Peter Junglas

Wie Mathematik die Modellbildung vereinfacht

Auszug. Im Fach "Simulationstechnik" müssen ingenieuerwissenschaftliche, physikalische und mathematische Kenntnisse miteinander verknüpft werden. Es bietet daher die Gelegenheit, den Studierenden zu zeigen, wie Mathematik ihre konkrete Arbeit erleichtert. Dies wird am Verfahren des Physical Modeling demonstriert, das möglich wurde aufgrund besserer Methoden zur algebraischen Vereinfachung, zur automatischen Indexreduktion und Lösung von differentiell-algebraischen Gleichungen.

Simulation einfacher mechanischer Systeme

Die in ingenieuerwissenschaftlichen Anwendungen bisher meistbenutzten Simulationsprogramme basieren auf der Signalflussmethode. Sie erfordern das Aufstellen der Bewegungsgleichungen, die dann mit Blockdiagrammen nachgebildet werden. Für das Beispielsystem 1 (Abb. 1) können die Studenten die Bewegungsgleichung

$$m\,\ddot{x} + c\,x = 0 \tag{1}$$

in der Regel leicht aufstellen, auch das Beispiel 2, bei dem die Feder durch eine Reihenschaltung zweier Federn ersetzt ist, bereitet ihnen wenig Schwierigkeiten. Größere Probleme macht meistens das Beispiel 3, die Reihenschaltung von Feder und Dämpfer incl. äußerer Anregung (Abb. 2) mit der Gleichung

$$\begin{aligned} \dot{s} &= \frac{1}{b}F(t) \\ x &= s + \frac{1}{c}F(t) \end{aligned} \tag{2}$$

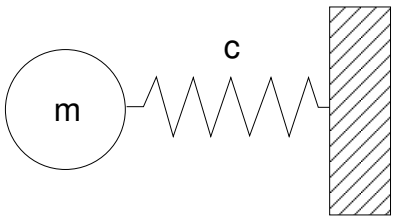

Abbildung 1: Beispiel 1

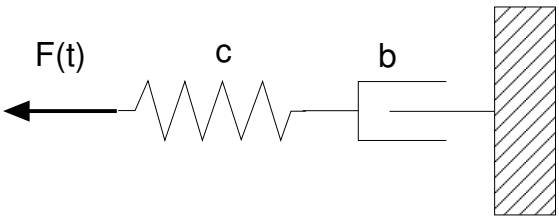

Abbildung 2: Beispiel 3

Sind die Gleichungen erst bekannt, ist das Erstellen eines Modells, etwa mit dem Programm Simulink [10], kein Problem, das Ergebnis für Gleichung (1) zeigt Abb. 3. Das Modell bildet die Differentialgleichung ab, aber von den ursprünglichen Bausteinen (Masse, Feder) ist nichts mehr zu sehen.

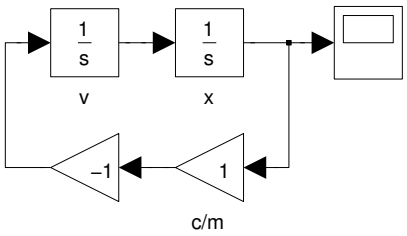

Abbildung 3: Simulink-Modell

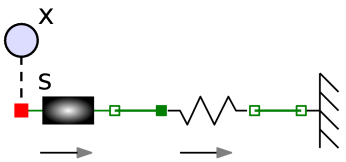

Abbildung 4: MapleSim-Modell 1

Bei neueren, auf "Physical Modeling" basierenden Systemen, z. B. MapleSim [6], verbindet man dagegen einfach Blöcke für die mechanischen Bauteile miteinander (Abb. 4 - 6), die Bewegungsgleichungen werden daraus automatisch hergeleitet. Dazu werden eine Menge neuer mathematischer Methoden benötigt ([2], [3]), die im Folgenden vorgestellt werden.

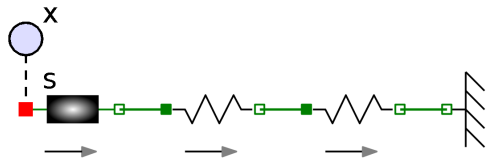

Abbildung 5: MapleSim-Modell 2

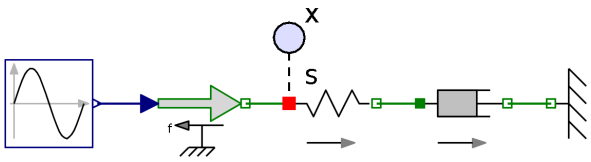

Abbildung 6: MapleSim-Modell 3

Automatische Erzeugung der Bewegungsgleichungen

Zunächst werden für jeden Anschluss an einem Baustein mehrere Variablen definiert, in den Beispielen jeweils eine Kraft f und ein Weg s. Diese Variablen werden in Potenzialvariable (hier: s) und Flussvariable (hier: f) unterschieden. Letztere sind in der Regel Zeitableitungen von erhaltenen Größen, für sie gilt die Konvention, dass sie bei positiven Werten in den Baustein hinein zeigen.
Alle Bausteine definieren Gleichungen zwischen ihren Anschlussgrößen und ggf. weiteren internen Größen. Z. B. hat ein Masse-Block mit den Anschlussvariablen s_1, f_1 und s_2, f_2 eine interne Größe v_m und erzeugt die Gleichungen

$$\begin{aligned} s_1 &= s_2 \\ v_m &= \dot{s}_2 \\ m\dot{v}_m &= f_1 + f_2 \end{aligned}$$

Ein Feder-Block hat keine internen Größen, seine Gleichungen sind

$$\begin{aligned} f_1 + f_2 &= 0 \\ f_2 &= c\,(s_2 - s_1) \end{aligned}$$

Der Einspannungsblock liefert die Gleichung

$$s_1 = 0$$

Darüber hinaus werden für jeden Verbindungsknoten weitere Gleichungen generiert: Alle dort zusammenlaufenden Potenzialvariablen haben denselben Wert, die Flussvariablen addieren sich zu 0 [5].
Für das Beispiel 1 ergeben sich damit 10 Anschlussgrößen und eine interne Größe (Abb. 7), zwischen denen folgende Beziehungen gelten:

$$\begin{aligned} s_1 &= s_2 && \text{(a)} \\ v_m &= \dot{s}_2 && \text{(b)} \\ m\,\dot{v}_m &= f_1 + f_2 && \text{(c)} \\ f_3 + f_4 &= 0 && \text{(d)} \\ f_4 &= c\,(s_4 - s_3) && \text{(e)} \\ s_5 &= 0 && \text{(f)} \end{aligned}$$

$$\begin{aligned} f_1 &= 0 && \text{(g)} \\ s_2 &= s_3 && \text{(h)} \\ f_2 + f_3 &= 0 && \text{(i)} \\ s_4 &= s_5 && \text{(j)} \\ f_4 + f_5 &= 0 && \text{(k)} \end{aligned}$$

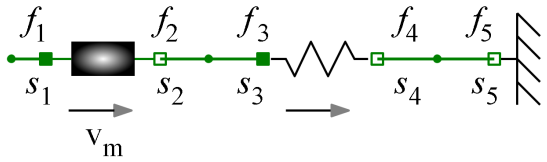

Abbildung 7: Größen im Beispiel 1

Schon für dieses einfache System erhält man 11 Gleichungen, davon zwei Differentialgleichungen (DGLs) für die Zustandsgrößen s_2 und v_m, sowie 9 algebraische Gleichungen. Für ein solches differentiell-algebraisches System (DAE-System) existiert kein universeller Solver, stattdessen muss es in ein einfacheres System überführt werden. Dies ist hier per Hand einfach, aber bei wesentlich größeren Systemen wird dazu ein hinreichend schneller Algorithmus benötigt.

Sortierung der Gleichungen mit dem Tarjan-Algorithmus

Zur Umformung des Systems wird der Tarjan-Algorithmus [9] verwendet, eigentlich ein Färbe-Verfahren zur Auffindung von Zusammenhangskomponenten in Graphen. Im ersten Schritt wird aus dem Gleichungssystem ein

bipartiter Graph abgeleitet, bei dem links die Gleichungen stehen, rechts die Variablen, bei Zustandsgrößen nur deren Ableitungen. Für jede Variable, die in einer Gleichung vorkommt, wird eine entsprechende Kante erzeugt. Das Ergebnis für Beispiel 1 zeigt Abb. 8.

Nun sollen die Gleichungen numeriert und alle Kanten rot oder blau gefärbt werden, wobei von jeder Variablen bzw. jeder Gleichung genau eine rote Kante ausgeht. Dazu sucht man zunächst Gleichungen, von denen genau eine ungefärbte Kante ausgeht. Eine solche Gleichung bekommt die kleinste unvergebene Nummer, die Kante wird rot gefärbt, alle anderen Kanten der zugehörigen Variablen blau. Nach einem Durchgang durch die Gleichungen sucht man unter den Variablen solche mit genau einer ungefärbten Kante. Diese Kante wird rot gefärbt; die zugehörige Gleichung bekommt die größte freie Nummer, die andere Kanten dieser Gleichung werden blau. Nach dem Durchgang durch die Variablen beginnt man wieder bei den Gleichungen und iteriert solange, bis der Graph komplett gefärbt ist oder der Algorithmus abbricht, weil es keine einzelnen ungefärbten Kanten mehr gibt. Das Ergebnis im Beispiel 1 zeigt Abb. 9, wobei rote Kanten gestrichelt, blaue gepunktet dargestellt sind.

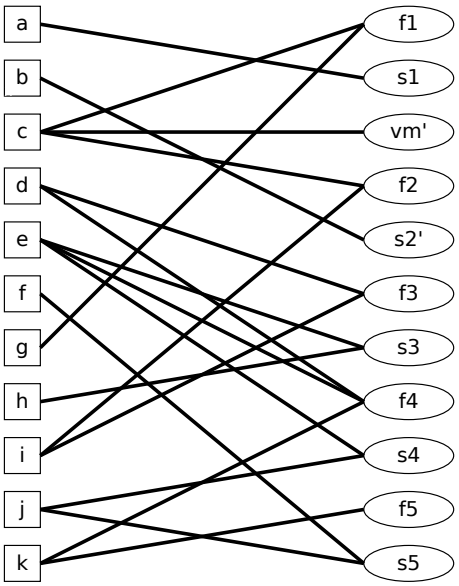

Abbildung 8: Graph für Beispiel 1

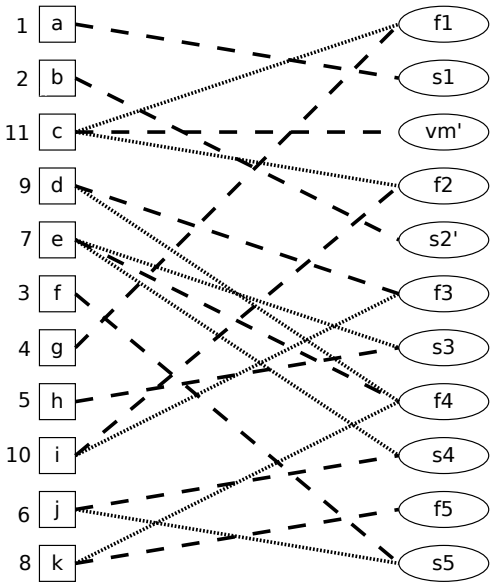

Abbildung 9: gefärbter Graph

Schließlich werden die Gleichungen gemäß ihrer Numerierung umsortiert und jede Gleichung nach der Variablen aufgelöst, die zur roten Kante gehört. Aus den mathematischen Gleichungen werden dabei Zuweisungen im Programm:

$$
\begin{array}{lcll}
s_1 & := & s_2 & \text{(a)} \\
\dot{s}_2 & := & v_m & \text{(b)} \\
s_5 & := & 0 & \text{(f)} \\
f_1 & := & 0 & \text{(g)} \\
s_3 & := & s_2 & \text{(h)} \\
s_4 & := & s_5 & \text{(j)}
\end{array}
\qquad
\begin{array}{lcll}
f_4 & := & c\,(s_4 - s_3) & \text{(e)} \\
f_5 & := & -f_4 & \text{(k)} \\
f_3 & := & -f_4 & \text{(d)} \\
f_2 & := & -f_3 & \text{(i)} \\
\dot{v}_m & := & \frac{1}{m}(f_1 + f_2) & \text{(c)}
\end{array}
$$

Bedenkt man noch, dass für die Zustandsgrößen s_2 und v_m Anfangswerte gegeben sind, erkennt man, dass das umgeordnete System eine gewöhnliche DGL darstellt, die von entsprechenden Solvern leicht gelöst werden kann.

Algebraische Schleifen

Stellt man für das Beispiel 2 (Masse und zwei Federn in Reihe) alle Gleichungen auf, die durch die Bausteine und Verbindungsknoten gegeben sind, erhält man 15 Gleichungen für 2 Zustandsgrößen und 13 algebraische Variablen. Wendet man darauf den Tarjan-Algorithmus an, erhält man den Graphen in Abb. 10.

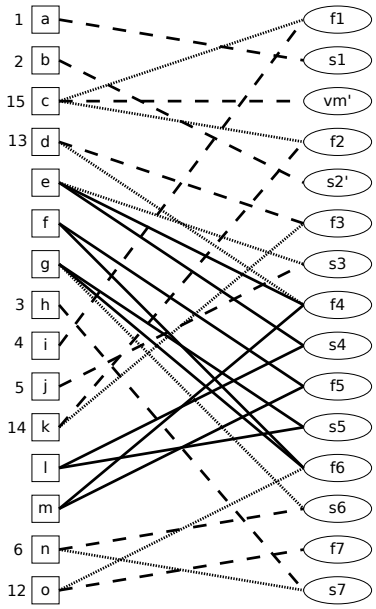

Abbildung 10: Graph für Beispiel 2

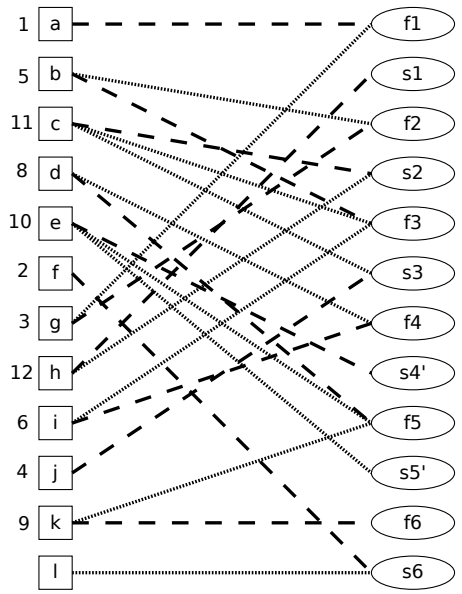

Abbildung 11: Graph für Beispiel 3

Hier ist der Algorithmus abgebrochen, ohne dass alle Kanten gefärbt wurden; es bleiben 5 Gleichungen und 5 Variablen übrig, die alle mehrere ungefärbte Kanten haben. Sortiert man die bereits numerierten Gleichungen und löst sie nach den zugehörigen Variablen auf, ergibt sich:

$$
\begin{aligned}
s_1 &:= s_2 \\
\dot{s}_2 &:= v_m \\
s_7 &:= 0 \\
f_1 &:= 0 \\
s_3 &:= s_2 \\
s_6 &:= s_7
\end{aligned}
\qquad
\begin{aligned}
f_4 - cs_4 &= -cs_3 \quad \text{(A)} \\
f_5 + f_6 &= 0 \quad \text{(B)} \\
f_6 + cs_5 &= cs_6 \quad \text{(C)} \\
s_4 - s_5 &= 0 \quad \text{(D)} \\
f_4 + f_5 &= 0 \quad \text{(E)}
\end{aligned}
\qquad
\begin{aligned}
f_7 &:= f_6 \\
f_3 &:= -f_4 \\
f_2 &:= -f_3 \\
\dot{v}_m &:= \frac{1}{m}(f_1 + f_2)
\end{aligned}
$$

Innerhalb der Zuweisungskette gibt es nun ein System aus 5 Gleichungen mit 5 Unbekannten, das im allgemeinen (nichtlinearen) Fall etwa mit einem Newton-Solver gelöst werden kann. In der Simulationstechnik spricht man von einer "algebraische Schleife", mathematisch handelt es sich um ein DAE-System vom Index 1. Es lässt sich von speziellern Solvern (etwa DASSL [1]) gut lösen.
In praktischen Anwendungen werden die algebraischen Schleifen oft sehr groß und können tausende Gleichungen umfassen. Normale Newton-Solver werden dann sehr langsam. Daher sind zur Zerlegung riesiger Schleifen in kleinere mit sehr viel weniger Variablen spezielle Methoden entwickelt worden ("Tearing"-Verfahren [4]).

Strukturell singuläre Systeme

Das Feder-Dämpfer-System (Beispiel 3) führt auf eine neue Art von Problem. Das Gleichungssystem enthält 12 Variablen, darunter 2 Zustandsgrößen:

$$
\begin{aligned}
f_1 &= -F(t) && \text{(a)} \\
f_2 + f_3 &= 0 && \text{(b)} \\
f_3 &= c\,(s_3 - s_2) && \text{(c)} \\
f_4 + f_5 &= 0 && \text{(d)} \\
f_5 &= b\,(\dot{s}_5 - \dot{s}_4) && \text{(e)} \\
s_6 &= 0 && \text{(f)}
\end{aligned}
\qquad
\begin{aligned}
f_1 + f_2 &= 0 && \text{(g)} \\
s_1 &= s_2 && \text{(h)} \\
f_3 + f_4 &= 0 && \text{(i)} \\
s_3 &= s_4 && \text{(j)} \\
f_5 + f_6 &= 0 && \text{(k)} \\
s_5 &= s_6 && \text{(l)}
\end{aligned}
$$

Der Tarjan-Algorithmus ergibt, dass von Gleichung (l) nur eine blaue Kante ausgeht (Abb. 11), sie liefert keine Variable. Eine solche Gleichung heißt "Nebenbedingung" ("Constraint"). Solche Systeme heißen "strukturell singulär", es handelt sich um DAEs mit Index > 1 (hier: 2), für die keine direkten Solver existieren. Physikalisch ist die Ursache im Beispiel klar: Die Gleichungen enthalten zwei Zustandsgrößen, das Modell hat aber nur einen Freiheitsgrad.

Die Grundidee zur Lösung besteht darin, durch Ableiten "geeigneter" Gleichungen den Index des Systems auf 1 zu reduzieren. Ein systematisches Verfahren stammt von Pantelides [8]: Zusätzlich zu den Constraints werden die im Graphen mit diesen zusammenhängenden Gleichungen bestimmt und alle durch ihre Ableitungen ersetzt. Im Beispiel heißt dies, dass (f) und (l) ausgetauscht werden gegen

$$\begin{aligned} \dot{s}_6 &= 0 \quad (\mathrm{f}') \\ \dot{s}_5 &= \dot{s}_6 \quad (\mathrm{l}') \end{aligned}$$

Dies liefert die neue Zustandsgröße s_6 und damit einen veränderten Graphen, der sich mit Tarjan problemlos färben lässt. Bei höherem Index muss man die Methode entsprechend oft iterieren.
Die neuen Gleichungen werfen allerdings auch neue Probleme auf: Da die Constraints (hier: $s_6 = 0$) ersetzt wurden, sind sie nur erfüllt, wenn entsprechende Anfangsbedingungen gewählt werden! Darüberhinaus sind aufwändige numerische Tricks nötig, um die Constraints im Laufe der Rechnung wenigstens näherungsweise zu erhalten.
Mit der Methode der Dummy-Ableitungen [7] lässt sich diese Schwierigkeit auf elegante Weise umgehen: Die neuen Gleichungen ersetzen die alten nicht, sondern werden hinzugefügt. Daduch gibt es jetzt mehr Gleichungen als Unbekannte (hier: 2). Nun werden von den Zustandsgrößen einige ausgewählt (hier: s_5, s_6) und nicht mehr als Zustandsgrößen aufgefasst. Konkret heißt dies, dass etwa s_5 und $ds_5 \equiv \dot{s}_5$ als unabhängige Größen verwendet werden. Die Variable ds_5 heißt "Dummy-Ableitung", ihr Zusammenhang zu s_5 ergibt sich nicht a priori, sondern vermöge der entsprechenden Gleichungen. Die Constraints bleiben bestehen, damit sind sie automatisch erhalten, auch das Problem der Anfangswerte ergibt sich nebenbei.

Schlussfolgerungen

Dass die Mathematik tatsächlich praktische Erleichterungen für die Ingenieure liefert, ist den Studierenden häufig nicht wirklich bewusst. Hier können sie es direkt sehen: Das Simulieren mit Physical Modeling stellt in vielen Fällen eine deutliche Vereinfachung gegenüber den herkömmlichen Methoden dar.
In diesem Zusammenhang stellt sich die Frage: Welche Mathematik braucht ein Maschinenbauer? Angesichts neuer Verfahren müssen die Mathematik-Curricula immer wieder überdacht werden.

Insbesondere die zunehmende Bedeutung der Simulationstechnik führte in den letzten Jahren zur Einführung der Numerischen Mathematik in die Grundausbildung. Dort werden wichtige Algorithmen wie Newtonverfahren oder DGL-Solver behandelt. Die hier vorgestellten speziellen Themen wie Graphen-Algorithmen oder DAEs gehören aber eher in die Spezialvorlesungen, wo sie im Anwendungskontext beschrieben werden können.

Literaturverzeichnis

[1] **Brenan, K.E.; Campbell, S.L.; Petzold, L.R.**: *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations.* SIAM, Philadelphia (1996).

[2] **Cellier, F. E.; Elmqvist, H.**: *Automated formula manipulation supports object-oriented continuous system modeling.* IEEE Control Systems, 2, **13**, 28-38 (1993).

[3] **Cellier, F. E.; Kofman, E.**: *Continuous System Simulation.* Springer-Verlag, New York (2010).

[4] **Elmqvist, H.; Otter, M.**: *Methods for Tearing Systems of Equations in Object-Oriented Modeling.* In: Proc. European Simulation Multiconference, Barcelona, Spain, 326-332 (1994).

[5] **Fritzson, P.**: *Principles of Object-Oriented Modeling and Simulation with Modelica 2.1.* Wiley & Son, New York (2004).

[6] **Maplesoft**. http://www.maplesoft.com/products/maplesim/

[7] **Mattsson, S. E.; Söderlind, G.**: *Index Reduction in Differential-Algebraic Equations Using Dummy Derivatives.* SIAM J. Sci. Comput., 3, **14**, 677-692 (1993).

[8] **Pantelides, C. C.**: *The Consistent Initialization of Differential-Algebraic Systems.* SIAM J. Sci. Stat. Comput., 2, **9**, 213-231 (1988).

[9] **Tarjan, R.**: *Depth-first Search and Linear Graph Algorithms.* SIAM J. Comput., 2, **1**, 146-160 (1972).

[10] **The MathWorks, Inc.** http://www.mathworks.com/products/simulink/index.html

Autor

Prof. Dr. rer. nat. Peter Junglas

Private Fachhochschule für Wirtschaft und Technik Vechta/Diepholz/Oldenburg

Schlesierstraße 13a

D-49356 Diepholz

E-Mail: peter@peter-junglas.de