Edition
Harri
Deutsch

Praxis der Simulationstechnik

Eine Einführung in signal- und objektorientierte Methoden

von

Peter Junglas

VERLAG EUROPA-LEHRMITTEL · Nourney, Vollmer GmbH & Co. KG
Düsselberger Straße 23 · 42781 Haan-Gruiten

Europa-Nr.: 57761

Autor:
Dr. Peter Junglas ist Professor für Physik und Informatik im Fachbereich Maschinenbau der Privaten Fachhochschule für Wirtschaft und Technik (FHWT) in Vechta/Diepholz/Oldenburg.

1. Auflage

Druck 5 4 3 2 1

ISBN 978-3-8085-5776-1

Satz: Satzherstellung Dr. Naake, 09618 Brand-Erbisdorf
Umschlaggestaltung: braunwerbeagentur, 42477 Radevormwald
Druck: Medienhaus Plump GmbH, 53619 Rheinbreitbach

Gewidmet meinem verehrten Lehrer Prof. Dr. Detlev Buchholz
zu seinem 70. Geburtstag

Vorwort

Simulationstechnik ist für angehende oder fertige Ingenieure nicht nur ein wichtiges, sondern auch ein schwieriges Thema: Neben einem grundlegenden Verständnis der Anwendungsgebiete – häufig gleich mehreren auf einmal – benötigt man gute Kenntnisse der verwendeten Software, meistens Programmiererfahrung und schließlich noch mathematisches Hintergrundwissen.

Wie soll man das alles lernen? Die Antwort dieses Buches lautet: anhand von vielen praktischen, systematisch aufeinander aufbauenden Beispielen. Jedes davon enthält die obigen Aspekte zumindest in rudimentärer Form, beleuchtet ein anderes Thema, vermittelt neue Methoden oder erweitert die Programmier-Trickkiste. Im Laufe von über 150 ausführlich vorgestellten Modellen aus vielen Bereichen des Maschinenbaus, der Mechatronik oder verwandter Bereiche erhält man so einen immer umfassender werdenden Einblick in die Methoden der Modellierung und Simulation.

Aber Nachmachen allein genügt nicht, man muss das neue Wissen auch praktisch ausprobieren. Dazu dienen 66 leichte bis anspruchsvolle Aufgaben, die helfen, seine Kenntnisse zu überprüfen und die geistigen Werkzeuge immer wieder zu schärfen. Zur Kontrolle, aber auch als Anregung für weitergehende Untersuchungen, sind alle Aufgaben mit vollständigen Lösungen versehen.

Für das praktische Ausprobieren benötigt man aber auch die entsprechende Software. Zum Einsatz kommen hier die weit verbreiteten Programme Simulink und MapleSim. Wer darauf keinen Zugriff hat und die entsprechenden Ausgaben scheut, kann stattdessen auch die OpenSource-Alternativen Xcos bzw. OpenModelica verwenden, auf deren Besonderheiten jeweils ausführlich eingegangen wird. Alle besprochenen Beispiel- und Lösungsmodelle findet man jeweils für beide Programme auf der Homepage des Buches.

Ein Wort zur Einschränkung: Unter »Simulationstechnik« versteht man häufig auch Berechnungen mit Finite-Elemente- oder Strömungsprogrammen, also räumlich kontinuierliche Fragestellungen. Diese benötigen aufwendige mathematische Methoden und teure Spezialsoftware. Hier soll es dagegen ausschließlich um Systeme mit endlich vielen Freiheitsgraden gehen, Massen werden zu Punkten oder starren Körpern. Die Schwierigkeit liegt dabei in der Kopplung der Komponenten, nicht in der Komplexität einzelner Grundgleichungen.

Hat man die ersten Klippen überwunden, ist der Spaßfaktor in der Regel nicht zu verachten, schließlich kann man »Gott spielen« in seiner eigenen Kunstwelt. Aber Obacht: Das Ziel ist das Verständnis realer Systeme, und der Weg vom Modell bis zur Realität ist oft weiter als man denkt. Oder in den Worten Schillers:

> Leicht beieinander wohnen die Gedanken, doch hart im Raume stoßen sich die Sachen. (Wallensteins Tod II, 2)

Danksagungen

Basis des Buchs ist die Vorlesung »Simulationstechnik«, die der Autor seit zehn Jahren an der Privaten Fachhochschule für Wirtschaft und Technik (FHWT) in Diepholz mit Studierenden des Maschinenbaus und des Wirtschaftsingenieurwesens veranstaltet. Sie bekamen immer wieder verschiedene Versionen des Skripts, der Beispiele und der Hausaufgaben und haben durch ihre Kritik dazu beigetragen, manche Schwächen auszumerzen. Ihnen allen danke ich herzlich.

Mein besonderer Dank gilt den folgenden ehemaligen Studierenden und jetzigen Ingenieuren, deren hervorragenden Hausarbeiten manches Beispiel zu verdanken ist: Das Kapitel 11 stützt sich wesentlich auf die Arbeiten von Mike David Breng, Christian Buschendorf, Thomas David, Nina Hartmann, Sören Hilbers, Hannes Mählmann, Achim Sauerbrey, Tony Schwarz, Marco Trautmann, Jürgen Volkers und Julia Witte, die in vier Gruppen im Verlauf einiger Jahre die Tücken der »Magnet-Schwebeanlage« gebändigt haben. Einige Probleme des Triebstrangs in Kap. 16 waren sehr vertrackt, aber nicht zu schwer für Michael Feldhaus, Sören Lübben und Christian Wittenbrink. Hermann Siepker zeigte in seiner Thesis kritisches Denken und wies den Autor auf ein konzeptionelles Problem mit den Beispielen zum Wärmetauscher hin. Schließlich haben Viktor Dick und Konstantin Rempel etliche der Aufgaben von Kap. 6 - 10 gelöst und dem Autor dabei zu einer besseren Einschätzung verholfen, wie viel Zeit (gute!) Studierende für die Aufgaben brauchen.

Für das sorgfältige Korrekturlesen eines für sie fachfremden dicken Manuskripts danke ich Jana Schlögl.

Mein Dank geht auch an Thomas Richard von Maplesoft Europe GmbH. Er hat das Lamentieren des Autors nicht nur stets freundlich ertragen, sondern auch häufig genug für schnelle Abhilfe gesorgt. Das gleiche gilt für die Entwicklerteams von Scilab/Xcos und OpenModelica, die mit großem Engagement an der Weiterentwicklung ihrer Software arbeiten.

Klaus Horn vom Verlag Europa-Lehrmittel zu danken, ist mir eine besondere Freude. Auch dieses Buchprojekt hat er wie immer freundlich und kompetent begleitet und sogar eine Verdopplung des ursprünglich avisierten Umfangs geduldig hingenommen.

Schließlich danke ich meiner Frau und meinen Kindern, die über drei Jahre hinweg ertragen haben, dass ich viel zu wenig Zeit für sie hatte – und meinen Kopf allzu oft woanders.

Inhaltsverzeichnis

1 Einleitung

1.1 Modellierung und Simulation

Simulieren heißt Experimentieren mit Modellen von interessierenden Systemen. [1)] Dabei geht es primär darum, ein besseres Verständnis für das Systemverhalten zu gewinnen. Während der Naturwissenschaftler damit vor allem seine Theorien überprüft, ist der Ingenieur eher am Zweck eines – meistens technischen – Systems ausgerichtet: Hat das System die gewünschte Funktion? Kann man diese auch mit einfacheren Mitteln erreichen?

Der erste Schritt ist die **Modellierung**, also die Entwicklung von Modellen, die die Komplexität des Ausgangssystems möglichst weitgehend reduzieren und trotzdem noch für die betrachtete Fragestellung relevante Aussagen machen können. Im Rahmen dieses Buches werden dies ausschließlich mathematisch formulierte Modelle sein. Wir werden vor allem **dynamische Systeme** betrachten, die als Funktionen der Zeit beschrieben werden können. Ihr Verhalten ist häufig festgelegt, wenn man die Werte einiger unabhängiger Variablen zu einem festen Zeitpunkt kennt; diese heißen **Zustandsgrößen**. Wenn die zwischen ihnen bestehenden Beziehungen mithilfe zugrunde liegender physikalischer Prinzipien gewonnen werden können, haben die resultierenden Modelle meistens die Form von Differenzialgleichungen.

Die entscheidende Frage lautet nun: Wie komme ich zum Modell? Die wichtigste Grundregel ist, so einfach wie möglich zu beginnen und die Komplexität nur dort zu erhöhen, wo es nötig ist. Beispielsweise wird man die Modellierung eines Fahrzeug-Federbeins mit einem einfachen Masse-Feder-System beginnen und erst dann weitere Bauelemente oder nichtlineares Verhalten hinzufügen, wenn es die Problemstellung erfordert. Die zugehörigen Differenzialgleichungen gewinnt man mit Standardmethoden der Mechanik – oder entnimmt sie der Literatur.

Bei komplexen Systemen ist »Teile und herrsche« das Grundprinzip: Entweder beginnt man mit einfachen Grundelementen, die man einzeln testet und dann zum Gesamtmodell zusammensetzt (**Bottom-up-Ansatz**) oder man zerlegt das System direkt in Grundmodule, deren Verhalten man anschließend verfeinert (**Top-down-Ansatz**). In beiden Fällen ist die Kenntnis möglichst vieler Basismodelle und verschiedener Techniken hilfreich, wie sie die zahlreichen Beispiele dieses Buches vermitteln.

Im zweiten Schritt, der eigentlichen **Simulation**, werden mit dem Modell dann Experimente durchgeführt, die Aufschluss über die gewünschten Aspekte des Systems geben können. Dazu müssen die aufgestellten Modellgleichungen mit Hilfe von Methoden der

[1)] Diese Formulierung ist angelehnt an [15]. Fast jedes Wort dieses Satzes ist es wert, genauer betrachtet zu werden, vgl. dazu etwa die unterschiedlichen Ansätze in [15], [53] und [7].

numerischen Mathematik gelöst werden. Neben den grundlegenden Differenzialgleichungen treten dabei oft noch weitere Schwierigkeiten auf: Mechanische Nebenbedingungen führen durch zusätzliche algebraische Gleichungen zu so genannten **DAE-Systemen**, Unstetigkeiten oder diskrete Ereignisse verkomplizieren die Lösung, Zufallskomponenten reduzieren die Aussagekraft einer einzelnen Berechnung.

Zum Glück gibt es inzwischen eine reichhaltige Auswahl an Lösungsverfahren (den **Solvern**), die für die meisten Probleme zu zufrieden stellenden Ergebnissen führen und in die gängigen Simulationsprogramme integriert sind. An der Frage, wie tief das Verständnis dieser Methoden sein muss, um die Ergebnisse und die Einschränkungen solcher Programme beurteilen zu können, scheiden sich die Geister – und die Lehrbücher. Gemäß dem Titel dieses Buches wird hier ein pragmatischer Ansatz vertreten und die Mathematik als Hilfswissenschaft betrachtet, die man weitgehend den Mathematikern überlassen darf. Trotzdem sollten sich Ingenieure nicht auf reine »Maschinenbediener« der Simulationstechnik reduzieren! Daher werden einige Abschnitte sich auch den verwendeten Lösungsverfahren widmen, ohne allerdings in die Tiefe vorzudringen. Auf diese Weise sollen die oft als »Black-Box« eingesetzten Programme zumindest etwas durchsichtig werden.

Dass Modellierung und Simulation nicht voneinander unabhängig sind, ist klar: Mathematische Fortschritte in der Simulation führen zu Änderungen im Modellieren. Dies wird besonders deutlich an den jüngsten Fortschritten der Simulationstechnik: Das bisherige Standardmodell der (gewöhnlichen) Differenzialgleichung führte zu signalorientierten Modellen, wie sie immer noch häufig verwendet und im ersten Teil des Buchs ausführlich beschrieben werden. Erst die Entwicklung von DAE-Solvern, die auf algebra-basierten Vorverarbeitungen beruhen, ermöglicht nun den Einsatz von objektorientierten Verfahren, die u. a. mechanische, elektrische oder hydraulische Grundbausteine zur Verfügung stellen und Thema des zweiten Teils sind.

1.2 Arbeiten mit dem Buch

Da man auch die Simulationstechnik am besten durch praktisches Üben erlernt, werden in diesem Buch alle Verfahren und theoretischen Hintergründe immer ausgehend von konkreten Beispielen entwickelt. Der größte Lernerfolg wird sich daher einstellen, wenn man beim Lesen die vorgestellten Modelle mithilfe der entsprechenden Software gleich nachbaut und alle Ergebnisse selbst überprüft. Erst dabei wird man auch die zahllosen Kniffe kennen lernen, die man sich mit zunehmender Erfahrung im Umgang mit den Programmen erarbeitet. Zur Kontrolle oder bei auftauchenden Schwierigkeiten kann man sich fertige Versionen von der Download-Seite des Buches herunterladen unter

`http://www.peter-junglas.de/fh/publications/simulation/index.html`

Aufgrund der wachsenden Komplexität der Beispiele erfordert ihr Aufbau in späteren Kapiteln einen recht hohen Zeitaufwand, den man durch Rückgriff auf die fertigen Modelle reduzieren kann. Allerdings bleibt dabei das Verständnis für die vielen Details, die in der Praxis das Leben erschweren, auf der Strecke – beim Lernen gibt es keine Abkürzung! Auf der anderen Seite will man aber auch nicht immer alles sofort

wissen. Gegen ein schnelleres Vorgehen durch einfaches Lesen – entspannt auf dem Sofa statt verkrampft am Computer – spricht dann nichts. Damit dies funktioniert, enthält das Buch zahllose Abbildungen, die die Modelle und ihren Aufbau zeigen sowie Simulationsergebnisse grafisch veranschaulichen.

Die Beispiele stammen überwiegend aus Bereichen, die im Grundstudium eines Maschinenbau-, Mechatronik- oder verwandten Studiums vermittelt werden, etwa der Mechanik, Elektrotechnik, Regelungstechnik oder Thermodynamik. Entsprechende Kenntnisse sind für eine Einordnung oder ein tieferes Verständnis nützlich, einige ergänzende Hintergrundinformationen findet man im Anhang. Ist man aber bereit, die angegebenen Beziehungen hinzunehmen, kann man umgekehrt die Modelle verwenden, um erste Kenntnisse dieser Gebiete »experimentell« zu erwerben. Dieses Vorgehen ist für den praktischen Simulationstechniker nicht ungewöhnlich, der gelegentlich Phänomene in seinen Modellen berücksichtigen muss, deren zugrunde liegende Theorie er sich dabei im Schnellverfahren erst aneignet.

Vertiefte mathematische Kenntnisse werden hier nicht vorausgesetzt, stattdessen werden die Grundideen der benutzten Verfahren in entsprechenden Kapiteln vorgestellt. Trotzdem ist es nützlich, sich darüber hinaus etwas mehr mit numerischer Mathematik zu befassen, um die Vor- und Nachteile verschiedener Solver und ihre Anwendbarkeit bei schwierigeren Problemen einschätzen zu können. Im übrigen gilt auch nach fast 90 Jahren noch die Erkenntnis von Vannevar Bush:

> Engineering can proceed no faster than the mathematical analysis on which it is based (zitiert nach [6]).

Auch Programmiererfahrung wird grundsätzlich nicht erwartet. Wer darüber verfügt, wird allerdings bald feststellen, dass der grafische Aufbau komplexer Modelle ähnlichen Regeln gehorcht wie das Erstellen von Programmen. Grundlegende Kenntnisse von Matlab sind jedoch schon zu Beginn hilfreich. Im weiteren Verlauf werden immer wieder einige Matlab-Zeilen den Modellen auf die Sprünge helfen, bis schließlich im Kap. 10 ganze Matlab-Programme eingefügt werden. Wer sich nur schnell ein paar Grundkenntnisse besorgen will, findet im Internet reichlich Einführungsskripte (z. B. das des Autors [44]); eine systematische und tiefgehende Einführung bietet [3]. Wer möchte, kann Matlab auch gleich verwenden, um das »normale« Programmieren zu erlernen, etwa mit [97].

Am Ende eines Abschnitts befinden sich Aufgaben, mit deren Hilfe die vorgestellten Methoden an weiteren Beispielen geübt werden können. Vor allem am Anfang sollen sie helfen, sich mit der verwendeten Software vertraut zu machen und weitere Grundfunktionen auszuprobieren. Dazu kommen zunehmend komplexere Projekte, die auch zum eigenen Experimentieren einladen. Gelegentlich sollen theoretische Verfahren ausprobiert oder angewendet werden. Die Bearbeitung möglichst vieler Aufgaben verbessert den Lernerfolg deutlich – eine rein theoretische Auseinandersetzung mit dem Stoff bereitet nicht auf die vielen kleinen Fallstricke des Simulations-Alltags vor!

Zu allen Aufgaben gibt es im Anhang ausführliche Lösungen, die zugehörigen Modelle finden sich im Download-Bereich. Dabei werden gelegentlich auch neue Konstruktionen vorgestellt oder es wird auf besondere praktische Schwierigkeiten hingewiesen. Die

Lektüre lohnt sich also, auch wenn man eine Aufgabe aus Zeitgründen nicht bearbeitet haben sollte. Natürlich kann man – gerade bei den komplexeren Beispielen – auch zu ganz anders gearteten Lösungen kommen. Da hier die Überprüfung durch die Praxis fehlt, sind vor allem die verwendeten Beziehungen, der korrekte Aufbau des Modells und die Plausibilität der Ergebnisse zu kontrollieren.

Grundsätzlich bauen die Kapitel des Buches aufeinander auf und sollten in der richtigen Reihenfolge gelesen werden. Hat man aber schon Grundkenntnisse und Erfahrung im Umgang mit den verwendeten Programmen, kann man je nach Interesse auch einige Kapitel überspringen oder in anderer Reihenfolge lesen. Dass man sich über die Eigenschaften unbekannter Blöcke oder Programmkonstruktionen notfalls mit dem entsprechenden Handbuch oder der Online-Hilfe informiert, wird sowieso erwartet. Die folgende Kurzübersicht soll dabei helfen, ggf. seinen eigenen Weg durch das Buch zu finden.

Der 1. Teil des Buches befasst sich mit signalorientierten Methoden. Die Modelle bestehen hier aus Blöcken, die definierte Ein- und Ausgänge haben und jeweils eine spezifische Operation implementieren. Sie werden mit Leitungen verbunden, auf denen Signale von einem Ausgang zu ggf. mehreren Eingängen geleitet werden. Signale können einfache Zahlen sein (jeweils pro Zeitschritt der Simulation ein Wert) oder mehrere, die zu einem Vektor oder gar einer Matrix zusammengefasst sind.

Thema des 2. Teils sind objektorientierte Methoden, die auch als »physikalische Modellierung« bezeichnet werden. Blöcke entsprechen hier nicht Berechnungsvorschriften, sondern in der Regel konkreten Objekten des Anwendungsbereichs. Sie haben verschiedenartige Anschlüsse, die meistens physikalische Kopplungen zwischen Bauelementen nachbilden, und werden mit Leitungen verbunden, die entsprechende Informationen verteilen. Die Richtung des Informationsflusses ist hier nicht a priori festgelegt, sondern ergibt sich erst im Kontext des Gesamtmodells. Komplexere Modelle enthalten oft Komponenten, die signalorientiert aufgebaut sind; daher sind Grundkenntnisse aus Teil 1 erforderlich.

Die Kapitel im einzelnen:

- Kap. 2, 3, 5 und 6.1 vermitteln die Grundlagen der signalorientierten Modellierung und den Umgang mit der Software.
- Kap. 4 erklärt den mathematischen Hintergund, der zum Grundverständnis der Solver nötig ist.
- Kap. 6.2/6.3 zeigt den Aufbau eines komplexen Beispiels.
- Kap. 7 untersucht, wie sich das Verhalten im Frequenzraum modellieren lässt.
- Kap. 8 und 9 enthalten wichtige, aber etwas fortgeschrittenere Ergänzungen.
- Kap. 10 beschreibt die Programmierung eigener Blöcke mit Matlab. Dafür sind gute Matlab-Kenntnisse nötig.
- Kap. 11 führt an einem Beispiel vor, wie Hardware in die Simulation integriert werden kann. Da dem Leser entsprechende Geräte vermutlich nicht zur Verfügung stehen, lassen sich einige der hier vorgestellten Modelle nicht am eigenen Rechner nachvollziehen.

- Kap. 12 beschreibt eine Methodik, die vor allem in ökologischen oder ökonomischen Simulationen verwendet wird.
- Kap. 13 stellt eine abstrakte Methode vor, die als Vorgänger der physikalischen Modellierung betrachtet werden kann, aber immer noch nutzbringend eingesetzt wird.
- Kap. 14 führt in das grundlegende Arbeiten bei objektorientierter Modellierung ein.
- Kap. 15 erklärt die Verfahren, die zur Simulation solcher Modelle erforderlich sind.
- Kap. 16 und 17 besprechen komplexere Beispiele aus der Fahrzeugtechnik bzw. der Mechanik.
- Kap. 18 zeigt exemplarisch, wie man mithilfe der Sprache Modelica einen neuen physikalischen Anwendungsbereich erschließt. Zum Verständnis sind Grundkenntnisse der Thermodynamik erforderlich.

1.3 Verwendete Programme

Für beide vorgestellten Modellierungsverfahren gibt es kommerzielle Software, die in der Industrie und in der Lehre erfolgreich eingesetzt wird. Es handelt sich dabei um umfangreiche Pakete mit einer ausgefeilten Benutzerführung, gut bestückten Sammlungen von Bausteinen und einer ausführlichen Dokumentation. Zwar werden davon jeweils günstige Studentenversionen vertrieben, dennoch bleibt eine Einstiegshürde, auch etwa für den Praktiker, der sich privat mit dem Thema beschäftigen möchte, oder für kleine Unternehmen, die einen Einstieg in eigene Simulationen erwägen.

Zum Glück gibt es inzwischen OpenSource-Alternativen, die zwar nicht den Umfang und die Qualität der großen Vorbilder erreichen, aber durchaus in der Lage sind, auch komplexe Simulationsvorhaben zu ermöglichen. Im Buch werden alle Modelle zunächst mithilfe der kommerziellen Programme erstellt, jedes Kapitel schließt aber mit einem Abschnitt, in dem gezeigt wird, wie man das freie Alternativprodukt für die gleichen Aufgaben verwenden kann. Die grundsätzliche Handhabung ist dabei oft sehr ähnlich und die meisten Modelle lassen sich auch damit realisieren. Im Download-Bereich werden – soweit vorhanden – immer beide Versionen zur Verfügung gestellt.

Das vermutlich am weitesten verbreitete Programm zur Simulation dynamischer Systeme ist Simulink von der Firma Mathworks [94]. Es ist ein Zusatzpaket zu Matlab und kann mithilfe von Matlab-Funktionen erweitert werden. Entsprechende Kenntnisse sind aber für die Erstellung von Modellen in vielen Fällen nicht nötig – wenn auch gelegentlich nützlich. Die Modelle wurden unter der Version R2011a entwickelt, sie sollten alle auch auf neueren Versionen funktionieren. Auch wenn das Programm natürlich ständig gepflegt und erweitert wird, ist seine Grundfunktionalität schon lange stabil. Daher sind Schwierigkeiten auch bei etwas älteren Simulink-Versionen nur selten zu erwarten.

Als OpenSource-Alternative wird das Programm Xcos eingesetzt, das Bestandteil von Scilab ist [89]. Die verwendete Version ist 5.4.0, sie enthält erstmals eine Matlab-artige Benutzeroberfläche. Xcos ist grundsätzlich sehr ähnlich zu Simulink, hat aber einige Einschränkungen in der Modellierung und eine kleinere Basis-Bibliothek. Auf der anderen

Seite verwendet es etwas andere Techniken für den Umgang mit diskreten Ereignissen und mit impliziten Blöcken, so dass einige Beispiele hier leichter zu implementieren waren als in Simulink. Die zugrunde liegende Scilab-Sprache ist Matlab sehr ähnlich, eine Einführung findet man in [13].

Für die im zweiten Teil beschriebene Physikalische Modellierung gibt es mehrere kommerzielle Programme, die sich alle noch stürmisch entwickeln. Nach ausgiebigen Tests entschied sich der Autor für MapleSim von der Firma MapleSoft [56], das auf dem bekannten Algebra-System Maple [55] basiert und in der hier verwendeten Version 6.0 einen einigermaßen ausgereiften Eindruck macht. Es kann mithilfe von Maple erweitert werden, versteht aber ebenfalls die Sprache Modelica, die sich inzwischen als Quasi-Standard im »Physical Modeling«-Bereich etabliert hat. Auch die umfangreiche Modelica-Standard-Bibliothek ist zu großen Teilen integriert, eine nahezu vollständige Unterstützung ist für zukünftige Versionen avisiert.

Das freie Alternativprodukt OpenModelica [76] basiert dagegen völlig auf Modelica, sein Kern ist ein spezieller Compiler für diese Sprache. In der hier verwendeten Version 1.9.0-beta4 sind wesentliche Teile der Modelica-Standard-Bibliothek enthalten. Die Benutzeroberfläche ist eher spartanisch, sie lässt noch manche Möglichkeiten vermissen und ist auch gelegentlich etwas instabil. Man kann sich aber meistens behelfen, indem man bestimmte Änderungen direkt am Modelica-Code vornimmt.

Übrigens hat auch Mathworks inzwischen eine Erweiterung von Simulink um Physical-Modeling-Komponenten entwickelt, basierend auf dem Paket Simscape [93]. Es benutzt allerdings nicht Modelica, sondern eine Mathworks-eigene Sprache, daher wurde es hier nicht in Betracht gezogen.

Alle vier verwendeten Programme sind für die gängigen Betriebssysteme Windows, Linux und MacOS verfügbar. Die Beispiele wurden alle auf einem Ubuntu-12.04-System erstellt, dort wurden auch die Screenshots angefertigt. Gelegentlich treten auf anderen Betriebssystemen kleinere Unterschiede in der Benutzeroberfläche auf, das sollte aber keine Probleme machen. Etwas störender kann dagegen die Verwendung verschiedener Codierungen sein (UTF-8, ISO-8859-1 oder Windows-1252), daher werden Umlaute in Modellen häufig umgeschrieben.

Teil I

Signalorientierte Methoden

2 Arbeiten mit Blockdiagrammen

Die Vorgehensweise bei der Modellerstellung und Simulation ist in allen in diesem Buch verwendeten Programmen ähnlich. Sie soll in diesem Kapitel an einigen Beispielen in Simulink und Xcos geübt werden. Dabei kommen viele wichtige Blöcke aus den mitgelieferten Bibliotheken zum Einsatz.

2.1 Erstellen einfacher Modelle in Simulink

Der Benutzer erstellt ein Modell in Simulink, indem er Blöcke aus einer mitgelieferten oder selbsterstellten Bibliothek in ein Arbeitsfenster kopiert und dort mit Signalleitungen verbindet. Aus dieser grafischen Beschreibung leitet Simulink die Gleichungen für die Zustandsgrößen ab – in den hier betrachteten Beispielen häufig Differenzialgleichungen – und löst sie mit Hilfe der in Matlab bereitstehenden Gleichungslöser (**Solver**). Zur Darstellung der Ergebnisse fügt der Anwender spezielle Ausgabe-Blöcke hinzu, die ausgewählte Größen etwa als Funktionen der Zeit anzeigen.

Ausgangsbasis ist in der Regel die mitgelieferte Blockbibliothek, die eine Vielzahl vordefinierter Blöcke zur Eingabe, Signalverarbeitung und Ausgabe der Ergebnisse enthält. Diese Blöcke sind nach verschiedenene Kriterien (etwa kontinuierlich/diskret, Ein-/Ausgabe) in Unterbibliotheken sortiert. Ergänzend kann man auch eigene Blöcke definieren, indem man sie aus vorhandenen Blöcken zusammensetzt oder sie – in Matlab, C++ oder Fortran – selbst programmiert.

Als erstes Beispiel soll ein Modell der Schwebung erstellt werden, d. h. der Überlagerung von zwei Sinusschwingungen verschiedener Frequenz. Zur Signalerzeugung verwendet es zwei Sinusgeneratoren, die Signalverarbeitung erledigt ein Addierer-Baustein, und ein Oszilloskop-Block zeigt das fertige Ergebnis. Man erwartet – bei nicht zu unterschiedlichen Frequenzen – eine auf- und abschwellende sinusförmige Schwingung.

Starten Sie zunächst das Programm Matlab und geben Sie im `Command Window` am Matlab-Prompt `>>` das Kommando

```
simulink
```

ein. Es öffnet sich ein neues Fenster, das die Unterbibliotheken der Simulink-Blockbibliothek anzeigt (Abb. 2.1).

Bei Doppelklick auf ein Symbol erscheint die zugehörige Blocksammlung, z. B. bei `Sources` die vorhandenen Signalquellen (Abb. 2.2).

Mit Klick auf den `Simulink`-Eintrag in der linken Leiste (`Libraries`) kommen Sie zurück zur Übersicht.

Abbildung 2.1 Blockbibliotheken von Simulink

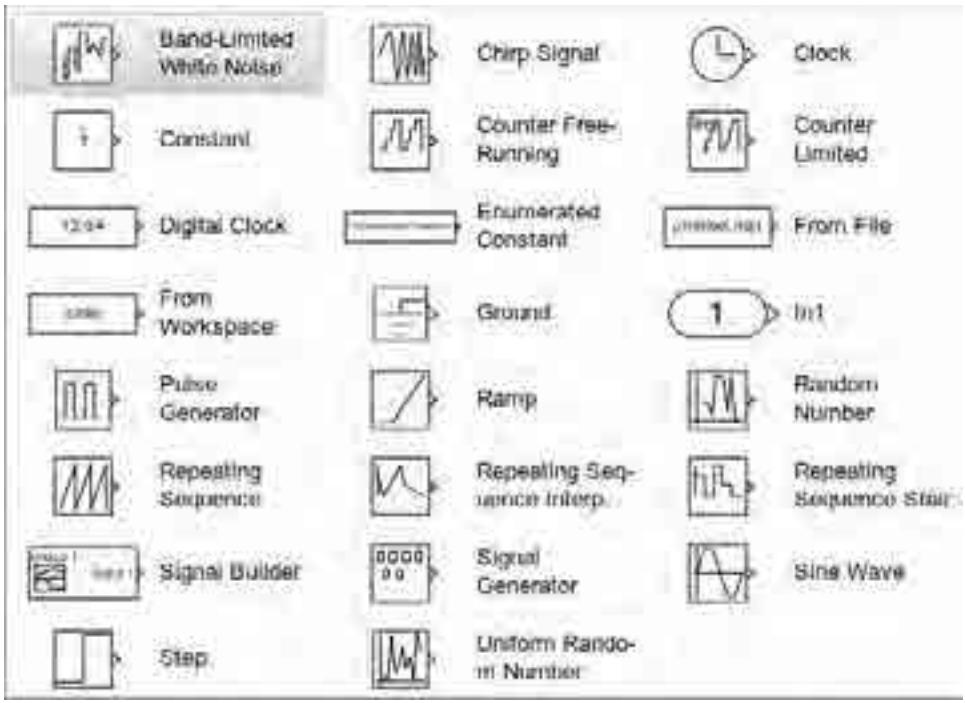

Abbildung 2.2 Bibliothek `Sources`

Wählen Sie nun im Menü den Punkt `File/New/Model`, um eine leere Arbeitsfläche zu erhalten, auf der das neue Modell erstellt werden kann.