Systeme der Regelungstechnik mit MATLAB® und Simulink®

Analyse und Simulation

von

Prof. Dr.-Ing. Helmut Bode

2., aktualisierte Auflage

Oldenbourg Verlag München

Lektorat: Dr. Gerhard Pappert
Herstellung: Tina Bonertz
Titelbild: Autor
Einbandgestaltung: hauser lacour

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

Library of Congress Cataloging-in-Publication Data
A CIP catalog record for this book has been applied for at the Library of Congress.

Rosenheimer Straße 143, 81671 München, Deutschland
www.degruyter.com/oldenbourg
Ein Unternehmen von De Gruyter

Gedruckt in Deutschland

Dieses Papier ist alterungsbeständig nach DIN/ISO 9706.

ISBN 978-3-486-73297-9
eISBN 978-3-486-76970-8

Vorwort

Die Notwendigkeit der computerunterstützten Analyse und Synthese dynamischer Systeme und das Vorhandensein leistungsfähiger Rechner haben zahlreiche Softwareprodukte auf den Markt gebracht. Die in diesem Buch auf die Belange der Analyse und Simulation nichtlinearer und linearer dynamischer Systeme und der linearen Regelungstechnik angewendete Software MATLAB® und Simulink® gehört mit zu den am weitesten verbreiteten Produkten dieser Art.

Auf der Grundlage des Prinzips Lehren und Lernen mit dem Computer soll den Studierenden eine Anweisung in die Hand gegeben werden, die es ihnen erlaubt, den in der Vorlesung vorgetragenen Stoff in Form von betreuten und selbständigen Übungen am Rechner zu erlernen, zu hinterfragen und zu festigen.

Neben den Studierenden richtet sich dieses Buch aber auch an Fachleute aus der Praxis, Forschung und Entwicklung, die entsprechende Aufgabenstellungen zu lösen haben.

Grundlagen für die in diesem Buch durchgeführten Berechnungen sind[1]:

MATLAB	Version 7.7	(R2008b)
Simulink	Version 7.2	(R2008b)
Control System Toolbox	Version 8.2	(R2008b)
Symbolic Math Toolbox	Version 5.1	(R2008b)

Für das Überlassen dieser Software bedanke ich mich bei The MathWorks, 3 Apple Hill Drive, Natick, MA 01760-2098 USA und speziell bei den Damen und Herren von The MathWorks Book Program Team.

Eine der wesentlichen Voraussetzungen für die Automatisierung technologischer Prozesse ist die Analyse und Simulation dynamischer Systeme. Die damit im Zusammenhang auftretenden Grundbegriffe werden im ersten Kapitel – Einleitung – behandelt.

In die notwendigen Grundoperationen von MATLAB und in die Grundbausteine der Simulink-Signalflusspläne wird in den Kapiteln zwei und drei eingeführt, sie können aber nicht die entsprechenden Handbücher bzw. die M-function help *name* / doc *name* ersetzen.

---

[1] Sämtliche im Buch behandelten Beispiele und functions wurden mit den MATLAB-Prerelease-Versionen R2009b getestet und sind darunter voll lauffähig.

Ohne Systemanalyse keine Computersimulation, d.h. die Analyse und Simulation dynamischer Systeme mit einem Softwareprogramm setzt das Vorhandensein mathematischer Modelle voraus. Die Beschreibung des Aufbaus der Modelle konkreter und abstrakter sowie linearer und nichtlinearer Systeme ist Gegenstand des vierten Kapitels.

Vertieft werden diese Zusammenhänge durch das Aufstellen der mathematischen Beziehungen zum dynamischen Verhalten an drei physikalisch unterschiedlich konkreten Systemen. Zwei davon liefern das gleiche abstrakte Modell in Form einer linearen Differenzialgleichung 1. Ordnung. Das Zeitverhalten des dritten Systems wird durch eine nichtlineare Differenzialgleichung 1. Ordnung beschrieben. Die dazugehörenden Simulink-Signalflusspläne werden angegeben.

Für eine computergestützte Simulation ist es notwendig, die mathematischen Modelle der zu simulierenden Systeme in einer geeigneten Form darzustellen. Unter MATLAB/Simulink ist besonders die Darstellung in Form der nichtlinearen bzw. linearen Zustandsraumbeschreibung, d.h. als Vektor-Matrix-Differenzial-Gleichungssystem bzw. als Signalflussplan, geeignet. Die dazu erforderlichen Grundlagen, einschließlich der Bildung der nichtlinearen Funktionen und der Linearisierung sind ebenfalls Inhalt des vierten Kapitels.

Gegenstand des fünften Kapitels ist die Vorgehensweise beim Erstellen mathematischer Prozessmodelle auf der Grundlage der Lagrange'schen Bewegungsgleichung 2. Art. Die behandelten Systeme sind:
Ein mechanisches System „Stab-Wagen".
Ein Gleichstrom-Scheibenläufer-Motor mit Seilscheibe und Umlenkrollen, als System „Antrieb" bezeichnet.
Ein System „Inverses Pendel" als Vereinigung der beiden ersten Systeme.
Ein als „Regelkreis" aufgebautes System, in welchem über eine lineare Zustandsrückführung zunächst das „Inverse Pendel" grenzstabilisiert wird, um es dann mittels einer Mitkopplung des Wegsignals über einen *PI*-Regler zum Regelkreis zu komplettieren und zu stabilisieren.
Das System „Netzwerk", hier handelt es sich um ein sprungfähiges elektrisches Netzwerk.
Schließlich ein RLC-Netzwerk, als System „Brückenschaltung".
Die mathematischen Modelle der im fünften Kapitel behandelten Beispiele repräsentieren eine große Klasse von nichtlinearen bzw. linearen Systemen, die durch nichtlineare bzw. lineare Differenzialgleichungen im Zustandsraum beschrieben werden. Für die linearen Systeme wird auch die Beschreibung durch Übertragungsfunktionen eingesetzt.

Das Bilden der Modelle wird Schritt für Schritt aufgezeigt, wobei Wert auf das Nachvollziehen möglichst vieler Schritte mit Hilfe der Symbolic Math Toolbox gelegt wurde.

Die im fünften Kapitel abgeleiteten Modelle werden in der Mehrzahl der späteren Beispiele immer wieder verwendet. Für jedes Modell existiert eine in MATLAB geschriebene *function*, mit der die Daten der Zustandsmodelle bzw. der Übertragungsfunktionen berechnet und ausgegeben werden können.

Die Gliederung der weiteren Kapitel richtet sich im Wesentlichen nach der Vorgehensweise bei der Vermittlung regelungstechnischer Grundlagen.

Das sechste Kapitel beinhaltet die Möglichkeiten der Beschreibung linearer, zeitinvarianter Systeme im Zeit- und Frequenzbereich.

Das siebente Kapitel hat Testsignale und ihre Zeitantworten zum Inhalt. Die Eigenschaften linearer, zeitinvarianter Systeme und die verschiedenen Arten der Modelltransformation sind Gegenstand des achten Kapitels.

Im neunten Kapitel werden die einzelnen Möglichkeiten des Zusammenschaltens zweier Systeme zu einem Gesamtsystem aufgezeigt, wobei die Beschreibung durch Übertragungsfunktionen und Zustandsmodelle getrennt behandelt wird.

Mein besonderer Dank gilt meinem Sohn, Dipl.-Ing. Stephan Bode, für die wertvollen fachlichen Hinweise und für die gewährte Unterstützung.

Frau Dr. Margit Roth und Herrn Anton Schmid vom Lektorat Buch Mathematik/Informatik/Naturwissenschaften/Technik danke ich für die Möglichkeit dieses Buch beim „Oldenbourg Wissenschaftsverlag“ erscheinen zu lassen und für die gute Zusammenarbeit.

Für die Unterstützung zu Fragen von MATLAB und Simulink bedanke ich mich bei Frau Dipl.-Ing. Desiree Somnitz, Technical Support Engineer von The MathWorks GmbH in Ismaning bei München.

Für die Hilfe bei der Beschaffung von historischem Material zur Automatisierung möchte ich mich u.a. bei Frau Steffi Lange von der Universitätsbibliothek der Otto-von-Guericke-Universität Magdeburg sowie Frau Hirschmann und Frau Carlisi von der Marktbücherei Postbauer-Heng bedanken.

Mein Dank wäre letztlich unvollständig, wenn er nicht meine Frau Rosemarie einschließen würde, für das sicherlich oft schwer aufzubringende Verständnis, meine zeitraubende Beschäftigung, die dieses Buch entstehen ließ, zu akzeptieren.

Für Letzteres danke ich auch unseren Kindern Constance mit Daniel und Stephan mit Manja.

Postbauer-Heng, im August 2009 Helmut Bode

Vorwort zur 2. Auflage

Ich bin dem Oldenbourg Wissenschaftsverlag sehr dankbar, dass er von diesem Buch eine zweite Auflage herausgeben will und mir somit die Möglichkeit gegeben ist, notwendige Korrekturen vorzunehmen. Da mir wiederum in dankenswerter Weise von „The MathWorks, 3 Apple Hill Drive, Natick, MA 01760-2098 USA“ die neuste MATLAB Version: 8.1.0.604 (R2013a) zur Verfügung gestellt wurde, konnte ich jeden M-file – Skripts (Gleichungen), Funktionen, Beispiele – und jede im Command Window ausgeführte Befehlszeile sowie die Signalflusspläne unter Simulink mit der Version R2013a ausführen, so dass alle numerischen und graphischen Aussagen dieses Buches den aktuellen Softwarestand repräsentieren.

Hilfe zu den einzelnen M-files findet sich unter „help *name*“ bzw. „See also *name*“.

Verweisen möchte ich noch auf die im ersten Kapitel zur Historie des Begriffes „Automatisierung“ gemachten Aussagen. Obwohl diese bereits in der ersten Auflage enthalten sind, erscheint mir die gesonderte Erwähnung hier angebracht.

Herrn Dr. Gerhard Pappert – Lektor Mathematik, Informatik, Wissenschaft, Technik – im Oldenbourg Verlag, danke ich für die gute Zusammenarbeit.

Für die schnelle Unterstützung zu Fragen von MATLAB und Simulink bedanke ich mich bei Herrn Dipl.-Ing. B. Hagenreiner und besonders bei Herrn M. Sc. (Mathematik) Christoph Stockhammer, Technical Support Engineer von The MathWorks GmbH in Ismaning bei München.

Mein Dank gilt u.a. auch Frau Anna Lobacheva aus St. Petersburg dafür, dass sie für mich die recht schwierige Aufgabe der Beschaffung von ergänzenden Daten zu den beiden Fußnoten 89 und 90 über das Ehepaar Faddejew / Faddejewa übernommen hat.

Auch für das Verständnis, welches meine Frau Rosemarie wieder für die vielen Stunden, die ich mit der Überarbeitung der zweiten Auflage dieses Buches verbracht habe, aufgebracht hat, sage ich herzlichen Dank.

Postbauer-Heng, den 12. Juni 2013 Helmut Bode

Inhalt

1 Einleitung

Zielstellung dieses Buches ist die Vermittlung von Wissen über die Möglichkeit das dynamische Verhalten technischer Systeme – Anlagen – mit Hilfe von Rechnern[2] so zu simulieren, dass daraus Rückschlüsse auf ihre Steuerbarkeit möglich sind und Entscheidungskriterien für die Automatisierung[3], d.h. für den gefahrlosen, wirtschaftlichen und qualitätsgerechten Betrieb der real existierenden Systeme, gefunden werden.

Das in Abb. 1.1 dargestellte „Konkrete System“ besteht aus einem abgeschlossenen Verband untereinander festverkoppelter Elemente mit Kopplungen zu seiner Umwelt.
Zum näheren Verständnis der im weiteren Verlauf immer wieder verwendeten Begriffe werden diese nachfolgend definiert, was bei dem Auftreten neuer Begriffe fortgesetzt wird.

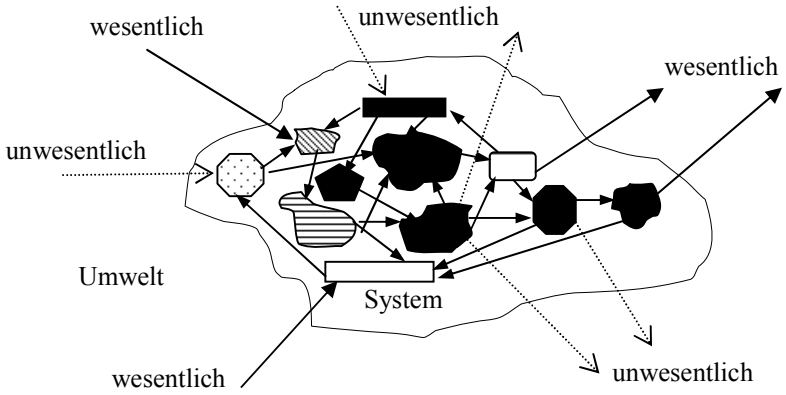

Abb. 1.1 *Konkretes System*

System
Unter einem System ist die Gesamtheit von Objekten (*Elementen*) *zu verstehen, die sich in einem ganzheitlichen Zusammenhang befinden. Durch ihre Wechselbeziehungen untereinander grenzen sie sich gegenüber ihrer Umgebung ab. Die Kopplungen der einzelnen Objekte untereinander sind wesentlich stärker, als die Kopplungen bzw. Wechselwirkungen zur Umwelt.*

---

2 Gewöhnlich auch als Computer bezeichnet.

3 Siehe Seite 3.

Anlage
Die Gesamtheit der maschinellen und anderen Ausrüstungen eines Betriebes, die zur Produktion oder Fertigung (Produktions- oder Fertigungsanlage), zur Energieerzeugung (Kraftanlage), zu Förder- oder Transportzwecken (Förder- oder Transportanlage) u.a. erforderlich sind, werden als Anlage bezeichnet.

Technologischer Prozess
Ein technologischer oder auch technischer Prozess ist ein sich über eine gewisse Zeit erstreckender strukturverändernder Vorgang, bei dem Stoffe, Energien oder Informationen transportiert bzw. umgeformt werden. Ein Prozess läuft in einem konkreten System ab.

Die Simulation ermöglicht es, den in einem bereits vorhandenen bzw. noch zu entwerfenden System ablaufenden technologischen Prozess zu untersuchen. Sie kann dabei unabhängig von dem Gefahrenpotenzial des technologischen Prozesses, seinen materiellen Werten sowie den Geschwindigkeiten des Prozessablaufs durchgeführt werden. Die bei der Simulation gewonnenen Kenntnisse, auf den real existierenden Prozess übertragen, dienen dazu ihn so zu entwerfen, dass er die an ihn gestellten Forderungen erfüllt.

Simulation
Die Simulation eines Systems, gleichgültig welcher Art, erfordert sein physikalisches, technisches oder abstraktes bzw. mathematisches Modell. Die Simulation mit Hilfe eines abstrakten bzw. mathematischen Modells setzt die Analyse des Systems voraus!

Ohne Systemanalyse, kein mathematisches Modell! Ohne mathematisches Modell, keine Computersimulation!

Analyse
Sie beinhaltet die Zerlegung eines Systems in seine Einzelteile bzw. Übertragungsglieder mit dem Ziel, die Wirkungswege der auf das System wirkenden oder im System herrschenden Signale aufzudecken und in ihrem Einfluss auf die Übertragungsglieder im Sinne des statischen und dynamischen Verhaltens des Gesamtsystems möglichst mathematisch zu beschreiben. Der Wirkungszusammenhang lässt sich anschaulich in Signalflussplänen darstellen.

Die hier behandelten Systeme lassen sich durch nichtlineare und lineare bzw. linearisierte mathematische Modelle beschreiben. Vielfach dient das geschaffene mathematische Modell dem Ziel, das dynamische Verhalten des betrachteten Systems – Regelstrecke – durch eine noch zu schaffende Regeleinrichtung – Regler – im Sinne der Automatisierung gezielt zu beeinflussen.

Die Synthese bzw. der Entwurf geeigneter Regler ist Gegenstand der Regelungstechnik. Sie beruht auf dem Prinzip der Rückkopplung. Die einfachste Struktur einer Regelung ist der in Abb. 1.2 dargestellte einschleifige Regelkreis.

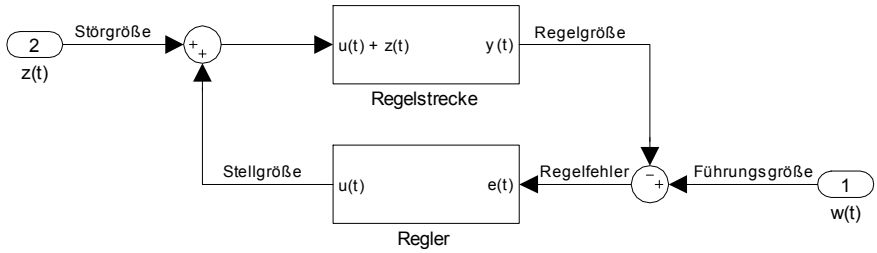

Abb. 1.2 *Der einschleifige Regelkreis, als einfachste Form einer Regelung mit Störung am Streckeneingang*

Der einschleifige Regelkreis besteht aus der Regelstrecke mit der zu regelnden Größe $y(t)$, die ständig gemessen und mit der Führungsgröße $w(t)$ – Sollwert – verglichen wird. Der aus dem Vergleich resultierende Regelfehler $e(t) = w(t) – y(t)$ wird im Regler entsprechend dessen Charakteristik zur Stellgröße $u(t)$ verarbeitet. Mit der Stellgröße ist die Regelstrecke so zu beeinflussen, dass die als Folge einer Störgröße $z(t)$ hervorgerufene Abweichung der Regelgröße kompensiert wird. Voraussetzung für eine Kompensation der Störung ist die Umkehr des Signalvorzeichens im Regelkreis – negative Rückführung.

Die Regelungstechnik gehört neben der Steuerungs- und Nachrichtentechnik, der Physiologie, der Mengenlehre und der mathematischen Logik zu den Wurzeln der Automatisierungstechnik, wobei diese die Mittel und Methoden zur Verfügung stellt, die für die Automatisierung von Prozessen aller Art notwendig sind.

> *Automatisierung*
> *Automatisierung ist das Ersetzen formalisierbarer geistiger Arbeit des Menschen durch technische Mittel zur zielgerichteten Beeinflussung von Prozessen.*

Der Begriff „Automatisierung“ ist im hier verwendeten Sinne zu Beginn des 20. Jh. im deutschen Sprachgebrauch aufgetaucht, so z.B. schreibt Beck im Jahre 1924 in [Beck-1924] u.a. „Für kleinere Kraftwerke gewinnt auch die A u t o m a t i s i e r u n g immer mehr Eingang; so hat die Hydroelectric Power Commission of Ontario den Bau von zwei automatischen Kraftwerken mit je drei Einheiten von 2.000 kVA in Angriff genommen, welche von einem dritten in 6,5 bzw. 10 km Entfernung befindlichen Kraftwerk aus gesteuert werden sollen.“
In den USA wahren die Auswirkungen des Einsatzes von automatischen Maschinen schon länger in der Diskussion, etwa seit den späten Achtzigerjahren des 19. Jahrhunderts so Ernest F. Lloyd in [Lloyd-1919]. In seinem Artikel „The American Automatic Tool“ führt er u.a. aus:

> *„Machinery may be roughly divided into two main groups, the first comprising machines whose principal purpose is to strengthen the arm of the worker, the second comprising those whose purpose is to supplant the worker or reduce his function to a minimum. All machines falling within the second group may be termed automatic tools.“*

> *Maschinenanlagen können etwa in zwei Hauptgruppen eingeteilt werden. Die erste beinhaltet Maschinen, deren prinzipieller Zweck es ist, den Arm des Arbeiters zu stärken, zu der zweiten gehören diejenigen, deren Zweck es ist, den Arbeiter zu ersetzen oder seinen Einfluss auf ein Minimum zu reduzieren. Alle Maschinen welche in die zweite Gruppe fallen, können „automatic tools" genannt werden.*

Arthur Pound zitiert in „The Iron Man in Industry" [Pound-1922] die Ausführungen von E. F. Lloyd über die zwei Hauptgruppen von Maschinenanlagen in einer etwas veränderten und erweiterten Form. Aus „Machinery" wird „Machine-tools" und die Bezeichnung ‚automatic tools' für die zweite Hauptgruppe von Maschinen taucht, aus nicht nachvollziehbaren Gründen, nicht mehr auf. Weiterhin wird in dem Zitat der Arm des Arbeiters nicht nur *gestärkt* sondern auch *verlängert* und *sein Willen als die wesentlichste Arbeitsfunktion* bleibt erhalten.

Pound führt einen Auslegerkran als Beispiel für die erste Maschinenart an. Für dessen Bedienung der Kranführer seine Muskeln und seinen Geist in gleicher Weise einsetzen muss, wie seine Vorfahren es taten, um ihre einfachen Hebel zu bewegen. Die erste Maschinenart entspricht der weiter unten angeführten Definition der Mechanisierung.

Die primäre Aufgabe der Maschinen der zweiten Gruppe ist, die Arbeit selbst auszuführen, wofür ihr Mechanismus erdacht ist. Somit ist es nicht erforderlich, dass der Bediener die einzelnen Schritte des Arbeitsablaufes kennt. Seine Aufgabe besteht lediglich in der Zuführung des Rohmaterials und dem Abtransport der fertigen Produkte. Die zweite Maschinenart entspricht der o. a. Definition der Automatisierung.

Arthur Pound schreibt, dass es selbstverständlich ist, dass der Grad der selbsttätig funktionierenden (automatischen) Maschinen nicht bei allen gleich hoch ist.

In der deutschen Ausgabe von Pounds Buch [Pound/Witte-1925] übersetzt die Bearbeiterin Irene M. Witte den Begriff „Machine-tools" mit „Werkzeugmaschine", was wohl in diesem Zusammenhang den Bereich der automatischen Maschinen stark einschränkt.

Schon zum Beginn der Zwanzigerjahre des letzten Jahrhunderts war in den USA klar, das sich die Gesellschaft in Richtung einer vollständigen Automatisierung entwickelt. Dazu ein Auszug aus dem III. Kapitel – Mind and Machine – von [Pound/Witte-1925] Seite 39/40, im Original Seite 37:

> *Das Bestreben des stets auf Neuerungen bedachten Menschen, sich satt zu essen, sich zu kleiden und seine anderen Bedürfnisse mit der geringsten Anstrengung zu befriedigen, hat im Laufe der Jahrhunderte in durchaus folgerichtiger Entwicklung die automatische Maschine hervorgebracht, die zuerst mit Widerstreben, dann aber in großen Mengen für die Produktion der notwendigen Erzeugnisse eingeführt wurde. Unsere Generation befindet sich auf dem Wege zur vollständigen Automatisierung* – its way to a complete automatization[4] *–, d.h. zu einer so vollständigen, wie sie die menschliche Natur überhaupt ertragen kann.*

---

4 vom Verfasser nach dem Original eingefügt

1931 schreibt K. Piche in „Die Automatisierung von Wasserkraftwerken“ [Piche-1931] „wie sich in Österreich der Gedanke, elektrische Wasserkraftanlagen aus wirtschaftlichen Erwägungen zu automatisieren, entwickelt hat und auch durchzusetzen beginnt.“ Er führt u.a. aus, dass „Mit der Steigerung der Ansprüche an Qualität der erzeugten und gelieferten Energie ...“ selbsttätig wirkende Regulatoren bzw. Regler für die verschiedensten Aufgaben im Kraftwerksbetrieb auf der mechanischen und elektrischen Seite „gebaut und deren Konstruktion verfeinert und schrittweise zur heutigen Vollkommenheit gebracht“ wurden. Was im Sinne der obigen Definition für die Automatisierung besonders relevant ist, beschreibt er wie folgt:

> *So wurde das Erfassen und Abwehren einer Störungsart nach der anderen dem Bedienungspersonal abgenommen und Apparaten übertragen und so immer ein weiterer Schritt zur A u t o m a t i s i e r u n g gemacht, ohne daß man vorläufig daran dachte, auf das Bedienungspersonal ganz zu verzichten.*

Weitere Aussagen zu diesem Thema finden sich u.a. über die Automatisierung des Fernsprechnetzes in [Vossische Zeitung-1932] und in [Dolezalek-1938] zur Automatisierung der Mengenfertigung. Kuhnert wurde mit der Dissertation zum Thema „Der Prozess der Automatisierung und Mechanisierung und seine Einwirkung auf den schaffenden Menschen“ [Kuhnert-1935] promoviert.

Nach [Nevins/Hill u.a.-1962] hat Harder[5] 1947 bei der Ford Motor Company, bei der er bis Mai 1962 tätig war, ein „Automation Department“ eingerichtet, hierbei verwendete er den Begriff „automation“, das englischsprachige Äquivalent zur deutschen Automatisierung.

D. S. Harder arbeitete in der Autoindustrie, u.a. bei der Yello Taxi Cab Company, der Chevrolet Company und der General Motors Corporation, ehe er zur Ford Motor Company wechselte.

Zu etwa der gleichen Zeit wie D. S. Harder, führte John Theurer Diebold[6] an der Harvard Business School ebenfalls das Wort „automation“ ein. Siehe auch [Diebold-1956].

In der Folge beschäftigten sich mehrere Autoren mit der Definition der Begriffe „Mechanisierung“ und „Automatisierung“, so z.B. [Kindler[7]-1957], [Roeper-1958] und sehr ausführlich Kortum in [Kortum-1961] und [Kortum-1962].

Materialien zur Beurteilung der ökonomischen und sozialen Folgen der Automation sind in [Pollock-1964] enthalten, wobei grundlegende Begriffe der Automatisierung z. T. nicht, ihrer Bedeutung entsprechend, auseinander gehalten werden.

In der Soziologie und Sozialpolitik taucht dieser Begriff schon im Jahre 1908 bei Max Weber[8] [Weber, Max-1908] auf. Er formuliert in Hinsicht auf eine Verbesserung der Kräfteöko-

---

5 Harder, Delmar S. *19.3.1892 Delhi, NY (USA) †September 1973 (USA), ab 1912 Ingenieur im Automobilbau

6 Diebold, John Theurer *8.6.1926 Weehawken, NJ †26.12.2005 Bedford Hills, NY (USA), Computervisionär

7 Kindler, Heinrich *29.11.1909 Breslau †23.02.1985 Dresden, Prof. für Regelungstechnik

8 Weber, Max *21.4.1864 Erfurt †14.7.1920 München, Nationalökonom, Sozialwissenschaftler

nomie eines Leistungsträgers: „»Körperlicher« und »geistiger« Arbeit gemeinsam ist in dieser Hinsicht vor allem der Vorgang der »Mechanisierung«, »Automatisierung[9]« möglichst vieler, anfänglich in allen ihren Einzelheiten durch gesondert bewußtwerdenden Willensimpuls und unter konstanter Inanspruchnahme der Aufmerksamkeit vollzogenen Bestandteile der Leistung.“, so dass „»Uebung« von Arbeitsleistungen stets wesentlich auch eine »Automatisierung« von ursprünglich im Bewußtsein artikulierten Willensimpulsen ist“.
Neben dieser psychophysischen[10] Automatisierung, spricht Weber noch von der „maschinellen Automatisierung“, welche im Zusammenhang mit der Arbeitszeitreduktion in den einzelnen Industrien steht, wobei er u.a. ausführt: „Namentlich das vielumstrittene Problem, inwieweit die zunehmende A u t o m a t i s i e r u n g des Arbeitsprozesses und die damit verbundene Ausschaltung des Einflusses der Leistung der Arbeiter auf das Maß der Intensität der Motoren- und Maschinenausnützung dem Satz: kurze Arbeitszeit = hohe Arbeitsintensität, Schranken setzt, entbehrt noch einer zugleich streng unbefangenen Erörterung ...“. Auch über den Vorgang der Mechanisierung im Zusammenhang mit körperlicher und geistiger Arbeit spricht Weber noch von der Rhythmisierung der Arbeit als Mittel der Mechanisierung.

Kurt Tucholsky[11] verwendet in seinem Beitrag „Der Bahnhofsvorsteher“ [Tucholsky-1924] den Begriff „Automatisierung des Betriebes“, worunter er eine immer wiederkehrende Tätigkeit versteht, so dass der Tätige die damit im Zusammenhang stehenden Eindrücke nicht mehr wahrnimmt, d.h. er wird zur Maschine. Für den Tätigen werden die Eindrücke zum Klipp-Klapp eines Automaten. Er sagt „Ich glaube, dass man sich mit der Automatisierung des Betriebes die besten Eindrücke verdirbt.“! Die hier gemeinte Automatisierung entspricht der o. a. „psychophysischen“ Automatisierung Webers. In seinem Beitrag „Berlin und die Provinz“ [Tucholsky-1928] schreibt er „Eine Mechanisierung, eine Automatisierung des Lebens hat eingesetzt ...“ geht aber nicht weiter auf diese Begriffe ein.

Der Große Duden [Duden-1929] führt „Automatisierung“ erstmals in seiner 10. Auflage von 1929 auf.

Die Vorstufe auf dem Weg zur Automatisierung ist die Mechanisierung.

> *Mechanisierung*
> *Die Mechanisierung beinhaltet die Übergabe schwerer körperlicher, gesundheitsschädlicher und zeitraubender Arbeiten des Menschen an Maschinen, welche die Befehle für die Ausführung ihrer Operationen vom bedienenden Menschen erhalten.*

Die Mittel und Methoden zur Lösung technischer Probleme im Sinne einer Automatisierung gehen im Wesentlichen auf die weitgehend allgemeingültigen Methoden und Betrachtungsweisen der *Kybernetik* zurück.

---

9 Sie könnte als „psychophysische“ Automatisierung bezeichnet werden.

10 psychophysisch → Psychophysik: Wissenschaft von den Wechselbeziehungen des Physischen und des Psychischen, von den Beziehungen zwischen Reizen und ihrer Empfindung [Duden-1989]

11 Tucholsky, Kurt *9.1.1890 Berlin †21.12.1935 Hindås (Göteborg), Schriftsteller

Die Kybernetik ist aus der Tatsache entstanden, dass bei den verschiedensten Wissenschaftsdisziplinen, wie z.B. der Mathematik, Technik, Biologie, Psychologie, Soziologie, Ökonomie, immer wieder analoge Probleme und Gesetzmäßigkeiten auftreten, die eine übergeordnete Wissenschaft vermuten lassen. Für diese übergeordnete Wissenschaft prägten Norbert Wiener[12] und Vertreter aus seinem wissenschaftlichen Umfeld den Begriff „Cybernetics" bzw. Kybernetik, was in Wieners 1948 erstmalig erschienenem Buch „Cybernetics or Control and Communication in the Animal and the Machine", in Deutsch „Kybernetik – Regelung und Nachrichtenübertragung in Lebewesen und Maschine" [Wiener-1968], anschaulich beschrieben ist.

Die Kybernetik hat sich trotz oder wegen ihres integrierenden Charakters zwischen den einzelnen Wissensgebieten nicht als eine selbständige, übergeordnete Disziplin durchsetzen können, da der notwendige Wissensumfang für die *Kybernetiker* viel zu umfangreich sein würde. Die in der Blütezeit der Kybernetik als ihre modernste und leistungsfähigste Errungenschaft postulierten universell programmierbaren Analog- und Digitalrechner [Peschel-1972], die den Menschen von routinemäßiger geistiger Arbeit befreien, sind Gegenstand der Informatik, einer selbständigen Wissenschaft, ohne die eine Automatisierung heute nicht mehr denkbar ist.

Kybernetik
Die Kybernetik stellt eine allgemeine Systemtheorie dar. Ihr Anwendungsbereich ist die gesamte objektive Realität, in der Begriffe wie System, Information, Signal, wesentliche und unwesentliche Kopplungen sowie Einflussgrößen auftreten.

Zur Konkretisierung des oben definierten Systems werden nachfolgend die Kopplungen zwischen den Objekten eines Systems sowie zwischen dem System und der Umwelt beschrieben.

Wesentliche und unwesentliche Kopplungen
Für die Modellbildung wesentliche Kopplungen sind die Informationen, die von Signalen getragen, von einem Objekt oder System zum anderen gelangen und zum Kontrollieren, Lenken und Leiten der Prozesse genutzt werden können.

Kopplungen, welche den Austausch von Energien und Stoffen realisieren, werden für die Modellbildung als unwesentlich betrachtet.

Für die Information als wesentliche Kopplung soll nachfolgende kurze Definition gelten:

Information
Eine Information ist eine Mitteilung über das Eintreten eines Ereignisses. Durch sie wird Ungewissheit über dieses Ereignis beim Empfänger der Information beseitigt. Der Informationsgehalt ist umso höher, je unbestimmter das Ereignis vorherzusehen war. Jede physikalische Größe kann in der Form eines Signals als Träger von Informationen dienen.

---

12 Wiener, Norbert *26.11.1894 Columbia (USA) †18.3.1964 Stockholm, Mathematiker, Kybernetiker

Der Verlauf eines Parameters bestimmt den Inhalt einer Information. Die für die Übertragung einer Information notwendigen Merkmale eines Signals, d.h. seine Werte oder sein Werteverlauf, heißen Informationsparameter.

Ein Signal muss mittels technischer Einrichtungen erfasst, verarbeitet, genutzt und übertragen sowie gespeichert werden können, so dass die in ihm enthaltenen Informationen in eindeutiger Weise reproduzierbar sind.

Die Signale lassen sich in Nutz- und Störsignale unterteilen. Im Sinne der Automatisierung wird ein Nutzsignal entweder am Eingang eines Systems zum Steuern des Prozesses verwendet, oder es verlässt als gesteuertes Signal das System über den Ausgang.

Störsignale sind von weit reichender Bedeutung, denn sie beeinflussen in negativer Weise den zielgerichteten Ablauf eines Prozesses, so dass in ihrer Beseitigung vielfach der Grund für eine Automatisierung liegt. Sie werden im Allgemeinen als Störungen oder Störgrößen bezeichnet.

Gründe die eine Automatisierung notwendig machen, sind neben der Beseitigung von Störungen, das Betreiben von Prozessen, die auf Grund der großen Änderungsgeschwindigkeiten ihrer Systemgrößen vom Menschen alleine nicht beherrscht werden können oder von denen eine Gefahr für Leben und Gesundheit der Bedienenden ausgeht.

Das Ziel der Automatisierung von Prozessen ist es, sie mit einem Höchstmaß an Wirtschaftlichkeit, Sicherheit und Zuverlässigkeit betreiben zu können sowie den Menschen weitgehend von Routinearbeiten zu entlasten.

Der Kompliziertheitsgrad und die in den Automatisierungsobjekten (Abb. 1.3), gespeicherten Energien sowie Sachwerte steigen bei Weiterentwicklungen überproportional an. Dies bedeutet, dass die Prozesse immer komplexer werden und ihr Gefahrenpotenzial für Mensch und Umwelt steigt. Damit wird der mit der Bedienung und Betreuung dieser Systeme beauftragte Personenkreis vor Aufgaben gestellt, die nur mit den Mitteln und Methoden der Automatisierungstechnik bewältigt werden können.

Die Kategorien Stoff, Energie, Information und Störung entsprechend Abb. 1.3 sind die Basis für die Betrachtungen an einem Prozess, der zu automatisieren ist.

Automatisierungstechnik
Ihre Aufgabe ist es, die vorliegenden allgemeingültigen Methoden und Gesetzmäßigkeiten der Automatisierung in die Praxis zu überführen, so dass sich gut handhabbare Verfahren für die Analyse bestehender Automatisierungsobjekte ebenso wie für die Synthese von Automatisierungssystemen ergeben.

Synthese
Sie beinhaltet den Entwurf der Automatisierungsanlage mit dem Ziel das gewünschte statische und dynamische Verhalten – Stabilität, Einschwingverhalten, Genauigkeit, Sicherheit, Wirtschaftlichkeit usw. – des Gesamtsystems unter Beachtung der auftretenden bzw. zu erwartenden Störungen zu erreichen.

Neben der Unmöglichkeit einer Gesamtoptimierung wird oft die Chance vergeben, einem Automatisierungsobjekt schon während der Phasen des Entwurfs und der Konstruktion ein günstiges automatisierungstechnisches Verhalten zu geben. Da vielfach der Automatisierungstechniker oder speziell der Regelungstechniker erst mit dem Entwurf beauftragt wird, wenn der Prozess bereits in seiner Grundkonzeption vorliegt, bleibt ihm nur noch die Möglichkeit vorgegebene Strukturen zu analysieren und daran den Regler bzw. den Regelalgorithmus anzupassen, was keinesfalls optimal sein wird.

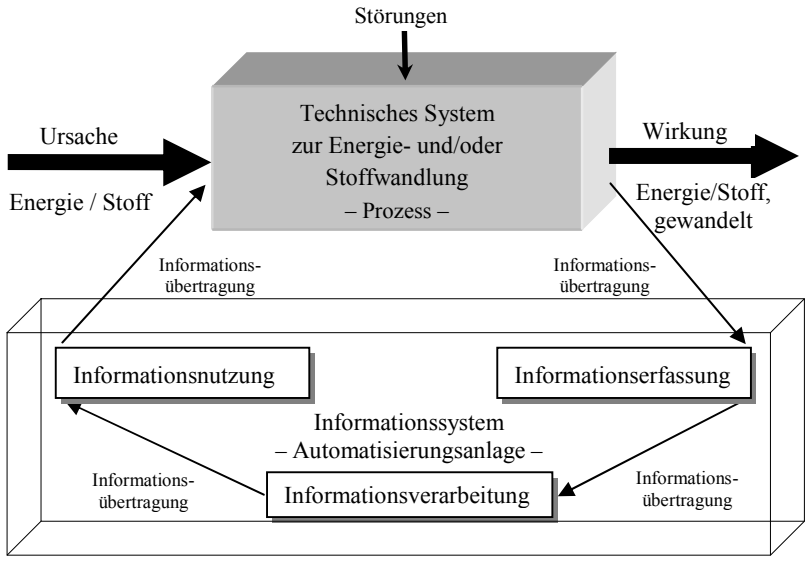

Abb. 1.3 *Wechselwirkungen zwischen einem System zur Energie- und/oder Stoffwandlung und einem System zur Informationswandlung*

Die nachfolgenden Kapitel sollen Hinweise und Anregungen geben, wie über die Analyse[13] dynamischer Systeme [Oppelt[14]-1964], [Reinisch[15]-1979] u.a. zum Zwecke der Modellbildung und dem Einsatz der Regelungstheorie[16] im Zusammenhang mit der leistungsfähigen Software MATLAB[17] und Simulink[18] die Voraussetzungen für eine fundierte Synthese des Informationssystems bzw. der Automatisierungsanlage entsprechend Abb. 1.3 geschaffen werden können.

---

13 Auch als theoretische Prozessanalyse bezeichnet.

14 Oppelt, Winfried *5.6.1912 Hanau †1.10.1999 Baldham/München, Prof. für Regelungstechnik in Darmstadt

15 Reinisch, Karl *21.08.1921 Dresden †24.01.2007 Ilmenau, Prof. für Regelungstechnik

16 Schmidt, Hermann *9.12.1894 Hanau †31.5.1968 Berlin, wurde am 25.11.1944 an die TH Berlin-Charlottenburg auf den ersten Lehrstuhl für Regelungstechnik in Deutschland berufen, Physiker, Mathematiker, Regelungstechniker [Jäger-1996] [Fasol-2001].

17 MATLAB® ist ein eingetragenes Warenzeichen der Firma „The MathWorks“.

18 Simulink® ist ein eingetragenes Warenzeichen der Firma „The MathWorks“.

2 Einführung in MATLAB

MATLAB ist eine leistungsfähige Software zur numerischen Lösung und graphischen Darstellung von wissenschaftlich-technischen Aufgabenstellungen. Der Name MATLAB steht für MATrix LABoratory. Eine Vielzahl mathematischer Funktionen bilden die Grundlage, die mittels leistungsfähiger Bausteine, wie z.B. der *Control System Toolbox* und ihrer vielen Erweiterungen sowie den Toolboxen *System Identification*, Version 8.2 (R2013a), *Signal Processing*, Version 6.19 (R2013a), *Fuzzy Logic*, *Neuronal Network* und die über die MuPAD[19] symbolic engine betriebene *Symbolic Math Toolbox,* gestartet mit *mupad* usw., fachspezifisch ergänzt wird.

MATLAB ist ein interaktives Programm, mit dem es neben der Vielzahl gelieferter M-functions möglich ist, eigene, mit Hilfe des *M-file*-Editors geschriebene *functions*, zu nutzen.
Die nachfolgenden Ausführungen erheben keinesfalls den Anspruch, alle Kommandos bzw. M-functions, die MATLAB beinhaltet, zu behandeln. Hierfür wird auf die Hilfefunktionen „*help name*“ bzw. „*doc name*“ und auf den *Function Browser* unter *Help* im *Command Window* verwiesen. Aus der zwischenzeitlich zahlreichen und vorwiegend anwendungsorientierten Literatur seien hier stellvertretend genannt: [Angermann u.a.-2011], [Benker-2003], [Glattfelder/Schaufelberger-1997], [Gramlich/Werner-2000], [Löwe-2001], [Lutz/Wendt-2010], [Paul-2004], [Pietruszka-2005], [Scherf-2003], [Schott-2004], [Schweizer-2013], [Überhuber u.a.- 2004].

Die in diesem Buch verwendete Software bezieht sich auf MATLAB Version 8.1 (R2013a), Simulink Version 8.1 (R2013a), Control System Toolbox Version 9.5 (R2013a), und auf die Symbolic Math Toolbox Version 5.10 (R2013a).

Direkte Aufzeichnungen von MATLAB-Routinen in Microsoft[20] Word 2002, 2003, 2007, und 2010 unter Microsoft Windows XP Home Edition können mit dem dafür vorgesehenen MATLAB Notebook erfolgen.

---

19 MuPAD, Version 5.10.0 (R2013a), ist ein Computeralgebra System (CAS) der Firma SciFace Software GmbH und Co. KG, jetzt Bestandteil von „The MathWorks“. Es ist ein Programm mit dem sich mathematische Problemstellungen bearbeiten lassen.

20 Microsoft® ist ein eingetragenes Warenzeichen der Firma „Microsoft Corporation“.

2.1 Eingaben

Bei der Verwendung von MATLAB zur Lösung numerischer und graphischer Aufgabenstellungen können sowohl die Daten als auch die Anweisungen direkt über die Tastatur oder indirekt über eine Skript-Datei – *script* – bzw. Funktions-Datei – *function* – eingegeben werden.

2.1.1 Direkte Eingabe

Jede direkte Eingabe – Anweisung – im MATLAB Command Window beginnt hinter dem Prompt „>>“ und wird durch Betätigen der Return-Taste „↵“ abgeschlossen, was zu ihrer Verarbeitung führt.

2.1.2 Der MATLAB-Editor

Die direkte Eingabe ist für kürzere Kommandofolgen geeignet. Längere oder immer wiederkehrende Folgen werden besser mit Hilfe von M-Dateien über einen Editor eingegeben.

Der MATLAB-Editor wird zum Schreiben von Skript-Dateien und zum Schreiben von anwenderspezifischen *functions* genutzt. Die „Namen“ der scripts und functions sind mit der Extension „*.m*“ versehen. Sie werden folglich auch als *M-Dateien* bzw. *M-files* bezeichnet.

Der MATLAB-Editor dient neben dem Schreiben von M-files auch ihrer Editierung und Fehlerbeseitigung. Kommentare, Schlüsselwörter, numerische Befehlsfolgen und „strings“ werden in unterschiedlichen Schriftfarben übersichtlich dargestellt. Die Abarbeitung der Befehlsfolgen erfolgt zeilenweise. Eine über eine Zeile hinausgehende Befehlsfolge kann durch „ … “ getrennt werden, so dass der in der nächsten Zeile befindliche Teil als zugehörig zum vorhergehenden Teil interpretiert wird. Für die Speicherung dieser M-functions empfiehlt es sich ein eigenes Verzeichnis einzurichten, welches nicht im MATLAB-Verzeichnis angesiedelt sein muss. Unter MATLAB ist im Current Directory nur der Path, unter dem sich das eigene Verzeichnis befindet, einzutragen.

2.1.3 Indirekte Eingabe über Skript-Dateien

Soll bei verschiedenen MATLAB-Sitzungen mit gleichen Beispielen bzw. Werten gerechnet werden, so bietet es sich an, die dazu notwendigen Daten und gegebenenfalls Berechnungen in einer Skript-Datei abzulegen. Wird diese unter Command Window nach dem Prompt durch Eingabe „Name“ und Betätigen der Return-Taste gestartet, so stehen die einmal eingegebenen notwendigen Daten und gegebenenfalls Berechnungen zur Verfügung.

Die indirekte Eingabe erfolgt über eine Skript-Datei in der die Variablenzuweisungen und die durchzuführenden Rechenoperationen aufgeführt werden. Dies geschieht mit Hilfe des MATLAB-Editors.

Beispiel 2.1
Zur Berechnung eines Zylindervolumens über die indirekte Eingabe ist ein M-file gesucht.
Lösung:[21]

```
% Beispiel 2.1
%    Berechnung des Volumens eines Zylinders
% mit dem Durchmesser d = 1 m und der Höhe h = 1 m.
% Vereinbarung der Werte
    d = 1;
    h = 1;
% Kreisfläche
    A = pi*d^2/4;
% Volumen
    V = A*h;
disp('            ')
disp('************************************************')
disp('              Lösung zum Beispiel 2.01')
disp('            Das Volumen eines Zylinders'),
fprintf(...
'  mit dem Durchmesser d = %d m und der Höhe h = %d m\n',...
    d,h)
fprintf('              beträgt V = %f m³!\n',V)
% Ende des Beispiels 2.1
************************************************
              Lösung zum Beispiel 2.01
            Das Volumen eines Zylinders
  mit dem Durchmesser d = 1 m und der Höhe h = 1 m
              beträgt V = 0.785398 m³!
```

2.1.4 Indirekte Eingabe über Funktionsdateien

Mit einer Funktionsdatei definiert ein Anwender seine Funktion. Dieser Funktion werden in einer Argumentliste Werte zur Berechnung neuer Werte übergeben, die die Funktion an die aufrufende Prozedur, welche auch das Command Window sein kann, zurückgibt. Die Funktionsdatei hat folgenden festen Aufbau:

```
function [1. Wert, ..., n-ter Wert] = Funktionsname (Argumentliste)
% Kommentar
Anweisungen
```

Die Funktionsdatei wird aufgerufen mit:

```
[1.Wert,~, …, n-ter Wert] = Funktionsname (Argumentliste):
```

Mit dem Aufruf:

```
help Funktionsname
```

wird der „Kommentar“ ausgegeben, falls vorhanden.

---

[21] Published with MATLAB® R2013a → Editor: „Beispiel2_01.m“ → PUBLISH → Publish → Edit Publishing Options ... → Edit Configurations: „Beispiel2_01.m“ → MATLAB expression: Beispiel2_01 → Publish → Web Browser-Fenster mit: „Beispiel2_01.html“ bzw. „Beispiel2_01.pdf“ ... usw.

2.1.5 Kommandos im Zusammenhang mit function

nargin, nargout	Anzahl der Eingabe- bzw. Ausgabeparameter
persistent var1 var2 …	Definiert in einer Funktion lokale Variable, die bei ihrem wiederholten Aufruf mit dem vorher festgelegten oder berechneten Wert belegt sind, siehe Beispiel 3.7.
isempty ('name')	Testet, ob eine Variable „empty“, also leer ist.

Beispiel 2.2
Es sind die Summe der Zahlen von 1 bis *n* und die ausgeführte Anzahl der Rechenschritte in Form einer *function* [S, k] = *summe*(*n*) gesucht. Es gilt: n = 100.

Lösung:[22]

```
function [S, k] = summe(n)
% Berechnet die Summe der Zahlen von 1 bis n und gibt die
% Anzahl der Rechenschritte aus. Entspricht Beispiel 2.2.
%
% Vereinbarung der Variablen
    i = 1; A(i) = 1; naus = nargout;
if nargin < 1, naus = 3; n = 0; end
% Berechnungen
for i = 2:(n+1), A(i)= A(i-1)+i; end
switch naus
    case 1
        if n > 1
            S = A(i-1);
        else
            S = A(1);
        end
    case 2
        if n > 1
            k = i-1;
            S = A(k);
        else
            k = n;
            S = A(n);
        end
    case 3
        error('Es fehlt die einzugebende Variable ''n''!')
end
disp('            ')
disp('************************************************')
disp('             Lösung zum Beispiel 2.2')
% Ende der Funktion summe
************************************************
             Lösung zum Beispiel 2.2
S =
    5050
k =
     100
```

[22] Published with MATLAB® R2013a → Editor: „summe.m: → PUBLISH → Publish → Edit Publishing Options ... → Edit Configurations: „summe.m“ → MATLAB expression: [S,k] = summe(100) → Publish → Web Browser-Fenster mit: „summe.html“ bzw. „summe.pdf“ ... usw..

2.2 Kommandos, Operationen, Werte, Funktionen

Nachfolgend werden auf den Inhalt dieses Buches zugeschnittene, nützliche Kommandos, Operationen, Werte und Funktionen aufgeführt.

2.2.1 Nützliche Kommandos

% Text	Leitet einen Kommentar ein.
Anweisung …	Die Anweisung wird in der nächsten Zeile fortgesetzt.
cd C:\matlab\eig	Wahl des Verzeichnis „eig“ in C:\MATLAB.
clear	Löscht alle Variablen aus dem Arbeitsspeicher.
clear name	Löscht die Variable „name“ aus dem Arbeitsspeicher.
delete datei	Löscht den file „datei“ aus dem aktuellen Verzeichnis.
diary datei	Alle folgenden Ein- und Ausgaben im Command Window werden in die Datei „datei“ geschrieben. Extension: z.B. .txt, .doc oder .m.
diary off	Beendet das Schreiben und schließt die aktuelle Datei. Durch Entfernen der Ergebnisse und Ersetzen der Extension „.txt“ bzw. „.doc“ durch „.m“ entsteht ein lauffähiger M-file.
diary on	Öffnet die zuletzt geschlossene Datei und schreibt weiter in sie.
dir / ls	Listet den Inhalt des aktuellen Verzeichnisses auf.
disp ('Text')	Gibt den 'Text' aus.
echo on bzw. off	Ausgabe im Command Window bzw. ihre Verhinderung.
format „Zahlenformat“	Legt das auszugebende „Zahlenformat“ fest.
format compact	Unterdrückt die Leerzeile bei der Ausgabe.
fprintf ('Text % x.yf\n', Wert)	Gibt 'Text' und einen oder mehrere numerische Werte aus. „x“: Länge des Feldes, „y“: Stellenzahl, „f“: Format
fzero(fun,x0)	Berechnet von „fun“ im Bereich von „x0“ die Nullstelle.
help „function"	Gibt den zur „function“ gehörenden Text aus.
load Daten.mat	Lädt die in Daten.mat gespeicherten Variablen in den Arbeitsspeicher.
save Daten.mat	Speichert alle im Arbeitsspeicher hinterlegten Variablen in Daten.mat.
save Daten.mat a b	Speichert die Variablen „a“ und „b“ in Daten.mat.
type „name“	Zeigt den Inhalt der Datei „name.m“ an.
Variable;	Das Semikolon verhindert die Ausgabe des Inhaltes von Variable.
what „Verzeichnis“	Gibt die im „Verzeichnis“ befindlichen Files an.
which „name“	Pfad und Verzeichnis, für die Datei „name.m“.
who	Liste mit den aktuellen Variablen im Arbeitsspeicher.
whos	Ausgabe einer erweiterten Variablenliste.

2.2.2 Grundoperationen mit den Variablen a und b

Operation	Command Window	
	Eingabe	Ausgabe
Zuweisung	>> a = 4; b = 5;	
Addition	>> s = a + b	s = 9
Subtraktion	>> d = a - b	d = -1
Multiplikation	>> m = a * b	m = 20
Division von rechts	>> r = a / b	r = 0.8000
Division von links	>> l = a \ b	l = 1.2500
Potenz	>> p = a^b	p = 1024
Quadratwurzel	>> w2 = sqrt(p)	w2 = 32
n-te Wurzel	>> wn = nthroot (s,a)	wn = 1.7321

2.2.3 Spezielle Werte und Variable, Cell Arrays

Beschreibung	Command Window	
	Eingabe	Ausgabe
ans: ist keine Ergebnisvariable angegeben, so wird das Resultat „ans“ zugewiesen	>> a + b	ans = 9
inf: ∞, infiniti	>> d0 = a/0	Warning: Divide by zero. d0 = Inf
i, j: imaginäre Einheit	>> i	ans = 0 + 1.0000i
NaN: Not-a-Number	>> nn = Inf/Inf	nn = NaN
pi: π	>> format long, fl = pi	fl = 3.14159265358979
{ }: Cell Array	>> A{1}=[1 4 3;0 5 8;7 2 9]; >> A{2}= 'cell array'; >> celldisp (A)	A{1} = 1 4 3 0 5 8 7 2 9 A{2} = cell array

2.2.4 Auswahl häufig benötigter Funktionen

Funktion	Command Window	
	Eingabe	Ausgabe
Binominalkoeffizient: $\binom{n}{k} = \frac{n!}{(n-k)!\,k!}$ nchoosek(n,k)	>> nchoosek(4,2)	ans = 6
E-Funktion e^x: exp(x)	>> e1 = exp(1)	e1 = 2.7183
Fakultät n!: factorial(n)	>> factorial(4)	ans = 24
Logarithmus, dekadischer, lg (x): log10(x)	>> lg1 = log10(e1)	lg1 = 0.4343
Logarithmus, natürlicher, ln(x): log(x)	>> lg = log(e1)	lg = 1
Zahl, rundet +Zahlen ab, –Zahlen auf fix(x)	>> fip = fix(e1) >> fim = fix(-e1)	fip = 2 fim = -2
Zahl, rundet ab zur nächsten ganzen floor(x)	>> fp = floor(e1) >> fm = floor(-e1)	fp = 2 fm = -3
Zahl, rundet zur nächsten ganzen round(*x*)	>> rp = round(e1) >> rm = round(-e1)	rp = 3 rm = -3

2.2.5 Operationen mit komplexen Zahlen

Funktion	Command Window	
	Eingabe	Ausgabe
komplexe Zahl: z	>> z = a + b*i	z = 4.0000 + 5.0000i
absoluter Wert von z, |z|: abs(z)	>> Z = abs(z)	Z = 6.4031
Winkel von z im Bogenmaß, arg(z): angle(z)	>> phi = angle(z)	phi = 0.8961
Winkel von z in Grad	>> phi_g = phi*180/pi	phi_g = 51.3402
konjugiertkomplexe Zahl zu z conj(z)	>> zc = conj(z)	zc = 4.0000 - 5.0000i
imaginärer Teil von z imag(z)	>> zi = imag(z)	zi = 5
realer Teil von z real(z)	>> zr = real(z)	zr = 4

2.2.6 Trigonometrische Funktionen

Zur Erläuterung der nachfolgend aufgeführten trigonometrischen Funktionen für Winkel beliebiger Größe, d.h. auch > 90°, wird das kartesische Koordinatensystem verwendet.

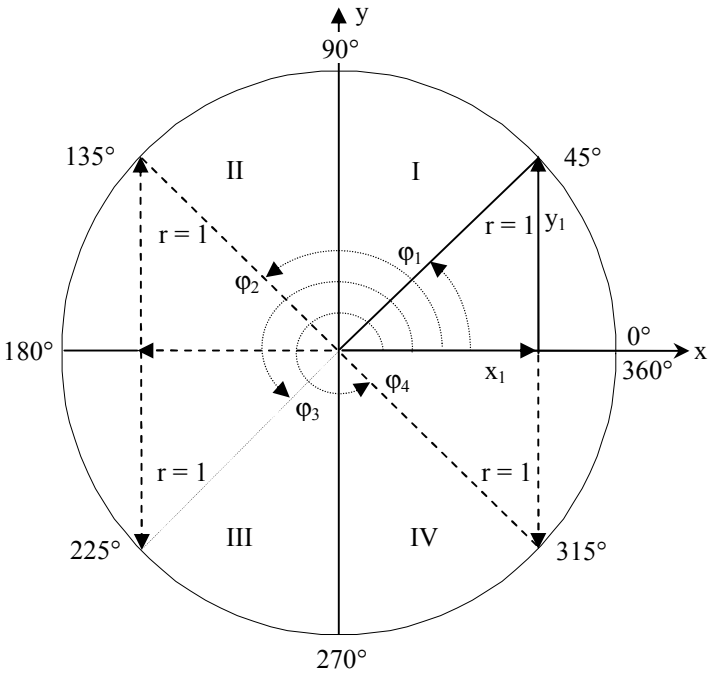

Abb. 2.1 *Definition der trigonometrischen Funktionen am Einheitskreis [KE-Mathe-1977]*

Sein Drehsinn ist dem Uhrzeigersinn entgegengesetzt gerichtet, welcher als der mathematisch positive – Linkssystem – bezeichnet wird. Durchläuft der freie Schenkel eines Winkels φ alle vier in Abb. 2.1 dargestellten Quadranten, so verändert sich seine Größe von 0° bis 360° bzw. im Bogenmaß von 0 bis 2π. Der freie Schenkel, er entspricht dem Radius des Einheitskreises r, bildet die Hypotenuse eines rechtwinkligen Dreiecks mit dem unveränderlichen Wert +1. Die Ankathete x sowie die Gegenkathete y verändern dagegen Länge und Vorzeichen beim Durchlauf. Mit den Definitionen der vier trigonometrischen Funktionen:

$$\sin\varphi = \frac{y}{r} \qquad \cos\varphi = \frac{x}{r} \qquad \tan\varphi = \frac{y}{x} \qquad \cot\varphi = \frac{x}{y} \tag{2.1}$$

ergeben sich die Vorzeichen der Funktionswerte in den vier Quadranten:

Funk-	Quadrant			
tion	I	II	III	IV
sin φ	+	+	–	–
cos φ	+	–	–	+
tan φ	+	–	+	–
cot φ	+	–	+	–

MATLAB stellt dazu folgende Funktionen bereit, wobei die Eingabe der Winkel im Bogenmaß oder in Grad möglich ist:

Trigonometrische MATLAB-Funktionen	
Winkel phi in rad	Winkel Phi in Grad
w = sin (phi)	w = sind (Phi)
w = cos (phi)	w = cosd (Phi)
w = tan (phi)	w = tand (Phi)
w = cot (phi)	w = cotd (Phi)

Die trigonometrischen Funktionen sind periodisch:

$$\left.\begin{aligned} \sin\left(\varphi° + n\cdot 360°\right) &= \sin\left(\widehat{\varphi} + n\cdot 2\pi\right) = \sin\varphi \\ \cos\left(\varphi° + n\cdot 360°\right) &= \cos\left(\widehat{\varphi} + n\cdot 2\pi\right) = \cos\varphi \\ \tan\left(\varphi° + n\cdot 180°\right) &= \tan\left(\widehat{\varphi} + n\cdot \pi\right) = \tan\varphi \\ \cot\left(\varphi° + n\cdot 180°\right) &= \cot\left(\widehat{\varphi} + n\cdot \pi\right) = \cot\varphi \end{aligned}\right\} \quad n = 0, \pm 1, \pm 2, \ldots \qquad (2.2)$$

d.h. die Sinus- und Kosinusfunktion wiederholen sich nach dem Durchlaufen aller vier, dagegen die Tangens- und Kotangensfunktion schon nach dem Durchlauf von 2 Quadranten.
Die trigonometrischen Funktionen ordnen jedem beliebigen Winkel φ eindeutig einen Funktionswert zu. Vielfach tritt aber der Fall auf, dass für einen gegebenen Funktionswert der Winkel zu bestimmen ist. Hierfür werden die Umkehr- oder Arkusfunktionen verwendet.

Umkehr- bzw. Arkusfunktionen in MATLAB	
Ergebnis: Winkel in rad	Ergebnis: Winkel in Grad
phi = asin (w)	Phi = asind (w)
phi = acos (w)	Phi = acosd (w)
phi = atan (w)	Phi = atand (w)
phi = acot (w)	Phi = acotd (w)

So beschreibt z.B. φ = arcsin(w) den Bogen – Arkus – dessen Sinus den Wert w hat. Durch Spiegelung an der Winkelhalbierenden des I. Quadranten, siehe Abb. 2.1, ergeben sich die Verläufe der Arkusfunktionen aus denen der trigonometrischen, d.h. für Bereiche, in denen die Funktionen monoton verlaufen und alle Funktionswerte annehmen – Monotonie-Intervalle – sind die trigonometrischen Funktionen umkehrbar.

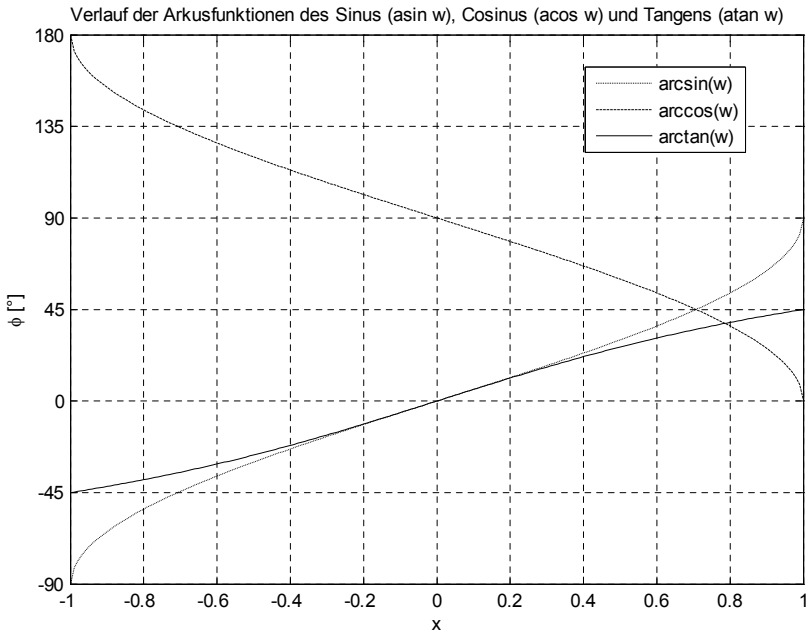

Abb. 2.2 *Verlauf der Arkusfunktionen, berechnet mit den M-functions asin(w), acos(w) und atan(w)*

Für die Monotonie-Intervalle ergeben sich verschiedene Wertevorräte der Arkusfunktionen mit den Hauptwerten:

$$\begin{aligned} -\tfrac{1}{2}\pi &\leq Arc\sin w \leq +\tfrac{1}{2}\pi \\ 0 &\leq Arc\cos w \leq +\pi \\ -\tfrac{1}{2}\pi &< Arc\tan w < +\tfrac{1}{2}\pi \\ 0 &< Arc\cot w < +\pi \end{aligned} \tag{2.3}$$

MATLAB hat für den häufig auftretenden Fall einer Umkehrfunktion des Tangens eine weitere Arkustangensfunktion geschaffen, die wie folgt definiert ist:

$$\text{phi} = \text{atan2 (Y,X)} \tag{2.4}$$

Die Argumente Y und X in der Gleichung (2.4) entsprechen den Katheten y und x in der Abb. 2.1. Die Funktion liefert für Werte, welche in den Quadranten I und II liegen, die Winkel $0 \leq \varphi \leq +\pi$ im Bogenmaß, d.h. $0° \leq \Phi \leq +180°$. Für Werte welche in den Quadranten III und IV liegen, die Winkel $0 \leq -\varphi \leq -\pi$, d.h. $0° \leq -\Phi \leq -180°$ für $180° < \Phi \leq 360°$.

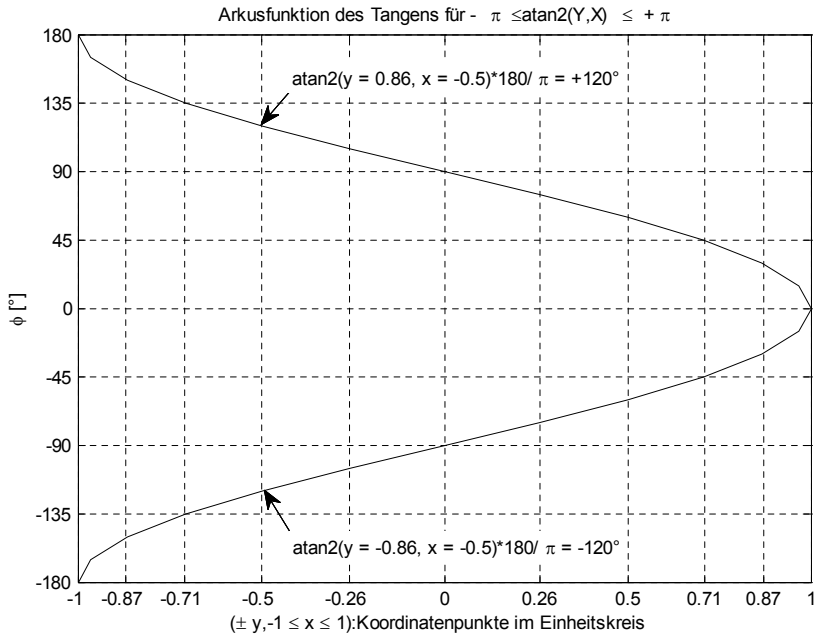

Abb. 2.3 *Verlauf der mit der M-function atan2(Y,X) berechneten Winkel –180° ≤ φ ≤ +180°*

Skript 2.1 zur Darstellung und Berechnung der trigonometrischen Funktionen

```
% Skript2_01.m zum Erzeugen der Abb. 2.02 und Abb. 2.03
% Berechnung und Darstellung der trigonometrischen Funktion

% Figur 1 -  Sinus- und Cosinus-Funktionen, nicht im Buch dargestellt!
phi = 0:1:360;
wc = cosd(phi); ws = sind(phi);
figure(1), plot(phi,ws,'-.',phi,wc,'--'), grid
% Größe der Figur 1
set(1,'PaperPosition',[0.6345    6.3452   18.0   13.5])
a = gca;
set(a,'XTickLabel',{0 45 90 135 180 225 270 315 360})
set(a,'xlim',[0 360])
set(a,'XTick',[0 45 90 135 180 225 270 315 360])
xlabel('\phi [°]'), ylabel('w')
title('Verlauf der Sinus- und Cosinusfunktion')
legend('sin(\phi)','cos(\phi)','Location','best')

% Abb. 2.02
% Figur 2 - Arkusfunktionen des Sinus, Cosinus und Tangens
x = -1:0.01:1;
Phis = asind(x); Phic = acosd(x); Phit = atand(x);
figure(2)
% Größe der Figur 2
set(2,'PaperPosition',[0.6345    6.3452   18.0   13.5])
plot(x,Phis,':k',x,Phic,'--k',x,Phit,'k'), grid
b = gca;
```

```
set(b,'YTickLabel',{-90 -45 0 45 90 135 180})
set(b,'ylim',[-90 180])
set(b,'YTick',[-90 -45 0 45 90 135 180])
ylabel('\phi [°]'), xlabel('x')
title(...
['Verlauf der Arkusfunktionen des Sinus (asin w), ',...
    'Cosinus (acos w) und Tangens (atan w)'])
legend('arcsin(w)','arccos(w)','arctan(w)','Location','best')
% Speichern als Abb2_02.emf
print -f2 -dmeta -r300 Abb2_02

% Abb. 2.03
% Figur 3 - atan2(Y,X)
Phi = -180:15:0; Y = sind(Phi); X = cosd(Phi);
k0 = 0.26; k1 = 0.5; k2 = 0.71; k3 = 0.87;
figure(3)
% Größe der Figur 3
set(3,'PaperPosition',[0.6345    6.3452   18.0   13.5])
plot(X,atan2(Y,X)*180/pi,'k',-X,atan2(abs(Y),-X)*180/pi,'k')
grid
c = gca; f = gcf;
set(c,'ylim',[-180 180])
Phi1 = -180:45:180;
set(c,'YTick',Phi1), set(c,'xlim',[-1 1])
set(c,'XTick',[-1 -k3 -k2 -k1 -k0 0 k0 k1 k2 k3 1])
title(...
    ['Arkusfunktion des Tangens für - \pi  \leq',...
    'atan2(Y,X)  \leq  + \pi'])
ylabel('\phi [°]')
xlabel(['(\pm y,-1 \leq x \leq 1):',...
    'Koordinatenpunkte im Einheitskreis '])
annotation(f,'textarrow',[0.3563 0.3235],[0.8443 0.7912],...
    'TextEdgeColor','none',...
    'TextColor',[0 0 0],...
    'TextBackgroundColor',[1 1 1],...
    'String',{'atan2(y = 0.86, x = -0.5)*180/\pi = +120°'});
annotation(f,'textarrow',[0.3563 0.3268],[0.1795 0.2473],...
    'TextEdgeColor','none',...
    'TextColor',[0 0 0],...
    'TextBackgroundColor',[1 1 1],...
    'String',{'atan2(y = -0.86, x = -0.5)*180/\pi = -120°'});
% Speichern als Abb2_03.emf
print -f3 -dmeta -r300 Abb2_03

% Berechnungen
Phi = [45 135 225 315];
Yb = [0 0.71 1  0.71  0 -0 -0.71 -1 -0.71 0];
Xb = [1 0.71 0 -0.71 -1 -1 -0.71  0  0.71 1];
F = [Phi; sind(Phi); cosd(Phi); tand(Phi); cotd(Phi)];
P = [Phi;asind(F(2,:));acosd(F(3,:)); atand(F(4,:));...
    atand(F(5,:))];
T = atan2(Yb,Xb)*180/pi;
disp('               Die Funktionswerte für')
disp('      sind(Phi), cosd(Phi), tand(Phi) und cotd(Phi)')
fprintf('      Phi %4.0f° %6.0f° %6.0f° %6.0f°\n', F(1,:))
fprintf('      sin %+4.4f %+4.4f %+4.4f %+4.4f\n', F(2,:))
fprintf('      cos %+4.4f %+4.4f %+4.4f %+4.4f\n', F(3,:))
fprintf('      tan %+4.4f %+4.4f %+4.4f %+4.4f\n', F(4,:))
fprintf('      cot %+4.4f %+4.4f %+4.4f %+4.4f\n', F(5,:))
disp(' ')
```

```
disp('                    Die Funktionswerte für')
disp('          asind(w), acosd(w), atand(w) und acotd(w)')
fprintf(['Ausgangswerte:     Phi', ...
    '    %4.0f° %4.0f° %4.0f° %4.0f°\n'], P(1,:))
fprintf(['Berechnete Werte:',...
    '  asind %+5.0f %+5.0f %+5.0f %+5.0f\n'], P(2,:))
fprintf(['                 ',...
    '  acosd %+5.0f %+5.0f %+5.0f %+5.0f\n'], P(3,:))
fprintf(['                 ',...
    '  atand %+5.0f %+5.0f %+5.0f %+5.0f\n'], P(4,:))
fprintf(['                 '...
    '  acotd %+5.0f %+5.0f %+5.0f %+5.0f\n'], P(5,:))
disp(' ')
Q = [1 1 1 2 2 3 3 4 4 4];
disp('  Die mit atan2(Y,X) berechneten Funktionswerte')
disp('  ------------------------------------------')
fprintf('     \t(Y,X)\t\t atan2(Y,X)\t Quadrant\n')
disp('  ------------------------------------------')
fprintf('     (%5g,%5g)\t %5g°\t\t\t%d\n',[Yb;Xb;T;Q])
% Ende des Skript2_01
```

2.3 Matrizen

Die Matrix[23, 24] ist ein rechteckiges Zahlenschema von *m* Zeilen und *n* Spalten. Die Zahlen, aus denen sich die Matrizen zusammensetzen, werden ihre *Elemente* genannt. Die Elemente können reelle oder komplexe Zahlen sein:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix} \tag{2.5}$$

Der erste Index *i* des Elements a_{ij} gibt die Zeile und der zweite Index *j* die Spalte an, in der dieses Element steht. Ist die Anzahl der Zeilen *m* gleich der Anzahl der Spalten *n*, so liegt eine quadratische Matrix vom Typ (n,n) vor.

In MATLAB wird zwischen Klein- und Großschreibung der Buchstaben unterschieden, so dass es sich anbietet, für Matrizen große und für Vektoren kleine Buchstaben zu verwenden.

---

[23] Sylvester, James Joseph *3.9.1814 London †15.3.1897 London, Mathematiker. Der Begriff „Matrix“ wurde 1850 von ihm eingeführt. [Dietrich/Stahl-1963]

[24] Cayley, Arthur *16.8.1821 Richmond, England †26.1.1885 Cambridge, Jurist und Mathematiker; verfasste 1858 die Theorie der Matrizen [Wußing u.a.-1992].

2.3.1 Matrizen und die Eingabe ihrer Elemente

Der allgemeinste Datentyp unter MATLAB ist eine Matrix vom Typ (*m*,*n*). Sie stellt ein zweidimensionales Feld – Array – dar. Für die Eingabe der Elemente der Matrix **A** im Command Window sind zwei Vorgehensweisen möglich:

1. Die Elemente werden hintereinander aufgeführt und durch ein Leerzeichen oder Komma getrennt. Der letzte Koeffizient einer Zeile wird mit einem *Semikolon* abgeschlossen. Vor dem ersten Element der ersten Zeile öffnet eine eckige Klammer, welche nach dem letzten Element der letzten Zeile schließt, hier entfällt das Semikolon für den Zeilenschluss:

$$\mathbf{A} = \left[a_{11} \quad a_{12} \quad \ldots \quad a_{1n}; \quad \ldots \quad ; \quad a_{m1} \quad a_{m2} \quad \ldots \quad a_{mn}\right] \tag{2.6}$$

2. Die Elemente werden hintereinander aufgeführt und durch ein Leerzeichen oder Komma getrennt. Nach dem letzten Koeffizienten jeder Zeile wird die Return-Taste ↵ betätigt. Vor dem ersten Element öffnet eine eckige Klammer, welche nach dem letzten Element geschlossen wird:

$$\begin{aligned} \mathbf{A} = \; & \left[a_{11} \quad a_{12} \quad \cdots \quad a_{1n} \quad \hookleftarrow\right. \\ & \ldots \quad \ldots \quad \cdots \quad \ldots \quad \hookleftarrow \\ & \left.a_{m1} \quad a_{m2} \quad \cdots \quad a_{mn}\right] \end{aligned} \tag{2.7}$$

Beispiel 2.3

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 0 & 11 & 12 \end{bmatrix}$$

Die Matrix A ist nach den beiden Möglichkeiten von MATLAB für die Eingabe von Matrizen, wie durch die Gleichungen (2.6) und (2.7) beschrieben, einzugeben.

Lösung:
Die Matrix **A** ist vom Typ (4,3) ≙ (*m* = 4 Zeilen, *n* = 3 Spalten).

1. Möglichkeit	2. Möglichkeit
>> A = [1 2 3;4 5 6;7 8 9;0 11 12]	>> A = [1 2 3 4 5 6 7 8 9 0 11 12]
Beide Eingabemöglichkeiten liefern das gleiche Ergebnis:	
A = 1 2 3 4 5 6 7 8 9 0 11 12	

Ist in einer Matrix mindestens ein Element eine Dezimalzahl, so werden alle Elemente in Dezimalform ausgegeben!

Beispiel 2.4
Die Matrix $\mathbf{A}_d$ = [11 12 13;14 15 16;17 18 19;10 1,1 –2] ist entsprechend Gleichung (2.6) unter Beachtung des Wechsels zwischen Dezimalkomma und Dezimalpunkt einzugeben.
Lösung:

```
>> Ad = [11 12 13;14 15 16;17 18 19;10 1.1 -2]
Ad =
      11.0000   12.0000   13.0000
      14.0000   15.0000   16.0000
      17.0000   18.0000   19.0000
      10.0000    1.1000   -2.0000
```

2.3.2 Eigenschaften einer Matrix

Der Typ einer Matrix

Matrizen sind durch ihren Typ (*m*,*n*), d.h. die Anzahl der Zeilen *m* und die Anzahl der Spalten *n*, gekennzeichnet. Die M-function zur Bestimmung des Typs einer Matrix lautet:

[mA,nA] = size(A) (2.8)

Beispiel 2.5
Gesucht sind die Anzahl der Zeilen und Spalten, d.h. der Typ (*m*,*n*), der Matrix **A**.

Lösung:

```
>> [mA,nA] = size(A)
mA =
      4
nA =
      3
```

Die Matrix **A** ist vom Typ (4,3), d.h. sie besteht aus 4 Zeilen und 3 Spalten.

Quadratische Matrizen

Quadratische Matrizen sind vom Typ (*n*,*n*) bzw. *n-ter* Ordnung. Sie haben bei der Beschreibung von linearen dynamischen Systemen im Zustandsraum, als *Systemmatrizen*, eine fundamentale Bedeutung. Mit der Systemmatrix **A** sind die Begriffe *charakteristisches Polynom*, *Eigenwert* und *Eigenvektor* verbunden.

Ausführliche Betrachtungen zu diesem Thema werden im Kapitel 8.3 angestellt.

Beispiel 2.6
Aus der Matrix **A** ist durch Streichen der dritten Zeile sowie der vierten und fünften Spalte eine quadratische Matrix **A***q* zu bilden.

Lösung:

```
>> Aq = A([1:2 4],1:3)
Aq =
     1     2     3
     4     5     6
     0    11    12
```

Die mögliche Anzahl quadratischer Untermatrizen einer Matrix

Die Anzahl *u* der quadratischen Untermatrizen vom Typ (*r*,*r*) einer Matrix vom Typ (*m*,*n*) berechnet sich mit Hilfe des *Binominalkoeffizienten*:

$$\binom{n}{k} = \frac{n!}{(n-k)!\,k!} \tag{2.9}$$

aus dem Produkt der *Kombinationen ohne Wiederholung* der *m* Zeilen und der *n* Spalten:

$$u = \binom{m}{r}\binom{n}{r} = \frac{m!\,n!}{(m-r)!\,(n-r)!\,(r!)^2} \tag{2.10}$$

Mit der M-function *nchoosek* berechnet sich die Anzahl *u* der quadratischen Untermatrizen:

$$u = \text{nchoosek(m,r)} * \text{nchoosek(n,r)} \tag{2.11}$$

Die Determinante einer quadratischen Matrix

Einer quadratischen Matrix vom Typ (*n*,*n*) kann eine Zahl zugeordnet werden, die ihre *Determinante* D_A heißt und aus den Elementen der Matrix **A** wie folgt gebildet wird:

$$D_A = \det \mathbf{A} = |\mathbf{A}| = \begin{vmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{vmatrix} = a_{1j}A_{1j} + a_{2j}A_{2j} + \cdots + a_{nj}A_{nj} \tag{2.12}$$

mit:

$$A_{kl} = (-1)^{k+l} D_{kl} \tag{2.13}$$

wobei D_{kl} diejenige Determinante $(n-1)$-*ter* Ordnung ist, die durch Streichen der *k-ten* Zeile und der *l-ten* Spalte aus *D* hervorgeht. Mit der M-function:

$$\text{d = det(A)} \tag{2.14}$$

wird der Wert der Determinante einer quadratischen Matrix berechnet.

Singuläre und nichtsinguläre quadratische Matrizen

Eine quadratische Matrix heißt *singulär*, wenn der Wert ihrer Determinante gleich null ist:

$$|\mathbf{A}| = 0 \tag{2.15}$$

Für diesen Fall kann die Inverse der Matrix nicht gebildet werden. Ist dagegen:

$$|\mathbf{A}| \neq 0 \tag{2.16}$$

dann heißt die quadratische Matrix *nichtsingulär* und ihre Inverse existiert.

Der Rang einer Matrix

Der Rang einer Matrix vom Typ (*m*,*n*) ist die Ordnungszahl derjenigen quadratischen Untermatrix höchster Ordnung, deren Determinante ungleich null ist. Er genügt folgender Ungleichung:

$$r(\mathbf{A}) \leq min(m,n) \tag{2.17}$$

und berechnet sich mit Hilfe der M-function:

$$\text{r = rank(A)} \tag{2.18}$$

Beispiel 2.7
Gesucht sind der Typ, der Rang und die Anzahl der quadratischen Untermatrizen vom Typ (*r*,*r*) der Matrix **M** = [1 –2 –3 0;2 3 8 7;–1 1 1 –1][25] sowie der Wert der Determinante der Untermatrizen M(1:3,1:3) und M(1:2,1:2). Für die Lösung ist ein M-Skript zu schreiben!

Lösung:

```
% Beispiel 2.7
% Gesucht sind der Typ(m,n), der Rang r und die Anzahl der quadratischen
% Untermatrizen vom Typ(r,r) der Matrix M
M = [1 -2 -3 0;2 3 8 7;-1 1 1 -1]; eval('M')
% sowie der Wert der Determinanten der Untermatrizen M3 = M(1:3,1:3) und
% M2 = M(1:2,1:2)
% Typ
[m,n] = size(M); fprintf('Typ(m,n) = (%d,%d)\n',m,n)
% Rang
r = rank(M); fprintf('Rang r = %d\n',r)
% Wert der Determinante der Untermatrizen M(1:2,1:3) und M(1:2,1:2)
dM3 = det(M(1:3,1:3));
fprintf('Wert der Determinante von M3(1:3,1:3) = %1.0f\n',dM3)
dM2 = det(M(1:2,1:2));
fprintf('Wert der Determinante von M2(1:2,1:2) = %d\n',dM2)
% Anzahl der quadratischen Untermatrizen
nuM = nchoosek(m,r)*nchoosek(n,r); fprintf('Aus der Matrix M lassen sich ')
fprintf('%d quadratische Untermatrizen bilden!\n',nuM)
% Ende des Beispiels 2.7
```

Die Ergebnisse sind auf der folgenden Seite dargestellt.

---

[25] nach [Bronstein/Semendjajew-1991]