populacji zmienia się w czasie w środowisku o ograniczonych zasobach. W ramach pracy porównane zostanie rozwiązanie numeryczne, uzyskane za pomocą biblioteki scipy, z dokładnym rozwiązaniem analitycznym, wyprowadzonym przy użyciu pakietu sympy.

Model wzrostu logistycznego jest rozwinięciem prostszego modelu wykładniczego. Jego kluczowym założeniem jest istnienie tzw. pojemności środowiska (K), która reprezentuje maksymalną liczbę osobników, jaką dane środowisko jest w stanie utrzymać. W przeciwieństwie do nieograniczonego wzrostu wykładniczego, w modelu logistycznym tempo wzrostu populacji zwalnia w miarę zbliżania się do tej granicy, co prowadzi do charakterystycznej krzywej w kształcie litery "S".

Podstawowe wzory i definicje:

Podstawą modelu jest nieliniowe równanie różniczkowe zwyczajne pierwszego rzędu:

gdzie:

N(t) – wielkość populacji w chwili czasu (t),

r – współczynnik wzrostu wewnętrznego, określający potencjalne tempo reprodukcji,

K – pojemność środowiska, czyli maksymalna wielkość populacji,

– chwilowa szybkość zmiany populacji.

Rozwiązaniem analitycznym powyższego równania, przy zadanym warunku początkowym N(0)=N0:

​

gdzie:

​ – początkowa wielkość populacji w czasie t = 0.

W celu ilościowej oceny dokładności rozwiązania numerycznego (​) względem rozwiązania analitycznego (​), wykorzystano dwie standardowe metryki błędu: średni błąd bezwzględny (MAE) oraz średni błąd kwadratowy (MSE). Wzory na te błędy przedstawiono poniżej:

gdzie:

n – liczba punktów pomiarowych w czasie,

– wartość populacji uzyskana z symulacji numerycznej w i-tym kroku,

– wartość populacji obliczona ze wzoru analitycznego w i-tym kroku.

# Opis rozwiązania

import numpy as np

import matplotlib.pyplot as plt

from scipy.integrate import solve\_ivp

import sympy as sp

import os

import pandas as pd

import time # Dodajemy do pomiaru czasu

# --- Krok 1: Utworzenie folderu na wyniki ---

output\_dir = "lista\_4/wyniki"

# --- Krok 2: Funkcja do symulacji i rysowania ---

def run\_and\_plot\_simulation(params, t\_span=(0, 25), num\_points=1000):

"""

Uruchamia symulację dla danych parametrów, rysuje wykresy i zapisuje je do plików.

"""

# Rozpakowanie parametrów

r, K, N0, label = params['r'], params['K'], params['N0'], params['label']

filename\_safe\_label = label.lower().replace(" ", "\_").replace(":", "")

# Definicja równania dla SciPy

def logistic\_growth\_rhs(t, N, r\_val, K\_val):

return r\_val \* N \* (1 - N / K\_val)

t\_eval = np.linspace(\*t\_span, num\_points)

# --- Rozwiązanie analityczne (SymPy) ---

t\_sym, r\_sym, K\_sym, N0\_sym = sp.symbols('t r K N0')

N\_sym = sp.Function('N')

ode = sp.Eq(N\_sym(t\_sym).diff(t\_sym), r\_sym \* N\_sym(t\_sym) \* (1 - N\_sym(t\_sym) / K\_sym))

exact\_solution\_expr = sp.dsolve(ode, ics={N\_sym(0): N0\_sym}).rhs

# Podstawienie wartości parametrów (r, K, N0) do wyrażenia

expr\_with\_params = exact\_solution\_expr.subs({r\_sym: r, K\_sym: K, N0\_sym: N0})

start\_time = time.time() # Pomiar czasu

N\_exact\_list = []

for t\_val in t\_eval:

# Podstawiamy konkretną wartość czasu 't\_val' do symbolu 't\_sym'

# i konwertujemy wynik (który jest obiektem SymPy) na typ float.

value = float(expr\_with\_params.subs(t\_sym, t\_val))

N\_exact\_list.append(value)

# Konwersja listy wyników z powrotem na tablicę NumPy

N\_exact = np.array(N\_exact\_list)

end\_time = time.time()

print(f" Czas obliczeń analitycznych: {end\_time - start\_time:.4f} s")

# --- Rozwiązanie numeryczne (SciPy) ---

sol = solve\_ivp(

fun=logistic\_growth\_rhs, t\_span=t\_span, y0=[N0], args=(r, K),

t\_eval=t\_eval, rtol=1e-8

)

N\_numeric = sol.y[0]

# --- Obliczenie błędów i wizualizacja ---

abs\_error = np.abs(N\_numeric - N\_exact)

mae = np.mean(abs\_error)

mse = np.mean((N\_numeric - N\_exact)\*\*2)

# Wykres 1: Porównanie

fig1, ax1 = plt.subplots(figsize=(10, 6))

ax1.plot(t\_eval, N\_numeric, 'k-', label='Rozwiązanie numeryczne (SciPy)', linewidth=4)

ax1.plot(t\_eval, N\_exact, 'r--', label='Rozwiązanie analityczne (SymPy)', linewidth=2)

ax1.set\_title(f"Model wzrostu logistycznego: {label}")

ax1.set\_xlabel('Czas [t]')

ax1.set\_ylabel('Wielkość populacji [N(t)]')

ax1.legend()

ax1.grid(True)

fig1.tight\_layout()

fig1.savefig(os.path.join(output\_dir, f"porownanie\_{filename\_safe\_label}.png"))

plt.close(fig1)

# Wykres 2: Błąd

fig2, ax2 = plt.subplots(figsize=(10, 6))

ax2.plot(t\_eval, abs\_error, label='Błąd bezwzględny', color='blue')

ax2.set\_title(f"Błąd bezwzględny dla: {label}")

ax2.set\_xlabel('Czas [t]')

ax2.set\_ylabel('Błąd bezwzględny')

ax2.grid(True)

ax2.set\_yscale('log')

ax2.legend()

fig2.tight\_layout()

fig2.savefig(os.path.join(output\_dir, f"blad\_{filename\_safe\_label}.png"))

plt.close(fig2)

return {"Opis": label, "MAE": mae, "MSE": mse}

# --- Scenariusze i uruchomienie ---

scenarios = [

{'label': '1. Scenariusz bazowy', 'r': 0.5, 'K': 1000, 'N0': 50},

{'label': '2. Szybki wzrost (r=2.0)', 'r': 2.0, 'K': 1000, 'N0': 50},

{'label': '3. Spadek populacji (N0 > K)', 'r': 0.5, 'K': 1000, 'N0': 1500},

{'label': '4. Niska pojemnosc srodowiska (K=500)', 'r': 0.5, 'K': 500, 'N0': 50},

{'label': '5. Start blisko granicy (N0=950)', 'r': 0.5, 'K': 1000, 'N0': 950}

]

results = []

for params in scenarios:

print(f"Uruchamiam symulację dla: '{params['label']}'...")

result = run\_and\_plot\_simulation(params)

results.append(result)

df\_results = pd.DataFrame(results)

print("\nPorównanie błędów dla różnych scenariuszy:")

print(df\_results.to\_string(index=False))

# Wyniki obliczeń

# Scenariusz bazowy (r=0.5, K=1000, N0=50)

A graph with a red line

AI-generated content may be incorrect.

**Rysunek 1.** Porównanie trajektorii numerycznej i analitycznej (dokładnej) dla scenariusza bazowego.

A graph showing a wave of blue lines

AI-generated content may be incorrect.

**Rysunek 2.** Wykres błędu bezwzględnego dla scenariusza bazowego.

# Szybki wzrost (r=2.0):

A graph with a red line

AI-generated content may be incorrect.

**Rysunek 3.** Porównanie trajektorii numerycznej i analitycznej (dokładnej) dla scenariusza z wysokim współczynnikiem wzrostu (r=2.0).

A graph showing a wave

AI-generated content may be incorrect.

**Rysunek 4.** Wykres błędu bezwzględnego w funkcji czasu dla scenariusza z wysokim współczynnikiem wzrostu.

# Spadek populacji (N0 > K):

A graph with a line

AI-generated content may be incorrect.

**Rysunek 5.** Porównanie trajektorii numerycznej i analitycznej (dokładnej) dla scenariusza spadku populacji (N0 > K).

A graph with blue lines

AI-generated content may be incorrect.

**Rysunek 6.** Wykres błędu bezwzględnego w funkcji czasu dla scenariusza spadku populacji.

# Niska pojemność środowiska (K=500):

A graph with a red line

AI-generated content may be incorrect.

**Rysunek 7.** Porównanie rozwiązania numerycznego z analitycznym (dokładnym) dla scenariusza z niską pojemnością środowiska (K=500).

A graph showing a blue line

AI-generated content may be incorrect.

**Rysunek 8.** Wykres błędu bezwzględnego w funkcji czasu dla scenariusza z niską pojemnością środowiska.

# Start blisko granicy (N0=950):

A graph with a red line

AI-generated content may be incorrect.

**Rysunek 9.** Porównanie trajektorii numerycznej i analitycznej (dokładnej) dla scenariusza ze startem blisko pojemności środowiska (N0=950).

A graph showing a wave

AI-generated content may be incorrect.

**Rysunek 10.** Wykres błędu bezwzględnego w funkcji czasu dla scenariusza ze startem blisko pojemności środowiska.

Podsumowanie dla każdego z przypadków

1. **Scenariusz bazowy:** Symulacja wiernie odwzorowuje klasyczną, esowatą dynamikę wzrostu logistycznego, przechodząc przez fazę powolnego startu, gwałtownej ekspansji i końcowego nasycenia przy osiągnięciu pojemności środowiska.
2. **Szybki wzrost:** Zwiększony współczynnik wzrostu r prowadzi do znacznie bardziej agresywnej i skróconej fazy ekspansji, w wyniku której populacja osiąga stan równowagi w znacznie krótszym czasie, co ilustruje wrażliwość modelu na ten parametr.
3. **Spadek populacji:** Ten przypadek demonstruje samoregulujący charakter modelu; populacja startująca powyżej pojemności środowiska w sposób stabilny maleje, dążąc do poziomu równowagi K, co potwierdza jego rolę jako atraktora.
4. **Niska pojemność środowiska:** Obniżona pojemność środowiska K skutecznie ogranicza maksymalną wielkość populacji, która stabilizuje się na niższym poziomie, co dowodzi, że zasoby środowiskowe są kluczowym czynnikiem limitującym wzrost.
5. **Start blisko granicy:** Startując blisko pojemności środowiska, system pomija fazę gwałtownego wzrostu i od razu wchodzi w etap nasycenia, co pokazuje zachowanie modelu w warunkach bliskich stanu równowagi i skutkuje najniższymi błędami numerycznymi.

|  |  |  |
| --- | --- | --- |
|  | MAE | MSE |
| 1. Scenariusz bazowy | 1.986482e-06 | 1.865908e-11 |
| 2. Szybki wzrost (r=2.0) | 2.474823e-06 | 1.471049e-11 |
| 3. Spadek populacji (N0 > K) | 1.138523e-06 | 2.124529e-12 |
| 4. Niska pojemność środowiska (K=500) | 1.037557e-06 | 5.227425e-12 |
| 5. Start blisko granicy (N0=950) | 8.899362e-07 | 1.157025e-12 |

Tabela 1. Obliczone średnie bezwzględne i kwadratowe błędy

# Wnioski i podsumowanie

* Przeprowadzona analiza i symulacje komputerowe pozwoliły na dokładne zbadanie modelu wzrostu logistycznego populacji oraz weryfikację poprawności jego implementacji numerycznej. Głównym wnioskiem jest potwierdzenie wysokiej zgodności rozwiązania numerycznego z analitycznym, co udokumentowano za pomocą niskich wartości błędów MAE i MSE (Tabela 1.) we wszystkich analizowanych scenariuszach.
* Badanie pięciu różnych scenariuszy w czytelny sposób zilustrowało kluczowy wpływ parametrów modelu na jego dynamikę. Wykazano, że współczynnik wzrostu r determinuje tempo osiągania stanu równowagi, pojemność środowiska K wyznacza stabilny poziom maksymalny populacji, a warunek początkowy N0​ decyduje o początkowym zachowaniu systemu – wzroście lub spadku w kierunku punktu równowagi.
* Szczegółowa analiza wykresów błędu bezwzględnego dostarczyła głębszych wniosków na temat działania solwera numerycznego (ode). Potwierdzono, że największe odchylenia od rozwiązania dokładnego występują w fazach najszybszych zmian symulowanego systemu, gdzie pochodna funkcji osiąga ekstremum. Dodatkowo, zaobserwowano, że **częstotliwość oscylacji błędu jest odwrotnie proporcjonalna do dynamiki systemu**. W okresach dużej dynamiki, jak na początku scenariusza spadku populacji, solwer używa małych kroków czasowych, co skutkuje gęstymi oscylacjami błędu. Natomiast w miarę stabilizacji systemu, solwer adaptacyjnie zwiększa krok czasowy, co wizualnie objawia się jako "rozciągnięcie" i spadek częstotliwości oscylacji widocznych na wykresie.
* Podsumowując, zrealizowany projekt nie tylko zweryfikował poprawność modelu logistycznego, ale również pozwolił na praktyczne zbadanie charakterystyki i zachowania nowoczesnych metod numerycznych.