Numerik

Ingenieurinformatik Teil 2, Sommersemester 2026

David Straub

Gliederung

1. Einführung in Matlab
2. Arbeiten mit Arrays
3. Funktionen und Kontrollstrukturen
4. Analysis
5. Lineare Algebra
6. Differentialgleichungen
7. **Einführung in Simulink** 👈

Fahrplan: Einführung in Simulink

Einheit 1 – Simulink I ⟶ Was ist Simulink? ⟶ Vom Anfangswertproblem (AWP) zum Signalflussplan ⟶ Simulink-Blöcke, Subsysteme und Parameter

Einheit 2 – Heute ⟶ DGL 2. Ordnung ⟶ mehrere Integratoren ⟶ Gedämpfte und angeregte Schwingung (Resonanz) ⟶ Nichtlineare DGL, Solver-Einstellungen, To Workspace

DGL 2. Ordnung in Simulink

Federschwinger (2. Ordnung)

$$m\ddot{x} + kx = 0, \qquad x(0) = x_0, \dot{x}(0) = v_0$$

Auflösen nach der höchsten Ableitung, dann Substitution $y_1 := x, y_2 := \dot{x}$:

$$\dot{y}_1 = y_2, \qquad \dot{y}_2 = -\frac{k}{m}\, y_1$$

Euler-Schritt – zwei Akkumulationszeilen:

	y_1, y_2-Notation	$x, \dot{x}$-Notation
Zeile 1	$y_{1,n+1} = y_{1,n} + \Delta t \cdot \dot{y}_{1,n}$	$x_{n+1} = x_n + \Delta t \cdot \dot{x}_n$
Zeile 2	$y_{2,n+1} = y_{2,n} + \Delta t \cdot \dot{y}_{2,n}$	$\dot{x}_{n+1} = \dot{x}_n + \Delta t \cdot \ddot{x}_n$

Zwei Akkumulationszeilen ⟶ **zwei Integratoren**.

Vom Euler-Schritt zum Signalflussplan

Jede Akkumulationszeile $u_{n+1} = u_n + \Delta t \cdot \dot{u}_n$ wird ein Integrator:

$$\xrightarrow{\ddot{x}} \boxed{\int dt} \xrightarrow{\dot{x}} \boxed{\int dt} \xrightarrow{x}$$

Die höchste Ableitung $\ddot{x} = -\frac{k}{m}\, x$ schließt die Rückkopplung:

$$\xrightarrow{\ddot{x}} \boxed{\int dt} \xrightarrow{\dot{x}} \boxed{\int dt} \xrightarrow{x} \boxed{\times (-k/m)} \xrightarrow{\ddot{x}} \circlearrowleft$$

Vorgehen: DGL n-ter Ordnung → Signalflussplan

Schritt	
1	DGL nach der **höchsten Ableitung** auflösen
2	n **Integratoren** in Reihe zeichnen
3	Ausgänge benennen: $y^{(n-1)}, \ldots, \dot{y}, y$
4	Rechte Seite als Signalpfad aufbauen
5	Ergebnis zurück in den ersten Integratoreingang; Anfangswerte einstellen

✍️ Aufgabe: Federschwinger in Simulink

Implementieren Sie den harmonischen Oszillator $m\ddot{x} + kx = 0$ mit $m = 1$, $k = 1$, $x(0) = 1$, $\dot{x}(0) = 0$ in Simulink.

1. Signalflussplan aufzeichnen (zwei Integratoren, Gain-Block)
2. Modell in Simulink aufbauen
3. Anfangswerte an den Integratoren einstellen
4. $x(t)$ und $\dot{x}(t)$ mit Scope darstellen (Mux verwenden)
5. Ergebnis mit der analytischen Lösung $x(t) = \cos(t)$ vergleichen

Extraaufgabe: k und m als Workspace-Variablen definieren.

Gedämpfte und angeregte Schwingung

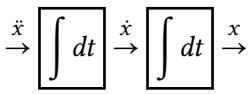
Gedämpfte Schwingung (2. Ordnung)

$$m\ddot{x} + d\,\dot{x} + k\,x = 0$$

Auflösen nach der höchsten Ableitung:

$$\ddot{x} = -\frac{d}{m}\,\dot{x} - \frac{k}{m}\,x$$

DGL 2. Ordnung ⟶ **zwei** Integratoren, aber jetzt zwei Signalpfade zurück:



Rückkopplung von $\dot{x}$ (über $-d/m$) **und** von x (über $-k/m$) in den Summierer.

✍️ Aufgabe: Gedämpfte Schwingung mit Anregung

$$m\ddot{x} + d\,\dot{x} + k\,x = F_0\cos(\omega t)$$

$m = 1$, $d = 0, 1$, $k = 1$, $F_0 = 1$, $x(0) = 0$, $\dot{x}(0) = 0$.

1. Signalflussplan skizzieren (Sine Wave-Block für die Anregung)
2. Modell in Simulink aufbauen
3. Simulieren für $\omega = 0,5$ – wie sieht die Antwort aus?
4. ω auf die Eigenfrequenz $\omega_0 = \sqrt{k/m} = 1$ ändern ⟶ **Resonanz**
5. Alle Parameter als Workspace-Variablen definieren

Nichtlineare DGL

Nichtlineare DGL: kein Problem für Simulink

Bei einer nichtlinearen DGL (z.B. v^2-Term) ändert sich am Signalflussplan **nichts Grundsätzliches** – nur ein zusätzlicher Block wird gebraucht:

Block	Funktion
Product	multipliziert zwei Eingangssignale: $u_1 \cdot u_2$

Um v^2 zu berechnen: beide Eingänge des Product-Blocks an dasselbe Signal v anschließen.

Analytisch unlösbar ⟶ numerisch trivial.

Beispiel: Bremsvorgang mit Luftwiderstand

Ein Flugzeug bremst nach der Landung. Die Geschwindigkeit $v(t)$ folgt aus:

$$\dot{v} = -c_1\, v - c_2\, v^2, \qquad v(0) = 300\,\frac{\mathrm{km}}{\mathrm{h}}$$

$c_1 = 0,4\,\mathrm{s}^{-1}$ (Reifenreibung), $c_2 = 0,004\,\mathrm{m}^{-1}$ (Luftwiderstand)

Signalflussplan:

$$\boxed{\text{Sum}} \xrightarrow{\dot{v}} \boxed{\int dt} \xrightarrow{v} \circlearrowleft$$

Rückkopplung liefert $-c_1 v$ (Gain) und $-c_2 v^2$ (Gain + Product).

✍️ Aufgabe: Bremsvorgang in Simulink

Implementieren Sie das Bremsmodell $\dot{v} = -c_1 v - c_2 v^2$ in Simulink.

Variante A – nur Elementarblöcke (Sum, Gain, Product, Integrator, Scope): - Welche Blöcke werden für v^2 benötigt? - Anfangswert: $v_0 = 300/3,6\,\mathrm{m/s}$

Variante B – Matlab Function Block: - `function v_dot = f(v)` implementieren - Wann wird das Flugzeug langsamer als $5\,\mathrm{m/s}$?

Solver und Datenexport

Solver-Einstellungen

Modeling → Model Settings → Solver

Einstellung	Bedeutung
ode45 (variabel)	Standard; passt Schrittweite automatisch an
ode4 (fest)	Runge-Kutta 4, feste Schrittweite – schnell, vorhersehbar
ode1 (Euler, fest)	anschaulich, aber instabil bei großen Schrittweiten
Max Step Size	begrenzt die maximale Schrittweite (wichtig für glatte Scope-Kurven)

Demonstration: Federschwinger mit ode1 und großer Schrittweite → Lösung divergiert. Dasselbe Modell, kleinere Schrittweite → stabil.

To Workspace: Ergebnisse exportieren

To Workspace-Block schreibt ein Signal als Matlab-Variable in den Workspace.

```
% Nach der Simulation im Command Window:
plot(tout, yout)          % tout und yout automatisch angelegt (Out-Block)
```

Alternativ: **Out**-Block + *Data Import/Export* in den Model Settings
→ `tout` (Zeitvektor) und `yout` (Matrix, eine Spalte pro Ausgang) werden automatisch gespeichert.

So lassen sich Simulink-Ergebnisse direkt in Matlab weiterverarbeiten: FFT, Maxima suchen, mit Messdaten vergleichen.

✍️ Aufgabe: Ergebnisse exportieren und auswerten

Ergänzen Sie das Federschwinger-Modell:

1. **To Workspace**-Block hinzufügen (Signal $x(t)$, Variable `x_sim`)
2. Simulation ausführen
3. In Matlab: analytische Lösung $x_{\text{ana}}(t) = \cos(t)$ berechnen und zusammen mit `x_sim` plotten
4. Maximalen Fehler $\max|\, x_{\text{sim}} - x_{\text{ana}} \,|$ berechnen
5. Schrittweite halbieren – wie ändert sich der Fehler?