HANSER

Leseprobe

Rainer Hagl

Informatik für Ingenieure

Eine Einführung mit MATLAB, Simulink und Stateflow

ISBN (Buch): 978-3-446-44363-1

ISBN (E-Book): 978-3-446-45116-2

Weitere Informationen oder Bestellungen unter

http://www.hanser-fachbuch.de/978-3-446-44363-1

sowie im Buchhandel.

Vorwort

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

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

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

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

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

Eine wichtige praktische Anwendung ist die Simulation dynamischer Systeme, welche beispielhaft anhand des Softwarepaketes Simulink gezeigt wird. In vielen Steuerungsaufgaben muss auf Ereignisse mit vorbestimmten Aktionen reagiert werden. Ereignisdiskrete Systeme werden daher in einem separaten Kapitel behandelt. Um die Arbeitsweise praxisnah dazustellen, wird in diesem Kapitel das Softwarepaket Stateflow verwendet. Manche Berechnungsaufgaben müssen aufgrund der vielen Einzelschritte auf mehrere Recheneinheiten verteilt werden, um in akzeptablen Wartezeiten Ergebnisse zu erhalten. Daher wird in einem eigenen Kapitel auf die Thematik „Paralleles Rechnen“ in Grundzügen eingegangen. Mathematische Umformungen und Vereinfachungen können computerunterstützt erfolgen. Im Kapitel „Symbolisches Rechnen“ wird einführend auf diese Möglichkeit eingegangen. Den Kapiteln zugeordnete Übungen erlauben eine Überprüfung des Lernfortschrittes. Weiteres Zusatzmaterial steht den Lesern unter *http://www.hanser-fachbuch.de/buch/Informatik+fuer+Ingenieure/9783446443631* zur Verfügung.

Im Buch haben sich sicherlich Fehler eingeschlichen. Vielleicht ist das eine oder andere auch nicht ganz verständlich. Über Rückmeldungen zu Fehlern oder Verbesserungsvorschläge würde ich mich sehr freuen, da diese zu einer kontinuierlichen Verbesserung der schriftlichen Unterlagen führen. Sie können mir diesbezüglich gerne eine E-Mail an *rainer.hagl@fh-rosenheim.de* senden. Für Ihre Unterstützung möchte ich mich bereits im Voraus bei Ihnen bedanken.

Danksagung

In den vergangenen Jahren wurde aufgrund von Erfahrungen in Vorlesungen und Übungen sowie technischen Weiterentwicklungen das Manuskript, das Basis für dieses Buch war, kontinuierlich angepasst. Diskussionen mit meinem Kollegen Prof. Zentgraf, mit dem ich die „Ingenieurinformatik“ an der Hochschule Rosenheim gemeinsam aufgebaut habe, waren sehr offen und hilfreich. Zu den Kapiteln 2 und 9 hat mein Kollege maßgebliche Anteile konzipiert und mir dankenswerterweise erlaubt, diese zu benutzen.

Für die kritische Durchsicht des Manuskriptes möchte ich mich zudem bei meinen Kollegen Prof. Franz Perschl, Prof. Martin Versen und Frau Julia Höllthaler sowie bei den Studierenden Herrn Johannes Hilverkus, Herrn Maximilian Lindinger, Herrn Florian Planthaler und Herrn Maximilian Stadler sehr herzlich bedanken.

Inhalt

2 Grundlagen der Programmierung

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

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

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

- Interpreter

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

- Compiler

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

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

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

---

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

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

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

2.1 Erste Schritte in MATLAB und Grundregeln

MATLAB ist primär eine

- interaktive Programmierumgebung und gleichzeitig
- eine Programmiersprache

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

Im Internet gibt es viele sehr hilfreiche Schulungsunterlagen, die zum Eigenstudium sehr geeignet sind.

Einführungsvideo Englisch

http://www.mathworks.de/videos/getting-started-with-MATLAB-68985.html

Tutorials Englisch

http://www.mathworks.de/academia/student_center/tutorials

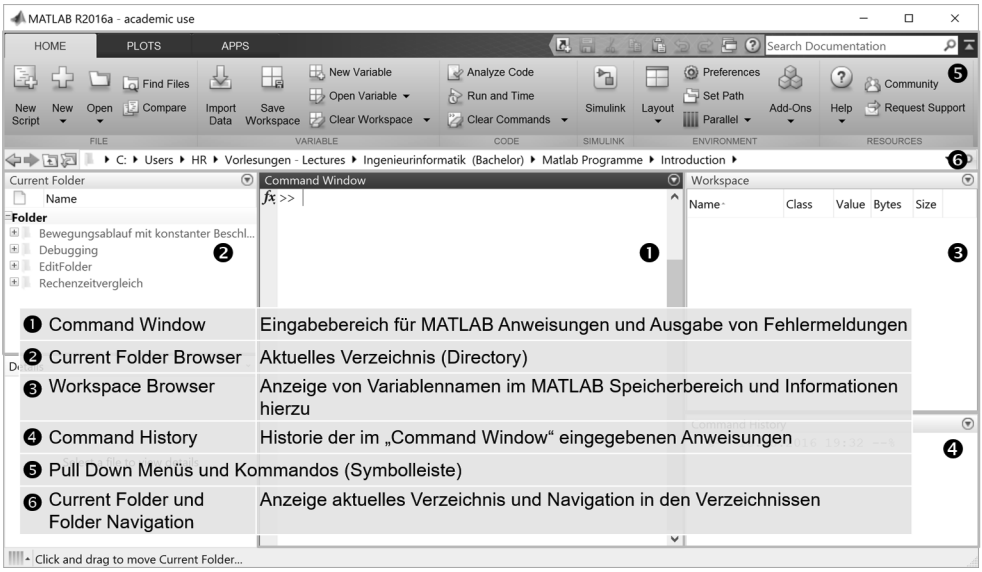
Beispiel- und Produktvideos in Englisch

http://www.mathworks.de/products/MATLAB/demos.html

2.1.1 Bedienoberfläche

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

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



❶ Command Window	Eingabebereich für MATLAB Anweisungen und Ausgabe von Fehlermeldungen
❷ Current Folder Browser	Aktuelles Verzeichnis (Directory)
❸ Workspace Browser	Anzeige von Variablennamen im MATLAB Speicherbereich und Informationen hierzu
❹ Command History	Historie der im „Command Window“ eingegebenen Anweisungen
❺ Pull Down Menüs und Kommandos (Symbolleiste)	
❻ Current Folder und Folder Navigation	Anzeige aktuelles Verzeichnis und Navigation in den Verzeichnissen

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

Bild 2.2 Layout

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

Tabelle 2.1 MATLAB Operatoren für die Grundrechenarten

MATLAB	Operator
Addition	+
Subtraktion	-
Multiplikation	*
Division	/

So können z. B. zwei Zahlen multipliziert werden:

```
2*3
```

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

```
ans =
   6
```

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

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

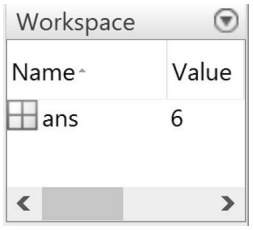


Bild 2.3 Workspace

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

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

```
pi
ans =
   3.1416
```

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

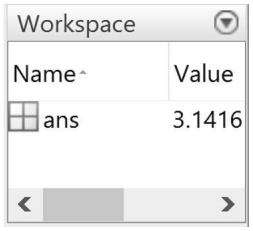


Bild 2.4 Aktualisierung

Damit kann, wie auf dem Taschenrechner, auch der Sinuswert bei 90 Grad ($\pi/2$) berechnet werden. Die Anweisung und Ergebnisanzeige haben folgendes Aussehen:

```
sin(pi/2)
ans =
   1
```

2.1.2 Wertezuweisung und Variablendefinition

Wertezuweisungen erfolgen mit dem Gleichheitszeichen (=). Die Befehlseingabe kann mit einem Komma (,) oder Semikolon (;) abgeschlossen werden. Nach Betätigung von „Enter“ erfolgt die Befehlsausführung. Zugewiesene Werte werden automatisch im MATLAB Arbeitsspeicher, dem „Workspace“, gespeichert. Berechnungen werden unmittelbar ausgeführt. Wird der Befehl ohne Semikolon oder mit einem Komma abgeschlossen, so erfolgt eine Anzeige des Befehlsergebnisses im „Command Window“. Bei einem Befehlsabschluss mit Semikolon erfolgt keine Anzeige. Die Variablen im Workspace werden im „Workspace Browser“ mit Wert (Value) und anderen beschreibenden Größen angezeigt. Die Größen, die angezeigt werden, sind abhängig von den durch den Nutzer im „Workspace Browser“ gewählten Einstellungen.

In MATLAB werden grundsätzlich englische Begriffe, Bezeichnungen und Notationen verwendet.

→ Dezimalzahlen: 0.5 **nicht:** 0,5

Als Variablennamen sind alphanumerische Zeichen, Unterstrich und nicht führende Nummern erlaubt.

In MATLAB werden folgende Zahlenformate unterstützt:

- Ganze Zahlen

  Beispiel: –2, 0, 100

- Dezimalzahlen

  Beispiel: –2.0, –1.25, 0.33

- Gleitkommazahlen

  Beispiel: $-1.25\ 10^{-3}$, $2\ 10^{9}$

  Die Eingabedefinition (Notation) ist eine Trennung von Matisse und Exponent mit dem Buchstaben e.

  Eingabe in MATLAB für die Beispiele: –1.25e-3, 2e9

- Komplexe Zahlen

  Beispiel: 3+2i, $-1.25+4\ 10^{9}$i

- Die Eingabedefinition (Notation) ist eine Summe aus Realteil und Imaginärteil. Beim Imaginärteil wird ein i angehängt. i ist dabei der Platzhalter für die imaginäre Einheit ($i^2 = -1$) und eine interne Konstante von MATLAB.
- Eingabe in MATLAB für die Beispiele: 3+2i, -1.25+4e9i

Es gibt alternative Eingabemöglichkeiten. So kann z. B. die Zahl 0.33 auch als .33, +0.33 oder +.33 eingegeben werden. Die rechnerinterne Darstellung von Zahlen und das Arbeiten mit komplexen Zahlen in MATLAB werden im Kapitel 4 (Zahlenformate) behandelt.

Am Beispiel aus Bild 1.15 sollen die ersten Schritte erklärt werden. Es wird von einer Masse von 1200 kg ausgegangen, auf die eine konstante Kraft von 800 N wirkt. Die Anweisungen für die Wertezuweisung zu den Variablen für die Masse (m) und die Kraft (F) lauten:

```
m=1200;
F=800;
```

Für die Berechnung der konstanten Beschleunigung ist die Anweisung:

```
a=F/m,
```

Da die Anweisung mit einem Komma abgeschlossen ist, erfolgt eine Ausgabe des berechneten Beschleunigungswertes.

```
a =
    0.6667
```

Nach diesen Anweisungen stellen sich das „Command Window“, der „Workspace Browser“ und die „Command History“ wie folgt dar (Bild 2.5).

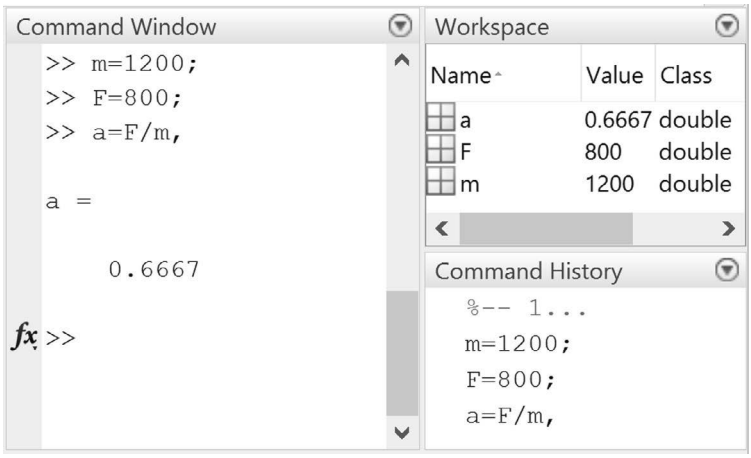


Bild 2.5 „Command Window“, „Workspace Browser“ und „Command History“

Allgemein gilt für Variablenzuweisungen (Variablendefinitionen) in MATLAB:

Name = Class(Value)

Im Deutschen werden folgende Begriffe verwendet:

Variablenname = Datentyp(Wert)

Der Name einer Variable sagt nichts über deren „Inhalt“ (Datentyp, Class) aus. Es kann sich unter anderem um eine Zahl, ein Textzeichen oder einen Wahrheitswert handeln.

MATLAB erlaubt es, dass der Datentyp (Class) nicht angegeben wird. Es wird automatisch der passende Datentyp gewählt. Zunächst soll nur die automatische Zuweisung des Datentyps benutzt werden. Im Kapitel 4 „Zahlenformate“ wird beispielhaft für Zahlen erläutert, wie Datentypen vom Nutzer definiert werden können und welche Auswirkungen dies auf den Speicherplatzbedarf hat. Bei Zahlenwerten wird von MATLAB automatisch der Datentyp (Class) „double“ gewählt.

Nachdem die Zuweisung eines Zahlenwertes oben bereits erläutert wurde, soll im Folgenden die Zuweisung von Textzeichen und Wahrheitswerten gezeigt werden. Die Zuweisung von Textzeichen oder Zeichenketten (Englisch: character oder string) erfolgt durch Setzen des zuzuweisenden Textes in Hochkomma:

```
s='Hochschule Rosenheim';
```

Der Variablenname ist s (Name: s), der Datentyp wird durch die Hochkommas automatisch character (Class: char) und der Wert ist Hochschule Rosenheim (Value: ‚Hochschule Rosenheim‘).

Die Zuweisung von Wahrheitswerten (Englisch: logical values), auch boolesche Werte genannt, die nur wahr (true) oder falsch (false) als Wert annehmen können, ist:

```
x=true;
```

Der Variablenname ist x (Name: x), der Datentyp ist logical (Class: logical) und der Wert ist 1 (Value: 1). Bei „false“ wäre der Wert 0.

Den Workspace nach all diesen Zuweisungen zeigt Bild 2.6. Dort ist tabellarisch für alle bis dato definierten Variablen der Variablenname (Name), der Datentyp (Class) und der Wert (Value) übersichtlich dargestellt. An den Icons am jeweiligen Namen ist auch bereits der Datentyp erkennbar.

Welche Größen angezeigt werden und deren Reihenfolge kann vom Nutzer ausgewählt werden. Wie man zur Auswahl der Größen gelangt, zeigt Bild 2.7.

Workspace

Name	Value	Class	Bytes	Size
a	0.6667	double	8	1x1
F	800	double	8	1x1
m	1200	double	8	1x1
s	'Hochschule Rosenheim'	char	40	1x20
x	1	logical	1	1x1

Bild 2.6 Workspace nach Zuweisungen von Zahlen, eines Textes und eines Wahrheitswertes

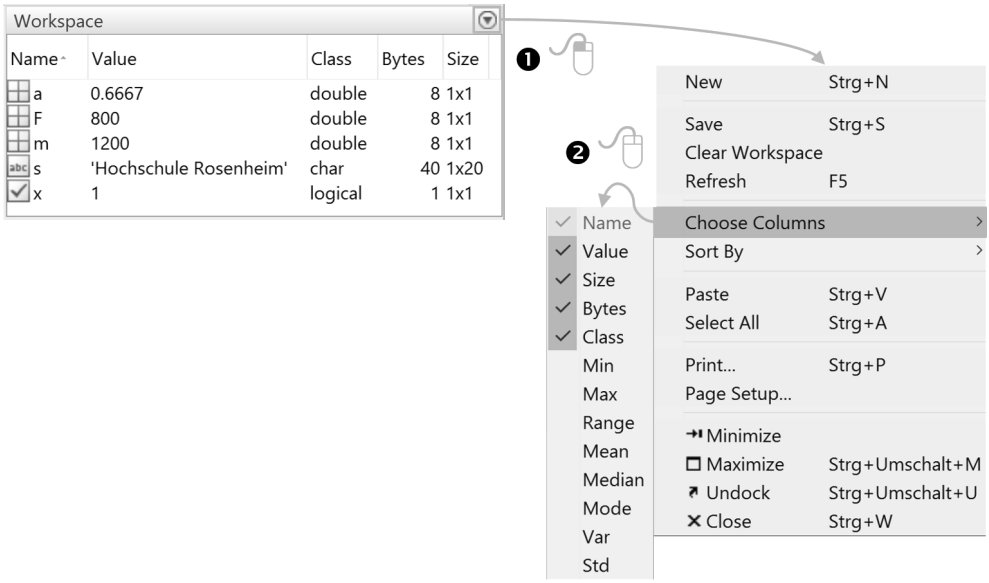


Bild 2.7 Auswahl der Größen, die im „Workspace Browser“ angezeigt werden

Bei Namen von Variablen ist es aus Gründen der Übersichtlichkeit und der geringeren Gefahr von Verwechselungen und damit Fehlern vorteilhaft, keine Abkürzungen, sondern sogenannte „sprechende Namen“ zu verwenden. Dies bedeutet zwar zunächst mehr Schreibaufwand, aber steigert insgesamt die Arbeitseffizienz.

Beim obigen einfachen Beispiel für die Berechnung der Beschleunigung ist dies nicht zwingend erforderlich. Mit sprechenden Variablennamen würde die Berechnung beispielsweise folgendermaßen aussehen.

```
Masse=1200;
Kraft=800;
Beschleunigung=Kraft/Masse;
```

Entsprechendes gilt für später behandelte Programm- und Dateinamen.

In Programmiersprachen, wie MATLAB, C oder C++, werden Zuweisungen von rechts nach links gelesen. So bedeutet:

```
F=F+200,
```

dass zum aktuellen Wert in der Variablen für die Kraft (Name: F; Value: 800, wie oben bereits definiert) der Wert 200 addiert wird. Der Ergebniswert 1000 wird nach links übergeben. Nach der Ausführung der Anweisung (Enter) erscheint im „Command Window“:

```
F =
   1000
```

Der Wert für die Kraft ist dann 1000 (Value: 1000) und wird auch so im „Workspace Browser“ angezeigt.

Anweisungen sind nicht als Gleichung im mathematischen Sinne zu deuten. Die oben gezeigte Anweisung wäre mathematisch falsch.

Zahlenvariablen haben keine Einheiten, sondern nur einen Wert. Für die Kraft kann nicht F = 100 N zugewiesen werden. Derjenige, der eine Variable definiert, muss selbst entscheiden, in welcher Einheit der Wert angegeben wird, und sich diese „merken“ und „dokumentieren“. Bei physikalischen Größen wird grundsätzlich empfohlen bei Werten von Variablen SI-Einheiten (Système international d’unités, Internationales Einheitensystem) zu nutzen. Dabei können Basiseinheiten oder daraus abgeleitete Einheiten mit besonderem Namen verwendet werden (Tabelle 2.2 und Tabelle 2.3). Dies ist insbesondere bei komplexeren Programmen übersichtlicher und weniger fehleranfällig.

Tabelle 2.2 SI-Einheiten – Basisgrößen und Basiseinheiten

Basisgröße und Dimensionsname	Größensymbol	Einheit	Einheitenzeichen
Länge	l	Meter	m
Masse	m	Kilogramm	kg
Zeit	t	Sekunde	s
Stromstärke	I	Ampere	A
Thermodynamische Temperatur	T	Kelvin	K
Stoffmenge	n	Mol	mol
Lichtstärke	I_v	Candela	cd

Tabelle 2.3 SI-Einheiten – abgeleitete Einheiten mit besonderem Namen (Auszug)

Größe	Einheit	Einheitenzeichen	In anderen SI-Einheiten	In SI-Basiseinheiten
Winkel (Ebene)	Radiant	rad	m/m	1
Frequenz	Hertz	Hz		1/s
Kraft	Newton	N	J/m	kg m/s^2
Druck	Pascal	Pa	N/m^2	kg/(m s^2)
Energie	Joule	J	Nm, Ws	kg m^2/(s^2)
Leistung	Watt	W	J/s, VA	kg m^2/(s^3)
Elektrische Ladung	Coulomb	C		A/s
Elektrische Spannung	Volt	V	W/A; J/C	kg m^2/(A s^3)
Elektrischer Widerstand	Ohm	Ω	V/A	kg m^2/(A^2 s^3)

Bevor ein Variablenname in einer Anweisung auf der rechten Seite verwendet werden kann, muss der Variablen ein Wert zugewiesen werden. Im bereits benutzten Beispiel für die Berechnung der Beschleunigung ist folgende Reihenfolge nicht möglich und führt zu einer Fehlermeldung:

```
m=1200;
a=F/m;
Undefined function or variable 'F'.
F=800;
```

Die Variable ‚F' hat in der zweiten Anweisungszeile noch keinen Wert. Es nützt nichts, wie hier gezeigt, in der dritten Zeile der Variablen einen Wert zuzuweisen. Die zweite Zeile wird nicht mehr ausgeführt und damit kein Wert für die Beschleunigung berechnet.

Anweisungen werden von oben nach unten ausgeführt.

Fehlermeldungen werden grundsätzlich in roter Schrift in Englisch angezeigt.

Die Ursache von Fehlermeldungen ist immer zu beheben, bevor weitergearbeitet werden kann.

2.1.3 Hilfeunterstützung und elektronische Dokumentation

Die Eingabe von help im „Command Window" liefert eine Übersicht zu Themengebieten, für die Informationen zur Verfügung stehen. Wird das gesuchte Themengebiet angeklickt, gibt es jeweils weitere Informationen.

Index

D

E

F

Z