Computer und Software 1

C. Köhler

6. Maple – Differentialgleichungen

Folien: Balint Aradi

Differentialgleichungen

Gewöhnliche Differentialgleichungen:

$$f(t, x(t), x^{(1)}(t), x^{(2)}(t), \ldots, x^{(n)}(t)) = 0 \qquad x^{(i)}(t) = \frac{d^i x(t)}{dt^i}$$

Ordnung der Differentialgleichung: Höchste vorkommende Ableitung von x
Grad der Differentialgleichung: Exponent der höchsten vorkommenden Ableitung
Wenn alle Exponenten der Ableitungen von x gleich 1 sind: lineare Differentialgleichung

$$\frac{d^2 x(t)}{dt^2} - t^2 x - t^3 = 0$$ lineare DGL, Ordnung 2, Grad 1

$$\frac{d^2 x(t)}{dt^2} - x(t)^3 = 0$$ nichtlineare DGL, Ordnung 2, Grad 1

$$\left(\frac{d^2 x(t)}{dt^2}\right)^3 - x(t) = 0$$ nichtlineare DGL, Ordnung 2, Grad 3

Wenn die Funktion von mehreren Variablen abhängt: **Partielle** Differentialgleichung

$$i\hbar \frac{d}{dt}\psi(x,t) = \frac{-\hbar^2}{2\mathrm{m}} \frac{d^2}{dx^2}\psi(x,t) + V(x)\psi(x,t)$$

Harmonischer Oszillator

Newtonsche Mechanik liefert die linearisierte Differentialgleichung für eine Schwingung:

$$F = m\,a = -k\,x \longrightarrow m\frac{d^2x(t)}{dt^2} = -k\,x(t)$$

Gewöhnliche lineare DG zweiten Grades

Gesucht wird die Funktion *x(t)*, die die Differentialgleichung erfüllt:

$$x(t) = C_1 \sin\left(\sqrt{\frac{k}{m}}t\right) + C_2 \cos\left(\sqrt{\frac{k}{m}}t\right)$$

Differentialgleichung zweiten Grades

↓

Zwei unbestimmte Konstanten

Konstanten werden über Anfangsbedingungen bestimmt:

$$x(0) = 5$$
$$v(0) = \frac{dx}{dt}(0) = 0 \longrightarrow C_1 = 0,\ C_2 = 5 \longrightarrow x(t) = 5\cos\left(\sqrt{\frac{k}{m}}t\right)$$

Eingabe in Maple

Funktionale Abhängigkeit von der unabhägigen Variablen t wird durch runde Klammern angezeigt, Ableitung dann mit der **diff()** Funktion oder mit dem **D**-Operator

```
> deq := m * diff(x(t), t$2) = -k * x(t);
```

$$deq := m\left(\frac{d^2}{dt^2}x(t)\right) = -k\,x(t)$$

x als Funktion von t

zweite Ableitung von x(t)

```
> deq2 := m * (D@@2)(x)(t) = - k * x(t);
```

$$deq2 := m\,\mathrm{D}^{(2)}(x)(t) = -k\,x(t)$$

Die Differentialgleichung wird mit **dsolve()** gelöst:

Integrationskonstanten

```
> dsolve(deq, x(t));
```

$$x(t) = \_C1\sin\left(\frac{\sqrt{k}\,t}{\sqrt{m}}\right) + \_C2\cos\left(\frac{\sqrt{k}\,t}{\sqrt{m}}\right)$$

```
> dsolve(deq2, x(t));
```

$$x(t) = \_C1\sin\left(\frac{\sqrt{k}\,t}{\sqrt{m}}\right) + \_C2\cos\left(\frac{\sqrt{k}\,t}{\sqrt{m}}\right)$$

Anfangsbedingungen

Anfangsbedingungen werden als zusätzliche Gleichungen an **dsolve()** übergeben.

```
> deq := m * diff(x(t), t$2) = -k * x(t);
```

$$deq := m\left(\frac{d^2}{dt^2}x(t)\right) = -k\,x(t)$$

```
> ini1 := x(0) = 5;
```

$$ini1 := x(0) = 5$$

Anfangsbedingung für Funktionswert

```
> ini2 := D(x)(0) = 0;
```

$$ini2 := D(x)(0) = 0$$

Anfangsbedingung für erste Ableitung

```
> sol := dsolve({deq, ini1, ini2}, x(t));
```

$$sol := x(t) = 5\cos\left(\frac{\sqrt{k}\,t}{\sqrt{m}}\right)$$

Differentialgleichung und Anfangsbedingungen als Menge

Ergebnis darstellen

Ergebnis ist eine Gleichung, kann also direkt nicht geplottet werden:

```
> sol;
```

$$x(t) = 5\cos(t)$$

```
> k := 1; m := 1;
```

$$k := 1$$
$$m := 1$$

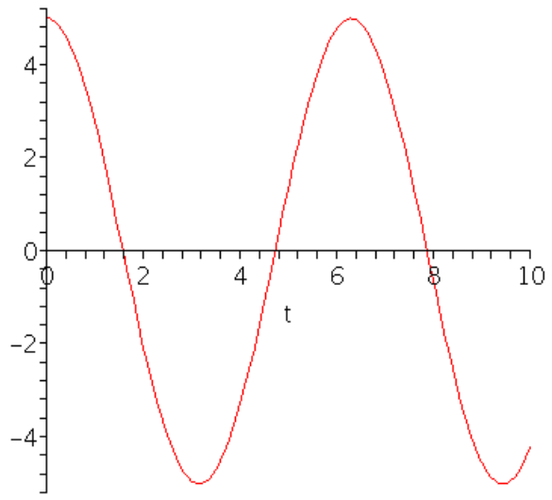
```
> plot(sol, t=0..10);
Warning, unable to evaluate the function to numeric
values in the region; see the plotting command's help
page to ensure the calling sequence is correct

Error, empty plot
```

Rechte Seite des Ergebnisses: Darstellbarer Ausdruck

```
> rhs(sol);
```

$$5\cos(t)$$

```
> plot(rhs(sol), t=0..10);
```



Ergebnis darstellen

Ähnlich wie bei **solve()**, kann das Ergebnis via **assign()** zugewiesen werden:

```
> sol;
```

$$x(t) = 5\cos(t)$$

```
> assign(sol);
> x(t);
```

$$5\cos(t)$$

Die funktionale Abhängigkeit von t muss immer angegeben werden, damit auf den Inhalt zugegriffen werden kann:

```
> x;
```

$$x$$

Die Variable wird durch Zuweisung des eigenen Namens (ohne Klammern) wieder freigegeben:

```
> x := 'x';
```

$$x := x$$

```
> x(t);
```

$$x(t)$$

Differentialgleichungssysteme

Differentialgleichungssystem: Miteinander gekoppelte Differentialgleichungen

$$\frac{dv_x(t)}{dt} = -\kappa\, v_x(t)\sqrt{v_x(t)^2 + v_y(t)^2}$$

$$\frac{dv_y(t)}{dt} = -\kappa\, v_y(t)\sqrt{v_x(t)^2 + v_y(t)^2}$$

Gewöhnliche Differentialgleichung höherer Ordnung können in ein System von Differentialgleichungen niedrigerer Ordnung konvertiert werden.

$$m\frac{d^2x(t)}{dt^2} = -k * x(t) \longrightarrow \begin{cases} \dfrac{dx(t)}{dt} = v(t) \\ m\dfrac{dv(t)}{dt} = -k\,x(t) \end{cases}$$

Unbekannte Funktion *x(t)*

Unbekannte Funktionen *x(t)* und *v(t)*

Differentialgleichungssysteme in Maple

Eingabe wie bei normalen Differentialgleichungen:

```
> deq1 := diff(x(t), t) = v(t);
```

$$deq1 := \frac{d}{dt} x(t) = v(t)$$

```
> deq2 := m * diff(v(t), t) = -k * x(t);
```

$$deq2 := m \left(\frac{d}{dt} v(t) \right) = -k\, x(t)$$

```
> ini1 := x(0) = 5;
```

$$ini1 := x(0) = 5$$

```
> ini2 := v(0) = 0;
```

$$ini2 := v(0) = 0$$

Das Gleichungssystem kann via **dsolve()** gelöst werden:

```
> sol := dsolve({deq1, deq2, ini1, ini2}, { x(t), v(t) });
```

$$sol := \left\{ x(t) = 5 \cos\left(\frac{\sqrt{k}\, t}{\sqrt{m}} \right), v(t) = -\frac{5 \sin\left(\frac{\sqrt{k}\, t}{\sqrt{m}} \right) \sqrt{k}}{\sqrt{m}} \right\}$$

Zu berechnende Funktionen als Menge

Einzelne Differentialgleichungen und Anfangsbedingungen als Menge

Lösung als Menge

Überprüfen von Ergebnissen

Ergebnisse können via **odetest()** auf Korrektheit überprüft werden.
Gibt **odetest()** Null zurück, erfüllt das Ergebnis die Gleichungen (und evtl. auch die Anfangsbedingungen):

```
> sol := dsolve({deq1, deq2, ini1, ini2}, { x(t), v(t) });
```

$$sol := \left\{ x(t) = 5\cos\left(\frac{\sqrt{k}\,t}{\sqrt{m}}\right),\ v(t) = -\frac{5\sin\left(\frac{\sqrt{k}\,t}{\sqrt{m}}\right)\sqrt{k}}{\sqrt{m}} \right\}$$

```
> odetest(sol, { deq1, deq2});
```

$$\{0\}$$

Die Lösung erfüllt das Differentialgleichungssystem

```
> odetest(sol, { deq1, deq2, ini1, ini2 });
```

$$\{0\}$$

Die Lösung erfüllt das Differentialgleichungssystem und die Anfangsbedingungen

Numerisches Lösen von DGLen

Bei komplizierteren Differentialgleichungen existiert oft keine (bzw. findet Maple keine) Lösung in geschlossener Form:

Mathematisches (nicht linearisiertes) Pendel:

$$\frac{d^2 x}{dt^2} = -\sin(x)$$

```
> deq1 := diff(x(t), t$2) = -sin(x(t));
```

$$deq1 := \frac{d^2}{dt^2} x(t) = -\sin(x(t))$$

```
> sol1 := dsolve({deq1, x(0)=0, D(x)(0)=1}, x(t));
```

$$sol1 := x(t) = \mathrm{RootOf}\left(\int_0^{\_Z} \frac{1}{\sqrt{2\cos(\_a) - 1}} d\_a - t \right)$$

Numerisches Lösen von DGLen

Differentialgleichungen können mit der Option **type=numeric** ausgewertet werden

```
> sol1 := dsolve({deq1, x(0)=0, D(x)(0)=1}, x(t),
  type=numeric);
```

$$sol1 := \mathbf{proc}(x\_rkf45) \ \ldots \ \mathbf{end\ proc};$$

Die numerische Integration ergibt eine Prozedur, die an beliebiger Stelle ausgewertet werden kann:

```
> sol1;
```

$$sol1$$

```
> sol1(2.0);
```

$$\left[t = 2.0,\ x(t) = 1.0046147244195442 5,\ \frac{d}{dt}x(t) = -0.26986499819378989 6 \right]$$

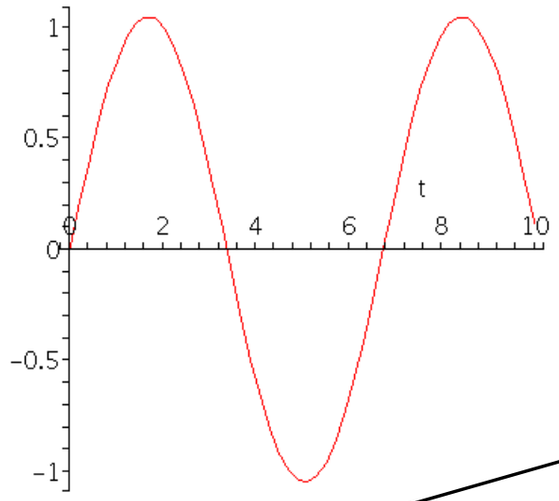
Achtung: Numerische Lösung (rkf45) kann ebenfalls fehlerbehaftet sein !

Graphische Auswertung

Die Funktion **odeplot()** (im Packet **plots**) ermöglicht die graphische Darstellung der Lösung einer DGL:

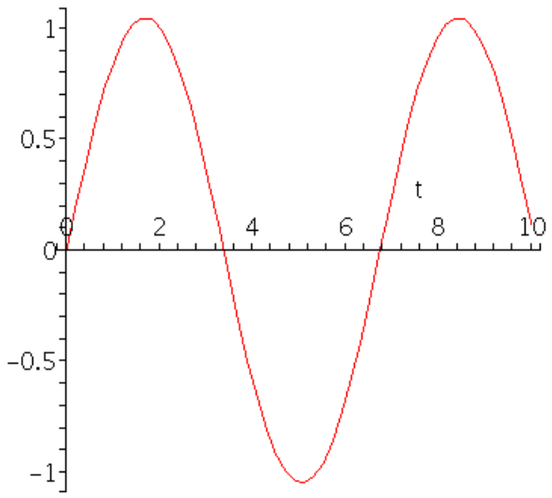
```
> with(plots):
> odeplot(sol1, t=0..10);
```

Zeichnet die Lösung in Abhängigkeit der Variable *t*



Explizite Achsenspezifikation
[x-Achse, y-Achse]

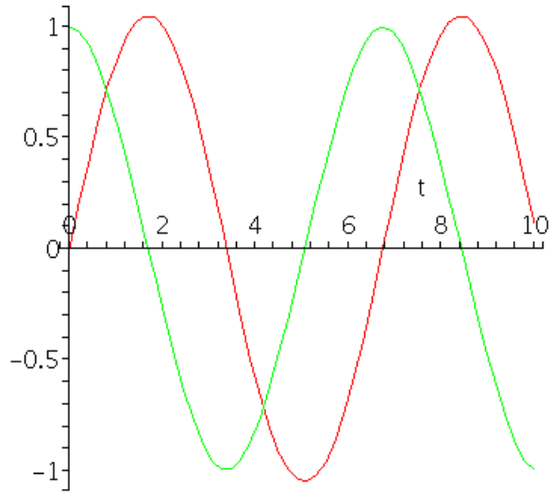
```
> odeplot(sol1, [ t, x(t) ], t=0..10);
```



Graphische Auswertung

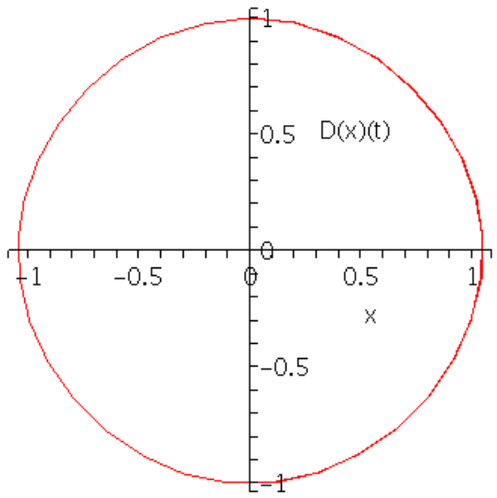
Wenn die Lösung mehrere Funktionen (oder mehrere Ableitungen einer Funktion) enthält, können diese mit **odeplot()** angezeigt werden:

```
> odeplot(sol1, [[ t, x(t) ], [ t, D(x)(t) ]], t=0..10);
```



Ort und Geschwindigkeit in Abhängigkeit der Zeit

```
> odeplot(sol1, [ x(t), D(x)(t) ], t=0..10);
```



Geschwindigkeit in Abhängigkeit der Auslenkung

Genauigkeit von numerischen Lösungen

Numerische Lösung ist immer mit Fehler behaftet, eventuell sogar unbrauchbar.
Die Genauigkeit der Integration läßt sich (u.a.) mit der Option **abserr=** verändern

```
> deq := D(x)(t) = x(t);
```

$$deq := \mathrm{D}(x)(t) = x(t)$$

```
> sol := dsolve({deq, x(0)=1}, x(t), type=numeric,
  abserr=1e-5);
```

$$sol := \mathbf{proc}(x\_rkf45) \ \ldots \ \mathbf{end\ proc};$$

```
> sol(0.2);
```

$$[t = 0.2,\ x(t) = 1.2214033153967693]$$

```
> sol2 := dsolve({deq, x(0)=1}, x(t), type=numeric,
  abserr=1e-10);
```

$$sol2 := \mathbf{proc}(x\_rkf45) \ \ldots \ \mathbf{end\ proc};$$

```
> sol2(0.2);
```

$$[t = 0.2,\ x(t) = 1.2214027320312457 2]$$

```
> exp(0.2);
```

$$1.221402758$$

Geschlossene Lösung: *exp(x)*

Fallstudie: Mathematisches Pendel (#1)

Die Auslenkug des mathematischen Pendels soll berechnet werden:

$$\frac{d^2 x}{dt^2} = -\sin(x)$$

Die rücktreibende Kraft (rechte Seite) soll explizit bzw. mit Taylorreihen bis zur 1 und 3. Ordnung berücksichtigt werden.

```
> deq_left := (D@@2)(x)(t);
```

$$deq\_left := \mathrm{D}^{(2)}(x)(t)$$

Linke Seite

```
> sin_apprx1 := convert(taylor(sin(x), x=0, 2), polynom);
```

$$sin\_apprx1 := x$$

Taylorreihe bis 1. Ordnung

```
> sin_apprx2 := convert(taylor(sin(x), x=0, 4), polynom);
```

$$sin\_apprx2 := x - \frac{1}{6} x^3$$

Taylorreihe bis 3. Ordnung

```
> ini := x(0) = 0, D(x)(0) = 0.8;
```

$$ini := x(0) = 0, \mathrm{D}(x)(0) = 0.8$$

Anfangsbedindungen als Sequenz

```
> sol := dsolve({ deq_left = -sin(x(t)), ini }, x(t),
  type=numeric);
```

$$sol := \mathbf{proc}(x\_rkf45) \ \ldots \ \mathbf{end\ proc};$$

$$\frac{d^2 x}{dt^2} = -\sin(x)$$

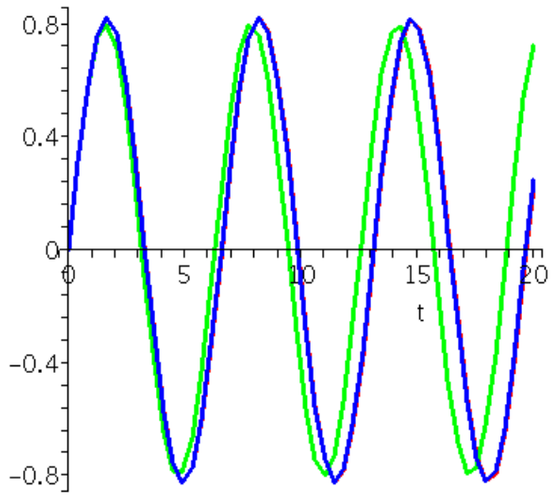
Fallstudie: Mathematischer Pendel (#2)

```
> sol1 := dsolve({ deq_left = -sin_apprx1(t), ini },
  x(t), type=numeric);
                    sol1 := proc(x_rkf45) ... end proc;
> sol2 := dsolve({ deq_left = -sin_apprx2(t), ini },
  x(t), type=numeric);
                    sol2 := proc(x_rkf45) ... end proc;
```

$$\frac{d^2x}{dt^2} = -x$$

$$\frac{d^2x}{dt^2} = -\left(x - \frac{1}{6}x^3\right)$$

```
> with(plots):
Warning, the name changecoords has been redefined
> pl := odeplot(sol, t=0..20, color=blue):
> pl1 := odeplot(sol1, t=0..20, color=green):
> pl2 := odeplot(sol2, t=0..20, color=red):
> display({pl, pl1, pl2});
```



Auswertung:

Einzelne Lösungen (Auslenkung in Abhängigkeit der Zeit) als Plots gespeichert, und gemeinsam angezeigt.

Ergänzende Bemerkungen (#1)

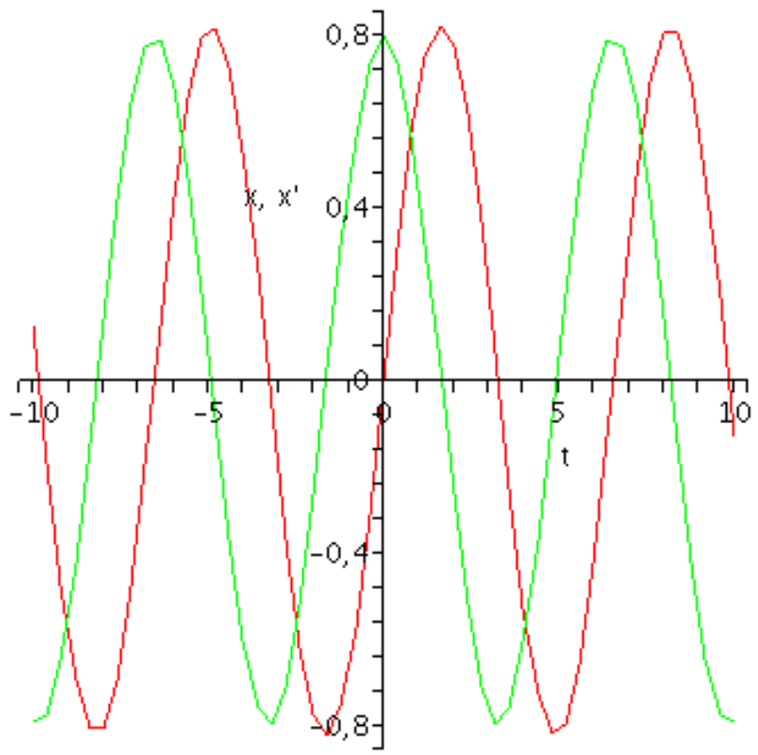
- Numerische Lösung einer Differentialgleichung liefert eine Prozedur als Ergebnis.
- Auswertung dieser Prozedur in bestimmtem Punkt liefert Liste von Ergebnissen:

```
> sol(0.2);
```

$$\left[t = 0.2,\ x(t) = 0.15893682302329897 9,\ \frac{d}{dt} x(t) = 0.7840868681811244 97 \right]$$

- Einzelne Komponenten (*x(t)* bzw. *dx(t)*) lassen sich via **odeplot()** darstellen.

```
> odeplot(sol, [[t, x(t)], [ t, diff(x(t), t)]]);
```



Aber: *Wie kann man x(t) oder dx(t) manipulieren (z.B. aus dx(t) = 0, t numerisch ermitteln)?*

Ergänzende Bemerkungen (#2)

- Mit **output=listprocedure** kann für jede Lösungsfunktion eine getrennte Prozedur erzeugt werden:

```
> sol_list := dsolve({ deq_left = -sin(x(t)), ini}, x(t),
  type=numeric, output=listprocedure);
```

$$sol\_list := \left[t = \mathrm{proc}(t) \ \ldots \ \mathrm{end\ proc;}, x(t) = \mathrm{proc}(t) \ \ldots \ \mathrm{end\ proc;}, \frac{d}{dt} x(t) = \mathrm{proc}(t) \ \ldots \ \mathrm{end\ proc;} \right]$$

Ergebnis: Liste von Prozeduren

- Einzelne Komponenten können via **eval()** aus der Liste „treffsicher“ extrahiert werden:

```
> dx_t := eval(diff(x(t), t), sol_list);
```

$$dx\_t := \mathrm{proc}(t) \ \ldots \ \mathrm{end\ proc;}$$

Extrahiere dx/dt aus der Prozedurliste

```
> dx_t(0.2);
```

$$0.7840868681811244 97$$

dx_t kann als eine normale Prozedur ausgewertet werden.

```
> t0 := fsolve(dx_t(t) = 0.0, t=0..2);
```

$$t0 := 1.639999924$$

Suche numerisch den Zeitpunkt, in dem *dx_t* verschwindet

Zugriff als sol_list[3] ist nicht zu empfehlen, da **Reihenfolge der Prozeduren in der Liste** nicht festgelegt ist!