Rozwiązywanie równań różniczkowych metodą Eulera i Runge-Kutty

Michał Saturczak
Informatyka Stosowana, Akademia Górniczo-Hutnicza
Metody Numeryczne

13 czerwca 2024

1 Wstęp teoretyczny

Wśród metod numerycznych, metoda Eulera oraz metoda Rungego-Kutty drugiego rzędu (RK2) są jednymi z najprostszych i najczęściej stosowanych podejść do rozwiązywania RZR. Metoda Eulera, oparta na dyskretyzacji pochodnej, stanowi podstawę dla wielu innych metod numerycznych. Mimo swojej prostoty, metoda Eulera może charakteryzować się ograniczoną dokładnością, szczególnie w przypadku problemów o dużej zmienności rozwiązania.

Metoda Rungego-Kutty drugiego rzędu (RK2) stanowi ulepszenie metody Eulera, wykorzystując średnią ważoną nachyleń w kilku punktach, co prowadzi do lepszej aproksymacji rozwiązania. RK2 oferuje wyższą dokładność niż metoda Eulera przy zachowaniu względnej prostoty implementacji.

2 Problem

2.1 Rozwiązanie problemu

- Napisano program w języku C++ z do rozwiązania problemów związanych z rozwiązaniem równań różniczkowych
- Zapisanie wyników do plików .csv dla trzech różnych wartości zmiennej t dla obu metod
- Wygenerowanie po 3 wykresów dla obu metod

2.2 Funkcje użyte do rozwiązania problemy

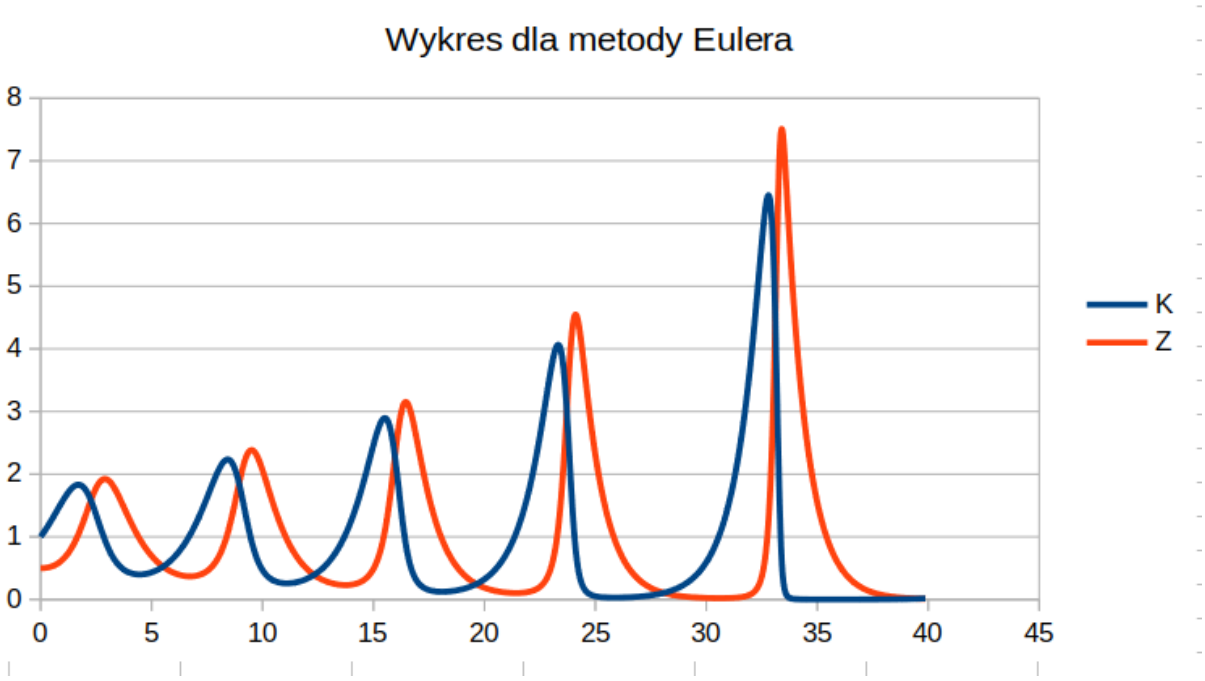
```
// Metoda Eulera
void eulerMethod(double t0, double tK, double dt,
const std::vector<double> &y0, std::ofstream &outfile)
{
  std::vector<double> y = y0;
  double t = t0;

  while (t <= tK)
  {
    outfile << t << "," << y[0] << "," << y[1] << std::endl;

    std::vector<double> dydt(2);
    volterraLotka(t, y, dydt);

    for (int i = 0; i < 2; ++i)
    {
      y[i] += dt * dydt[i];
    }
    t += dt;
  }
}

void rungeKutta2(double t0, double tK, double dt,
const std::vector<double> &y0, std::ofstream &outfile)
{
  std::vector<double> y = y0;
  double t = t0;

  while (t <= tK)
  {
    outfile << t << "," << y[0] << "," << y[1] << std::endl;

    std::vector<double> k1(2), k2(2);
```

```
    // k1
    volterraLotka(t, y, k1);

    // k2
    for (int i = 0; i < 2; ++i)
    {
      k2[i] = y[i] + dt * k1[i];
    }
    volterraLotka(t + dt, k2, k2);

    // Aktualizacja y
    for (int i = 0; i < 2; ++i)
    {
      y[i] += (dt / 2.0) * (k1[i] + k2[i]);
    }

    t += dt;
  }
}
```

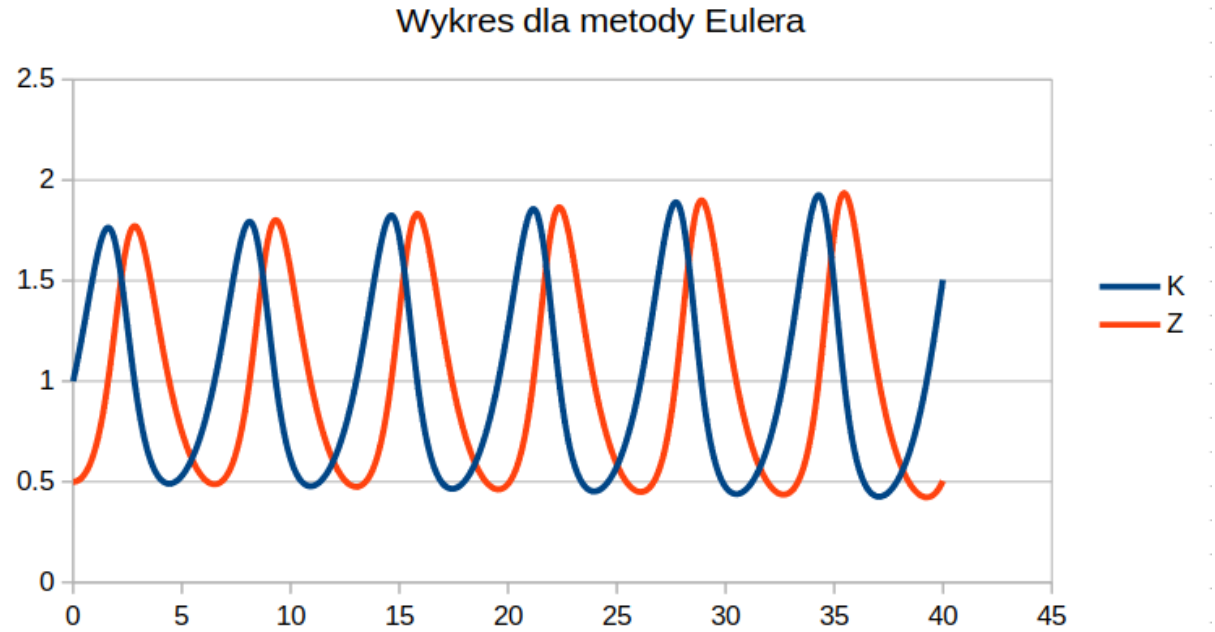
3 Wyniki

3.1 Wykresy dla Metody Eulera

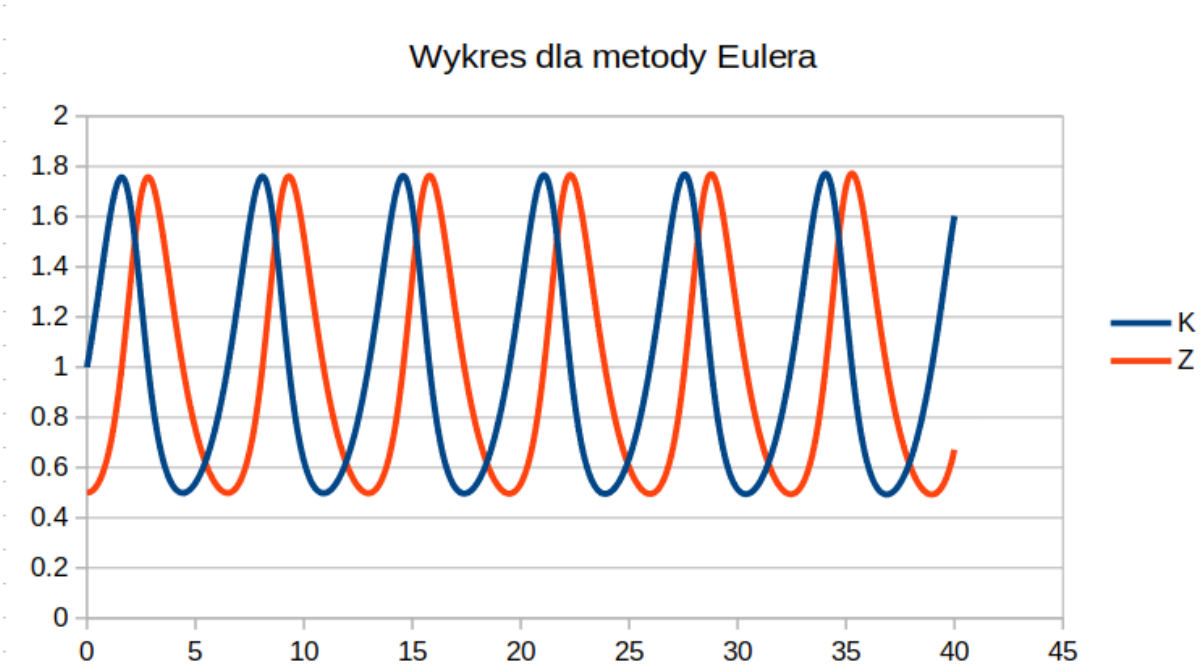
3.1.1 $t = 0.1$



3.1.2 $t = 0.01$

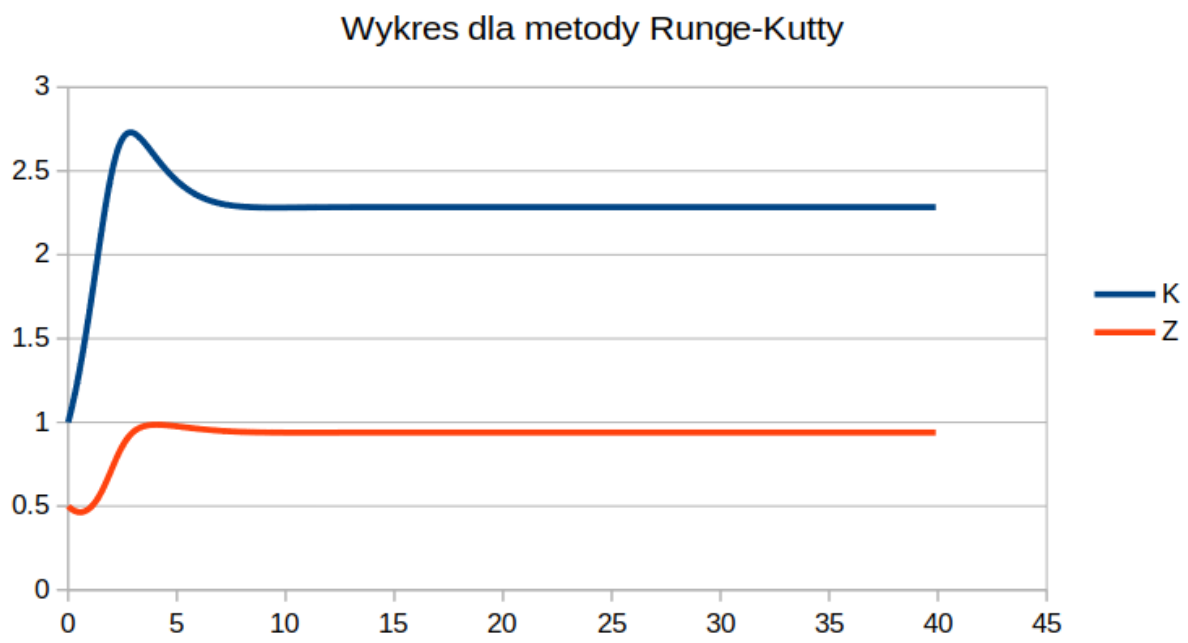


3.1.3 $t = 0.001$

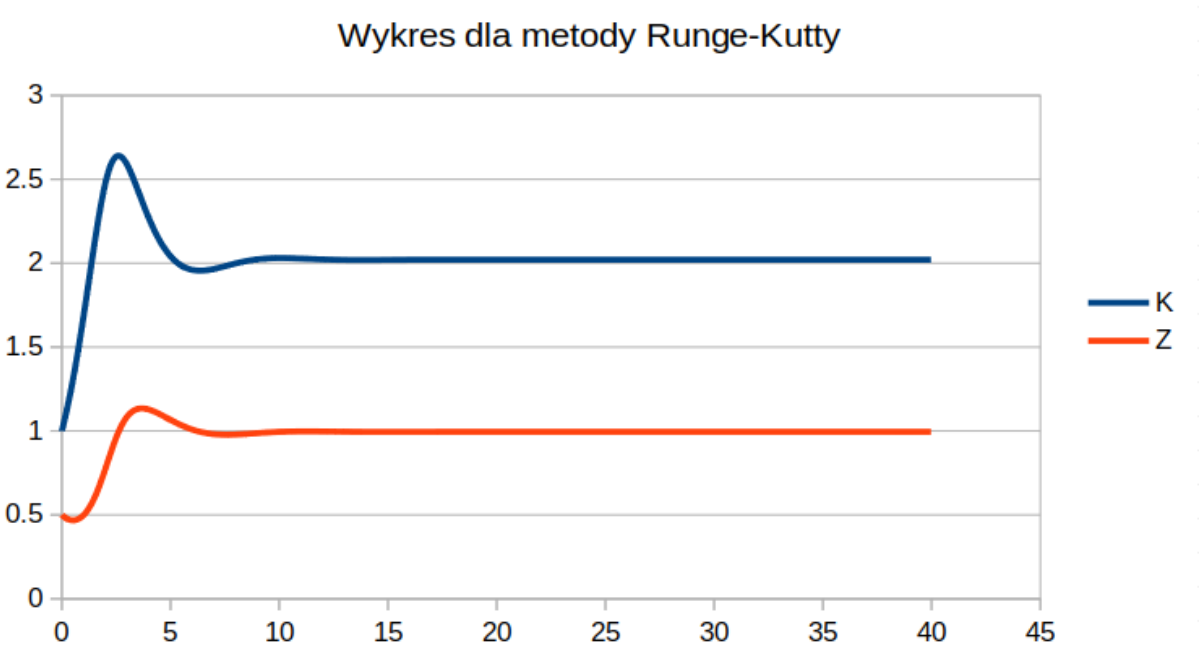


3.2 Wykresy dla Metody Runge-Kutty

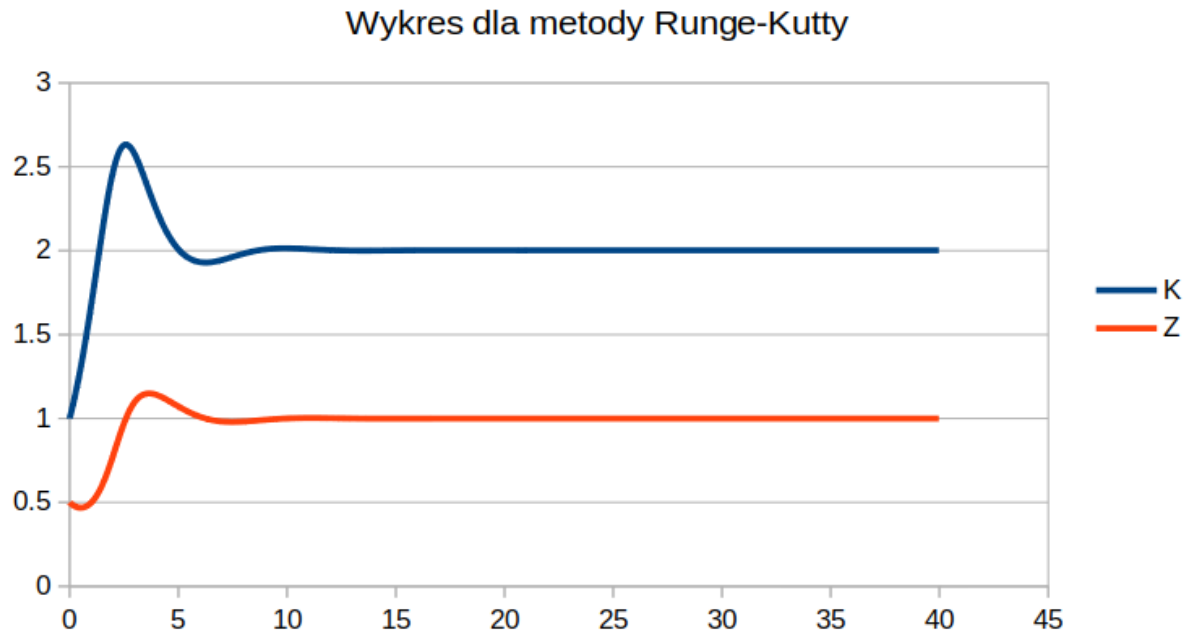
3.2.1 $t = 0.1$



3.2.2 $t = 0.01$



3.2.3 $t = 0.001$



4 Wnioski

Metoda Eulera, ze względu na swoją prostotę, często generuje rozwiązania o mniejszej dokładności, szczególnie w przypadku problemów charakteryzujących się dużą zmiennością rozwiązania. Obserwowane rozbieżności między wynikami metody Eulera a oczekiwanymi wartościami mogą wynikać z akumulacji błędów lokalnych, które narastają wraz z postępem symulacji.

Zastosowanie metody Rungego-Kutty drugiego rzędu (RK2) pozwala na uzyskanie bardziej precyzyjnych wyników, dzięki uwzględnieniu dodatkowych informacji o nachyleniu funkcji w kolejnych punktach. Niemniej jednak, możliwe jest, że zaobserwowane problemy z rozwiązaniem uzyskanym metodą RK2 wynikają z błędów implementacyjnych, takich jak nieprawidłowe obliczenie współczynników wagowych czy nieodpowiedni dobór kroku czasowego.