Numerik

Beispiel 2.) $f \in C([t_0, T], \mathbb{R}^{n+1})$ $f\ldots$ Lipschitzstetig

$$y'(t) = f(t, y(t)) \qquad t \in [t_0, T] \quad y(t_0) = y_0$$

$$y_{k+1} = y_k + \tau\left[(1-\theta) f(t_k, y_k) + \theta f(t_{k+1}, y_{k+1})\right] \quad k \in \mathbb{N}_0 \quad \theta \in [0,1]$$

a.) Dissipativ, wenn $Re(z) < |z|^2 \cdot (\theta - \frac{1}{2})$ $\quad z = \tau\lambda$

$$y_{k+1} = y_k + \tau \cdot \left[(1-\theta)\lambda y_k + \theta\lambda \cdot y_{k+1}\right]$$

$$\Leftrightarrow y_{k+1} - \tau\lambda\theta \cdot y_{k+1} = y_k + \tau\lambda(1-\theta) \cdot y_k$$

$$\Leftrightarrow (1 - \tau\lambda\theta) \cdot y_{k+1} = (1 + \tau\lambda \cdot (1-\theta)) y_k$$

$$\tau\lambda = z \Leftrightarrow y_{k+1} = \frac{1 + z \cdot (1-\theta)}{1 - z\theta} y_k = \frac{1 - z\theta + z}{1 - z\theta} y_k = \left(1 + \frac{z}{1 - z\theta}\right) y_k$$

$$\Rightarrow r_\theta(z) = 1 + \frac{z}{1 - z\theta}$$

$$z = a + bi \Rightarrow r_\theta(z) = 1 + \frac{a+bi}{1 - a\theta - b\theta i} \cdot \frac{(1-a\theta) + b\theta i}{(1-a\theta) + b\theta i} = 1 + \frac{a + bi - a^2\theta - \cancel{ab\theta i} + \cancel{ab\theta} i - b^2\theta}{(1-a\theta)^2 - b^2\theta^2 i^2} =$$

$$= \underline{1 + \frac{a - \theta \cdot (a^2+b^2)}{(1-a\theta)^2 + b^2\theta^2} + \frac{b}{(1-a\theta)^2 + b^2\theta^2}}$$

$$\Rightarrow |r_\theta(z)| = 1 + \frac{2a - 2\theta(a^2+b^2)}{(1-a\theta)^2 + b^2\theta^2} + \frac{a^2 - 2a\theta \cdot (a^2+b^2) + \theta^2(a^2+b^2)^2 \overset{\text{Imaginärteil}}{+ b^2}}{\left[(1-a\theta)^2 + b^2\theta^2\right]^2}$$

$$= 1 + \frac{2a - 2\theta(a^2+b^2)}{(1-a\theta)^2 + b^2\theta^2} + \frac{(a^2+b^2) \cdot (1 - 2a\theta + \theta^2 a^2 + \theta^2 b^2)}{\left[(1-a\theta)^2 + b^2\theta^2\right]^2} =$$

$$= 1 + \frac{2a - 2\theta(a^2+b^2)}{(1-a\theta)^2 + b^2\theta^2} + \frac{(a^2+b^2) \cdot \cancel{\left[(1-a\theta)^2 + \theta^2 b^2\right]}}{\left[(1-a\theta)^2 + b^2\theta^2\right]^{\cancel{2}}} = 1 + \frac{2a - 2\theta(a^2+b^2) + (a^2+b^2)}{(1-a\theta)^2 + b^2\theta^2} =$$

$$= 1 + \frac{2a + (a^2+b^2) \cdot (1 - 2\theta)}{(1-a\theta)^2 + b^2\theta^2} \qquad \text{Klar: Nenner} > 0$$

$\hookrightarrow$ gilt $a < (a^2+b^2) \cdot (\theta - \frac{1}{2}) \quad | \cdot (-2)$

$$\Leftrightarrow -2a > (a^2+b^2) \cdot (1 - 2\theta)$$

$$\Leftrightarrow 0 > 2a + (a^2+b^2) \cdot (1 - 2\theta) \Rightarrow \text{Zähler} < 0 \Rightarrow |r_\theta(z)| < 1$$ ✓

b.) Methode dispersiv, wenn $Im(z) \neq 0$ für $z = \lambda\tau$ $\Rightarrow b \neq 0$ für $z = a + bi$

$$\tan(\phi) = \frac{Im(r_\theta(z))}{Re(r_\theta(z))} = \frac{\frac{b}{(1-a\theta)^2 + b^2\theta^2}}{\frac{(1-a\theta)^2 + b^2\theta^2 + a - \theta\cdot(a^2-b^2)}{(1-a\theta)^2 + b^2\theta^2}} = \frac{b}{(1-a\theta)^2 + b^2\theta^2 + a - \theta\cdot(a^2-b^2)} \neq 0 \text{ weil } b \neq 0$$

$$\Rightarrow \tan(\phi) \neq 0 \quad \Rightarrow \phi \neq 0 \quad \checkmark$$

c.) Methode konservativ, wenn $\theta \neq 0$

$$y_{k+1} = y_k + \tau\cdot\left[(1-\theta)\cdot f(t_k, y_k) + \theta\cdot f(t_{k+1}, y_{k+1})\right] \qquad k \in \mathbb{N}_0 \quad \theta \in [0,1]$$

$$\Rightarrow \frac{U_{k+1} - U_k}{\tau} = (1-\theta)BU_k + \theta BU_{k+1} \quad \Rightarrow U_{k+1} - \tau\theta BU_{k+1} = \tau(1-\theta)BU_k + U_k \qquad B \text{ schiefsymmetrisch}$$

$$\Leftrightarrow (I - \tau\theta B)U_{k+1} = (I + \tau\cdot(1-\theta)B)U_k$$

Weil B schiefsymmetrisch $\Rightarrow$ $\cdot\, B^T = -B$

$\cdot\, U^TBU = 0$

$$\Rightarrow \|U_{k+1}^T U_{k+1}\| = \|U_{k+1}^T U_{k+1} + \underbrace{U_{k+1}^T(-\tau\theta B)U_{k+1}}_{0}\| = \|U_{k+1}^T(I - \tau\theta B)U_{k+1}\| = \|U_{k+1}^T(I + \tau(1-\theta)B)U_k\| =$$

$$= \|U_{k+1}^T U_k + \tau\cdot(1-\theta)U_{k+1}^T BU_k\|$$

Und $\|U_k^T U_k\| = \|U_k^T U_k + U_k^T\cdot(\tau\cdot(1-\theta)B)U_k\| = \|U_k^T(I + \tau\cdot(1-\theta)B)U_k\| =$

$$= \|U_k^T(I - \tau\theta B)U_{k+1}\| = \|U_k^T U_{k+1} - \tau\theta U_k^T BU_{k+1}\| = \|(U_k^T U_{k+1} - \tau\theta U_k^T BU_{k+1})^T\| =$$

$$= \|U_{k+1}^T U_k - \tau\theta U_{k+1}^T B^T U_k\| = \|U_{k+1}^T U_k + \tau\theta U_{k+1}^T B U_k\|$$

$$\Rightarrow \|U_{k+1}^T U_k + \tau\cdot\underline{(1-\theta)}U_{k+1}^T BU_k\| \overset{!}{=} \|U_{k+1}^T U_k + \tau\underline{\theta} U_{k+1}^T BU_k\|$$

$$\Rightarrow 1 - \theta = \theta \quad \Rightarrow \theta = \frac{1}{2}$$

```
th = 3/4;
t0 = 0;
T = 2 * pi;
y0 = [2;2];
Q = [0 -1;1 0];
F = @(y) (1-norm(y,2)^2) * y + Q*y;

Yv=[]; l = 1; m = 0;
```

Variablen bekommen jeweilige Werte zugewiesen

```
for k = 1:5
    M = 2^(10+k);
    y = y0; yv = y;
    ta = T/M;
    for k=2:M+1
        y_k_j = y0;
        for j=1:10
            y_k_jn = y + ta * ((1-th)*F(y) + th*F(y_k_j));
            y_k_j = y_k_jn;
        end
        y  = y_k_j;
        yv = [yv,y];
    end
    Yv = [Yv,[yv(1,(1+l):l:end)';yv(2,(1+l):l:end)']];
    l  = 2*l; m = m+1;
end
```

10-malige Verfeinerung innerhalb von y_k auf y_k+1

```
err = max(abs(Yv(:,1:m-1)-Yv(:,2:m)),[],1);
ord = log2(mean(err(1:m-2)./err(2:m-1)));
disp(sprintf('ord = %0.10f\n',ord));
```

Auswertung

```
rd = 0.9798722101
```

Ergebnis

```
% Dieser Code zeigt die Wirkung von theta in der theta-Methode fuer die
%     Wellengleichung.
% Fuer theta in (0,1/2) gibt es keine Stabilitaet fuer dieses Problem.
% Fuer theta = 1/2 ist die Methode 2. Ordnung, aber mit nicht glatten
%     Anfangswerten, die viele Fouerier Komponenten haben, fahren diese
%     Komponenten numerisch mit verschiedenen Geschwindigkeiten wegen
%     Dispersion, und Schwingungen entstehen!
% Fuer theta in (1/2,1] ist die Methode 1. Ordnung, und mit nicht glatten
%     Anfangswerten gibt es keine Schwingungen, weil diese durch
%     Dissipativitaet gedaempft werden!

N  = 256;                          % Anzahl der Zellen
om = 1;                            % omega
h  = 1/N;
x  = linspace(0,+1,N+1)';          % Schnittstellen
ta = h;                            % tau
th = 0.75;                         % theta
I  = speye(2*N-1);
Q  = spdiags([-ones(N-1,1),ones(N-1,1)],[0,1],N-1,N)/h;
B  = [sparse(N,N),-Q';Q,sparse(N-1,N-1)];
                                   % theta-Methode
A  = I - th*om*ta*B;               % fuer k+1, linke Seite
C  = I + (1-th)*om*ta*B;           % fuer k, rechte Seite

% U = sin(pi*x(2:N));              % glatte Anfangswerte
U = sign(x(2:N)-1/4) ...           % nicht glatte Anfangswerte
  - sign(x(2:N)-3/4);

G = -om*Q'*U;                      % Gradient
V = zeros(N-1,1);                  % Geschwindigkeit
W = [G;V];                         % Zustand

plot(x,[0;U;0]); axis([0,+1,-3,+3]); drawnow;

for k=2:N+1
    Ua = U;
    Va = V;
    W  = A\(C*W);
    G  = W(1:N);
    V  = W(N+1:2*N-1);
    U  = Ua + ta*(th*V + (1-th)*Va);
    plot(x,[0;U;0]); axis([0,+1,-3,+3]); drawnow;
end
```

```
clear;
clc;
%Beispiel 4 Blatt 11
%a
M=100;
T=1;
omeg=0.5;
tau=T/M;
F=[];
t=linspace(0,T,M+1);
err=[];
N=[50,100,200,400];
for i=1:4
    h  = 1/N(i);
    x  = linspace(0,1,N(i)+1)';
    u  = kron(sin(pi*x),cos(omeg*pi*t));
    Q=zeros(N(i)-1,N(i));
    for j=1:N(i)-1
        Q(j,j)=-1;
        Q(j,j+1)=1;
    end
    L=(1/h^2)*Q*Q.';

    V_k  = sin(kron((1:(N(i)-1))',(1:(N(i)-1)))*pi/N(i))*sqrt(2/N(i));

    la = sin(pi*(1:(N(i)-1))'/(2*N(i))).^2*4/h^2;

    u_h = sin(pi*x(2:N(i)));
    for k=2:M+1
        Z = spdiags(cos(-sqrt(la)*omeg*t(k)),0,N(i)-1,N(i)-1);
        u_h = [u_h,V_k*Z*V_k'*sin(pi*x(2:N(i)))];
    end
    u_h = [zeros(1,M+1);u_h;zeros(1,M+1)];
    err = [err,norm(u_h(:)-u(:),inf)];


end
ord= log2(mean(err(1:4-1)./err(2:4)))
disp(sprintf('pure semi disrete ord = %0.10f',ord));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%b
N=100;
M=[50,100,200,400];
Q=zeros(N-1,N);
h=1/N;
%Q definieren (ohne 1/h)
for j=1:N-1
        Q(j,j)=-1/h;
        Q(j,j+1)=1/h;
end
L=Q*Q.';
B  = [sparse(N,N),-Q';Q,sparse(N-1,N-1)];
V  = sin(kron((1:(N-1))',(1:(N-1)))*pi/N)*sqrt(2/N);
la = sin(pi*(1:(N-1))'/(2*N)).^2*4/h^2;
err=[];
x=linspace(0,1,N+1)';
for i=1:4
    tau=1/M(i);
    t=linspace(0,T,M(i)+1);
    u = sin(pi*x(2:N));
    U = sin(pi*x(2:N));
    G = -omeg*Q'*U;
    V_k = zeros(N-1,1);
    W = [G;V_k];
    for k=2:M(i)+1
        Z = spdiags(cos(-sqrt(la)*omeg*t(k)),0,N-1,N-1);
        u = [u,V*Z*V'*sin(pi*x(2:N))];
        W = [W,(speye(2*N-1) - 0.5*tau*omeg*B) \ ((speye(2*N-1) + 0.5*tau*omeg*B)*W(:,k-1))];
        G = [G,W(1:N,k)];
        V_k = [V_k,W(N+1:2*N-1,k)];
```

```
        U = [U,U(:,k-1) + tau*(V_k(:,k)+V_k(:,k-1))/2];
    end
    u = [zeros(1,M(i)+1);u;zeros(1,M(i)+1)];
    U = [zeros(1,M(i)+1);U;zeros(1,M(i)+1)];
    err = [err,norm(u(:)-U(:),inf)];

end
ord = log2(mean(err(1:4-1)./err(2:4)))
disp(sprintf('fully to semi disrete ord = %0.10f',ord));
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%c)
N_v = [50,100,200,400];
err  = [];
for N=N_v
    M  = N;
    t  = linspace(0,1,M+1);
    tau = 1/M;
    h  = 1/N;
    Q = zeros(M-1,M);
    for l= 1:M-1
    Q(l,l) = -1/h;
    Q(l,l+1) = 1/h;
    end
    L  = Q*Q';
    B = zeros(2*M-1, 2*M-1);
    B(1:M,M+1:2*M-1) = -Q.';
    B(M+1:2*M-1,1:M) = Q;
    x  = linspace(0,1,N+1)';
    u  = kron(sin(pi*x),cos(omeg*pi*t));
    V  = sin(kron((1:(N-1))',(1:(N-1)))*pi/N)*sqrt(2/N);
    la = sin(pi*(1:(N-1))'/(2*N)).^2*4/h^2;
    U = sin(pi*x(2:N));
    G = -omeg*Q'*U;
    V_k = zeros(N-1,1);
    W = [G;V_k];
    for k=2:M+1
        W = [W,(speye(2*N-1) - 0.5*tau*omeg*B) \ ((speye(2*N-1) + 0.5*tau*omeg*B)*W(:,k-1))];
        G = [G,W(1:N,k)];
        V_k = [V_k,W(N+1:2*N-1,k)];
        U = [U,U(:,k-1) + tau*(V_k(:,k)+V_k(:,k-1))/2];
    end
    U = [zeros(1,M+1);U;zeros(1,M+1)];
    err = [err,norm(u(:)-U(:),inf)];
end
ord = log2(mean(err(1:4-1)./err(2:4)))
disp(sprintf('fully to semi disrete ord = %0.10f',ord));
```

bsp6.m

```
Q = [0,-1;1,0];
y0 = [1;0];

T = 8 * pi;
M=60;
tau = T/M;

A = [1/2+sqrt(3)/6,0;-sqrt(3)/3,1/2+sqrt(3)/6];
b = [1/2;1/2];
theta = [1/2+sqrt(3)/6; 1/2-sqrt(3)/6];

QUOT_INV = inv(eye(2) - tau*A(1,1)*Q);

yk = y0;
for k = 1:M
    y1 = QUOT_INV * yk;
    y2 = QUOT_INV * (yk - tau*sqrt(3)/3*Q*y1);
    yk = yk + tau*Q/2*(y1+y2);
end

diff_arg = abs(atan2(yk(2), yk(1)) - atan2(0,1));
fprintf("norm diff: %.15f\n", abs(norm(yk) - norm([1;0])));
fprintf("arg diff: %.15f\n", diff_arg);

%b
% plot(sign(abs(1 + (x+i*y)/(1-(x+i*y)*(1/2 + sqrt(3)/6)) - sqrt(3)/6 * ((x+i*y)/(1-(x+i*y)*(1/2 + sqrt(3)/6)))^2)-1), x=-10..10, y=-10..10)
```

UE Numerik
Stefan Kreiner
01610951
2021-01-15

6)

$$\begin{array}{c|cc} \frac{1}{2}+\frac{\sqrt{3}}{6} & \frac{1}{2}+\frac{\sqrt{3}}{6} & 0 \\ \frac{1}{2}-\frac{\sqrt{3}}{6} & -\frac{\sqrt{3}}{3} & \frac{1}{2}+\frac{\sqrt{3}}{6} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

$$a := \left(\frac{1}{2}+\frac{\sqrt{3}}{6}\right)$$

$$y_{k,1} = y_k + \tau\left(a\, f(t_k + a\tau, y_{k,1})\right)$$

$$y_{k,2} = y_k + \tau\left(-\frac{\sqrt{3}}{3} f\left(t_k + \left(\frac{1}{2}+\frac{\sqrt{3}}{6}\right)\tau, y_{k,1}\right) + a f\left(t_k + \left(\frac{1}{2}-\frac{\sqrt{3}}{6}\right)\tau, y_{k,2}\right)\right)$$

$$y_{k+1} = y_k + \frac{\tau}{2}\left(f(t_k + a\tau, y_{k,1}) + f\left(t_k + \left(\frac{1}{2}-\frac{\sqrt{3}}{6}\right)\tau, y_{k,2}\right)\right)$$

a) $f(t, y(t)) = \lambda y(t)$

$$y_{k,1} = y_k + \tau a \lambda y_{k,1} \rightsquigarrow y_{k,1} = \frac{y_k}{1-\tau a\lambda}$$

$$y_{k,2} = y_k + \tau\left(-\frac{\sqrt{3}}{3}\lambda y_{k,1} + a\lambda y_{k,2}\right)$$

$$= y_k + \tau\left(-\frac{\sqrt{3}}{3}\right)\lambda y_{k,1} + \tau a \lambda y_{k,2}$$

$$\rightsquigarrow y_{k,2} = \frac{y_k + \lambda\tau\left(-\frac{\sqrt{3}}{3}\right) y_{k,1}}{1-\lambda\tau a}$$

$$y_{k+1} = y_k + \frac{\tau}{2}\left(\lambda y_{k,1} + \lambda y_{k,2}\right)$$

$$= y_k + \frac{\lambda\tau}{2}\left(\frac{y_k}{1-\lambda\tau a} + y_{k,1}\left(1 + \frac{\lambda\tau\left(-\frac{\sqrt{3}}{3}\right)}{1-\lambda\tau a}\right)\right)$$

$$= y_k + \frac{\lambda\tau}{2}\left(\frac{y_k}{1-\lambda\tau a} + \frac{y_k}{1-\lambda\tau a}\left(1 + \frac{\lambda\tau\left(-\frac{\sqrt{3}}{3}\right)}{1-\lambda\tau a}\right)\right)$$

$$= y_k\left(1 + \frac{\lambda\tau}{2}\left(\frac{1}{1-\lambda\tau a}\right)\left(1 + 1 + \frac{\lambda\tau\left(-\frac{\sqrt{3}}{3}\right)}{1-\lambda\tau a}\right)\right)$$

$$= y_k\left(1 + \frac{\lambda\tau}{2}\left(\frac{2}{1-\lambda\tau a} - \frac{\sqrt{3}}{3}\frac{\lambda\tau}{(1-\lambda\tau a)^2}\right)\right)$$

$$= y_k\left(1 + \frac{\lambda\tau}{1-\lambda\tau a} - \frac{\sqrt{3}}{6}\left(\frac{\lambda\tau}{1-\lambda\tau a}\right)^2\right)$$

$$g(k) := \frac{k}{1-ka} \implies y_{k+1} = y_k\left(1 + g(\lambda\tau) - \frac{\sqrt{3}}{6} g(\lambda\tau)^2\right)$$

$$r(z) := 1 + g(z) - \frac{\sqrt{3}}{6} g(z)^2$$

$$\|y_{k+1}\| = \|y_k \cdot r(\lambda\tau)\| = |r(\lambda\tau)| \cdot \|y_k\|$$

$$\|y_{k+1}\| = \|y_k\| \Leftrightarrow |r(\lambda\tau)| = 1$$

$|r(i)| \overset{\rightsquigarrow \text{Wolframalpha } (\sim 0{,}936)}{<} 1 \Rightarrow$ nicht konservativ (etwa für $\lambda = i \cdot \frac{1}{\tau}$)

b) analog hergeleitet

$\rightsquigarrow$ Wolframalpha: plot ~~||~~ with a few different resolutions

c) Matlab (Falsch)

d) $y'(t) = Q\,y(t) \qquad (\Rightarrow f(t, y(t)) = Q\,y(t))$

$$\Rightarrow y_{k+1} = y_k + \frac{\tau}{2}\left(Q\,y_{k,1} + Q\,y_{k,2}\right)$$

$$y_{k,1} = y_k + \tau a\,Q\,y_{k,1} \Rightarrow -y_k = (\tau a Q - I)\,y_{k,1}$$

bzw $\quad y_{k,1} - \tau a\,Q\,y_{k,1} = y_k$ bzw $y_{k+1} = (I - \tau a Q)^{-1}\,y_k$

$$y_{k,2} = y_k + \tau\left(-\tfrac{\sqrt{3}}{3}\right)Q\,y_{k,1} + \tau a\,Q\,y_{k,2}$$

$$\Leftrightarrow (I - \tau a Q)\,y_{k,2} = y_k + \tau\left(-\tfrac{\sqrt{3}}{3}\right)Q\,y_{k,1}$$

$$\Leftrightarrow y_{k,2} = (I - \tau a Q)^{-1}\left(y_k + \tau\left(-\tfrac{\sqrt{3}}{3}\right)Q\,y_{k,1}\right)$$

Wenn man die berechnung von $(I - \tau a Q)^{-1}$ also einmal am anfang macht ~~sollte~~ und dann nicht dazugezählt wird reichen diese zwei Lösungen also aus.

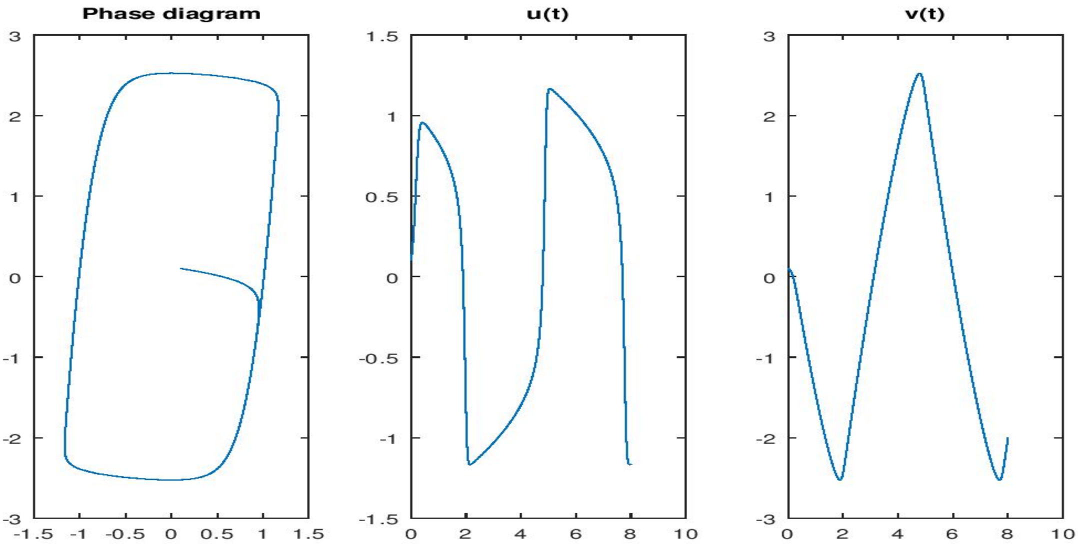
Selbstverstandlich muss man auch y_{k+1} berechnen, aber das ist ein ganz normale matrixmult. $\left(y_{k+1} = y_k + \frac{\tau}{2}Q\,(y_{k,1} + y_{k,2})\right)$

Christoph Raunjak	Blatt 11	Numerik I SS20

Aufgabe 8

☑ a.) ☐ b.) ☑ c.) ☐ d.)

Phasenraum und Lösung der Hysterese-DGL mittels adaptiver RK4/RK5 Methode nach Dormand-Prince:



Code:

Variblen und Parameter und die zu verwendende Butcher-Table festlegen.

```matlab
%***************************************************************************
%*                         Worksheet 11 ex 8                               *
%*                                                                         *
%*                Implementing an adaptive Runge-Kutta Method              *
%*                                                                         *
%* We implement the adaptive RK4/RK5 Method of Dormand-Prince to solve the *
%* hysteresis differential equation. We compare the results with the Matlab *
%* integrated method ode45. We also use just the RK4 and RK5 method and compare*
%* the results.                                                            *
%*                                                                         *
%* Author:  Christoph Raunjak                                              *
%* email:   christoph.raunjak@edu.uni-graz.at                              *
%* vs 2.0, 14.1.2020                                                       *
%***************************************************************************

clc,clear;

%model the differential equation
a = 2; b = 2; c = 10; u0 = [0.1;0.1];
T = 8;
F = @(t,u) [c * u(1) * (1- u(1)^2) + a * u(2);-b * u(1)];



% Dormand-Prince Butcher table
% b gives the fifth order solution
% bt gives the fourth order solution

q = 7; nu = 4;
D = [           0,           0,           0,          0,             0,          0,      0; ...
             +1/5,           0,           0,          0,             0,          0,      0; ...
            +3/40,       +9/40,           0,          0,             0,          0,      0; ...
           +44/45,      -56/15,       +32/9,          0,             0,          0,      0; ...
      +19372/6561,-25360/2187,  +64448/6561, -212/729,             0,          0,      0; ...
       +9017/3168,     -355/33,  +46732/5247,  +49/176,   -5103/18656,          0,      0; ...
          +35/384,           0,   +500/1113, +125/192,    -2187/6784,    +11/84,      0];
th = [          0;        +1/5;       +3/10;     +4/5;          +8/9;        +1;     +1];
b  = [    +35/384;           0;   +500/1113; +125/192;    -2187/6784;    +11/84;      0];
bt = [+5179/57600;           0; +7571/16695; +393/640; -92097/339200; +187/2100; +1/40];
db = [  +71/57600,           0,   -71/16695, +71/1920, -17253/339200,   +22/525, -1/40];
```

Aufgabe a.)

```
%-------------------------------------------------------------------------------
%a.) ord > 4 (correct)
%Solve u'=F(t,u(t)) with RK4
UU4 = [];
l = 1; m = 0;
for k = 1:6
    ta = T/(2^(7+k));
    t = 0; tv = t;
    u = u0; uv = u;
    U = zeros(2,q); %storing the u_{k,i}
    while(t<T)
        U(:,1) = u;
        %calculate u_{k,i}
        for i = 2:q
            U(:,i) = u;
            for j = 1:(i-1)
                U(:,i) = U(:,i) + ta * D(i,j)*F(t + ta*th(j), U(:,j));
            end
        end
        %calculate u_{k+1}
        for i = 1:q
            u = u + ta * bt(i) * F(t + ta * th(i), U(:,i));
        end
        uv = [uv, u];
        t = t + ta; tv = [tv,t];
        ta = min(T-t,ta);
    end
    UU4 = [UU4,[uv(1,(1+l):l:end)';uv(2,(1+l):l:end)']];
    l  = 2*l; m = m+1;
endfor

%calculate the order
err = max(abs(UU4(:,1:m-1)-UU4(:,2:m)),[],1);
ord = log2(mean(err(1:m-2)./err(2:m-1)));
disp(sprintf('ord RK4 = %0.10e',ord));

%-------------------------------------------------------------------------------
```

Aufgabe b.)

Exakkt gleicher Code mit Ausnahme dass für die Berechnung von u_{k+1} der Vektor b anstatt von bt verwendet wird (2. Zeile unter %calculate u_{k+1})

Adaptive Methode

```
% Using the adaptive RK4/RK5 method (RKA)
% time stepping parameters
tamin = T/(2^13); tamax = T/(2^8);
demin = 0.5; demax = 2.0;
ep = 1.0e-4;
Fcount = 0; %count calls of function F
ta = tamax;
t = 0; tv = t;
u = u0; uv = u;
U = zeros(2,q);
while (t < T)
    refine = true;
    while (refine)
    %RK5 from bevor
        U(:,1) = u;
        for i = 2:q
          U(:,i) = u;
          for j = 1:(i-1)
            U(:,i) = U(:,i) + ta * D(i,j)*F(t + ta*th(j), U(:,j));
            Fcount += 1;
          end
        end
        un = u;
        for i = 1:q
          un = un + ta * b(i)* F(t + ta*th(i), U(:,i));
          Fcount += 1;
        end
        %-----------------------------------------------------------------------
        %calculate du (differenze RK4 and RK5)
        du = zeros(size(u));
        for i = 1:q
            du = du + db(i) * F(t + ta * th(i), U(:,i));
            Fcount += 1;
        end
        du = norm(du,inf);
        %-----------------------------------------------------------------------

        %alternativ
        %Anfällig für Auslöschung
        %-----------------------------------------------------------------------
        %ut = u;
        %for i = 1:q
        %ut = ut + ta * bt(i) * F(t + ta*th(i), U(:,i));
        %end
        %du = norm((ut-un)/ta,inf);
        %-----------------------------------------------------------------------

        %adapt ta
        de = max([demin, min(demax, nthroot(ep / du, nu))]);
        tn = min([T-t,tamax, max(tamin, ta*de)]);
        if (de >= 1)
           refine = false;
        else
           ta = tn;
        end

     end

     %store final solution for this timestep
     u = un; uv = [uv,u];
     t = t + ta; tv = [tv, t];
     if (t >= T)
         break;
     end
     ta = tn;
end
%-------------------------------------------------------------------------------
```

Aufgabe c.) und d.)
Verwendet die Ergebnisse der adaptiven Methode.

```
%--------------------------------------------------------------------------
%c.) mean(|u-u_star|_2) < 1.8e-3 (correct)
%solve with Matlab
opts = odeset('RelTol',ep,'AbsTol',ep);
[tm,xm] = ode45(F,[0,T],u0,opts);
tm = tm';
xm = xm';
u_ad = uv;
t_ad = tv;
%interpolate the RKA solution in order to compare to matlab solution
xi = [interp1(t_ad,u_ad(1,:),tm,'linear','extrap'); ...
      interp1(t_ad,u_ad(2,:),tm,'linear','extrap')];

disp(sprintf('durchschnittlicher RKA Fehler = %0.10e', ...
    mean(sqrt((xi(1,:)-xm(1,:)).^2 + (xi(2,:)-xm(2,:)).^2))));

%--------------------------------------------------------------------------

%--------------------------------------------------------------------------
%d.) Fcount = 4453 (incorrect)
% Show Fcount, which was increased every time F got evaluated in the adaptive
% method above
disp('Anzahl der Evaluierungen:')
disp(Fcount);

%--------------------------------------------------------------------------

%--------------------------------------------------------------------------
```