

Stefan Adam

MATLAB und Mathematik kompetent einsetzen

Eine Einführung für Ingenieure
und Naturwissenschaftler

Zweite Auflage

```
% Figur starten
```

```
figure(1); clf;
```

```
% Gitter der Variablen-Werte und Funktion Z(X,Y)
```

```
[X,Y] = meshgrid(-0.1:0.1:2.8,0:0.2:4.8);
```

```
Z = (sin(X)+0.18979*cos(5*X)).*sin(Y).*exp(-0.35*Y)+0.5;
```

```
% 3D Darstellung; weiter zeichnen
```

```
surf(X,Y,Z); hold on
```

```
% Rahmen-Viereck
```

```
levx = [X(1,1) X(25,1) X(25,30) X(1,30) X(1,1)];
```

```
levy = [Y(1,1) Y(25,1) Y(25,30) Y(1,30) Y(1,1)];
```

```
plot3(levx,levy,0.5*ones(5,1),'k'); plot(levx,levy,Z)
```

```
% Markier-Kreuze der stationaeren Punkte
```

```
xlinx = [-0.08 0.08 0 0 0]; xliny = [0 0 0 -0.1 0.1];
```

```
% lokale (anonyme) Funktion definieren und aufrufen
```

```
putmrk = @(x,y) plot(xlinx+x,xliny+y,'k','linewidth',2)
```

```
putmrk(0.327,1.235); putmrk(1.3114,1.235); putmrk(2.307,1.235);
```

```
putmrk(1.969,1.235); putmrk(1.5114,4.375);
```

```
% 3D Achsen-Box Bereiche, mit Kamerawinkel
```

```
axis([-0.3 3 -0.4 5.2 0 1.2]); grid off; view(-58, 29);
```

```
% zugehoerige Hoehenlinien-Graph
```

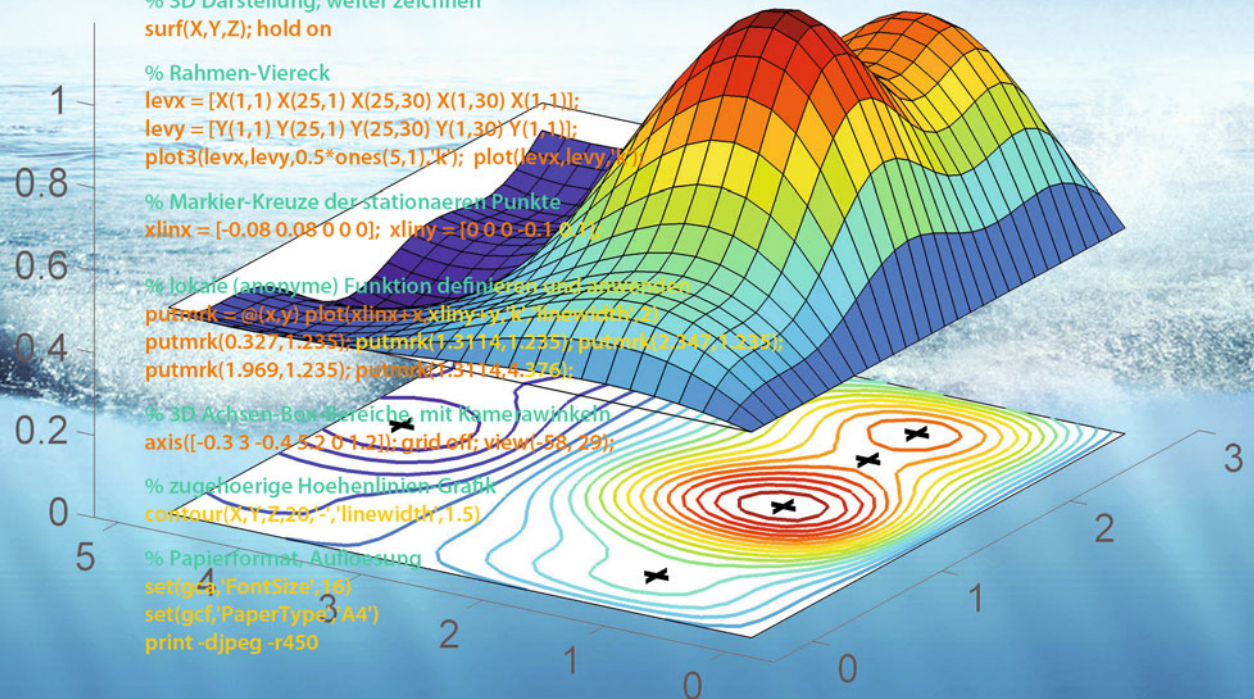
```
contour(X,Y,Z,20,'-','linewidth',1.5)
```

```
% Papierformat, Aufloessung
```

```
set(gcf,'FontSize',16)
```

```
set(gcf,'PaperType','A4')
```

```
print -djpeg -r450
```



Stefan Adam

**MATLAB und Mathematik
kompetent einsetzen**

Stefan Adam

MATLAB und Mathematik kompetent einsetzen

Eine Einführung für Ingenieure und Naturwissenschaftler

2. Auflage

WILEY-VCH
Verlag GmbH & Co. KGaA

Autor

Stefan Adam

Rue des Parcs 53
2000 Neuchâtel
Schweiz

2. Auflage 2017

■ Alle Bücher von Wiley-VCH werden sorgfältig erarbeitet. Dennoch übernehmen Autoren, Herausgeber und Verlag in keinem Fall, einschließlich des vorliegenden Werkes, für die Richtigkeit von Angaben, Hinweisen und Ratschlägen sowie für eventuelle Druckfehler irgendeine Haftung.

**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.

© 2017 WILEY-VCH Verlag GmbH & Co. KGaA,
Boschstr. 12, 69469 Weinheim, Germany

Alle Rechte, insbesondere die der Übersetzung in andere Sprachen, vorbehalten. Kein Teil dieses Buches darf ohne schriftliche Genehmigung des Verlages in irgendeiner Form – durch Photokopie, Mikroverfilmung oder irgendein anderes Verfahren – reproduziert oder in eine von Maschinen, insbesondere von Datenverarbeitungsmaschinen, verwendbare Sprache übertragen oder übersetzt werden. Die Wiedergabe von Warenbezeichnungen, Handelsnamen oder sonstigen Kennzeichen in diesem Buch berechtigt nicht zu der Annahme, dass diese von jedermann frei benutzt werden dürfen. Vielmehr kann es sich auch dann um eingetragene Warenzeichen oder sonstige gesetzlich geschützte Kennzeichen handeln, wenn sie nicht eigens als solche markiert sind.

Umschlaggestaltung Adam Design, Weinheim
Typesetting le-tex publishing services GmbH,
Leipzig

Print ISBN 978-3-527-41262-4
ePDF ISBN 978-3-527-68028-3
ePub ISBN 978-3-527-68026-9
Mobi ISBN 978-3-527-68027-6
obook ISBN 978-3-527-80170-1

Gedruckt auf säurefreiem Papier

Dieses Buch widme ich meinen Leserinnen und Lesern!

Ich wünsche Ihnen viel Elan und Entdeckerfreude beim Eintauchen in die Welt von MATLAB und Mathematik!

Inhaltsverzeichnis

Vorwort *XV*

1	Grundkenntnisse von MATLAB	<i>1</i>
1.1	Bekannschaft schließen mit MATLAB	<i>1</i>
1.1.1	Die Arbeitsoberfläche von MATLAB	<i>1</i>
1.1.2	Zum Einstieg: Berechnungen mit einfachen Zahlen	<i>2</i>
1.1.3	Befehlsstruktur: ein erster Überblick	<i>4</i>
1.1.4	Berechnung oder Formel-Manipulation?	<i>6</i>
1.1.5	Tabellen, Vektoren und Matrizen	<i>11</i>
1.1.6	Hintergrundinformation und Hilfsfunktionen	<i>13</i>
1.1.7	Datenaustausch mit Files	<i>15</i>
1.2	Grundlagen der Matrizenrechnung	<i>20</i>
1.2.1	Definitionen und Fachausdrücke	<i>20</i>
1.2.2	Indizieren der Matrixelemente	<i>24</i>
1.2.3	Das Transponieren einer Matrix	<i>24</i>
1.2.4	Addition und Subtraktion von Matrizen	<i>25</i>
1.2.5	Das Produkt von zwei Matrizen	<i>26</i>
1.2.6	Die Einheitsmatrix	<i>30</i>
1.2.7	Kann man durch Matrizen dividieren?	<i>31</i>
1.3	Matrizenrechnung mit MATLAB	<i>33</i>
1.3.1	Einstieg in die Matrizenrechnung mit MATLAB	<i>33</i>
1.3.2	Indizieren in MATLAB	<i>37</i>
1.3.3	Beispiele zur Schleifenprogrammierung	<i>39</i>
1.3.4	Turmmatrizen (Permutationsmatrizen)	<i>40</i>
1.3.5	Einfache Beispiele von linearen Gleichungssystemen	<i>42</i>
1.3.6	Matrizen zur Darstellung von Daten	<i>43</i>
1.4	Schritte zum eigenen Programm	<i>46</i>
1.4.1	Skript-M-Files und Funktions-M-Files	<i>46</i>
1.4.2	Objekt-Orientiertes Programmieren	<i>52</i>
1.5	Einfache grafische Darstellungen mit MATLAB	<i>56</i>
1.5.1	Funktionsdarstellungen	<i>57</i>
1.5.2	Polygone, Kreise, Sterne	<i>60</i>
1.5.3	Flächen malen	<i>62</i>

1.5.4	Properties von grafischen Objekten	64
1.6	Übersicht über die wichtigsten Grundbefehle in MATLAB	65
1.6.1	In MATLAB definierte Operatoren und Grundbefehle	65
1.6.2	Das Definieren von Zahlen, Matrizen und Vektoren	68
1.6.3	Schleifen und Bedingungen	70
1.6.4	Mathematische Funktionen	71
1.6.5	Grundfunktionen im symbolischen Modus	72
1.6.6	„struct“- und „cell“-Variablen	73
1.6.7	Grafische Darstellungen	74
1.7	MATLAB Grundlagen aktivieren	76
2	Auffrischen der Elementarmathematik	91
2.1	Basiswissen zum Funktionsbegriff	91
2.1.1	Funktionen als spezielle Relationen	91
2.2	Linienplots in MATLAB	96
2.2.1	Grundfunktionen kennenlernen mit MATLAB	97
2.2.2	Kurven in Parameterdarstellung	100
2.2.3	Spiralen	102
2.2.4	Zykloiden	103
2.2.5	Weitere Mathematische Klassiker	106
2.2.6	Die „Versiera di Agnesi“	107
2.2.7	Interpolationsfunktionen	110
2.2.8	Ausflug ins Dreidimensionale	114
2.3	Folgen und Reihen	116
2.3.1	Arithmetische Folgen und Reihen	117
2.3.2	Geometrische Folgen und Reihen	119
2.3.3	Die Anwendung bei Zinsberechnungen	122
2.3.4	Beherrschbare Unendlichkeit	125
2.3.5	Fibonacci-Folgen	128
2.4	Keine Angst vor komplexen Zahlen!	129
2.4.1	Die Rechenregeln für komplexe Zahlen	130
2.4.2	Die n -ten Einheitswurzeln	134
2.4.3	Die n -ten Wurzeln aus beliebigen Zahlen	135
2.4.4	Komplexe Zahlen näher kennenlernen	136
2.4.5	Beschreibung von stationären Schwingungen	138
2.5	Elementarmathematik aktivieren	141
3	Basiswissen zur Linearen Algebra	151
3.1	Lineare Gleichungssysteme und ihre Lösbarkeit	151
3.1.1	Gleichungssystem und zugehörige Matrixgleichung	151
3.1.2	Die verschiedenen Fälle der Lösbarkeit	152
3.1.3	Die Bedingungen zur eindeutigen Lösbarkeit – Regularität	152
3.1.4	Die wichtigsten Fachausdrücke der Lösbarkeitsdiskussion	153
3.1.5	Lineare Abhängigkeit von Vektoren	154
3.1.6	Lineare Systeme und ihre Teilräume	158

- 3.1.7 Die Determinante einer Matrix 160
- 3.2 Anwendungen von linearen Gleichungssystemen 162
 - 3.2.1 Gleichungssysteme aus Tabellenkalkulationen 162
 - 3.2.2 Kirchhoff'sche Netze 163
 - 3.2.3 Statik von Tragwerken 166
 - 3.2.4 Dünn besetzte Matrizen 169
 - 3.2.5 Polynombestimmung 169
- 3.3 Orthogonalität und Projektionen 171
 - 3.3.1 Orthogonale Vektoren 171
 - 3.3.2 Projektionen von Vektoren 173
 - 3.3.3 Orthogonale Teilräume 175
 - 3.3.4 Orthogonale Matrizen 175
- 3.4 Lösungsverfahren für lineare Gleichungssysteme 177
 - 3.4.1 Die Bedeutung der Dreiecksmatrizen 177
 - 3.4.2 Der Gauß-Algorithmus 177
 - 3.4.3 Der Gauß-Algorithmus mit MATLAB 180
 - 3.4.4 Das Vertauschen von Zeilen: Pivot-Suche 181
 - 3.4.5 Die L-R-Zerlegung 183
 - 3.4.6 Der Gauß-Jordan-Algorithmus 185
 - 3.4.7 Singuläre Systeme 185
 - 3.4.8 Die Q-R-Zerlegung 187
- 3.5 Eigenwerte und Eigenvektoren 190
 - 3.5.1 Definition von Eigenwerten und Eigenvektoren 190
 - 3.5.2 Wiederholte Abbildungen durch Matrizen 192
 - 3.5.3 Lösungsmethoden für Eigenwertprobleme 193
 - 3.5.4 Stabilität von Systemen 196
- 3.6 Probleme mit der endlichen Rechengenauigkeit 197
 - 3.6.1 Die Zahlendarstellung im Computer 197
 - 3.6.2 Auslöschung, Stabilität und Wohldefiniertheit 201
 - 3.6.3 Die Kondition einer Matrix 203
 - 3.6.4 Die Option `digits` 204
- 3.7 Lineare Algebra aktivieren 205

- 4 Ebenen- und Raumgeometrie 217**
 - 4.1 Vektoren in der Elementargeometrie 217
 - 4.1.1 Addition und Subtraktion von Vektoren 218
 - 4.1.2 Produkte zwischen Vektoren 220
 - 4.2 Beispiele aus der Raumgeometrie 222
 - 4.2.1 Geometrische Grundelemente 222
 - 4.2.2 Geometrische Grundaufgaben 226
 - 4.2.3 Anwendungsbeispiele 231
 - 4.3 Längen und Winkel in höheren Dimensionen 232
 - 4.4 Matrixformulierung geometrischer Abbildungen 236
 - 4.5 Abbildungen in homogenen Koordinaten 240
 - 4.5.1 Das Prinzip der homogenen Koordinaten 240

4.5.2	Homogene Koordinaten in der Ebene	240
4.5.3	Homogene Koordinaten im Raum	246
4.6	Vektorgeometrie aktivieren	250
5	Funktionensysteme, Fourier-Transformation und Faltung	259
5.1	Unendliche Reihen von Funktionen	259
5.1.1	Potenzreihen	259
5.1.2	MacLaurin- und Taylor-Entwicklungen	261
5.1.3	Integration mit Potenzreihen	263
5.2	Orthogonalpolynome	264
5.2.1	Orthogonalität von Funktionen	264
5.2.2	Die Wirkung der Orthogonalität	265
5.2.3	Tschebyscheff-Polynome	267
5.3	Fourier-Reihen, Fourier-Transformation	269
5.3.1	Definition der Fourier-Reihen	269
5.3.2	Die Berechnung der Fourier-Koeffizienten	271
5.3.3	Das Fourier-Spektrum	272
5.4	Diskrete Fourier-Transformation und FFT	276
5.4.1	Definition der diskreten Fourier-Transformation	277
5.4.2	Aliasing, Nyquist-Frequenz, „sampling“	278
5.4.3	Das Prinzip der schnellen Fourier-Transformation	280
5.4.4	M-Files zur Demonstration des FFT-Prinzips	283
5.5	Die Fourier-Transformation näher kennenlernen	286
5.6	Die einfache Faltung	289
5.6.1	Das Prinzip der einfachen Faltung	289
5.6.2	Die Faltung als Multiplikation von Polynomen	291
5.6.3	Die Formel zur Faltung von Zahlenfolgen	292
5.6.4	Beispiele von einfachen Faltungen	294
5.6.5	Die Faltung von kontinuierlichen Funktionen	295
5.7	Zirkuläre Faltung – Faltungssatz	295
5.7.1	Die Definition der zirkulären Faltung	295
5.7.2	Der Faltungssatz	296
5.7.3	Zwei- und mehrdimensionale Faltungen	299
5.8	Funktionensystem- Faltungs- und Fourier-Theorie aktivieren	300
6	Funktionen von mehreren Variablen	311
6.1	Grundbegriffe der Funktionen von mehreren Variablen	311
6.1.1	Die Funktionsdefinition	311
6.1.2	Grafische Darstellung	312
6.1.3	Differenzieren von Funktionen mit mehreren Variablen	313
6.1.4	Illustration der partiellen Ableitung	314
6.2	Das Bilden von partiellen Ableitungen	318
6.2.1	Grundprinzip des partiellen Ableitens	318
6.2.2	Ableitungstabelle für Grundfunktionen	318
6.2.3	Ableitungsregeln für zusammengesetzte Funktionen	319

6.2.4	Beispiele von partiellen Ableitungen	319
6.2.5	Partielle Ableitungen im symbolischen Modus	320
6.3	Partielle Ableitungen und das totale Differential	321
6.3.1	Die Formel für das totale Differential	321
6.3.2	Anwendung zur Berechnung der Volumenausdehnung	322
6.3.3	Empfindlichkeit der Eigenfrequenz	323
6.3.4	Kommerzielle Einflussanalyse	323
6.3.5	Das Optimierungsprinzip in mehreren Variablen	324
6.4	Höhenlinien- und Flächenplots	325
6.4.1	Höhenlinien	326
6.4.2	Dreidimensionale Flächendarstellungen	328
6.4.3	Die Funktion Meshgrid	330
6.4.4	Darstellung der Gradientenvektoren	330
6.4.5	Kombinierte Flächen- und Konturdarstellungen	331
6.5	Ausgleichsrechnung	333
6.5.1	Geradenfit als Beispiel	333
6.5.2	Allgemeine lineare Ausgleichsprobleme	335
6.6	Algorithmen zur Ausgleichsrechnung	337
6.6.1	Normalengleichungen und Fehlergleichungen	338
6.6.2	Singular Value Decomposition	342
6.7	Die Methode der Lagrange-Multiplikatoren	344
6.7.1	Optimierungsprobleme mit Nebenbedingungen	344
6.7.2	Beispiele für Lagrange-Multiplikatoren	346
6.8	Nichtlineare Gleichungssysteme	347
6.9	Kenntnisse von Funktionen mehrerer Variablen aktivieren	350
7	Differentialgleichungen	359
7.1	Die Bedeutung von Differentialgleichungen in Physik und Technik	359
7.1.1	Was ist eine Differentialgleichung?	360
7.1.2	Grundtypen von Differentialgleichungen	361
7.2	Beispiele zu den Differentialgleichungs-Typen	363
7.2.1	Gewöhnliche Differentialgleichungen	363
7.2.2	Partielle Differentialgleichungen	365
7.3	Analytische Lösungen von Differentialgleichungen	367
7.3.1	Lösungs-Prinzipien	367
7.3.2	Beispiele analytischer Lösungen	369
7.3.3	Oszillatorgleichungen	377
7.4	Lösungen mit Laplace-Transformationen	381
7.4.1	Das Lösungsprinzip	381
7.5	Numerische Lösungsverfahren für Anfangswertprobleme	386
7.5.1	Das Grundprinzip der Lösung von Anfangswertproblemen	386
7.5.2	Das Euler-Verfahren	387
7.5.3	Runge-Kutta Verfahren	388
7.5.4	Explizite und implizite Verfahren	393
7.6	Anfangswertprobleme mit MATLAB lösen	395

- 7.6.1 Radioaktive Zerfälle 395
- 7.6.2 Der schiefe Wurf, ein Körper im Gravitationsfeld 397
- 7.6.3 Der gedämpfte harmonische Oszillator 400
- 7.6.4 Demonstration des Steifheit-Effektes 402
- 7.6.5 Geladene Teilchen im Magnetfeld 405
- 7.6.6 $E \times B$ -Drift: Elektrische und magnetische Felder 406
- 7.7 Schnuppern am Chaos 407
- 7.7.1 Der Lorenz'sche Strange Attractor 407
- 7.8 Kenntnisse über Differentialgleichungen aktivieren 410

8 Grundlagen der Statistik 421

- 8.1 Motivation: Überblick über große Datenmengen 421
 - 8.1.1 Die Schuhgrößen als Beispiel 421
 - 8.1.2 Schlüsselzahlen zum Charakterisieren von Verteilungen 422
 - 8.1.3 Die Formeln zur Median-Familie 423
 - 8.1.4 Die Formeln zu Mittelwert und Standard-Abweichung 425
 - 8.1.5 Der grafische Test einer Verteilung 428
- 8.2 Regressions-Analyse 429
 - 8.2.1 Korrelations-Untersuchungen für zwei Dimensionen 429
- 8.3 Wahrscheinlichkeitsrechnung 433
 - 8.3.1 Die Grundelemente von Glücksspielen 433
 - 8.3.2 Anordnungs- und Auswahlformeln 438
 - 8.3.3 Wahrscheinlichkeit, mathematisch definiert 443
 - 8.3.4 Beispielprobleme 445
- 8.4 Statistische Verteilungen 449
 - 8.4.1 Dichte und Wahrscheinlichkeitsverteilung 449
 - 8.4.2 Diskrete Verteilungen 450
 - 8.4.3 Stetige Verteilungen 453
- 8.5 Stichproben und Tests 457
 - 8.5.1 Der Ablauf einer Stichprobe 457
 - 8.5.2 Statistische Tests 459
- 8.6 Kenntnisse zu den Grundlagen der Statistik aktivieren 462

Anhang A MATLAB professionell einsetzen 467

- A.1 Erweiterungen in grafischer Richtung 467
 - A.1.1 Audio-Video-Sequenzen und Webinare 467
 - A.1.2 Erstellen von grafischen Benutzeroberflächen mit GUIDE 468
 - A.1.3 Simulink 468
- A.2 Die Ausdehnung der Einsatzmöglichkeiten 469
 - A.2.1 Erweiterungen im Basispaket 469
 - A.2.2 Zusatzpakete 469
 - A.2.3 Die weltweite Benutzergemeinschaft 470
 - A.2.4 Rückmeldungen und weitere Beispiele 470

Literaturhinweise 471

Zum guten Ende 473

Stichwortverzeichnis 475

Vorwort

Einführung – Mathematik im Einsatz?

Verweilen Sie einmal ganz kurz bei der Betrachtung Ihrer mathematischen Kenntnisse: Wieviel Mathematik haben Sie schon „gehabt“? Und wieviel davon können Sie umsetzen in die Erarbeitung eines konkreten Berechnungsergebnisses?

Bei einer großen Zahl von Studierenden der Naturwissenschaft oder der Ingenieurfächer, aber auch bei vielen Praktikern in diesen Berufen ist die Mathematik als abstrakte intellektuelle Beschäftigung wesentlich stärker ausgeprägt als deren tatsächliche Anwendung im Verlauf von Berechnungen. Mit diesem Buch wird ein Weg aufgezeigt, um die zwei Aspekte der Mathematik in ein harmonischeres Gleichgewicht zu bringen.

Dadurch ergibt sich auf natürliche Weise eine spezielle Auswahl bei den zu behandelnden Fragestellungen sowie eine an die Anforderungen der Ingenieure und Naturwissenschaftler angepasste Gewichtung der präsentierten mathematischen Themen. Bei dieser Auswahl der Themen und ihrer Gewichtung steht das Erarbeiten von konkreten, in Zahlen ausdrückbaren Resultaten im Vordergrund. Die hier beim Lösen von konkreten Problemen eingesetzten Teilgebiete der Mathematik gehören daher zum Themenkreis der Angewandten Mathematik sowie der Numerischen Mathematik.

Das Ziel dieses Kurses

Mit dem Kurs „MATLAB und Mathematik kompetent einsetzen“, angesiedelt in ein bis zwei der mittleren Semester des Ingenieur- oder Naturwissenschaftsstudiums, soll eine Brücke geschlagen werden zwischen den in den unteren Semestern erarbeiteten mathematischen Grundlagen und dem Einsatz der Mathematik in den verschiedenen Ingenieur-Disziplinen sowie in der wissenschaftlichen Modellbildung und Analyse. Dank des tief angesetzten Niveaus der erforderlichen Vorkenntnisse und des vielfältigen Übungsmaterials ist dieses Buch auch hervorragend geeignet zum Selbststudium, für Studenten ebenso wie für erfahrene Praktiker.

Mathematik ist ein wichtiges Werkzeug der Ingenieure und Naturwissenschaftler, dessen Beherrschung einen wesentlichen Teil des Berufserfolges ausmachen

kann. Das Ziel dieses Kurses ist deshalb das Herauslösen der Mathematik aus den verstaubten Aktenordnern, das Abbauen von Berührungängsten und das Einpflanzen in den Katalog Ihrer persönlichen Fähigkeiten. Der Weg zu diesem Ziel ist ein unverkrampfter, spielerischer Umgang mit vielfältigen Berechnungsproblemen, wobei uns das Programm MATLAB sehr gute Dienste leisten wird. MATLAB kann uns den Aufwand der eigentlichen Berechnung abnehmen, dadurch können wir uns auf den Lösungsablauf konzentrieren. Die häufige Verwendung grafischer Darstellungen vermittelt ein verstärktes Verständnis für die Zusammenhänge zwischen mathematischen Strukturen. Gleichzeitig erhalten wir vertiefte Einblicke in die Funktionsweise von Berechnungsabläufen durch die vielfältigen Möglichkeiten, eigene kleine Programme zu erstellen.

Die Kombination der zwei Themen „Mathematik“ und „MATLAB“ bewirkt keine Aufspaltung in zwei Ziele verschiedener Richtung, sondern führt zu einer gegenseitigen Verstärkung der beiden Teile. Die enge Verflechtung der beiden Aspekte fördert das übergeordnete Ziel: Ihre persönliche Kompetenz zum Lösen von Berechnungsproblemen aufzubauen.

Ein Wort an die Leserinnen dieses Buches

Bei Bezeichnungen von menschlichen Personen trifft man immer häufiger auf die ausführliche Erwähnung beider Geschlechtsformen, wie in „Leserinnen und Leser“ oder auf Kunstwörter wie „LeserInnen“. Zugunsten der flüssigen Lesbarkeit wurde in diesem Buch auf solche Formen verzichtet und durchgehend die althergebrachte einfache Bezeichnung in der männlichen Form verwendet. Mit dieser Form waren in der Bedeutung der deutschen Sprache schon immer alle Menschen beiderlei Geschlechts gemeint. Es wäre schade, wenn sich Personen weiblichen Geschlechts durch diesen Entscheid ausgeschlossen fühlten, denn das Buch möchte ihnen mit dem Einbringen einer ganzheitlichen, bildhaften Komponente in die Mathematik besonders entgegenkommen.

Die empfohlene Arbeitsweise

Erschrecken Sie nicht! In diesem Vorwort fehlt die Behauptung, dass Sie durch die Benutzung dieses Buches alles wie von selbst, ohne Ihren eigenen Einsatz lernen werden. Das Weglassen eines solchen „verkaufsfördernden“ Versprechens, das dann doch nicht eingelöst werden kann, geschieht mit voller Absicht. Es liegt mir viel daran, meine Leser als vernünftige Partner ernst zu nehmen. Ernsthaftigkeit beim Verfolgen des Ziels, Mathematik zum persönlichen, vielfältig einsetzbaren Werkzeug zu machen, bedeutet den Willen zum Engagement, zur eigenen Mitarbeit. Ernsthaftigkeit bedeutet aber in keiner Weise einen Zwang zur Verkrampfung – im Gegenteil, mit Lockerheit, Kreativität, Phantasie und Experimentierfreude kann man eine größere Arbeitsleistung erbringen ohne dies als Belastung zu empfinden. Dies ist ein Lehr-, Übungs- und Animationsbuch mit Schwergewicht auf der Animation.

Die Vielzahl der Übungen und der breit gestreute Bezug auf verschiedenartige Aspekte der realen Welt möchten zum aktiven Mitmachen animieren. Die häufige Verwendung von grafischen Darstellungen in den Übungsbeispielen und die Hinweise auf bildhafte Modelle zur Veranschaulichung der abstrakten mathematischen Strukturen dient der Einbindung der synthetischen Hirnfunktion in den Lernprozess. Durch dieses Vermeiden der einseitigen Belastung des Gehirns auf der analytischen Seite wird die Erinnerungsfähigkeit wesentlich gesteigert.

Lassen Sie sich mitreißen, Mathematik zu erleben und erfahren Sie durch den praktischen Einsatz von MATLAB die schrittweise Steigerung Ihrer eigenen Fähigkeiten!

Hinweise zur zweiten Auflage des Buches

Drei Aspekte fanden bei der intensiven Überarbeitung des Buches für die zweite Auflage besondere Beachtung.

Der wichtigste davon war der Wunsch, die starke Weiterentwicklung von MATLAB einzubinden. Verschiedene neue Datentypen, Einstieg in die Objekt-orientierte Programmierung, der verstärkte Einbezug von symbolischer Algebra und eine Vielzahl von Hilfsprozeduren für den Umgang mit externen Daten beschreiben einen Teil der weit gehenden Neuerungen.

Zum Zweiten war mir durch die Lektüre der Bücher von Verena Steiner bewusst geworden, dass ein Text einen gewissen Rhythmus braucht. Die Anforderungen an die Konzentration des Lesers dürfen nicht konstant auf dem höchsten Niveau verbleiben, es braucht immer wieder Erholungsphasen. Abwechslung zwischen dem Eintauchen in Details und dem Überblicken des Erlebten erleichtert das Lernen. Deshalb wurden überall Merkpunkte eingefügt und das Stichwortverzeichnis wurde intensiv überarbeitet.

Als Drittes möchte ich das schon in der ersten Auflage wichtige Ziel nochmals unterstreichen, den Leser zu aktivieren. Es besteht ein entscheidender Unterschied zwischen dem passiven Durchlesen eines mathematischen Themas und dem kreativen Akt, selbst eine Lösung zu einem Problem zu finden.

Einige technische Änderungen müssen noch kurz erwähnt werden: Die Daten-CD der ersten Auflage wurde durch einen Zugriff auf das Internet ersetzt. Beispiele von M-Files, sowie die Lösungen zu den Übungen wurden ebenfalls ins Internet verschoben. Dadurch konnten die Beispiel-Lösungen mit ausführlicheren Erklärungen versehen werden.

Der Leser findet die Daten bei www.wiley-vch.de/textbooks/ (suchen Sie bitte nach „A“ wie Adam) unter dem Punkt „Dozentenmaterial“. Zugriff darauf erhalten Sie mit der E-Mail-Adresse adam@wiley-vch.de und dem Passwort:

hGcR3.

Ein neues Kapitel mit einer Einführung in die Wahrscheinlichkeitsrechnung und Statistik fand seinen Platz anstelle der Musterlösungen. Das ehemals separate Kapitel zur symbolischen Algebra wurde auf die einzelnen Themen verteilt.

Stefan Adam

1

Grundkenntnisse von MATLAB®

1.1

Bekanntheit schließen mit MATLAB

1.1.1

Die Arbeitsoberfläche von MATLAB

Der Start der Applikation MATLAB¹⁾ erfolgt abhängig vom Betriebssystem auf unterschiedliche Weise:

- in den Microsoft Windows Systemen und Mac's durch einen Doppelklick auf das Icon von MATLAB oder durch Anwählen des Untermenüs in der Abfolge Programme -> ... -> MATLAB
- auf Unix/Linux-Systemen durch Eintippen des Kommandos `matlab &` (Starten von MATLAB mit dem Programmnamen und installieren als unabhängigen Prozess durch das Zeichen „&.“) Bei neueren Desktop Manager Programmen kann MATLAB ebenfalls durch Anwählen eines Icons gestartet werden.

Achten Sie darauf, dass im Kommandomodus das Startkommando `matlab &` in Kleinbuchstaben eingegeben wird, obwohl Sie den Markennamen MATLAB immer in Großbuchstaben gesetzt sehen.

Darauf erscheint kurz das quadratische MATLAB-Markenzeichen, Banner genannt. Dieses Bild ist die Visualisierung der Schwingung einer L-förmigen Membran. Kurz danach zeigt sich das in drei vertikale Streifen unterteilte Hauptfenster von MATLAB.

Die meisten Aktionen spielen sich im mittleren Streifen im **Kommandofenster** ab (Fenstertitel: Command Window): Eingabe von Berechnungs-Befehlen, Abfrage von Daten und Hilfstexten, sowie Aufruf von Funktionen und Programmen. Ebenfalls im Kommandofenster erfolgt die Ausgabe von Resultaten, kurzen Texten und Fehlermeldungen.

Die Hilfsfenster in den beiden Seitenstreifen vermitteln nützliche Information zum aktuellen Dateipfad und zu bisher ausgeführten Befehlen, sowie zu den dabei erzeugten Daten.

1) MATLAB® ist ein eingetragenes Warenzeichen von The MathWorks Inc., Natick, MA, USA.

MATLAB und Mathematik kompetent einsetzen, 2. Auflage. Stefan Adam.

©2017 WILEY-VCH Verlag GmbH & Co. KGaA. Published 2017 by WILEY-VCH Verlag GmbH & Co. KGaA.

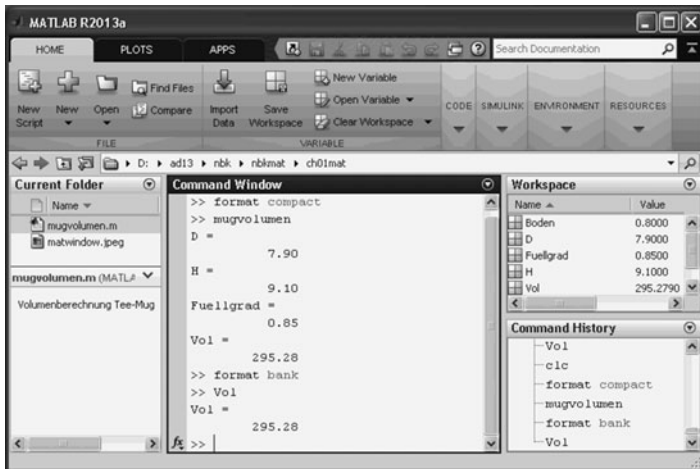


Abb. 1.1 Das Hauptfenster von MATLAB.

Im Kommandofenster, anschließend an die zwei nach rechts zeigenden Winkel `>>` wartet MATLAB auf die Eingabe des nächsten Befehls. Eventuell muss der Text-Cursor noch mit der Maus dorthin gesetzt werden. Bei einer längeren Berechnung kann sich die Eingabe-Bereitschaft verzögern.

Grafische Darstellungen werden in unabhängigen Fenstern mit der Bezeichnung `figure` (und einer nachfolgenden Zahl) gezeigt.

MATLAB-Befehle bestehen aus direkten Anweisungen zum Ausführen einer Berechnung, Zuweisungen von Daten, Aufrufen von Funktionen oder vielseitigen Kombinationen davon.

1.1.2

Zum Einstieg: Berechnungen mit einfachen Zahlen

Das Programm MATLAB ist ein äußerst vielseitiges Werkzeug für alle Arten von Berechnungen und Simulationen. Somit muss logischerweise das Rechnen mit einfachen Zahlen darin enthalten sein: Nachdem Sie MATLAB gestartet haben, können Sie also als Einstieg ein paar einfache Berechnungen eintippen wie z. B.:

```
>> 2 + 5 <ret>           (Das Drücken der <return>-Taste am Zeilenende wird im
                          Weiteren stillschweigend vorausgesetzt)
```

Die Resultatausgabe versieht den Resultatwert mit dem Namen `ans`:

```
ans =
     7
>> 2 * 5 + 7
```

Jede neue Berechnung überschreibt `ans` mit dem neuen Wert.

```
ans =
    17
```

```

>> 7^2                                (die MATLAB-Schreibweise für 72, also 7 hoch 2 = 7 · 7)
    ans =
        49                                und als Gegensatz
>> 2^7
    ans =
        128                                und auch noch
>> 2^-7
    ans =
        0.0078                            (hier genügen 4 Dezimalstellen nicht)
>> format long                        (verlangt Ausgabe mit 15 Dezimalen)
    ans =
        0.007812500000000
>> format short                        (Format zurücksetzen)

```

Wie bei einem Taschenrechner können große oder sehr kleine Zahlen mit einem Exponenten-Zusatz („E“ oder „e“, gefolgt von positiver oder negativer ganzer Zahl) eingegeben werden: (Beispiel Flugzeit des Lichtes von der Sonne zur Erde: 150 Millionen km dividiert durch 300 000 km/s)

```

>> 150E6 / 300e3
    ans =
        500.0000                            Resultat in Sekunden
>> tlichtmin = ans/60                    geläufiger in Minuten
    tlichtmin =
        8.3333

```

Für die Berechnung der mittleren Dichte der Erde werden mit Vorteil zuerst die Variablen Radius und Masse definiert

```

>> Erad = 6371*1000 ;                    inkl. Umwandlung km in m
>> Emass = 5.9736e24;                    in kg
>> Edicht = Emass/(4*pi/3*Erad^3)
    Edicht =
        5.5147e+03                        in kg/m3, in bekannten Einheiten 5,5 kg/L

```

Nun noch eine näher liegende Berechnung:

```

>> Vol = (7.9-2*0.3) ^ 2 * pi /4 * (9.1-0.8) ...
    * 0.85
    Vol =
        295.2790

```

In der obigen Volumenberechnung ist es fast unumgänglich, dass man auch den darin vorkommenden Zahlen, und nicht nur dem Resultat, einen Namen zuweist. Das hilft, die durchgeführte Berechnung zu verstehen und zu dokumentieren. Dadurch wird aber die Eingabe mehrzeilig. Am besten schreibt man die Befehle in ein **Skript**.

Skripts sind Dateien mit dem Namen-Zusatz „.m“ (sogenannte M-Files), in denen eine Serie von MATLAB-Befehlen als reiner Text aufgelistet ist. Am besten erstellt man M-Files mit dem MATLAB-eigenen Editor. Dessen Fenster erscheint

im Mittelstreifen oberhalb des Command Window, wird aber mit Vorteil abgekoppelt in ein separates Fenster.

Das M-File `mugvolumen.m` hat in unserem Beispiel die folgende Form:

```
% Volumenberechnung Tee-Mug
D = 7.9 ;
H = 9.1 ;
Wand = 0.3; Boden = 0.8;
Fuellgrad = 0.85;
Vol = (D-2*Wand)^2 *pi/4 * (H-Boden) * Fuellgrad
```

Sobald dieses File unter dem Namen `mugvolumen.m` abgespeichert ist, bewirkt der Befehl `>> mugvolumen` (Filename, aber ohne Zusatz!) die Ausführung der darin aufgezeichneten Befehle. Das Resultat ist dasselbe wie beim einzelnen Befehl

```
Vol =
    295.2790
```

Der Mug fasst 295 cm^3 , also knapp drei Deziliter.

1.1.3

Befehlsstruktur: ein erster Überblick

Aus diesen einfachen Beispielen ersieht man bereits einige Grundprinzipien von MATLAB, welche in der Anwendung immer wieder auftreten:

Die Zahl $\pi = 3,141\,592\dots$ für die Kreisberechnung ist vordefiniert und unter dem Namen `pi` abrufbar.

Bei **Skript-M-Files** wird eine Reihe von Befehlen in ein Textfile mit dem Filenamen-Zusatz `.m` geschrieben. Der ganze Block gelangt dann durch Eintippen des Filenamens (ohne den Zusatz `.m`) zur Ausführung.

Das Arbeiten mit ganzen Blöcken (Skripts) von Befehlen heißt in der Informatik Makro-Programmierung. (Ausblick: Im Innern eines Blocks können in beliebiger Tiefe auch andere Blöcke aufgerufen werden; Rekursion ist aber nur für Funktionen erlaubt.)

Die Steuerung der **Bildschirm-Ausgabe** erfolgt mit `format ..` (Beispiel: Feinstruktur-Konstante und Inverse)

Befehl	Stellen	a	$1/a$
<code>format short</code>	4	0.0073	137.0360
<code>format long</code>	15	0.007297352569800	1.370359990743064e+02
<code>format long e</code>	15	7.297352569800e-03	1.370359990743064e+02
<code>format bank</code>	2	0.01	137.04

Für Zeilenabstände `eng/locker` mit: `format compact/format loose`.

Merkpunkte: Konventionen der Kommandosprache

- **Zeilen:** MATLAB-Befehle stehen meist einzeln, jeder in einer Zeile. Lange Befehle können durch drei Punkte . . . am Ende der Zeile eine Fortsetzung verlangen (auch mehrfach).
- **Ausgabesteuerung durch Befehls-Abschluss:** Nach der Ausführung eines Befehls wird das dazugehörige Zwischenresultat im Kommandofenster angezeigt. Durch einen Strichpunkt am Zeilenende wird diese Ausgabe unterdrückt.
- **Mehrere Befehle in einer Zeile:** Mehrere Befehle können durch Komma oder Strichpunkt getrennt in dieselbe Zeile geschrieben werden. Komma als Trennung bewirkt Zwischenresultat-Ausgabe, Strichpunkt wie bei einzelnen Befehlen dessen Unterdrückung.

Merkpunkte: Verwendung von Namen/Variablennamen

- Die zu verarbeitenden Zahlen, sowie Zwischen- und Schlussresultate in MATLAB sollten **Namen** (= Variablennamen, Bezeichner) erhalten. Diese können später dazu dienen die mathematischen Größen wieder zu verwenden und helfen erst noch die Übersicht zu bewahren.
- Für jede Berechnung kann der Programmbenutzer den Resultatnamen durch die Formulierung 'name' = vorgeben, wie z. B. in $c2 = a^2 + b^2$ (gefolgt von $c = \text{sqrt}(c2)$) für die Berechnung der Hypotenuse nach Pythagoras).
- **Namen** sind in MATLAB empfindlich auf **Groß- und Kleinschreibung** (englisch: „case sensitive“). Somit ist also A etwas anderes als a. Namen müssen mit einem Buchstaben beginnen, dahinter dürfen auch Zahlen enthalten sein. Ungewohnt ist, dass keine Bindestriche in den Namen erlaubt sind (auch nicht bei Filenamen), weil MATLAB diese als Minuszeichen interpretiert.
- Als Reaktion auf das Eintippen eines bereits definierten Variablennamens gibt MATLAB den unter diesem Namen gespeicherten Wert (bzw. die Werte bei Vektoren und Matrizen) auf dem Bildschirm (im Kommandofenster) aus.
- Falls der Benutzer bei einer Berechnung keinen Resultatnamen angibt, setzt MATLAB immer automatisch denselben Allerweltsnamen `ans` ein (Kurzform von „answer“). Dies erlaubt unmittelbar nach der Berechnung noch die Zuweisung an einen richtigen Namen z. B. durch $c2 = \text{ans}$.
- **Vorsicht!** Bei jeder Berechnung (Wertzuweisung) wird der Resultatname ohne Warnung mit dem neuen Wert überschrieben! Sie haben also keine

andere Chance, das vorhergehende ans zurückzugewinnen, außer indem Sie die dazugehörige Berechnung von Grund auf neu ausführen!
 Ähnlich ergeht es Ihnen, wenn Sie zweimal hintereinander
`a = 'Berechnung'` schreiben, dann ist das vorherige a überschrieben.
Fazit: Wählen Sie nicht zu kurze und vielfältige Variablennamen!

1.1.4

Berechnung oder Formel-Manipulation?

MATLAB kann beides! Für Formel-Manipulation braucht man allerdings den Zusatz „symbolic toolbox“ (in Studentenlizenz enthalten).

In der **Berechnungs-Umgebung**, im normalen Arbeits-Modus von MATLAB werden Zahlen verarbeitet, um damit ein **in Zahlen ausdrückbares Resultat** zu erhalten. Den verwendeten Variablen müssen also vor ihrer Verwendung Zahlenwerte zugeordnet werden. Sonst kommt die Fehlermeldung `Undefined function or variable`.

Die verwendeten Namen werden bei ihrer ersten Zuweisung automatisch deklariert. (Im Gegensatz zu vielen anderen Programmiersprachen.)

Im **Symbolic-Modus** (Formel-Modus) müssen alle in einer Formel verwendeten symbolischen Variablen als solche deklariert werden. Das erfolgt z. B. mit dem Befehl `syms s t u` (Variablen durch Leerschläge getrennt).

Das **Ziel des symbolischen Rechnens ist eine Formel**. Darum wird diese Art der Verarbeitung auch Formel-Modus genannt. Eingegebene Formeln werden umgestellt oder vereinfacht, zu gegebenen Ausdrücken werden Ableitungen oder unbestimmte Integrale ermittelt; von Gleichungssystemen oder Differentialgleichungen werden analytische Lösungen bestimmt.

Übungen zur expliziten Schreibweise von Produkten

Die Tatsache, dass in MATLAB das Multiplikationszeichen immer explizit geschrieben werden muss, braucht für den Anfang ein wenig Übung. Daraus ergibt sich eine glänzende Gelegenheit, die binomischen Formeln aufzufrischen.

Weil dies für beide Arbeits-Modi gilt, ergibt sich ein direkter Quervergleich zwischen Berechnung und Formel-Manipulation.

Binomische Formeln im Symbolic-Modus

Der Symbolic-Modus zeigt diese Formeln mit Hilfe der Funktion `expand()`:

```
% Deklaration von s und t als symbolische Variablen
syms s t
expand( (s-t)^2)
ans = s^2 - 2*s*t + t^2
expand( (s-t)^3)
ans = s^3 - 3*s^2*t + 3*s*t^2 - t^3
expand( (s-t)^4)
ans = s^4 - 4*s^3*t + 6*s^2*t^2 - 4*s*t^3 + t^4
```

und wieder zurück

```
factor(ans)
ans = (s-t)^4
```

und nur so zum Spaß:

```
expand((s-t)^50)
```

Versuchen Sie nun selbst, die ausmultiplizierte Form von $(s-t)^2$ und $(s-t)^3$ einzugeben. Achten Sie auf die Eingabe aller Multiplikationszeichen.

Testen Sie Ihre Eingabe mit `factor(ans)`.

Binomische Formeln im Berechnungs-Modus

Im Berechnungs-Modus müssen zuerst den Variablen a und b Zahlenwerte zugewiesen werden.

Der Test, ob Sie die Formel richtig eingegeben haben ergibt sich aus dem Zahlenwert (es muss gelten $p3 = q3$, $p4 = q4$ etc.)

Fehlende Multiplikations-Operatoren ergeben die Fehlermeldung `Unexpected MATLAB expression` oder `Undefined function or variable 'ab'`. Ergänzen Sie die fehlenden Terme bei den Fragezeichen!

```
a = 10 , b = 2
p3 = (a-b)^3
q3 = a^3 - 3*a^2*b + 3*a*b^2 - b^3 }
p4 = (a-b)^4
q4 = a^4 - 4*a^3*b + 6*a^2*b^2 ..?
p5 = (a-b)^5
q5 = a^5 - 5*a^4*b + 10*a^3*b^2 ..?
```

Merkpunkt: Der Multiplikations-Operator

- Die **Multiplikation** von zwei nebeneinander stehenden Größen muss durch das „*“-**Zeichen** explizit verlangt werden. Bei Zahlen ist dies auch in der Mathematik erforderlich: 22 bedeutet etwas anderes als $2 * 2$ (bzw. $2 \cdot 2$). Bei der Verwendung von Buchstaben als symbolische Zahlen darf man jedoch in der Mathematik die Kurzform $2ab$ schreiben statt $2 * a * b$.
In MATLAB wird hingegen immer die ausführliche Schreibweise verlangt, also ist nur $2*a*b$ eine richtige Eingabe. Diese Konvention gilt übrigens bei fast allen Programmiersprachen.

Klammern in algebraischen Ausdrücken

Die Prioritätsregeln für arithmetische Operatoren werden populär mit dem Satz „Punkt-Rechnung kommt vor Strich-Rechnung“ memorisiert (mit Multiplikations-Punkt und Divisions-Doppelpunkt).

Nochmals genau definiert gilt:

- Addition „+“ und Subtraktion „-“ haben tiefste Priorität;
- Multiplikation „*“ und Division „/“ die nächst höhere und
- Potenzieren „^“ die höchste.

konkret: bei $4 + 5 * 3^2$ muss zuerst $3^2 = 9$ berechnet werden, dann $5 * 9 = 45$ und am Schluss $4 + 45 = 49$.

Um eine Abweichung davon zu erzwingen braucht man Klammern.

In MATLAB sind für die Strukturierung von arithmetischen Ausdrücken **ausschließlich runde Klammern** erlaubt.

Das Horner-Schema im Formel-Modus Das Horner-Schema zur Auswertung von Polynomen liefert einen weiteren, eindrücklichen Quervergleich zwischen Formel-Manipulation und Berechnung. Zuerst die Darstellung im Formel-Modus:

Das Polynom 4. Grades $P_4 = q_4 * x^4 + q_3 * x^3 + q_2 * x^2 + q_1 * x + q_0$ mit den Koeffizienten q_4, q_3, q_2, q_1, q_0 kann im Formel-Modus als normales Polynom definiert werden, wie oben dargestellt. In der Schreibweise des Horner-Schemas werden alle Faktoren „ x “ einzeln ausgeklammert. Der Formel-Modus enthält eine Funktion zur Erzeugung des Horner-Schemas aus der normalen Polynomdarstellung. Diese wird anschließend aufgerufen.

```
>>syms q4 q3 q2 q1 q0 x
>>P4 = q4*x^4 + q3*x^3 + q2*x^2 + q1*x + q0
>>P4H = (((q4*x + q3)*x + q2)*x + q1)*x + q0
>>P4Hf = horner(P4)
P4Hf =
q0 + x*(q1 + x*(q2 + x*(q3 + q4*x)))
```

Das Horner-Schema im Berechnungs-Modus Um ein Polynom numerisch, im Berechnungs-Modus zu bearbeiten, müssen wir den Koeffizienten konkrete Zahlen zuweisen. Wir nehmen das Beispiel: $P_4N = x^4 + x^3 - 6 * x^2 - 4 * x + 8$ und dessen Auswertung am Wert $x = 1,5$. (Nicht vergessen, den symbolic-Status von x zu beenden!)

Die Koeffizienten fassen wir in einer Zahlenfolge mit dem Namen `cf` zusammen. Dies geschieht durch Einschließen in eckige Klammern.

```
>>clear x
>>cf = [1 1 -6 -4 8];
>>x = 1.5;
>>pval=(((cf(1)*x +cf(2))*x +cf(3))*x +cf(4))*x +cf(5)
pval =
-3.0625
```

Berechnung mit einem Funktions-M-File Von da ist es nun ein kleiner Schritt zum Erstellen einer ersten selbst programmierten Funktion. Dazu schreibt man die folgenden Zeilen in ein Textfile mit dem Namen `hornereval.m` (schon die zweite Art von M-File).

```

function pv = hornereval(cv, xv)
% function pv = hornereval(cv, xv) Polynom-Evaluation
%   cv = Koeffizienten-Folge, absteigend,
%   xv = auszuwertende x-Positionen (Vektoren erlaubt)
nkoef = length(cv);
pv = cv(1); % Start der Polynomwert(e)
for cnum = 2:nkoef % in jedem Schritt: (..)*xv + cv(cnum)
    pv = pv.*xv + cv(cnum); % ',.*' erlaubt xv-Vektoren
end

```

Sobald dieses File abgespeichert ist, funktioniert der Befehl

```

>>pval = hornereval(cf,1.5)
pval =
    -3.0625

```

Aber beliebige andere Koeffizienten-Folgen und ganze Zahlenfolgen (Vektoren) von x -Werten sind ebenso möglich. Das Prinzip einer Funktion mit Parameterübergabe und Resultatrückgabe erlaubt die vielfache Anwendung eines einmal programmierten Berechnungsablaufes mit immer neuen Datensätzen.

Merkpunkte zum Funktions-Beispiel

- Die Funktionsdeklaration in der ersten Zeile enthält das Codewort `function` gefolgt von den Rückgabeparametern, dem Gleichheitszeichen, dem Funktionsnamen und den Eingabeparametern in runden Klammern.
Der Filename (ohne `.m`) muss mit dem Funktionsnamen übereinstimmen.
Mehrere Rückgabeparameter werden in eckige Klammern eingeschlossen. Den Rückgabeparametern müssen im Innern der Funktion Werte zugewiesen werden.
- Der Punkt vor dem Multiplikations-Stern verhindert, dass MATLAB die Multiplikation als Matrizenprodukt einstuft. Punkt-Stern verlangt dagegen elementweise Multiplikation.
- Die Kommentare (nach dem `%`-Zeichen) bei der Funktionsdeklaration werden beim Befehl `help hornereval` angezeigt, sind also eine Mini-Dokumentation.
- Der Befehl im Innern der `for`-Schleife wird für die Werte der Schleifenvariablen `cnum = 2,3,4, etc.` bis `nkoef` je einmal ausgeführt.

Die neue Funktion ist bereit zur Anwendung, verbunden mit einer Grafik:

```
xpt = -3:3; xfin = -3.5:0.1:3.5; % x grob / fein
ppt = hornereval(cf,xpt) % Funktionswerte zu xpt
pfin = hornereval(cf,xfin); % Funktionswerte zu xfin
% grafische Darstellung
plot(xpt,ppt,'+') ; hold on ;
plot(xfin,pfin) ; grid on; hold off;
```

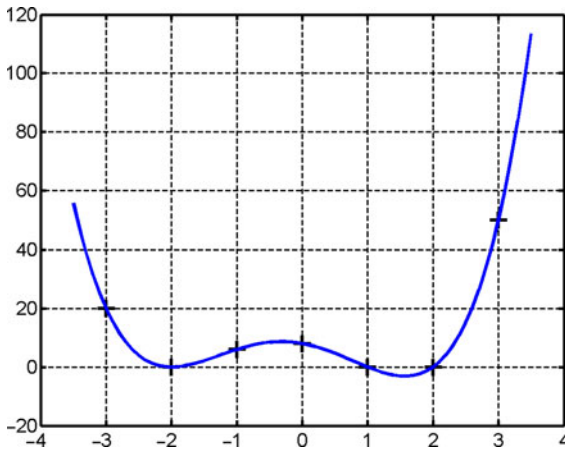


Abb. 1.2 Grafik des Beispiel-Polynoms.

M-Files

Die reiche Vielfalt der Anwendungsmöglichkeiten von MATLAB beruht darauf, dass komplexe Berechnungsaufgaben auf einfachste Weise in Teilaufgaben zerlegt werden können. Die in MATLAB vorhandenen Zerlegungsmechanismen reichen von einfachen Skript-M-Files über Funktions-M-Files bis zu Klassendefinitionen der Objektorientierten Programmierung.

Achtung! Die Bezeichnung M-File verwendet traditionell ein großes „M“, obwohl der Namen-Zusatz „.m“ klein geschrieben wird.

Bei einem **Skript-M-File** werden einfach alle MATLAB-Befehle zum Erledigen einer Teilaufgabe in ein File mit dem Namenszusatz „.m“ geschrieben. Die Eingabe des entsprechenden Namens in MATLAB lässt dann diese Befehlsfolge ablaufen. Man nennt die Funktionalität in der Informatik auch Makroaufruf.

Die Verwendung von **Funktions-M-Files** ist eng damit verwandt. Beim Aufruf werden der Funktion allerdings Parameter übergeben, die in runden Klammern an den Funktionsnamen angehängt werden. In den meisten Fällen haben Funktionen auch einen oder sogar mehrere Rückgabeparameter; dies ist aber nicht unbedingt erforderlich.

Ein großer Teil der vielfältigen, in MATLAB zur Verfügung stehenden Befehle beruht auf solchen Funktions-M-Files. Für komplexere Anwendungen werden