# Dokumentacja projektu laboratoryjnego numer 4 przedmiot MNUM

Marcin Dziedzic

10 czerwca 2018

## Spis treści



## 1 Zadanie 1

### 1.1 Polecenie

Ruch punktu jest opisany równaniami:

$$x_1' = x_2 + x_1(0,2 - x_1^2 - x_2^2)$$

$$x_2' = -x_1 + x_2(0,2 - x_1^2 - x_2^2)$$

Należy obliczyć przebieg trajektorii na przedziale [0.20] dla następujących warunków początkowych:

$$a) x_1(0) = 8, x_2(0) = 7$$

$$b)x_1(0) = 0, x_2(0) = 0,4$$

$$c)x_1(0) = 5, x_2(0) = 0$$

$$d)x_1(0) = 0,01, x_2(0) = 0,001$$

### 1.2 Ogólny opis zagadnienia rozwiązywania układu równań różniczkowych

Rozważane jest zagadnienie układu równań (pogrubione wartości to wektory) różniczkowych zwyczajnych pierwszego rzędu (ale mogą być one nieliniowe). Dane jest równanie (układ rówań):
$\mathbf{x}'(t) = \mathbf{f}(t, \mathbf{x})$, przy czym $\mathbf{x}$ to szukana funkcja. Znany jest przedział, na którym szukamy $\mathbf{x}$: $t \in [a, b]$ oraz warunki początkowe: $\mathbf{x}(a)$, .Wyróżnia się metody jednokrokowe, bazujące tylko na punckie otzymanym w poprzedniej iteracji, oraz metody wielokrokowe, które opierają się na większej liczbie punktów.

## 2 Metoda RK4 ze stałym krokiem

### 2.1 Opis algorytmu

Metody Rungego - Kutty to grupa metod jednokrokowych. Na przedziale $[t_n, t_n + h]$ obliczane są pochodne $\mathbf{x}$ (poprzez podstawienie do funkcji $\mathbf{f}$ danej równaniu), w różnych punktach w badanym przedziale, a następnie badana funkcja jest przybliżana przez pewną liniową kombinację tych pochodnych. Kompromisem pomiędzy dokładnością (rzędem metody) a nakładem obliczeń na jedną iterację jest metoda RK4 (obliczenie pochodnej w 4 punktach). Wzory opisujące jedną iterację:

$$\mathbf{x_{n+1}} = \mathbf{x_n} + \frac{1}{6}h(\mathbf{k_1} + 2\mathbf{k_2} + 2\mathbf{k_3} + \mathbf{k_4})$$

$$\mathbf{k_1} = \mathbf{f}(t_n, \mathbf{x_n})$$

$$\mathbf{k_2} = \mathbf{f}(t_n + \frac{1}{2}h, \mathbf{x_n} + \frac{1}{2}h\mathbf{k_1})$$

$$\mathbf{k_3} = \mathbf{f}(t_n + \frac{1}{2}h, \mathbf{x_n} + \frac{1}{2}h\mathbf{k_2})$$

$$\mathbf{k_4} = \mathbf{f}(t_n + h, \mathbf{x_n} + h\mathbf{k_3})$$

### 2.2 Kod funkcji zwracającej punkt określony równaniami

```
function [D] = md_fx(x)
% Funkcja zwracajaca pochodne
D(1) = x(2) + x(1)*(0.2 -x(1)^2 -x(2)^2);
D(2) = -x(1) + x(2)*(0.2 -x(1)^2 -x(2)^2);
end
```

### 2.3 Kod funkcji zwracającej rozwiązania metodą RK4

```
function [Y] = md_rk4s(x,timelimit,stp)
% Rozwiazanie ukladu metoda Rungego-Kutty czwartego rzedu
% x - stan poczatkowy
% timelimit - zakres czasu
% step - rozmiar kroku

    Y = zeros(ceil(timelimit/stp),3); %macierz stanow x1, x2 i czasu
```

```
    hstp=stp/2; %polowa kroku
    for i = 1:(ceil(timelimit/stp))
        Y(i,3) = i*stp;
        k1 = md_fx(x);
        k2 = md_fx(x+hstp*k1);
        k3 = md_fx(x+hstp*k2);
        k4 = md_fx(x+stp*k3);
        x=x+(1/6)*stp*(k1+2*k2+2*k3+k4);
        Y(i,1:2) = x;
        %zapisanie punktu do wektora
    end
end
```

### 2.4 Kod programu generujący dane wynikowe dla podpunktu 1

```
% Realizacja zadania 1
% metoda RK4 ze stalym krokiem
clear;
zero=[8 7; 0 0.4; 5 0; 0.01 0.001]; %wektor stanow poczatkowych
step = 0.0004; %krok

for k = 3:3
    data = md_rk4s(zero(k,:),20,step);
    h = figure;
    plot(data(:,1),data(:,2),'-o');
    l = size(data,1);
    hold on;
    grid on;
    name = ['metoda RK4 krok:' num2str(step) ' podpunkt:' num2str(k)];
    title(name);
    saveas(h,name,'jpg');
end
```

### 2.5 Dobieranie długości kroku

Największym wyzwaniem w powyższej metodzie jest dobranie odpowiedniej długości kroku. Moim zadaniem było dobranie go w taki sposób, aby był wystarczający do uzyskania założonej dokładności, ale nie powinien być znacznie mniejszy od wartości, przy której wymagana dokładność jest osągalna. Postanowiłem dokonać tego interaktywnie, na zasadzie prób i błędów. Posługiwałem się metodą przeszukiwania binarnego. Zainicjalizowałem zakres przeszukiwań od kroku bardzo małego do bardzo dużego, następnie wybierałem wartość środkową. W zależnośći, czy chciałem osągnąć lepszą lub gorszą aproksymację wybierałem lewą lub prawą połowę kolejnego przedziału.

### 2.6 Wnioski

Metoda Rungego-Kutty dla podanych równań jest szybka i stabilna numeryczie przy odpowiednim kroku. Zmniejszając krok możaby coraz dokładniej wyznaczyć trajektorię. Dobór odpowiedniej długości kroku w sposób interaktywny jest problematyczny ze względu na różnorodność kroków w zaleznosci od pountow startowych. Metoda róznież ma dość wysoki nakład obliczeniowy ponieważ jest on zależny tylko od przedziału i długości kroku, dlatego nie ma znaczenia czy równania są skomplikowane czy bardzo proste.

## 3 Metody wielokrokowe

Metody wielokrokwe w odróżnieniu od iteracyjnych uzywają do wyznaczenia punktu wartości obliczonych w poprzednich krokach. Wynika z tego że przy rozpoczynaniu rozwiązywania zagadnienia metodą wielokrokową musimy wyznaczyć punkty początkowe (zazwiczaj jedną z metod iteracyjnych) a następnie na podstawie wartości początkowch obliczane są kolejne punkty. Medod tych używa się w przypadku gdy wyznaczanie wartości bezpośrednio ze wzoru jest czasochłonne, ponieważ metody wielokrokowe rzadziej wykorzystują bezpośrednie obliczanie wartości funkcji. Wyróżniamy metody jawne które wykorzystują jedynie wartości funkcji w poprzednio obliczonych punktach oraz niejawne kore dodatkowo korzystają z wartości w punkcie bierzącym.

## 3.1 Metoda predyktor-korektor Adamsa

Metoda predykcyjno-korekcyjna łączy zalety metod jawnych i niejawnych. Dzięki temu zachowuje wysoki rząd i małą stałą błędu na szerokim obszarze przy zachowaniu małej liczby iteracji. Praktyczne realizacja metody polega na obliczeniu czterech członów: P - predykcja, E - ewaluacja, K - korekcja, E - kolejna ewaluacja. Dla metody predyktor - korektor Adamsa rozwiązanie możemy przybliżyć następującymi równaniami:

$$P : y_n^{[0]} = y_{n-1} + h\sum_{j=1}^{k} \beta_j f_{n-j}$$

$$E : f_n^{[0]} = f(x_n, y_n^{[0]})$$

$$K : y_n = y_{n-1} + h\sum_{j=1}^{k} \beta_j^* f_{n-j} + h\beta_0^* f_n^{[0]}$$

$$E : f_n = f(x_n, y_n)$$

## 3.2 Realizacja funkcji RK4 w programie Matlab

```
function [Y] = pkadams(x,timelimit,stp)
%funkcja zwracajaca wektor wyznacziony metoda PK adamsa
%dane wejsciowe
%x - wektor danych
%timelimit - zakres czasu
%stp - dlugosc kroku

    Y = zeros(timelimit/stp,3); %wektor stanow x1, x2, czasu, i bledu
    hstp=stp/2;
    for i = 1:3 %generowanie punktow poczatkowych
        Y(i,3) = i*stp; %generowanie czasu
        k1 = f_x(x); %pochodna w punkcie y(xn);
        k2 = f_x(x+hstp*k1); %pochodna w punkcie y(xn+stp/2*k1)
        k3 = f_x(x+hstp*k2); %pochodna w punkcie y(xn+stp/2*k2)
        k4 = f_x(x+stp*k3); %pochodna w punkcie y(xn+stp*k3)
        x=x+(1/6)*stp*(k1+2*k2+2*k3+k4); %obliczenie nastepnego punktu
        Y(i,1:2) = x; %zapisanie punktu do wektora
    end
    for i = 4:(timelimit/stp)
        Y(i,3) = i*stp; %gererowanie czasu
        tmp = x + stp/24*(55*f_x(x) - 59*f_x(Y(i-1,1:2)) + 37*f_x(Y(i-2,1:2)) - 9*f_x(Y(i-3,1:2))); % predykcja i ewaluacja
        x = x + stp/24*(9*f_x(tmp) + 19*f_x(x) - 5*f_x(Y(i-1,1:2)) + f_x(Y(i-2,1:2))); % korekcja i ewaluacja
        Y(i,1:2) = x; %zapis wyniku
    end
end
```

## 3.3 Skrypt generujący wykresy do zadania

```
%Realizacja podpunktu 2 metoda metoda predyktor-korektor Adamsa 4-rzedu
clear;
zero=[8 7; 0 0.4; 5 0; 0.01 0.001]; %wektor stanow poczatkowych
step = 0.5; %krok

for k = 1:4

    data = rk4static(zero(k,:),20,step);

    h = figure('visible','off');
    plot(data(:,1),data(:,2),'-o');
    l = size(data,1);
    hold on;
    xl = get(gca,'xlim');
    yl = get(gca,'ylim');
    zl = get(gca,'zlim');
    %scatter3(data(:,1),data(:,2),repmat(zl(1),l,1),'.');
    %scatter3(data(:,1),repmat(yl(2),l,1),data(:,3),'.');
    %scatter3(repmat(xl(2),l,1),data(:,2),data(:,3),'.');
    grid on;
    name = ['metoda Adamsa krok:' num2str(step) ' podpunkt:' num2str(k)
        ];
    title(name);
    saveas(h,name,'jpg');
end
```

### 3.4 Dobieranie długości kroku

Podobnie jak w poprzednim podpunkcie, długość kroku była wprowadzana przez użytkownika w trybie interaktywnym.

### 3.5 Wnioski

Metoda predyktor-korektor Adamsa zdecydowanie dokładniej określa trajektorium niż metoda RK4 dla tego samego kroku, jednak potrzebuje znacznie mniejszego kroku do "ustabilizowania się". Różnica pomiędzy krokiem dla którego metoda jest zbieżna a krokiem dla którego jest dokłądna jest niewielka w przeciwieństwie do metody RK4. Dla dużych punktów początkowych metoda podobnie jak RK4 potrzebuje dosyć małego kroku aby uzyskać zbierzność na danym przedziale.

## 4 Metoda RK4 ze zmiennym krokiem

### 4.1 Opis algorytmu

Jeżeli chodzi o metodę obliczeń to jest ona identyczna jak w tej ze stałym krokiem. Na szczególną uwagę zasługuje sposób oceny błędu i na tej podstawie regulacji długości kroku. Określanie błedu jest realizowane metodą połowienia kroku. W pojedyńczej iteracji obliczne są wartości funkcji dla całego kroku i jego połowy, następnie jeżeli różnica między otrzymanymi wartościami jest mniejsza niż założony współczynnik dokładności, to krok jest zwiększany, a gdy większa to zmniejszany. Jako wartość funkcji brana pod uwagę jest zawsze ta wyznaczona za pomocą dwóch mniejszych kroków. Długość kroku jest dodatkowo mnożona przez współczynnik bezpieczeństwa tak aby wzrost kroku nie był zbyt drastyczny, i aby również drastycznie nie zmalał, co doprowadziło by do znacznego wydłużenia czasu obliczeń.

### 4.2 Realizacja funkcji w programie Matlab

```
function [Y] = rk4dynamic(x,timelimit,stp)
%RK4 Rozwiazanie ukladu metoda Rungego-Kutty czwartego rzedu ze zmiennym
%krokiem
%x - stan poczatkowy
%timelimit - zakres czasu
%step - rozmiar kroku


    Y=zeros(10,3);
    x1 = x;
    x2 = x;

    time = 0; %zmienna przechowujaca aktualny przedzial czasu
    i=1; %iterator indeksu wektorow

    while(time<=timelimit)

        k1 = f_x(x1);
        k2 = f_x(x1+(stp/2)*k1);
        k3 = f_x(x1+(stp/2)*k2);
        k4 = f_x(x1+stp*k3);
        x1=x1+(1/6)*stp*(k1+2*k2+2*k3+k4);

        k1 = f_x(x2);%obliczenie wartosci z polowicznym krokeim
        k2 = f_x(x2+(stp/4)*k1);
        k3 = f_x(x2+(stp/4)*k2);
        k4 = f_x(x2+(stp/2)*k3);
        x2=x2+(1/6)*(stp/2)*(k1+2*k2+2*k3+k4);
        k1 = f_x(x2);
        k2 = f_x(x2+(stp/4)*k1);
        k3 = f_x(x2+(stp/4)*k2);
        k4 = f_x(x2+(stp/2)*k3);
        x2=x2+(1/6)*(stp/2)*(k1+2*k2+2*k3+k4);

        R = (x2-x1)/15;

        if(abs(min(R))<0.00001) %kryterium bledu wzglednego
            stp = stp*1.1;
        else
            if(stp>0.05)
                stp = stp*0.9;
            end
        end
        Y(i,1:2) = x2;
        time = time+stp;
        disp(stp);
        Y(i,3) = time; %zapisanie czasu
        i = i+1;
    end
end
```

### 4.3 Funkcja generująca rozwiązania do zadania

```
%Realizacja podpunktu 3 metoda metoda RK4 zmienny krok
```

```
clear;
zero=[8 7; 0 0.4; 5 0; 0.01 0.001]; %wektor krokow
step = 0.01; %krok

for k = 1:4

    data = rk4dynamic(zero(k,:),20,step);

    h = figure('visible','off');
    plot(data(:,1),data(:,2),'-o');
    l = size(data,1);
    hold on;
    xl = get(gca,'xlim');
    yl = get(gca,'ylim');
    zl = get(gca,'zlim');
   %scatter3(data(:,1),data(:,2),repmat(zl(1),l,1),'.');
   %scatter3(data(:,1),repmat(yl(2),l,1),data(:,3),'.');
   %scatter3(repmat(xl(2),l,1),data(:,2),data(:,3),'.');
    grid on;
    name = ['metoda RK4 zmienny krok podpunkt:' num2str(k)];
    title(name);
    saveas(h,name,'jpg');
end
```

### 4.4 Winoski

Metoda RK4 ze zmiennym krokiem znacznie wydajniej radzi sobie z obliczeniami ze względu na istotne zmniejszenie liczby kroków na gładkich odcinkach funkcji. Zaimplementowana prze ze mnie megoda doboru długości kroku nie nalezy jednak do wyrafinowanych. Przez co wzrost wydajności nie jest szczególnie mocno zauważalny.

## 5 Porównanie z wynikami funkcji ode45

Funkcja ode45 dobrała znacznie większe kroki co widać na wykresach, przez to czas obliczeń był rzędy razy mniejszy niż zaimplementowanych przeze mnie metod. Jeżeli chodzi o dokładność to wykresy wygenerowane funkcą ode45 są dużo mniej "gładkie"jednak krok jest dobierany równomiernie, i nie widać nakładjących się na siebie punktów na wykresie, co świadczy o optymalności metody.

## 6 Wnioski końcowe

W przypadku metod jednopunktowych stałokrokowych najlepiej wypada metoda adamsa ze względu na najniższy czas wykonania obliczeń, jednak jest ona zbierzna w wąskim przedziale kroku. Metoda RK4 ze zmiennym krokiem również bardzo dokładnie wyznacza trajektoriam jednak zdecydowanie przegrywa z metodą ode45 ze zmiennym krokiem pod względem wydajności. Przypuszczem że zastosowanie zmiennego kroku dla metody predyktor-korektor Adamsa mogło by dać najlepszy rezultat dla danych z tego zdania, ponieważ wymaga najmniejszej liczby kroków przy wysokiej dokładności.