

Praktikum 1 : DGL

André Harms, Oliver Steenbuck

02.11.2011

Inhaltsverzeichnis

1	Steife Differentialgleichungen	2
1.1	Betrachtete Gleichung	2
1.2	Simulink/Analogrechner	2
1.3	Iterationsgleichungen	3
1.3.1	Euler (expl)	3
1.3.2	Euler (impl)	3
1.3.3	Runge-Kutta 2	3
1.4	Matlab Programme	3
1.4.1	Euler (expl)	3
1.4.2	Euler (impl)	3
1.4.3	RungeKutta	4
1.4.4	stiff	5
1.5	Ergebnisdrucke	5
1.6	Interpretation der Ergebnisse	5
2	Van-der-Pol-DGL	5
2.1	Simulink/Analogrechner	5
2.2	Zu DGL 1 Ordnung transformierte DGL	6
2.3	Iterationsgleichungen	6
2.3.1	Euler (expl)	6
2.3.2	Runge-Kutta 2	6
2.4	Ergebnisdrucke	6
2.5	Interpretation der Ergebnisse	6
3	Lorenz-Attraktor mit RK 2	6
3.1	Simulink/Analogrechner	6
3.2	Iterationsgleichungen	6
3.2.1	Runge-Kutta 2	6

3.3	Matlab Programme	6
3.3.1	Lorenz	6
3.4	Ergebnisdrucke	6
3.5	Interpretation der Ergebnisse	6

Abbildungsverzeichnis

1	Simulink-Schaltbild: Steife DGL	2
2	Simulink-Schaltbild: Van-der-Pol-DGL	5

Listings

1	Explizites Euler-Verfahren	3
2	Implizites Euler-Verfahren	3
3	Runge Kutta	4

1 Steife Differentialgleichungen

1.1 Betrachtete Gleichung

$$y(0) = 1 \quad (1)$$

$$y' = 10 - 500 \cdot y + 5000 \cdot x \quad (2)$$

1.2 Simulink/Analogrechner

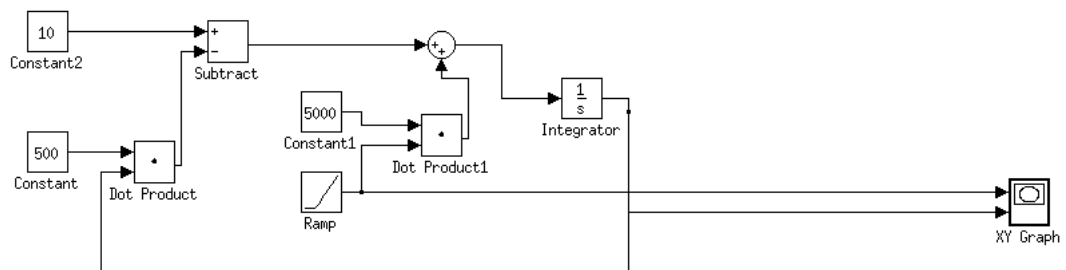


Abbildung 1: Simulink-Schaltbild: Steife DGL

1.3 Iterationsgleichungen

1.3.1 Euler (expl)

$$u_{j+1} = u_j + h \cdot (10 - 500 \cdot u_j + 5000 \cdot x_j) \quad (3)$$

1.3.2 Euler (impl)

1.3.3 Runge-Kutta 2

$$k_{1j} = f(x_j, u_i) \quad (4)$$

$$k_{2j} = f(x_j + h, u_j + h \cdot k_{1j}) \quad (5)$$

$$u_{j+1} = u_j + \frac{h}{2} \cdot (k_{1j} + k_{2j}) \quad (6)$$

1.4 Matlab Programme

1.4.1 Euler (expl)

Listing 1: Explizites Euler-Verfahren

```
1 function [x, y] = euler_expl(h, xend, f)
2     ska_y_i = 1;
3     vec_ytmp = [];
4     vec_xtmp = [];
5
6     vec_range = [0:h:xend];
7     for ska_cur_x=vec_range
8         ska_y_i = ska_y_i + h * f(ska_cur_x, ska_y_i);
9         vec_xtmp = [vec_xtmp, ska_cur_x];
10        vec_ytmp = [vec_ytmp, ska_y_i];
11    end
12
13    y = vec_ytmp;
14    x = vec_xtmp;
15 end
```

1.4.2 Euler (impl)

Listing 2: Implizites Euler-Verfahren

```
1 function [x, y] = euler_expl(h, xend)
```

```
2     ska_y_i = 1;  
3     vec_ytmp = [];  
4     vec_xtmp = [];  
5  
6     vec_range = [0:h:xend];  
7     for ska_cur_x=vec_range  
8         ska_y_i = ska_y_i + h * mtp0101(ska_cur_x, ska_y_i);  
9         vec_xtmp = [vec_xtmp, ska_cur_x];  
10        vec_ytmp = [vec_ytmp, ska_y_i];  
11    end  
12  
13    y = vec_ytmp;  
14    x = vec_xtmp;  
15 end
```

1.4.3 RungeKutta

Listing 3: Runge Kutta

```
1 function [x, y] = rk2(h, xend, f)  
2     ska_y_i = 1;  
3     vec_ytmp = [];  
4     vec_xtmp = [];  
5  
6     vec_range = [0:h:xend];  
7     for ska_cur_x=vec_range  
8         k1 = f(ska_cur_x, ska_y_i);  
9         k2 = f(ska_cur_x + h, ska_y_i + h * k1);  
10        ska_y_i = ska_y_i + h/2 * (k1 + k2);  
11        vec_xtmp = [vec_xtmp, ska_cur_x];  
12        vec_ytmp = [vec_ytmp, ska_y_i];  
13    end  
14  
15    y = vec_ytmp;  
16    x = vec_xtmp;  
17 end
```

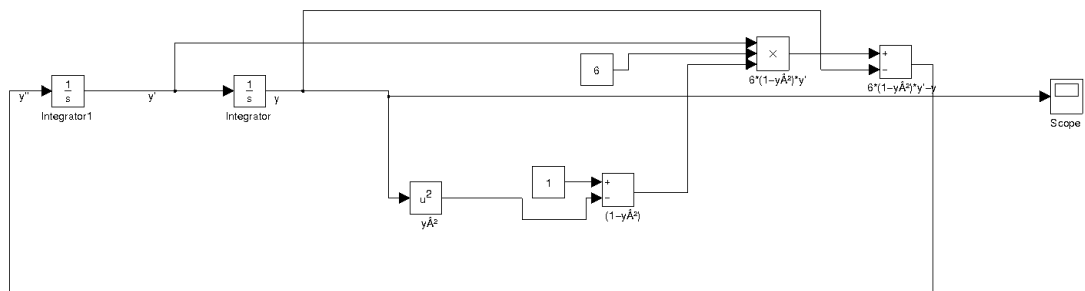
1.4.4 stiff**1.5 Ergebnisausdrucke****1.6 Interpretation der Ergebnisse****2 Van-der-Pol-DGL****2.1 Simulink/Analogrechner**

Abbildung 2: Simulink-Schaltbild: Van-der-Pol-DGL

2.2 Zu DGL 1 Ordnung transformierte DGL

2.3 Iterationsgleichungen

2.3.1 Euler (expl)

2.3.2 Runge-Kutta 2

2.4 Ergebnisausdrucke

2.5 Interpretation der Ergebnisse

3 Lorenz-Attraktor mit RK 2

3.1 Simulink/Analogrechner

3.2 Iterationsgleichungen

3.2.1 Runge-Kutta 2

3.3 Matlab Programme

3.3.1 Lorenz

3.4 Ergebnisausdrucke

3.5 Interpretation der Ergebnisse