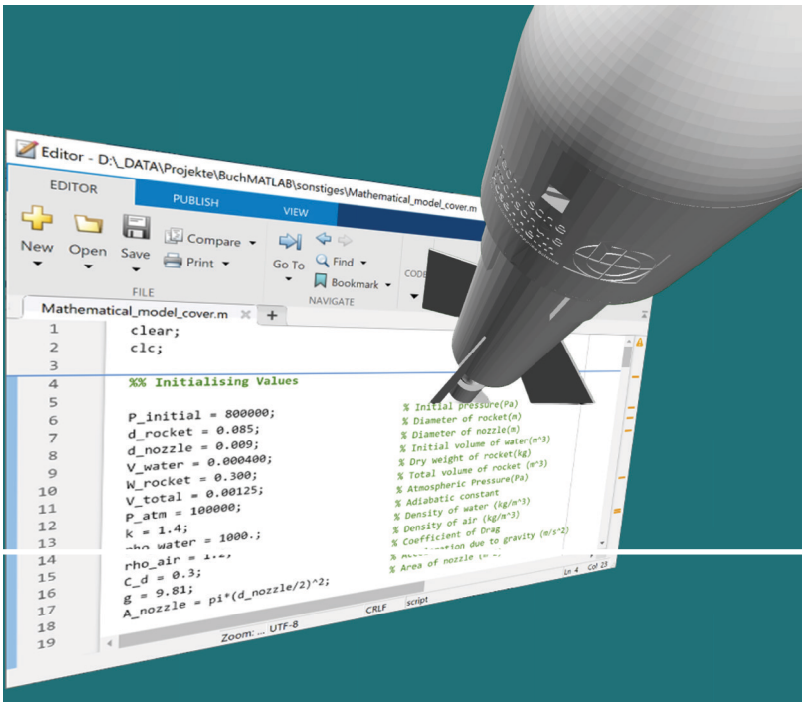
Rainer Hagl
Frank King
Peter Zentgraf

PLUS.HANSER-FACHBUCH.DE

Ingenieurinformatik

Eine Einführung mit MATLAB, Simulink und Stateflow



2., überarbeitete Auflage

HANSER

Hagl/King/Zentgraf

Ingenieurinformatik

Rainer Hagl/Frank A. King/Peter Zentgraf

Ingenieurinformatik

Eine Einführung mit MATLAB©, Simulink© und Stateflow©

2., neu bearbeitete Auflage

HANSER

Die Autoren:

Prof. Dr.-Ing. Rainer Hagl, Technische Hochschule Rosenheim

Prof. Dr.-Ing. Frank A. King, Technische Hochschule Rosenheim

Prof. Dr.-Ing. Peter Zentgraf, Technische Hochschule Rosenheim

MIX
Papier aus verantwortungsvollen Quellen
FSC® C083411

Bibliografische Information der Deutschen Nationalbibliothek:
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.

Internet: www.hanser-fachbuch.de

Lektorat: Frank Katzenmayer
Herstellung: Frauke Schafft
Covergestaltung: Max Kostopoulos
Coverkonzept: Marc Müller-Bremer, www.rebranding.de, München
Titelbild: © Frank A. King/Peter Zentgraf
Satz: Eberl & Koesel Studio, Kempten
Druck und Bindung: CPI books GmbH, Leck
Printed in Germany

Print-ISBN 978-3-446-47515-1
E-Book-ISBN 978-3-446-47730-8

Vorwort

Dieses Lehrbuch stellt eine Einführung in grundlegende Themen der Ingenieurinformatik dar. Schwerpunkt ist die zeiteffiziente Analyse und der Entwurf von technischen Systemen im Ingenieurbereich mittels Software aus dem Bereich der computerunterstützten Entwicklung (Computer Aided Engineering, CAE). Das Thema Ingenieurinformatik wird beispielhaft anhand der Entwicklungsumgebung MATLAB, Simulink und Stateflow zur Analyse technischer Systeme und zur Entwicklung elektronischer Steuergeräte verdeutlicht. Diese Entwicklungsumgebung ist in der Industrie weltweit quasi als Standard für die beschriebenen Aufgaben etabliert. Dadurch ist ein hoher Praxisbezug gegeben.

Das Fachbuch ist insbesondere für die Bachelorausbildung von Studierenden der Ingenieurwissenschaften in folgenden Studienschwerpunkten konzipiert:

- Elektro- und Informationstechnik
- Mechatronik
- Maschinenbau
- Automatisierungstechnik
- Energietechnik
- Gebäudetechnik

Es eignet sich ebenso für technisch Interessierte, die sich in die Thematik einarbeiten wollen.

Zunächst wird in der Einführung auf die Entwicklungen eingegangen, die zum heutigen Einsatz von Computern geführt haben. Es wird die weite Verbreitung der computerunterstützten Entwicklung und Fertigung im Ingenieurbereich dargestellt. Grundlegende Kenntnisse, Zusammenhänge und Werkzeuge bei der Softwareentwicklung werden auf Basis der im technischen und naturwissenschaftlichen Bereich weit verbreiteten Sprache MATLAB gezeigt. Diese sind allgemein gültig und lassen sich weitestgehend auf andere Programmiersprachen übertragen. In vielen Projekten ist es hilfreich, grafische Bedienoberflächen zu verwenden. Dieses Themenfeld wird am Beispiel der in MATLAB eingebauten Möglichkeiten dargestellt. Die digitale Arbeitsweise von Computern führt zu mehr oder weniger starken Begrenzungen des Wertebereiches von Zahlen und damit einhergehend auch zu Einschränkungen der Berechnungsergebnisse. Insbesondere bei der Entwicklung elektronischer Steuergeräte, bei denen die Herstellkosten ein wichtiger Faktor sind, ist die Wertebereichsbegrenzung ein wichtiger Punkt bei der Entwicklung. Eine Einführung hierzu wird im Kapitel „Zahlenformate“ gegeben. Viele Berechnungsaufgaben können aufgrund ihrer Komplexität nur auf Computern mit Näherungsverfahren gelöst werden. Grundlegende Zusammenhänge werden im Kapitel „Numerische Integration“ behandelt.

Eine wichtige praktische Anwendung ist die Simulation dynamischer Systeme, welche beispielhaft anhand des Softwarepaketes Simulink gezeigt wird. In vielen Steuerungsaufgaben muss auf Ereignisse mit vorbestimmten Aktionen reagiert werden. Ereignisdiskrete Systeme werden daher in einem separaten Kapitel behandelt. Um die Arbeitsweise praxisnah dazustellen, wird in diesem Kapitel das Softwarepaket Stateflow verwendet. Manche Berechnungsaufgaben müssen aufgrund der vielen Einzelschritte auf mehrere Recheneinheiten verteilt werden, um in akzeptablen Wartezeiten Ergebnisse zu erhalten. Daher wird in einem eigenen Kapitel auf die Thematik „Paralleles Rechnen“ in Grundzügen eingegangen. Mathematische Umformungen und Vereinfachungen können computerunterstützt erfolgen. Im Kapitel „Symbolisches Rechnen“ wird einführend auf diese Möglichkeit eingegangen. Den Kapiteln zugeordnete Übungen erlauben eine Überprüfung des Lernfortschrittes. Weiteres Zusatzmaterial steht den Lesern unter *plus.hanser-fachbuch.de* zur Verfügung. Den Zugangscode finden Sie auf der ersten Seite des Buches.

In den Jahren seit dem Erscheinen der ersten Auflage blieb die Zeit nicht stehen. Augenfälligste Veränderung ist der nun geschlechtergerechte Buchtitel sowie das Hinzukommen zweier Autoren. Auch konnten von den Autoren weitere Erfahrungen beim Einsatz des Buches in Vorlesungen und Übungen gesammelt werden. Diese sind, zusammen mit technischen Weiterentwicklungen, in die vorliegende zweite Auflage eingeflossen. Dies betrifft insbesondere das Kapitel zu grafischen Bedienoberflächen (Apps), dessen Hauptteil auf aktuellem Stand neu aufgebaut wurde.

Eine Vielzahl konstruktiver Rezensionen der ersten Auflage hat zur Beseitigung von Fehlern und inhaltlichen Klarstellungen in der zweiten Auflage geführt. Hierfür möchten sich die Autoren und der Verlag herzlich bedanken. Allerdings konnten nicht alle Anregungen berücksichtigt werden, da diese zum Teil konträr waren. Bitte haben Sie dafür Verständnis.

Auch in dieser zweiten Auflage haben sich sicherlich noch Fehler eingeschlichen. Vielleicht ist das eine oder andere auch nicht ganz verständlich. Über Rückmeldungen zu Fehlern oder Verbesserungsvorschläge würden wir uns sehr freuen, da diese zu einer kontinuierlichen Verbesserung führen. Sie können uns diesbezüglich gerne eine E-Mail an

frank.king@th-rosenheim.de

peter.zentgraf@th-rosenheim.de

senden. Für Ihre Unterstützung möchten wir uns bereits im Voraus bei Ihnen bedanken.

März 2023 Rainer Hagl | Frank A. King | Peter Zentgraf

Danksagung

Für die zahlreichen konstruktiven Diskussionen und Anregungen rund um die Ausbildung im Bereich der Ingenieurinformatik und Simulation möchten wir uns bei unseren Kollegen Prof. Franz Perschl und Prof. Martin Versen bedanken. Daneben gebührt noch immer ein herzlicher Dank allen, die durch ihr Zutun am Gelingen der vorhergehenden ersten Auflage beteiligt waren.

Der Verlag und die Autoren haben sich mit der Problematik einer gendergerechten Sprache intensiv beschäftigt. Um eine optimale Lesbarkeit und Verständlichkeit sicherzustellen, wird in diesem Werk auf Gendersternchen und sonstige Varianten verzichtet; diese Entscheidung basiert auf der Empfehlung des Rates für deutsche Rechtschreibung. Grundsätzlich respektieren der Verlag und die Autoren alle Menschen unabhängig von ihrem Geschlecht, ihrer Sexualität, ihrer Hautfarbe, ihrer Herkunft und ihrer nationalen Zugehörigkeit.

Zum Cover des Buches:

Der abgebildete Programmcode und die animierte Rakete wurden in dem Projekt „water rocket“ an der Technischen Hochschule Rosenheim im Jahr 2021 unter der Leitung von Prof. Zentgraf entwickelt. Die Projekt-Unterlagen und der zugehörige fünfminütige Filmbeitrag können unter

https://www.th-rosenheim.de/die-hochschule/labore/regelungstechnik/water-rocket

eingesehen werden.

Inhalt

Hinweis

Folgende Handelsnamen werden häufig verwendet, ohne jedes Mal den Rechteinhaber zu nennen:

- MATLAB® ist eine eingetragene Marke der The MathWorks, Inc.
- Simulink® ist eine eingetragene Marke der The MathWorks, Inc.
- Stateflow® ist eine eingetragene Marke der The MathWorks, Inc.
- Handle Graphics® ist eine eingetragene Marke der The MathWorks, Inc.
- MATLAB® Compiler™ ist eine eingetragene Marke der The MathWorks, Inc.

Für die gezeigten Screenshots und zum Test der aufgeführten Programme bzw. Programmausschnitte wurde die MATLAB Revision R2022a verwendet.

Links ins Internet

Zum Teil enthält das Manuskript Informationen, die im Internet zu finden sind. Weiterführende Informationen im Internet sind durch folgendes Symbol gekennzeichnet:

Die Informationen waren bei der Ausarbeitung des Manuskriptes im Internet verfügbar. Es kann jedoch nicht gewährleistet werden, dass sie beim Öffnen des Links immer noch vorhanden sind oder Adressen sich nicht geändert haben.

Formelsymbole

Im gesamten Manuskript wurde versucht, durchgängige und eindeutige Formelsymbole zu verwenden. Bei der ersten Verwendung eines Formelsymbols werden dessen Bezeichnung in Deutsch und Englisch sowie die dazugehörige SI-Einheit und gegebenenfalls wichtige daraus abgeleitete Einheiten angegeben.

F	Kraft	*Force*	N
M	Drehmoment	*Torque*	Nm
x	Position	*Position*	m

Zur Erhöhung der Übersichtlichkeit werden an manchen Stellen diese Angaben wiederholt.

Programmbeispiele

Die mit Icons gekennzeichneten Programmbeispiele sind kapitelweise unter *plus.hanser-fachbuch.de* abgelegt.

Icon	Files	
	MATLAB	Script
		Function
		Figure
		App
		MAT-file
		MEX-file
	Simulink und Stateflow	Model
	Project	

Von dort können diese frei für Ausbildungszwecke, die nicht kostenpflichtig sind, heruntergeladen werden.

1 Einführung

Die Arbeit von Ingenieuren bei der Produktentwicklung erfolgt immer stärker computerunterstützt. Grund hierfür ist, dass Computer bei gleichen oder sinkenden Kosten kontinuierlich leistungsfähiger werden. Selbst von der erforderlichen Computerleistung sehr anspruchsvolle Aufgaben können immer mehr am Arbeitsplatzrechner eines Ingenieurs gelöst werden. Programme (Software), die die Arbeit von Ingenieuren unterstützen, werden unter dem englischen Begriff „Computer Aided Engineering“, oder abgekürzt CAE, zusammengefasst. Sowohl bei der Entwicklung mechanischer als auch elektronischer Baugruppen gibt es eine Vielzahl von Programmen, die einzelne Aufgaben bei der Produktentwicklung unterstützen (Bild 1.1).

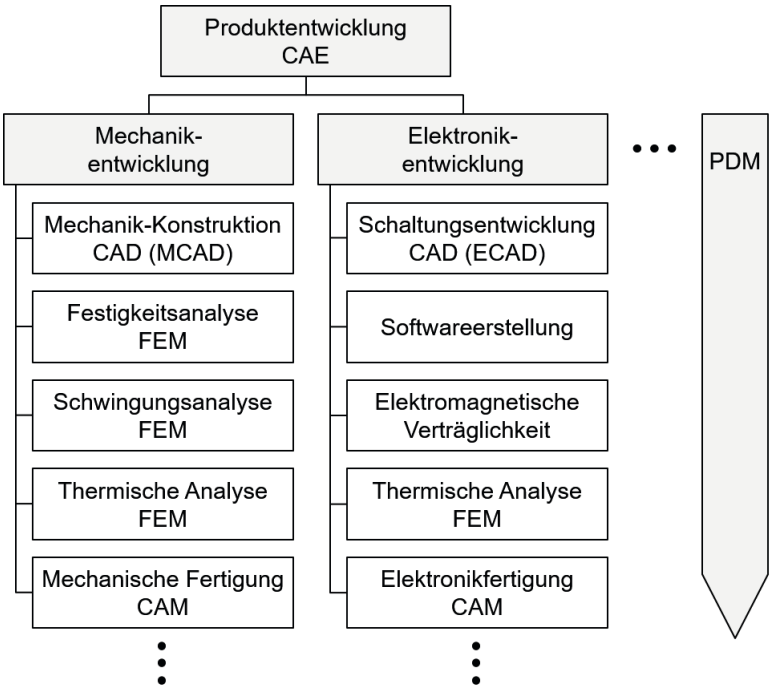


Bild 1.1 Computerunterstützung bei der Produktentwicklung (Beispiele)

Die Konstruktion mechanischer Teile und Baugruppen erfolgt fast ausschließlich mit CAD-Systemen (CAD: Computer Aided Design). In Bild 1.2 ist die Veränderung der Arbeitsweise in der mechanischen Konstruktion vom Arbeiten am Zeichenbrett mit Bleistift und Tusche hin zum Arbeiten am Computer mit dreidimensionaler Darstellung gezeigt. Die Berech-

nung der mechanischen Festigkeit, der Verformung und des Schwingungsverhaltens erfolgt mit Software, die auf speziellen mathematischen Verfahren basiert. Diese Verfahren sind unter dem Begriff FEM (Finite-Elemente-Methode) zusammengefasst. Weitere Anwendungen im Ingenieurbereich sind z. B. thermische Analysen, Strömungssimulationen oder die Berechnung von elektrischen und magnetischen Feldern. Einige Anwendungsbeispiele zeigen Bild 1.3, Bild 1.4 und Bild 1.5.

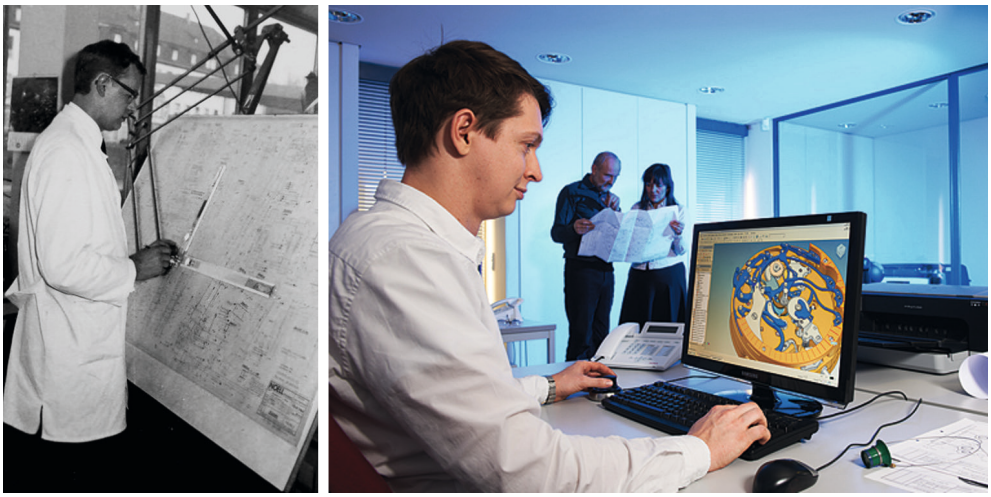


Bild 1.2 Mechanische Konstruktion. Linkes Bild: Konstruktion am Zeichenbrett (© Ing. B. P. Hennek, Würzburg 1968), rechtes Bild: Konstruktion mit CAD-System (© Tutima Uhrenfabrik GmbH Ndl. Glashütte, 2016)

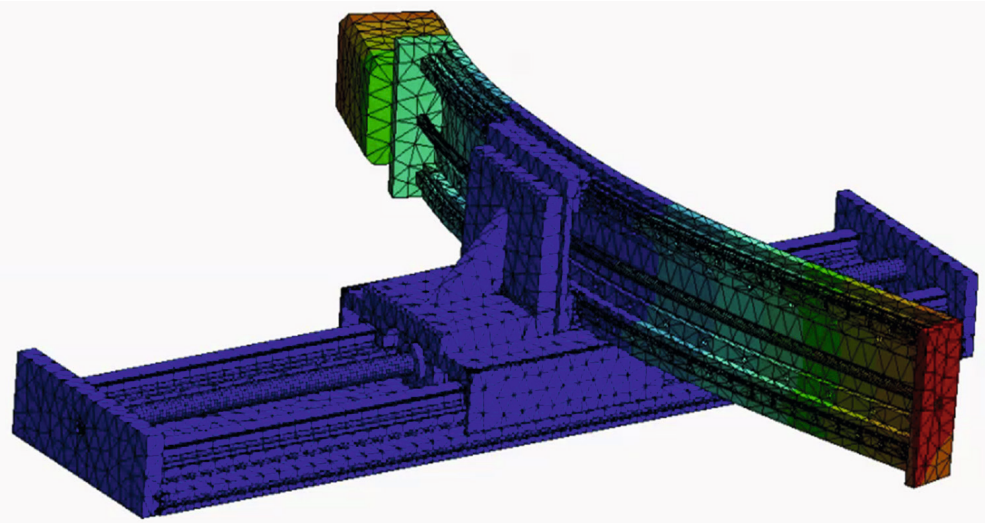


Bild 1.3 Anwendungsbeispiele Finite-Elemente-Berechnung – Verformungsanalyse XY-Tisch (© Labor für elektrische Antriebstechnik, Hochschule Rosenheim, 2016)

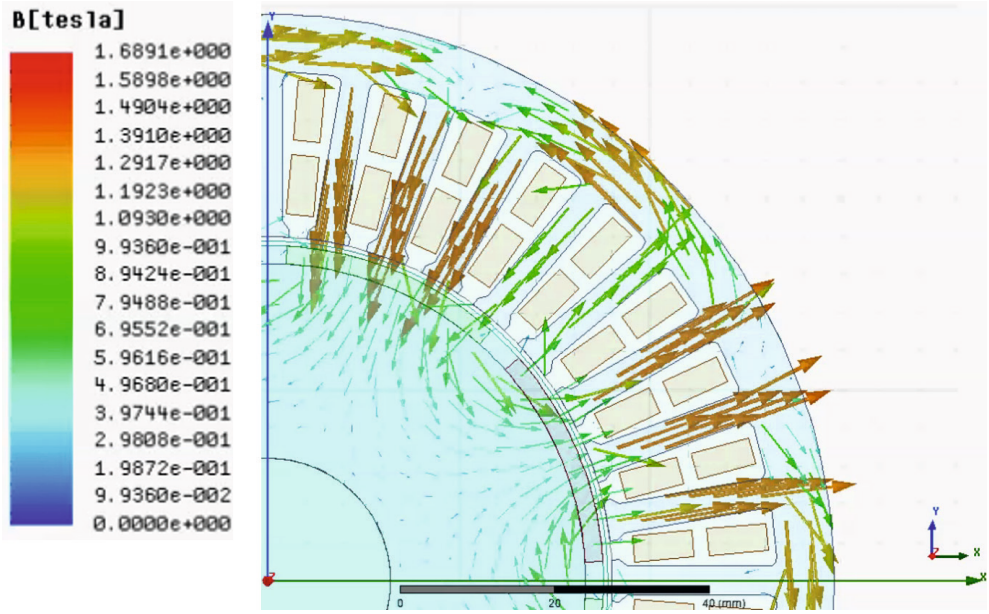


Bild 1.4 Anwendungsbeispiele Finite-Elemente-Berechnung – Magnetfeldberechnung eines Elektromotors (© Labor für elektrische Antriebstechnik, Hochschule Rosenheim, 2016)

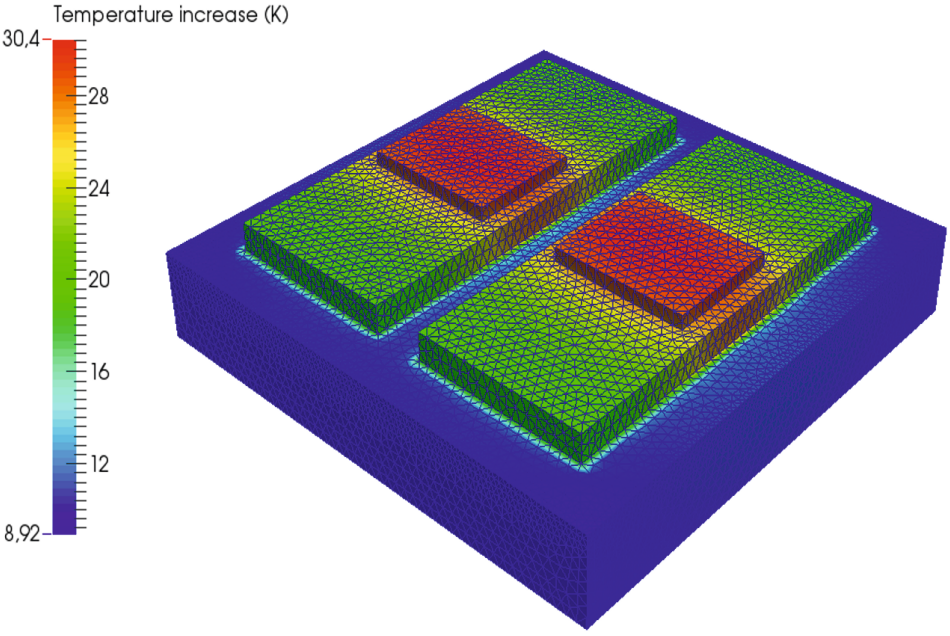


Bild 1.5 Anwendungsbeispiele Finite-Elemente-Berechnung – Temperaturverteilung von Halbleiterchips auf einem Schaltungsträger (© Prof. Dr. techn. Norbert Seliger, Labor für Leistungselektronik, Hochschule Rosenheim, 2016)

Um alle Dokumente und Daten der Produktentwicklung in einem System zu speichern und zu verwalten, gibt es Software zum Produktdatenmanagement (PDM). Neben der Entwicklung speichern dort auch andere Unternehmensbereiche, wie Produktion, Service oder Vertrieb, produktspezifische Informationen ab. Alle für ein Produkt relevanten Informationen werden dort archiviert.

Einige Teilbereiche der computerunterstützten Entwicklung und dazugehörige Produktbeispiele zeigt Tabelle 1.1.

Tabelle 1.1 CAE-Bereiche und Beispielprodukte für den jeweiligen Bereich

Bereich		Software
Computer Aided Design CAD		
Mechanisch	M-CAD	CATIA®, NX™, Creo Parametric®, Solid Edge®, SolidWorks®,...
Elektronisch	E-CAD	cadence®, SYNOPSYS®,...
Computer Aided Manufacturing CAM		
Mechanisch	M-CAM	hyperMill®, NX™, PowerMILL®, Tebis™,...
Elektronisch	E-CAM	
Ingenieurmathematik		
Finite Element Berechnung (Finite Element Methode)	FEM	ABAQUS®, ANSYS®, LS-DYNA®, NASTRAN®, PERMAS®,...
Lösung mathematischer Aufgaben		GNU Octave, SCILab, Maple®, Mathematica®, MATLAB®,...
Regelungs- und Steuerungseinrichtungen		
Modellbasierte Entwicklung		Simulink®, Stateflow®

- CATIA®, SolidWorks® sind eingetragene Marken der Dassault Systèmes SE
- NX™, Solid Edge® sind Marken der Siemens Product Lifecycle Management Software, Inc.
- Creo Parametric® ist eine eingetragene Marke der PTC, Inc.
- cadence® ist eine eingetragene Marke der Cadence Design Systems, Inc.
- SYNOPSYS® ist eine eingetragene Marke der Synopsys, Inc.
- hyperMILL® ist eine eingetragene Marke der OPEN MIND Technologies AG
- PowerMILL® ist eine eingetragene Marke der Delcam Ltd
- tebis™ ist eine eingetragene Marke der Tebis Technische Informationssysteme AG
- ABAQUS® ist eine eingetragene Marke von Abaqus, Inc.
- ANSYS® ist eine eingetragene Marke von Ansys, Inc.
- LS-DYNA® ist eine eingetragene Marke von Livermore Software Technology Corp.
- NASTRAN® ist eine eingetragene Marke der National Aeronautics Space Administration. MSC Nastran ist eine häufig eingesetzte Version die von der MSC Software Corporation entwickelt und gewartet wird.
- PERMAS® ist eine eingetragene Marke der Intes GmbH
- Maple™ ist eine Marken der Waterloo Maple, Inc.
- Mathematica® ist eine eingetragene Marke der Wolfram Research, Inc.
- MATLAB®, Simulink®, Stateflow® sind eingetragene Marken der The MathWorks, Inc.

1.1 Historie Rechenmaschinen

Für sehr viele Produkte werden heute computerbasierte Systeme eingesetzt. Viele Produkte sind durch die Entwicklung leistungsfähiger, miniaturisierter, kostengünstiger und verbrauchsarmer Computer erst möglich geworden. Hierzu zählen:

- Personal Computer, für den einzelnen Büroarbeitsplatz, den Privatgebrauch und den mobilen Einsatz (Notebooks, Tablet, …)
- Smartphones
- Navigationssysteme

- Antiblockiersysteme und Stabilisierungssysteme
- Spielkonsolen
- Flexible und hochautomatisierte Produktionseinrichtungen, wie Industrieroboter, Werkzeugmaschinen, Verpackungsmaschinen etc.

Die Entwicklung hin zu heutigen Computersystemen hat eine lange Historie. Erste Rechenmaschinen zur Lösung einfacher Berechnungsaufgaben wurden mechanisch aufgebaut. Die erste bekannte Rechenmaschine, die alle vier Grundrechenarten (Addition, Subtraktion, Multiplikation und Division) ausführen konnte, war die Leibniz'sche Rechenmaschine (Bild 1.6). Basis für die ersten teilweise elektrischen Rechenmaschinen waren Telefonrelais in Kombination mit Schrittschaltwerken (Bild 1.7). Mit ca. 2000 Telefonrelais wurde der erste, voll funktionsfähige, programmierbare Computer der Welt (Zuse Z3) aufgebaut (Bild 1.8). Einige technische Daten dieses Computers zeigt Tabelle 1.2.

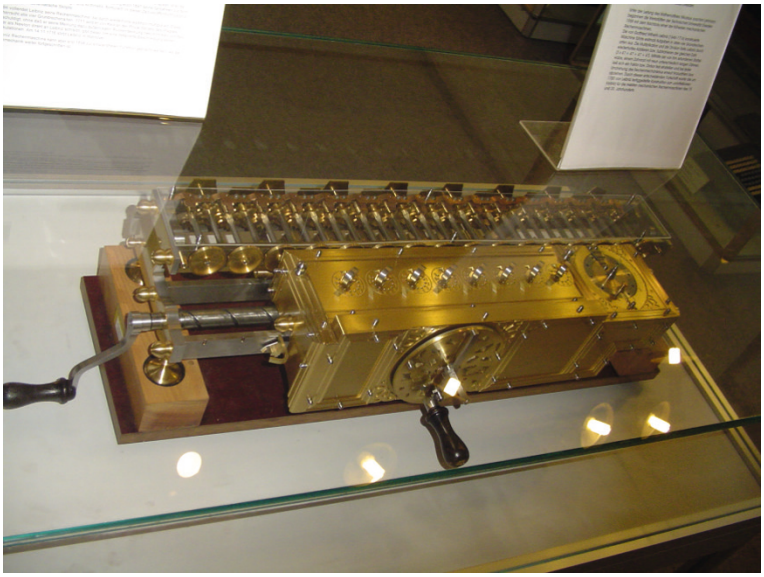


Bild 1.6 Mechanische Rechenmaschine (Leibniz'sche Rechenmaschine), entwickelt von Wilhelm Leibniz ca. 1700 in Dresden (© Kolossos, Technische Sammlungen der Stadt Dresden, Wikipedia, 2016) *https://upload.wikimedia.org/wikipedia/commons/9/92/Leibnitzrechenmaschine.jpg*

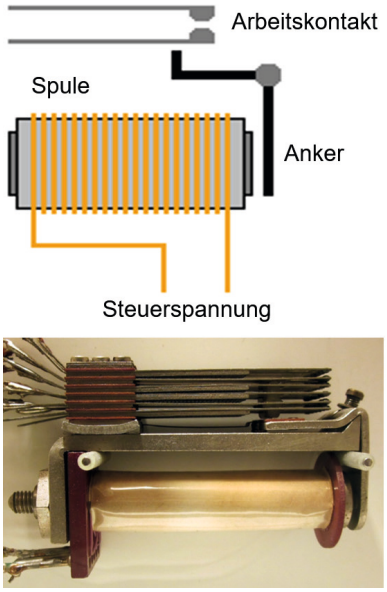


Bild 1.7 Oben: Relaisprinzip und Relais aus Zuse Z3 (© oben: Nogo, Wikipedia, 2016; unten: Denis Apel, Wikipedia, 2016)
https://commons.wikimedia.org/wiki/File:Relais_ruhe.png#/media/File:Relais_ruhe.png
https://de.wikipedia.org/wiki/Zuse_Z3#/media/File:Elektromagnetischerspeicher_zuse_relais.jpg

Bild 1.8 Elektromechanische Rechenmaschine (Zuse Z3), entwickelt von Konrad Zuse 1941 in Berlin (© Archiv Deutsches Museum, München, 2016) *http://www.deutsches-museum.de/archiv*

Tabelle 1.2 Technische Daten des ersten Computers (ZUSE Z3)

Taktfrequenz	ca. 5,4 Hz
Rechenwerk (2 Register)	600 Relais
Speicher (64 × 22 bit)	1400 Relais
Fließkommaeinheit	22 bit
Aufgenommene Leistung	4 kW
Addition	3 Takte
Multiplikation	16 Takte
Division	28 Takte
Ein-/Ausgabe-Einheit	Lochkartenleser für handgestanzten 35 mm breiten Film

Die elektromechanischen Lösungen hatten keine hohe Verfügbarkeit und waren mit hohem Wartungsaufwand verbunden. Sie wurden von rein elektrischen Systemen, die zunächst Elektronenröhren (Bild 1.9) nutzten, abgelöst. Der erste elektrische Universalrechner (ENIAC) wurde an der University of Pennsylvania im Auftrag der US-Armee von 1942 bis 1946 entwickelt (Bild 1.10).

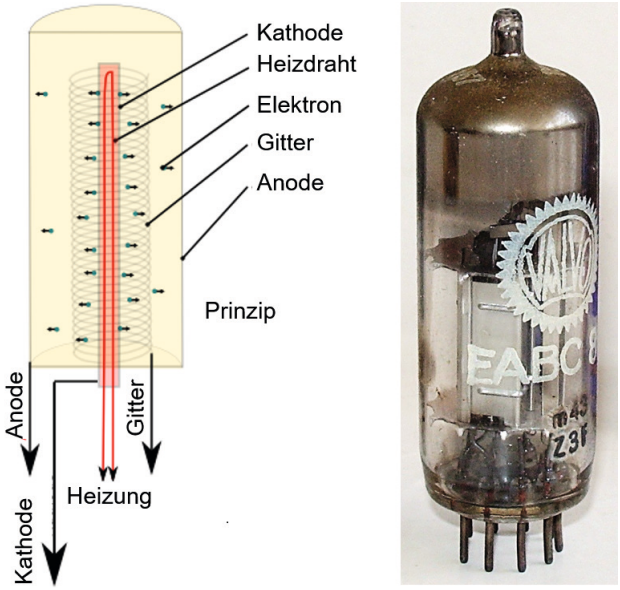


Bild 1.9 Elektronenröhre (© links: Stefan Riepl, Wikipedia, 2016; rechts: Ausschnitt 32-bit-Maschine, Wikipedia, 2016)
https://de.wikipedia.org/wiki/Elektronenr%C3%B6hre#/media/File:Elektronenroehre_real.png
https://de.wikipedia.org/wiki/Elektronenr%C3%B6hre#/media/File:Radio_vacuum_tubes.jpg

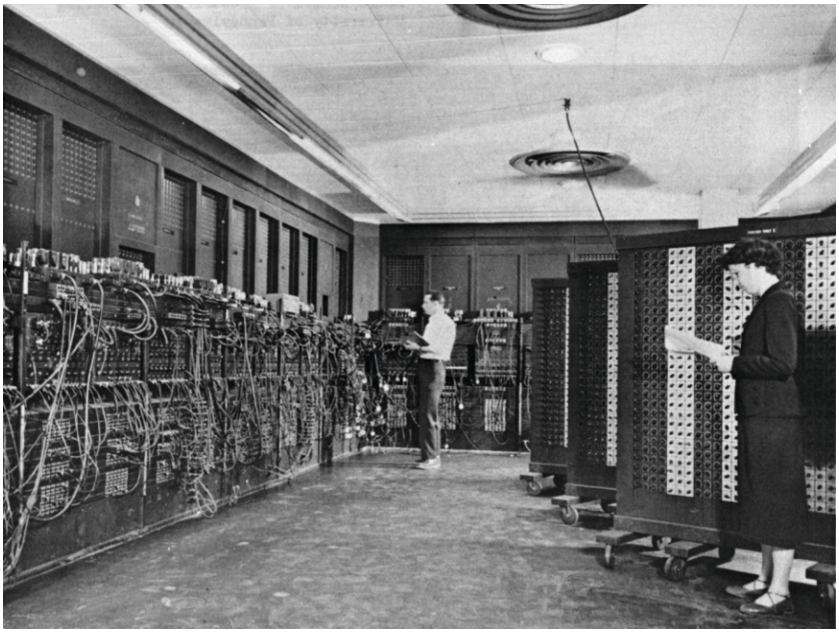


Bild 1.10 Elektrische Rechenmaschine mit Elektronenröhren, Electronic Numerical Integrator and Computer (ENIAC) (© U.S. Army, Wikipedia, 2016) *https://de.wikipedia.org/wiki/ENIAC#/media/File:Eniac.jpg*

Durch die Entwicklung von Halbleitern und Leiterplatten konnten elektronische Systeme stark verkleinert und in der Leistungsaufnahme reduziert werden. Insbesondere die Halbleiterbauelemente Transistor (Bild 1.11) und Diode ermöglichten, im Vergleich zu Lösungen auf Basis von Elektronenröhren, kompaktere, wartungsärmere und leistungsfähigere Rechenmaschinen. Die erste vollständig elektronische Rechenmaschine wurde von den Bell-Forschungslaboratorien für die U.S. Air Force von 1953 bis 1955 (TRADIC) entwickelt (Bild 1.12).

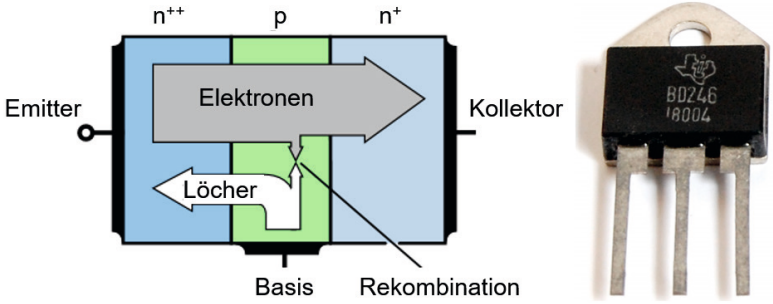


Bild 1.11 Links: Funktionsprinzip Transistor, rechts: Bauelement (© links: Kai Martin, Cepheiden, Wikipedia, 2016; rechts: Ausschnitt Benedikt Seidl, Wikipedia, 2016)
https://de.wikipedia.org/wiki/Transistor#/media/File:NPN_transistor_basic_operation.svg
https://upload.wikimedia.org/wikipedia/commons/0/0e/Transistors-white.jpg

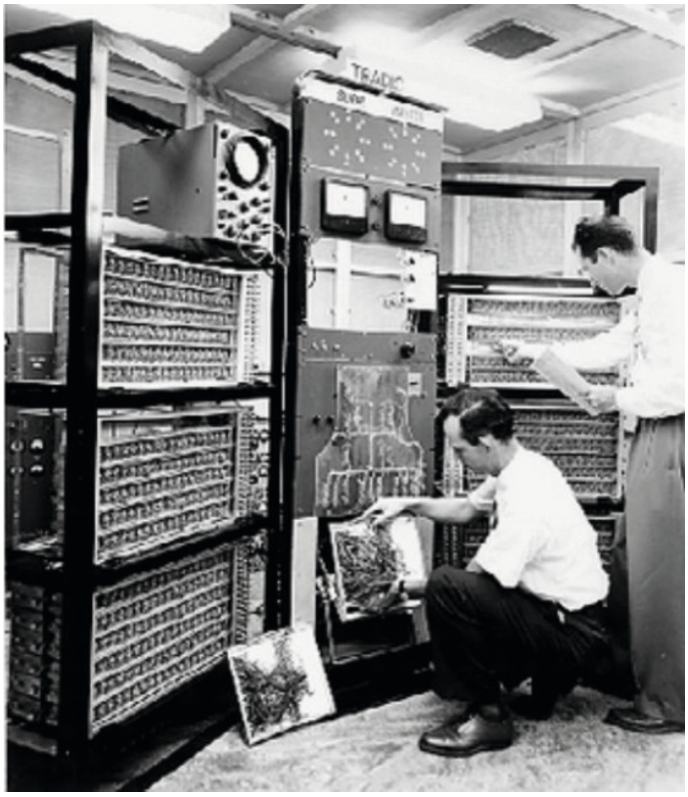


Bild 1.12 Elektronische Rechenmaschine mit Transistoren und Dioden, TRansistorized Airborne DIgital Computer (TRADIC) (© Wikipedia, 2016) *https://en.wikipedia.org/wiki/TRADIC#/media/File:TRADIC_computer.jpg*

Der Aufbau einer Elektronikschaltung mit einzelnen Bauelementen und deren elektrische Verbindung mittels einer Leiterplatte wird als diskrete Elektronik bezeichnet. Durch die Entwicklung integrierter Schaltkreise (IC, Integrated Circuit) war es möglich, sehr viele elektronische Bauelemente auf einem Substratmaterial (Halbleiter) zu realisieren. Gleichzeitig konnten in diesem Material die elektrischen Verbindungen hergestellt werden. Der Platzbedarf für komplexere Schaltungen sank dadurch stark. Es entstanden die ersten Schaltkreise für Berechnungsaufgaben auf Basis dieser Technologie, sogenannte Mikroprozessoren. Der erste ab 1971 in Serie gefertigte und am freien Markt verfügbare Mikroprozessor (Bild 1.13) wurde von Intel® entwickelt. Er bestand aus 2250 Transistoren. Sie bilden bis heute den Kern einer Rechenmaschine bzw. eines Computers.

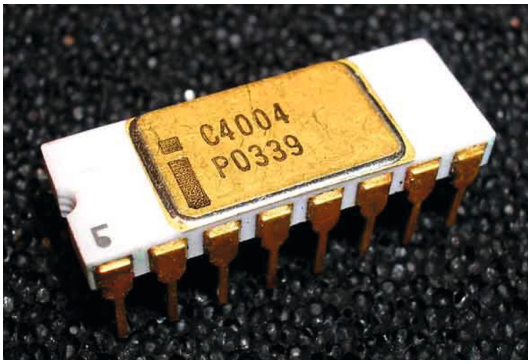


Bild 1.13 Erster in Serie gefertigter und am freien Markt verfügbarer 4-Bit-Mikroprozessor (Intel® 4004) (© Peter1912, Wikipedia, 2016) *https://de.wikipedia.org/wiki/Intel_4004#/media/File:C4004_two_lines.jpg*

Durch die Verfügbarkeit von Mikroprozessoren und deren rasante Weiterentwicklung konnten Computer sehr schnell stark verkleinert und in der Rechenleistung gesteigert werden. Gleichzeitig sanken die Kosten. Dies führte zur Marktreife von Computern für den Privatgebrauch und den Arbeitsplatz. Einen der ersten Heimcomputer, der von MITS in den Jahren 1974 bis 1975 in den USA entwickelt wurde (Altair 8800), zeigt Bild 1.14.

Bild 1.14 Einer der ersten auf einem Mikroprozessor basierenden Heimcomputer: Altair 8800 (© Michael Holley, Wikipedia, 2016) *https://upload.wikimedia.org/wikipedia/commons/0/01/Altair_8800_Computer.jpg*

Im Jahr 1965 hat Gordon Moore (einer der Gründer von Intel®) die Behauptung aufgestellt, dass sich die Anzahl an Transistoren pro Flächeneinheit, bei gleichen Kosten für die Halbleiterfläche, regelmäßig verdoppelt. Als Zeitraum für die Verdopplung wurden anfangs 12 Monate angegeben, mittlerweile wird von 18 bzw. 24 Monaten ausgegangen. Dieser Zusammenhang wird als Moore'sches Gesetz (Moore's Law) bezeichnet. Bis heute verhält sich die Entwicklung von Mikroprozessoren nach dieser Gesetzmäßigkeit. Das Moore'sche Gesetz ist kein Naturgesetz, sondern basiert auf der Beobachtung von Moore, dass die Kosten eines Halbleiterschaltkreises verfahrensbedingt mit sinkender und steigender Komponentenanzahl anstiegen. Bei niedriger Komponentenanzahl wird das Halbleitermaterial nicht voll ausgenutzt. Bei zu hoher Komponentenanzahl steigen die Fertigungskosten durch aufwändigere Herstellungsverfahren bzw. niedrigere Ausbeuten (mehr Ausschuss) an. Dazwischen gibt es ein Kostenoptimum. Die von Moore aufgestellte Gesetzmäßigkeit bezieht sich immer auf das Kostenoptimum. Vereinfacht ausgedrückt steigt die Rechenleistung bei gleichen Kosten kontinuierlich an. Einige Experten gehen davon aus, dass die auf dem Moore'schen Gesetz basierende Entwicklung sich in einigen Jahren nicht mehr fortsetzen lässt.

1.2 Computerunterstützung bei der Lösung mathematischer Aufgaben

Wie in vielen anderen Bereichen ist zur Lösung von Aufgaben bei der Produktentwicklung eine mathematische Beschreibung erforderlich. Ein sehr einfaches Beispiel für eine mathematische Beschreibung ist die horizontale Bewegung einer Masse, auf die in Bewegungsrichtung eine Kraft wirkt (Bild 1.15). Die Bewegungsgrößen, welche die Bewegung beschreiben, sind:

- Position
- Geschwindigkeit
- Beschleunigung

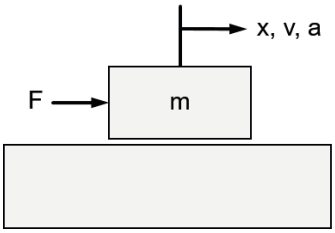


Bild 1.15 Bewegung einer Masse

Die mathematischen Zusammenhänge für die Bewegungsgrößen lauten ganz allgemein:

$$x(t) = \int v(t)\,dt \tag{1.1a}$$

$$v(t) = \int a(t)\,dt \tag{1.1b}$$

Bei konstanter Masse gilt für die Beschleunigung:

$$a(t) = \frac{1}{m} F(t) \tag{1.2}$$

x	Position	*Position*	m
v	Geschwindigkeit	*Velocity*	m/s
a	Beschleunigung	*Acceleration*	m/s²
t	Zeit	*Time*	s
m	Masse	*Mass*	kg
F	Kraft	*Force*	N

Wirkt auf die Masse eine konstante Kraft in Bewegungsrichtung, so ist die Beschleunigung ebenfalls konstant und keine Funktion der Zeit:

$$a(t) = a = \frac{F}{m} \tag{1.3}$$

Damit kann die Lösung für den zeitlichen Verlauf der Position und der Geschwindigkeit direkt erfolgen (geschlossene Lösung). Unter der Annahme, dass die Bewegung zum Zeitpunkt $t = 0$ beginnt und die Position und Geschwindigkeit zu diesem Zeitpunkt jeweils null ist, gilt:

$$v(t) = \int a(t)\, dt = a \int_0^t dt = a\, t \;;\; v(t=0) = 0 \tag{1.4a}$$

$$x(t) = \int v(t)\, dt = a \int_0^t t\, dt = \frac{1}{2} a t^2 \;;\; x(t=0) = 0 \tag{1.4b}$$

Beginnt die Bewegung zum Zeitpunkt $t = t_1$ und hat die Position zu diesem Zeitpunkt den Wert x_1 und die Geschwindigkeit den Wert v_1, gilt:

$$v(t) = a \int_{t_1}^t dt = a(t - t_1) + v_1 \;;\; v(t = t_1) = v_1 \tag{1.5a}$$

$$x(t) = \int_{t_1}^t v(t)\, dt = \frac{1}{2} a (t - t_1)^2 + v_1 (t - t_1) + x_1 \;;\; x(t = t_1) = x_1 \tag{1.5b}$$

Ändert sich die Kraft beliebig mit der Zeit, so gibt es keine geschlossene Lösung zur Beschreibung der Bewegung. Für derartige mathematische Aufgaben werden Verfahren der numerischen Mathematik genutzt. Als Teilgebiet der Mathematik beschäftigt sich die numerische Mathematik mit der Entwicklung und der Analyse von Algorithmen für kontinuierliche mathematische Aufgaben. Numerische Verfahren liefern als Ergebnis Zahlenwerte.

Viele auf mathematischen Beschreibungen basierende Aufgaben von Ingenieuren lassen sich durch numerische Berechnungsverfahren lösen. Das Ergebnis einer numerischen

Berechnung ist aufgrund der verwendeten approximativen Berechnung eine näherungsweise Lösung, d. h., es handelt sich um keine exakte Lösung. Während die exakte Lösung eine unendliche Genauigkeit besitzt, haben näherungsweise Lösungsverfahren eine endliche Genauigkeit. Für viele Aufgaben ist eine endliche Genauigkeit allerdings vollkommen ausreichend.

Die Einsetzbarkeit von numerischen Lösungsverfahren und deren immer häufigere Nutzung zur Lösung komplexer Aufgaben ist sehr eng mit der Entwicklung bei automatischen Rechenmaschinen verbunden. Erst durch leistungsfähige und kostengünstige Computer kam es zur breiten Anwendung numerischer Verfahren zur Lösung von Aufgaben im Ingenieurbereich. Mittlerweile gehört der Einsatz von numerischen Verfahren zum Ingenieuralltag. Ein wichtiger Bereich der Anwendung sind Finite-Elemente-Berechnungen.

Im Gegensatz zur zahlenmäßigen Berechnung bei numerischen Lösungsverfahren wird bei symbolischen Lösungsverfahren mit Symbolen gearbeitet. Symbole bedeuten in diesem Zusammenhang eine Rechnung mit Buchstaben bzw. Variablen. Vorteilhaft ist, dass das Ergebnis unabhängig vom Zahlenwert der Variablen allgemein gültig und exakt ist. Nachteilig ist, dass die manuelle Berechnung der symbolischen Lösung oft sehr zeitaufwändig ist, es leicht zu Fehlern kommen kann und manche Aufgabenstellungen mit Symbolen nicht lösbar sind. Die Lösung für eine mathematische Aufgabenstellung mit Variablen wird zunehmend computerunterstützt durchgeführt (Computeralgebrasystem, CAS). Dadurch werden der Zeitaufwand zur Lösung der mathematischen Aufgabe und die Fehleranfälligkeit stark reduziert. Ein Beispiel für eine symbolische Aufgabe ist die Lösung einer quadratischen Gleichung:

$$3x^2 + bx - 7 = 0 \tag{1.6}$$

Die Eingabe zur Lösung mit einem Produkt der symbolischen Computeralgebra lautet beispielsweise:

```
solve(3*x^2+b*x=7,x);
```

Als Ergebnis liefert die Software das in diesem einfachen Fall bekannte Ergebnis:

$$-\frac{b}{6}+\frac{\sqrt{x^2+84}}{6}, -\frac{b}{6}-\frac{\sqrt{x^2+84}}{6}$$

Weit verbreitete Software für symbolische Mathematik ist Maple® (Mathematical manipulation Language) und Mathematica®.

Je nach mathematischer Aufgabe eignen sich eine oder mehrere aus den beschriebenen Wegen zur Ergebnisermittlung:

- Geschlossene Lösung
- Numerische Lösung
- Symbolische Lösung

Numerische Lösungen werden vor allem in Kapitel 5 „Numerische Integration“ und Kapitel 6 „Zeitgesteuerte Systeme (Simulink)“ vorgestellt und angewendet. Im Kapitel 9 „Symbolisches Rechnen“ werden symbolische Lösungen behandelt. Das ingenieurmäßige Arbeiten steht dabei im Vordergrund, weshalb auf tiefergehende theoretische Hintergründe bewusst verzichtet wird.

Es gibt eine Vielzahl von Programmen bzw. Software, welche die Lösung mathematischer Probleme stark vereinfachen. Sie sind teils auf spezielle Fragestellungen der Produktentwicklung zugeschnitten. Einige Beispiele für nicht auf spezielle Aufgaben zugeschnittene Programme zeigt Tabelle 1.3.

Tabelle 1.3 Beispiele für Mathematikprogramme

Name	Herkunft	Proprietäre Software	Freie Software	Homepage
Maple®	Maple®	x		*http://www.maplesoft.com/*
MATLAB®	MathWorks®	x		*http://www.mathworks.de*
Mathematica®	Wolfram Research ®	x		*http://www.wolfram.com/mathematica/*
GNU Octave	GNU-Projekt		x	*http://www.octave.org/*
Scilab	Scilab-Konsortium		x	*http://www.scilab.org/*

MATLAB ist eine kommerzielle Software zur Lösung mathematischer Aufgaben und zur grafischen Darstellung der Ergebnisse. MATLAB ist insbesondere für die Matrizenverarbeitung optimiert. Daraus entstand auch die Bezeichnung MATLAB (MATrix LABoratory). MATLAB wird hauptsächlich für folgende Aufgaben eingesetzt:

- Numerische Simulation
- Messwertanalysen
- Regelungstechnische Analysen und Optimierungen
- Signalanalyse
- Grafische Visualisierung

MATLAB wurde Ende der 1970er Jahre an der Universität New Mexico entwickelt, um Studierenden die Lösung von Aufgaben der linearen Algebra ohne Programmierkenntnisse zu ermöglichen. Auf Basis dieser Idee wurde die Firma MathWorks 1984 gegründet, und es entstanden kommerzielle Produkte.

Im Bereich der freien Software gibt es Open-Source-Projekte, welche teilweise sehr der Funktionalität von MATLAB entsprechen und mehr oder weniger kompatibel dazu sind. Hierzu zählen z. B. GNU Octave, FreeMat und Scilab.

Um die Grundlagen der Ingenieurinformatik darzustellen, wird im Folgenden MATLAB benutzt. Es ist weltweit in der Industrie und Forschung sehr verbreitet und ein Quasi-Standard. Dadurch wird eine praxisorientierte Darstellung auf dem aktuellen Stand der Technik gewährleistet. Nach Aussagen des Herstellers gibt es weltweit ca. eine Million aktive Lizenzen, wobei unter einer Lizenz teilweise auch sehr viele Anwender gleichzeitig arbeiten.

Zusätzlich sind MATLAB und die dazugehörigen Softwaremodule Teil einer durchgängigen Entwicklungsumgebung für Regelungs- und Steuerungssysteme bis zur Erzeugung der Software für das Serienprodukt. Insgesamt reduzieren sich dadurch die Entwicklungszeiten und die Entwicklungskosten. In Stellenanzeigen werden häufig Kenntnisse in MATLAB gewünscht.

1.3 Modellbasierte Steuergeräteentwicklung

In vielen Investitions- und Konsumgütern sind heutzutage elektronische Steuergeräte ein wesentlicher Bestandteil. Einige Beispiele hierfür sind:

- Kraftfahrzeuge (Benzineinspritzung, Antiblockiersystem, Stabilisierungssystem, Klimaanlage, Abstandsregelung, Spurüberwachung, …)
- Produktionsmaschinen (speicherprogrammierbare Steuerungen, numerische Steuerungen, …)
- Hausgeräte (Herd, Kühlschrank, Waschmaschine, …)

Der weitaus größte Anteil an Funktionen von elektronischen Steuergeräten ist in Software realisiert. In diesem Abschnitt soll lediglich ein erster Eindruck von der Bedeutung der Informatik bei der Produktentwicklung an einigen wenigen Beispielen vermittelt werden.

Die Entwicklung elektronischer Steuergeräte erfolgt zunehmend modellbasiert. Kernelement der modellbasierten Entwicklung ist ein meist mathematisches Modell des zu steuernden oder zu regelnden Systems. Eine wesentliche Zielsetzung ist es, dieses Modell in allen Phasen der Entwicklung zu nutzen, d. h. von der ersten Idee bis zur Serie. Ein wichtiger Vorteil der modellbasierten Entwicklung ist die Effizienz- und Qualitätssteigerung durch automatische Codegenerierung. Die Entwicklungswerkzeuge zur modellbasierten Entwicklung nutzen meist grafische Elemente, um das Systemverhalten zu beschreiben. Hauptsächlich sind dies Blockdiagramme, Schaltpläne (elektrisch, hydraulisch, …), Flussdiagramme und Zustandsgraphen. Grafische Elemente zur Beschreibung des Systemverhaltens werden in den Kapiteln 6 „Zeitgesteuerte Systeme (Simulink)“ und 7 „Ereignisdiskrete Systeme (Stateflow)“ detailliert behandelt. Im Vergleich zu textuellen Beschreibungen sind grafische Beschreibungen anschaulicher und damit auch für denjenigen, der nicht die Beschreibung erstellt hat, in kürzerer Zeit zu verstehen.

Die Produkte MATLAB, Simulink und Stateflow der Firma MathWorks sind weltweiter Quasi-Standard für die modellbasierte Entwicklung. MATLAB ist die Basis für Simulink und Stateflow. Simulink dient der zeitgesteuerten Simulation und Stateflow der ereignisorientierten Simulation. Beide Produkte sind wichtige Bausteine einer integrierten und durchgängigen Entwicklungsumgebung für Steuerungs- und Regelungssysteme (modellbasierte Steuergeräteentwicklung). Insgesamt gibt es von MathWorks knapp 100 Produkte (Firmenangabe) für einfache bis hin zu komplexen Berechnungsaufgaben und für die modellbasierte Entwicklung. Eine Übersicht nach Themenbereichen zeigt Bild 1.16. Detaillierte Informationen finden sich auf der Homepage von MathWorks.

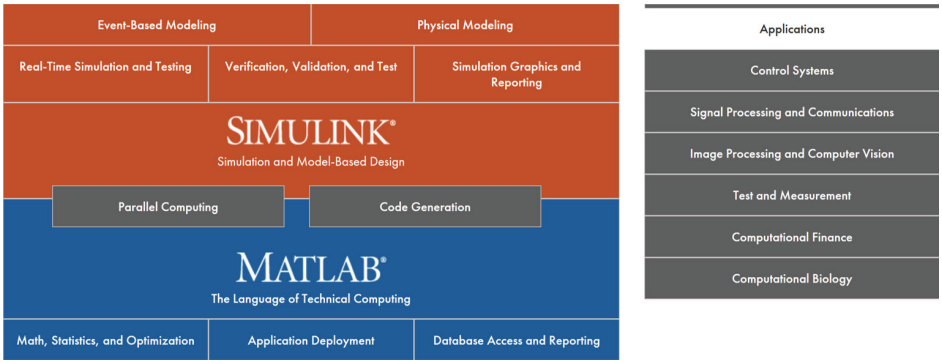


Bild 1.16 MATLAB und Simulink Produktfamilie und Applikationen (© MathWorks, Homepage, 2016) *http://de.mathworks.com/products/pfo/*

Passend zu MATLAB und Simulink gibt es für verschiedene technische Bereiche (wie Mathematik, Messtechnik, Bildverarbeitung), aber auch nicht-technische Bereiche (wie Finanzwirtschaft, Biologie, …) applikationsspezifische Programme. Diese Programme werden in sogenannten Toolboxen zusammengefasst. Eine Toolbox unterstützt die im jeweiligen Anwendungsbereich üblicherweise notwendigen Funktionen. Die Toolboxen werden von MathWorks separat angeboten und es entstehen dadurch Zusatzkosten. Die Verwendung von Funktionen einer Toolbox ist, im Vergleich zur eigenständigen Programmierung der Funktionen, deutlich zeit- und kosteneffizienter. Gleichzeitig ist die Sicherheit in einem Projekt höher, da die Funktionen vom Hersteller und bei bereits seit Längerem angebotenen Funktionen auch vielen Anwendern getestet sind.

Einzelne Phasen der Produktentwicklung mit der in der jeweiligen Phase eingesetzten Methodik, Hardware und Software zeigt Tabelle 1.4 und Bild 1.17.

Tabelle 1.4 Computerunterstützte und automatisierte Entwicklung. Beispiel: modellbasierte Entwicklung von Steuergeräten

Entwicklungsphase	Methode	Hardware	Software
❶ Virtuelles System	Mathematisches Modell/ Computersimulation	PC, Hochleistungs-computer, ...	Engineering Software MATLAB®, Simulink®, Stateflow®, ...
↓	Automatische Codegenerierung für Echtzeitrechner		
❷ Funktionsmuster Teilsystem	Rapid Control Prototyping	Echtzeitrechner ①	Engineering Software Real-Time Interface (RTI), ControlDesk®, Simulink Real-Time®, ...
↓	Automatische Codegenerierung für Echtzeitrechner und/ oder Serienprozessor		
❸ Prototyp	Rapid Control Prototyping	Echtzeitrechner und /oder Prototypen-Elektronik ①	Engineering Software und/ oder Prototypen-Software
↓	Automatische Codegenerierung für Serienprozessor		
❹ Vorseriengerät	Test mit seriennaher Hardware und Software ②	Vorserienelektronik	Vorseriensoftware
↓	Serienfreigabe		
❺ Seriengerät		Serienelektronik	Seriensoftware

↓ Phasenübergang
① Industriell: z. B. dSPACE Hardware (wie AutoBox, MicroAutoBox, MicroLabBox,…), Speedgoat Hardware;
Für Ausbildungszwecke: z. B. Raspberry Pi, Arduino
② Unter Anderem Hardware in the Loop (HIL). Vorseriensteuergerät wird mit Computermodell des zu steuernden Systems gekoppelt.
MATLAB®, Simulink®, Stateflow® und Simulink Real-Time® sind eingetragene Marken der The MathWorks, Inc.
Real-Time Interface (RTI) ist ein Produkt und ControlDesk® eine eingetragene Marke der dSPACE GmbH
Raspberry Pi® ist eine eingetragene Marke der Raspberry Pi Foundation
Arduino® ist eine eingetragene Marke der Arduino LLC.

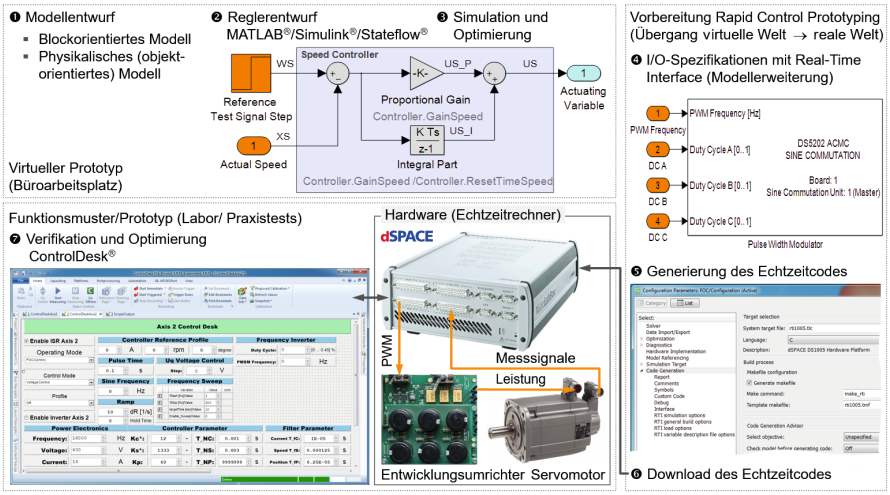


Bild 1.17 Computergestützter Entwicklungsablauf. Applikationsbeispiel: elektrische Antriebstechnik im Labor der Hochschule Rosenheim

Bild 1.18 zeigt eine grafische Bedienoberfläche für Aufgaben in der Steuergeräteentwicklung für Kraftfahrzeuge. Teilaspekte hieraus werden in den folgenden Kapiteln vertiefend behandelt.

Bild 1.18 Steuergeräteentwicklung in der Automobilindustrie. Experimentier- und Visualisierungssoftware für Hardware-in-the-Loop Test und Rapid Control Prototyping (© dSPACE GmbH, 2016)

Ein Beispiel für die modellbasierte Entwicklung elektronischer Steuergeräte, wie es im Bereich der Forschung und Entwicklung mechatronischer Systeme an der Hochschule für angewandte Wissenschaften Rosenheim eingesetzt wird, zeigt Bild 1.19.

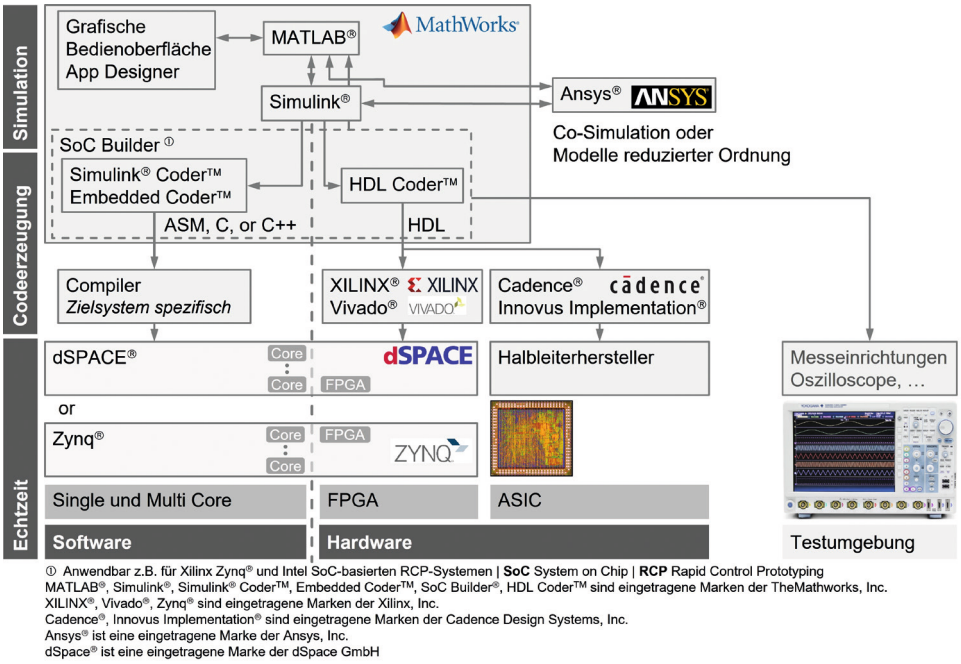


Bild 1.19 Entwicklungsumgebung zur modellbasierten Entwicklung elektronischer Steuergeräte an der Hochschule Rosenheim

Zentraler Bestandteil der ersten Entwicklungsphase (virtuelles System, siehe Tabelle 1.15) ist grundsätzlich die Simulation der Regelungs- und Steuerungsverfahren auf Basis von MATLAB, Simulink und Stateflow. Falls für einzelne Simulationsaufgaben erforderlich, erfolgt eine Kopplung mit hierauf spezialisierter Berechnungssoftware, wie z. B. aus dem Bereich der Finite-Elemente-Berechnung (in Bild 1.19 wird die FEM-Software Ansys® verwendet). Bei der Kopplung gibt es zwei Varianten:

- Einmalige Übernahme von Modellparametern vor Beginn der Simulation

Diese Variante ist bezüglich der Berechnungszeit vorteilhaft, da Modellparameter einmalig in der spezialisierten Berechnungssoftware ermittelt werden. Während der eigentlichen Simulation des Regelungs- oder Steuerungssystems ist kein Datenaustausch erforderlich. Dies kann z. B. bei linearen strukturmechanischen Fragestellungen genutzt werden, indem in der FEM-Berechnungsumgebung eine Ordnungsreduktion durchgeführt und eine Zustandsraumbeschreibung des mechanischen Systems erzeugt wird. Die Daten der Zustandsraumbeschreibung werden als Datei automatisch in MATLAB eingelesen und in MATLAB oder Simulink weiterverarbeitet.

- Co-Simulation

Diese Variante ist bezüglich der Berechnungszeit aufwändiger, da in jedem Integrationsschritt des numerischen Integrationsverfahrens die spezialisierte Berechnungssoftware aufgerufen wird und dort eine Berechnung für den nächsten Zeitschritt erfolgen muss. Während der Simulation des Regelungs- oder Steuerungssystems ist ein permanenter Datenaustausch erforderlich. Dies kann z. B. zur Lösung nichtlinearer strukturmechanischer Fragestellungen genutzt werden, die mit der ersten Variante nicht lösbar sind.

Für die zweite und dritte Entwicklungsphase (Funktionsmuster oder Teilsystem und Prototyp, siehe Tabelle 1.15) sind Echtzeitsysteme, in denen die digitalen Regelungs- und Steuerungsverfahren ablaufen, erforderlich. Abhängig davon, ob die Verfahren in Software oder Hardware realisiert werden, erfolgt die erforderliche Codegenerierung automatisiert aus der bereits in der ersten Entwicklungsphase verwendeten virtuellen Systembeschreibung. Im Einzelnen sind dies:

- Bei einer softwarebasierten Lösung auf Mikroprozessorbasis

  Codegenerierung mittels Embedded Coder für den jeweiligen C-Compiler des verwendeten Mikroprozessors.

- Bei einer hardwarebasierten Lösung mittels FPGA, PLD, ASIC, …

  HDL-Codegenerierung mittels HDL Coder.

Zur praktischen Erprobung der Regelungs- und Steuerungsfunktionen werden zwei Plattformen eingesetzt. Dies sind hochperformante Rapid-Control-Prototyping-Systeme der Firma dSPACE oder Zync-basierte Systeme. Bei den dSPACE-Systemen werden die Schnittstellen zum zu erprobenden System, wie z. B. D/A-Wandler, A/D-Wandler oder PWM-Ansteuerung, bereits vom Hersteller zur Verfügung gestellt. Die Zync-basierten Systeme haben hierfür im Wesentlichen eigenentwickelte Peripherie. Beide Plattformen zur Erprobung besitzen Dual-Core-Prozessoren und umfangreiche FPGA-Hardware. Damit ist es möglich, Aufgaben auf mehrere Prozessoren zu verteilen und ggf. auch gemischte Systeme aus software- und hardwarebasierten Teilsystemen zu erproben. Erforderliche Messeinrichtungen, wie z. B. Mixed-Signal-Oszilloskope oder Leistungsanalysatoren, kommunizieren ebenso wie die beiden Erprobungsplattformen mit MATLAB.

In Bild 1.20 ist exemplarisch ein Laboraufbau zur praktischen Erprobung von Steuerungs- und Regelungsverfahren für elektrische Antriebe, welche modellbasiert entwickelt werden, gezeigt.

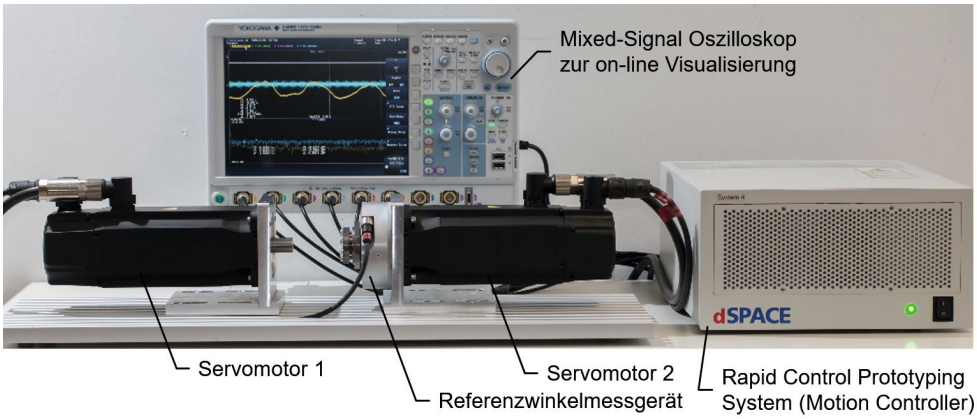


Bild 1.20 Rapid Control Prototyping Versuchsstand zur Erprobung von Steuerungs- und Regelungsverfahren im Labor für mechatronische Systeme der Hochschule für angewandte Wissenschaften Rosenheim

2 Grundlagen der Programmierung

Zur Programmierung von Computern gibt es, ähnlich wie bei natürlichen Sprachen, eine sehr große Anzahl an Programmiersprachen. Diese können sich, wie sogenannte „Maschinensprachen bzw. Assemblersprachen“, sehr nahe an den Abläufen in den elektronischen Schaltungen orientieren. Deutlich abstrakter aufgebaut sind sogenannte „Hochsprachen“, bei denen es meist nicht erforderlich ist, Kenntnisse über die Arbeitsweise der elektronischen Schaltungen zu besitzen. Hierzu zählen z. B. C, C++, Visual Basic®*, Java™, HTML, MATLAB® und Phyton®. Die Anweisungen von Hochsprachen müssen durch Interpreter oder Compiler in für die jeweils verwendete elektronische Schaltung ausführbare Anweisungen umgesetzt werden. „Hochsprachen“ haben zwei wesentliche Vorteile im Vergleich zu „Maschinensprachen bzw. Assemblersprachen“:

- Einfacher, auch für vollständig Unerfahrene, zu erlernen.
- Lösungen für Aufgaben sind damit einfacher und schneller zu formulieren, d. h., die Tätigkeit der Programmierung ist zeit- und kosteneffizienter.

Der wesentliche Unterschied zwischen Interpretern und Compilern bei der Umsetzung ist:

- Interpreter

  Die im Programm enthaltenen Anweisungen werden während des Programmablaufes Schritt für Schritt analysiert und umgesetzt. Das Programm wird interpretiert.

- Compiler

  Alle im Programm enthaltenen Anweisungen werden vor dem Programmablauf in einem separaten Vorgang vollständig analysiert und umgesetzt. Das Programm wird kompiliert. Erst danach kann der Programmablauf gestartet werden.

Der Vorteil von Interpretern ist, dass der separate Vorgang und zeitliche Aufwand für das Kompilieren entfällt. Entscheidender Nachteil ist der deutlich langsamere Programmablauf. Je öfter das gleiche Programm ausgeführt wird, umso vorteilhafter ist es bezüglich des Gesamtzeitaufwandes, Compiler zu verwenden. Die Grenzen zwischen beiden Lösungen der Umsetzung sind oft fließend und in einigen Hochsprachen werden beide Lösungen angeboten.

In diesem Kapitel werden die Grundlagen der Programmierung am Beispiel der Entwicklungsumgebung MATLAB dargestellt. Viele Methoden und Funktionen gibt es so oder in ähnlicher Weise auch in Programmierumgebungen anderer Hochsprachen. Ist ein Grund-

---

* Visual Basic® ist eine eingetragene Marke der Microsoft Corporation.
Java™ ist eine eingetragene Marke der Oracle Corporation.
Phyton® ist eine eingetragene Marke der Python Software Foundation.

verständnis fürs Programmieren vorhanden, lassen sich vergleichsweise schnell andere Programmiersprachen erlernen.

Das Kapitel ist keine detaillierte Darstellung von MATLAB. Hierfür gibt es die in die MATLAB Entwicklungsumgebung integrierte Hilfe und Dokumentation. Im Kapitel werden vielmehr die wichtigsten Zusammenhänge dargestellt, um einen Überblick für den Einstieg zu erhalten.

2.1 Erste Schritte in MATLAB und Grundregeln

MATLAB ist primär eine

- interaktive Programmierumgebung und gleichzeitig
- eine Programmiersprache

zur computerunterstützten Lösung insbesondere mathematischer und naturwissenschaftlicher Aufgaben. Ebenso wie bei anderen Hochsprachen können in MATLAB Programme erstellt und interpretiert oder kompiliert werden.

Im Internet gibt es viele sehr hilfreiche Schulungsunterlagen, die zum Eigenstudium sehr geeignet sind. Unter *https://de.mathworks.com/videos.html* stellt The MathWorks, Inc. zahlreiche, teils auch deutschsprachige Videos zur Verfügung.

www	Videos und Webinare von The MathWorks, Inc. (Deutsch und Englisch) *https://de.mathworks.com/videos.html*
www	Einführungsvideo Englisch *https://de.mathworks.com/videos/getting-started-with-matlab-68985.html*
www	Tutorials Englisch *https://matlabacademy.mathworks.com/*

2.1.1 Bedienoberfläche

Beim Start von MATLAB erscheint eine in mehrere Bereiche eingeteilte Bedienoberfläche (MATLAB Desktop) (Bild 2.1). Die wichtigsten Bereiche sind markiert.

Das Aussehen der Bedienoberfläche variiert abhängig von der Version und den Voreinstellungen! Es kann vom Anwender im Pull-Down-Menü „Layout“ verändert werden (Markierung in Bild 2.2). Nicht geübte Anwender sollten die Einstellung auf „Default“ stellen.

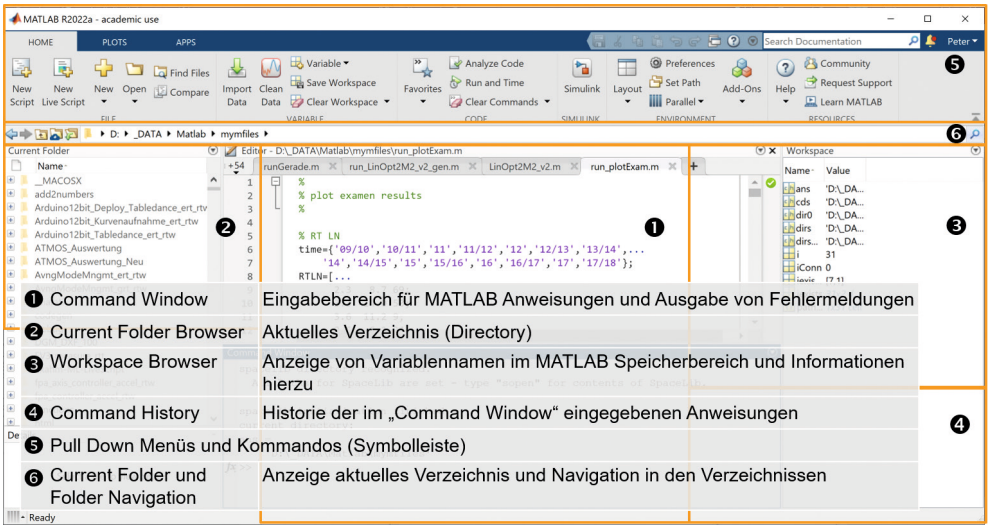


Bild 2.1 Oberfläche beim Programmstart (MATLAB Desktop) und wichtige Bereiche

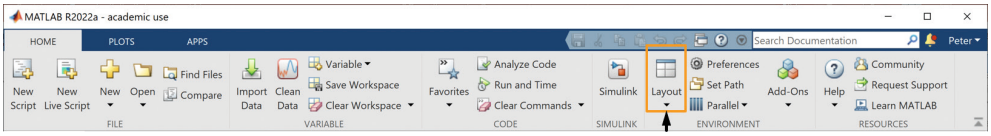


Bild 2.2 Layout

MATLAB kann im einfachsten Fall wie ein Taschenrechner genutzt werden. Dazu erfolgt im „Command Window“ hinter dem „>>“-Zeichen (Eingabeaufforderung, Prompt) die Eingabe, was berechnet werden soll. Für Grundrechenarten werden die in Tabelle 2.1 gezeigten Symbole (MATLAB Operatoren) benutzt.

Tabelle 2.1 MATLAB Operatoren für die Grundrechenarten

MATLAB	Operator
Addition	+
Subtraktion	-
Multiplikation	*
Division	/

So können z. B. zwei Zahlen multipliziert werden:

```
2*3
```

Nach dem Drücken der Eingabetaste (Enter-Taste, ↵), im Folgenden nur noch mit „Enter“ bezeichnet, erhält man folgende Ergebnisanzeige:

```
ans =
    6
```

ans ist eine Abkürzung für „answer“. Gleichzeitig ist es eine Standardvariable von MATLAB, die immer dann benutzt wird, wenn keine eigene Variable zugewiesen wird. Eigene Variablenzuweisungen werden im nächsten Schritt erklärt.

Die neu angelegte Variable ans wird im „Workspace Browser“ angezeigt (siehe Bild 2.3).

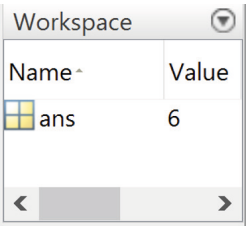


Bild 2.3 Workspace

Immer nach Betätigen der Eingabetaste wird die Eingabe vom MATLAB Interpreter dahingehend überprüft, ob es sich um eine gültige MATLAB Anweisung handelt, d.h. die MATLAB Syntax (Regeln, „Grammatik“) eingehalten wird. Ist dies der Fall, wird die Anweisung ausgeführt, ansonsten erfolgt eine Fehlermeldung.

Wie beim Taschenrechner gibt es vordefinierte Konstanten. Beim Taschenrechner finden sich diese auf vorbelegten Tasten. Bei MATLAB werden sie über Namen angesprochen. So wird z.B. π über den Namen „pi“ eingegeben. Gibt man „pi“ ein, erhält man als Antwort den Wert.

```
pi
ans =
    3.1416
```

Zu beachten ist, dass der Wert der Variablen ans sich dadurch auch automatisch ändert, wie dies in Bild 2.4 des „Workspace Browser“ dargestellt ist.

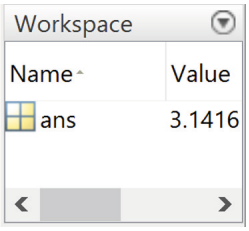


Bild 2.4 Aktualisierung