Raport lab 4

Jan Komorowski

Celem zajęć Lab 4 jest zapoznanie się z podstawowymi algorytmami rozwiązującymi równania różniczkowe numerycznie.

W programie zaimplementowane zostały dwa algorytmy rozwiązujące równania różniczkowe w postaci:

$$\begin{cases} \frac{dy}{dt} = \lambda * y(t) \\ y(t_0) = y_0 \end{cases}.$$

W tym celu korzystamy z dwóch metod: Eulera i Rungego-Kutty 4-ego rzędu. Pierwsza z nich korzysta z faktu, że pochodna funkcji w punkcie oznacza jej przyrost lub nachylenie prostej stycznej oraz w warunku początkowego $y(t_0) = y_0$. Jest to metoda prosta w implementacji, ale niestety bardzo niestabilna i potrzebuje małego skoku i względnie dużego przedziału (od t_0 do t_k, żeby otrzymać zadowalające przybliżenie. Metoda Rungego-Kutty 4-ego rzędu korzysta ze schematu iteracyjnego $y_{n+1} = y_n + \Delta y_n$

$$\Delta y_n = \frac{1}{6}(k_1 + k_2 + k_3 + k_4)$$

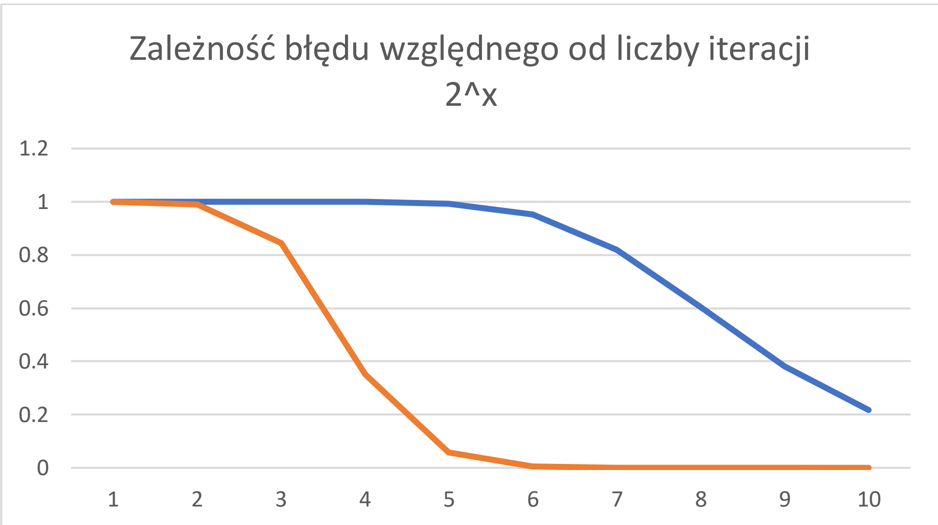
Gdzie

$$k_1 = hf(x_n, y_n)$$
$$k_2 = hf(x_n + \frac{h}{2}, y_n + \frac{1}{2}k_1)$$
$$k_3 = hf(x_n +, y_n + \frac{1}{2}k_2)$$
$$k_4 = hf(x_n + h, y_n + k_3)$$

Metoda ta jest o wiele wydajniejsza i do uzyskania zadawalającego przybliżenia pożądanego zagadnienia, nie potrzeba wiele iteracji. Jest ona również prosta w implementacji i praktycznie nie ma żadnych większych ograniczeń. Poniżej przedstawiony jest wykres zależności błędu względnego od liczby iteracji, gdzie argumentami są kolejne potęgi dwójki. Wartości początkowe to:

$$\lambda = 2,$$
$$y_0 = 1$$
$$t_0 = 0$$

Wykres przedstawiony jest dla chwili $t_k = 8$.



Kolorem pomarańczowym przedstawiona jest zależność dla metody Rungego-Kutty 4-ego rzędu, a niebieskim Eulera. Wartości zbliżone do jedynki oznaczają bardzo duży błąd, zaś zbliżone do zera, bardzo mały błąd. Jak widać algorytm zaznaczony na pomarańczowo potrzebował tylko 64 iteracji, żeby otrzymać bardzo dokładne przybliżenie. Niestety metoda zaznaczona kolorem niebieskim, nawet po 1024 iteracjach nie zbliżyła się do poziomu drugiej metody.

Raport lab 4

Jan Komorowski

Podsumowując metoda Eulera może być stosowana w niektórych przypadkach, gdzie bardzo dokładne przybliżenia nie są wymagane oraz wykonujemy odpowiednią ilość powtórzeń. W większości przypadków o wiele lepszym rozwiązaniem będzie metoda Rungego-Kutty 4-ego rzędu. Zapewnia ona dużą dokładność przy stosunkowo niewielkiej liczbie wykonanych pętli.

Uwagi do kodu:

Program korzysta z zmodyfikowanej biblioteczki „rk4". W samej funkcji rk4 została zaimplementowana pętla oraz dodany został argument chwili, dla której ma być pokazana wartość. Poniżej przedstawiona jest dana modyfikacja:

```
double rk4(double x0, double y0, double h, double (*fun)(double, double),double t)
{
    double q = fabs((t-x0)/h);
    double y1;
    double k1,k2,k3,k4;
    for (int i = 0; i < q; ++i) {
        k1 = h * fun(x0, y0);
        k2 = h * fun(x0 + h / 2., y0 + k1 / 2.);
        k3 = h * fun(x0 + h / 2., y0 + k2 / 2.);
        k4 = h * fun(x0 + h, y0 + k3);
        y1 = y0 + (k1+2.*k2+2.*k3+k4)/6.;
        y0=y1;
    }
    return y1;
}
```