Eine nicht allzu lange Einführung in $\LaTeX\,2_\varepsilon$

Erstellt von
Stefan Krause (`s.krause@online.de`)

Korrekturgelesen von
Daniel Saure

Zweite, komplett überarbeitete Auflage
7. April 2012

Inhaltsverzeichnis

Vorwort

Mehr als zwei Jahre sind nun nach der ersten offiziellen Auflage vergangen. Die zweite Auflage ist zwar komplett überarbeitet, aber der Leser wird, mit Ausnahme des veränderten Layouts, nur an wenigen Details Unterschiede erkennen. Die Kapitel- und Abschnittsstruktur ist gleich geblieben, so dass das Inhaltsverzeichnis mehr oder weniger unverändert ist. Es erschien mir nicht sinnvoll, ein Werk, das in seinem Aufbau beliebt war, komplett umzustrukturieren. Statt dessen sind einige LaTeX-technische Korrekturen eingeflossen, und diese Einführung ist nun aktuell in einer neuen Rechtschreibung gesetzt.

Abgesehen von diesen anscheinend kleinen, aber für mich relevanten Veränderungen ist alles Wichtige gesagt. Meine Intention bei dieser Einführung ist weiterhin dieselbe, wie im folgenden Vorwort zur früheren Version genannt. Ich bedanke mich bei meinen treuen Leserinnen und Lesern, die mir immer wieder bestätigt haben, dass ihnen mein Werk tatsächlich weiterhilft, und die es auch an andere Freunde und Bekannte weiterempfohlen haben.

Einige Inhalte der alten Auflage habe ich noch nicht überarbeitet, so dass sie in diesem Dokument noch nicht enthalten sind. Zu diesem Zwecke bleibt die alte Auflage noch auf dem Server verfügbar, solange sie Informationen bietet, die es noch nicht in dieses Dokument geschafft haben. Ferner gibt es an manchen Stellen fehlende Referenzen, die LaTeX durch **??** kenntlich macht. Diese beziehen sich auf eben diese Abschnitte, die in der neuen Auflage noch nicht existieren.

Wien, im Jänner 2009

Vorwort zur ersten Auflage

Ich habe mich dazu entschlossen, meine Einführung um ein Vorwort zu erweitern, weil ich darauf angesprochen wurde, daß ein solches fehlen würde. Dies gibt mir nun die Möglichkeit, ein paar allgemeinere Dinge niederzuschreiben, die primär nichts mit LaTeX zu tun haben. Aufgrund meiner Erfahrung, daß das Vorwort durchaus auch von anderen Personen als dem Autor des restlichen Textes verfaßt sein kann, habe ich den Empfängern meiner Mailingliste die Möglichkeit gegeben, sich zu LaTeX und meiner Einführung zu äußern. Diese Kommentare finden sich als Zitate auf der nächsten Seite.

Wie der Leser vielleicht schon im obigen Absatz bemerkt hat, ist diese Einführung in alter Rechtschreibung verfaßt. Das liegt daran, daß ich mich mit der neuen (oder neuesten oder welcher Version auch immer) nicht wirklich identifizieren kann, was mich aber natürlich nicht davon abhält, LaTeXs Sprachpakete für die neue Rechtschreibung zu erläutern.

Ich versuche, mit dieser Anleitung auf einem möglichst aktuellen Stand zu sein, d. h. daß ich primär diejenigen Befehle erkläre, die momentan empfohlen werden. Prinzipiell ist dies einer der Gründe, weshalb diese Einführung entstanden ist, da ich leider erfahren mußte, daß viele LATEX–Anleitungen im Internet veraltet sind. Ich überlasse es dem aufmerksamen Leser zu entscheiden, ob es mir besser gelungen ist. An einigen Stellen nenne ich die Namen veralteter Befehle mit der entsprechenden Bemerkung, daß sie vermieden werden sollen. Allerdings habe ich es genauso vermieden, dem LATEX–Benutzer vorzugaukeln, er müsse unbedingt dieses und jenes neueste Paket einbinden. Meine Erfahrung zeigt mir, daß die neuesten Pakete oder Zeichensätze nicht immer auf dem jeweiligen System verfügbar sind; ich spreche hier insbesondere die Pakete fontenc mit bestimmter Fontcodierung sowie lmodern an.

Selbstverständlich erhebt mein Dokument keinen Anspruch auf Vollständigkeit, statt dessen habe ich viele Dinge, die ich zunächst nicht für wichtig empfand, nach hinten verschoben oder (noch) gar nicht angesprochen. Deshalb erscheint es manchmal so, daß sich etwas, was inhaltlich offenbar zusammengehört, an zwei oder sogar drei unterschiedlichen Stellen wiederfindet. Der Grund ist, daß der Leser nach den ersten zwei Kapiteln wissen soll, wie er Wörter kursiv setzt, eine Abschnittsüberschrift erzeugt oder eine numerierte Liste generiert, aber sicherlich zunächst nicht damit konfrontiert werden möchte, wie er altdeutsche Schriften verwenden oder römisch numerierte Fußnoten erzeugen kann. Wer irgend etwas noch genauer wissen möchte, der kann entweder einer URL aus dem Anhang folgen, sich die offizielle Dokumentation (ebenfalls im Anhang) eines Pakets durchlesen oder einfach nach einem Befehl googlen.

In natürlicher Weise wird dem mathematischen Formelsatz – einer der Stärken von LATEX – relativ viel Raum gewährt; dies sind insbesondere die Kapitel 3–5. Allerdings ist es zunächst viel wichtiger, den normalen textlichen Rahmen zu beherrschen, von den richtigen Anführungszeichen bis zum Inhaltsverzeichnis. Trotz der vielen mathematischen Formeln sollen andere wichtige Themen wie Tabellen und Bilder natürlich nicht zu kurz kommen, aber auch ganz anderes wie mehrere Stichwortverzeichnisse, farbiger Text, Sütterlin–Buchstaben, japanische Schriftzeichen, Lautschrift, Musiknoten, usw. war mir wichtig. Leider ist einiges davon immer noch nicht fertiggestellt ...

Wiesbaden, im Sommer 2006

Kommentare

Auch eine kurze Danksagung soll an dieser Stelle nicht fehlen. Zum einen sind es in alphabetischer Reihenfolge Michael Diether, Marcel Gruner, Jens Mandavid und Christan Pausch, denen ich für mehr oder weniger tiefsinnige Diskussionen über LATEX danken möchte. Zum anderen möchte ich Tanja Becker, Anne Eisenbürger, Thorsten Essig, Mathias Grimm, Sebastian Immesberger, Patrick Jahn, Christopher Muth, Matthias Reinhardt, Alexander Rolbetzki, Daniel Saure und Christian Schmitz erwähnen, die diese Einführung durch Korrekturen, Vorschläge und Wünsche bereichert haben. Hier nun die Kommentare der Leser:

„Der Anfänger kann damit TEXen lernen, der Fortgeschrittene braucht es als ständiges Nachschlagewerk, und selbst wer gut TEXen kann, kann aus diesem Skript noch eine Menge lernen. Man findet fast alles, was man sucht (die Ausnahmen gibt's dann vom Autor persönlich), und noch viel mehr.“ *Tanja Becker*

„Diese TEX-Einführung, teilweise in didaktisch präziser Form, dann wieder mikroskopisch genau und insgesamt logisch strukturiert, richtet sich sowohl an Anfänger als auch an Fortgeschrittene. Ein Wunderwerk zum Lesen und Staunen, das auf keiner Festplatte fehlen sollte.“
Mathias Grimm

„Diese Einführung in LaTeX dient nicht nur Anfängern beim Erlernen der wichtigsten Befehle und Handgriffe zur Erstellung von PS-/PDF-Dokumenten, sondern dient vielmehr, dank des sehr umfassenden Stichwortverzeichnisses, auch fortgeschrittenen Benutzern als kompaktes Nachschlagewerk.“ *Sebastian Immesberger*

„Alles was der Mathematiker zum TeXen wissen muss, verflucht gutes Skript!!!“ *Patrick Jahn*

„Gut.“ *Matthias Reinhardt*

„Eine wunderbar verständliche Einführung in LaTeX, mit der man sehr schnell die grundlegenden Handgriffe beherrscht. Durch die regelmäßigen Aktualisierungen und Erweiterungen lernt man nie aus und befindet sich immer auf einem aktuellen Stand. Zudem ist sie durch das umfangreiche Stichwortverzeichnis ein wunderbares Nachschlagewerk für den Fall, dass man mal wieder was vergessen hat oder sich unsicher ist.“ *Sebastian Schlitt*

„Bevor ich dieses Skript gelesen habe, war ich arm, unglücklich und übergewichtig. Doch das ist nun alles vorbei! Stefans Skript hat mich besser, schneller und effizienter mit TeX arbeiten lassen, mich glücklich gemacht und sogar gegen meine Gewichts- und Geldprobleme geholfen. Es ist so vielfältig einsetzbar, dass ein Schweizer Messer blass vor Neid wird! Man kann daraus TeX lernen, es ausdrucken und den Ausdruck sogar verwenden, um Geschenke darin einzupacken. (Und das sind nur einige von vielen Möglichkeiten!) Alles in allem ist dieses Skript vermutlich die großartigste Erfindung seit dem ferngesteuerten Toilettenpapierabwickler. Danke, Stefan, mein erster Sohn wird deinen Namen tragen, auch wenn es eine Tochter wird!“ *Felix Schneider*

Kapitel 1

Grundlagen

1.1 LaTeX und diese Einführung

TeX ist ein Textsatzprogramm, das im Jahre 1977 von Donald E. Knuth entwickelt wurde und insbesondere auf wissenschaftliche Dokumente spezialisiert ist. Die Dokumente sind klein und portabel, weil sie in reinem ASCII-Text vorliegen; TeX-Compiler, die Postscript- oder PDF-Dateien erzeugen, gibt es für viele Betriebssysteme. An einigen wenigen Stellen merkt man TeX noch an, dass es von englischsprachigen Entwicklern stammt.

Da TeX relativ schwierig zu benutzen ist, hat Leslie Lamport 1985 eine Erweiterung entwickelt, die sich LaTeX nennt. Zusammen mit der American Mathematical Society (AMS), die sich um noch bessere mathematische Befehle bemühte, ist die aktuelle Version LaTeX 2_ε entstanden. Dies ist die momentan offizielle Version; was sich aus dem LaTeX3-Projekt ergibt, wird die Zukunft zeigen müssen.

Dem, der unter Windows LaTeX-Dokumente verfassen möchte, empfehle ich, folgende Programme in der angegebenen Reihenfolge zu installieren. All diese Programme sind frei im Internet erhältlich:

- GhostScript
  Aktuell[1]: GPL GhostScript 9.05 (etwa 13 MB)
  `http://sourceforge.net/projects/ghostscript/files/GPLGhostscript/9.05/`
- GhostView
  Aktuell: GhostView 5.0 (etwa 2 MB)
  `http://pages.cs.wisc.edu/~ghost/gsview/get50.htm`
- MikTex
  Aktuell: Basic MikTex 2.9 Installer (etwa 160 MB)
  `http://www.miktex.org/2.9/setup`
- TexnicCenter
  Aktuell: TexnicCenter 1.0 Stable RC1 (etwa 4,7 MB)
  `http://sourceforge.net/projects/texniccenter/files/TeXnicCenter/1.0StableRC1/`

Es bleibt zu erwähnen, dass es immer wieder neue Updates für MikTex gibt und dass es lohnenswert ist, diese auch wirklich zu installieren.

[1] Alle angegebenen aktuellen Versionen sind vom 2. April 2012.

Unter Linux lassen sich äquivalente Programme in der Regel problemlos über einen Paketmanager installieren. Meistens beinhaltet schon die Standardinstallation eine PostScript-Unterstützung (häufig auch GhostScript) sowie Viewer für DVI und PS. Die am weitesten verbreitete LATEX-Distribution unter Linux ist TexLive. Einfachere und mächtige Editoren mit Syntax- und Klammerhighlighting gibt es zu Hauf, sowohl für Gnome als auch KDE. Als vollständige Entwicklungsumgebungen seien Kile und Eclipse genannt. Von vielen dieser Programme gibt es Pendants für Windows als Alternative zum TexnicCenter.

Diese Einführung soll dazu dienen, die wichtigsten Befehle an Beispielen zu erläutern, ohne dass der Leser dabei mit länglichen Texten konfrontiert wird. Fast jedes Beispiel wird daher mit Quelltext und Ausgabe angegeben, und zwar immer in der folgenden Form:

`> Beschreibung von Eingabe und Ausgabe`
`=` Beschreibung von Eingabe und Ausgabe
Dabei kennzeichnet das `>` den Quelltext und das `=` das Ergebnis.

1.2 Befehle, Umgebungen und Pakete

\ { } LATEX-Befehle beginnen mit einem Backslash \ und dürfen nur Buchstaben enthalten, es sei denn, sie bestehen nur aus einem Zeichen, das dem Backslash folgt. Es wird zwischen Groß- und Kleinschreibung der Befehle unterschieden, und es gibt sogar viele Befehle, die sich nur dadurch unterscheiden. Die Argumente eines Befehls werden in geschweiften Klammern angehängt, für einen Befehl mit drei Argumenten z. B. `\befehl{eins}{zwei}{drei}`. Besteht ein Argument nur aus einem Zeichen, dann können die Klammern auch wegfallen, also z. B. `\befehl e{zwei}d`. Ist das Zeichen, das einem Befehlsnamen folgt, wieder ein Buchstabe (hier `e`), dann muss durch ein Leerzeichen deutlich gemacht werden, wo der Befehl aufhört. `\befehle{zwei}d` würde zu einem Fehler führen, da `\befehle` unbekannt ist. Leerzeichen, die an solchen Stellen im Quelltext stehen, werden daher nicht ausgegeben.

[ ] Die Argumente dürfen nicht einfach weggelassen werden; wenn für einen Befehl vereinbart wurde, dass er drei Argumente bekommt, dann muss er sie auch erhalten. Soll ein Argument leer bleiben, dann ist ein leeres Klammerpaar `{}` zu setzen. Einige Befehle besitzen optionale Argumente, die in eckigen Klammern übergeben werden, wie z. B. `\befehl[eins]{zwei}`. Sind die Argumente dann nicht vorhanden, wie z. B. in `\befehl{zwei}`, verwendet LATEX Standardwerte für den Befehl. Insbesondere dürfen die eckigen Klammern beim Fehlen der Argumente nicht eingegeben werden.

\begin \end {...} Umgebungen beginnen mit `\begin{name}` und enden mit `\end{name}`; darin ist `name` der Name der Umgebung. Auch Umgebungen dürfen Argumente erhalten, die in der Form `\begin{name}{arg1}` übergeben werden, der Befehl zum Beenden der Umgebung jedoch niemals. Prinzipiell können Umgebungen beliebig geschachtelt werden. Es gibt einige Befehle, die nur innerhalb von bestimmten Umgebungen verwendet werden dürfen. Das Klammerpaar `{` und `}`, ohne dass davor ein Befehl steht, heißt anonyme Umgebung. Sie kann benutzt werden, um einigen Befehlen einen Geltungsbereich zuzuweisen, was später noch deutlich werden wird.

\usepackage Ähnlich wie in anderen Programmiersprachen stehen nicht alle Befehle standardmäßig zur Verfügung. Viele Befehle müssen durch Einbinden von sog. Paketen erst „nachgeladen“ werden, damit sie benutzt werden können. In der Regel lassen sich Pakete an der Dateierweiterung `sty` erkennen. Eingebunden werden Pakete durch den Befehl `\usepackage{name}`, wobei auch mehrere Pakete durch Komma getrennt angegeben werden können. Manche Pakete können Einstellungen erhalten, die durch ein optionales Argument in der Form `\usepackage[einst]{name}` übergeben werden.

1.3 Das LaTeX 2ε-Dokument

Ein LaTeX 2ε-Dokument besteht aus einer Präambel (zu vergleichen mit einem Header, z. B. bei HTML) und einem Hauptteil. Das kleinstmögliche Dokument hat daher folgende Gestalt:

```
> \documentclass{article}
  \begin{document}
  \end{document}
```

\documentclass document

Die Präambel besteht hier nur aus der verpflichtenden ersten Zeile, die ein LaTeX 2ε-Dokument charakterisiert. Wer eine (fremde) Datei vor sich hat, die mit `\documentstyle` beginnt, ist mit einem älteren/veralteten LaTeX-Dokument konfrontiert. Der Präambel folgt als Hauptteil eine Umgebung mit dem Namen `document`.

Die Präambel dient dem Auswählen der Dokumentenklasse, Einbinden von Paketen, Festlegen der Sprache(n), Einstellen von Seitenrändern, Definieren von eigenen Befehlen usw. Einige Befehle dieser Art können aber auch noch im Hauptteil benutzt werden. Innerhalb der `document`-Umgebung – und nur dort – wird der zu druckende Text angegeben; in der Präambel darf kein Befehl stehen, der in diesem Moment eine Ausgabe erzeugen würde.

Compiliert man das obige Beispiel, dann erhält man zwar 0 Fehler, 0 Warnungen und 0 zu volle/leere Boxen, aber allerdings auch 0 Seiten. Das liegt einfach daran, dass keine echte Ausgabe erzeugt wird; selbst einige Leerzeichen würden LaTeX nicht dazu bringen, eine Leerseite anzulegen. Wie man zunächst auf ganz naive Weise Text einfügen kann, erläutern wir in Abschnitt 1.4.

Der einleitende Befehl `\documentclass` besitzt ein Argument, das die Dokumentenklasse beschreibt. Diese kann eine der vier Standardklassen letter, article, report oder book sein. Grundsätzlich stehen noch viel mehr allgemeine und spezielle Klassen zur Verfügung; es ist sogar möglich, eigene Klassendateien zu erstellen. `\documentclass` kann ein optionales Argument erhalten, mit dem globale Einstellungen festgelegt werden.

letter article report book

1.3.1 Dokumentoptionen

Für die Papiergröße stehen `a4paper` (DIN A4), `a5paper` (DIN A5), `b5paper` (DIN B5), `letterpaper` (Letter), `legalpaper` (Legal) und `executivepaper` (Executive) zur Verfügung. Eines davon sollte unbedingt angegeben werden.

a4paper a5paper b5paper letterpaper legalpaper executivepaper

Für den Standardschriftgrad stehen `10pt` (10 Punkte; Standard), `11pt` (11 Punkte) und `12pt` (12 Punkte) zur Verfügung. In mindestens 90 % der Fälle ist 11 Punkte die geeignete Wahl. Wenn es besonders schön, wichtig oder gut lesbar sein soll oder wenn in der Mathematik viele doppelte Indizes vorkommen, bietet sich auch 12 Punkte an. 10 Punkte ist in aller Regel zu klein und eignet sich für Texte, die einfach nur weniger Papier brauchen sollen.

10pt 11pt 12pt

Die Option `oneside` (Standard bei letter, article und report) aktiviert den einseitigen Druck, `twoside` (Standard bei book) aktiviert den doppelseitigen Druck. Beim doppelseiten Druck bewirkt `openright` (Standard bei book), dass Kapitel immer auf rechten Seiten beginnen, während `openany` (Standard bei report) den Beginn von Kapiteln links und rechts zulässt.

oneside twoside openright openany

Mit `onecolumn` (Standard) wird das gesamte Dokument einspaltig gesetzt, mit `twocolumn` zweispaltig. Diese Option ist nur selten zu empfehlen, da es eine Umgebung gibt, die Textstücke in mehrere Spalten setzen kann (auch mehr als nur 2).

onecolumn twocolumn

Wenn man einen Titel (mit Dokumentenname, Autor, Datum, usw.) erstellen möchte, dann bewirkt `titlepage` (Standard bei report und book), dass dieser Titel auf einer separaten ersten Seite gesetzt wird, und `notitlepage` (Standard bei article), dass der Text des Dokuments direkt unter dem Titel beginnt.

titlepage notitlepage

final
draft

Mit `final` (Standard) wird das Dokument ganz normal als endgültige Fassung gesetzt, mit `draft` werden für Bilder nur Platzhalter eingesetzt, was Zeit beim Compilieren, Speicherplatz, Zeit zum Anzeigen auf dem Monitor und Toner spart. Zusätzlich zeigt `draft` zu volle/leere Boxen mit einem Balken am Rand an.

fleqn
reqno
leqno

Abgesetzte mathematische Formeln werden standardmäßig zentriert, mit `fleqn`[2] können sie linksbündig (mit etwas Einzug) plaziert werden. Mit `reqno`[3] (Standard) stehen die Formelnummern am rechten Rand, mit `leqno`[4] am linken Rand.

Mehrere optionale Argumente werden durch Kommata getrennt, auf deren Reihenfolge kommt es nicht an. Die erste Zeile könnte also folgendermaßen aussehen:

> `\documentclass[a4paper,11pt,twoside,twocolumn,leqno]{book}`

1.4 Eingabe von Text

Gewöhnlicher Fließtext kann ganz normal eingegeben werden. Darunter fallen

> `ABCDEFGHIJKLMNOPQRSTUVWXYZ`
= ABCDEFGHIJKLMNOPQRSTUVWXYZ
> `abcdefghijklmnopqrstuvwxyz`
= abcdefghijklmnopqrstuvwxyz
> `0123456789`
= 0123456789
> `.,:;!?/()[]'+*=@`
= .,:;!?/()[]'+*=@

Alle anderen Zeichen haben Sonderbedeutungen, die im Laufe der Einführung klar werden. Wie man trotzdem ein Dollarzeichen oder geschweifte Klammern erzeugt, wird im Abschnitt 1.5 erläutert. Länderspezifische Zeichen wie Anführungszeichen oder die deutschen Umlaute sind in Abschnitt 1.6 beschrieben.

Tippt man einfach mal einen etwas längeren Text als Quelltext ein, so stellt man folgende Dinge fest, die LATEX automatisch tut:

- Der ganze Text erscheint im Blocksatz.
- Um den Blocksatz besser aussehen zu lassen, wird eine automatische Silbentrennung durchgeführt. Diese ist allerdings zunächst auf englische Sprache eingestellt.
- Die Seiten werden unten in der Mitte nummeriert.

%

Insbesondere werden alle Zeichen, die nicht in der obigen Tabelle aufgeführt sind, gar nicht bzw. falsch angezeigt oder rufen eine Fehlermeldung hervor. Das Prozentzeichen z. B. teilt LATEX mit, dass der Rest der Zeile als Kommentar anzusehen und damit zu ignorieren ist:

> `Einleitung:  % Hier geht's los!`
= Einleitung:

LATEX beherrscht auch noch andere Kleinigkeiten beim Setzen der Buchstaben. Zum einen wird der Abstand zweier Buchstaben individuell an die Buchstaben angepasst, z. B.

> `To T{}o`
= To To

[2] `fleqn` steht für „flush left equations".
[3] `reqno` steht für „right equation numbers".
[4] `leqno` steht für „left equation numbers".

Das verhindert unschöne Leerräume zwischen zwei Buchstaben. Mit dem Klammerpaar `{}` wird der individuelle Abstand zwischen dem „T“ und „o“ unterbunden, was hier als Demonstration dienen soll.

Zum anderen wird ein „f“ mit einem nachfolgenden „f“, „i“ oder „l“ zu einer sog. Ligatur verschmolzen. Eine solche Ligatur kann, wenn sie unerwünscht ist, mit `\/` unterbunden werden: \/

```
> Koffer finden fliegen Sauerstoffflasche auffliegen auffinden auf\/finden
```
= Koffer finden fliegen Sauerstoffflasche auffliegen auffinden auffinden

1.5 Spezielle Zeichen

In LaTeX gibt es vier verschiedene Arten, einen Wortzwischenraum zu erzeugen. In den allermeisten Fällen wird dies einfach das Leerzeichen sein: Leerzeichen

```
> mehrere Woerter durch Leerzeichen getrennt
```
= mehrere Woerter durch Leerzeichen getrennt

Mehrere Leerzeichen hintereinander bewirken nur ein einziges:

```
> mehrere Leerzeichen     veraendern nichts
```
= mehrere Leerzeichen veraendern nichts

Damit die Zeilen im Quelltext nicht zu lang werden, kann man beliebig Zeilenumbrüche einfügen; diese erzeugen jeweils einen Wortzwischenraum. Möchte man den Zwischenraum unterbinden, kann man am Ende der Zeile `%` verwenden, um den Zeilenumbruch auszukommentieren: Zeilenumbruch

```
> a
  b%
  c
```
= a bc

Die Tilde `~` wird verwendet, wenn ein festes Leerzeichen gesetzt werden soll, d. h. eines, an dem nicht in die nächste Zeile umgebrochen werden darf/soll. Dies eignet sich z. B. in folgenden Situationen: ~ \,

```
> siehe S.~38; Herr B.~Meier; d.\,h. also z.\,B. gar nichts
```
= siehe S. 38; Herr B. Meier; d. h. also z. B. gar nichts

Für Abkürzungen, in denen mehrere Teile mit einem Punkt abgekürzt werden, soll ein kleinerer Abstand gesetzt werden, der mit `\,` erzeugt wird.

Ein Backslash mit folgendem Leerzeichen, also `\␣`, wird benötigt, wenn ein ganz normales Leerzeichen nach einem Befehl ohne Argumente gesetzt werden soll, denn normalerweise werden Leerzeichen nach einem solchen Befehl ignoriert, weil sie das Ende des Befehlsnamens kennzeichnen. Bei den folgenden vier wichtigen (!) Befehlen ist dies der Fall; `\AmS` befindet sich im Paket amsmath: \TeX \LaTeX \LaTeXe \AmS

```
> \TeX\ ist alt; \LaTeX\ ist toll; \LaTeXe\ ist neu; \AmS\ macht Formeln
```
= TeX ist alt; LaTeX ist toll; LaTeX 2_ε ist neu; $\mathcal{AMS}$ macht Formeln

LaTeX unterscheidet vier verschiedene Zeichen für halbhohe Striche: - -- ---

```
> Trenn-strich, Binde--strich, 2--3 Eier, langer Gedanken---strich, $-1$
```
= Trenn-strich, Binde–strich, 2–3 Eier, langer Gedanken—strich, -1

`-` erzeugt einen Trennstrich, wie er am Zeilenende bei getrennten Wörtern benutzt wird. `--` kann man als Bindestrich für zusammengesetzte Wörter (natürlich ist hier auch `-` möglich) oder als kurzen Gedankenstrich verwenden. Er sollte immer als Bis-Strich (10–12 Uhr, u. ä.) verwendet werden. `---` wird meist in englischsprachigen Texten ohne Leerzeichen als langer

Gedankenstrich eingesetzt. - im Mathematikmodus (daher die beiden `$`) erzeugt ein Minuszeichen, das genauso lang ist wie das entsprechende Pluszeichen in +1.

\dots

Statt mehrere Punkte hintereinander einzugeben, kann man mit `\dots` Auslassungspunkte erzeugen:

```
> So nicht ..., aber so \dots
```
= So nicht ..., aber so …

\# \$ \% \& _ \{ \}

Einige Zeichen können in LaTeX nicht direkt gesetzt werden, weil sie eine besondere Bedeutung besitzen. Dazu gehören insbesondere der Backslash und die geschweiften Klammern, aber auch noch ein paar andere. In der Regel setzt man einfach einen Backslash vor das Zeichen:

```
> \# \$ \% \& \_ \{ \}
```
= # $ % & _ { }

\textasciicircum \textless \textgreater \textbackslash \textasciitilde \textbar \textmu

Einige Zeichen werden durch längere Befehle erzeugt:

```
> \textasciicircum\ \textless\ \textgreater\ \textbackslash\ \textasciitilde
  \textbar\ \textmu
```
= ˆ < > \ ˜ | µ

Der Befehl `\textmu` befindet sich im Paket `textcomp`.

\S \textperthousand \textcopyright \textregistered \texttrademark

Einige besondere Befehle sind:

```
> \S\ \textperthousand\ \textcopyright\ \textregistered\ \texttrademark
```
= § ‰ © ® ™

Der Befehl `\textperthousand` befindet sich im Paket `textcomp`, die letzten drei angegebenen Befehle werden von diesem Paket verändert.

fontenc T1 OT1

Um von vielen Vorteilen profitieren zu können, sollte das Paket `fontenc` mit der Option `T1` eingebunden werden. Außerdem ist der T1-Schriftsatz aktueller als der noch standardmäßig aktivierte OT1-Schriftsatz, der auch mit `OT1` benutzt werden kann. In einigen Zeichen unterscheiden sich die Codierungen, manche Zeichen sind mit OT1 auch schlicht nicht möglich. Diese Stellen werde ich entsprechend kennzeichnen. Das erste Beispiel dazu ist, dass mit der T1-Codierung auch direkt < und > für das Kleiner- und Größerzeichen benutzt werden kann.

1.6 Länderspezifische Sonderzeichen

In vielen Sprachen – ebenfalls im Deutschen – werden bestimmte Sonderzeichen in der Schrift benötigt. Dazu gehören einerseits akzentuierte Buchstaben (wie „ä“), zusätzliche Buchstaben (wie „ß“), aber andererseits auch Anführungszeichen.

1.6.1 Akzente und besondere Buchstaben

Grundsätzlich kann jeder Akzent über jeden Buchstaben gesetzt werden, auch wenn es diese Kombination in keiner Sprache geben sollte. Die Akzente werden horizontal über den Buchstaben zentriert. Je nach Höhe des Buchstabens verändert sich die vertikale Position des Akzents.

\i \j

Um über ein „i“ bzw. „j“ einen Akzent zu setzen, gibt es die beiden folgenden Befehle, die ein „i“ bzw. „j“ ohne Punkt erzeugen:

```
> \i\ \j
```
= ı ȷ

\' \` \^

Akut, Gravis und Zirkumflex:

```
> \'a \'y \'c \'A \'Y \'C \`e \`u \`m \`E \`U \`M \^o \^s \^i\ \^O \^S \^I
```
= á ý ć Á Ý Ć è ù m̀ È Ù M̀ ô ŝ î Ô Ŝ Î

Punkt, Trema (deutsche Umlaute) und Tilde:
`> \.e \.r \.m \.E \.R \.M \"u \"x \"s \"U \"X \"S \~a \~o \~n \~A \~O \~N`
= ė ṙ ṁ Ė Ṙ Ṁ ü ẍ s̈ Ü Ẍ S̈ ã õ ñ Ã Õ Ñ

\. \" \~

Umgekehrter Zirkumflex, Strich für langen Vokal und Bogen für kurzen Vokal:
`> \v c \v s \v z \v C \v S \=a \=\i\ \=u \=A \=I \u e \u o \u y \u E \u O`
= č š ž Č Š ā ī ū Ā Ī ĕ ŏ y̆ Ĕ Ŏ

\v \= \u

Zwei Striche, Kringel und Bogen zu folgendem Buchstaben:
`> \H o \H n \H s \H O \H N \r a \r x \r p \r A \r X \t ae \t oe \t AE`
= ő n̋ s̋ Ő N̋ å x̊ p̊ Å X̊ a͡e o͡e A͡E

\H \r \t

Cedille und Ogonek unter dem Buchstaben:
`> \c c \c s \c t \c C \c S \k e \k a \k n \k E \k A`
= ç ş ţ Ç Ş ę ą n̨ Ę Ą

\c \k

Punkt und Strich unter dem Buchstaben:
`> \d o \d r \d k \d O \d R \b u \b n \b p \b U \b N`
= ọ ṛ ḳ Ọ Ṛ u̱ ṉ p̱ U̱ Ṉ

\d \b

Das Ogonek ist nur in der T1-Codierung verfügbar.

Die vier drunterstehenden Akzente können auch auf einen Buchstaben angewendet werden, der schon einen anderen Akzent auf sich trägt:
`> \c{\v c} \k{\'e} \d{\'a} \b{\~n}`
= ç̌ ę̀ ạ́ ṉ̃

Weitere besondere Buchstaben:
`> \aa\ \AA\ \ae\ \AE\ \oe\ \OE\ \o\ \O\ \l\ \L\ \ss\ \SS`
= å Å æ Æ œ Œ ø Ø ł Ł ß SS

Das ß verändert sich beim Wechsel von OT1 zu T1 merklich. Nur in T1 verfügbar sind:
`> \dh\ \DH\ \dj\ \DJ\ \ng\ \NG\ \th\ \TH`
= ð Đ đ Đ ŋ Ŋ þ Þ

\aa \AA \ae \AE \oe \OE \o \O \l \L \ss \SS \dh \DH \dj \DJ \ng \NG \th \TH

Um LaTeX mitzuteilen, dass wir ein Dokument in deutscher Sprache schreiben wollen, muss man das Paket german oder ngerman einbinden. Das erste steht für die alte Rechtschreibung, das zweite für die neue, was z. B. wegen der Silbentrennung entscheidend ist. Ich werde im Folgenden (n)german schreiben, wenn ich eines dieser beiden Pakete meine.

Mit dem Paket (n)german ist es möglich, die komplizierte Eingabe der deutschen Umlaute sowie des ß abzukürzen:
`> "a "o "u "A "O "U "s`
= ä ö ü Ä Ö Ü ß

Davon sollte man bei deutschen Texten immer Gebrauch machen.

german ngerman "

Das Paket inputenc, das in der Form
`> \usepackage[latin1]{inputenc}`
eingebunden werden kann, erleichert die Eingabe von Sonderzeichen noch stärker (unabhängig von (n)german). Es ermöglicht, im Prinzip alle auf der Tastatur erhältlichen Zeichen direkt eingeben zu können; dazu gehören á, è und î, aber auch insbesondere die deutschen Umlaute und das ß. Allerdings bleiben die Quelltexte damit nicht mehr zwangsläufig plattformunabhängig. So kann es passieren, dass die Umlaute eines unter Windows erstellten Quelltexts nicht mehr richtig unter Linux dargestellt werden können. Wer seine Quelltexte auf mehreren Plattformen benutzen möchte und sich gleichzeitig unsicher ist, sollte auf inputenc verzichten.

inputenc latin1

1.6.2 Andere Sonderzeichen

“ ” ‘ ’

Die englischen Anführungszeichen werden folgendermaßen gesetzt:
`> ‘‘English quotation marks’’ or ‘English quotation marks’`
= “English quotation marks” or ‘English quotation marks’

\glqq \grqq \glq \grq \flqq \frqq \flq \frq "‘ "’ "< ">

Mit dem Paket (n)german erhält man die deutschen[5] und die französischen[6] Anführungszeichen durch:
`> \glqq deutsche Zeichen\grqq\ oder \glq deutsche Zeichen\grq`
= „deutsche Zeichen“ oder ‚deutsche Zeichen‘
`> \flqq guillemets\frqq\ ou \flq guillemets\frq`
= «guillemets» ou ‹guillemets›
Oder die doppelten einfacher durch:
`> "‘Anführungszeichen"’ "<guillemets">`
= „Anführungszeichen“ «guillemets»

!‘ ?‘ < >

Die umgedrehten spanischen Satzzeichen können ebenfalls gesetzt werden, allerdings abhängig von der Codierung. In T1 muss man `!‘` und `?‘` verwenden:
`> !‘Eso es! ?‘Qué pasó?`
= ¡Eso es! ¿Qué pasó?
In OT1 ist es < und >:
`> <Eso es! >Qué pasó?`
= ¡Eso es! ¿Qué pasó?

\textsterling \euro \textyen

Mit dem Paket eurosym kann das Euro-Symbol benutzt werden, mit textcomp das Yen-Symbol:
`> 150~\textsterling, 100~\euro, 80~\textyen`
= 150 £, 100 €, 80 ¥

[5] `glq` steht für „German left quote“, `grq` für „German right quote“. Die zwei `q` stehen für doppelte Anführungszeichen.
[6] `flq` steht für „French left quote“, `frq` für „French right quote“. Die zwei `q` stehen für doppelte Anführungszeichen.

Kapitel 2

Formatierung und Gliederung

2.1 Schriftformatierung

Mit `\textsf`[1] schaltet man auf die serifenlose und mit `\texttt`[2] auf die Typewriterschrift um: \textsf \texttt \textrm

```
> Beispiel \textsf{Beispiel} \texttt{Beispiel}
```

= Beispiel Beispiel `Beispiel`

Mit `\textrm`[3] kann man innerhalb einer dieser Schriftfamilien in die Standardschrift zurückschalten:

```
> \textsf{Zurückschalten \textrm{in die} Standardschriftart}
```

= Zurückschalten in die Standardschriftart

Mit `\textit`[4] schaltet man auf den kursiven, mit `\textsl`[5] auf den geneigten und mit `\textsc`[6] auf den Kapitälchenschnitt um: \textit \textsl \textsc \textup

```
> Beispiel \textit{Beispiel} \textsl{Beispiel} \textsc{Beispiel}
```

= Beispiel *Beispiel* *Beispiel* BEISPIEL

Mit `\textup`[7] kann man innerhalb einer dieser Schriftschnitte in den aufrechten zurückschalten:

```
> \textit{Zurückschalten \textup{in den} aufrechten Schriftschnitt}
```

= *Zurückschalten* in den *aufrechten Schriftschnitt*

Der Befehl `\emph`[8] wechselt vom aufrechten in den kursiven Schriftschnitt und umgekehrt: \emph

```
> Hiermit \emph{kann \emph{man hin-} und} zurückschalten.
```

= Hiermit *kann* man hin- *und* zurückschalten.

Mit `\textbf`[9] schaltet man in das fette Schriftgewicht: \textbf \textmd

```
> Beispiel \textbf{Beispiel}
```

= Beispiel **Beispiel**

[1] `sf` steht für „sans serif“.
[2] `tt` steht für „typed text“.
[3] `rm` steht für „roman“.
[4] `it` steht für „italic“.
[5] `sl` steht für „slanted“.
[6] `sc` steht für „small caps“.
[7] `up` steht für „upright“.
[8] `emph` steht für „emphasized“.
[9] `bf` steht für „bold face“.

Mit `\textmd`[10] kann man innerhalb des fetten Schriftgewichts in das leichtere zurückschalten:
`> \textbf{Zurückschalten \textmd{in das} leichtere Gewicht}`
= **Zurückschalten** in das **leichtere Gewicht**

\textnormal

Man kann `\textnormal` verwenden, um Familie, Schnitt und Gewicht jeweils auf den Standardwert zurückzusetzen:
`> \textnormal{Beispiel}`
bewirkt also dasselbe wie
`> \textrm{\textup{\textmd{Beispiel}}}`
Wie die Standardwerte verändert werden können, ist in Abschnitt 7.2 erklärt.

\tiny \scriptsize \footnotesize \small \large \Large \LARGE \huge \Huge

Als verschiedene Schriftgrade stehen zur Verfügung:
`> {\tiny Beispiel} {\scriptsize Beispiel} {\footnotesize Beispiel}`
= Beispiel Beispiel Beispiel
`> {\small Beispiel} {\large Beispiel} {\Large Beispiel}`
= Beispiel Beispiel Beispiel
`> {\LARGE Beispiel} {\huge Beispiel} {\Huge Beispiel}`
= Beispiel Beispiel Beispiel
Man beachte die abweichende Syntax: Die Befehle erhalten nicht den größer oder kleiner zu schreibenden Text als Argument in geschweiften Klammern; statt dessen werden der Befehl und der Text in eine anonyme Umgebung gesetzt. Eine geschweifte Klammer direkt nach dem Befehl wäre unnötig.

\normalsize

Man kann auch mit einem Befehl in den normalen Schriftgrad zurückschalten:
`> {\Large Zurückschalten in \normalsize den normalen Schriftgrad}`
= Zurückschalten in den normalen Schriftgrad

\textlarger \textsmaller

Das Paket relsize erlaubt eine Schriftgradveränderung relativ zur aktuellen Größe:
`> {\footnotesize Dies ist \textlarger{eine Stufe} größer.}`
= Dies ist eine Stufe größer.
`> {\Large Dies ist \textsmaller{eine Stufe} kleiner.}`
= Dies ist eine Stufe kleiner.
Diese Befehle arbeiten wieder mit dem Text als Argument.

Selbstverständlich lassen sich Schriftfamilie, -schnitt, -gewicht und -grad im Prinzip beliebig miteinander kombinieren:
`> \textit{ein\Large\ \textbf{sehr \textsf{viel} längeres} Beispiel}`
= *ein* ***sehr*** **viel** ***längeres*** *Beispiel*
… aber eben nur im Prinzip. Es gibt einige Kombinationen, für die es keine Schrift gibt, so z. B. hier für die fett-kursive serifenlose Schrift. In diesem Fall erhält man eine Compilerwarnung, und es werden geeignete vorhandene Schriften substituiert. Wer also standardmäßig die serifenlose Schrift eingestellt hat, kann nicht fett und kursiv gleichzeitig benutzen.

Alle Schriftgradbefehle können auch ohne \ als Umgebung verwendet werden, z. B.:
`> Schrift \begin{Large}Schrift\end{Large}`
= Schrift Schrift

\rmfamily \sffamily \ttfamily

Alternativ zu den `\text##`-Befehlen lassen sich Schriftfamilie, -schnitt und -gewicht auch durch Befehle einstellen, deren Syntax den Schriftgradbefehlen gleicht. Diese lauten für die Familien `\rmfamily`, `\sffamily` und `\ttfamily`, für die Schnitte `\upshape`, `\itshape`, `\slshape`

[10] `md` steht für „medium“.

und `\scshape` und für die Gewichte `\mdseries` und `\bfseries`. Die veralteten Befehle `\rm`, `\sf`, `\tt`, `\it`, `\sl`, `\sc` und `\bf` sollten nicht mehr benutzt werden, da sie nicht zum neuen Schriftsystem von LaTeX 2_ε kompatibel sind. Das Analogon zu `\textnormal` ist `\normalfont`.

\upshape
\itshape
\slshape
\scshape
\mdseries
\bfseries
\normalfont

2.2 Absatzformatierung

Der Beginn eines Absatzes

```
> wird von \LaTeX\ normalerweise um ein bestimmtes Maß eingezogen. Wer das
  nicht möchte, kann es auch global abstellen, vgl. Abschnitt~6.2.
```

= wird von LaTeX normalerweise um ein bestimmtes Maß eingezogen. Wer das nicht möchte, kann es auch global abstellen, vgl. Abschnitt 6.2.

Dieser Absatzeinzug wird auch in dieser Einführung verwendet. In allen Beispielen hingegen, die nicht zur Demonstration dieses Einzugs dienen, ist er abgestellt, weil sonst verwirrender Leerraum entstünde.

\\
*

Mit `\\` beginnt LaTeX eine neue Zeile, allerdings keinen neuen Absatz. Daher werden im folgenden Beispiel die zweite und dritte Zeile nicht eingerückt. Verwendet man statt `\\` den Befehl `\\*`, dann wird an diesem Zeilenumbruch kein Seitenwechsel zugelassen.

```
> Erste Zeile im Absatz; \\
  zweite Zeile im Absatz; \\
  dritte Zeile im Absatz.
```

= Erste Zeile im Absatz;
zweite Zeile im Absatz;
dritte Zeile im Absatz.

\newline
\linebreak

Statt `\\` kann auch `\newline` verwendet werden. Es gibt nur sehr wenige Situationen, in denen sie sich unterscheiden (z. B. Abschnitt 8.1). `\linebreak` führt ebenfalls einen manuellen Zeilenwechsel durch, behält aber den Blocksatz in der vorherigen Zeile bei:

```
> aaa bbb ccc \linebreak
  xxx yyy zzz
```

= aaa bbb ccc
xxx yyy zzz

Im Allgemeinen entsteht dabei – wie in diesem Beispiel – eine sog. „Underfull box".

Leerzeile
\par

Um einen neuen Absatz zu erzeugen, muss man eine Leerzeile eingeben:

```
> Erster Absatz.

  Zweiter Absatz.
```

= Erster Absatz.
Zweiter Absatz.

Dann sind beide Zeilen eingerückt, weil sie beide einen Absatzanfang darstellen. Statt einer Leerzeile kann man am Ende des Absatzes auch den Befehl `\par` eingeben; man benötigt dann nicht mal einen Zeilenumbruch im Quelltext. Zeilenwechsel (`\\`) und Absatzwechsel (Leerzeile bzw. `\par`) dürfen nie zusammen benutzt werden, weil dann eine „Underfull box" entstünde.

\noindent
\indent

Der Befehl `\noindent` vor dem Anfang eines Absatzes unterbindet den Einzug einmalig:

```
> Erster Absatz. \par
  \noindent Zweiter Absatz.
```

= Erster Absatz.
Zweiter Absatz.

Der Befehl `\indent` bewirkt genau das Gegenteil.

flushleft flushright center

Die standardmäßige Formatierung von Absätzen ist der Blocksatz; natürlich kann man ihn auch aufheben und statt dessen Flattersatz links- oder rechtsbündig oder auch zentrierten Text setzen. Die folgenden Umgebungen erzeugen jeweils einen neuen Absatz mit der gewünschten Einstellung:

`> \begin{flushleft} linksbündiger Text \end{flushleft}`

= linksbündiger Text

`> \begin{flushright} rechtsbündiger Text \end{flushright}`

= rechtsbündiger Text

`> \begin{center} zentrierter Text \end{center}`

= zentrierter Text

\raggedright \raggedleft \centering

Es gibt aber auch äquivalente Befehle, die mit derselben Syntax wie die Schriftgradbefehle `\large` usw. benutzt werden. Damit sie sich nicht über das ganze restliche Dokument erstrecken, müssen sie in eine anonyme Umgebung aus `{` und `}` gesetzt werden. Im Gegensatz zu den oben genannten Umgebungen erzeugen sie keinen neuen Absatz und auch keinen zusätzlichen Abstand. `\raggedright` setzt links(!)bündigen, `\raggedleft` rechts(!)bündigen und `\centering` zentrierten Text.

\centerline

Mit `\centerline` kann eine einzelne zentrierte Zeile erzeugt werden. Dieser Befehl ist allerdings Plain-TEX (und nicht LATEX) und sollte daher nicht benutzt werden. Als Alternative kann der Befehl `\hspace` aus Abschnitt 2.4 verwendet werden.

2.3 Silbentrennung

german ngerman

Passt ein Wort einerseits nicht mehr in die aktuelle Zeile und würde andererseits beim vollständigen Übernehmen dieses Wortes in die nächste Zeile zu viel Leerraum entstehen, dann trennt LATEX das Wort an geeigneter Stelle. Damit die richtigen Algorithmen angewendet werden, muss die richtige Sprache eingestellt werden. Dies geschieht durch Einbinden des Pakets german (alte Rechtschreibung) bzw. ngerman (neue Rechtschreibung). Eine manuelle Trennung wird gelegentlich dann notwendig, wenn die Wörter Umlaute, ß oder Bindestriche (z. B. „-“) enthalten, falsch getrennt werden oder an der vorgeschlagenen Stelle nicht getrennt werden sollen. Mit dem neuen T1-Schriftsatz werden die Trennhilfen bei Wörtern mit Umlauten oder ß fast immer überflüssig.

\-

Innerhalb eines Wortes kann man bedingte Trennstriche, oder anders gesagt, Trennhilfen angeben:

`> Der Ur\-instinkt der Tiere \dots`

= Der Urinstinkt der Tiere ...

Dies ist ein klassisches Beispiel dafür, dass eine korrekte Trennung vielleicht unerwünscht ist. `\-` bewirkt, dass das Wort nur noch dort getrennt werden darf, allerdings muss man dies bei jedem weiteren Auftreten desselben Worts wiederholen. Selbstverständlich darf `\-` mehrfach in einem Wort eingesetzt werden.

\mbox

Um die Trennung eines Wortes ganz zu verhindern, kann der Befehl `\mbox` verwendet werden:

`> Herbert \mbox{Maier}`

= Herbert Maier

Innerhalb von `\mbox` dürfen auch mehrere Wörter stehen; dann verhindert der Befehl zusätzlich, dass an den Leerzeichen ein Zeilenumbruch durchgeführt wird.

`"-` `""` `"=` `"~`

Das Paket (n)german bietet noch weitere Trennhilfen an: Mit `"-` kann eine zusätzliche Trennmöglichkeit definiert werden, wobei alle anderen erlaubt bleiben. Der Befehl `""` hingegen erlaubt eine bedingte Trennung, ohne dass ein Trennstrich eingesetzt wird. Als Beispiel soll Folgendes dienen:

```
> Die Java-Klasse \texttt{JButton} des Pakets \texttt{swing} besitzt eine
  Methode \texttt{JButton.""get""Vertical""Text""Position()}, die \dots
```

= Die Java-Klasse `JButton` des Pakets `swing` besitzt eine Methode `JButton.getVertical TextPosition()`, die ...

Wörter mit Bindestrichen können nur noch dort getrennt werden. Dieses Problem ist behoben, wenn man `"=` statt `-` benutzt. Mit `"~` dagegen kann ein Bindestrich gesetzt werden, an dem nicht getrennt werden darf. Ein komplizierteres Beispiel dazu ist `(Haupt"~)""Straße`.

\slash

Gelegentlich werden Wörter durch einen Schrägstrich verbunden, z. B. „Frankfurt/Main" oder „eine positive/negative Zahl". Um einen Zeilenumbruch nach dem Schrägstrich zu ermöglichen, kann man `/` durch `\slash` ersetzen, also beispielsweise `Frankfurt\slash Main`.

\hyphenation

Um die Trennung eines Wortes global für das ganze Dokument einzustellen, kann man in der Präambel Folgendes verwenden:

```
> \hyphenation{Ur-instinkt Maier}
```

Nun wird innerhalb des Dokuments das Wort „Urinstinkt" nur nach „Ur" und das Wort „Maier" niemals mehr getrennt. Mit dem T1-Schriftsatz können auch Wörter mit Umlauten und ß auf diese Weise richtig getrennt werden.

\discretionary

Der Vollständigkeit halber: Um nach der alten Rechtschreibung „Zucker" und „Schiffahrt" in „Zuk-ker" und „Schiff-fahrt" zu trennen, kann der Befehl `\discretionary` für eine frei definierbare Trennrichtlinie dienen:

```
> \discretionary{Zuk-}{ker}{Zucker} und Salz;
  \discretionary{Schiff-}{fahrt}{Schiffahrt} auf dem Rhein
```

= Zucker und Salz; Schiffahrt auf dem Rhein

Die ersten beiden Argumente geben die beiden Teile im Fall der Trennung an, das dritte das zusammengeschriebene Wort.

Noch ein Negativbeispiel:

```
> Allerdings können auch "‘zu leere"’ oder "‘zu volle Boxen"’ auftreten, wenn
  Bindestrich-Wortschöpfungen am Zeilenende stehen oder \mbox{wenn
  Zeilenumbrüche mit entsprechenden Befehlen unterdrückt werden,} so dass
  nicht alles in die Zeile passt.
```

= Allerdings können auch „zu leere" oder „zu volle Boxen" auftreten, wenn Bindestrich-Wortschöpfungen am Zeilenende stehen oder wenn Zeilenumbrüche mit entsprechenden Befehlen unterdrückt werden, so dass nicht alles in die Zeile passt.

Das Problem kann z. B. durch Eingabe von `Bindestrich"=Wortschöpfungen` beseitigt werden, und der Befehl `\mbox` wurde hier offensichtlich ohne Sinn verwendet.

2.4 Abstände

mm cm in em ex pt sp

Um Abstände und andere Längen festzulegen, unterstützt LaTeX u. a. folgende Maßeinheiten:

`mm`: Millimeter
`cm`: Zentimeter (= 10 mm)
`in`: Zoll (= 25,4 mm)
`em`: Breite des Zeichens „M" in der aktuellen Schrift
`ex`: Höhe des Zeichens „x" in der aktuellen Schrift

`pt`: Punkt ($= \frac{1}{72{,}27}$ in $\approx 0{,}35$ mm)
`sp`: Skalierter Punkt ($= \frac{1}{65\,536}$ pt), TEXs kleinste Einheit

\hspace \hspace* Horizontaler Abstand kann mit `\hspace` erzeugt werden. Der Leerraum wird unterdrückt, wenn er auf einen Zeilenwechsel fällt; wenn er trotzdem erscheinen soll, muss `\hspace*` verwendet werden:

```
> Name:\hspace{3cm}Vorname:\hspace{3cm}Geburtsdatum:
```
= Name: Vorname: Geburtsdatum:
```
> \hspace*{2cm}Name:
```
=   Name:

Wenn man hier Wert auf genaue Abstände legt, sollte man – wie im Beispiel – keine Leerzeichen vor bzw. nach den Befehlen eingeben. Die Folge ␣`\hspace{3cm}`␣ würde nämlich einen Abstand von 3 cm zuzüglich zwei Leerzeichenbreiten erzeugen.

\fill \hfill Statt einer expliziten Länge kann auch `\fill` benutzt werden; `\hspace{\fill}` kann zu `\hfill` abgekürzt werden:

```
> Grundlagen\hfill Seite~3
```
= Grundlagen   Seite 3
```
> \hspace*{\fill}Name: Mustermann\hfill Vorname: Max\hspace*{\fill}
```
=   Name: Mustermann   Vorname: Max

Alle in einer Zeile auftretenden `\fill`s werden so gedehnt, dass sie gleich lang sind und die Zeile insgesamt ausfüllen.

Mit diesem Befehl kann ein Absatz mit einer einzigen Zeile sehr einfach zentriert oder rechtsbündig gesetzt werden:

```
> \hspace*{\fill}rechts\par
  \hspace*{\fill}zentriert\hspace*{\fill}\par
  \noindent \hspace*{\fill}zentriert\hspace*{\fill}
```
=   rechts
zentriert
zentriert

Man beachte die in diesem Fall störende Wirkung des Absatzeinzuges.

\vspace \vspace* Analog zu den horizontalen Befehlen gibt es `\vspace`, `\vspace*` und `\vfill`, die vertikalen Leerraum erzeugen:

```
> \dots\ gewollt hätten. \vspace{3ex}

  Dabei handelt es sich \dots
```
= … gewollt hätten.

Dabei handelt es sich …

Vor oder nach `\vspace` muss auf jeden Fall noch ein Absatzwechsel (Leerzeile oder `\par`) stehen, weil der Befehl nur zwischen zwei Absätzen Abstand erzeugen kann.

\vfill `\vfill` eignet sich gut, wenn Text auf selbst erstellten Titelseiten verteilt werden soll:

```
> \vspace*{\fill}
  Einführung in \LaTeX \par
  \vfill
  Max Mustermann \par
  \vspace*{\fill}
```

=

Einführung in LaTeX

Max Mustermann

Um bei einem Zeilenwechsel innerhalb eines Absatzes zusätzlichen Abstand hinzuzufügen, kann der Befehl \\ (analog auch *) ein optionales Argument erhalten: \\ *

```
> aaaaa \\[2ex]
  bbbbb
```

= aaaaa

bbbbb

Statt `\vspace` kann man auch einen der folgenden Befehle verwenden, um etwas zusätzlichen Abstand zwischen zwei Absätzen zu erzeugen: \smallskip \medskip \bigskip

```
> aaaaa \par\smallskip
  bbbbb \par\medskip
  ccccc \par\bigskip
  ddddd
```

= aaaaa

bbbbb

ccccc

ddddd

Aufpassen muss man, wenn die nach einem \\ folgende Zeile mit dem Zeichen [beginnen soll. Dann würde nämlich der nach [stehende Text als optionales Argument interpretiert, und in der Regel wird das keine gültige Längenangabe sein, so dass es zu einer Fehlermeldung kommt. Das Problem lässt sich z. B. folgendermaßen lösen:

```
> Unterschrift \\{}
  [Max Mustermann]
```

= Unterschrift
[Max Mustermann]

Die leere Umgebung teilt LaTeX mit, dass kein optionales Argument von \\ kommen wird.

Neben `\hfill` gibt es noch zwei andere Befehle, um horizontalen Leerraum aufzufüllen: \hrulefill \dotfill

```
> Name: \hrulefill\ Vorname: \hrulefill\ Geburtsdatum: \hrulefill
```

= Name: ________________ Vorname: ________________ Geburtsdatum: ________________

```
> Name: \dotfill\ Vorname: \dotfill\ Geburtsdatum: \dotfill
```

= Name: .... Vorname: .... Geburtsdatum: ....

Eher seltsam und ungebräuchlich sind wohl die folgenden Füllbefehle, die ich der Vollständigkeit halber noch erwähne: \leftarrowfill \rightarrowfill \upbracefill \downbracefill

```
> A \leftarrowfill\ B \rightarrowfill\ C \upbracefill\ D \downbracefill\ E
```

= $A \xleftarrow{\qquad\qquad} B \xrightarrow{\qquad\qquad} C \underbrace{\qquad\qquad\qquad\qquad\qquad} D \overbrace{\qquad\qquad\qquad\qquad\qquad} E$

2.5 Aufzählungen

itemize \item

LaTeX unterstützt auf sehr einfache Weise das Erstellen von Aufzählungen. Die einfachste ist:

```
> \begin{itemize}
   \item Erster Eintrag
   \item Zweiter Eintrag
  \end{itemize}
```

=
- Erster Eintrag
- Zweiter Eintrag

Der Befehl `\item` leitet jeweils einen neuen Punkt ein. Zwischen `\begin{itemize}` und `\item` dürfen zwar noch Befehle stehen, allerdings keine, die eine Ausgabe produzieren, sonst gibt es den Fehler „`Something's wrong--perhaps a missing \item`".

Es werden vier Verschachtelungstiefen unterstützt:

```
> \begin{itemize}
   \item Erste Stufe
    \begin{itemize}
     \item Zweite Stufe
      \begin{itemize}
       \item Dritte Stufe
        \begin{itemize}
         \item Vierte Stufe
        \end{itemize}
      \end{itemize}
    \end{itemize}
  \end{itemize}
```

=
- • Erste Stufe
  - – Zweite Stufe
    - ∗ Dritte Stufe
      - · Vierte Stufe

Dabei nehmen der zusätzliche horizontale Einzug links und der Abstand der einzelnen Punkte zueinander von Ebene zu Ebene ab. Außerdem sind die Zeichen je nach Ebene unterschiedlich; das obige Beispiel zeigt die Standardeinstellung, die natürlich verändert werden kann (vgl. Abschnitt 6.4).

enumerate

Eine nummerierte Liste kann mit `enumerate` erzeugt werden:

```
> \begin{enumerate}
   \item Erster Eintrag
   \item Zweiter Eintrag
  \end{enumerate}
```

=
1. Erster Eintrag
2. Zweiter Eintrag

Auch hier sind vier Ebenen möglich:

```
> \begin{enumerate}
   \item Erste Stufe
    \begin{enumerate}
     \item Zweite Stufe
```

```
    \begin{enumerate}
     \item Dritte Stufe
      \begin{enumerate}
       \item Vierte Stufe
      \end{enumerate}
    \end{enumerate}
  \end{enumerate}
\end{enumerate}
```

=

1. Erste Stufe
   (a) Zweite Stufe
       i. Dritte Stufe
          A. Vierte Stufe

Auch hier kann man, wie wir später sehen werden (vgl. Abschnitt 6.4), das Format der Nummerierung ändern.

Die etwas seltener gebrauchte Aufzählung ist: description

```
> \begin{description}
   \item[Ente] Erster Eintrag
   \item[Fuchs] Zweiter Eintrag
  \end{description}
```

= **Ente** Erster Eintrag

Fuchs Zweiter Eintrag

Hier erhält `\item` ein optionales Argument, das aus den fett zu setzenden Wörtern besteht.

2.6 Dokumentgliederung und Inhaltsverzeichnis

LaTeX-Dokumente lassen sich, sofern sie nicht in der Klasse letter geschrieben sind, in Kapitel, Abschnitte und Unterabschnitte gliedern. In der Klasse article stehen beispielsweise die Befehle `\section`, `\subsection`, `\subsubsection`, `\paragraph` und `\subparagraph` zur Verfügung:

\section \subsection \subsubsection \paragraph \subparagraph

```
> \section{Naturwissenschaften} Biologie, Physik, Chemie, \dots
   \subsection{Biologie} Tiere, Pflanzen, \dots
    \subsubsection{Tiere} Säugetiere, Vögel, Reptilien, \dots
     \paragraph{Säugetiere} Katzen, Hunde, Nagetiere, \dots
      \subparagraph{Katzen} Tiger, Leopard, Hauskatze, \dots
```

= **1 Naturwissenschaften**

Biologie, Physik, Chemie, ...

1.1 Biologie

Tiere, Pflanzen, ...

1.1.1 Tiere

Säugetiere, Vögel, Reptilien, ...

Säugetiere Katzen, Hunde, Nagetiere, ...

Katzen Tiger, Leopard, Hauskatze, ...

In der Standardeinstellung erzeugen die ersten drei eine Überschrift mit darunter beginnendem Text, die beiden letzten dagegen setzen den Text in dieselbe Zeile; die Einleitung mit `\subparagraph` wird zusätzlich eingerückt. Alle Überschriften sind fett, `\section` bzw. `\subsection` zusätzlich mit `\Large` bzw. `\large` größer. Außerdem nehmen die Abstände mit zunehmender Gliederungstiefe ab. (In Wirklichkeit sind sie viel größer als im Beispiel, erkennbar am vorliegenden Dokument.) Die ersten drei Ebenen werden nummeriert, und zwar jeweils mit arabischen Ziffern. All diese Einstellungen lassen sich verändern; dazu mehr in Abschnitt 6.4.

\tableofcontents

Mit einem einzigen Befehl lässt sich ein Inhaltsverzeichnis erstellen:

`> \tableofcontents`

= **1 Naturwissenschaften** **1**

1.1 Biologie 1

1.1.1 Tiere 1

Standardmäßig werden nur die Einträge bis `\subsubsection` ins Inhaltsverzeichns übernommen. Wie man einstellen kann, bis zu welcher Stufe Einträge ins Inhaltsverzeichnis aufgenommen und bis zu welcher Stufe die Überschriften nummeriert werden, wird in Abschnitt 6.4 erklärt.

Beim Inhaltsverzeichnis wird zum ersten Mal ein spezielles Merkmal von LaTeX deutlich. Es gibt Befehle wie dieser hier zum Erstellen des Inhaltsverzeichnisses, die ein zweimaliges Compilieren des Quelltextes erfordern. Beim ersten Durchlauf werden die Inhalte und Seitennummern aller Gliederungsbefehle gesammelt und in einer Datei gespeichert, in diesem Fall TexDokument.toc. Beim nächsten Durchlauf wird diese Datei an der Stelle des Befehls `\tableofcontents` eingesetzt.

\chapter
\part

In den Dokumentenklassen report und book gibt es noch einen Befehl `\chapter`, der ein neues Kapitel anlegt. Dieses beginnt immer auf einer neuen Seite mit deutlich mehr oberem Seitenrand. Man kann einstellen, auf welchen Seiten Kapitel beginnen dürfen, vgl. dazu den Unterabschnitt 1.3.1. Ferner gibt es darüber noch `\part`, mit dem das Dokument in verschiedene Teile unterteilt werden kann. `\part` erzeugt sogar eine gesonderte Seite, auf der nur „`Teil xx`“ und der Name zentriert draufsteht. `\part` ist auch in article verfügbar, erzeugt dort aber nur eine Überschrift, anstatt eine komplette Seite anzulegen.

2.7 Fußnoten und Seitenformatierung

\footnote

Fußnoten werden einfach mit `\footnote` gesetzt. Das einzige Argument dieses Befehls sind die Daten, die in der Fußnote angezeigt werden sollen. Darin können sich beliebiger Text, mathematische Formeln sowie Tabellen befinden.

```
> \dots\ werden die Einsteinkoeffizienten\footnote{nach Albert Einstein,
  1879--1955} genannt.
```

= . . . werden die Einsteinkoeffizienten[1] genannt.

[1] nach Albert Einstein, 1879–1955

\pagenumbering

Das Format der Seitennummerierung kann mit dem Befehl `\pagenumbering` modifiziert werden. Als Argument sind möglich:

arabic roman Roman alph Alph

- `arabic`: arabische Zahlen 1, 2, 3, 4, . . . (Standard)
- `roman`: römische Zahlen mit Kleinbuchstaben i, ii, iii, iv, . . .
- `Roman`: römische Zahlen mit Großbuchstaben I, II, III, IV, . . .

- `alph`: Kleinbuchstaben a, b, c, d, ...
- `Alph`: Großbuchstaben A, B, C, D, ...

Bei jedem Aufruf von `\pagenumbering` beginnt die Zählung wieder von Neuem bei 1 (bzw. i, ...). Bei der Formatierung mit Buchstaben tritt ab Seite 27 ein Fehler auf.

\clearpage \cleardoublepage

Der Befehl `\clearpage` erzwingt einen Seitenumbruch, wobei keine komplett leeren Seiten erzeugt werden. Mehrmaliges Aufrufen von `\clearpage` hintereinander wird ignoriert. `\cleardoublepage` erzwingt bei doppelseitigem Druck einen Seitenumbruch zu einer rechten Seite; im Zweifel entsteht dann eine leere linke Seite. Dieser Befehl sollte unbedingt vor `\pagenumbering` benutzt werden, weil sonst gegebenfalls die linken Seiten des Dokuments ungerade Seitenzahlen erhalten.

\newpage \pagebreak

Der Befehl `\newpage` bewirkt im wesentlichen dasselbe wie `\clearpage`; für die Unterschiede siehe Abschnitt 8.4. Mit `\pagebreak` kann ein Seitenwechsel erzwungen werden, wenn der automatisch generierte nicht erwünscht ist. LaTeX versucht dann, den Text auf der mit `\pagebreak` beendeten Seite so zu dehnen, dass er am unteren Rand endet.

a4wide scrartcl scrreprt scrbook

Die Standardseitenränder von LaTeX sind sehr groß. Wer sie zunächst auf einfachste Weise verkleinern will, kann das Paket a4wide einbinden, auch wenn dieses nicht mehr verwendet werden soll. Die Gründe dafür sind, dass die Plattformunabhängigkeit nicht garantiert ist, dass die Kopfzeilenhöhe zu klein eingestellt wird und manche Details mehr. Die LaTeX-Community empfiehlt, stattdessen die Klassen scrartcl, scrreprt bzw. scrbook zu verwenden, aber auch über deren Seitenlayout lässt sich streiten.[11] Genaueres zu den Seitenrändern folgt in Abschnitt 6.2.

2.7.1 Kopf- und Fußzeilen

\pagestyle \thispagestyle empty plain headings myheadings

Der Befehl `\pagestyle` definiert das Seitenlayout und bestimmt das Aussehen von Kopf- und Fußzeilen. Als Argumente sind möglich:

- `empty`: leere Kopf- und Fußzeile
- `plain`: leere Kopfzeile, zentrierte Seitenzahl in der Fußzeile (Standard für letter, article und report)
- `headings`: laufendes Kapitel in der Kopfzeile der linken Seite, laufender Abschnitt in der Kopfzeile der rechten Seite, Seitennummer jeweils außen in der Kopfzeile (Standard für book)
- `myheadings`: benutzerdefinierte Einstellungen

`\pagestyle` definiert das Layout für alle folgenden Seiten. Soll das Layout einer einzelnen Seite verändert werden, so kann `\thispagestyle` mit denselben Möglichkeiten benutzt werden.

Ein Paket für besser anpassbare Seitenstile wird in Abschnitt 7.5 beschrieben. Im Folgenden beschäftigen wir uns genauer mit `headings` und `myheadings`. In beiden Fällen steht die Seitenzahl außen in der Kopfzeile, außer bei Kapitelanfangsseiten, auf denen sie zentriert in der Fußzeile steht.

Bei der Option `headings` steht bei einseitigem Druck die oberste Gliederungsebene (Kapitel bei report und book, Abschnitt bei article) links in der Kopfzeile, bei doppelseitigem Druck die oberste Gliederungsebene auf den linken Seiten innen, die nächste Ebene auf den rechten Seiten innen. Diese Überschriften werden mit den Nummern versehen, schräggestellt und in Großbuchstaben gesetzt. Bei `myheadings` steht außer der Seitennummer zunächst nichts in der Kopfzeile.

[11] Ich mag sie nicht ... ;-\

\chaptermark \sectionmark \subsectionmark \subsubsectionmark \paragraphmark \subparagraphmark \markright \markboth \leftmark \rightmark

Zu den Gliederungsbefehlen `\chapter`, `\section`, usw. gibt es die Befehle `\chaptermark`, `\sectionmark`, `\subsectionmark`, `\subsubsectionmark`, `\paragraphmark` und `\subparagraphmark`. Sie können benutzt werden, um z. B. einen verkürzten Titel in die Kopfzeile zu schreiben:

```
> \chapter{Eine ungewöhnlich lange Kapitelüberschrift}
  \chaptermark{Lange Überschrift}
```

Ungeschickterweise[12] müssen alle diese Befehle außer `\chaptermark` vor *und* nach `\section` usw. benutzt werden, sinnvollerweise beidesmal mit demselben Argument.

Auch hier werden die Nummern ergänzt und alles in Großbuchstaben umgewandelt. Wen dieser Automatismus stört oder wer `myheadings` verwenden will, der kann sich `\markright` und `\markboth` bedienen. `\markright` erhält ein Argument und legt fest, was in der Kopfzeile der rechten Seiten des Dokuments außer der Seitenzahl stehen soll. `\markboth` erhält zwei Argumente, das erste für die Kopfzeile der linken und das zweite für die der rechten Seiten. Dieser Befehl ist nur zusammen mit beidseitigem Druck sinnvoll.

Zu jeder Zeit stehen in den Befehlen `\leftmark` und `\rightmark` die momentanen Kopfzeileninhalte:

```
> \leftmark \\ \rightmark
```

= KAPITEL 2. FORMATIERUNG UND GLIEDERUNG
2.7. FUSSNOTEN UND SEITENFORMATIERUNG

[12] Ich weiß tatsächlich nicht genau, woran das liegt.

Kapitel 3

Mathematische Zeichen

Zunächst ein kleiner Vorgriff auf die mathematischen Umgebungen: Mit `$ $` wird eine mathematische Inline-Umgebung erzeugt. `\[ \]` hingegen erzeugt eine abgesetzte Formel, die in der Zeile zentriert wird. Leerzeichen im Quelltext werden im Mathematikmodus ignoriert. Gelegentlich benutzen wir schon `^` bzw. `_`, um das folgende Zeichen als Exponent bzw. Index zu setzen, genaueres dazu in Abschnitt 4.1.

3.1 Buchstaben und Ziffern

Normale lateinische Buchstaben können einfach eingegeben werden:
`> $abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ$`
= $abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ$
Auch arabische Ziffern funktionieren ganz normal:
`> $0123456789$`
= 0123456789
Insbesondere werden Ziffern nicht kursiv gesetzt. Man beachte den Unterschied zwischen kursiven Buchstaben (`\textit{Diffusion}` ergibt *Diffusion*) und Buchstaben im Mathematikmodus (`$Diffusion$` ergibt $Diffusion$).

Ein paar spezielle Buchstaben: \hbar
`> $\hbar \hslash \imath \jmath \ell$` \hslash
= $\hbar \hslash \imath \jmath \ell$ \imath
`\hbar` (alternativ `\hslash`) steht für das reduzierte Plancksche Wirkungsquantum. `\imath` und `\jmath` dienen dazu, Akzente über ein `i` und `j` zu setzen. `\ell` ist eine Form des Buchstabens l, die besser von der Zahl 1 zu unterscheiden ist. \jmath \ell

`\mathcal` erzeugt kalligraphische Großbuchstaben: \mathcal
`> $\mathcal{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$` eucal
= $\mathcal{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \CMcal
Es gibt auch noch den Befehl `\cal`, der aber veraltet ist und nicht mehr verwendet werden soll. Mit dem Paket `eucal` werden statt dessen folgende Zeichen erzeugt:
`> $\mathcal{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$`
= $\mathcal{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
Die ursprünglichen Zeichen können dann mit `\CMcal` erzeugt werden.

`\mathscr` (Paket `mathrsfs`) erzeugt Großbuchstaben in Schreibschrift: \mathscr
`> $\mathscr{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$` \mathfrak
= $\mathscr{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$

`\mathfrak` (Paket `amsfonts`) erzeugt Klein-, Großbuchstaben und Ziffern im Frakturstil:
`> $\mathfrak{abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789}$`
= $\mathfrak{abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789}$
Es gibt noch den Befehl `\frak`, der aber veraltet ist und nicht mehr verwendet werden soll.

Die Verwendung dieser kalligraphischen und Frakturbuchstaben ist uneinheitlich. Gelegentlich werden Operatoren mit `\mathcal` gesetzt, z. B. $\mathcal{R}$, $\mathcal{S}$ und $\mathcal{T}$, oder auch die Laplace- und Fourier-Transformation mit $\mathcal{L}$ und $\mathcal{F}$. Für die letzteren beiden werden manchmal auch $\mathscr{L}$ und $\mathscr{F}$ benutzt. $\mathcal{O}$ wird häufig für das Landau-Symbol benutzt, was aber nur dann sinnvoll ist, wenn nicht „klein-o" ebenfalls vorkommt. Häufig anzutreffen sind $\mathcal{C}$ und $\mathcal{L}$ für die stetigen Funktionen und die Lebesgue-Räume. Geläufig sind ebenfalls $\mathcal{B}$ für Basen, $\mathcal{A}$ und $\mathcal{F}$ für Sigma-Algebren und $\mathcal{T}$ für Topologien. Frakturbuchstaben werden eigentlich ausschließlich für Ideale benutzt, z. B. um ein Primideal $\mathfrak{p}$ von einer Primzahl p zu unterscheiden.

\mathbb \math-bbm \mathds

`\mathbb`[1] (Paket `amsfonts` oder `amssymb`) erzeugt Großbuchstaben mit Doppelstrich:
`> $\mathbb{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$`
= $\mathbb{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
`\mathbbm` (Paket `bbm`) erzeugt Klein-, Großbuchstaben sowie die 1 und 2 mit Doppelstrich:
`> $\mathbbm{abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ12}$`
= $\mathbbm{abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ12}$
`\mathds` (Paket `dsfont`) erzeugt Großbuchstaben und wenige andere Zeichen mit Doppelstrich:
`> $\mathds{ahkABCDEFGHIJKLMNOPQRSTUVWXYZ1}$`
= $\mathds{ahkABCDEFGHIJKLMNOPQRSTUVWXYZ1}$
Die `\mathbbm`-Zeichen haben den Vorteil, dass sie als einzige mit Hilfe von `\boldmath` (vgl. Abschnitt 5.3) fett gesetzt werden können.

Am häufigsten sieht man die `\mathbb`-Zeichen, obwohl gerade $\mathbb{N}$ und $\mathbb{R}$ ziemlich hässlich sind. Wenn nicht jemand Vektoren so schreibt, sind solche Buchstaben ausnahmslos (Zahlen-) Mengen, nämlich $\mathbb{N}$ (natürliche Zahlen), $\mathbb{P}$ (Primzahlen), $\mathbb{Z}$ (ganze Zahlen), $\mathbb{Q}$ (rationale Zahlen), $\mathbb{R}$ (reelle Zahlen), $\mathbb{C}$ (komplexe Zahlen), $\mathbb{H}$ (Quaternionen), $\mathbb{O}$ (Cayley-Oktaven), $\mathbb{K}$ (meist $\mathbb{R}$ oder $\mathbb{C}$) und $\mathbb{F}$ (ein (endlicher) Körper). Andere Buchstaben trifft man sehr selten an. $\mathbb{1}$ wird gelegentlich für die Einheitsmatrix, den Spaltenvektor mit lauter Einsen oder die Indikatorfunktion benutzt.

\mathrm \mathsf \mathtt \mathit \mathbf

Ähnlich zu `\textrm` usw. kann man auch Schriftfamilie, -schnitt und -gewicht im Mathematikmodus einstellen:
`> $\mathrm{ab12} \mathsf{ab12} \mathtt{ab12} \mathit{ab12} \mathbf{ab12}$`
= $\mathrm{ab12}\mathsf{ab12}\mathtt{ab12}\mathit{ab12}\mathbf{ab12}$
Auch hier ist die Verwendung von `\rm`, `\sf`, `\tt`, `\it` und `\bf` veraltet, vgl. Abschnitt 2.1.

Für diejenigen, die gerne die Eulersche Zahl e, die imaginäre Einheit i oder das d in Differentialen aufrecht schreiben wollen, ist `\mathrm` die richtige Wahl. Wer ganz genau hinschaut, sieht einen Unterschied zwischen U_{min} (`U_{min}`) und U_{min} (`U_{\mathit{min}}`).

\alpha ... \xi \pi ... \omega \varepsilon \vartheta \varpi \varrho \varsigma \varphi

Die griechischen Kleinbuchstaben sind:
```
> $\alpha \beta \gamma \delta \epsilon \zeta \eta \theta \iota \kappa \lambda
  \mu \nu \xi \pi \rho \sigma \tau \upsilon \phi \chi \psi \omega$
```
= $\alpha\beta\gamma\delta\epsilon\zeta\eta\theta\iota\kappa\lambda\mu\nu\xi\pi\rho\sigma\tau\upsilon\phi\chi\psi\omega$
Ein Omikron gibt es nicht. Es gibt einige modifizierte griechische Kleinbuchstaben:
`> $\varepsilon \vartheta \varpi \varrho \varsigma \varphi$`
= $\varepsilon\vartheta\varpi\varrho\varsigma\varphi$

[1] `bb` steht für „blackboard bold".

Die Variante ε sollte man immer statt ϵ verwenden. ς gibt es in der Mathematik nicht, und ϖ ist immer die lemniskatische Konstante.

Die griechischen Großbuchstaben sind:

> $\Gamma \Delta \Theta \Lambda \Xi \Pi \Sigma \Upsilon \Phi \Psi \Omega$

= $\Gamma\Delta\Theta\Lambda\Xi\Pi\Sigma\Upsilon\Phi\Psi\Omega$

Diejenigen griechischen Großbuchstaben, die genauso wie die lateinischen aussehen, gibt es nicht. In kursiver Form (Paket `amsmath`) lauten sie:

> $\varGamma \varDelta \varTheta \varLambda \varXi \varPi \varSigma
  \varUpsilon \varPhi \varPsi \varOmega$

= $\varGamma\varDelta\varTheta\varLambda\varXi\varPi\varSigma\varUpsilon\varPhi\varPsi\varOmega$

Hier empfehle ich die kursiven Varianten, weil es einheitlicher aussieht.

Üblicherweise wird `\Delta` auch für Differenzen benutzt, z. B. $m = \Delta y/\Delta x$ in Steigungsdreiecken. Auch den Laplace-Operator schreibt man als Δf. In beiden Fällen sähe `\varDelta` komisch aus.

\Gamma \Delta \Theta \Lambda \Xi \Pi \Sigma \Upsilon \Phi \Omega \varGamma \varDelta \varTheta \varLambda \varXi \varPi \varSigma \varUpsilon \varPhi \varOmega

3.2 Abstände und Interpunktionen

Im Mathematikmodus werden Leerzeichen grundsätzlich ignoriert; deshalb kann man sie dazu verwenden, den Formelquelltext übersichtlicher zu gestalten. Man sollte diese Möglichkeit auch unbedingt nutzen, weil die Formeln im Quelltext sonst völlig unlesbar werden. Statt dessen werden die Abstände von LaTeX selbst festgelegt, und um dies zu realisieren, sind die Zeichen in verschiedene Klassen eingeteilt; dazu später mehr.

Der Punkt `.` ist z. B. für Dezimalzahlen in amerikanischer Schreibweise vorgesehen und erhält keinen Zusatzabstand:

> 3.14159

= 3.14159

.

Beim Komma `,` wird rechts ein kleiner Abstand eingefügt:

> a,b,x,y

= a,b,x,y

Möchte man das Komma für Dezimalzahlen verwenden, so sollte man es mit `{` und `}` klammern:

> $3,14159 \quad 3{,}14159$

= $3,14159 \quad 3{,}14159$

, {,}

Auch das Semikolon `;` erhält rechts einen kleinen Abstand:

> $a,b;x,y$

= $a,b;x,y$

;

Für zuätzlichen Leerraum gibt es folgende Abstandsbefehle:

> $a\,b$	= $a\,b$
> $a\:b$	= $a\:b$
> $a\;b$	= $a\;b$
> $a\quad b$	= $a\quad b$
> $a\qquad b$	= $a\qquad b$
> $a\!b$	= $a\!b$

\, \: \; \quad \qquad \!

Diese Befehle sollte man erst hinzufügen, nachdem man das Ergebnis ohne manuelle Abstände angeschaut hat. Und häufig gibt es sauberere Alternativen zu diesen Befehlen.

Schlecht bis gar nicht lesbar sind Intervalle $[1{,}5,2{,}5]$ mit Quelltext `[1{,}5,2{,}5]`. Hier sollte man entweder das Dezimalkomma durch einen Punkt ersetzen (was in Deutschland eigentlich nicht üblich ist), ein Semikolon statt des Kommas in der Mitte setzen oder den Abstand nach dem Komma in der Mitte vergrößern (etwa durch `\,`). Die Variante mit dem Semikolon, die manche Autoren ohnehin benutzen, löst das analoge Problem nicht, das bei Tupeln auftritt, in denen ein Semikolon sehr fragwürdig wäre.

3.3 Verknüpfende Symbole

+ - \pm \mp \oplus \ominus

Verknüpfungssymbole für Addition und Subtraktion:

`> $x+y \quad x-y \quad x\pm y \quad x\mp y \quad U\oplus V \quad x\ominus y$`

= $x+y \quad x-y \quad x\pm y \quad x\mp y \quad U\oplus V \quad x\ominus y$

$+$ und $-$ sind Addition und Subtraktion, $\pm$ und $\mp$ die Kombination aus beiden. $\oplus$ und $\ominus$ werden gelegentlich für alternative Verknüpfungen benutzt, ersteres immer für die direkte Summe. Mit $\mathbb{R}^+$ und $\mathbb{R}^-$ werden manchmal positive und negative reelle Zahlen bezeichnet.

\cdot / : \bullet * \star \times \div \odot \otimes \oslash \circ

Verknüpfungssymbole für Multiplikation und Division:

`> $x\cdot y \quad x/y \quad x:y \quad x\bullet y \quad f*g \quad x\star y$`

= $x\cdot y \quad x/y \quad x:y \quad x\bullet y \quad f*g \quad x\star y$

`> $x\times y \quad x\div y \quad x\odot y \quad U\otimes V \quad x\oslash y$`

= $x\times y \quad x\div y \quad x\odot y \quad U\otimes V \quad x\oslash y$

`> $f\circ g$`

= $f\circ g$

Der Befehl `\ast` wird nicht benötigt.

$\cdot$ und $/$ sind Multiplikation und Division, ersteres auch das Euklidische Skalarprodukt. Als G/N schreibt man auch Faktorgruppen usw. und Körpererweiterungen L/K. $:$ ist üblich für den Index $[G:N]$ und den Erweiterungsgrad $[L:K]$, eigentlich nie als Division und bitte auch nicht für Funktionsdefinitionen – dazu siehe `\colon` auf Seite 29. $\bullet$, $*$ und $\star$ können alternative Produkte sein, ersteres z. B. auch ein Skalarprodukt, zweiteres häufig ein Faltungsprodukt. Mit A^* wird meistens die adjungierte Abbildung bzw. adjungierte Matrix bezeichnet oder mit V^* der Dualraum. $\times$ ist das kartesische Produkt von Mengen oder das Vektor-/Kreuzprodukt zweier Vektoren, selten eher unmathematisch benutzt als normales Produkt, hochgestellt in $R^\times$ die Einheitengruppe eines Rings. $\odot$, $\otimes$ und $\oslash$ können alternative Multiplikationen/Divisionen sein, zweiteres das Tensorprodukt oder das Produkt von Sigma-Algebren und Maßen, hochgestellt $V^{\otimes n}$ das n-fache Tensorprodukt von V mit sich selbst. $\circ$ ist fast ausschließlich die Verkettung von Abbildungen, alternativ eine besondere Multiplikation, hochgestellt das Gradmaß 30° eines Winkels oder das Innere A° einer Menge.

Weitere solche Symbole (Paket amssymb):

`> $x\boxplus y \quad x\boxminus y \quad x\boxdot y \quad x\boxtimes y$`

= $x\boxplus y \quad x\boxminus y \quad x\boxdot y \quad x\boxtimes y$

\cup \cap \sqcup \sqcap \uplus \setminus \vee \wedge \lor \land

Mengenverknüpfungen und logische Verknüpfungen:

`> $A\cup B \quad A\cap B \quad A\sqcup B \quad A\sqcap B \quad A\uplus B$`

= $A\cup B \quad A\cap B \quad A\sqcup B \quad A\sqcap B \quad A\uplus B$

`> $A\setminus B \quad A\vee B \quad A\wedge B \quad A\lor B \quad A\land B$`

= $A\setminus B \quad A\vee B \quad A\wedge B \quad A\vee B \quad A\wedge B$

`\lor`[2] bzw. `\land`[3] stimmen mit `\vee` bzw. `\wedge` überein.

[2] `lor` steht für „logical or“.
[3] `land` steht für „logical and“.

$\cup$ und $\cap$ stehen für Vereinigung und Durchschnitt von Mengen, $\sqcup$ und $\sqcap$ sind Alternativen davon, $\uplus$ wird gelegentlich für die disjunkte Vereinigung benutzt. $\setminus$ ist die Differenz von Mengen. $\vee$ und $\wedge$ werden in der Stochastik oft für Maximum und Minimum zweier Funktionen benutzt, zweiteres auch immer für das Dach-/Keilprodukt. In Booleschen Ausdrücken sind $\vee$ und $\wedge$ das logische Oder und Und; hierfür sollte man aber dann `\lor` und `\land` benutzen.

Weitere solche Symbole (Paket amssymb):

`> $A\Cup B \quad A\Cap B \quad A\veebar B \quad A\barwedge B$`
= $A\Cup B \quad A\Cap B \quad A\veebar B \quad A\barwedge B$
`> $A\doublebarwedge B \quad A\curlyvee B \quad A\curlywedge B$`
= $A\doublebarwedge B \quad A\curlyvee B \quad A\curlywedge B$

Gruppenverknüpfungen (die letzen beiden im Paket amssymb): \wr
`> $G\wr J \quad N\ltimes G \quad G\rtimes N$` \ltimes
= $G\wr J \quad N\ltimes G \quad G\rtimes N$ \rtimes
$\wr$ bezeichnet das Kranzprodukt, $\ltimes$ und $\rtimes$ das semidirekte Produkt von Gruppen.

Alle Verknüpfungsbefehle erzeugen links und rechts geeigneten vordefinierten Abstand. Fehlt der linke Operand, z. B. weil „$-$“ als Vorzeichen und nicht als Rechenzeichen benutzt werden soll, dann wird kein Abstand zum rechten Operanden erzeugt:

`> $-a \quad b-a$`
= $-a \quad b-a$
Wir werden später sehen, dass LaTeX dies in bestimmten Situationen falsch interpretiert, was zu einem falschen Leerraum nach einem negativen Vorzeichen führt.

3.4 Vergleichende Symbole

Gleichheits-, Kleiner- und Größerzeichen: =
`> $x=y \quad x<y \quad x>y \quad x\leq y \quad x\geq y$` <
= $x=y \quad x<y \quad x>y \quad x\leq y \quad x\geq y$ >
`> $x\le y \quad x\ge y \quad x\ll y \quad x\gg y$` \leq
= $x\le y \quad x\ge y \quad x\ll y \quad x\gg y$ \geq
`\le` und `\ge` sind Abkürzungen für `\leq` und `\geq`. \le

$\leq$ wird gelegentlich für Untergruppe bzw. Untervektorraum benutzt, $<$ wenn sie bzw. er echt ist. $\ll$ und $\gg$ stehen für „sehr viel kleiner“ und „sehr viel größer“. \ge \ll \gg

Weitere solche Symbole (Paket amssymb):

`> $x\leqslant y \quad x\geqslant y \quad x\leqq y \quad y\geqq y$`
= $x\leqslant y \quad x\geqslant y \quad x\leqq y \quad y\geqq y$
`> $x\lneq y \quad x\gneq y \quad x\lneqq y \quad x\gneqq y$`
= $x\lneq y \quad x\gneq y \quad x\lneqq y \quad x\gneqq y$
`> $x\lesssim y \quad x\gtrsim y \quad x\lessapprox y \quad x\gtrapprox y$`
= $x\lesssim y \quad x\gtrsim y \quad x\lessapprox y \quad x\gtrapprox y$
`> $x\lnsim y \quad x\gnsim y \quad x\lnapprox y \quad x\gnapprox y$`
= $x\lnsim y \quad x\gnsim y \quad x\lnapprox y \quad x\gnapprox y$
`> $x\lessgtr y \quad x\gtrless y \quad x\lesseqgtr y \quad x\gtreqless y$`
= $x\lessgtr y \quad x\gtrless y \quad x\lesseqgtr y \quad x\gtreqless y$
`> $x\lesseqqgtr y \quad x\gtreqqless y \quad x\lll y \quad x\ggg y$`
= $x\lesseqqgtr y \quad x\gtreqqless y \quad x\lll y \quad x\ggg y$

`\prec` `\succ` `\preceq` `\succeq`

Geschwungene Kleiner- und Größerzeichen:
`> $x\prec y \quad x\succ y \quad x\preceq y \quad x\succeq y$`
= $x\prec y \quad x\succ y \quad x\preceq y \quad x\succeq y$
Diese Zeichen sind bei Ordnungsrelationen, Nachfolgerrelationen und gerichteten Mengen gebräuchlich.

Weitere solche Symbole (Paket amssymb):
`> $x\preccurlyeq y \quad x\succcurlyeq y \quad x\precsim y \quad x\succsim y$`
= $x\preccurlyeq y \quad x\succcurlyeq y \quad x\precsim y \quad x\succsim y$
`> $x\precapprox y \quad x\succapprox y \quad x\precnsim y \quad x\succnsim y$`
= $x\precapprox y \quad x\succapprox y \quad x\precnsim y \quad x\succnsim y$
`> $x\precnapprox y \quad x\succnapprox y$`
= $x\precnapprox y \quad x\succnapprox y$

`\sim` `\simeq` `\approx` `\cong` `\equiv` `\propto` `\asymp` `\perp` `\parallel` `\mid`

Ähnlich, ungefähr, usw.:
`> $x\sim y \quad x\simeq y \quad x\approx y \quad x\cong y \quad x\equiv y$`
= $x\sim y \quad x\simeq y \quad x\approx y \quad x\cong y \quad x\equiv y$
`> $x\propto y \quad x\asymp y \quad g\perp h \quad g\parallel h \quad a\mid b$`
= $x\propto y \quad x\asymp y \quad g\perp h \quad g\parallel h \quad a\mid b$

$\sim$ kann ähnlich, äquivalent oder proportional sein, $\simeq$ wird für isomorph und homöomorph verwendet. $\approx$ bedeutet sehr häufig ungefähr oder auch diffeomorph, $\cong$ ist das gewöhnliche Zeichen für isomorph und steht auch für kongruent in der Geometrie. $\equiv$ heißt einerseits kongruent (äquivalent bezüglich modulo) oder identisch gleich. $\propto$ ist proportional, $\asymp$ asymptotisch, $\perp$ orthogonal, $\parallel$ parallel und $\mid$ teilt. $v_\parallel$ steht für einen parallelen Anteil des Vektors v.

`\subset` `\supset` `\subseteq` `\supseteq` `\sqsubset` `\sqsupset` `\sqsubseteq` `\sqsupseteq` `\in` `\ni` `\owns`

Mengenzeichen:
`> $A\subset B \quad A\supset B \quad A\subseteq B \quad A\supseteq B$`
= $A\subset B \quad A\supset B \quad A\subseteq B \quad A\supseteq B$
`> $A\sqsubset B \quad A\sqsupset B \quad A\sqsubseteq B \quad A\sqsupseteq B$`
= $A\sqsubset B \quad A\sqsupset B \quad A\sqsubseteq B \quad A\sqsupseteq B$
`> $x\in A \quad A\ni x \quad A\owns x$`
= $x\in A \quad A\ni x \quad A\ni x$
`\owns` stimmt mit `\ni` überein.

$\subseteq$ und $\supseteq$ sind Teilmenge und Obermenge mit explizit zugelassener Gleichheit, $\subset$ und $\supset$ verbieten manchmal die Gleichheit, manchmal nicht. Die eckigen $\sqsubset$, $\sqsupset$, $\sqsubseteq$ und $\sqsupseteq$ sind wieder Alternativen. $\in$ und $\ni$ stehen für Elemente aus Mengen.

Weitere solche Symbole (Paket amssymb):
`> $A\subseteqq B \quad A\supseteqq B \quad A\subsetneq B \quad A\supsetneq B$`
= $A\subseteqq B \quad A\supseteqq B \quad A\subsetneq B \quad A\supsetneq B$
`> $A\subsetneqq B \quad A\supsetneqq B \quad A\Subset B \quad A\Supset B$`
= $A\subsetneqq B \quad A\supsetneqq B \quad A\Subset B \quad A\Supset B$
`> $N\lhd G \quad G\rhd N \quad N\unlhd G \quad G\unrhd N$`
= $N\lhd G \quad G\rhd N \quad N\unlhd G \quad G\unrhd N$

`\not` `\neq` `\nparallel` `\nmid` `\notin` `\ne`

Alle Relationszeichen können mit dem Befehl `\not` durchgestrichen werden, z. B.:
`> $a\not>b \quad g\not\perp h \quad A\not\subseteq B$`
= $a\not>b \quad g\not\perp h \quad A\not\subseteq B$
Ausnahmen sind:
`> $x\neq y \quad g\nparallel h \quad a\nmid b \quad x\notin A$`
= $x\neq y \quad g\nparallel h \quad a\nmid b \quad x\notin A$

Statt `\not=` verwendet man meist das gleichaussehende `\neq` (abgekürzt `\ne`). `\nparallel`, `\nmid` (beide im Paket amssymb) und `\notin` sind den Konstruktionen mit `\not` unbedingt vorzuziehen.

Andere durchgestrichene Symbole (Paket amssymb):

```
> $x\nless y \quad x\ngtr y \quad x\nleq y \quad x\ngeq y$
```
= $x\nless y \quad x\ngtr y \quad x\nleq y \quad x\ngeq y$
```
> $x\nleqslant y \quad x\ngeqslant y \quad x\nprec y \quad x\nsucc y$
```
= $x\nleqslant y \quad x\ngeqslant y \quad x\nprec y \quad x\nsucc y$
```
> $Ax\npreceq y \quad x\nsucceq y \quad \nsubseteq B \quad A\nsupseteq B$
```
= $x\npreceq y \quad x\nsucceq y \quad A\nsubseteq B \quad A\nsupseteq B$
```
> $x\nsim y \quad x\ncong y \quad A\nsubseteqq B \quad A\nsupseteqq B$
```
= $x\nsim y \quad x\ncong y \quad A\nsubseteqq B \quad A\nsupseteqq B$

3.5 Akzente

Die folgenden Befehle setzen einen bestimmten Akzent über das folgende Zeichen: \hat \check \bar \vec \tilde \dot \ddot \dddot \ddddot \breve \acute \grave \mathring

```
> $\hat a\hat Y \quad \check a\check Y \quad \bar a\bar Y \quad \vec a\vec Y$
```
= $\hat a\hat Y \quad \check a\check Y \quad \bar a\bar Y \quad \vec a\vec Y$
```
> $\tilde a\tilde Y \quad \dot a\dot Y \quad \ddot a\ddot Y$
```
= $\tilde a\tilde Y \quad \dot a\dot Y \quad \ddot a\ddot Y$
```
> $\dddot a\dddot Y \quad \ddddot a\ddddot Y \quad \breve a\breve Y$
```
= $\dddot a\dddot Y \quad \ddddot a\ddddot Y \quad \breve a\breve Y$
```
> $\acute a\acute Y \quad \grave a\grave Y \quad \mathring a\mathring Y$
```
= $\acute a\acute Y \quad \grave a\grave Y \quad \mathring a\mathring Y$

Die Befehle `\dddot` und `\ddddot` sind im Paket amsmath. Das eingebundene Paket mathdots gibt dabei eine Compilerwarnung aus, weil es diese beiden Akzente verändert.

Im Paket amsxtra befinden sich folgende Befehle: \sphat \spcheck \sptilde \spbreve \spdot \spddot \spdddot

```
> $(ab)\sphat \quad (ab)\spcheck \quad (ab)\sptilde \quad (ab)\spbreve$
```
= $(ab)^{\wedge} \quad (ab)^{\vee} \quad (ab)^{\sim} \quad (ab)^{\smallsmile}$
```
> $(ab)\spdot \quad (ab)\spddot \quad (ab)\spdddot$
```
= $(ab)^{\cdot} \quad (ab)^{\cdot\cdot} \quad (ab)^{\cdot\cdot\cdot}$

Zwei Akzente kann man auch über mehrere Zeichen erstrecken: \widehat \widetilde

```
> \[ \widehat a \quad \widehat{AB} \quad \widehat{vwxyz} \]
```
=
$$\widehat a \quad \widehat{AB} \quad \widehat{vwxyz}$$
```
> \[ \widetilde a \quad \widetilde{AB} \quad \widetilde{vwxyz} \]
```
=
$$\widetilde a \quad \widetilde{AB} \quad \widetilde{vwxyz}$$

Ohne das amssymb-Paket lassen sich `\widehat` und `\widetilde` nicht ganz so weit dehnen.

Das Paket undertilde stellt einen Befehl zur Verfügung, mit dem man eine (lange) Tilde unter Buchstaben erzeugen kann: \utilde

```
> \[ \utilde a \quad \utilde{AB} \quad \utilde{vwxyz} \]
```
=
$$\utilde a \quad \utilde{AB} \quad \utilde{vwxyz}$$

Man kann auch zwei Akzente auf einen Buchstaben setzen:

```
> \[ \dot{\vec v} \quad \hat{\hat f} \quad \tilde{\bar z} \]
```
=
$$\dot{\vec v} \quad \hat{\hat f} \quad \tilde{\bar z}$$

Für weitere Pfeile und Striche über oder unter Zeichen siehe Abschnitt 4.1.

3.6 Pfeile

\leftarrow \rightarrow \leftrightarrow … \gets \to

Einfache waagerechte Pfeile:

```
> $x\leftarrow y \quad x\rightarrow y \quad x\leftrightarrow y$
```
= $x\leftarrow y \quad x\rightarrow y \quad x\leftrightarrow y$
```
> $x\longleftarrow y \quad x\longrightarrow y \quad x\longleftrightarrow y$
```
= $x\longleftarrow y \quad x\longrightarrow y \quad x\longleftrightarrow y$
```
> $x\gets y \quad x\to y$
```
= $x\gets y \quad x\to y$

`\gets` und `\to` sind Abkürzungen für `\leftarrow` und `\rightarrow`. Als Konvergenz- und Abbildungspfeil reichen $\leftarrow$ und $\rightarrow$; die langen Varianten sind selten notwendig.

\Leftarrow \Rightarrow \Leftrightarrow … \iff

Doppelte waagerechte Pfeile:

```
> $A\Leftarrow B \quad A\Rightarrow B \quad A\Leftrightarrow B$
```
= $A\Leftarrow B \quad A\Rightarrow B \quad A\Leftrightarrow B$
```
> $A\Longleftarrow B \quad A\Longrightarrow B \quad A\Longleftrightarrow B$
```
= $A\Longleftarrow B \quad A\Longrightarrow B \quad A\Longleftrightarrow B$
```
> $A\iff B$
```
= $A\iff B$

`\iff`[4] ist eine Abkürzung für `\Longleftrightarrow`, allerdings mit etwas elastischem Abstand links und rechts vom Zeichen. Als Folge- und Äquivalenzpfeil bevorzuge ich die langen Varianten.

\uparrow \downarrow … \Updownarrow

Senkrechte Pfeile:

```
> $x\uparrow y \quad x\downarrow y \quad x\updownarrow y$
```
= $x\uparrow y \quad x\downarrow y \quad x\updownarrow y$
```
> $A\Uparrow B \quad A\Downarrow B \quad A\Updownarrow B$
```
= $A\Uparrow B \quad A\Downarrow B \quad A\Updownarrow B$

\nearrow \searrow \nwarrow \swarrow

Diagonale Pfeile:

```
> $x\nearrow y \quad x\searrow y \quad x\nwarrow y \quad x\swarrow y$
```
= $x\nearrow y \quad x\searrow y \quad x\nwarrow y \quad x\swarrow y$

\mapsto \longmapsto \hookleftarrow \hookrightarrow \leftharpoonup … \rightleftharpoons

Weitere wichtige Pfeile:

```
> $x\mapsto y \quad x\longmapsto y \quad A\hookleftarrow B$
```
= $x\mapsto y \quad x\longmapsto y \quad A\hookleftarrow B$
```
> $A\hookrightarrow B \quad x\leftharpoonup y \quad x\leftharpoondown y$
```
= $A\hookrightarrow B \quad x\leftharpoonup y \quad x\leftharpoondown y$
```
> $x\rightharpoonup y \quad x\rightharpoondown y \quad x\rightleftharpoons y$
```
= $x\rightharpoonup y \quad x\rightharpoondown y \quad x\rightleftharpoons y$

`\rightleftharpoons` wird durch das Paket `amssymb` etwas verändert.

$\mapsto$ ist der Abbildungspfeil für Elemente, die lange Version braucht man selten, ebenso beim Einbettungspfeil $\hookrightarrow$. Die halben Pfeile $\rightharpoonup$ usw. bedeuten meistens schwache Konvergenz, die schwach* Konvergenz erhält man durch $\rightharpoonup^*$ (`\rightharpoonup^*`).

\xleftarrow \xrightarrow

Dehnbare Pfeile (Paket `amsmath`):

```
> \[ x \xleftarrow[a\to0]{n\to1} y \quad x \xrightarrow[a\to0]{n\to1} y \]
```
=
$$x \xleftarrow[a\to0]{n\to1} y \quad x \xrightarrow[a\to0]{n\to1} y$$

[4] `iff` steht für „if and only if“.

Weitere solche Symbole (Paket amssymb):

`> $x\leftleftarrows y \quad x\rightrightarrows y \quad x\upuparrows y$`

= $x\leftleftarrows y \quad x\rightrightarrows y \quad x\upuparrows y$

`> $x\downdownarrows y \quad x\leftrightarrows y \quad x\rightleftarrows y$`

= $x\downdownarrows y \quad x\leftrightarrows y \quad x\rightleftarrows y$

`> $A\Lleftarrow B \quad A\Rrightarrow B \quad x\leftrightharpoons y$`

= $A\Lleftarrow B \quad A\Rrightarrow B \quad x\leftrightharpoons y$

`> $x\upharpoonleft y \quad x\upharpoonright y \quad x\downharpoonleft y$`

= $x\upharpoonleft y \quad x\upharpoonright y \quad x\downharpoonleft y$

`> $x\downharpoonright y \quad x\curvearrowleft y \quad x\curvearrowright y$`

= $x\downharpoonright y \quad x\curvearrowleft y \quad x\curvearrowright y$

`> $x\dashleftarrow y \quad x\dashrightarrow y \quad x\twoheadleftarrow y$`

= $x\dashleftarrow y \quad x\dashrightarrow y \quad x\twoheadleftarrow y$

`> $x\twoheadrightarrow y \quad x\leftarrowtail y \quad x\rightarrowtail y$`

= $x\twoheadrightarrow y \quad x\leftarrowtail y \quad x\rightarrowtail y$

Weitere solche Symbole (Paket stmaryrd):

`> $x\mapsfrom y \quad x\longmapsfrom y \quad x\Mapsfrom y \quad x\Mapsto y$`

= $x\mapsfrom y \quad x\longmapsfrom y \quad x\Mapsfrom y \quad x\Mapsto y$

`> $x\Longmapsfrom y \quad x\Longmapsto y$`

= $x\Longmapsfrom y \quad x\Longmapsto y$

3.7 Sonstige Symbole

Symbole, die nach rechts wirken: \forall

`> $\forall\,x\in M \quad \exists\,x\in M \quad \neg A \quad \lnot A$` \exists

= $\forall\,x\in M \quad \exists\,x\in M \quad \neg A \quad \lnot A$ \neg

`> $\Re(z) \quad \Im(z) \quad \partial f/\partial x \quad \nabla f(x,y,z)$` \lnot

= $\Re(z) \quad \Im(z) \quad \partial f/\partial x \quad \nabla f(x,y,z)$ \Re \Im

`> $\#M \quad \angle(u,v) \quad \wp(z)$` \partial

= $\#M \quad \angle(u,v) \quad \wp(z)$ \nabla

`\lnot`[5] stimmt mit `\neg` überein. \#

Weitere solche Symbole (Paket amssymb): \angle

`> $\nexists\,x\in M \quad \complement A \quad \sphericalangle(u,v)$` \wp

= $\nexists\,x\in M \quad \complement A \quad \sphericalangle(u,v)$

$\forall$, $\exists$ und $\nexists$ sind die Quantoren, $\neg$ eine Negation und $\complement$ eine Möglichkeit fürs Komplement. $\Re$ und $\Im$ sind eine Möglichkeit für Real- und Imaginärteil, ∂ steht für partielle Ableitungen oder den Rand ∂A einer Menge A, ∇ für den Nabla-Operator bzw. den Gradienten. $\#$ ist die Mächtigkeit einer Menge, $\angle$ und $\sphericalangle$ ein Winkel und $\wp$ die Weierstraßsche p-Funktion.

Symbole, die nach links wirken: !

`> $n! \quad f'(x) \quad f'''(x) \quad f\colon A\to B$` '

= $n! \quad f'(x) \quad f'''(x) \quad f\colon A\to B$ \colon

Der Befehl `\prime` wird nicht benötigt; statt dessen kann man einfach `'` verwenden. Mehr als drei Striche (f''') sollte man vermeiden, und insbesondere werden sie nicht mit `^` hochgestellt. Zur Definition von Funktionen u. ä. sollte man `\colon` statt `:` verwenden.

[5] `lnot` steht für „logical not“.

\dagger \ddagger \bot \top

Symbole zum Hoch- und Tiefstellen:

```
> $A^\dagger \quad A^\ddagger \quad A^\bot \quad v_\bot \quad A^\top$
```

= $A^\dagger \quad A^\ddagger \quad A^\bot \quad v_\bot \quad A^\top$

\infty \empty-set \aleph \amalg \mho

Alleinstehendes:

```
> $\infty \quad \emptyset \quad \aleph \quad \amalg \quad \mho$
```

= $\infty \quad \emptyset \quad \aleph \quad \amalg \quad \mho$

Weitere solche Symbole (Paket stmaryrd):

```
> $\lightning$
```

= ↯

\sharp \flat \natural \clubsuit \spade-suit \heartsuit \dia-mondsuit

Nicht-mathematische Zeichen:

```
> $\sharp \quad \flat \quad \natural$
```

= $\sharp \quad \flat \quad \natural$

```
> $\clubsuit \quad \spadesuit \quad \heartsuit \quad \diamondsuit$
```

= $\clubsuit \quad \spadesuit \quad \heartsuit \quad \diamondsuit$

Kapitel 4

Mathematische Konstrukte

4.1 Hoch, tief, drüber und drunter

Exponenten und Indizes: ^

```
> $x^3 \quad a_4 \quad x^{m+3/2} \quad a_{2n-k} \quad x_{12}^{-2}$
```
_

= $x^3 \quad a_4 \quad x^{m+3/2} \quad a_{2n-k} \quad x_{12}^{-2}$

```
> $x_{a_i+a_k}^{n^2} \quad \varepsilon_\mu {}^{\nu\varrho} {}_\sigma$
```

= $x_{a_i+a_k}^{n^2} \quad \varepsilon_\mu {}^{\nu\varrho} {}_\sigma$

```
> \[ f|_{\mathbbm Q} \quad f(1) = f(x)\Bigr|_{x=1} \quad F(x)\Bigr|_a^b \]
```

=
$$f|_{\mathbb{Q}} \quad f(1) = f(x)\Bigr|_{x=1} \quad F(x)\Bigr|_a^b$$

(Für `\Bigr` siehe Abschnitt 4.4.)

Man kann auch Exponenten und Indizes von links an Variablen setzen:

```
> $_ZB_Y \cdot _YA_X = _ZC_X \quad {_ZB_Y} \cdot {_YA_X} = {_ZC_X}$
```

= $_ZB_Y \cdot _YA_X = _ZC_X \quad {_ZB_Y} \cdot {_YA_X} = {_ZC_X}$

Man beachte jedoch, wie die geschweiften Klammern im rechten Beispiel die Abstände beeinflussen. Im linken Beispiel wird nämlich das zweite `_Y` als Index von `\cdot` interpretiert. Um an größere Objekte etwas von links dranschreiben zu können, kann man sich `\vphantom` (siehe Abschnitt 7.1) bedienen.

Drüber- und Drunterschreiben (Paket `amsmath`): \overset

```
> \[ a \overset{4.1}{=} b \quad a^2 \underset{0<a<b}{<} b^2 \]
```
\underset

=
$$a \overset{4.1}{=} b \quad a^2 \underset{0<a<b}{<} b^2$$

```
> \[ a < b \; \overset{(*)}{\underset{x<0}{\Longrightarrow}} \; ax > bx \]
```

=
$$a < b \; \overset{(*)}{\underset{x<0}{\Longrightarrow}} \; ax > bx$$

Es gibt auch noch den Befehl `\stackrel`, der aber veraltet ist und nicht mehr verwendet werden soll.

Über- und Unterstreichen: \overline

```
> $\overline{A\cup B} \quad \underline{a+b} \quad \overline{\underline{abc}}$
```
\underline

= $\overline{A\cup B} \quad \underline{a+b} \quad \overline{\underline{abc}}$

Geschweifte Klammern drüber und drunter: \overbrace

```
> \[ \overbrace{x^2 - (x+1) (x-1)}^{=1} \]
```
\underbrace

=
$$\overbrace{x^2 - (x+1) (x-1)}^{=1}$$

```
> \[ \underbrace{x^2 + 1}_{\geq1 \; \forall\,x} \]
```
=
$$\underbrace{x^2 + 1}_{\geq1 \; \forall\,x}$$

\overleftarrow \underleftarrow \overrightarrow \underrightarrow \overleftrightarrow \underleftrightarrow

Pfeile drüber und drunter:
```
> $\overleftarrow{ABC} \quad \underleftarrow{ABC}$
```
= $\overleftarrow{ABC} \quad \underleftarrow{ABC}$
```
> $\overrightarrow{ABC} \quad \underrightarrow{ABC}$
```
= $\overrightarrow{ABC} \quad \underrightarrow{ABC}$
```
> $\overleftrightarrow{ABC} \quad \underleftrightarrow{ABC}$
```
= $\overleftrightarrow{ABC} \quad \underleftrightarrow{ABC}$

4.2 Brüche und Wurzeln

\frac

„Normale“ Brüche:
```
> $\frac{3x}{7a} \quad \frac\pi2 \quad x^{\frac43}$
```
= $\frac{3x}{7a} \quad \frac\pi2 \quad x^{\frac43}$
```
> \[ \frac{3x}{7a} \quad x^{\frac43} \quad \frac{x + \frac1x}{\frac1x - x} \]
```
=
$$\frac{3x}{7a} \quad x^{\frac43} \quad \frac{x + \frac1x}{\frac1x - x}$$

Im Inline-Modus sollten Brüche nur mit Vorsicht benutzt werden, da sie sehr klein werden bzw. den Zeilenabstand vergrößern können. Im Zweifel ist die Schreibweise `$\pi/2$` oder eine abgesetzte Formel vorzuziehen. Zähler und Nenner eines Bruchs in einer abgesetzten Formel werden wie im Inline-Modus gesetzt, weswegen Doppelbrüche kleinere Zähler und Nenner bekommen.

\cfrac

Für Kettenbrüche[1] gibt es ein eigenes Kommando (Paket amsmath):
```
> \[ \cfrac1{1 + \cfrac3{1 + 4}} \]
```
=
$$\cfrac1{1 + \cfrac3{1 + 4}}$$

Ein optionales `l` oder `r` setzt den Zähler links- oder rechtsbündig:
```
> \[ \cfrac[r]2{1 + \cfrac[l]5{1 + 7}} \]
```
=
$$\cfrac[r]2{1 + \cfrac[l]5{1 + 7}}$$

\dfrac \tfrac

Große[2] und kleine[3] Brüche (Paket amsmath):
```
> \[ \frac{x + \dfrac1x}{\dfrac1x - x} \quad \tfrac{3x}{7a} \]
```
=
$$\frac{x + \dfrac1x}{\dfrac1x - x} \quad \tfrac{3x}{7a}$$

[1] `cfrac` steht für „continued fraction“.
[2] `dfrac` steht für „displaystyle fraction“.
[3] `tfrac` steht für „textstyle fraction“.

Binomialkoeffizienten (Paket amsmath): \binom

```
> $\binom nk \quad \binom{2n+1}{2k-1}$
```

= $\binom nk \quad \binom{2n+1}{2k-1}$

```
> \[ \binom nk \quad \binom{2n+1}{2k-1} \]
```

= $$\binom nk \quad \binom{2n+1}{2k-1}$$

Es gibt auch noch den Befehl `\choose` mit anderer Syntax, der aber veraltet ist und nicht mehr verwendet werden soll.

Ebenso gibt es hier (Paket amsmath): \dbinom \tbinom

```
> \[ \frac{\dbinom42 \dbinom{28}8}{\dbinom{32}{10}} \quad \tbinom nk \]
```

= $$\frac{\dbinom42 \dbinom{28}8}{\dbinom{32}{10}} \quad \tbinom nk$$

Wurzeln: \sqrt

```
> $\sqrt{10} \quad \sqrt{(x+y)^2-(x-y)^2} \quad \sqrt[5]{1024}$
```

= $\sqrt{10} \quad \sqrt{(x+y)^2-(x-y)^2} \quad \sqrt[5]{1024}$

```
> \[ \sqrt{\frac{p^2}4 - q} \quad \sqrt[3]{1 + \sqrt{\sqrt[4]{10} + 1}} \]
```

= $$\sqrt{\frac{p^2}4 - q} \quad \sqrt[3]{1 + \sqrt{\sqrt[4]{10} + 1}}$$

Wenn einem die Positionierung des n in der n-ten Wurzel nicht gefällt, kann man dieses Zeichen mit Hilfe von zwei Befehlen (Paket amsmath) verschieben: \leftroot \uproot

```
> $\sqrt[n]{20} \quad \sqrt[\leftroot{-1} \uproot{2} n]{20}$
```

= $\sqrt[n]{20} \quad \sqrt[\leftroot{-1} \uproot{2} n]{20}$

4.3 Große Symbole

Je nachdem, ob diese Symbole im Inline-Mathematikmodus oder in einer abgesetzten Formel verwendet werden, sehen sie verschieden aus und zeigen unterschiedliches Verhalten.

Symbole ohne Grenzen als Inline-Formel: \sum \prod \coprod \bigoplus \bigodot \bigotimes \bigcup \bigcap \bigsqcup \biguplus \bigvee \bigwedge \int \iint \iiint \oint

```
> $\sum a_k \quad \prod a_k \quad \coprod a_k \quad \int f$
```

= $\sum a_k \quad \prod a_k \quad \coprod a_k \quad \int f$

```
> $\bigoplus U_k \quad \bigodot U_k \quad \bigotimes U_k \quad \iint f$
```

= $\bigoplus U_k \quad \bigodot U_k \quad \bigotimes U_k \quad \iint f$

```
> $\bigcup A_k \quad \bigcap A_k \quad \bigsqcup A_k \quad \iiint f$
```

= $\bigcup A_k \quad \bigcap A_k \quad \bigsqcup A_k \quad \iiint f$

```
> $\biguplus A_k \quad \bigvee A_k \quad \bigwedge A_k \quad \oint f$
```

= $\biguplus A_k \quad \bigvee A_k \quad \bigwedge A_k \quad \oint f$

Symbole ohne Grenzen als abgesetzte Formel:

```
> \[ \sum a_k \quad \prod a_k \quad \coprod a_k \quad \int f \]
```

= $$\sum a_k \quad \prod a_k \quad \coprod a_k \quad \int f$$

```
> \[ \bigoplus U_k \quad \bigodot U_k \quad \bigotimes U_k \quad \iint f \]
```

= $$\bigoplus U_k \quad \bigodot U_k \quad \bigotimes U_k \quad \iint f$$

```
> \[ \bigcup A_k \quad \bigcap A_k \quad \bigsqcup A_k \quad \iiint f \]
```

= $$\bigcup A_k \quad \bigcap A_k \quad \bigsqcup A_k \quad \iiint f$$

```
> \[ \biguplus A_k \quad \bigvee A_k \quad \bigwedge A_k \quad \oint f \]
```

= $$\biguplus A_k \quad \bigvee A_k \quad \bigwedge A_k \quad \oint f$$

Ausgewählte Symbole mit Grenzen als Inline-Formel:

```
> $\sum_{k=1}^n a_k \quad \bigoplus_{k=1}^n U_k \quad \int_a^b f$
```

= $\sum_{k=1}^n a_k \quad \bigoplus_{k=1}^n U_k \quad \int_a^b f$

```
> $\bigcup_{k=1}^n A_k \quad \bigvee_{k=1}^n A_k \quad \oint_\gamma f$
```

= $\bigcup_{k=1}^n A_k \quad \bigvee_{k=1}^n A_k \quad \oint_\gamma f$

Ausgewählte Symbole mit Grenzen als abgesetzte Formel:

```
> \[ \sum_{k=1}^n a_k \quad \bigoplus_{k=1}^n U_k \quad \int_a^b f \]
```

= $$\sum_{k=1}^n a_k \quad \bigoplus_{k=1}^n U_k \quad \int_a^b f$$

```
> \[ \bigcup_{k=1}^n A_k \quad \bigvee_{k=1}^n A_k \quad \oint_\gamma f \]
```

= $$\bigcup_{k=1}^n A_k \quad \bigvee_{k=1}^n A_k \quad \oint_\gamma f$$

In den abgesetzten Formeln werden die Grenzen bei Integalen standardmäßig seitlich gesetzt, weil das Integralzeichen selbst schon sehr lang ist. Bei allen anderen Symbolen stehen die Grenzen drüber und drunter.

\limits

Verschieben der Grenzen mit `\limits`:

```
> $\sum\limits_{k=1}^n a_k \quad \bigoplus\limits_{k=1}^n U_k$
```

= $\sum\limits_{k=1}^n a_k \quad \bigoplus\limits_{k=1}^n U_k$

```
> $\bigcup\limits_{k=1}^n A_k \quad \int\limits_a^b f$
```

= $\bigcup\limits_{k=1}^n A_k \quad \int\limits_a^b f$

```
> \[ \int\limits_a^b f \quad \oint\limits_\gamma f \]
```

= $$\int\limits_a^b f \quad \oint\limits_\gamma f$$

\nolimits

Verschieben der Grenzen mit `\nolimits`:

```
> \[ \sum\nolimits_{k=1}^n a_k \quad \bigcup\nolimits_{k=1}^n A_k \]
```

= $$\sum\nolimits_{k=1}^n a_k \quad \bigcup\nolimits_{k=1}^n A_k$$

Das Paket amsmath besitzt die vier Optionen `sumlimits` (Standard), `nosumlimits`, `intlimits` und `nointlimits` (Standard). Sie legen fest, ob die Voreinstellung für die Grenzen an den Summenzeichen (u. ä.) bzw. Integralzeichen in abgesetzten Formeln drüber und drunter bzw. seitlich daneben ist. Gelegentlich erzeugen diese Optionen seltsame Fehlermeldungen, nämlich wenn ein anderes Paket (z. B. amsxtra) vorher eingebunden wurde, das automatisch amsmath (ohne Optionen) lädt. In diesem Fall bindet man amsmath mit den Optionen einfach vor dem anderen ein.

Mehrere Zeilen in der Grenze (Paket amsmath): \substack subarray

```
> \[ \prod_{\substack{i=0 \\ i\neq m}}^n (x-x_i) \]
```

= $$\prod_{\substack{i=0 \\ i\neq m}}^n (x-x_i)$$

```
> \[ \prod_{\begin{subarray}{l} i=0 \\ i\neq m \end{subarray}}^n (x-x_i) \]
```

= $$\prod_{\begin{subarray}{l} i=0 \\ i\neq m \end{subarray}}^n (x-x_i)$$

Die `subarray`-Umgebung erhält ein zusätzliches Argument, je nachdem, ob die Zeilen zentriert (`c`) oder linksbündig (`l` – oder jeder andere String außer `c`) angeordnet werden sollen. Bei `\substack` hingegen sind die Zeilen immer zentriert.

Zeichen direkt an große Symbole setzen (Paket amsmath): \sideset

```
> \[ \sideset{}{'} \sum_{k\in\mathbbm Z} \frac1{k^2} \]
```

= $$\sideset{}{'}\sum_{k\in\mathbb{Z}} \frac1{k^2}$$

```
> \[ \sideset{_{aaa}^b}{_c^{ddd}} \sum_{k=1}^n x_k \]
```

= $$\sideset{_{aaa}^b}{_c^{ddd}}\sum_{k=1}^n x_k$$

Weitere solche Symbole (Paket stmaryrd):

```
> $\bigsqcap_{k=1}^n A_k \quad \bignplus_{k=1}^n A_k$
```

= $\bigsqcap_{k=1}^n A_k \quad \bignplus_{k=1}^n A_k$

```
> $\bigcurlyvee_{k=1}^n A_k \quad \bigcurlywedge_{k=1}^n A_k$
```

= $\bigcurlyvee_{k=1}^n A_k \quad \bigcurlywedge_{k=1}^n A_k$

```
> \[ \bigsqcap_{k=1}^n A_k \quad \bignplus_{k=1}^n A_k \]
```

= $$\bigsqcap_{k=1}^n A_k \quad \bignplus_{k=1}^n A_k$$

```
> \[ \bigcurlyvee_{k=1}^n A_k \quad \bigcurlywedge_{k=1}^n A_k \]
```

= $$\bigcurlyvee_{k=1}^n A_k \quad \bigcurlywedge_{k=1}^n A_k$$

4.4 Klammern

Einfache Klammern: ( )

```
> $a\cdot(b+c) \quad [-1,1] \quad \{2,3,5,7,11\} \quad \langle u,v\rangle$
```

= $a\cdot(b+c) \quad [-1,1] \quad \{2,3,5,7,11\} \quad \langle u,v\rangle$

```
> $\lceil\pi\rceil \quad \lfloor1.2345\rfloor \quad |x| \quad \|u\|$
```

= $\lceil\pi\rceil \quad \lfloor1.2345\rfloor \quad |x| \quad \|u\|$

Das Paket stmaryrd stellt noch die beiden folgenden Klammern zur Verfügung:

```
> $\mathbbm C \llbracket z\rrbracket$
```

= $\mathbb{C}\llbracket z\rrbracket$

[ ] \{ \} \langle \rangle \lceil \rceil \lfloor \rfloor | \|

\left \right

`\left` und `\right` passen die Klammergröße dem eingeschlossenen Text an:

```
> \[ \left( \frac12 \right) \quad \left[ -\sqrt2, \sqrt2 \right] \]
```

=
$$\left(\frac12 \right) \quad \left[-\sqrt2, \sqrt2 \right]$$

Diese müssen immer paarweise auftreten, wobei die linke und rechte Klammersorte verschieden sein dürfen:

```
> \[ \left| \sum_{k=1}^n \psi_k(x) \right\rangle \]
```

=
$$\left| \sum_{k=1}^n \psi_k(x) \right\rangle$$

`\left` und `\right` sind auch dann notwendig, wenn die Klammern zwar nicht größer sein müssen, wenn LATEX jedoch falsch interpretiert, welche Klammern öffnend und schließend sind. Das kann insbesondere bei offenen Intervallen der Fall sein:

```
> $x\in ]-1,1[ \quad x\in \left] -1,1 \right[$
```

= $x\in]-1,1[\quad x\in \left] -1,1 \right[$

.

Wird zu einem `\left` keine rechte Klammer benötigt, so muss man `\right.` schreiben:

```
> \[ f(x) = \left\{ ... \right. \]
```

=
$$f(x) = \left\{ \begin{array}{ll} x-3\,, & x<7 \\ 11-x\,, & x\geq 7 \end{array} \right.$$

Eine bessere Alternative für abschnittsweise definierte Funktionen folgt später.

\bigl \bigr \Bigl \Bigr \biggl \biggr \Biggl \Biggr

Mit `\bigl`, `\bigr` usw. (Paket `amsmath`) können Klammern manuell vergrößert werden:

```
> \[ \left( a\cdot(b+c) \right)^2 \quad \bigl( a\cdot(b+c) \bigr)^2 \]
```

=
$$\left(a\cdot(b+c) \right)^2 \quad \bigl(a\cdot(b+c) \bigr)^2$$

Hier ist die zweite Variante vor allem bei längeren Formeln deutlich leichter zu lesen.

```
> \[ \bigl( \quad \Bigl[ \quad \biggl\{ \quad \Biggl\langle \]
```

=
$$\bigl(\quad \Bigl[\quad \biggl\{ \quad \Biggl\langle$$

```
> \[ \bigr\rceil \quad \Bigr\rfloor \quad \biggr| \quad \Biggr\| \]
```

=
$$\bigr\rceil \quad \Bigr\rfloor \quad \biggr| \quad \Biggr\|$$

`\bigl` und `\bigr` müssen nicht zusammen auftreten.

\lgroup \rgroup \lmoustache \rmoustache

Die folgenden Klammern benötigen grundsätzlich ein vorgestelltes `\left`, `\bigl`, usw.:

```
> \[ \bigl\lgroup a+b \bigr\rgroup \]
```

=
$$\bigl\lgroup a+b \bigr\rgroup$$

```
> \[ \left\lmoustache \frac12 - \frac13 \right\rmoustache \]
```

=
$$\left\lmoustache \frac12 - \frac13 \right\rmoustache$$

\backslash

`\left` und `\right` bzw. `\bigl` und `\bigr` usw. können auch mit folgenden „Klammern“ verwendet werden:

```
> \[ \bigl\uparrow \quad \Bigl\downarrow \quad \biggl\Uparrow \]
```

=
$$\bigl\uparrow \quad \Bigl\downarrow \quad \biggl\Uparrow$$

```
> \[ \bigr\Downarrow \quad \Bigr/ \quad \biggr\backslash \]
```

= $$\bigr\Downarrow \quad \Bigr/ \quad \biggr\backslash$$

Für Skalarprodukte (wer sie so schreiben möchte) und Mengen eignet sich `\mid`, das eigentlich das Zeichen für „teilt“ darstellt: \mid

```
> $\langle x \mid y\rangle \quad \mathbbm R^+ = \{x\in\mathbbm R \mid x>0\}$
```

= $\langle x \mid y\rangle \quad \mathbb R^+ = \{x\in\mathbb R \mid x>0\}$

Für eine „Klammer“ zwischen `\bigl` und `\bigr` kann `\bigm` (Paket `amsmath`) benutzt werden, was zusammen mit `|` prinzipiell ein vergrößertes `\mid` darstellt: \bigm \Bigm \biggm \Biggm

```
> $U = \bigl\{ v\in V \bigm| A(v)=0 \bigr\}$
```

= $U = \bigl\{ v\in V \bigm| A(v)=0 \bigr\}$

Analog funktionieren die Befehle `\Bigm`, `\biggm` und `\Biggm`.

Benötigt man eine „mittlere Klammer“ zu `\left` und `\right` (z. B. weil sogar `\Biggm` zu klein ist), so kann man `\middle` verwenden: \middle

```
> \[ U = \left\{ ... \middle| a\in\mathbbm R \right\} \quad
  U = \left\{ ... \;\middle|\; a\in\mathbbm R \right\} \]
```

= $$U = \left\{ \begin{pmatrix} 2a \\ 0 \\ -a \end{pmatrix} \middle| a\in\mathbb R \right\} \quad U = \left\{ \begin{pmatrix} 2a \\ 0 \\ -a \end{pmatrix} \;\middle|\; a\in\mathbb R \right\}$$

Leider erzeugt `\middle` keine angemessenen Abstände wie `\bigm` usw., so dass man sie selbst eingeben muss. `\;` davor und dahinter ist gut geeignet.

\sin \cos \tan \cot \sec \csc \sinh \cosh \tanh \coth \arcsin \arccos \arctan \exp \ln \lg \log \inf \sup \min \max \lim \liminf \limsup \hom \dim \det \ker \arg \gcd \deg \Pr

4.5 Funktionsnamen

Versucht man, mathematische Funktionen wie den Sinus mit `sinx` zu setzen, erhält man ein eher unzufriedenstellendes Ergebnis. Statt dessen benutzt man den Befehl `\sin`:

```
> $sinx \quad \sin x$
```

= $sinx \quad \sin x$

Der Befehl `\sin` erzeugt den Funktionsnamen in aufrechter Schrift mit einem kleinen Abstand davor und dahinter.

Es gibt eine ganze Menge vordefinierter Befehle von diesem Typ:

```
> $\sin x \quad \cos x \quad \tan x \quad \cot x \quad \sec x \quad \csc x$
```

= $\sin x \quad \cos x \quad \tan x \quad \cot x \quad \sec x \quad \csc x$

```
> $\sinh x \quad \cosh x \quad \tanh x \quad \coth x$
```

= $\sinh x \quad \cosh x \quad \tanh x \quad \coth x$

```
> $\arcsin x \quad \arccos x \quad \arctan x$
```

= $\arcsin x \quad \arccos x \quad \arctan x$

```
> $\exp x \quad \ln x \quad \lg x \quad \log x \quad \inf A \quad \sup A$
```

= $\exp x \quad \ln x \quad \lg x \quad \log x \quad \inf A \quad \sup A$

```
> $\min A \quad \max A \quad \lim a_n \quad \liminf a_n \quad \limsup a_n$
```

= $\min A \quad \max A \quad \lim a_n \quad \liminf a_n \quad \limsup a_n$

```
> $\hom(V,V) \quad \dim V \quad \det A \quad \ker A$
```

= $\hom(V,V) \quad \dim V \quad \det A \quad \ker A$

```
> $\arg z \quad \gcd(a,b) \quad \deg p \quad \Pr(A)$
```

= $\arg z \quad \gcd(a,b) \quad \deg p \quad \Pr(A)$

Natürlich kann man für die Befehle, die es erfordern, auch einen Index erzeugen:

`> \[ \log_2 1024 \quad \lim_{n\to\infty} a_n \quad \dim_K V \]`

= $$\log_2 1024 \quad \lim_{n\to\infty} a_n \quad \dim_K V$$

Die Definition des Befehls entscheidet darüber, ob dieser Index im abgesetzten Modus drunter oder dahinter steht. Im Inlinemodus steht er immer dahinter:

`> $\log_2 1024 \quad \lim_{n\to\infty} a_n \quad \dim_K V$`

= $\log_2 1024 \quad \lim_{n\to\infty} a_n \quad \dim_K V$

\limits \nolimits \substack subarray

Auch hier kann man mit `\limits` bzw. `\nolimits` die jeweils andere Schreibweise erzwingen (vgl. Abschnitt 4.3). Dasselbe gilt für `\substack` und `subarray`. Die entsprechenden Optionen des `amsmath`-Pakets lauten `namelimits` und `nonamelimits`.

\bmod \mod \pmod \pod

Für die Modulo-Rechnung gibt es einige Befehle, die „mod“ ausgeben, allerdings in verschiedenen Formatierungen; `\mod` und `\pod` sind im Paket `amsmath`:

`> $a\equiv b \bmod{p^2} \quad a\equiv b \mod{p^2}$`

= $a\equiv b \bmod{p^2} \quad a\equiv b \mod{p^2}$

`> $a\equiv b \pmod{p^2} \quad a\equiv b \pod{p^2}$`

= $a\equiv b \pmod{p^2} \quad a\equiv b \pod{p^2}$

Es besteht kein Unterschied zum abgesetzten Modus:

`> \[ a\equiv b \bmod{p^2} \quad a\equiv b \mod{p^2} \]`

= $$a\equiv b \bmod{p^2} \quad a\equiv b \mod{p^2}$$

`> \[ a\equiv b \pmod{p^2} \quad a\equiv b \pod{p^2} \]`

= $$a\equiv b \pmod{p^2} \quad a\equiv b \pod{p^2}$$

\varliminf \varlimsup \projlim \injlim \varprojlim \varinjlim

Das `amsmath`-Paket stellt noch zur Verfügung:

`> $\varliminf a_n \quad \varlimsup a_n$`

= $\varliminf a_n \quad \varlimsup a_n$

`> $\projlim G_n \quad \injlim G_n \quad \varprojlim G_n \quad \varinjlim G_n$`

= $\operatorname{proj\,lim} G_n \quad \operatorname{inj\,lim} G_n \quad \varprojlim G_n \quad \varinjlim G_n$

\operatorname \operatorname*

Einerseits fehlen eine ganze Menge von Funktionen wie Areasinushyperbolicus, Spur, Bild, usw., andererseits haben manche Befehle wie `\ker` oder `\gcd` eine englische Schreibweise. Mit dem bereits veralteten Befehl `\operatorname` kann man Text als einen solchen eigenen Funktionsnamen formatieren:

`> $\operatorname{arsinh} x \quad \operatorname{Bild} A$`

= $\operatorname{arsinh} x \quad \operatorname{Bild} A$

Den Befehl `\operatorname*` verwendet man, wenn der Index im abgesetzten Modus unter dem Wort stehen soll:

`> \[ \operatorname*{Res}_{z=a} f(z) \]`

= $$\operatorname*{Res}_{z=a} f(z)$$

Der neue Befehl `\DeclareMathOperator` wird in Abschnitt 6.1 erklärt.

4.6 Matrizen und Auslassungspunkte

pmatrix & \\

Für Matrizen stellt das Paket `amsmath` vorgefertigte Befehle zur Verfügung:

```
> \[ \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\
  \sin\alpha & \cos\alpha & 0 \\
  0 & 0 & 1 \end{pmatrix} \]
```

$$= \quad \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Dabei wechselt der Befehl `&` in die nächste Spalte und `\\` in die nächste Zeile der Matrix.

Statt `pmatrix`[4] mit runden Klammern als Begrenzer kann man auch `bmatrix`[5] (eckige), `Bmatrix`[6] (geschweifte), `vmatrix`[7] (Determinantenstriche) und `Vmatrix`[8] (Normstriche) verwenden: bmatrix Bmatrix vmatrix Vmatrix

```
> \[ \begin{bmatrix} 1 & 2 \\ 3 & -1 \end{bmatrix} \quad
  \begin{Bmatrix} x^2 & 0 \\ 0 & y^2 \end{Bmatrix} \]
```

$$= \quad \begin{bmatrix} 1 & 2 \\ 3 & -1 \end{bmatrix} \quad \begin{Bmatrix} x^2 & 0 \\ 0 & y^2 \end{Bmatrix}$$

```
> \[ \begin{vmatrix} 1 & 2 \\ 3 & -1 \end{vmatrix} \quad
  \begin{Vmatrix} x^2 & 0 \\ 0 & y^2 \end{Vmatrix} \]
```

$$= \quad \begin{vmatrix} 1 & 2 \\ 3 & -1 \end{vmatrix} \quad \begin{Vmatrix} x^2 & 0 \\ 0 & y^2 \end{Vmatrix}$$

Auch in Matrizen kann `\\` ein optionales Argument für zusätzlichen Zeilenabstand erhalten:

```
> \[ \begin{pmatrix}\frac12&\frac13 \\[0.8ex] \frac13&\frac14\end{pmatrix} \]
```

$$= \quad \begin{pmatrix}\frac12&\frac13 \\[0.8ex] \frac13&\frac14\end{pmatrix}$$

Mit den Befehlen `\cdots`[9], `\vdots`[10] und `\ddots`[11] kann man Auslassungspunkte in Matrizen erzeugen: \cdots \vdots \ddots

```
> \[ E = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & & \vdots \\
  \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix} \]
```

$$= \quad E = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix}$$

Die Auslassungspunkte können natürlich auch außerhalb von Matrizen benutzt werden: \ldots \dots

```
> $a_1\cdots a_n \quad a_1,\ldots,a_n \quad a_1+\ldots+a_n$
```

= $a_1\cdots a_n \quad a_1,\ldots,a_n \quad a_1+\ldots+a_n$

Dabei setzt `\ldots`[12] drei Punkte auf die Grundreihe. Auch im Mathematikmodus kann `\dots` verwendet werden, und die drei Punkte werden je nach Kontext als `\ldots` oder `\cdots` gesetzt.

Das Paket `mathdots` stellt einen Befehl zur Verfügung, um diagonale Punkte in die andere Richtung zu erzeugen: \iddots

```
> \[ \begin{pmatrix} & & y \\ & \iddots & \\ x & & \end{pmatrix} \]
```

[4] `p` steht für „parentheses“.
[5] `b` steht für „brackets“.
[6] `B` steht für „braces“.
[7] `v` steht für „vertical“.
[8] `V` steht für „Vertical“.
[9] `c` steht für „centered“.
[10] `v` steht für „vertical“.
[11] `d` steht für „diagonal“.
[12] `l` steht für „lower“.

= $$\begin{pmatrix} & & y \\ & \iddots & \\ x & & \end{pmatrix}$$

\hdotsfor Mit `\hdotsfor` (Paket amsmath) können horizontale Punkte über mehrere Spalten erstreckt werden:

```
> \[ \begin{pmatrix} 1&0&1&0&1&0 \\ 2 & \hdotsfor{4} & 2 \end{pmatrix} \]
```

= $$\begin{pmatrix} 1&0&1&0&1&0 \\ 2 & \hdotsfor{4} & 2 \end{pmatrix}$$

matrix Matrizen ohne Klammern können durch `matrix` erzeugt werden, z. B.

```
> \[ \left( \begin{matrix} 3 & -1 \\ 7 & 4 \end{matrix} \;\middle|\;
  \begin{matrix} 1 & 0 \\ 0 & 1 \end{matrix} \right) \quad
  \begin{pmatrix} \begin{matrix} 0 & -5 \\ 1 & 2 \end{matrix} & \mathrm O \\
  \mathrm O & \begin{matrix} 0 & -5 \\ 1 & 2 \end{matrix} \end{pmatrix} \]
```

= $$\left(\begin{matrix} 3 & -1 \\ 7 & 4 \end{matrix} \;\middle|\; \begin{matrix} 1 & 0 \\ 0 & 1 \end{matrix} \right) \quad \begin{pmatrix} \begin{matrix} 0 & -5 \\ 1 & 2 \end{matrix} & \mathrm O \\ \mathrm O & \begin{matrix} 0 & -5 \\ 1 & 2 \end{matrix} \end{pmatrix}$$

array Für Matrizen mit anderer Ausrichtung der Spalten oder senkrechten Strichen sowie für spezielle Konstruktionen kann man die Umgebung `array` verwenden:

```
> \[ \left( \begin{array}{cc|cc} 3 & -1 & 1 & 0 \\ 7 & 4 & 0 & 1 \end{array}
  \right) \quad \begin{array}{clr} 1 & 2 & 3 \\ 111 & 222 & 333 \end{array} \]
```

= $$\left(\begin{array}{cc|cc} 3 & -1 & 1 & 0 \\ 7 & 4 & 0 & 1 \end{array} \right) \quad \begin{array}{clr} 1 & 2 & 3 \\ 111 & 222 & 333 \end{array}$$

Die Syntax ist genau dieselbe wie bei `pmatrix` usw., nur dass die Umgebung ein Argument zur Spaltendefinition erhält. In dieser kann man – ähnlich wie bei `subarray` – `l`, `r` und `c` für linksbündige, rechtsbündige und zentrierte Spalten verwenden. Der senkrechte Strich `|` in der Spaltendefinition steht für eine durchgehende senkrechte Linie in der „Tabelle". Weiteres dazu ist im Abschnitt 8.1 zu finden.

cases Mit der `array`-Umgebung könnte man auch eine abschnittsweise definierte Funktion wie auf Seite 36 setzen. Das Paket amsmath stellt jedoch hierfür die vorgefertigte Umgebung `cases` zur Verfügung:

```
> \[ f(x) = \begin{cases} x-3 \,, & x<7 \\ 11-x \,, & x\geq7 \end{cases} \]
```

= $$f(x) = \begin{cases} x-3 \,, & x<7 \\ 11-x \,, & x\geq7 \end{cases}$$

small-matrix Wer unbedingt Matrizen in kleinerer Form im Fließtext braucht, kann sich mit dem Paket amsmath folgendermaßen behelfen:

```
> Kleine Matrizen können durch $\bigl( \begin{smallmatrix} a & b \\ c & d
  \end{smallmatrix} \bigr)$ erzeugt werden.
```

= Kleine Matrizen können durch $\bigl(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix} \bigr)$ erzeugt werden.

4.7 Kommutative Diagramme

Im Wesentlichen gibt es zwei Pakete für kommutative Diagramme. amscd ist speziell für sie gemacht, unterstützt aber keine schrägen Pfeile oder Pfeile mit besonderen Enden, dafür sehen die Pfeilspitzen aber genauso aus wie in $\to$ (`\to`). `xy` hingegen stellt noch ganz andere Zeichenbefehle zur Verfügung, unterstützt aber auch alle Pfeilrichtungen und -sorten. Allerdings haben die Standardpfeile die Gestalt $\longrightarrow$, was sich von den anderen mathematischen Pfeilen unterscheidet.

Für das Paket `amscd` betrachten wir zunächst ein Beispiel:

CD
@>>>
@<<<
@VVV
@AAA
\\

```
> \[ \begin{CD}
   V   @>\varphi>>   W \\
   @V{I_V}VV   @VV{I_W}V \\
   V^{\vee\vee}   @>>{\varphi^{\vee\vee}}>   W^{\vee\vee}
  \end{CD} \]
```

=

$$\begin{tikzcd} V \arrow[r, "\varphi"] \arrow[d, "I_V"'] & W \arrow[d, "I_W"] \\ V^{\vee\vee} \arrow[r, "\varphi^{\vee\vee}"'] & W^{\vee\vee} \end{tikzcd}$$

Die vier Ecken wurden ganz normal mit `V`, `W`, `V^{\vee\vee}` und `W^{\vee\vee}` erzeugt. Die Pfeile in die vier Richtungen werden von `@>>>` (nach rechts), `@<<<` (nach links), `@VVV` (nach unten) und `@AAA` (nach oben) ausgegeben. Eingaben zwischen dem ersten und dem zweiten `>`, `<`, `V` bzw. `A` werden über bzw. links neben den Pfeil, Eingaben zwischen dem zweiten und dritten dieser Zeichen werden unter bzw. rechts neben den Pfeil geschrieben.

`@.` steht für einen unsichtbaren horizontalen oder vertikalen Pfeil. Eine Doppellinie für Gleichheit wird horizontal von `@=` und vertikal von `@|` erzeugt.

@.
@=
@|

Vom anderen Paket `xy` beschreiben wir hier nur die Funktionen, die wir für die Diagramme benötigen, und binden es wie folgt ein:

xy
\xymatrix
&
\\

```
> \usepackage[arrow,curve,matrix]{xy}
```

Die Umgebung heißt ebenfalls `xy`. Darin benutzt man den Befehl `\xymatrix`, in dem man wie gewöhnlich mit `&` und `\\` Zeilen und Spalten setzen kann:

```
> \[ \begin{xy} \xymatrix{A & B \\ C & D} \end{xy} \]
```

=

$$\begin{array}{cc} A & B \\ \\ C & D \end{array}$$

Für die Pfeile folgendes Beispiel:

\ar
^
_
|

```
> \[ \begin{xy} \xymatrix{
   A \ar[r] \ar[d] & B \ar[d]_{\widetilde\varphi} \ar[rdd]^\varphi & \\
   C \ar[r]^{\widetilde\psi} \ar[rrd]_\psi & D \ar[rd]|\alpha & \\
   & & E
  } \end{xy} \]
```

=

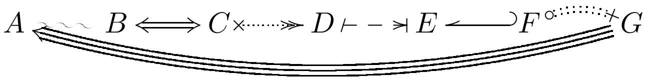
$$\begin{tikzcd} A \arrow[r] \arrow[d] & B \arrow[d, "\widetilde\varphi"'] \arrow[rdd, "\varphi"] & \\ C \arrow[r, "\widetilde\psi"] \arrow[rrd, "\psi"'] & D \arrow[rd, "\alpha" description] & \\ & & E \end{tikzcd}$$

Ein Pfeil wird in diejenige Tabellenzelle geschrieben, von der er losläuft. Dazu dient der Befehl `\ar`, der in eckigen Klammern eine Folge der Buchstaben `r`, `l`, `d` und `u` erhält. Diese beschreiben, wohin der Pfeil zeigen soll. Mit einem nachfolgenden `^`, `_` oder `|` setzt man etwas links oder rechts an den Pfeil (in Pfeilrichtung gesehen) oder auf den Pfeil.

@{}
@/^/
@/_/

Das folgende Beispiel zeigt die möglichen Modifikationen der Pfeile:

```
> \[ \begin{xy} \xymatrix{
    A \ar@{~}[r] & B \ar@{<=>}[r] & C \ar@{x.>>}[r] & D \ar@{/-->|}[r] &
    E \ar@^{<-)}[r] & F \ar@/^/@{o:+}[r] & G \ar@/^6mm/@3{->}[llllll]
  } \end{xy} \]
=
```



Der Modifikator `@{}` definiert Anfang, Schaft und Ende des Pfeils. Zwingend zwischen den geschweiften Klammern ist eine der Angaben `-`, `=`, `.`, `:`, `--` oder `~`, die den Schaft genauso wie angegeben aussehen lassen. Davor oder dahinter kann, unabhängig voneinander, eine der Angaben `>`, `<`, `>>`, `<<`, `|`, `||`, `>|`, `|<`, `)`, `(`, `/`, `//`, `x`, `o` oder `+` stehen, wodurch die Pfeilspitze modifiziert wird. Entfällt die Angabe, so wird keine Spitze gezeichnet. Zwischen `@` und `{` des Modifikators darf noch eines der Zeichen `2`, `3`, `^` oder `_` stehen. `2` führt zu einem doppelten (unsinng bei `=` und `:`), `3` zu einem dreifachen Schaft. `^` und `_` verschieben die Pfeilspitzen nach links oder rechts (in Pfeilrichtung gesehen); dabei werden `>`, `<`, `>>` und `<<` zu halben Pfeilen, und `|`, `||`, `)` und `(` werden tatsächlich verschoben; die anderen Spitzen bleiben unberührt.

Der Modifikator `@//` krümmt den Pfeil, und zwar mit `^` nach links (also Rechtskurve) und mit `_` nach rechts (also Linkskurve). Zusätzlich kann noch eine Länge angegeben werden, die das Maß der Krümmung definiert.

@<>
^//
_//
|//

Mit zwei weiteren Modifikatoren kann ein Pfeil senkrecht zu seinem Schaft und eine Pfeilbeschriftung entlang des Schafts verschoben werden. Dazu ein Beispiel:

```
> \[ \begin{xy} \xymatrix{
    A \ar@<1mm>[r]^f & B \ar@<1mm>[l]^{f^{-1}} \ar[rr]_/5mm/\varphi & & C
  } \end{xy} \]
=
```

$$A \underset{f^{-1}}{\overset{f}{\rightleftarrows}} B \xrightarrow[\varphi]{\qquad\qquad} C$$

Der Modifikator `@<>` verschiebt den Pfeil um die Länge, die zwischen den spitzen Klammern angegeben ist. Mit `//` nach dem `^`, `_` oder `|` der Beschriftung wird diese um die angegebene Länge verschoben.

Kapitel 5

Mathematische Umgebungen

5.1 Inline-Modus

Mit `$ $` wird ein Inline-Mathematikmodus definiert: $ $

```
> Die Gleichung $3x=1$ hat genau eine Lösung in $\mathbbm R$.
```
math

= Die Gleichung $3x = 1$ hat genau eine Lösung in $\mathbb{R}$.

Alternativ kann auch die Umgebung `math` benutzt werden:

```
> Die Gleichung \begin{math} 0x=7 \end{math} ist unlösbar.
```

= Die Gleichung $0x = 7$ ist unlösbar.

Allein wegen der Schreibersparnis ist die Umgebung mit `$ $` vorzuziehen.

LaTeX besitzt die Fähigkeit, Inline-Umgebungen an geeigneter Stelle in die nächste Zeile umzubrechen, wenn das nötig sein sollte:

```
> Komposition: Für zwei Funktionen $f\colon\mathbbm R\supseteq A\ni x\mapsto
  y\in B\subseteq\mathbbm R$ und $g\colon\mathbbm R\supseteq B\ni y\mapsto
  z\in C\subseteq\mathbbm R$ definiert man \dots
```

= Komposition: Für zwei Funktionen $f\colon \mathbb{R} \supseteq A \ni x \mapsto y \in B \subseteq \mathbb{R}$ und $g\colon \mathbb{R} \supseteq B \ni y \mapsto z \in C \subseteq \mathbb{R}$ definiert man …

Die `$ $`-Umgebung ist robust und kann daher auch in Befehlen wie `\section` stehen:

```
> \section{Gleichungen der Form $x^2+px+q$}
```

= **2.6 Gleichungen der Form $x^2 + px + q$**

Wie man den Mathematikmodus fett setzen kann, betrachten wir in Abschnitt 5.3. Im Übrigen ist die `math`-Umgebung nicht robust und zerbricht in den Gliederungsbefehlen, z. B. beim Kopieren ins Inhaltsverzeichnis.

5.2 `equation`-, `gather`- und `align`-Umgebung

Zunächst sei erwähnt, dass innerhalb aller nun folgenden Mathematik-Umgebungen Leerzeilen im Quelltext verboten sind. In der Regel gibt LaTeX dann mehrere Fehlermeldungen aus. Wer den Quelltext von sehr langen Formeln dennoch lesbarer gestalten möchte, kann Zeilen einfügen, die ausschließlich ein `%` (Kommentar) enthalten.

Um eine einzeilige abgesetzte Formel zu erzeugen, dient die `equation`-Umgebung: equation

```
> Die Gleichung
  \begin{equation} x^2 + px + q = 0 \end{equation}
  kann beispielsweise mit der $pq$-Formel gelöst werden.
```

= Die Gleichung

$$x^2 + px + q = 0 \tag{5.1}$$

kann beispielsweise mit der pq-Formel gelöst werden.

Die so erzeugte Formel wird automatisch am rechten Seitenrand nummeriert.

equation* \[\]

Um die Nummerierung zu unterdrücken, kann die Umgebung `equation*` verwendet werden. Dies ist dann äquivalent zur abkürzenden Schreibweise mit `\[ \]`:

```
> Die Reihe
  \[ \sum_{k=2}^\infty \frac1{k^2-1} \]
  berechnen wir \dots
```

= Die Reihe

$$\sum_{k=2}^{\infty} \frac{1}{k^2-1}$$

berechnen wir …

gather gather* \\

Um mehrere Gleichungen untereinanderzuschreiben, kann man `gather` (Paket `amsmath`) verwenden:

```
> Es gelten die Beziehungen
  \begin{gather}
   \ln(x\cdot y) = \ln x + \ln y \,, \\
   \sin(x+y) = \sin x\cos y + \cos x\sin y \,.
  \end{gather}
```

= Es gelten die Beziehungen

$$\ln(x \cdot y) = \ln x + \ln y \,, \tag{5.2}$$

$$\sin(x + y) = \sin x \cos y + \cos x \sin y \,. \tag{5.3}$$

Jede Zeile erhält ihre eigene Nummer; man kann sie alle unterdrücken, indem man `gather*` benutzt. `\\` wird verwendet, um in die nächste Zeile zu wechseln.

align align* &

Mit `align` (Paket `amsmath`) kann man mehrere Zeilen an je einem Zeichen ausrichten:

```
> Es gelten die Beziehungen
  \begin{align*}
   \ln(x\cdot y) &= \ln x + \ln y \,, \\
   \sin(x+y) &= \sin x\cos y + \cos x\sin y \,.
  \end{align*}
```

= Es gelten die Beziehungen

$$\begin{aligned} \ln(x \cdot y) &= \ln x + \ln y \,, \\ \sin(x + y) &= \sin x \cos y + \cos x \sin y \,. \end{aligned}$$

Das Zeichen `&` kennzeichnet dabei das Symbol, an dem ausgerichtet werden soll. Dieses kann von Zeile zu Zeile unterschiedlich sein.

\display-break

Es ist zu beachten, dass innerhalb einer `gather`- und `align`-Umgebung standardmäßig kein Seitenumbruch durchgeführt wird. Natürlich kann man die Umgebung schließen und neu öffnen; eine andere, in der Regel bessere Möglichkeit bietet `\displaybreak` (Paket `amsmath`):

```
> \begin{align*}
   \ln(x\cdot y) &= \ln x + \ln y \,, \displaybreak[0] \\
   \sin(x+y) &= \sin x\cos y + \cos x\sin y \,.
  \end{align*}
```

Mit dem Argument wird eine Dringlichkeit (steigend von 0 nach 4) angegeben; 4 erzwingt einen Seitenumbruch, genauso wie kein angegebenes Argument.

Möchte man dieses Verhalten globalisieren, so kann man z. B. in der Präambel den Befehl `\allowdisplaybreaks` (Paket `amsmath`) – ebenfalls mit einer Zahl zwischen 0 und 4 – benutzen. Dadurch wird an jedem Zeilenwechsel in einer Formelumgebung ein Seitenwechsel mit der angegebenen Dringlichkeit erzeugt. Wenn er an einer Stelle wiederum nicht erwünscht ist, muss er mit `\\*` unterbunden werden.

\allowdisplaybreaks
*

5.3 Weitere wichtige Befehle

\text

Der Befehl `\text` (Paket `amsmath`) wechselt vom Mathematikmodus zurück in den Textmodus:

```
> \[ C\bigl( [a,b], \mathbbm R \bigr) = \bigl\{ f\colon[a,b]\to\mathbbm R
  \bigm| f \text{ ist stetig auf } [a,b] \bigr\} \]
```

= $$C\bigl([a,b],\mathbb{R}\bigr) = \bigl\{f\colon[a,b]\to\mathbb{R} \bigm| f \text{ ist stetig auf } [a,b]\bigr\}$$

Innerhalb von `\text` gilt dieselbe Schriftformatierung wie an der Stelle, an der die Mathematikumgebung geöffnet wurde:

```
> \textsc{Aus $a=b \text{ und } c=d$ folgt \dots}
```

= AUS $a=b$ UND $c=d$ FOLGT …

Außerdem kann man mit `$ $` natürlich wieder einen Mathematikmodus einfügen:

```
> $f(x)=0 \text{ gilt $\mu$-fast überall.}$
```

= $f(x)=0$ gilt μ-fast überall.

Das erste Beispiel dient nur der Demonstration; klarerweise würde man statt dessen besser `$a=b$ und $c=d$` schreiben.

\intertext

Der Befehl `\intertext` (Paket `amsmath`) erlaubt es, normalen Fließtext zu setzen, ohne die mathematische Umgebung beenden zu müssen. Das ist dann hilfreich, wenn die Ausrichtung mittels `align` erhalten bleiben soll. Dabei steht vor `\intertext` kein `\\`:

```
> Es gilt
  \begin{align*}
   \ln(x\cdot y) &= \ln x + \ln y
   \intertext{und}
   \sin(x+y) &= \sin x\cos y + \cos x\sin y \,.
  \end{align*}
```

= Es gilt

$$\ln(x\cdot y) = \ln x + \ln y$$

und

$$\sin(x+y) = \sin x\cos y + \cos x\sin y\,.$$

\displaystyle
\textstyle
\scriptstyle
\scriptscriptstyle

Wir haben schon gesehen, dass man in einer Textformel mit `\dfrac` einen Bruch wie in einer abgesetzten Formel und in einer abgesetzten Formel mit `\tfrac` einen Bruch wie in einer Textformel setzen kann. Dies funktioniert noch allgemeiner:

```
> Das Integral $\displaystyle \int_a^b f(x)\,\mathrm{d}x$ lässt sich \dots
```

= Das Integral $\displaystyle\int_a^b f(x)\,\mathrm{d}x$ lässt sich …

```
> \[ A\Bigl( {\textstyle \int_a^b f(x)\,\mathrm{d}x} \Bigr) = \frac12 \]
```

= $$A\Bigl(\int_a^b f(x)\,\mathrm{d}x\Bigr) = \frac{1}{2}$$

`\displaystyle` schaltet in den abgesetzten Schriftmodus, `\textstyle` in den Inline-Schriftmodus. Dieser bleibt dann so lange erhalten, bis ein neuer solcher Befehl folgt oder die Umgebung, in der `\displaystyle` bzw. `\textstyle` steht, geschlossen wird. Daher steht das Integral im zweiten Beispiel in einer anonymen Umgebung (durch `{` und `}`), und `\textstyle` wirkt nicht mehr auf `\frac`. Es gibt auch noch `\scriptstyle` bzw. `\scriptscriptstyle`, die auf den Schriftmodus umschalten, der für Exponenten/Indizes erster bzw. zweiter Stufe verwendet wird.

\boldmath \unboldmath

Der Mathematikmodus kann auch komplett fett gesetzt werden; dazu gibt es die Befehle `\boldmath` und `\unboldmath`. Die Befehle müssen *außerhalb* das Mathematikmodus stehen:

```
> $f\colon D\to\mathbbm R$ heißt \boldmath\textbf{differenzierbar in
  $x_0\in D$}\unboldmath, wenn \dots
```

= $f\colon D\to\mathbb{R}$ heißt **differenzierbar in $\boldsymbol{x_0\in D}$**, wenn …

Mit dem Paket stmaryrd gibt LaTeX bei jeder Mathematikumgebung innerhalb von `\boldmath` eine Warnung aus, weil einige Symbole dieses Pakets nicht im fetten Schriftgewicht existieren.

\boxed

Schließlich kann man Formeln mit einem Kasten versehen:

```
> \[ \boxed{\int_{\partial M} \omega = \int_M \mathrm{d}\omega} \]
```

= $$\boxed{\int_{\partial M}\omega = \int_M \mathrm{d}\omega}$$

5.4 Geschachtelte Formeln und Formelnummern

\nonumber

Mit dem Befehl `\nonumber` (Paket amsmath) kann man gezielt Formelnummern unterdrücken:

```
> Für die Koeffizienten ergibt sich:
  \begin{align}
   a &= 2 \\
   b &= 3 \nonumber \\
   c &= -1 \nonumber \\
   d &= 0
  \end{align}
```

= Für die Koeffizienten ergibt sich:

$$a = 2 \tag{5.4}$$
$$b = 3$$
$$c = -1$$
$$d = 0 \tag{5.5}$$

\tag \tag*

Mit `\tag` und `\tag*` (Paket amsmath) kann der Formelzeile eine Nummer zugewiesen werden:

```
> Für die Koeffizienten ergibt sich:
  \begin{align*}
   a &= 2 \tag{x} \\
   c &= -1 \tag*{y}
  \end{align*}
```

= Für die Koeffizienten ergibt sich:

$$a = 2 \tag{x}$$
$$c = -1 \tag*{y}$$

Der Inhalt wird allerdings nicht im Mathematikmodus gesetzt; `\tag*` unterdrückt die runden Klammern, die `\tag` standardmäßig einfügt. Um Gleichungen mit $(*)$ zu markieren, eignet sich `\tag{$*$}`.

Möchte man allen Gleichungen zusammen eine (zentrierte) Formelnummer geben, so kann man dies leicht mit `aligned` (Paket `amsmath`) lösen: aligned

```
> Für die Koeffizienten ergibt sich:
  \begin{equation} \begin{aligned}
   a &= 2 \\
   c &= -1
  \end{aligned} \end{equation}
```

= Für die Koeffizienten ergibt sich:

$$\begin{aligned} a &= 2 \\ c &= -1 \end{aligned} \tag{5.6}$$

Die `aligned`-Umgebung kann nur innerhalb einer abgesetzten Mathematikumgebung verwendet werden, weil sie selbst keinen Mathematikmodus öffnet. Die Formelnummer wird im Beispiel von `equation` generiert.

Nach demselben Muster funktioniert auch `gathered` (Paket `amsmath`): gathered

```
> Einige Umformungen:
  \begin{gather} \begin{gathered}
   \sin(x+y) = \sin x\cos y + \cos x\sin y \,, \\
   \mathrm{e}^{x+y} = \mathrm{e}^x \mathrm{e}^y \,;
  \end{gathered} \\ \begin{gathered}
   \cos(x+y) = \cos x\cos y - \sin x\sin y \,, \\
   \ln(x\cdot y) = \ln x + \ln y \,.
  \end{gathered} \end{gather}
```

= Einige Umformungen:

$$\begin{gathered} \sin(x+y) = \sin x\cos y + \cos x\sin y \,, \\ \mathrm{e}^{x+y} = \mathrm{e}^x \mathrm{e}^y \,; \end{gathered} \tag{5.7}$$

$$\begin{gathered} \cos(x+y) = \cos x\cos y - \sin x\sin y \,, \\ \ln(x\cdot y) = \ln x + \ln y \,. \end{gathered} \tag{5.8}$$

`gathered` bzw. `aligned` kann man auch wie im folgenden Beispiel benutzen:

```
> \[ (a+2b)x^2 + (2c-a)x + (b+c) = 3x^2 - 5x + 1 \quad\Longrightarrow\quad
  \left\{ \begin{aligned}
   a + 2b &= 3 \\
   2c - a &= -5 \\
   b + c &= 1
  \end{aligned} \right. \]
```

= $$(a+2b)x^2 + (2c-a)x + (b+c) = 3x^2 - 5x + 1 \quad\Longrightarrow\quad \left\{ \begin{aligned} a + 2b &= 3 \\ 2c - a &= -5 \\ b + c &= 1 \end{aligned} \right.$$

subequations

Im Paket amsmath befindet sich noch eine Umgebung subequations, mit der sich Unternummern bei Gleichungen erzeugen lassen:

```
> Es gilt
  \begin{subequations} \begin{align}
   \ln(x\cdot y) &= \ln x + \ln y \,, \\
   \sin(x+y) &= \sin x\cos y + \cos x\sin y \,.
  \end{align} \end{subequations}
```

= Es gilt

$$\ln(x \cdot y) = \ln x + \ln y \tag{5.9a}$$

$$\sin(x + y) = \sin x \cos y + \cos x \sin y\,. \tag{5.9b}$$

Innerhalb der Umgebung darf auch beliebiger zusätzlicher Quelltext stehen, da sie nichts anderes tut, als einen zusätzlichen Buchstaben a, b, c, usw. an die Gleichungsnummer anzuhängen. Es ist sogar möglich, die Umgebung ineinanderzuschachteln; dann erstellt LATEX Gleichungsnummern wie 5.9aa, 5.9ab, usw.

5.5 Weitere Formelumgebungen

align
align*
aligned

Die Umgebungen align(*) und aligned können auch für mehrere Spalten benutzt werden:

```
> \begin{align*}
   a &= b   &   x &= y \\
   a &= b + c   &   x &= y + z
  \end{align*}
=
```

$$\begin{aligned} a &= b & x &= y \\ a &= b + c & x &= y + z \end{aligned}$$

Dabei geben die & sowohl die Zeichen zur Ausrichtung als auch die Spaltentrennung an.

```
> \begin{equation} \begin{aligned}
   a &= b   &   x &= y \\
   a &= b + c   &   x &= y + z
  \end{aligned} \end{equation}
=
```

$$\begin{aligned} a &= b & x &= y \\ a &= b + c & x &= y + z \end{aligned} \tag{5.10}$$

In der aligned-Umgebung wird der Spaltenabstand deutlich kleiner gesetzt.

alignat
alignat*

Möchte man den Spaltenabstand selbst festlegen, so kann man alignat(*) (Paket amsmath) verwenden:

```
> \begin{alignat*}{2}
   a &= b   &   x &= y \\
   a &= b + c   &   x &= y + z
  \end{alignat*}
=
```

$$\begin{alignedat}{2} a &= b & x &= y \\ a &= b + c & x &= y + z \end{alignedat}$$

Das zusätzliche Argument beim Beginn der Umgebung gibt die Anzahl der Spalten an. Wie man sieht, wird zunächst gar kein Spaltenabstand erzeugt; mit z. B. einem `\qquad` an einer geeigneten Stelle kann man ihn manuell einfügen:

```
> \begin{alignat*}{2}
   a &= b  &  x &= y \\
   a &= b + c \qquad  &  x &= y + z
  \end{alignat*}
=
```

$$\begin{alignedat}{2} a &= b \qquad & x &= y \\ a &= b + c \qquad & x &= y + z \end{alignedat}$$

Der Vollständigkeit halber sei auch noch `xalignat(*)` (Paket `amsmath`) erwähnt: xalignat xalignat*

```
> \begin{xalignat*}{2}
   a &= b  &  x &= y \\
   a &= b + c  &  x &= y + z
  \end{xalignat*}
=
```

$$\begin{aligned} a &= b & x &= y \\ a &= b + c & x &= y + z \end{aligned}$$

Mit `flalign(*)` (Paket `amsmath`) werden die Zeilen komplett ausgefüllt: flalignat flalignat*

```
> \begin{flalign}
   a &= b  &  x &= y \\
   a &= b + c  &  x &= y + z
  \end{flalign}
=
```

$$a = b \qquad\qquad x = y \tag{5.11}$$

$$a = b + c \qquad\qquad x = y + z \tag{5.12}$$

Hat man eine Formel, die nicht in eine Zeile passt, so kann man sie mittels `multline(*)` (Paket `amsmath`) aufspalten: multline multline* \shoveleft \shoveright

```
> \begin{multline}
   a + b \\
   = c + d \\
   = e + f
  \end{multline}
=
```

$$\begin{multline} a + b \\ = c + d \\ = e + f \tag{5.13} \end{multline}$$

Hier werden die erste Zeile linksbündig, die letzte rechtsbündig und alle dazwischen zentriert gesetzt. Um eine Zeile entgegen dieser Voreinstellung links- oder rechtsbündig zu setzen, benutzt man `\shoveleft` oder `\shoveright` (beide Paket `amsmath`). Die Formelzeile selbst wird dann zum Argument dieses Befehls, also z. B. `\shoveleft{= c + d}`.

Eine andere Möglichkeit, eine zu lange Formel in mehrere Zeilen zu verteilen, bietet `split`: split

```
> \begin{equation} \begin{split}
   a + b &= c + d \\
   &= e + f
  \end{split} \end{equation}
```

=

$$\begin{split} a + b &= c + d \\ &= e + f \end{split} \tag{5.14}$$

`split` verhält sich ähnlich zu `aligned`, nur dass es keine mehrfachen Ausrichtungen in einer Zeile erlaubt.

Die Umgebung `eqnarray(*)` sollte nicht mehr benutzt werden, weil sie u. a. beim Generieren der Formelnummern Probleme bereiten kann. Sie ist alter LaTeX-Standard und nicht mehr mit amsmath kompatibel.

5.6 Umgebungen für Definitionen, Sätze, Lemmata, usw.

\newtheorem

LaTeX erlaubt es auf relativ einfache Weise, eigene Umgebungen zu definieren, die automatisch einen Titel wie „Definition“, „Satz“, „Lemma“, usw. und eine fortlaufende Nummer erhalten. Dazu betrachten wir zuerst das folgende Beispiel. Die erste Zeile stehe in der Präambel, die weiteren zwischen `\begin{document}` und `\end{document}` eines article-Dokuments:

```
> \newtheorem{defin}{Definition}[section]  % Präambel
  \section{Überschrift}  % Dokument
   \begin{defin} Eine Zahl $a>0$ heißt positiv. \end{defin}
   \begin{defin} Eine Zahl $a<0$ heißt negativ. \end{defin}
  \section{Überschrift}
   \begin{defin} Eine Zahl $a\geq0$ heißt nicht-negativ. \end{defin}
   \begin{defin} Eine Zahl $a\leq0$ heißt nicht-positiv. \end{defin}
```

= **1 Überschrift**

Definition 1.1 *Eine Zahl $a > 0$ heißt positiv.*

Definition 1.2 *Eine Zahl $a < 0$ heißt negativ.*

2 Überschrift

Definition 2.1 *Eine Zahl $a \geq 0$ heißt nicht-negativ.*

Definition 2.2 *Eine Zahl $a \leq 0$ heißt nicht-positiv.*

Im Beispiel definiert der Befehl `\newtheorem` eine Umgebung mit dem Namen `defin`. „Definition“ ist der Text, der in fett als Einleitung gesetzt wird. In diesen Theorem-Umgebungen wird der folgende Fließtext standardmäßig kursiv gesetzt. Das dritte (optionale) Argument gibt an, dass sich die Nummerierung an `\section` orientieren soll. Das bedeutet, dass an die Nummerierung, die zum letzten Mal von `\section` erzeugt wurde, die Nummer der Definition angehängt wird. Lässt man das optionale Argument weg, dann erhält die Definition lediglich eine eigene Nummer. Im obigen Beispiel wären dann „Definition 1“ bis „Definition 4“ entstanden.

Fügt man in der Präambel

```
> \newtheorem{satz}{Satz}[section]
```

hinzu, so wird eine Umgebung `satz` definiert, die unabhängig von den Definitionen nummeriert wird. Im Dokument wird es dann eine Definition 1.1 und einen Satz 1.1 geben. Wenn dies nicht erwünscht ist, kann man LaTeX sagen, dass die Nummerierung der Sätze und Definitionen zusammen geschehen soll:

```
> \newtheorem{satz}[defin]{Satz}
```

Dann verwenden die Sätze dieselbe Nummerierung wie die Definitionen; genauer gibt es nur einen Zähler, der gemeinsam verwendet wird. Man beachte die unterschiedlichen Positionen des optionalen Arguments.

Möchte man eine Umgebung haben, die gar nicht nummeriert wird, so kann man dazu die gesternte Version benutzen: \newtheorem*

```
> \newtheorem*{bemer}{Bemerkung}  % Präambel
  \begin{bemer} Dies ist nur ein Spezialfall. \end{bemer}  % Dokument
```

= **Bemerkung** *Dies ist nur ein Spezialfall.*

Ferner kann die Umgebung ein optionales Argument aufnehmen, das als Beschreibung in Klammern gesetzt wird:

```
>  \begin{defin}[positiv] Eine Zahl $a>0$ heißt positiv. \end{defin}
```

= **Definition 1.1 (positiv)** *Eine Zahl* $a > 0$ *heißt positiv.*

Wann immer man diese Theoreme benutzen möchte, sollte man das Paket ntheorem einbinden, auch wenn das obige Beispiel auch ohne funktioniert. Im Folgenden werden die Vorteile sichtbar werden.

5.6.1 Formatierungsbefehle

Das Paket ntheorem stellt u. a. die drei Befehle `\theoremstyle`, `\theoremheaderfont` und `\theorembodyfont` zur Verfügung; diese sollen im folgenden längeren Beispiel im book-Stil erläutert werden: \theoremstyle \theoremheaderfont \theorembodyfont

```
> \theoremstyle{changebreak}  % Präambel
  \theoremheaderfont{\sffamily\upshape\bfseries}
  \theorembodyfont{\upshape}
  \newtheorem{defin}{Definition}[section]
  \theorembodyfont{\itshape}
  \newtheorem{satz}[defin]{Satz}
  \newtheorem{hsatz}[defin]{Hauptsatz}
  \theorembodyfont{\upshape}
  \newtheorem{kor}[defin]{Korollar}
  \theoremstyle{plain}
  \theoremheaderfont{\normalfont\bfseries}
  \newtheorem{beisp}{Beispiel}[chapter]
  \chapter{Demo}  % Dokument        (Der Kapitelbeginn ist im Ergebnis nicht zu sehen.)
   \section{Überschrift}
    \begin{defin} Eine Zahl $a>0$ heißt positiv. \end{defin}
    \begin{satz}[Abgeschlossenheit der Addition]
     Aus $a,b>0$ folgt $a+b>0$. \end{satz}
    \begin{beisp} Es ist $3,4>0$ und $3+4=7>0$. \end{beisp}
    \begin{kor} Aus $a>0$ folgt $2a=a+a>0$. \end{kor}
   \section{Überschrift}
    \begin{hsatz}[Abgeschlossenheit der Multiplikation]
     Aus $a,b>0$ folgt $a\cdot b>0$. \end{hsatz}
    \begin{beisp} Aus $7,2>0$ folgt $7\cdot2=14>0$. \end{beisp}
    \begin{kor} Aus $a>0$ folgt $a^2=a\cdot a>0$. \end{kor}
    \begin{beisp} Es ist $5>0$ und $5^2=25>0$. \end{beisp}
```

= **1.1 Überschrift**

1.1.1 Definition
Eine Zahl $a > 0$ heißt positiv.

1.1.2 Satz (Abgeschlossenheit der Addition)
Aus $a, b > 0$ folgt $a + b > 0$.

Beispiel 1.1 Es ist $3, 4 > 0$ und $3 + 4 = 7 > 0$.

1.1.3 Korollar
Aus $a > 0$ folgt $2a = a + a > 0$.

1.2 Überschrift

1.2.1 Hauptsatz (Abgeschlossenheit der Multiplikation)
Aus $a, b > 0$ folgt $a \cdot b > 0$.

Beispiel 1.2 Aus $7, 2 > 0$ folgt $7 \cdot 2 = 14 > 0$.

1.2.2 Korollar
Aus $a > 0$ folgt $a^2 = a \cdot a > 0$.

Beispiel 1.3 Es ist $5 > 0$ und $5^2 = 25 > 0$.

Der Befehl `\theoremstyle` legt die Formatierung fest. Für das Argument stehen die folgenden Möglichkeiten zur Auswahl:

- `plain` (Standard): Sieht genauso aus wie ohne das Paket ntheorem.
- `change`: Vertauscht die Nummern mit der Überschrift.
- `margin`: Setzt die Nummer in den linken Rand.
- `break`, `changebreak` und `marginbreak`: Wie `plain`, `change` und `margin`, nur dass der jeweils folgende Text in einer neuen Zeile unter der Überschrift beginnt.

Der Befehl `\theoremheaderfont` legt die Schriftformatierung für die Überschriften, `\theorembodyfont` die für den Fließtext fest. Alle drei Befehle können mehrfach benutzt werden, um den verschiedenen Umgebungen unterschiedliche Formatierungen zu geben.

Vieles in diesem Beispiel ist stilistisch nicht nachahmenswert, sondern soll nur der Demonstration dienen. Die Beispiele sind vollkommen anders formatiert. Zunächst einmal haben sie einen eigenen Zähler, der sich am Kapitel orientiert statt am Abschnitt wie die anderen Theoreme. Das Wort „Beispiel" samt Nummer wird in der Serifenschrift gesetzt, und der Text beginnt in derselben Zeile hinter der Nummer. Dieser Fließtext ist wie bei den Definitionen und Korollaren, aber im Gegensatz zu Sätzen, aufrecht dargestellt. Die Tatsache, dass Definitionen und Sätze mit verschiedenen `\theorembodyfont`-Befehlen formatiert werden, ist hingegen durchaus üblich.

5.6.2 Unannehmlichkeiten

Leider produziert das ntheorem-Paket immer noch einiges, das furchtbar aussieht, und zusätzlich hängt das nach meiner Erfahrung auch noch von der Distribution bzw. vom Betriebssystem ab. Zum Beispiel stimmt im Modus `\theoremstyle{break}` der Abstand zu einer folgenden Zeile nicht unbedingt:

```
> \begin{defin}[Konvergenz einer Reihe]
   Es sei $\sum\limits_{k=1}^\infty a_k$ eine Reihe \dots
  \end{defin}
```

= **Definition 1 (Konvergenz einer Reihe)**
Es sei $\sum\limits_{k=1}^\infty a_k$ *eine Reihe ...*

Ein sehr hässlicher Ausweg ist z. B.

```
> \begin{defin}[Konvergenz einer Reihe] ~\\[-3ex]
   Es sei $\sum\limits_{k=1}^\infty a_k$ eine Reihe \dots
  \end{defin}
```

= **Definition 1 (Konvergenz einer Reihe)**
Es sei $\sum\limits_{k=1}^\infty a_k$ *eine Reihe ...*

Im Modus `\theoremstyle{plain}` beginnt eine Aufzählung leider in derselben Zeile, was zu folgendem hässlichen Ergebnis führt:

```
> \begin{defin}[Konvergenz einer Reihe]
   \begin{enumerate}
    \item Es sei $\sum\limits_{k=1}^\infty a_k$ eine Reihe \dots
    \item Sie heißt absolut konvergent, \dots
   \end{enumerate}
  \end{defin}
```

= **Definition 1 (Konvergenz einer Reihe)** *1. Es sei* $\sum\limits_{k=1}^\infty a_k$ *eine Reihe ...*

2. Sie heißt absolut konvergent, ...

Hier ist der Ausweg mit der Tilde allein aber einfacher:

```
> \begin{defin}[Konvergenz einer Reihe] ~
   \begin{enumerate}
    \item Es sei $\sum\limits_{k=1}^\infty a_k$ eine Reihe \dots
    \item Sie heißt absolut konvergent, \dots
   \end{enumerate}
  \end{defin}
```

= **Definition 1 (Konvergenz einer Reihe)**

1. *Es sei* $\sum\limits_{k=1}^\infty a_k$ *eine Reihe ...*
2. *Sie heißt absolut konvergent, ...*

Eine Alternative zur obigen hässlichen Lösung ist die folgende: Wir benutzen `\theoremstyle{plain}`, was mit der Tilde zu schönen Aufzählungen führt. Soll der Fließtext ebenfalls darunter beginnen, so schreiben wir:

```
> \begin{defin}[Konvergenz einer Reihe] ~\\
   Es sei $\sum\limits_{k=1}^\infty a_k$ eine Reihe \dots
  \end{defin}
```

= **Definition 1 (Konvergenz einer Reihe)**
Es sei $\sum\limits_{k=1}^\infty a_k$ *eine Reihe ...*

Hier wird der Abstand dann vernünftig gesetzt.

5.7 Das Paket mathtools

Das Paket mathtools ist im Vergleich zum amsmath-Paket relativ neu. Es ist entstanden, um einerseits Bugs zu beseitigen und andererseits mathematische Ergänzungen bereitzustellen. Beispielsweise verbessert es die Positionierung der geschweiften Klammern von `\overbrace` und `\underbrace`, die nur im Standardschriftgrad 10pt optimal gesetzt wurden. Insofern ist das Paket auch dann empfehlenswert, wenn die im Folgenden beschriebenen Erweiterungen nicht benötigt werden.

\xleftrightarrow \xLeftarrow \xRightarrow \xLeftrightarrow \xmapsto \xhookleftarrow \xhookrightarrow \xleftharpoonup \xleftharpoondown \xrightharpoonup \xrightharpoondown \xrightleftharpoons \xleftrightharpoons

Analog zu `\xleftarrow` und `\xrightarrow` werden andere dehnbare Pfeile definiert:

```
> \[ A \xleftrightarrow{a-b} B \xLeftarrow{b-c} C \xRightarrow{c-d} D
  \xLeftrightarrow{d-e} E \xmapsto{e-f} F \xhookleftarrow{f-g} G
  \xhookrightarrow{g-h} H \]
```

=
$$A \xleftrightarrow{a-b} B \xLeftarrow{b-c} C \xRightarrow{c-d} D \xLeftrightarrow{d-e} E \xmapsto{e-f} F \xhookleftarrow{f-g} G \xhookrightarrow{g-h} H$$

```
> \[ A \xleftharpoonup{a-b} B \xleftharpoondown{b-c} C \xrightharpoonup{c-d}
  D \xrightharpoondown{d-e} E \xrightleftharpoons{e-f} F
  \xleftrightharpoons{f-g} G \]
```

=
$$A \xleftharpoonup{a-b} B \xleftharpoondown{b-c} C \xrightharpoonup{c-d} D \xrightharpoondown{d-e} E \xrightleftharpoons{e-f} F \xleftrightharpoons{f-g} G$$

\overbracket \underbracket

Analog zu `\overbrace` und `\underbrace` sind `\overbracket` und `\underbracket` definiert:

```
> \[ \overbracket{0=0}^{\text{klar}} \quad
  \underbracket[0.05mm][2.5mm]{(a+1)^2-(a-1)^2}_{=4a} \]
```

=
$$\overbracket{0=0}^{\text{klar}} \quad \underbracket[0.05mm][2.5mm]{(a+1)^2-(a-1)^2}_{=4a}$$

Die beiden optionalen Argumente definieren dabei die Strichstärke und die Höhe der Klammer.

\splitfrac \dsplitfrac

Weiterhin werden die beiden folgenden Befehle eingeführt:

```
> \[ \frac{\splitfrac{ax^2 + by^2}{+ cx + dy + e}}{xy} \quad
  \frac{\splitdfrac{ax^2 + by^2}{+ cx + dy + e}}{xy} \]
```

=
$$\frac{\splitfrac{ax^2 + by^2}{+ cx + dy + e}}{xy} \quad \frac{\splitdfrac{ax^2 + by^2}{+ cx + dy + e}}{xy}$$

Natürlich kann man die Befehle auch im Nenner verwenden.

\mathllap \mathclap \mathrlap

Die Befehle `\mathllap`, `\mathrlap` und `\mathclap` sind Analoga zu `\llap`, `\rlap` und `\clap`, die aber in dieser Einführung noch nicht erklärt wurden. Sie setzen im mathematischen Modus Formelteile in eine Box der Breite 0, so dass sie mit anderen überlappen (können). Im Allgemeinen führt dies zu übereinandergedruckten Zeichen, aber es gibt auch viele sinnvolle Anwendungen, z. B.:

```
> \[ \omega = \sum_{\mathclap{1\leq i<k\leq n}} f_{ik} \,
  \mathrm{d}x_i \wedge \mathrm{d}x_k \]
```

=
$$\omega = \sum_{\mathclap{1\leq i<k\leq n}} f_{ik} \, \mathrm{d}x_i \wedge \mathrm{d}x_k$$

Durch `\mathclap` wird der Leerraum links und rechts vom Summenzeichen unterbunden. Das `c` bewirkt, dass der Index zentriert wird, `l` bzw. `r` hätte ihn links- bzw. rechtsbündig gesetzt.

\smashoperator

Ferner gibt es den Befehl `\smashoperator`, der in `_` und `^` automatisch `\mathclap` einsetzt:

```
> \[ \omega = \smashoperator[r]{\sum_{1\leq i<k\leq n}} f_{ik} \,
  \mathrm{d}x_i \wedge \mathrm{d}x_k \]
```

=
$$\omega = \sum_{1\leq i<k\leq n} f_{ik}\,\mathrm{d}x_i \wedge \mathrm{d}x_k$$

Das `r` als optionales Argument bewirkt, dass nur der Leerraum rechts vom Operator unterdrückt wird. Statt dessen sind noch `l` und `lr` möglich.

Für Klammern steht nun auch noch folgender Befehl zur Verfügung (am besten in der Präambel verwenden): \DeclarePairedDelimiter

```
> \DeclarePairedDelimiter{\ceil}{\lceil}{\rceil}
```

Diese Zeile definiert `\ceil` in der folgenden Weise:

```
> \[ \ceil{\frac n2} \quad \ceil*{\frac n2} \quad \ceil[\Big]{\frac n2} \]
```

=
$$\lceil\frac n2\rceil \quad \left\lceil\frac n2\right\rceil \quad \Big\lceil\frac n2\Big\rceil$$

Der „normale“ Befehl setzt die Klammern davor und dahinter, die bei `\DeclarePairedDelimiter` definiert wurden. Die gesternte Version benutzt zusätzlich `\left` und `\right`; zur manuellen Vergrößerung kann das optionale Argument `\big`, `\Big`, `\bigg` oder `\Bigg` sein.

Manchmal sehen zwei oder mehrere Indizes nebeneinander nicht gut aus, z. B. in \adjustlimits

```
> \[ \limsup_{n\to\infty} a_n = \lim_{n\to\infty} \sup_{k\geq n} \{a_k\} \]
```

=
$$\limsup_{n\to\infty} a_n = \lim_{n\to\infty} \sup_{k\geq n} \{a_k\}$$

Wer das beheben möchte, kann `\adjustlimits` verwenden:

```
> \[ \limsup_{n\to\infty} a_n =
  \adjustlimits \lim_{n\to\infty} \sup_{k\geq n} \{a_k\} \]
```

=
$$\limsup_{n\to\infty} a_n = \adjustlimits \lim_{n\to\infty} \sup_{k\geq n} \{a_k\}$$

Möglicherweise schadet auch ein kleiner zusätzlicher Abstand dazwischen nichts.

5.7.1 Ergänzungen zu bestimmten Befehlen

Zu allen Matrixumgebungen gibt es eine zusätzliche gesternte Form, die ein optionales Argument besitzt: pmatrix∗ bmatrix∗ Bmatrix∗ vmatrix∗ Vmatrix∗ matrix∗

```
> \[ \begin{pmatrix*}[r] 3 & -12 \\ -12 & 7 \end{pmatrix*} \]
```

=
$$\begin{pmatrix*}[r] 3 & -12 \\ -12 & 7 \end{pmatrix*}$$

Wie immer stehen `l` für links-, `r` für rechtsbündige und `c` (Standard) für zentrierte Ausrichtung.

Die Umgebung `dcases` ist eine Variante von `cases`, die die Alternativen mit `\displaystyle` setzt: dcases dcases∗

```
> \[ f(x) = \begin{dcases} \frac1x & \text{für } x \neq 0 \,, \\
  0 & \text{für } x = 0 \end{dcases}
```

=
$$f(x) = \begin{dcases} \frac1x & \text{für } x \neq 0 \,, \\ 0 & \text{für } x = 0 \end{dcases}$$

Die gesternte Version wendet zusätzlich `\text` auf die Einträge der rechten Spalte an:

```
> \[ f(x) = \begin{dcases*} \frac1x & falls der Ausdruck definiert ist, \\
  0 & sonst \end{dcases*} \]
```

=
$$f(x) = \begin{dcases*} \frac1x & falls der Ausdruck definiert ist, \\ 0 & sonst \end{dcases*}$$

\shortintertext

Es gibt einen Befehl `\shortintertext` als Alternative zu `\intertext`, der nicht so große Abstände davor und dahinter erzeugt. Die Syntax ist natürlich genauso dieselbe, vergleiche dazu also Seite 45.

multlined

Genauso, wie es `gathered` und `aligned` zu `gather` und `align` gibt, definiert das Paket die Umgebung `multlined` als Variante von `multline`:

```
> \begin{gather*}
    a = b \\ \begin{multlined} c \\ = d \end{multlined} \\ e = f
  \end{gather*}
=
```

$$\begin{gathered} a = b \\ \begin{multlined} c \\ = d \end{multlined} \\ e = f \end{gathered}$$

Das Paket führt auch Ergänzungen zur Formatierung von Formelnummern ein; siehe dazu aber Abschnitt 6.6.

Kapitel 6

LaTeX-Interna und Benutzerdefiniertes

6.1 Definition eigener Befehle und Umgebungen

Mit `\newcommand` kann man einen neuen Befehl definieren: \newcommand \ensuremath

```
> \newcommand{\eps}{\varepsilon}
  Sei $\eps>0$ gegeben.
```

= Sei $\varepsilon > 0$ gegeben.

```
> \newcommand{\dx}{\,\mathrm{d}x}
  \[ \int_a^b f(x) \dx \]
```

=
$$\int_a^b f(x)\,\mathrm{d}x$$

```
> \newcommand{\KrDa}{\ensuremath{\clubsuit\text{D}}}
  Die beiden Spieler mit \KrDa\ spielen zusammen.
```

= Die beiden Spieler mit ♣D spielen zusammen.

Mit `\ensuremath` (Paket `amsmath`) stellt man sicher, dass etwas im Mathematikmodus gesetzt wird, und daher kann `\KrDa` im Text- und Mathematikmodus verwendet werden.

In der Regel stehen die Befehlsdefinitionen in der Präambel. Falls nicht, dann sind sie lokal, d. h. sie gelten nur für die aktuelle Umgebung. Daher gibt

```
> {\newcommand{\xyz}{xyz} \xyz} \xyz
```

= xyz

nur ein „xyz“ aus, aber auch den Fehler „`Undefined control sequence`“ beim zweiten `\xyz`. Entsprechendes gilt für die folgenden Befehle.

Für die Namen neu definierter Befehle gilt dasselbe für bestehende Befehle: Dem Backslash folgt ein beliebiges druckbares Zeichen oder eine Buchstabenfolge. Beispielsweise könnte man `\?` definieren, allerdings sollte man es vermeiden, Befehlsnamen zu vergeben, die aus nur einem Zeichen bestehen. Solche Namen kollidieren zu schnell mit LaTeX-eigenen Befehlen.

Soll der neue Befehl Argumente erhalten, sieht die Definition folgendermaßen aus: #1

```
> \newcommand{\intdx}[1]{\int #1 \,\mathrm{d}x}
  \[ \intdx{f(x)} \]
```

=
$$\int f(x)\,\mathrm{d}x$$

```
> \newcommand{\fracpart}[2]{\frac{\partial #1}{\partial #2}}
  \[ \fracpart Ex \]
```

=
$$\frac{\partial E}{\partial x}$$

Dabei gibt die Zahl in eckigen Klammern die Anzahl der Argumente an, und für `#1`, `#2` usw. werden die darin übergebenen Werte eingesetzt. Ein Befehl kann maximal neun Argumente erhalten.

\newenvironment Auf dieselbe Art und Weise können eigene Umgebungen definiert werden:

```
> \newenvironment{anfende}{\par\hrulefill\vspace{-1ex}\par Anfang\par}
   {\par\hspace*{\fill}Ende\par\vspace{-2ex}\hrulefill\par}
  \begin{anfende} xyz \end{anfende}
```

=

Anfang

xyz

Ende

`\newenvironment` benötigt drei Argumente, nämlich den Namen und die beiden Code-Teile, die bei `\begin` und `\end` eingefügt werden.

Man kann auch Befehle mit optionalen Argumenten selbst erstellen:

```
> \newcommand{\xseq}[1][x]{#1_1,\ldots,#1_n}
  $\xseq \quad \xseq[y] \quad \xseq[(u+v)]$
```

= $x_1,\ldots,x_n \quad y_1,\ldots,y_n \quad (u+v)_1,\ldots,(u+v)_n$

Die zweite eckige Klammer bewirkt, dass LaTeX das Argument als optional ansieht und bei dessen Fehlen im Aufruf den Inhalt dieser zweiten Klammer als Standardwert verwendet. Es kann jedoch nur das erste Argument als optional definiert werden.

##1 Manchmal ist es nötig, innerhalb einer Befehls-/Umgebungsdefinition einen weiteren Befehl zu definieren. Ein Beispiel dazu:

```
> \newenvironment{anschrift}[1]{%
   \begin{minipage}{#1}%
   \newcommand{\strasse}[1]{\textbf{Straße: ##1}}%
  }{\end{minipage}}
```

Auf die Argumente, die der innen definierte Befehl erhält, greift man mit `##1`, `##2` usw. zurück. Ein solcher Befehl existiert dann nur innerhalb der Umgebung `anschrift`.

\renewcommand Mit `\renewcommand` kann ein schon bestehender Befehl überschrieben werden:

```
> \renewcommand{\binom}[2]{\left[ \frac{#1}{#2} \right]}
  \[ \binom{2n}{3k} \]
```

=
$$\left[\frac{2n}{3k} \right]$$

Es ist mit `\renewcommand` nicht möglich, in der Neudefinition eines Befehls den Befehl in seiner alten Definition wiederzuverwenden. Man erhält in diesem Fall eine Stack-Overflow-Fehlermeldung in der Form „`TeX capacity exceeded, sorry [input stack size=xxx]`“, wobei `xxx` die momentane Stackgröße darstellt. Hingegen ist es möglich, die Anzahl der Argumente zu verändern.

\providecommand Wird `\renewcommand` auf einen Befehl angewendet, den es noch nicht gibt, wird ein Fehler ausgegeben. Wenn man nicht weiß, ob es einen Befehl schon gibt, dann kann man ihn mit `\providecommand` zur Verfügung stellen. Allerdings sollte man `\providecommand` nur mit äußerster Sorgfalt verwenden, damit nichts Unerwünschtes passiert.

\renewenvironment Auch die Implementation von Umgebungen kann modifiziert werden. Möchte man z. B. die Einträge in Matrizen, die mit `pmatrix` erzeugt werden, immer rechtsbündig (statt zentriert) haben, so kann man Folgendes definieren:

```
> \renewenvironment{pmatrix}{\left( \! \begin{array}{rrrrrrrrr}}{\end{array}
   \! \right)}
```

```
\[ \begin{pmatrix} -1 & 3 \\ 4 & -2 \end{pmatrix} \]
```

= $$\begin{pmatrix} -1 & 3 \\ 4 & -2 \end{pmatrix}$$

Allerdings sind die Abstände in dieser Matrix nun ein wenig größer, weil die Umgebung `pmatrix` anders definiert ist. (Und eigentlich sollte man hierfür `pmatrix*` aus mathtools benutzen.)

Statt der veralteten Befehle `\operatorname` und `\operatorname*` verwendet man nun: \Declare-MathOperator \Declare-MathOperator*

```
> \DeclareMathOperator{\Bild}{Bild}
  \DeclareMathOperator*{\esssup}{ess\,sup}
  \[ \Bild f \quad \esssup_{x\in\mathbbm R} \{f(x)\} \]
```

= $$\operatorname{Bild} f \quad \operatorname*{ess\,sup}_{x\in\mathbb{R}} \{f(x)\}$$

Diese beiden Befehle können allerdings nur in der Präambel verwendet werden.

Da es keinen Befehl „RedeclareMathOperator“ o. ä. gibt, muss man weiterhin auf die beiden veralteten Befehle zurückgreifen, wenn man eine Definition abändern will, z. B.: \operatorname

```
> \renewcommand{\Re}{\operatorname{Re}}
  $\Re z$
```

= $\operatorname{Re} z$

6.2 Längen

LATEX verwaltet neben Befehlen und Umgebungen auch Längen, die im Quelltext genauso wie Befehle aussehen, d. h. mit vorangestelltem Backslash. Zwei Beispiele sind `\parindent` und `\parskip`. Sie geben an, um wieviel die erste Zeile eines jeden Abstatzes eingerückt und wieviel zusätzlicher Abstand zwischen zwei Absätzen eingeschoben wird. Es gibt zwei wesentliche Befehle zur Manipulation von Längen: \setlength \addtolength

```
> \setlength{\parindent}{0em}
  \addtolength{\parskip}{1ex}
```

Der erste Befehl setzt den Absatzeinzug auf 0, der zweite vergrößert den Abstand zwischen zwei Absätzen um 1ex. Grundsätzlich muss auch eine Länge 0 eine Einheit bekommen. Längen können auch negative Werte annehmen, insbesondere darf das zweite Argument von `\addtolength` negativ sein, um eine Länge zu verkleinern. Wie die Befehle `\newcommand` usw. sind auch die Auswirkungen von `\setlength` und `\addtolength` lokal, d. h. sie gelten nur bis zum Ende der aktuellen Umgebung.

Wenn man den Absatzeinzug abstellt, benötigt man unbedingt einen Abstand zwischen den Absätzen, aber die obige Variante bringt einige Schwierigkeiten mit sich. Beispielsweise erhalten auch die Aufzählungspunkte in Listen diesen Zusatzabstand, was unschön aussieht. Zusätzliche Informationen dazu gibt es in Abschnitt 7.3.

Es gibt eine ganze Reihe Längen, die die Anordnung des Textes einschließlich der Kopf- und Fußzeilen auf dem Papier steuern. Horizontal sind dies im Wesentlichen drei:

- `\oddsidemargin`: linker Rand der ungeraden, d. h. rechten Seiten, bei einseitigem Druck aller Seiten; genauer: der Abstand vom linken Papierrand zum linken Rand des Textkörpers minus ein Zoll \oddsidemargin \evensidemargin \textwidth
- `\evensidemargin`: bei doppelseitigem Druck linker Rand der geraden, d. h. linken Seiten; genauer: der Abstand vom linken Papierrand zum linken Rand des Textkörpers minus ein Zoll

- `\textwidth`: Breite des Textes; genauer: der Abstand vom linken zum rechten Rand des Textkörpers

Vertikal sind ein paar mehr Längen wichtig:

\topmargin \headheight \headsep \textheight \footskip

- `\topmargin`: oberer Rand; genauer: der Abstand vom oberen Papierrand zum oberen Rand der Kopfzeile minus ein Zoll
- `\headheight`: Höhe der Kopfzeile; genauer: der Abstand vom oberen zum unteren Rand der Kopfzeile
- `\headsep`: Abstand zwischen Kopfzeile und Text; genauer: der Abstand vom unteren Rand der Kopfzeile zum oberen Rand des Textkörpers
- `\textheight`: Höhe des Textes; genauer: der Abstand vom oberen zum unteren Rand des Textkörpers
- `\footskip`: Abstand zwischen Text und Fußzeile; genauer: der Abstand vom unteren Rand des Textkörpers zum unteren (!) Rand der Fußzeile

\hoffset \voffset

Ferner gibt es die beiden Längen `\hoffset` und `\voffset`, die beide standardmäßig auf null stehen. Weist man ihnen andere Werte zu, so wird das gesamte Layout nach rechts bzw. unten verschoben. Insbesondere kann man beide auf `-1in` stellen, um den standardmäßigen Offset von einem Zoll zu kompensieren.

\marginsize

Das veraltete Paket `a4wide`, das alleine durch Einbinden die ziemlich großen Seitenränder verkleinert, wurde schon angesprochen. Weiterhin bietet `anysize` eine einfache Möglichkeit, die Seitenränder einzustellen:

```
> \marginsize{3cm}{2cm}{2.5cm}{2.5cm}
```

Die vier Argumente beschreiben in der Reihenfolge den linken, rechten, oberen und unteren Rand der Seite. Bei doppelseitigem Druck wird bei allen geraden Seiten der linke mit dem rechten Rand vertauscht; die beiden Argumente sollte man dann eher als inneren und äußeren Rand bezeichnen. Die übergebenen Ränder sind die echten Werte ohne den oben genannten Offset von einem Zoll.

\newlength \settowidth \settoheight \settodepth

Um neue Längen anzulegen, dient der Befehl `\newlength`:

```
> \newlength{\xyzwidth}
  \settowidth{\xyzwidth}{xyz}
  xyz abc \\
  \hspace*{\xyzwidth} abc
```

= xyz abc
      abc

Der Befehl `\settowidth` misst die Breite des zweiten Arguments aus und weist dieses Maß der Länge im ersten Argument zu. Analog funktionieren `\settoheight` für die Höhe von der Grundlinie aus bis zum oberen Rand und `\settodepth` für die Tiefe von der Grundlinie aus bis zum unteren Rand.

Einige Längen, häufig vor allem solche, die als vertikale Abstände benutzt werden, sind elastisch. Sie besitzen ein „normales“ Maß, das unter bestimmen Umständen höchstens um einen angegebenen Betrag verlängert oder verkürzt werden kann. Ein Beispiel für die Zuweisung eines elastischen Maßes folgt im nächsten Absatz.

\smallskipamount \medskipamount \bigskipamount

Die drei bekannten Befehle `\smallskip`, `\medskip` und `\bigskip` greifen intern auf den Befehl `\vspace` mit einer bestimmten Länge zurück. Diese Längen sind elastisch und heißen `\smallskipamount`, `\medskipamount` und `\bigskipamount`. Die kürzeste der drei Längen ist beispielsweise mit

```
> \setlength{\smallskipamout}{3pt plus 1pt minus 1pt}
```
vordefiniert.

6.3 Boxen

Eines der Grundkonzepte von LATEX sind die sog. Boxen, die Teile der auszugebenden Seite enthalten. Der Algorithmus, eine Seite mit Text zu füllen, beginnt damit, Boxen um einzelne Buchstaben herum zu erzeugen, um diese dann zu Wörtern und schließlich zu Zeilen „zusammenzukleben“. Aus den Zeilen entstehen dann die Absätze, aus den Absätzen die Seiten.

Mit `\makebox` wird, wie mit `\mbox`, eine Box erzeugt, die nicht umgebrochen werden darf: \makebox

```
> aaa \makebox{bbb} ccc
```
= aaa bbb ccc

Der Unterschied ist, dass `\makebox` zwei optionale Argumente erhalten kann, die die Breite der Box und die horizontale Ausrichtung des darin befindlichen Textes bestimmen:

```
> Dies \makebox[7cm][s]{hier ist ein toller Test für die} Box.
```
= Dies hier ist ein toller Test für die Box.

Das zweite Argument kann `l` für linksbündig, `c` für zentriert (Standard), `r` für rechtsbündig und `s` für blocksatzartig (d. h. gedehnt) sein. Letzteres funktioniert nur, wenn sich elastische Abstände wie z. B. Leerzeichen in der Box befinden.

Die beiden Befehle `\fbox` bzw. `\framebox` arbeiten jeweils genauso wie `\mbox` bzw. `\makebox`, nur dass sie die Box umrahmen: \fbox \framebox \fboxrule \fboxsep

```
> aaa \fbox{bbb} ccc
```
= aaa $\boxed{\text{bbb}}$ ccc

Beide Befehle benutzen die Länge `\fboxrule` als Dicke des Rahmens und `\fboxsep` als Abstand des Rahmens von der Schrift.

Um Schrift von der Grundlinie anzuheben oder abzusenken, dient der Befehl `\raisebox`: \raisebox

```
> Dieser \raisebox{0.5ex}{Text} verschwindet \raisebox{-0.2ex}{na}%
  \raisebox{-0.4ex}{ch} \raisebox{-0.6ex}{u}\raisebox{-0.8ex}{nt}%
  \raisebox{-1ex}{e}\raisebox{-1.2ex}{n.}
```
= Dieser Text verschwindet nach unten.

Wie man sieht, darf das Maß, um das nach oben verschoben werden soll, auch negativ sein.

Auch `\raisebox` kann zwei optionale Argumente erhalten:

```
> aaa aaa \\
  bbb \raisebox{0.5ex}[4ex][3ex]{bbb} \\
  ccc ccc
```
= aaa aaa

bbb bbb

ccc ccc

Die erste optionale Länge teilt LATEX mit, wie weit der Inhalt von der Grundlinie aus nach oben reicht, der zweite dementsprechend wie weit nach unten. Im Beispiel sind diese Längen deutlich größer als sie sein müssten, so dass Leerraum nach oben und unten erzeugt wird.

Das Paket graphicx stellt den Befehl `\rotatebox` zur Verfügung, um Objekte um einen beliebigen Winkel zu drehen: \rotatebox

```
> \rotatebox[origin=c]{30}{Das} \rotatebox[origin=c]{90}{ist}
  \rotatebox[origin=c]{180}{toll!}
```

= Das ist toll!

Dabei bewirkt der Zusatz `origin=c`, dass das Objekt um seinen Mittelpunkt gedreht wird. *Achtung:* Einige DVI-Viewer zeigen den gedrehten Text nicht korrekt an.

\parbox Ein wichtiger Befehl ist `\parbox`, der eine sog. Absatzbox mit angegebener Breite erzeugt:

```
> aaa \parbox{1cm}{xxx \\ xxx \\ xxx \\ xxx}
  bbb \parbox[t]{1cm}{xxx \\ xxx \\ xxx \\ xxx}
  ccc \parbox[c]{1cm}{xxx \\ xxx \\ xxx \\ xxx}
  ddd \parbox[b]{1cm}{xxx \\ xxx \\ xxx \\ xxx}
  eee
```

```
                                  xxx
                                  xxx
      xxx                xxx      xxx
      xxx                xxx
= aaa       bbb xxx  ccc       ddd xxx   eee
      xxx                xxx
      xxx       xxx      xxx
                xxx
                xxx
```

Das erste Argument gibt die Breite der Box an, das zweite ihren Inhalt. Das optionale Argument, das `t` für „top“, `c` für „center“ oder `b` für „bottom“ sein kann, gibt die Ausrichtung der Box zum Rest der Zeile an. Die Standardeinstellung ist `c`.

Der Befehl besitzt noch zwei optionale Argumente, nämlich eine manuelle Höhe der Box sowie die Ausrichtung des Textes in ihr:

```
> aaa \parbox[c][1cm][t]{1cm}{xxx}
  bbb \parbox[c][1cm][b]{1cm}{xxx}
  ccc \parbox[c][1cm][c]{1cm}{xxx}
  ddd \parbox[c][1cm][c]{1cm}{xxx \\ xxx}
  eee
```

```
      xxx                         xxx
= aaa       bbb      ccc xxx   ddd       eee
                                  xxx
                xxx
```

Alle Boxen sind 1 cm hoch, aber der Text in ihnen wird jeweils unterschiedlich ausgerichtet.

minipage In gewisser Hinsicht sind `\parbox`en eingeschränkt, z. B. funktionieren Fußnoten nicht wie normal. Für solche (und andere) Fälle benutzt man besser die Umgebung `minipage`. Diese Umgebung stellt ein kleines Pendant einer Seite dar – nur ohne Kopf- und Fußzeile:

```
> \begin{minipage}{6cm} Physiker: \\
   Albert Einstein\footnote{gelebt 1879--1955}
  \end{minipage}
  \hfill
  \begin{minipage}{6cm} Chemiker: \\
   Pierre Curie\footnote{gelebt 1859--1906}
  \end{minipage}
```

= Physiker:
Albert Einstein[a]

[a]gelebt 1879–1955

Chemiker:
Pierre Curie[a]

[a]gelebt 1859–1906

Jede Minipage nummeriert die Fußnoten unabhängig von der eigentlichen Seite, und zwar standardmäßig mit kursiven Kleinbuchstaben.

Genauso wie die `\parbox` besitzt die `minipage` ein optionales Argument für die vertikale Ausrichtung, das neben dem standardmäßigen `c` für zentriert auch `t` für „top" oder `b` für „bottom" sein kann.

Außer den Befehlen und Längen gibt es noch weitere Objekte, die mit einem Backslash beginnen, nämlich sog. Boxregister. Diese können mit `\newsavebox` angelegt, mit `\sbox` oder `\savebox` gefüllt und mit `\usebox` ausgegeben werden. Die Argumente von `\sbox` und `\savebox` sind dieselben wie die von `\mbox` und `\makebox`, nur dass das erste Argument zusätzlich das Boxregister angibt. \newsavebox \sbox \savebox \usebox

```
> \newsavebox{\xyzbox}
  \sbox{\xyzbox}{xyz}\usebox{\xyzbox} \\
  {\sbox{\xyzbox}{abc}\usebox{\xyzbox}} \\
  \usebox{\xyzbox}
```

= xyz
abc
xyz

Auch hier – genauso wie bei `\renewcommand` – ist die Zuweisung an das Boxregister lokal, d. h. sie wirkt nur bis zum Ende der aktuellen Umgebung. Im Beispiel ist das die anonyme Umgebung in der zweiten Zeile.

Ein wesentlicher Unterschied ist jedoch, dass beim Füllen eines Boxregisters der Inhalt tatsächlich sofort ausgeführt wird. Man vergleiche dazu

```
> {\newcommand{\test}{$\alpha$}\test \renewcommand{\alpha}{neu}\test}
```

= αneu

mit

```
> {\newsavebox{\test} \sbox{\test}{$\alpha$}\usebox{\test}
  \renewcommand{\alpha}{neu}\usebox{\test}}
```

= $\alpha\alpha$

6.4 Zähler und benutzerdefinierte Nummerierungen I

page part chapter section subsection subsubsection paragraph subparagraph enumi enumii enumiii enumiv footnote mpfootnote equation

Ein wichtiges Konzept in LaTeX sind Zähler. Sie werden benutzt, um die Kapitel und Abschnitte oder auch die verschiedenen Stufen der `enumerate`-Umgebung zu nummerieren; ebenso gibt es einen Zähler für die Seitennummer. Zähler werden – im Gegensatz zu Befehlen und Längen – ohne Backslash geschrieben. Sie können nur ganzzahlige (allerdings auch negative) Werte annehmen, und ihr Wertebereich ist gemessen an den Anforderungen fürs Nummerieren beliebig groß. Genauer liegt der Wert eines Zählers im vorzeichenbehafteten Vierbyte-Bereich, d. h. es gilt $-2\,147\,483\,647 \leq z \leq 2\,147\,483\,647$ für jeden Zähler z.

Die Standardzähler lauten `page` für die Seite, `part`, `chapter`, `section`, `subsection`, `subsubsection`, `paragraph` und `subparagraph` für die Gliederungsstufen, `enumi`, `enumii`, `enumiii` und `enumiv` für die Stufen der nummerierten Listen, `footnote` für die Fußnoten, `mpfootnote` für die Fußnoten in `minipage`-Umgebungen und `equation` für die Gleichungen.

Zähler können nicht direkt ausgedruckt werden; zur Ausgabe werden einige Befehle benötigt, die ihn in eine druckbare Version übersetzen. Dazu entsprechende Beispiele: \arabic \alph \Alph \roman \Roman

```
> \arabic{footnote} \alph{chapter} \Alph{section} \roman{page} \Roman{chapter}
```

= 0 f D lxiii VI

Dabei formatieren `\arabic` den Zähler als arabische Ziffern, `\alph` als Kleinbuchstabe, `\Alph` als Großbuchstabe, `\roman` als römische Zahl mit Kleinbuchstaben und `\Roman` als römische Zahl mit Großbuchstaben. Dabei müssen formatierungsbedingte Grenzen eingehalten

\fnsymbol \textasteriskcentered \textdagger \textdaggerdbl \S \P \textbardbl \the

werden: zwischen 1 und 26 für die Buchstaben sowie positiv und nicht zu groß für die römische Darstellung. Eine weitere Formatierung kann mit `\fnsymbol` erzeugt werden, das eigentlich für Fußnoten gedacht ist. Aus dem Zähler, der zwischen 1 und 9 liegen muss, entstehen die Zeichen ∗ (`\textasteriskcentered`), † (`\textdagger`), ‡ (`\textdaggerdbl`), § (`\S`), ¶ (`\P`), ‖ (`\textbardbl`), ∗∗ (`\textasteriskcentered\textasteriskcentered`), †† (`\textdagger\textdagger`) und ‡‡ (`\textdaggerdbl\textdaggerdbl`).

Für jeden Zähler ist automatisch ein Befehl definiert, der eine formatierte Version des Zählers enthält. Man erhält diesen Befehl, indem man dem Zähler `\the` voranstellt. Er enthält gegebenenfalls mehr als nur eine der obigen Formatierungen, wie man im folgenden Beispiel sieht:

```
> \thesection
= 6.4
```

Im book-Stil sind sie folgendermaßen definiert:

```
> \renewcommand{\thechapter}{\arabic{chapter}}
  \renewcommand{\thesection}{\thechapter.\arabic{section}}
  \renewcommand{\thesubsection}{\thesection.\arabic{subsection}}
```

\newcounter

Jeder Zähler kann einen oder mehrere Rücksetzzähler besitzen. Sie bewirken, dass der eigentliche Zähler auf 0 zurückgesetzt wird, sobald einer der Rücksatzzähler inkrementiert wird. Dies findet offensichtlich bei den Gliederungsbefehlen Verwendung, weswegen sie im book-Stil folgendermaßen definiert sind:

```
> \newcounter{chapter}
  \newcounter{section}[chapter]
  \newcounter{subsection}[section]
```

Der Befehl `\newcounter` definiert einen neuen Zähler. Das optionale Argument am Ende gibt den Rücksetzzähler an.

\setcounter \addtocounter \stepcounter \refstepcounter \value

Um den Wert eines Zählers zu manipulieren, dienen folgende Befehle:

```
> \setcounter{page}{11} \addtocounter{section}{-3} \stepcounter{mpfootnote}
```

`\setcounter` setzt einen Zähler auf einen Wert, `\addtocounter` addiert einen Wert, der auch negativ sein darf, zum Zähler und `\stepcounter` inkrementiert einen Zähler. `\stepcounter` (und seine Variante `\refstepcounter`, die in Abschnitt 6.6 erläutert wird) bewirkt das oben genannte Rücksetzen aller Zähler, die den inkrementierten als Rücksetzzähler besitzen.

Neben den formatierten Ausgaben gibt es noch den Befehl `\value`, der den Wert eines Zählers zurückgibt. Eine mögliche Anwendung ist

```
> \newcounter{mypage}
  \setcounter{mypage}{\value{page}}
```

Dadudrch wird der eigene Zähler `mypage` auf den aktuellen Wert des Seitenzählers gesetzt.

\@addtoreset \makeatletter \makeatother

Schließlich gibt es noch Befehle, um Rücksetzzähler zu einem Zähler hinzuzufügen oder sie zu entfernen:

```
> \makeatletter
  \@addtoreset{footnote}{section}
  \makeatother
```

Nun wird der Fußnotenzähler bei jedem neuen Abschnitt auf 0 zurückgesetzt. (Im Übrigen könnte dies dazu führen, dass zwei Fußnoten auf derselben Seite dieselbe Nummer erhalten.) Das @-Zeichen zeigt an, dass es sich um einen internen Befehl handelt. Dies ist nicht weiter problematisch, nur muss der Bereich, in dem der Befehl benutzt wird, von `\makeatletter` und `\makeatother` umschlossen sein.

Mit dem Paket remreset kann man einen Rücksetzzähler entfernen: \@remove-fromreset

```
> \makeatletter
  \@removefromreset{footnote}{chapter}
  \makeatother
```

Nun beginnt die Zählung der Fußnoten nicht mehr in jedem Kapitel erneut bei 1.

Für die Gliederung des Dokuments sind zwei weitere Zähler von Interesse: `secnumdepth` bestimmt, bis zu welcher Tiefe Überschriften nummeriert werden sollen. Beispielsweise bewirkt (secnumdepth, tocdepth)

```
> \setcounter{secnumdepth}{1}
```

dass nur bis zu `\section` nummeriert wird; alle Überschriften ab `\subsection` erhalten keine Nummern mehr. Dabei steht 0 für `\chapter`, 1 für `\section`, 2 für `\subsection`, usw. Der Zähler `tocdepth` hingegen bestimmt, welche Tiefe das Inhaltsverzeichnis haben soll, z. B.

```
> \setcounter{tocdepth}{2}
```

Nun werden Überschriften bis einschließlich `\subsection` ins Inhaltsverzeichnis aufgenommen.

6.5 Benutzerdefinierte Nummerierungen II

Innerhalb der `itemize`- und `enumerate`-Umgebung leitet der Befehl `\item` einen neuen Punkt ein. Mit einem optionalen Argument kann man die Marke verändern: (itemize, enumerate, \item)

```
> \begin{enumerate}
   \item erstens
   \item[$*$] zweitens
   \item drittens
  \end{enumerate}
```

=
1. erstens

   $*$ zweitens

2. drittens

Insbesondere wird der Zähler bei einem Eintrag mit besonderer Marke nicht erhöht, weswegen der eigentlich dritte Eintrag mit „2.“ beschriftet wird.

Die verschiedenen Symbole der `itemize`-Umgebung werden, je nach Verschachtelungsgrad, von den Befehlen `\labelitemi`, `\labelitemii`, `\labelitemiii` und `\labelitemiv` erzeugt, z. B. kann man sie auch verändern: (\labelitemi, \labelitemii, \labelitemiii, \labelitemiv)

```
> \begin{itemize} \renewcommand{\labelitemi}{$\clubsuit$}
   \item erstens,
   \item zweitens
  \end{itemize}
  und
  \begin{itemize}
   \item drittens.
  \end{itemize}
```

=
- $\clubsuit$ erstens,
- $\clubsuit$ zweitens

und

- drittens.

\textbullet \textendash \textasteriskcentered \textperiodcentered \labelenumi \labelenumii \labelenumiii \labelenumiv

Da im Beispiel der Befehl `\labelitemi` innerhalb der `itemize`-Umgebung umdefiniert wurde, gilt er auch nur für diese Umgebung. In den meisten Fällen wird man eine solche Umdefinition in die Präambel schreiben, so dass sie für alle Aufzählungen dieser Stufe gültig ist. Die Standardeinstellungen für die vier Stufen sind • (`\textbullet`), **–** (`\bfseries\textendash`), ∗ (`\textasteriskcentered`) und · (`\textperiodcentered`).

Die Marken der `enumerate`-Umgebung werden in analoger Weise mit `\labelenumi`, `\labelenumii`, `\labelenumiii` und `\labelenumiv` erzeugt. Diese Befehle beinhalten eigentlich ausnahmslos die Formatierungen `\theenumi`, `\theenumii`, `\theenumiii` und `\theenumiv` der `enumerate`-Zähler `enumi`, `enumii`, `enumiii` und `enumiv`. Ihre Definitionen sind standardmäßig „`\thenumi.`“, „`(\theenumii)`“, „`\theenumiii.`“ und „`\theenumiv.`“. Zum Beispiel:

```
> \renewcommand{\labelenumi}{(\theenumi)}
  \renewcommand{\theenumii}{\Alph{enumii}}
  \renewcommand{\labelenumii}{\textbf{\theenumii:}}
  \begin{enumerate}
   \item erstens
    \begin{enumerate}
     \item zweitens
     \item drittens
    \end{enumerate}
  \end{enumerate}
```

= (1) erstens

  A: zweitens

  B: drittens

\part \chapter \section \subsection \subsubsection \paragraph \subparagraph \part∗ \chapter∗ \section∗ \subsection∗ \subsubsection∗ \paragraph∗ \subparagraph∗ \partname \chaptername \footnote

Die Gliederungsbefehle `\part`, `\chapter`, `\section`, `\subsection`, `\subsubsection`, `\paragraph` und `\subparagraph` können vor ihrem Pflichtargument ein optionales Argument (zwischen `[` und `]`) erhalten. Wird es angegeben, dann wird – statt dem Pflichtargument – sein Inhalt ins Inhaltsverzeichnis und in Kopf- oder Fußzeilen (je nach Einstellung) kopiert. Dies ist u. a. dann nötig, wenn im Pflichtargument manuelle Zeilenumbrüche oder Fußnoten stehen, z. B. bei

```
> \section[Einführung -- Eingabe von Text]{Einführung \\ Eingabe von Text}
```

oder

```
> \section[Abelsche Gruppen]{Abelsche\footnote{nach Niels Abel} Gruppen}
```

Alle diese Befehle besitzen auch eine ∗-Form, wodurch Nummerierung und Eintragen in das Inhaltsverzeichnis unterbunden werden. Bei `\part*` und `\chapter*` wird zusätzlich der Text „Teil“ und „Kapitel“, dem normalerweise die Nummer folgen würde, weggelassen.

Standardmäßig werden Teile und Kapitel mit „Part“ und „Chapter“ eingeleitet. Das Paket `(n)german` stellt die beiden Befehle `\partname` und `\chaptername` auf „Teil“ und „Kapitel“ um. Natürlich kann man sich des bekannten Befehls `\renewcommand` bedienen, um noch anderen Text ausgeben zu lassen.

Auch `\footnote` besitzt ein optionales Argument, das eine Zahl sein muss. Diese Zahl wird für die Nummerierung verwendet, und gleichzeitig wird der Fußnotenzähler nicht inkrementiert. Im folgenden Beispiel erhalten zwei aufeinanderfolgende Fußnoten dieselbe Nummer:

```
> Der Satz von
  Bolzano\footnote{nach Bernhard Bolzano, 1781--1848}-%
  Weierstraß\footnote[\value{footnote}]{nach Karl Weierstraß, 1815--1897}
  besagt, dass \dots
```

= Der Satz von Bolzano[1]-Weierstrass[1] besagt, dass ...

[1]nach Bernhard Bolzano, 1781–1848
[1]nach Karl Weierstraß, 1815–1897

Dabei wird mit `\value{footnote}` der aktuelle Wert des Fußnotenzählers erzeugt.

6.6 Querverweise

\label \ref \eqref \pageref

LaTeX unterstützt automatische Querverweise innerhalb eines Dokuments. Dabei kann auf alles referenziert werden, was von LaTeX eine Nummer erhält:

```
> \subsection{Querverweisbeispiel} \label{subsec:verwbsp}
  Zunächst zwei Formeln:
  \begin{align}
   \sin(x+y) &= \sin x\cos y + \cos x\sin y \,, \label{eq:sin} \\
   \cos(x+y) &= \cos x\cos y - \sin x\sin y \,. \label{eq:cos}
  \end{align}
  \begin{enumerate}
   \item \label{item:first} Hier im Unterabschnitt~\ref{subsec:verwbsp} auf
    Seite~\pageref{subsec:verwbsp} sind \eqref{eq:sin} und \eqref{eq:cos}
    die Additionstheoreme von Sinus und Cosinus.
   \item Fasst man Punkt~\ref{item:first} zusammen, so muss man
    (\ref{eq:sin}/\ref{eq:cos}) schreiben.
  \end{enumerate}
```

= **6.6.1 Querverweisbeispiel**

Zunächst zwei Formeln:

$$\sin(x+y) = \sin x \cos y + \cos x \sin y\,, \tag{6.1}$$
$$\cos(x+y) = \cos x \cos y - \sin x \sin y\,. \tag{6.2}$$

1. Hier im Unterabschnitt 6.6.1 auf Seite 67 sind (6.1) und (6.2) die Additionstheoreme von Sinus und Cosinus.
2. Fasst man Punkt 1 zusammen, so muss man (6.1/6.2) schreiben.

Dabei legt `\label` eine Marke an, d. h. verknüpft den angegebenen Bezeichner mit der Nummer des referenzierten Objekts. Der Befehl `\ref` gibt die Nummer des referenzierten Objekts aus, `\eqref` (Paket amsmath) diese Nummer in Klammern (gedacht für Gleichungen, daher „eq“) und `\pageref` die Nummer der Seite, auf der die Marke generiert wurde.

Es hat sich eingebürgert, dem Markenbezeichner ein Kürzel mit Doppelpunkt voranzustellen, das den Typ des referenzierten Objekts beschreibt. Wie im Beispiel kann dies `subsec` für einen Unterabschnitt, `eq` für eine Gleichung, o. ä. sein.

Beim Verwenden solcher Querverweise muss das Dokument zweimal compiliert werden. Hat man mit `\label` eine neue Marke erzeugt, so erhält man nach dem ersten Compilerdurchlauf eine Warnung „`Label(s) may have changed. Rerun to get cross-references right.`“. Bei diesem ersten Durchlauf wurden alle Referenzen in einer Datei TexDokument.aux gespeichert, so dass sie für den nächsten Durchlauf zu Verfügung stehen. Hat man hingegen `\ref` o. ä. mit einem unbekannten Bezeichner verwendet, so erhält man zwei Warnungen, nämlich eine der Form „`Reference ‘xxx’ on page yyy undefined on input line zzz.`“ und eine zweite „`There were undefined references.`“. Genauso kann es zur Warnung „`There were multiply-defined labels.`“ kommen, wenn man dieselbe Marke mehrfach vergibt.

\refstepcounter

Von `\refstepcounter` wird der übergebene Zähler genauso um 1 erhöht wie von `\stepcounter`. Zusätzlich kann allerdings mit `\label` direkt danach eine Marke erzeugt werden:

```
> \centerline{Aus der Differenzierbarkeit folgt die Stetigkeit.%
  \refstepcounter{equation}\label{eq:diff}\quad(\theequation)}
```

= Aus der Differenzierbarkeit folgt die Stetigkeit. (6.3)

Mit `\stepcounter` wäre dies nicht möglich gewesen.

\vref

Um Quelltexte der Form „`ref{eq:sin} auf Seite~\pageref{eq:sin}`“ zu vereinfachen, dient das Paket varioref, das für Dokumente in deutscher Sprache als

```
> \usepackage[(n)german]{varioref}
```

eingebunden werden muss. Es definiert den Befehl `\vref`:

```
> Aus Gleichung~\vref{eq:sin} folgt \dots
```

= Aus Gleichung 6.1 auf der vorherigen Seite folgt …

Normalerweise erzeugt es „`xxx auf Seite yyy`“ – allerdings auch nur dann, wenn sich zwischen `\label` und `\vref` mindestens eine Seite befindet. Ansonsten werden die Konstruktionen „`auf der vorherigen Seite`“ (siehe Beispiel) oder „`auf der folgenden Seite`“ ausgegeben, oder die Seitenangabe entfällt, weil sich Marke und Referenz auf derselben Seite befinden.

\newtagform
\usetagform

Das Paket mathtools stellt noch Befehle für Formelnummern zur Verfügung:

```
> \newtagform{bracetags}{\{}{\}}
  {\usetagform{bracetags} \begin{equation} a + b = c + d \end{equation}}
```

= $$a + b = c + d \qquad \{6.3\}$$

`\newtagform` definiert eine neues Formelnummerformat und erhält dazu erst den Namen, dann den Text links von der Formelnummer und dann den Text rechts von ihr. `\usetagform` schaltet ein bestimmtes Formelnummerformat ein; mit dem Argument `default` wird der Standard (runde Klammern) wiederhergestellt. Für den Fall, dass mehrere verschiedene Formelnummerformate verwendet werden, muss man beachten, dass bei `\eqref` das Format verwendet wird, das aktuell eingestellt ist, und nicht das, was bei der Definition der Gleichung aktiv war.

\mathtoolsset
showonlyrefs

Außerdem kann der Befehl

```
> \mathtoolsset{showonlyrefs=true}
```

aus dem Paket mathtools benutzt werden. Dieser bewirkt, dass Formelnummern nur dann ausgegeben werden, wenn auch auf sie referenziert wird. Dazu reicht ein `\label` nicht aus; vielmehr muss es tatsächlich ein `\eqref` geben, das einen entsprechenden Querverweis erzeugt. Auch die mit `\tag` und `\tag*` generierten manuellen Formelnummern werden ohne Referenz ausgeblendet. Mit der Zuweisung `false` werden wieder alle Formelnummern angezeigt. Auch wenn man diesen Befehl für gewöhnlich nur einmal in der Präambel stehen hat, kann man ihn auch mehrfach im Dokument verwenden.

6.7 Mathematische Interna

\smash

Manchmal kommt es zu uneinheitlicher Formatierung, wenn Ausdrücke mit unterschiedlichem Schriftbild unter Wurzeln u. ä. stehen, so wie in $\sqrt{a} = \sqrt{p}$. Um dies zu verhindern, unterdrückt der Befehl `\smash`, dass Rücksicht auf die Höhe oder Tiefe eines Objekts genommen wird:

```
> $\sqrt{a} = \sqrt{p} \quad \sqrt{a} = \sqrt{\smash[b]{p}}$
```

= $\sqrt{a} = \sqrt{p} \quad \sqrt{a} = \sqrt{\smash[b]{p}}$

Das Argument `b` steht für „bottom“, so dass die Unterlänge des p ignoriert wird, um die Größe und Position des Wurzelzeichens zu berechnen. `t` würde die Oberlänge ignorieren, was hier aber nicht angebracht ist. Um $\sqrt{a} = \sqrt{b}$ zu korrigieren, siehe `\vphantom` in Abschnitt 7.1.

`\frac` und `\binom` sowie deren Abarten werden intern durch den Befehl `\genfrac` (Paket `amsmath`) realisiert. Dieser Befehl erhält sechs Argumente in der folgenden Reihenfolge: Zeichen für die links begrenzende Klammer; Zeichen für die rechts begrenzende Klammer; Strichdicke (leer = Standardbruchstrich); Schriftgrad (0 für `\displaystyle`, 1 für `\textstyle`, usw.); Zähler; Nenner.

\frac
\binom
\genfrac

```
> \[ \genfrac{}{}{}{}{a+b}{x+y} \quad \genfrac{(}{)}{0pt}{1}{n+1}{k-1} \]
```

= $$\genfrac{}{}{}{}{a+b}{x+y} \quad \genfrac{(}{)}{0pt}{1}{n+1}{k-1}$$

Diese beiden Befehle entsprechen `\frac` und `\tbinom`.

Der veraltete Befehl `\atop` soll nicht mehr benutzt werden; statt dessen verwendet man nun Folgendes:

```
> \[ \genfrac{}{}{0pt}{}{xyz}{\alpha+\beta} \]
```

= $$\genfrac{}{}{0pt}{}{xyz}{\alpha+\beta}$$

Neben `\ldots` und `\cdots`, die nach der Position der generierten Punkte benannt sind, gibt es auch noch folgende Befehle:

\ldots
\cdots
\dotsc
\dotsb
\dotsm
\dotsi
\dotso

```
> \[ x_1,\dotsc,x_n \quad x_1+\dotsb+x_n \quad x_1\dotsm x_n \quad
  \int_{\Omega_1}\dotsi\int_{\Omega_n} \quad x_1\leq\dotso\leq x_n \]
```

= $$x_1,\dotsc,x_n \quad x_1+\dotsb+x_n \quad x_1\dotsm x_n \quad \int_{\Omega_1}\dotsi\int_{\Omega_n} \quad x_1\leq\dotso\leq x_n$$

Dabei erzeugen die Befehle in der Reihenfolge Punkte zwischen Kommata (`c`), Punkte zwischen binären Operatoren (`b`), Multiplikationspunkte (`m`), Punkte zwischen Integralen (`i`) und Punkte für alle anderen Stellen (`o`ther).

6.7.1 Symbolklassen

Einen Akzent, wie z. B. einen Punkt, kann man auch auf Verknüpfungssymbole setzen: \mathbin

```
> $A\cup B \quad A\dot\cup B$
```

= $A\cup B \quad A\dot\cup B$

Allerdings ist die Ausgabe wenig zufriedenstellend; die Abstände sind nicht korrekt, weil LaTeX das Objekt `\dot\cup` nicht mehr als Verknüpfungssymbol erkennt. `\mathbin` stellt die Eigenschaft, verknüpfendes Symbol zu sein, wieder her:

```
> \newcommand{\dotcup}{\mathbin{\dot\cup}}
  $A\cup B \quad A\dotcup B$
```

= $A\cup B \quad A\mathbin{\dot\cup} B$

Natürlich kann man auch ganz andere Symbole zu Verknüpfungssymbolen machen:

```
> \newcommand{\remain}{\mathbin{\%}}
  \newcommand{\xor}{\mathbin{\mathrm{xor}}}
  $a\remain b \quad A\xor B$
```

= $a \mathbin{\%} b \quad A \mathbin{\mathrm{xor}} B$

Ähnliches gilt für Vergleichssymbole und `\mathrel`: \mathrel

```
> \newcommand{\entspr}{\mathrel{\widehat=}}
  \newcommand{\unlhdO}{\mathrel{\unlhd_O}}
  $180^\circ\entspr\pi \quad N\unlhdO G \quad H\not\unlhdO G$
```

= $180^\circ \mathrel{\widehat=} \pi \quad N \mathrel{\unlhd_O} G \quad H \not\mathrel{\unlhd_O} G$

Auch wenn im zweiten Fall die Abstände ohne das `\mathrel` (fast) genauso aussehen, ist es immer vorteilhaft, das Zeichen mit `\mathrel` zu definieren.

Wer es ganz genau haben möchte, kann Folgendes schreiben:

```
> Falls $U\perp W$ gilt, so schreiben wir statt $V=U\oplus W$ auch
  $V=U\mathbin{\perp}W$.
```

= Falls $U \perp W$ gilt, so schreiben wir statt $V = U \oplus W$ auch $V = U \mathbin{\perp} W$.

Im Übrigen ist `\perp` nichts anderes als `\mathrel{\bot}`.

\mathord

Die dritte Symbolklasse ist `\mathord`, die ganz gewöhnliche Zeichen beschreibt:

\math-punct

```
> $a\mathord{+}b \quad a{+}b$
```

= $a\mathord{+}b \quad a{+}b$

Hierbei wird also der Leerraum entfernt. Das Setzen in geschweifte Klammern bewirkt dasselbe; vgl. dazu die Schreibweise `{,}` für das deutsche Dezimalkomma. Das Komma und das Semikolon werden von LaTeX mit `\mathpunct` gesetzt.

\math-open \math-close

Ferner kann man spezifizieren, dass ein Zeichen einen öffnenden oder schließenden Charakter besitzen soll. Auf Seite 36 haben wir „$x \in] -1, 1[$" in „$x \in \mathopen]-1,1\mathclose[$" korrigiert, indem wir `\left` und `\right` verwendet haben. Die noch bessere Variante macht von `\mathopen` und `\mathclose` Gebrauch:

```
> $x\in\mathopen] -1,1 \mathclose[$
```

= $x\in\mathopen] -1,1 \mathclose[$

Wer $<$ und $>$ verwenden will, um spitzere Klammern als mit `\langle` und `\rangle` zu erhalten, kommt um diese Befehle nicht herum:

```
> $<x>=<y> \quad \mathopen< x \mathclose> = \mathopen< y \mathclose>$
```

= $<x>=<y> \quad \mathopen< x \mathclose> = \mathopen< y \mathclose>$

\mathop

Die letzte Symbolklasse ist `\mathop`; hiermit werden alle großen Symbole wie z. B. `\sum` definiert. Dazu ein Beispiel:

```
> \[ \dot\bigcup_{n\geq0} A_n \quad \mathop{\dot\bigcup}_{n\geq0} A_n \]
```

=
$$\dot\bigcup_{n\geq0} A_n \quad \mathop{\dot\bigcup}_{n\geq0} A_n$$

Man beachte, dass es im ersten Fall nicht mit `\limits` möglich ist, etwas unter das Symbol zu schreiben.

| \| \lvert \rvert \lVert \rVert

Genaugenommen müsste man LaTeX auch bei jedem Auftreten von `|` und `\|` sagen, ob die „Klammer" an dieser Stelle öffnend oder schließend ist. Dafür gibt es aber schon vorgefertigte Befehle, die man aber wegen der verminderten Lesbarkeit des Quelltextes nur dann, wenn es nötig ist, benutzen sollte:

```
> $|-a| \quad \lvert-a\rvert \quad \|-b\| \quad \lVert-b\rVert$
```

= $|-a| \quad \lvert-a\rvert \quad \|-b\| \quad \lVert-b\rVert$

Diese Befehle befinden sich im Paket `amsmath`. Natürlich kann man sich auch z. B. mit

```
> \newcommand{\abs}[1]{\lvert #1 \rvert}
```

einen eigenen Betragsbefehl definieren. Die Befehle `\vert` und `\Vert` hingegen sind aber wirklich unnötig, weil sie nur einen Ersatz für `|` und `\|` darstellen. Analoges gilt für `\lbrack`, `\rbrack`, `\lbrace` und `\rbrace` statt `[`, `]`, `\{` und `\}`. Entscheiden muss man sich auch hier:

```
> $\|\cdot\| \quad \lVert\cdot\rVert$
```

= $\|\cdot\| \quad \lVert\cdot\rVert$

Kapitel 7

Zusätzliche Zeichen und Formatierungen

7.1 Zusätzliche Zeichen und Ausgabebefehle

Der Befehl `\today` druckt das aktuelle Datum aus: \today
`> \today`
= 7. April 2012
Dabei wird das Ausgabeformat von der eingestellten Sprache bestimmt.

Weitere Standardzeichen für den Textmodus:

```
> \dq\ \P\ \textasteriskcentered\ \textbardbl\ \textbullet\ \textdagger\
  \textdaggerdbl\ \textdollar\ \textemdash\ \textendash\ \textordfeminine\
  \textordmasculine\ \textparagraph\ \textperiodcentered\ \textsection
```

= " ¶ ∗ ‖ • † ‡ $ — – ª º ¶ · §
Viele dieser Symbole werden durch das Paket textcomp verändert, `\dq` befindet sich im Paket (n)german. `\textemdash` ist 1 em und `\textendash` 0,5 em lang; sie entsprechen `---` und `--`. Ebenso sind `\$`, `\P` und `\S` Abkürzungen für `\textdollar`, `\textparagraph` und `\textsection`.

Vor jede Art von Strich kann man `\nobreakdash` setzen: \nobreakdash
`> E\nobreakdash-Mail, Seiten 4\nobreakdash--7`
Dies verhindert einen Zeilenumbruch an diesem Strich. Meistens, vor allem bei Zahlen, ist es einfacher und im Quelltext übersichtlicher, wenn man den Ausdruck mit `\mbox` zusammenhält, also z. B. `\mbox{4--7}`. \mbox

Mit dem Befehl `\symbol` kann ein Zeichen mit einem bestimmten ASCII-Code ausgegeben werden. Im Folgenden sind diese Zeichen dargestellt: \symbol

0: `	9: ¯	18: „	27: ﬀ	36: $	45: -	54: 6	63: ?
1: ´	10: ˙	19: «	28: ﬁ	37: %	46: .	55: 7	64: @
2: ˆ	11: ¸	20: »	29: ﬂ	38: &	47: /	56: 8	65: A
3: ˜	12: ˛	21: –	30: ﬃ	39: ’	48: 0	57: 9	66: B
4: ¨	13: ‚	22: —	31: ﬄ	40: (	49: 1	58: :	67: C
5: ˝	14: ‹	23:	32: ␣	41:)	50: 2	59: ;	68: D
6: ˚	15: ›	24: ₒ	33: !	42: *	51: 3	60: <	69: E
7: ˇ	16: “	25: ı	34: "	43: +	52: 4	61: =	70: F
8: ˘	17: ”	26: ȷ	35: #	44: ,	53: 5	62: >	71: G

72: H	95: _	118: v	141: Ŋ	164: ď	187: ż	210: Ò	233: é
73: I	96: ‘	119: w	142: Ő	165: ě	188: ij	211: Ó	234: ê
74: J	97: a	120: x	143: Ŕ	166: ę	189: ¡	212: Ô	235: ë
75: K	98: b	121: y	144: Ř	167: ğ	190: ¿	213: Õ	236: ì
76: L	99: c	122: z	145: Ś	168: ĺ	191: £	214: Ö	237: í
77: M	100: d	123: {	146: Š	169: ľ	192: À	215: Œ	238: î
78: N	101: e	124: |	147: Ş	170: ł	193: Á	216: Ø	239: ï
79: O	102: f	125: }	148: Ť	171: ń	194: Â	217: Ù	240: ð
80: P	103: g	126: ˜	149: Ţ	172: ň	195: Ã	218: Ú	241: ñ
81: Q	104: h	127: -	150: Ű	173: ŋ	196: Ä	219: Û	242: ò
82: R	105: i	128: Ă	151: Ů	174: ő	197: Å	220: Ü	243: ó
83: S	106: j	129: Ą	152: Ÿ	175: ŕ	198: Æ	221: Ý	244: ô
84: T	107: k	130: Ć	153: Ź	176: ř	199: Ç	222: Þ	245: õ
85: U	108: l	131: Č	154: Ž	177: ś	200: È	223: SS	246: ö
86: V	109: m	132: Ď	155: Ż	178: š	201: É	224: à	247: œ
87: W	110: n	133: Ě	156: IJ	179: ş	202: Ê	225: á	248: ø
88: X	111: o	134: Ę	157: İ	180: ť	203: Ë	226: â	249: ù
89: Y	112: p	135: Ğ	158: đ	181: ţ	204: Ì	227: ã	250: ú
90: Z	113: q	136: Ĺ	159: §	182: ű	205: Í	228: ä	251: û
91: [	114: r	137: Ľ	160: ă	183: ů	206: Î	229: å	252: ü
92: \	115: s	138: Ł	161: ą	184: ÿ	207: Ï	230: æ	253: ý
93:]	116: t	139: Ń	162: ć	185: ź	208: Ð	231: ç	254: þ
94: ˆ	117: u	140: Ň	163: č	186: ž	209: Ñ	232: è	255: ß

Um alle 256 Zeichen darzustellen, ist der T1-Schriftsatz notwendig; mit OT1 lassen sich nur die Zeichen 0–127 ansprechen.

Im Paket textcomp befinden sich folgende Zeichen:

```
> \capitalacute{A} \capitalbreve{A} \capitalcaron{A} \capitalcedilla{A}
  \capitalcircumflex{A} \capitaldieresis{A} \capitaldotaccent{A}
  \capitalgrave{A} \capitalhungarumlaut{A} \capitalmacron{A} \capitalogonek{A}
  \capitalring{A} \capitaltilde{A}
```

= Á Ă Ǎ A̧ Â Ä Ȧ À A̋ Ā Ą Å Ã

```
> \textzerooldstyle\ \textoneoldstyle\ \texttwooldstyle\ \textthreeoldstyle\
  \textfouroldstyle\ \textfiveoldstyle\ \textsixoldstyle\ \textsevenoldstyle\
  \texteightoldstyle\ \textnineoldstyle
```

= 0 1 2 3 4 5 6 7 8 9

```
> \textacutedbl\ \textasciiacute\ \textasciibreve\ \textasciicaron\
  \textasciidieresis\ \textasciigrave\ \textasciimacron\ \textbrokenbar\
  \textdblhyphen\ \textgravedbl\ \textinterrobang\ \textinterrobangdown\
  \textquotesingle\ \textquotestraightbase\ \textquotestraightdblbase\
  \textthreequartersemdash\ \texttildelow\ \texttwelveudash
```

= ˝ ´ ˘ ˇ ¨ ` ¯ ¦ ⹀ ‶ ‽ ⸘ ' ‚ „ — ˷ —

```
> \textbaht\ \textcent\ \textcentoldstyle\ \textcolonmonetary\ \textcurrency\
  \textdollaroldstyle\ \textdong\ \textflorin\ \textguarani\ \textlira\
  \textnaira\ \textpeso\ \textwon
```

= ฿ ¢ ¢ ₡ ¤ $ ₫ ƒ ₲ £ ₦ ₱ ₩

```
> \textdiv\ \textdownarrow\ \textfractionsolidus\ \textlangle\ \textlbrackdbl\
  \textleftarrow\ \textlnot\ \textlquill\ \textminus\ \textonehalf\
```

```
  \textonequarter\ \textonesuperior\ \textopenbullet\ \textpm\ \textrangle\
  \textrbrackdbl\ \textrightarrow\ \textrquill\ \textsurd\ \textthreequarters\
  \textthreesuperior\ \texttimes\ \texttwosuperior\ \textuparrow
```
= ÷ ↓ / ⟨ ⟦ ← ¬ { − ½ ¼ ^{1} ◦ ± ⟩ ⟧ → } √ ¾ ^{3} × ^{2} ↑

```
> \textbigcircle\ \textblank\ \textborn\ \textcelsius\ \textcircledP\
  \textcopyleft\ \textdegree\ \textdied\ \textdiscount\ \textdivorced\
  \textestimated\ \textleaf\ \textmarried\ \textmho\ \textmusicalnote\
  \textnumero\ \textohm\ \textpertenthousand\ \textpilcrow\ \textrecipe\
  \textreferencemark\ \textservicemark
```
= ◯ ␢ ⋆ ℃ ℗ 🄯 ° † ⁒ ⚮ ℮ 🍃 ⚭ ℧ ♪ № Ω ‱ ¶ ℞ ※ ℠

Der Befehl `\rule` erzeugt ein ausgefülltes Rechteck: \rule

```
> \rule{2cm}{2mm} \rule{0.2pt}{3mm}\rule[1.5mm]{1in}{0.2pt}\rule{0.2pt}{3mm}
```
\hrule

=

Die beiden Argumente definieren Breite und Höhe; das optionale Argument gibt an, um wieviel das Rechteck nach oben geschoben werden soll. Der Befehl `\hrule` ohne Argumente erzeugt eine Linie über die gesamte Breite der Seite:

```
> xyz\hrule xyz
```
= xyz

xyz

Man kann Abstände erzeugen, die genau der Breite eines Textes entsprechen: \phantom

```
> eins \phantom{zwei} drei \phantom{vier} fünf sechs \\
  eins zwei \phantom{drei} vier \phantom{fünf} sechs
```
= eins   drei   fünf sechs
eins zwei   vier   sechs

Auf ähnliche Art und Weise arbeiten `\hphantom` bzw. `\vphantom`; hierbei wird lediglich der entsprechende horizontale bzw. vertikale Leerraum erzeugt: \hphantom \vphantom

```
> $\sqrt{a} = \sqrt{b}$ und $\sqrt{\vphantom{bp}a} = \sqrt{\vphantom{p}b}$
```
= $\sqrt{a} = \sqrt{b}$ und $\sqrt{\vphantom{bp}a} = \sqrt{\vphantom{p}b}$

7.2 Zusätzliche Schriftformatierung

Es ist auch außerhalb des Mathematikmodus möglich, hochgestellte Zeichen zu erzeugen, z. B.: \textsuperscript

```
> 30~cm$^3$ --- 30~cm\textsuperscript{3}
```
= 30 cm^3 — 30 cm^3

Die bereits eingeführten Befehle `\textnormal` bzw. `\normalfont` setzen die Schriftformatierung auf die Standardwerte `\textrm` bzw. `\rmfamily`, `\textup` bzw. `\upshape` und `\textmd` bzw. `\mdseries` zurück. Dadurch, dass man

```
> \renewcommand{\familydefault}{\sfdefault}
```
in der Präambel verwendet, kann man das gesamte Dokument in der serifenlosen Schrift setzen. Diese Variante ist besser als gleich nach `\begin{document}` mit `\sffamily` zu beginnen, da so `\textnormal` bzw. `\normalfont` auf den umgestellten Standard zurückgreifen. Die anderen Werte sind `\rmdefault` und `\ttdefault` für `\familydefault`, `\updefault`, `\itdefault`, `\sldefault` und `\scdefault` für `\shapedefault` sowie `\mddefault` und `\bfdefault` für `\seriesdefault`.

\familydefault \rmdefault \sfdefault \ttdefault \shapedefault \updefault \itdefault \sldefault \scdefault \seriesdefault \mddefault \bfdefault

Heutzutage ist es unüblich, Wörter zu unterstreichen, denn dies ist ein Relikt aus den Zeiten der Schreibmaschinen, als es keine andere Möglichkeit zur Texthervorhebung gab. Statt dessen sollte man besser fette Buchstaben (in Überschriften) oder kursive (im Fließtext) benutzen. Wer dennoch Wörter unterstreichen möchte, sollte unbedingt das Paket ulem einbinden. Zum einen definiert es den Befehl `\emph` um, so dass dieser unterstreicht, anstatt auf kursiv umzuschalten:

\emph

`> Nun wird \emph{unterstrichen} statt kursiv gesetzt.`

= Nun wird unterstrichen statt kursiv gesetzt.

\uline \uuline \uwave \sout \xout

Zum anderen definiert es folgende Befehle:

`> \uline{Schriftprobe} \uuline{Schriftprobe} \uwave{Schriftprobe}`

= Schriftprobe Schriftprobe Schriftprobe

`> \sout{Schriftprobe} \xout{Schriftprobe}`

= ~~Schriftprobe~~ ~~Schriftprobe~~

\ul \st \hl

Ein ähnliches Paket ist soul, wie man hier sieht:

`> \ul{Schriftprobe} \st{Schriftprobe} \hl{Schriftprobe}`

= Schriftprobe ~~Schriftprobe~~ Schriftprobe

Der farbige Hintergrund wird nur erzeugt, wenn das Paket color (vgl. Abschnitt ??) ebenfalls eingebunden ist; ansonsten unterstreicht `\hl` genauso wie `\ul`.

\so

Wirklich erwähnenswert wird das Paket allerdings durch die Fähigkeit, dass es Wörter gesperrt drucken kann:

`> Dies \so{sind drei gesperrte} Wörter. \so{Die {\LaTeX}-Einführung}`

= Dies s i n d d r e i g e s p e r r t e Wörter. D i e LATEX - E i n f ü h r u n g

Beim Sperren werden drei Sorten von Abständen unterschieden: zwischen zwei Buchstaben, zwischen zwei Wörtern, vor dem ersten bzw. nach dem letzten gesperrten Wort. Einige Befehle, z. B. `\LaTeX`, erzeugen Fehlermeldungen, wenn sie in `\so` benutzt werden; um sie oder anderen Inhalt, z. B. Ziffern, nicht zu sperren, kann man sie in `{` und `}` setzen.

\caps

Ein weiterer Befehl des Pakets in `\caps`:

`> GROSS \caps{GROSS} \textsc{Klein} \caps{Klein}`

= GROSS GROSS KLEIN KLEIN

Er setzt den Text als Kapitälchen und bewirkt eine minimale Sperrung bei Großbuchstaben, was das Lesen des Textes erleichtert.

\frenchspacing \nonfrenchspacing

Im englischsprachigen Schriftsatz ist es üblich, nach Interpunktionen an einem Satzende mehr Freiraum zu lassen als normalerweise zwischen den Wörtern oder nach einem Komma. Dies wird „Non French Spacing“ genannt – im Gegensatz zum „French Spacing“, bei dem alle Leerzeichen gleich groß sind, was im deutschen Schriftsatz üblich ist. Standardmäßig ist in LATEX Non French Spacing eingestellt. Das Paket (n)german hingegen stellt den Schriftsatz auf French Spacing um. Es ist allerdings auch erlaubt, manuell hin- und herzuschalten, und zwar an beliebiger Stelle jeweils mit `\frenchspacing` und `\nonfrenchspacing`.

\@

Im Modus Non French Spacing interpretiert LATEX genau die Punkte, die einem Kleinbuchstaben folgen, als Satzendepunkte. Deshalb ist die folgende Ausgabe nicht korrekt:

`> Bei d.\,h. gibt’s Probleme. Auch am Satzende bei der EU. Alles klar.`

= Bei d. h. gibt’s Probleme. Auch am Satzende bei der EU. Alles klar.

Der Befehl `\@` korrigiert die falsch gesetzten Leerabstände, und zwar: Beginnt nach Kleinbuchstabe und Punkt *kein* neuer Satz, so bewirkt der Befehl `\@` *hinter* dem Punkt, dass ein normaler Abstand erzeugt wird. Beginnt nach Großbuchstabe (oder Ziffer, . . .) und Punkt *ein* neuer Satz, so bewirkt der Befehl `\@` *vor* dem Punkt, dass ein größerer Abstand erzeugt wird:

`> Bei d.\,h.\@ gibt’s Probleme. Auch am Satzende bei der EU\@. Alles klar.`

= Bei d. h. gibt’s Probleme. Auch am Satzende bei der EU. Alles klar.

Die falsch gesetzten Leerräume wirken im Blocksatz deutlich schlimmer, weil die größeren Abstände zwischen den Sätzen stärker gedehnt werden. Vor allem bei den Abkürzungen sollte man auf die Korrektur der Leerräume achten. Am besten definiert man sich dazu eigene Befehle:

```
> \newcommand{\eg}{e.\,g.\@}
```

\linespread \baselinestretch

Selbstverständlich kann man auch den Zeilenabstand einstellen. Eine Möglichkeit stellt der Befehl `\linespread` dar. Beispielsweise verändern die Aufrufe `\linespread{0.9}`, `\linespread{1.5}` bzw. `\linespread{2}` den Zeilenabstand auf 90 %, 150 % bzw. 200 %. Die Änderung wird aber erst nach einem Schriftformatbefehl (z. B. `\Large`, `\bfseries`, `\normalfont`, ...) wirksam. Dasselbe bewirkt das Kommando `\renewcommand{\baselinestretch}{1.5}`; im Prinzip greift `\linespread` darauf zurück.

\setstretch \singlespacing \onehalfspacing \doublespacing \baselineskip

Durch `\linespread` wird der Zeilenabstand überall verändert, auch in Fußnoten und Ähnlichem, wo dies nicht erwünscht ist. Das Paket setspace treibt ein wenig Aufwand, um den Zeilenabstand in solchen Bereichen wieder auf 100 % zurückzustellen. Es definiert den Befehl `\setstretch`, der genauso wie `\linespread` benutzt wird. Speziell für einfachen, eineinhalbfachen und doppelten Zeilenabstand gibt es die argumentlosen Befehle `\singlespacing`, `\onehalfspacing` und `\doublespacing`.

Auf keinen Fall sollte man die Länge `\baselineskip` verstellen. Sie beschreibt den Abstand der Grundlinien zweier aufeinanderfolgender Zeilen und hängt natürlich von der Schriftart und vom Schriftgrad ab.

\lettrine

```
> \lettrine{In}{itia}len können mit dem Paket \textsf{lettrine} erzeugt
  werden; der Befehl besitzt denselben Namen. Das erste Argument gibt die
  Buchstaben an, die als Initial gesetzt werden sollen, das zweite diejenigen,
  die in Kapitälchen folgen.
```

= InITIAlen können mit dem Paket lettrine erzeugt werden; der Befehl besitzt denselben Namen. Das erste Argument gibt die Buchstaben an, die als Initial gesetzt werden sollen, das zweite diejenigen, die in Kapitälchen folgen.

Sinnvollerweise setzt man keinen oder alle Buchstaben des ersten Wortes nach dem Initial in Kapitälchen und üblicherweise auch nur einen Buchstaben als Initial. Mit einem optionalen Argument, z. B. `\lettrine[lines=3]{I}{}`, kann man die Anzahl der Zeilen bestimmen, die das Initial einnehmen soll.

7.3 Zusätzliche Absatzformatierung

Eine Möglichkeit für Zitate ist die folgende: quotation

```
> \begin{quotation}
   Die Umgebung \texttt{quotation} beginnt einen neuen Absatz, der links und
   rechts um ein bestimmtes Maß eingezogen ist. Außerdem wird die erste Zeile
   eingerückt, auch wenn man diese Option im Dokument ausgeschaltet hat.
  \end{quotation}
```

= Die Umgebung `quotation` beginnt einen neuen Absatz, der links und rechts um ein bestimmtes Maß eingezogen ist. Außerdem wird die erste Zeile eingerückt, auch wenn man diese Option im Dokument ausgeschaltet hat.

quote

```
> \begin{quote}
   Die Umgebung \texttt{quote} bewirkt dasselbe, nur dass der Absatzanfang
   niemals eingerückt wird. \par
```

```
  Statt dessen erscheint automatisch ein kleiner Abstand zwischen den
  Absätzen.
\end{quote}
```

= Die Umgebung `quote` bewirkt dasselbe, nur dass der Absatzanfang niemals eingerückt wird.

Statt dessen erscheint automatisch ein kleiner Abstand zwischen den Absätzen.

verse

```
> \begin{verse}
  Bei der Umgebung \texttt{verse} beginnt die erste Zeile jedes Absatzes
  weiter links als die übrigen Zeilen; \\
  ebenso sogar jede Zeile nach einem manuellen Umbruch mit
  \texttt{\textbackslash\textbackslash}.
\end{verse}
```

= Bei der Umgebung `verse` beginnt die erste Zeile jedes Absatzes weiter links als die übrigen Zeilen;
ebenso sogar jede Zeile nach einem manuellen Umbruch mit \\.

multicols

```
> \begin{multicols}{2}
  Mit dem Paket \textsf{multicol} können Textteile mehrspaltig formatiert
  werden. Der Inhalt wird dazu einfach in eine Umgebung
  \texttt{mul}\-\texttt{ticols} geschrieben. Das Argument gibt die Anzahl der
  Spalten an. Standardmäßig wird der Inhalt so verteilt, dass alle Spalten
  gleich lang sind. \par\columnbreak
  Es ist jedoch wie hier möglich, die Spalte manuell zu wechseln. Die
  Umgebung darf sich auch über mehrere Seiten erstrecken.
\end{multicols}
```

= Mit dem Paket multicol können Textteile mehrspaltig formatiert werden. Der Inhalt wird dazu einfach in eine Umgebung `multicols` geschrieben. Das Argument gibt die Anzahl der Spalten an. Standardmäßig wird der Inhalt so verteilt, dass alle Spalten gleich lang sind.

Es ist jedoch wie hier möglich, die Spalte manuell zu wechseln. Die Umgebung darf sich auch über mehrere Seiten erstrecken.

Für den mehrspaltigen Druck gibt es u. a. zwei Längen, und zwar `\columnsep` und `\columnseprule`. Die erste legt den Abstand zwischen je zwei Spalten, die zweite die Dicke der Trennlinie fest. Beide Längen werden auch von der `\documentclass`-Option `twocolumn` benutzt. Ein Beispiel:

```
> \setlength{\columnsep}{8mm}
  \setlength{\columnseprule}{0.25pt}
```

\verb

Der Befehl `\verb` setzt Text in der Typewriter-Schrift und ohne Interpretation der Sonderzeichen:

```
> \verb!Ein Beispiel  für \verb, mehrere    Leerzeichen und %, @, &.!
= Ein Beispiel  für \verb, mehrere    Leerzeichen und %, @, &.
```

Das Zeichen direkt nach `\verb` (hier `!`) dient als Markierungszeichen; tritt es zum zweiten Mal auf, dann wird der `\verb`-Befehl beendet. Gut geeignet sind dafür Zeichen wie `!`, `|`, `#`, usw. Im Argument von `\verb` kann kein Zeilenwechsel durchgeführt werden. Ferner kann der Befehl niemals im Argument eines anderen Befehls auftreten.

Um eine ganze Umgebung in dieser Weise zu setzen, kann man `verbatim` verwenden: verbatim \verb* verbatim*

```
> \begin{verbatim}
  for i := 1 to 100 do
    WriteLn(i);
  \end{verbatim}
= for i := 1 to 100 do
    WriteLn(i);
```

Bei zu langen Zeilen druckt LaTeX über den Rand, allerdings sind automatische Seitenwechsel möglich, d. h. die Umgebung kann sich über mehrere Seiten erstrecken. Zusätzlich gibt es die beiden *-Formen `\verb*` und `verbatim*`. Bei ihnen werden alle vorkommenden Leerzeichen als ␣ dargestellt.

Die Befehle `\indent` und `\noindent` wurden schon erwähnt. In einigen wenigen Fällen wird man global der standardmäßigen Absatzeinrückung einen zusätzlichen Absatzabstand vorziehen. Dazu sollte man aber nicht stur die beiden Längen `\parindent` und `\parskip` anpassen, sondern besser das Paket parskip einbinden. Wer dagegen eher spezielle Dinge mit sehr wenigen Seiten (Flyer, Grußkarten, ...) erstellen will, setzt am besten `\parindent` auf 0 und benutzt `\vspace`, `\smallskip`, u. ä. für die vertikalen Abstände. \indent \noindent

Damit im Blocksatz keine störenden Lücken entstehen, versucht LaTeX sein Bestes, geeignete Umbrüche zwischen den Wörtern oder innerhalb eines Wortes (Trennung) zu finden. Es gibt mehrere Gründe, warum dieser Algorithmus u. U. fehlschlagen kann; dazu gehören schmale Textbreite, große Schrift, untrennbare Wörter, `\mbox` usw. Empfindet LaTeX die Wortabstände in einer Zeile als zu groß, dann kommt es zur Meldung „`Underfull \hbox (badness xx) in paragraph at lines yy-zz`“. Möchte man den Absatz nicht umformulieren, kann man ihn in die Umgebung `sloppypar` setzen, die größere Wortabstände erlaubt. Diese Einstellung kann man auch mit dem Befehl `\sloppy` erreichen, die bis auf weiteres gilt, d. h. bis zum Ende der aktuellen Umgebung oder bis zum Befehl `\fussy`, der den Standard wiederherstellt. Man wird diese Befehle allerdings selten benötigen, weil LaTeX bei schmaler Textbreite in Tabellen und mehrspaltigem Druck selbständig auf `\sloppy` umschaltet. sloppypar \sloppy \fussy

```
> \shapepar{\nutshape} Das Paket ...
=
```

\shapepar \nutshape \circleshape \diamondshape \heartshape \squareshape

Das Paket shapepar erlaubt Absätze in verschiedenen Formen, hier z. B. eine Schraubenmutter. Solche Absätze werden automatisch auf der Seite zentriert. Sie sehen natürlich nur mit viel Text gut aus, was aber viel Ressourcen zum Compilieren benötigt. Deshalb sollte man das Paket nur in kleineren Dokumenten wie Glückwunschkarten o. ä. verwenden.

Die Formatierung erstreckt sich immer bis zum Ende des Absatzes. Statt `\nutshape` kann man auch `\circleshape` (Kreis), `\diamondshape` (auf der Spitze stehende Raute), `\heartshape` (Herz) oder `\squareshape` (Quadrat) benutzen.

\circlepar \diamondpar \heartpar \nutpar \squarepar \CDlabel

Als Abkürzung kann man auch die Befehle `\circlepar`, `\diamondpar`, `\heartpar`, `\nutpar` oder `\squarepar` verwenden. Sie erhalten jeweils den zu formatierenden Absatz als Argument. Insbesondere fügen `\diamondpar` am Anfang und Ende ein ♢ und `\heartpar` am Ende ein ♡ hinzu, wodurch die Spitzen besser ausgefüllt werden.

Ferner stellt das Paket den Befehl `\CDlabel` zur Verfügung, dessen Syntax `\circlepar` usw. gleicht. Sinnvollerweise richtet sich die Größe dieser Form nicht nach der Textmenge, sondern ist immer so groß wie ein Label für eine CD. Man muss dann selbst dafür sorgen, dass der Absatz eine geeignete Länge besitzt.

7.4 Weitere Fußnoten, Endnoten und Marginalien

An einigen Stellen, z. B. in mathematischen Formeln, sind Fußnoten nicht erlaubt. In diesem Fall kann man eine Marke für eine Fußnote setzen und den Text erst außerhalb erstellen:

\footnotemark \footnotetext

```
> \[ A \in \mathbbm{C}^{n\times n} \text{ heißt hermitesch\footnotemark,
  wenn } A^* = A \,. \]
  \footnotetext{nach Charles Hermite, 1822--1901}
```

= $$A \in \mathbb{C}^{n\times n} \text{ heißt hermitesch}^1\text{, wenn } A^* = A\,.$$

[1] nach Charles Hermite, 1822–1901

Auch `\footnotemark` kann wie `\footnote` ein optionales Argument für die Nummer erhalten.

\footnotemargin

Für weitere Einstellungen im Fußnotenlayout dient das Paket footmisc. Es definiert u. a. die Länge `\footnotemargin`, die den linken Einzug der ersten Zeile beschreibt. Als Paketoptionen sind z. B. die folgenden interessant:

- bottom: Setzt die Fußnoten an den Fuß der Seite. Da dies der Standard ist, ist die Option eigentlich nur notwendig, wenn Gleitobjekte (vgl. Abschnitt 8.4) mit der Option `b` existieren.
- hang: Setzt die Fußnotenmarke linksbündig. Ist dann `\footnotemargin` gleich 0, so wird die Fußnote um die Breite der Marke eingerückt, ansonsten um den Wert, den `\footnotemargin` angibt. Die Befehle `\hangfootparindent` und `\hangfootparskip` beschreiben bei dieser Option die Einrückung der folgenden Absätze und den Abstand zwischen den Absätzen. (\hangfootparskip)
- multiple: Diese Option bewirkt, dass bei Konstruktionen wie `\footnote{erste}\footnote{zweite}` ein Komma zwischen den Marken eingefügt wird, also z. B. „[3,4]“ anstatt des verwirrenden „[34]“. Das Komma ist im Befehl `\multfootsep` gespeichert, der natürlich verändert werden kann. (\multfootsep)
- para: Die Gesamtheit aller Fußnoten auf einer Seite wird als ein Absatz gesetzt, was möglicherweise bei sehr vielen kurzen Fußnoten geeignet ist.
- perpage: Die Nummerierung der Fußnoten beginnt auf jeder Seite von neuem bei 1.
- symbol: Verwendet Symbole (*, †, ...) für die Nummerierung der Fußnoten.

\endnote \endnotemark \endnotetext \theendnotes \notesname

Mit dem Paket endnotes kann man Endnoten definieren. Die Befehle `\endnote`, `\endnotemark` und `\endnotetext` funktionieren exakt genauso wie die entsprechenden Befehle für die Fußnoten. Der Befehl `\theendnotes` gibt die Endnoten mit einer entsprechenden Überschrift aus. Dazu zunächst ein Beispiel:

```
> a\endnote{aaa} b\endnote[99]{bbb} c\endnotemark\endnotetext{ccc}
  \theendnotes
```

= a[1] b[99] c[2]

Notes

[1]aaa
[99]bbb
[2]ccc

Ein Nachteil ist, dass sich der Text „Notes“ nicht nach der aktuellen Sprache richtet. Aber auch dieser Text ist wie üblich in einem Makro gespeichert, hier `\notesname`, und kann mit `\renewcommand` umdefiniert werden.

Endnoten werden mit dem Zähler `endnote` nummeriert, der aber genauso voreingestellt ist wie der Fußnotenzähler. Wenn man beide parallel benutzen möchte, dann sollte man unbedingt `\thefootnote` oder `\theendnote` umdefinieren. Die Schriftgröße für die Endnoten wird von `\enotesize` eingestellt. Durch Umdefinieren z. B. auf `\normalsize` kann die Schriftgröße angepasst werden. \enotesize

Für Randnotizen, sog. Marginalien, steht der Befehl `\marginpar` zur Verfügung: \marginpar

```
> Marginalien\marginpar{\centering\footnotesize Margi\-nalien} stehen im Rand.
```

= Marginalien stehen im Rand. Marginalien

Bei doppelseitigem Druck kann der Befehl ein optionales Argument erhalten, das dann als Text verwendet wird, wenn die Randnotiz auf einer linken Seite gesetzt wird.

Es gibt einige Längen und Befehle, die die Platzierung der Marginalien steuern. Die Längen sind `\marginparsep` für den Abstand der Randnotiz zum Text, `\marginparwidth` für die Breite der Randnotiz und `\marginparpush` für den minimalen Abstand zweier Randnotizen. Der Befehl `\reversemarginpar` bewirkt, dass ab sofort alle Marginalien nun innen (also links auf rechten Seiten und umgekehrt) stehen, `\normalarginpar` stellt den Standard wieder her. \marginparsep \marginparwidth \marginparpush \reversemarginpar \normalmarginpar

7.5 Zusätzliche Seitenformatierung

Um ein Dokument komplett im Querformat zu setzen, bindet man

```
> \usepackage[dvips,landscape]{geometry}
```

ein. Die Anzeigeprogramme erkennen dies und drehen die Ansicht auf Querformat. geometry

Benutzt man statt dessen das Paket lscape, so kann man den Inhalt einzelner Seiten auf Querformat drehen, obwohl die Seiten selbst einschließlich ihrer Kopf- und Fußzeilen im Hochformat bleiben. Sogar die Fußnoten und Gleitumgebungen (vgl. Abschnitt 8.4) werden korrekt mitgedreht. Die zu drehenden Seiten werden einfach in die Umgebung `landscape` gesetzt; zwangsläufigerweise erzeugt sie davor und dahinter einen Seitenwechsel. landscape

```
> \begin{landscape}
   erste Zeile\footnote{Fußnote}\par
   \vfill\par
   letzte Zeile
  \end{landscape}
```

Der Befehl `\enlargethispage` verlängert die aktuelle Seite um die Länge im Argument. Wird statt dessen die gesternte Form `\enlargethispage*` verwendet, so werden alle Gummilängen auf der Seite auf ihr Minimum reduziert. Im Beispiel \enlargethispage \enlargethispage*

```
> \enlargethispage{\baselineskip}
```

wird die Seite um eine Zeile verlängert. Die Länge `\baselineskip` ist der Abstand zweier aufeinanderfolgender Grundlinien. \baselineskip

Das Paket fancyhdr stellt schönere Seitenstile bereit, genaugenommen einen, der mit

```
> \pagestyle{fancy}
```

aktiviert werden kann. Meistens (d. h. bei vielen Dokumentenklassen) erhält man die Warnung „`\headheight is too small`“. Der der Warnung folgenden Zeile „`Make it at least`“ kann man entnehmen, welcher Wert für `\headheight` benötigt wird; diesen kann man mit `\setlength` zuweisen.

Beim fancy-Stil bestehen die Kopf- und Fußzeile jeweils aus drei Teilen, je einem links, zentriert und rechts. Ferner können linke und rechte Seiten unterschiedlich gestaltet werden. Allerdings muss dieser Seitenstil erst konfiguriert werden, wofür es mehrere Befehle gibt:

\lhead \chead \rhead \lfoot \cfoot \rfoot \fancy-head \fancy-foot \fancyhf

- `\lhead`, `\chead`, `\rhead`, `\lfoot`, `\cfoot`, `\rfoot`: Diese Befehle definieren, was links, zentriert bzw. rechts im Kopf bzw. Fuß stehen soll. Das optionale Argument kann dazu benutzt werden, um bei doppelseitigem Druck die linken Seiten anders als die rechten zu formatieren.
- `\fancyhead`, `\fancyfoot`: Diese Befehle definieren ebenfalls, was im Kopf bzw. im Fuß stehen soll; dabei dient das optionale Argument dazu, alle Positionen (s. u.) anzugeben, an die der folgende Ausdruck gesetzt werden soll.
- `\fancyhf`: Wie oben, nur dass bei den Positionen (s. u.) noch Kopf und Fuß spezifiziert werden können/müssen.

Für die Positionen gilt Folgendes:

- `o`, `e`: nur ungerade Seiten, nur gerade Seiten; keine Angabe: jede Seite.
- `l`, `c`, `r`: nur linker Teil, nur zentrierter Teil, nur rechter Teil; keine Angabe: alle Teile.
- `h`, `f`: nur Kopfzeile, nur Fußzeile; keine Angabe: Kopf- und Fußzeile.

Beispiele:

- `\fancyhf{}` löscht alle Einträge in Kopf- und Fußzeile.
- `\fancyhead[l]{Diplomarbeit}` setzt „Diplomarbeit“ links in die Kopfzeile.
- `\fancyhead[re,lo]{Diplomarbeit}` setzt „Diplomarbeit“ rechts in die Kopfzeile gerader und links in die Kopfzeile ungerader Seiten.
- `\fancyfoot[c]{\thepage}` setzt die Seitennummer zentriert in die Fußzeile.
- `\fancyhf[hle,fco]{\thepage}` setzt die Seitennummer links in die Kopfzeile gerader Seiten und zentriert in die Fußzeile ungerader Seiten.
- `\rhead{Diplomarbeit}` setzt „Diplomarbeit“ rechts in die Kopfzeile.
- `\cfoot{\thepage}` setzt die Seitennummer zentriert in die Fußzeile.

\leftmark \right-mark

Die Befehle `\leftmark` und `\rightmark` beinhalten das, was üblicherweise auf linken und rechten Seiten in der Kopfzeile steht. Häufig werden sie folgendermaßen verwendet:

```
> \fancyhead[re]{\leftmark}
  \fancyhead[lo]{\rightmark}
```

Dadurch werden das laufende Kapitel und der laufende Abschnitt in der Kopfzeile jeweils innen ausgegeben. In dieser Einführung hingegen stehen sie immer außen.

\head-rulewidth \footrule-width \fancy-headoffset \fancy-footoffset \fancyhf-offset

Interessant sind noch die beiden Befehle `\headrulewidth` und `\footrulewidth`, die die Dicken der Trennlinie an der Kopf- und Fußzeile festlegen. Sie sind aus internen Gründen als Befehle und nicht als Längen angelegt, so dass sie mit `\renewcommand` modifiziert werden müssen. Analog zu `\fancyhead`, `\fancyfoot` und `\fancyhf` gibt es `\fancyheadoffset`, `\fancyfootoffset` und `\fancyhfoffset`. Sie erhalten ein optionales Argument mit derselben Bedeutung wie bei `\fancyhead` und eine Länge im zwingenden Argument, die angibt, um wieviel die Kopf- bzw. Fußzeilenlinie übersteht. Beispiele:

- `\fancyhfoffset{0cm}` macht alle Linien bündig (Standard).
- `\fancyheadoffset{1cm}` macht die Kopfzeilenlinie innen und außen 1 cm länger.
- `\fancyfootoffset[le,ro]{-1cm}` macht die Fußzeilenlinie auf geraden Seiten links und auf ungeraden rechts 1 cm kürzer.

Der Befehl `\chapter` stellt den Seitenstil auf `plain` zurück, so dass der fancy-Stil unterdrückt wird. Das ist z. B. dann problematisch, wenn man die Seitennummer auf den übrigen Seiten unten außen stehen haben möchte. Allerdings ist auch der Seitenstil der Kapitelanfangsseiten konfigurierbar, und zwar: \fancypagestyle

```
> \fancypagestyle{plain}{
  \fancyhf{}
  \fancyfoot[le,ro]{\thepage}
 }
```

Wer in dem Modus schreibt, dass Kapitel nur auf rechten Seiten aufgehen, möchte eine möglicherweise vorhergehende leere linke Seite komplett weiß, d. h. ohne Kopf- und Fußzeile haben. Dazu kann man vor jedem `\chapter`-Befehl

```
> \clearpage\thispagestyle{empty}\cleardoublepage
```

eingeben. Der leeren Seite wird dadurch der Seitenstil `empty` zugewiesen.

Manchmal wüsste man gerne die Anzahl der Seiten in einem Dokument, um z. B. Seitennummerierungen der Form „Seite 3 von 7“ zu erstellen. Dies ist mit dem Paket lastpage möglich: lastpage \AtEndDocument

```
> Dieses Dokument umfasst \pageref{LastPage}~Seiten.
```

= Dieses Dokument umfasst xxxii Seiten.

Das Paket gerät hier in Schwierigkeiten, weil das Dokument mit einem Stichwortverzeichnis mit römisch nummerierten Seiten endet. Die Seitenreferenz gibt nämlich nichts anderes als die zuletzt generierte Seitennummer zurück, die im Zweifel nichts mit der Seitenanzahl zu tun haben muss. Das Paket arbeitet jedoch problemlos, wenn alle Seiten fortlaufend mit einem Format nummeriert sind. Dann kann man auch auf jede Seite die Seitenanzahl schreiben:

```
> \renewcommand{\thepage}{Seite \arabic{page} von \pageref{LastPage}}
  \AtEndDocument{\renewcommand{\thepage}{\arabic{page}}}
  \usepackage{lastpage}
```

Wichtig ist die Reihenfolge, d. h. dass das Paket erst nach den anderen beiden Befehlen eingebunden wird. Der Befehl in `\AtEndDocument` setzt die Zählerformatierung von `page` wieder auf den Standard zurück, damit *danach* vom Paket lastpage die korrekte Marke erstellt werden kann.

7.6 Stichwortverzeichnisse

Um ein Stichwortverzeichnis zu erstellen, bindet man am besten das Paket imakeidx ein. Da es mehrere Verzeichnisse erlaubt, muss jedes einzelne Verzeichnis zunächst definiert werden. Für das Standardverzeichnis wird man bevorzugt die Zeile \makeindex \index \printindex

```
> \makeindex[title=Stichwortverzeichnis]
```

verwenden wollen. Mit dem Befehl `\index` wird ein Eintrag ins Verzeichnis hinzugefügt:

```
> Eine \textbf{Gruppe}\index{Gruppe} ist eine Menge \dots
```

Dadurch wird ein Eintrag mit dem Namen „Gruppe“ mit der entsprechenden Seitenzahl erzeugt. Durch den Befehl `\printindex` kann an beliebiger Stelle das Verzeichnis ausgegeben werden. Der Befehl `\makeindex` akzeptiert eine Spaltenanzahl:

```
> \makeindex[title=Stichwortverzeichnis, columns=3]
```
Standard ist 2.

Möchte man ein zweites Stichwortverzeichnis (oder noch mehr) erstellen, dann muss man jeweils unterschiedliche Namen vergeben, z. B.
```
> \makeindex[name=persons, title=Personenverzeichnis]
```
für ein Personenverzeichnis. Der Name wird an zwei Stellen verwendet: Die für uns direkt sichtbare Stelle ist als optionales Argument für `\index` und `\printindex`. Mit
```
> \index[persons]{Knuth, Donald E.}
```
wird ein Eintrag zum Personenverzeichnis hinzufügt, und mit
```
> \printindex[persons]
```
wird das Verzeichnis ausgegeben.

Um die zweite Stelle zu verstehen, müssen wir uns genauer damit beschäftigen, wie die Verzeichnisse entstehen. Wird das Dokument compiliert, sucht LaTeX alle `\index`-Befehle ab und fügt den Eintrag mit Seitenzahl in eine Datei mit Erweiterung idx hinzu. Der Unterschied zum Inhaltsverzeichnis ist, dass diese Datei nun noch bearbeitet werden muss, bevor sie eingebunden werden kann. Dazu gehört insbesondere das Sortieren der Einträge. Dies erledigt ein Programm namens MakeIndex.exe, das ein Bestandteil der LaTeX-Distribution ist. Es sortiert die Einträge, fasst sie zusammen, formatiert sie und speichert sie in einer Datei mit der Erweiterung ind. Ferner wird eine Logdatei mit der Erweiterung ilg erstellt. MakeIndex muss für jedes Stichwortverzeichnis je einmal aufgerufen werden, aber dies erledigt LaTeX automatisch für uns. Die erwähnte zweite Stelle für den Namen ist der Dateiname; während für das namenlose Standardverzeichnis der Projektname benutzt wird, heißen die weiteren Verzeichnisse so wie der Name, also hier z. B. persons.idx, usw.

MakeIndex stellt eine Vielzahl von Möglichkeiten bereit, um Sortierkriterien und Formatierung für jeden einzelnen Eintrag zu beeinflussen. Deklariert man das Standardverzeichnis wie oben mit `\makeindex` und verwendet `\printindex` am Ende des Dokuments, so entsteht durch

```
> Befehle\index{Befehle} \\
  eigene Befehle\index{Befehle!eigene} \\
  Index\index{Index|(} \\
  \LaTeX\index{LaTeX@\LaTeX} \\
  Philosoph\index{Philosoph} \\
  3.14159\index{pi@$\pi$!Näherung} \\
  wichtig\index{wichtig|textbf}
  \clearpage
  eigene Befehle\index{Befehle!eigene!umdefinieren|textit} \\
  Latten\index{Latten} \\
  Index\index{Indexbefehl@\textbackslash index} \\
  Ausrufezeichen\index{Ausrufezeichen@"!} \\
  $\varphi$\index{phi@$\varphi$} \\
  Index\index{Index|)} \\
  Befehle\index{Befehle} \\
  Kommandos\index{Kommandos|see{Befehle}} \\
  $\pi$\index{pi@$\pi$}
```
das folgende Stichwortverzeichnis:

= **Stichwortverzeichnis**

Ein normaler Eintrag (`Latten`) erzeugt diesen Eintrag mit Seitennummer; taucht der Eintrag mehrfach auf (`Befehle`), dann werden mehrere Seitennummern durch Komma getrennt angegeben. Drei oder mehr aufeinanderfolgende Seiten werden in der Form „x–y" geschrieben. Mit dem Ausrufezeichen `!` können Untereinträge (`Befehle!eigene`) generiert werden. Mit `@` wird dem Eintrag ein Sortierschlüssel vorangestellt (`pi@$\pi$` oder `LaTeX@\LaTeX`). Dies ist z. B. dann notwendig, wenn der Eintrag aus Sonder- bzw. mathematischen Zeichen besteht. Mit `|(` und `|)` kann ein Bereich (`Index|(` und `Index|)`) geöffnet und geschlossen werden, so dass eine Seitennummer der Form „x–y" erzeugt wird. Mit `|see` kann auf ein anderes Stichwort (`Kommandos|see{Befehle}`) verwiesen werden; alle anderen Ergänzungen mit `|` dienen der Formatierung der Seitenzahl (`wichtig|textbf`). Um z. B. ein `!` oder `@` dennoch ausgeben zu können, muss es mit `"` maskiert werden (`Ausrufezeichen@"!`).

Der Befehl `\seename` regelt, was bei `|see` ausgegeben wird. Wer den Text ändern will, kann z. B. \seename \see

```
> \renewcommand{\seename}{vgl.}
```

verwenden. Zusammen mit dem ulem-Paket werden diese Wörter unterstrichen, weil sie mit `\emph` implementiert sind, was meistens nicht erwünscht ist. Abhilfe kann man sich mit

```
> \renewcommand{\see}[2]{\textit{\seename} #1}
```

schaffen.

Allerdings funktioniert `\index{\{}` nicht, um eine geschweifte Klammer ins Stichwortverzeichnis hinzuzufügen. Man erhält eine Fehlermeldung, noch bevor MakeIndex aufgerufen wird. Die Lösung besteht darin, `\{` durch `$\lbrace$` (analog `$\rbrace$`) zu ersetzen. \lbrace \rbrace

7.7 Weitere Verzeichnisse

Ein Literaturverzeichnis erstellt man am einfachsten mit der Umgebung `thebibliography`. Wir geben zunächst ein Beispiel an: thebibliography \bibitem

```
> \begin{thebibliography}{0}
   \bibitem{Dem08} \textsc{W. Demtröder}, \textit{Experimentalphysik 1:
    Mechanik und Wärme}, Springer 2008
   \bibitem{Koe04} \textsc{K. Königsberger}, \textit{Analysis 1},
    Springer 2004
  \end{thebibliography}
```

= [1] W. Demtröder, *Experimentalphysik 1: Mechanik und Wärme*, Springer 2008

[2] K. Königsberger, *Analysis 1*, Springer 2004

Der Befehl `\bibitem` legt einen neuen Eintrag an, die Nummerierung erfolgt mit [1], [2], [3], usw. Das Argument der Umgebung ist der längste Nummerierungseintrag ohne die eckigen Klammern; gewöhnlich verwendet man `0`, wenn man weniger als 10 Einträge hat, `00` für weniger als 100, usw.

\cite

Das Argument von `\bibitem` ist eine Marke, ähnlich denen in `\label`. Mit `\cite` referenziert man auf einen Literatureintrag:

```
> Genaueres ist in \cite{Dem08} und \cite[S.~52]{Koe04} zu finden. \\
  Beide Bücher \cite{Dem08,Koe04} behandeln Stoff des Grundstudiums.
```

= Genaueres ist in [1] und [2, S. 52] zu finden.
Beide Bücher [1, 2] behandeln Stoff des Grundstudiums.

Das optionale Argument von `\cite` ist für Zusatzangaben innerhalb der Quelle gedacht, wie hier z. B. eine Seitenzahl oder etwas Ähnliches.

Die Marken `Dem08` und `Koe04` sind nur interne Marken so wie andere Querverweise auch. Möchte man die sichtbaren Marken [1], [2], usw. durch etwas anderes ersetzen, dann muss man an `\bibitem` noch ein optionales Argument übergeben:

```
> \begin{thebibliography}{Kön04}
    \bibitem[Kön04]{Koe04} \textsc{K. Königsberger}, \textit{Analysis 1},
     Springer 2004
  \end{thebibliography}
```

= [Kön04] K. Königsberger, *Analysis 1*, Springer 2004

Der Befehl `\cite{Koe04}` erzeugt dann natürlich [Kön04] im Text.

\listoftables
\listoffigures

Es gibt zwei weitere Standardverzeichnisse, nämlich je eines für Tabellen und Abbildungen. Die beiden Befehle dazu lauten `\listoftables` und `\listoffigures`, und sie funktionieren genauso wie `\tableofcontents`. Die Überschriften sind „Tabellenverzeichnis“ und „Abbildungsverzeichnis“, und auch bei ihnen muss man das Dokument in der Regel zweimal compilieren. Bisher haben wir allerdings noch nichts zu Tabellen und Bildern sowie Gleitumgebungen gesagt; dies diskutieren wir in Kapitel 8. Am Ende des Dokuments befindet sich zur Demonstration das Tabellen- und Abbildungsverzeichnis dieser Einführung.

\addcontentsline

Die Daten für das Inhalts-, Tabellen- und Abbildungsverzeichnis werden in den Dateien Projektname.toc[1], Projektname.lot[2] und Projektname.lof[3] gespeichert. In Verbindung mit einer dieser Dateierweiterungen fügt der Befehl `\addcontentsline` eine zusätzliche Zeile in das Verzeichnis ein, z. B.

```
> \addcontentsline{toc}{section}{ein manueller Abschnitt}
  \addcontentsline{lot}{table}{eine manuelle Tabelle}
  \addcontentsline{lof}{figure}{eine manuelle Abbildung}
```

Statt `section` ist natürlich auch jede andere Gliederungsstufe möglich, die in der verwendeten Dokumentenklasse verfügbar ist.

\addtocontents

Wenn man etwas Allgemeineres zu einem der drei Verzeichnisse hinzufügen möchte, kann man `\addtocontents` benutzen. Die Syntax ist ähnlich wie bei `\addcontentsline`, nur ohne das zweite Argument. Der Text im letzten Argument wird einfach in die Datei geschrieben; dies kann beispielsweise `\clearpage` sein, um einen manuellen Seitenwechsel einzufügen.

\listtheorems

Das Paket ntheorem (vgl. Abschnitt 5.6) stellt Befehle zur Verfügung, um die Theoreme aufzulisten. Die Liste unterscheidet sich allerdings in mindestens zweierlei Hinsicht von den anderen: Erstens erzeugt der Befehl weder eine Überschrift (im book-Stil auch keine neue Seite) noch etwas Abstand vor bzw. nach Liste. Zweitens muss man dem Befehl die Umgebungsnamen übergeben, die er auflisten soll, wodurch es z. B. möglich ist, getrennte Listen für Definitionen und Sätze anzulegen. Ein Beispiel für ein book-Dokument:

[1] toc steht für „table of contents“.
[2] lot steht für „list of tables“.
[3] lof steht für „list of figures“.

```
> \chapter*{Verzeichnis der Theoreme}
  \addcontentsline{toc}{chapter}{Verzeichnis der Theoreme}
  \subsection*{Definitionen:}
  \listtheorems{defin}
  \subsection*{Sätze und Lemmata:}
  \listtheorems{satz,hsatz,lemma}
```

Mit `\theoremlisttype` kann die Liste formatiert werden. Als Argument sind möglich: \theorem-listtype

- `all`: alle Theoreme
- `allname`: wie `all`, nur mit dem Wort „Definition", „Satz", usw. dazu
- `opt`: nur die Theoreme, die mit Hilfe des optionalen Arguments benannt wurden
- `optname`: wie `opt`, nur mit dem Wort „Definition", „Satz", usw. dazu

Auch in das Verzeichnis der Theoreme kann man Dinge manuell hinzufügen. Beispielsweise fügt `\addtheoremline{defin}{Topologie}` eine Definition mit Namen „Topologie" hinzu; die gesternte Version des Befehls unterdrückt die Nummer. Mit `\addtotheoremfile` hingegen, der nur ein zwingendes Argument besitzt, kann man wieder etwas Beliebiges (z. B. `\clearpage`) in die Liste hinzufügen. Fügt man vor dem zwingenden ein optionales hinzu, das wieder eine Theorem-Bezeichnung wie `defin` sein muss, dann wird der Text nur zu dieser entsprechenden Liste hinzugefügt. \addtheoremline \addtheoremline* \addtotheoremfile

7.8 Titelseite, Vorwort, Einleitung und Anhänge

Ein großes `book`-Dokument kann mit den Befehlen `\frontmatter`, `\mainmatter`, `\appendix` und `\backmatter` gegliedert werden. Sie wirken sich auf die Seiten- und Kapitelnummerierung aus. Ein Beispiel: \frontmatter \backmatter \appendix \backmatter

```
> \frontmatter
  \chapter{Vorwort}
  \tableofcontents
  \chapter{Einleitung}
  \mainmatter
  \chapter{Frühgeschichte}
  \chapter{Mittelalter}
  \chapter{Neuzeit}
  \appendix
  \chapter{Jahreszahlen}
  \backmatter
  \chapter{Danksagung}
  \cleardoublepage
  \addcontentsline{toc}{chapter}{Stichwortverzeichnis}
  \printindex
```

`\frontmatter` schaltet die Seitennummerierung auf i, ii, iii, … um und unterbindet die Nummerierung der Kapitel. `\mainmatter` stellt die Seitenummerierung, erneut von vorne beginnend, auf 1, 2, 3, … zurück und nummeriert die Kapitel in der Standardform 1, 2, 3, …; `\appendix` stellt die Kapitelnummerierung, erneut von vorne beginnend, auf A, B, C, … um und ersetzt die Überschrift „Kapitel" durch „Anhang". `\backmatter` schaltet die Kapitelnummerierung schließlich wieder aus.

Sobald das Dokument doppelseitig (also mit `twoside`, was Standard bei book ist) gesetzt wird, führen die beiden Befehle, die die Seitennummerierung umstellen, also `\frontmatter` und `\mainmatter`, den Befehl `\cleardoublepage` aus, damit die ungeraden Seiten weiterhin rechts sind. Das Kapitel direkt nach einem dieser beiden Befehle geht also immer rechts auf, auch wenn die Dokumentoption `openany` aktiv ist. Aus einem ähnlichen Grund steht vor dem `\addcontentsline`-Befehl, der das Stichwortverzeichnis ins Inhaltsverzeichnis einträgt, ein `\cleardoublepage`. Nur so erhält der Eintrag auch die richtige Seitennummer. In einem einseitigen Dokument ist er durch `\clearpage` zu ersetzen.

LaTeX unterstützt auch Titelseiten, wie sie bei wissenschaftlichen Veröffentlichungen üblich sind. Auf ihr werden Titel, Autor(en) und Datum angezeigt. Ein fehlender Titel erzeugt einen Fehler, fehlende Autoren lediglich eine Warnung. Die Information wird bei report und book auf eine separate Seite (also echte Titelseite) gesetzt, bei article beginnt der folgende Text unter diesen Angaben. Diese Einstellungen lassen sich durch die Dokumentoptionen titlepage und notitlepage umstellen; siehe dazu Unterabschnitt 1.3.1.

\title \author \thanks \and \date \maketitle abstract

In der Präambel definieren die Befehle `\title`, `\author` und `\date` den Titel, die Autoren und das Datum. Letzteres ist standardmäßig auf `\today`, also auf das aktuelle Datum eingestellt. In `\author` kann `\thanks` benutzt werden, um eine Fußnote zu einem Autor zu erzeugen; dies ist prinzipiell nichts anderes als `\footnote` mit auf `\fnsymbol` umgestellter Formatierung. Mehrere Autoren lassen sich mit `\and` verbinden, was einen mäßigen Abstand erzeugt. Im Dokument – häufig direkt nach `\begin{document}` – wird mit `\maketitle` die Titelseite angelegt. Ferner kann man mit der Umgebung `abstract` eine Zusammenfassung schreiben; allerdings nicht im book-Stil, denn bei solch großen Dokumenten würde man die Zusammenfassung wohl als eigenes Kapitel machen wollen.

Ein mehr oder weniger vollständiges Beispiel könnte also so aussehen:

```
> [...]
  \title{\LaTeX-Einführung}
  \author{Stefan Krause\thanks{TU Wien} \and
   Daniel Saure\thanks{Universität Mainz}}
  \date{heute}
  \begin{document}
   \maketitle
   \begin{abstract}
    Diese Einführung \dots
   \end{abstract}
  [...]
```

\input

Wenn die Dokumente groß werden, ist es auch günstig, den Quelltext auf mehrere Dateien zu verteilen. Der Befehl `\input` fügt den Inhalt der als Argument angegebenen Datei stur an der Stelle ein. Standardmäßig wird die Erweiterung tex verwendet, wenn keine angegeben ist, und Pfadbezeichnungen werden – wie auf Unix-Systemen üblich – mit dem normalen Schrägstrich `/` geschrieben. Selbstverständlich kann auch eine Datei mehrfach eingebunden werden, und der Befehl kann verschachtelt werden.

\include

Der Befehl `\include` hingegen ist etwas intelligenter. Er fügt die Datei ebenfalls ein, beginnt aber immer auf einer leeren Seite, d. h. er ruft vor dem Einbinden `\clearpage` auf. Zusätzlich wird eine Hilfsdatei mit der Erweiterung aux angelegt, in der z. B. die Inhaltsverzeichniseinträge geführt werden. Dieser Mechanismus verbietet es in der Regel, eine Datei mehrfach einzubinden, was aber mit `\include` ohnehin nie erwünscht ist.

\includeonly

Mit den Hilfsdateien ist es möglich, nur einige Dateien einzubinden und trotzdem alle

Seitenzahlen, sonstige Zähler und Querverweise korrekt zu behalten. Dies leistet der Befehl `\includeonly`, wie im folgenden Beispiel deutlich wird:

```
> [...]
  \includeonly{Kapitel1,Kapitel4}
  \begin{document}
   \include{Kapitel1}
   \include{Kapitel2}
   \include{Kapitel3}
   \include{Kapitel4}
  \end{document}
```

Sofern das Dokument einmal korrekt ohne `\includeonly` erstellt wurde, dann erzeugt dieser Quelltext eine Ausgabedatei nur mit den Dateien Kapitel1.tex und Kapitel4.tex. Würde man nur die anderen beiden `\include`-Befehle auskommentieren, dann würden sich die Seitenzahlen verschieben und eventuelle Referenzen auf diese Kapitel würden zu **??** werden.

Kapitel 8

Tabellen und Bilder

8.1 tabular-Umgebung

Mit der `tabular`-Umgebung können Tabellen erstellt werden. Die grundsätzliche Syntax ist dieselbe wie die bei der Umgebung `array` im Mathematikmodus:

tabular l c r & \\

```
> \begin{tabular}{lcr}
   aaaaa & bbbbb & ccccc \\
   a & b & c
  \end{tabular}
```

=

aaaaa	bbbbb	ccccc
a	b	c

`&` wechselt in die nächste Spalte, `\\` in die nächste Zeile der Tabelle. Das Argument der Umgebung ist die Deklaration der Spalten, `l` für eine linksbündige, `c` für eine zentrierte und `r` für eine rechtsbündige Spalte. Der Zwischenraum wird automatisch erzeugt.

Möchte man etwas anderes statt des Abstands zwischen die Spalten setzen, so hat man folgende Möglichkeiten:

| || @ !

```
> \begin{tabular}{@{}l|l||l!{xxx}l@{yyy}l!{x}|@{y}l}
   aaa & aaa & aaa & aaa & aaa & aaa
  \end{tabular}
```

= aaa | aaa ‖ aaa xxx aaayyyaaa x|yaaa

`|` erzeugt eine senkrechte Linie durch die gesamte Tabelle und verbreitert den Spaltenabstand um die Breite der Linie, `||` erzeugt analog eine Doppellinie. `@` ersetzt den Spaltenabstand durch sein in geschweiften Klammern folgendes Argument. Vor der ersten Spalte bewirkt z. B. `@{}`, dass die Tabelle keinen linken Rand bekommt, der häufig unerwünscht ist, wenn sie nicht umrahmt ist. `!` (Paket `array`) fügt zwischen zwei Spalten das Argument ein, ohne jedoch den Spaltenabstand zu entfernen. Wie im Beispiel können `@` und `!` auch links bzw. rechts von `|` und `||` benutzt werden.

Innerhalb einer `l`-, `c`- und `r`-Spalte führt LaTeX keinen automatischen Zeilenumburch durch. Dafür gibt es:

p m b

```
> \begin{tabular}{|p{3cm}|p{3cm}|}
   die erste Spalte & noch eine weitere Spalte
  \end{tabular}
```

=

die erste Spalte	noch eine weitere Spalte

```
> \begin{tabular}{|m{3cm}|m{3cm}|}
   die erste Spalte & noch eine weitere Spalte
  \end{tabular}
```

=

die erste Spalte	noch eine weitere Spalte

```
> \begin{tabular}{|b{3cm}|b{3cm}|}
   die erste Spalte & noch eine weitere Spalte
  \end{tabular}
```

=

die erste Spalte	noch eine weitere Spalte

Alle drei Spaltenspezifikatoren `p`, `m` und `b` erstellen eine Spalte im Blocksatz mit der angegebenen Breite, sie unterscheiden sich jedoch in der vertikalen Ausrichtung der Zellen in einer Zeile. `m` und `b` befinden sich im Paket array.

*

Hilfreich für das Erzeugen mehrerer gleicher Spalten ist der Spezifikator `*`:

```
> \begin{tabular}{*{4}{|p{15mm}}|}
   aaa & bbb & ccc & ddd
  \end{tabular}
```

=

aaa	bbb	ccc	ddd

Er wiederholt die Spaltendefinition aus seinem zweiten Argument so oft, wie es durch das erste Argument angegeben ist.

\\ \newline

Um einen manuellen Zeilenumbruch in Spalten, die mit `p`, `m` oder `b` erzeugt wurden, zu erzwingen, kann man nicht `\\` verwenden, da ja dieser Befehl in die nächste Zeile der Tabelle wechselt. Statt dessen muss man hier `\newline` benutzen:

```
> \begin{tabular}{|p{3cm}|p{3cm}|}
   die erste Spalte & noch eine \newline weitere Spalte
  \end{tabular}
```

=

die erste Spalte	noch eine weitere Spalte

> <

Bei allen Spaltenspezifikatoren `l`, `c`, `r`, `p`, `m` und `b` ist es möglich, Kommandos einzufügen, die beim Eintreten in die Spalte oder beim Verlassen der Spalte ausgeführt werden:

```
> \begin{tabular}{|>{\bfseries}l|>{$}l<{$}|>{\slshape}l|}
   xxx & a+b & yyy \\
   \sffamily X & c\cdot d & Y
  \end{tabular}
```

=

xxx	$a+b$	*yyy*
X	$c \cdot d$	*Y*

Dabei bewirkt `>` (Paket array) vor einem `l` usw., dass nach dem Eintreten in diese Spalte zunächst die hinter `>` stehenden Befehle ausgeführt werden, und zwar natürlich in jeder Zeile. Analoges gilt für `<` (Paket array) hinter einem `l` usw. vor dem Verlassen der Spalte. In der zweiten Zeile wurde zusätzlich zu `\bfseries` von `>` noch `\sffamily` direkt eingefügt. Dieser Befehl wirkt aber nur bis zum nächsten `&` (oder auch `\\`), d. h. er gilt nur für diese Zelle.

\newcolumntype

Es ist sogar möglich, eigene Spaltenspezifikatoren zu definieren:

```
> \newcolumntype{M}{>{$}p{3cm}<{$}}
  \begin{tabular}{|M|M|}
   a+b & a\cdot b
```

```
  \end{tabular}}
```

= | $a+b$ | $a \cdot b$ |

Horizontale Linien können mit dem Befehl `\hline` eingefügt werden, zweimal hintereinander erzeugt er eine Doppellinie: \hline

```
> \begin{tabular}{l|l} \hline
   aaa & bbb \\ \hline\hline
   ccc & ddd \\
   eee & fff \\ \hline
  \end{tabular}
```

=

aaa	bbb
ccc	ddd
eee	fff

Im hier auftretenden Fall, dass die Tabelle durch eine waagerechte Linie abgeschlossen werden soll, muss ausnahmsweise nach der letzten Zeile ein `\\` stehen.

Horizontale Linien, die sich nicht über die gesamte Tabellenbreite erstrecken sollen, können mit `\cline` gesetzt werden: \cline

```
> \begin{tabular}{|l|l|l|l|} \hline
   x & x & x & x \\ \cline{1-2}
   x & x & x & x \\ \cline{1-1}\cline{4-4}
   x & x & x & x \\ \cline{2-3}
   x & x & x & x \\ \hline
  \end{tabular}
```

=

x	x	x	x
x	x	x	x
x	x	x	x
x	x	x	x

Mit `\multicolumn` kann eine Zelle gesetzt werden, die sich über mehrere Spalten erstreckt: \multicolumn

```
> \begin{tabular}{|lll|} \hline
   \multicolumn{3}{|c|}{Tabelle} \\
   a & b & c \\ \hline
  \end{tabular}
```

=

Tabelle		
a	b	c

Das erste Argument gibt die Anzahl der Spalten an, das zweite die Formatierung der neuen Zelle und das dritte den Text. Man beachte, dass im Beispiel die senkrechten Linien ebenfalls neu definiert werden müssen.

`\multicolumn` kann sogar für eine Spalte sinnvoll sein:

```
> \begin{tabular}{|>{$}r<{^\circ$}|>{$}r<{$}|} \hline
   \multicolumn{1}{|c|}{$\alpha$} &
    \multicolumn{1}{c|}{$\cos\alpha$} \\ \hline
   45 & 0{,}7071 \\
   90 & 0{,}0000 \\
   150 & -0{,}8660 \\ \hline
  \end{tabular}
```

=

α	$\cos\alpha$
45°	0,7071
90°	0,0000
150°	−0,8660

8.1.1 Längen

\tabcolsep \arraycolsep

Es gibt noch einige Längen, die die `tabular`-Umgebung beeinflussen. Die erste unter ihnen ist `\tabcolsep`; sie bestimmt den Abstand vom Text zum Rand der Zelle.

```
> {\setlength{\tabcolsep}{5mm}
  \begin{tabular}{|c|cc|} \hline
   x & \rule{5mm}{0.25mm} & y \\ \hline
  \end{tabular}}
```

=

x	——	y

Für die `array`-Umgebung muss man statt `\tabcolsep` die Länge `\arraycolsep` verwenden.

\extrarowheight

`\extrarowheight` definiert die zusätzliche Höhe einer Zelle, nicht aber die Tiefe. Es wird also nur über den Zelleneinträgen Leerraum erzeugt, nicht unter ihnen. Die Länge hat in derselben Weise Einfluss auf die `align`-Umgebung.

```
> {\setlength{\extrarowheight}{3mm}
  \begin{tabular}{|c|c|} \hline
   x & y \\ \hline
   x & y \\ \hline
  \end{tabular}}
```

=

x	y
x	y

Wer auch eine Tiefe, d. h. zusätzlichen Leerraum nach unten hinzufügen möchte, kann dies bei jedem `\\`-Befehl per optionales Argument tun:

```
> {\setlength{\extrarowheight}{1mm}
  \begin{tabular}{|c|c|} \hline
   x & y \\[1mm] \hline
   x & y \\[1mm] \hline
  \end{tabular}}
```

=

x	y
x	y

\arrayrulewidth \doublerulesep

Die Dicke der horizontalen und vertikalen Linien in `tabular`- und `array`-Umgebungen wird von `\arrayrulewidth` gesteuert, der Abstand zwischen den beiden Linien einer Doppellinie von `\doublerulesep`.

```
> {\setlength{\arrayrulewidth}{0.4mm}
  \setlength{\doublerulesep}{2mm}
  \begin{tabular}{c|c||c} \hline
   a & b & c \\ \hline
  \end{tabular}}
```

=

a	b	c

8.2 tabbing-Umgebung

Die Umgebung `tabbing` kann für einfachere Tabellen verwendet werden. Dabei wird jeder Eintrag linksbündig gesetzt, die Einträge können nicht umgebrochen werden (höchstens mit `\parbox`en, was aber zu keinem zufriedenstellenden Ergebnis führt), die Breiten der Spalten richten sich nach der ersten Zeile, und es können keine Linien gezogen werden. Ein kleiner Vorteil ist, dass innerhalb der Umgebung ein Seitenwechsel stattfinden darf. Wir geben zuerst ein ausführliches Beispiel:

tabbing
\\
\kill
\=
\>
\+
\-
\pushtabs
\poptabs

```
> \begin{tabbing}
   Osterglocke \= violett \= 100 \= 10,00~\euro \kill
   Rose \> rot \> 10 \> 8,30~\euro \\
   \> weiß \> 10 \> 9,20~\euro \\
   Osterglocke \> \> 20 \> 12,90~\euro \\[1ex]
   \pushtabs
   Außerdem im Angebot: \\
   Blumentöpfe \= 8,00~\euro \\
   Erde \> 4,50~\euro \\[1ex]
   \poptabs
   Tulpe \> rot \> 8 \> 5,10~\euro \+ \\
    violett \> 8 \> 5,30~\euro \\
    gelb \> 12 \> 7,00~\euro \- \\
   Nelke \> rosa \> 20 \> 14,20~\euro \\
   Usambaraveilchen \> violett \> 1 \> 5,90~\euro
  \end{tabbing}
```

=

Rose	rot	10	8,30 €
	weiß	10	9,20 €
Osterglocke		20	12,90 €

Außerdem im Angebot:

Blumentöpfe	8,00 €
Erde	4,50 €

Tulpe	rot	8	5,10 €
	violett	8	5,30 €
	gelb	12	7,00 €
Nelke	rosa	20	14,20 €
Usambaraveilchen	violett	1	5,90 €

Der Befehl `\=` setzt einen Tabulator an die aktuelle Position, `\kill` beendet die Zeile, ohne sie auszugeben. Die erste Zeile im Beispiel dient dazu, die Tabulatoren so zu setzen, dass die längsten auftretenden Einträge hineinpassen. Sind die Tabulatoren durch die erste Zeile gesetzt worden, dann kann man in jeder folgenden Zeile mit `\>` zum nächsten Tabulator springen. `\\` wechselt wie üblich in die nächste Zeile, auch hier ist ein optionales Argument für zusätzlichen Zeilenabstand möglich.

Der Befehl `\+` am Ende einer Zeile bewirkt, dass alle folgenden Zeilen automatisch um einen Tabulator eingerückt sind, also so, als würde man jedesmal `\>` an den Anfang der Zeilen schreiben. Natürlich kann `\+` auch mehrfach hintereinander benutzt werden. `\-` am Ende einer Zeile macht diese Einrückungen schrittweise wieder rückgängig. Mit `\pushtabs` können die momentanen Tabulatorpositionen abgespeichert und später mit `\poptabs` wieder

„geladen“ werden. Dazwischen können die Tabulatoren beliebig verändert werden, wobei auch nach `\pushtabs` die alten Positionen weiterbenutzt werden können. Das erste Auftreten von `\=` setzt neue Tabulatoren.

In der letzten Zeile des Beispiels sieht man auch, wie sich `\>` stur zum nächsten Tabulator bewegt – in diesem Fall sogar rückwärts –, obwohl der Eintrag „Usambaraveilchen“ viel zu lang ist.

\= \' \` \a

In der `tabbing`-Umgebung wird der Befehl `\=` verwendet, um Tabulatoren zu setzen, obwohl er eigentlich schon für einen Akzent vordefiniert ist. Dasselbe gilt für `\'` und `` \` ``, deren Funktionen im folgenden Beispiel erläutert werden. Um die Akzente dennoch verwenden zu können, gibt es `\a'`, `` \a` `` und `\a=`:

```
> \begin{tabbing}
   Latein\hspace{1cm} \= Deutsch \\
   par\a=ere \> gehorchen \\
   par\u ere \> erzeugen, gebären \\
   \textit{mpl} \' parentes \> Eltern \` \textsf{vgl. engl.} parents
  \end{tabbing}
```

= Latein Deutsch

	Latein	Deutsch	
	parēre	gehorchen	
	parĕre	erzeugen, gebären	
mpl	parentes	Eltern	vgl. engl. parents

`\'` setzt Text vor die erste Spalte in den linken Rand, `` \` `` hingegen rechtsbündig in die Zeile.

8.3 Bilder einbinden

Es gibt drei Standardausgabeformate für LaTeX: DVI (Device Independent Format), PS (Postscript) und PDF (Portable Document Format). PS wird immer aus DVI erstellt, PDF jedoch kann aus PS oder direkt aus TEX gewonnen werden. DVI und alle Formate, die daraus erzeugt wurden, können nur Bilder im EPS-Format (Encapsulated Postscript) verarbeiten. Das EPS-Bild sollte unbedingt eine sog. Bounding Box enthalten, die leider nur sehr wenige Bildbearbeitungsprogramme erzeugen. Sehr gut jedoch eignet sich Gimp, das als Freeware-Programm für Windows und Linux erhältlich ist.

\include-graphics

Wenn man eine EPS-Grafik mit Bounding Box erstellt hat, kann man sie mit dem Paket graphicx einbinden:

```
> \includegraphics{lok.eps}
```

=


Dabei muss die Bilddatei in demselben Ordner wie die LaTeX-Datei liegen. Insbesondere wird sie bei DVI nur verlinkt, was zu kleinen DVI-Dateien führt, wodurch man aber gezwungen wird, die Bilddateien immer mitzukopieren. In PS und PDF werden die Bilder eingebettet, so dass nach dem Compilieren keine externen Dateien mehr notwendig sind.

Befindet sich das Bild in einem anderen Ordner, so muss man mit Hilfe von absoluten oder relativen Pfaden den Speicherort der Grafik angeben:

```
> \includegraphics{D:/Daten/Latex/lok.eps}
  \includegraphics{../lok.eps}
```

Auch unter Windows-Systemen müssen „normale“ Schrägstriche in den Pfaden verwendet werden.

Man kann LaTeX allerdings auch sagen, wo es nach den Bildern zu suchen hat. Dies bewerkstelligt man mit `\graphicspath` (Paket graphicx): \graphicspath

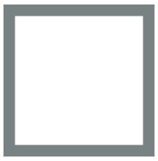
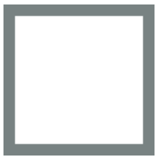
```
> \graphicspath{{D:/Daten/Latex/}{../}}
  \includegraphics{lok.eps}
```

Die etwas seltsame Syntax ist wie folgt: Im Argument können mehrere Pfadangaben stehen. Jede einzelne ist von einem Paar geschweifter Klammern umgeben, wobei sich zwischen den Pfadangaben kein zusätzliches Trennzeichen befindet. Jede Pfadangabe muss mit einem Schrägstrich enden.

Erzeugt man aus den Quelltexten direkt das PDF-Dokument, also ohne dazwischen DVI zu erstellen, kann man nur Bilder im JPEG-Format einbinden. Diese werden ebenfalls eingebettet, so dass man zum Betrachten keine Bilddateien mehr benötigt.

Die Grundlinie eines Bildes ist grundsätzlich die untere Kante, was bei der Ausrichtung mit umgebendem Text Schwierigkeiten bereiten kann:

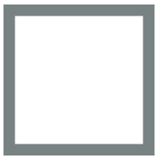
```
> xxx \includegraphics{quad.eps} xxx
  \raisebox{-9.2mm}{\includegraphics{quad.eps}} xxx
```

= xxx  xxx  xxx

Hier wurde `\raisebox` verwendet, um die Ausrichtung zu verändern.

```
> \begin{wrapfigure}[6]{r}{22mm}
   \hfill\includegraphics{quad.eps}
  \end{wrapfigure}
  Das Paket \textsf{wrapfig} dient dazu, ...
```

wrapfigure



Das Paket wrapfig dient dazu, Bilder zu setzen, um die der Text herumfließen kann. Allerdings kann hier unter „Bild“ alles Mögliche verstanden werden; die Umgebung `wrapfigure`, die vor dem umfließenden Text stehen muss, kann beliebige LaTeX-Kommandos enthalten. Das erste Argument von `wrapfigure` muss `l` für links oder `r` für rechts sein, das zweite gibt die Breite des Einzugs an. Vor dem ersten Argument kann noch ein optionales Argument übergeben werden, das die Anzahl der einzurückenden Zeilen angibt; dabei zählt eine abgesetzte Formel als drei Zeilen. Allerdings ist dieses Argument in der Regel unnötig, weil die Anzahl der Zeilen automatisch berechnet wird. Im vorliegenden Beispiel hätte diese Berechnung zu einer Zeile mehr geführt.

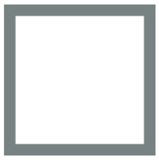


Abbildung 8.1: Das erste Bild in einer Gleitumgebung.

8.4 Gleitende Objekte

Die Befehle, die wir bisher zum Erstellen und Einbinden von Tabellen und Bildern gesehen haben, setzen die entsprechende Konstruktion direkt an die Stelle, wie es der Quelltext vorschreibt. Dies kann bei vielen Tabellen und Bildern im Dokument einerseits dazu führen, dass wegen ungünstiger Seitenumbrüche viel Leerraum gelassen werden muss. Andererseits ist es mit gewissem Aufwand verbunden, eine Tabellen- und Bildunterschrift zu platzieren oder diese Objekte zu nummerieren und gar in eine Liste von Tabellen und Abbildungen einzutragen. Wie man die Listen erstellt, haben wir bereits in Abschnitt 7.7 diskutiert.

table figure

Um diese Arbeit zu erleichtern, stellt LaTeX die beiden Gleitobjektumgebungen `table` und `figure` zur Verfügung:

```
> \begin{table}[b]
   \centering
   \begin{tabular}{|l|l|l|} \hline
    eins & zwei & drei \\ \hline
   \end{tabular}
   \caption{Die erste Tabelle in einer Gleitumgebung.} \label{tab:eins}
  \end{table}
  \begin{figure}[t]
   \centering
   \includegraphics{quad.eps}
   \caption{Das erste Bild in einer Gleitumgebung.} \label{fig:quad}
  \end{figure}
```

\clearpage \newpage

LaTeX versucht, die gleitende Tabelle und das gleitende Bild an eine geeignete Position zu setzen. Durch die Angabe `b` für „bottom“ und `t` für „top“ im Beispiel wird die Tabelle ans Ende der Seite und das Bild an den Anfang der Seite gesetzt. Zwei weitere Einstellungen sind `h` für „here“ und `p` für „page“; letzteres bewirkt, dass eine gesonderte Seite mit Gleitobjekten angelegt wird. Bei der Verwendung von Gleitobjekten manifestiert sich auch der Unterschied zwischen `\clearpage` und `\newpage`: Der erste Befehl arbeitet zunächst alle aufgelaufenen Gleitobjekte ab, indem er so viele Gleitobjektseiten wie nötig anlegt, und eröffnet erst danach die neue Seite für den Fließtext. Meistens ist `\clearpage` die bessere Wahl.

\caption table figure tablename figurename

Wenn mit dem Befehl `\caption` eine Tabellen- bzw. Bildunterschrift angegeben wird, dann erscheint diese mit einer automatisch generierten Nummer unter dem Objekt. Die Zähler lauten `table` und `figure`, deren Formatierungen `\thetable` und `\thefigure` und die Befehle für die Wörter „Tabelle“ und „Abbildung“ `\tablename` und `\figurename`. Wie üblich können die Befehle `\label` und `\ref` auf die automatisch generierte Nummer angewendet werden.

eins	zwei	drei

Tabelle 8.1: Die erste Tabelle in einer Gleitumgebung.

Tabelle

Tabelle 8.2a: erste Tabelle in `subtables`

Tabelle

Tabelle 8.2b: zweite Tabelle in `subtables`

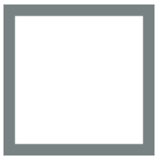


Abbildung 8.2a: erste Abbildung in `subfigures`

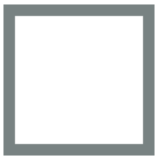


Abbildung 8.2b: zweite Abbildung in `subfigures`

Es können auch mehrere gewünschte Positionen als Argument übergeben werden, z. B. `hbp`. Dies bewirkt (wegen des `h`), dass das Gleitobjekt an der Stelle, wo es definiert ist, eingefügt wird, sofern LaTeX dies vertreten kann, d. h. wenn nicht zu viel Leerraum entsteht. Stellt LaTeX jedoch fest, dass die Positionierung `h` nicht geeignet ist, versucht es `b` oder im Zweifelsfall auch `p`. Die Standardeinstellung ist `tbp`.

Der Algorithmus zur Platzierung der Gleitobjekte ist relativ komplex; insbesondere wird er von einigen Zählern, Befehlen und Längen gesteuert. Darunter fallen u. a.:

topnumber bottomnumber totalnumber \topfraction \bottomfraction \textfraction \floatsep \textfloatsep \intextsep subtables subfigures

`topnumber`, `bottomnumber`, `totalnumber`: Diese drei Zähler geben an, wie viele Gleitobjekte maximal am Seitenanfang, am Seitenende bzw. insgesamt auf einer Seite platziert werden dürfen.

`\topfraction`, `\bottomfraction`: Diese beiden Befehle geben den Anteil an, der maximal durch Gleitobjekte oben bzw. unten belegt werden darf.

`\textfraction`: Dieser Befehl gibt den Anteil an, der minimal durch Text belegt werden muss.

`\floatsep`, `\textfloatsep`, `\intextsep`: Diese drei Gummilängen geben den Abstand zwischen zwei `t`/`b`-Gleitobjekten, den Abstand eines `t`/`b`-Gleitobjekts zum Text bzw. den Abstand eines `h`-Gleitobjekts vom Text an.

Ähnlich wie bei den Gleichungsnummern kann man mit dem Paket subfloat Unternummern bei den gleitenden Tabellen und Abbildungen erstellen:

```
> \begin{subtables}
   \begin{table}[t]
    \centering \begin{tabular}{|c|} Tabelle \end{tabular}
    \caption{erste Tabelle in \texttt{subtables}}
   \end{table}
   \begin{table}[t]
    \centering \begin{tabular}{|c|} Tabelle \end{tabular}
    \caption{zweite Tabelle in \texttt{subtables}}
```

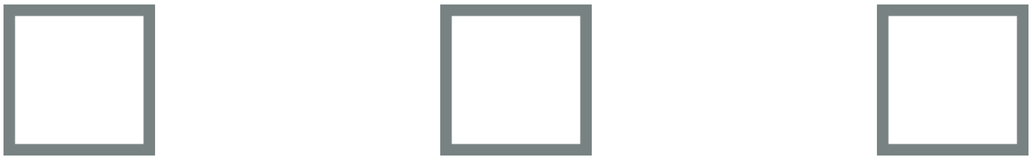
```
  \end{table}
 \end{subtables}
 \begin{subfigures}
  \begin{figure}[t]
   \centering \includegraphics{quad.eps}
   \caption{erste Abbildung in \texttt{subfigures}}
  \end{figure}
  \begin{figure}[t]
   \centering \includegraphics{quad.eps}
   \caption{zweite Abbildung in \texttt{subfigures}}
  \end{figure}
 \end{subfigures}
```

Tabellen bzw. Abbildungen innerhalb dieser Umgebungen erhalten einen Kleinbuchstaben hinter der Nummer, der hochgezählt wird. Die Platzierung der Gleitumgebungen wird nicht beeinflusst.

8.4.1 Ergänzungen

\subfloat Das Paket subfig hingegen kann mehrere unternummerierte Abbildungen in einer `figure`-Umgebung verwalten:

```
> \begin{figure}[h]
   \centering
   \subfloat{\hspace*{1cm}\includegraphics{quad.eps}\hspace*{1cm}}\hfill
   \subfloat[Unterschrift]{\label{subfig:quadb}%
    \hspace*{1cm}\includegraphics{quad.eps}\hspace*{1cm}}\hfill
   \subfloat[Listeneintrag][Unterschrift]{\label{subfig:quadc}%
    \hspace*{1cm}\includegraphics{quad.eps}\hspace*{1cm}}
   \caption{Abbildung mit \textsf{subfig}} \label{fig:quad}
  \end{figure}
```



(b) Unterschrift (c) Unterschrift

Abbildung 8.3: Abbildung mit subfig

Die Gleitumgebung besteht aus drei Befehlen `\subfloat` getrennt durch zwei `\hfill`. Das zwingende Argument von `\subfloat` enthält das einzubindende Bild. Im Beispiel wurde links und rechts 1 cm Leerraum eingefügt, damit die Beschriftung ohne Umbruch unter das Bild passt. Im optionalen Argument kann eine Bildunterschrift generiert werden; einen eventuellen `\label`-Befehl platziert man am besten als erstes im zwingenden Argument.

lofdepth Wird vor diesem optionalen Argument ein weiteres optionales angegeben, so legt dieses den Text fest, der im Abbildungsverzeichnis eingetragen wird. Damit diese Einträge aber überhaupt generiert werden, muss man den Zähler `lofdepth` – analog zum Zähler `tocdepth` beim Inhaltsverzeichnis – mit

```
> \setcounter{lofdepth}{2}
```

mindestens auf 2 setzen. Das Abbildungsverzeichnis befindet sich zu Demonstrationszwecken am Ende vom Dokument.

Selbstverständlich kann wie üblich mit dem Befehl `\ref` eine Referenz generiert werden. Der Reihe nach erzeugen `\ref{fig:quad}`, `\ref{subfig:quadb}` und `\ref{subfig:quadc}` die Ausgaben 8.3, 8.3b und 8.3c. Alternativ kann man `\subref` benutzen, um mit `\subref {subfig:quadb}` und `\subref{subfig:quadc}` die geklammerten Ausgaben (b) und (c) zu erzeugen, die nicht die Nummer der Abbildung an sich enthalten. \ref \subref

Auch wenn das Paket subfig heißt, kann der Befehl `\subfloat` genausogut in einer `table`-Umgebung benutzt werden. Der Zähler, der verändert werden muss, damit die untergeordneten Tabellen im Tabellenverzeichnis sichtbar werden, heißt analog `lotdepth`. lotdepth

Das Paket float erlaubt das Erstellen eigener Gleitumgebungen. Die Syntax von `\newfloat` ist an die von `\newtheorem` angelehnt: \newfloat \floatname

```
> \newfloat{program}{tbp}{lop}[chapter]
```

Das erste Argument ist der Name der Umgebung, das zweite die Standardpositionierung, das dritte die Dateierweiterung (wie lot und lof) und das vierte (optionale) die Gliederungsstufe, in der die Gleitumgebungen nummeriert werden. Einen Namen weist man der Umgebung folgendermaßen zu:

```
> \floatname{program}{Java-Program}
```

Wie üblich erstellt man dann eine Gleitumgebung:

```
> \begin{program}[h]
   \begin{verbatim}
  public class HelloWorld {
    public static main void(String[] args) {
      System.out.println("Hallo Welt!");
    }
  }
   \end{verbatim}
   \caption{Hallo Welt}
  \end{program}
```

```
public class HelloWorld {
  public static main void(String[] args) {
    System.out.println("Hallo Welt!");
  }
}
```

Java-Programm 8.1: Hallo Welt

Anhang A

Informationen und Dokumentationen

A.1 Internetseiten

A.1.1 Allgemeines

DANTE e. V. (Deutschsprachige Anwendervereinigung TEX e. V.):
`http://www.dante.de/`
„DANTE, Deutschsprachige Anwendervereinigung TEX e. V., wurde am 14. April 1989 in Heidelberg gegründet. Der Zweck des Vereins ist die Betreuung und Beratung von TEX-Benutzern im gesamten deutschsprachigen Raum."

FAQ zu DANTE und TEX:
`http://www.dante.de/faq/de-tex-faq/`
„Fragen und Antworten über TEX, LATEX und DANTE e. V. – Dieser Text enthält häufig gestellte Fragen und passende Antworten zum TEX-System und DANTE e. V."

TUG (TEX Users Group):
`http://www.tug.org/`
„The TEX Users Group (TUG) was founded in 1980 for educational and scientific purposes, to provide an organization for those who have an interest in typography and font design, and are users of the TEX typesetting system invented by Donald Knuth."

CTAN (Comprehensive TEX Archive Network):
`http://www.ctan.org/`
„The Comprehensive TEX Archive Network is the authoritative collection of materials related to the TEX typesetting system."

TEX-Seite der AMS (American Mathematical Society):
`http://www.ams.org/tex/`
„The American Mathematical Society (AMS) offers access to a family of TEX-related products for the convenience of authors and MathSci searchers."

A.1.2 Einführungen

„The Not So Short Introduction to LATEX 2_ε":
`http://www.dante.de/CTAN/info/lshort/english/lshort.pdf`
Über 150 Seiten Einführung in LATEX

A.1.3 Listen und Übersichten

Comprehensive Symbol List:
`ftp://tug.ctan.org/pub/tex-archive/info/symbols/comprehensive/symbols-a4.pdf`
„This document lists 4947 symbols and the corresponding LaTeX commands that produce them. Some of these symbols are guaranteed to be available in every LaTeX 2_ε system; others require fonts and packages that may not accompany a given distribution and that therefore need to be installed.“

„Der LaTeX-Index – Eine Befehlsübersicht im World Wide Web“:
`http://www.weinelt.de/latex/`
Eine LaTeX-Befehlsübersicht von Dipl.-Inf. (FH) Jürgen Weinelt, kurz und knapp jeweils mit Beispiel

„The LaTeX Font Catalogue“:
`http://www.tug.dk/FontCatalogue/`
Eine umfangreiche Liste zu Fonts in LaTeX, mit weiteren Informationen zum Einbinden usw.

A.1.4 Kürzere, speziellere Dokumente

„Das LaTeX 2_ε-Sündenregister – oder – Veraltete Befehle, Pakete und andere Fehler“:
`ftp://ftp.tex.ac.uk/tex-archive/info/l2tabu/german/l2tabu.pdf`
Informationen über veraltete Befehle und Pakete und deren Ersetzungen

„LaTeX-Tipps“:
`http://www.math.uni-magdeburg.de/~mkoeppe/texpert.ps`
Ein paar gute Hinweise zur Satztechnik

„Math mode - v.2.41“
`ftp://tug.ctan.org/pub/tex-archive/info/math/voss/mathmode/Mathmode.pdf`
Über 100 Seiten zum mathematischen Formelsatz: schön anzuschauendes Dokument (Quelltextbeispiele sogar farbig), enthält aber auch noch nicht-AmsMath-kompatible Befehle

„Tabellensatz mit LaTeX“:
`http://www.dante.de/dante/events/mv37/programm/handouts/hoeppner-tabellen.pdf`
Befehle und Erweiterungen der `tabular`-Umgebung, Tabellen über mehrere Seiten

„A Tutorial on `\expandafter`“:
`http://www.tug.org/TUGboat/Articles/tb09-1/tb20bechtolsheim.pdf`
Eine Erklärung zu `\expandafter`, wie der Name schon sagt – für Experten

A.2 Dokumentationen von Paketen

Paket a4wide:
`http://mirror.ctan.org/macros/latex/contrib/a4wide/a4wide.pdf`

Paket amscd:
`http://www.tug.org/texlive/Contents/live/texmf-dist/doc/latex/amsmath/amscd.pdf`

Paket amsfonts:
`http://mirror.ctan.org/fonts/amsfonts/doc/amsfonts.pdf`

Paket amsmath:
`ftp://ftp.ams.org/ams/doc/amsmath/amsldoc.pdf`

Paket amssymb:
`http://mirror.ctan.org/fonts/amsfonts/doc/amssymb.pdf`
Paket amsxtra:
`http://www.tug.org/texlive/Contents/live/texmf-dist/doc/latex/amsmath/amsxtra.pdf`
Paket anysize:
`http://mirror.ctan.org/macros/latex/contrib/anysize/anysize.pdf`
Paket array:
`http://mirror.ctan.org/macros/latex/required/tools/array.pdf`
Paket bbm:
`http://mirror.ctan.org/macros/latex/contrib/bbm/bbm.pdf`
Paket beamer:
`ftp://ftp.dante.de/pub/tex/macros/latex/contrib/beamer/doc/beameruserguide.pdf`
Vor allem muss ich das Kapitel 5: „Guidelines for creating presentations“ besonders empfehlen!
Paket bookmark:
`http://mirror.ctan.org/macros/latex/contrib/oberdiek/bookmark.pdf`
Paket color:
`http://mirror.ctan.org/macros/latex/required/graphics/grfguide.pdf`
Paket curves:
`http://mirror.ctan.org/macros/latex/contrib/curves/curves.pdf`
Paket dsfont:
`http://mirror.ctan.org/fonts/doublestroke/dsdoc.pdf`
Paket endnotes:
`http://mirror.ctan.org/macros/latex/contrib/endnotes/endnotes.pdf`
Paket eucal:
`http://mirror.ctan.org/fonts/amsfonts/doc/euscript.pdf`
Paket eurosym:
`http://mirror.ctan.org/fonts/eurosym/doc/testeuro.pdf`
Paket fancyhdr:
`http://mirror.ctan.org/macros/latex/contrib/fancyhdr/fancyhdr.pdf`
Paket float:
`http://mirror.ctan.org/macros/latex/contrib/float/float.pdf`
Paket footmisc:
`http://mirror.ctan.org/macros/latex/contrib/footmisc/footmisc.pdf`
Paket fp:
`http://www.tug.org/tetex/tetex-texmfdist/doc/latex/fp/readme.fp`
Paket geometry:
`http://mirror.ctan.org/macros/latex/contrib/geometry/geometry.pdf`
Paket graphicx:
`http://mirror.ctan.org/macros/latex/required/graphics/grfguide.pdf`
Paket hyperref:
`http://mirror.ctan.org/macros/latex/contrib/hyperref/doc/manual.pdf`
Paket imakeidx:
`http://mirror.ctan.org/macros/latex/contrib/imakeidx/imakeidx.pdf`

Paket inputenc:
`http://www.tug.org/texmf-dist/doc/latex/base/inputenc.pdf`
Paket lastpage:
`http://mirror.ctan.org/macros/latex/contrib/lastpage/lastpage.pdf`
Paket lettrine:
`http://mirror.ctan.org/macros/latex/contrib/lettrine/doc/lettrine.pdf`
Paket listings:
`http://mirror.ctan.org/macros/latex/contrib/listings/listings.pdf`
Paket lscape:
`http://mirror.ctan.org/macros/latex/required/graphics/grfguide.pdf`
Paket mathdots:
`http://mirror.ctan.org/macros/generic/mathdots/mathdots.pdf`
Paket mathrsfs:
`http://mirror.ctan.org/macros/latex/contrib/jknappen/mathrsfs.rme`
Paket mathtools:
`http://mirror.ctan.org/macros/latex/contrib/mh/mathtools.pdf`
Paket multicol:
`http://mirror.ctan.org/macros/latex/required/tools/multicol.pdf`
Paket musixtex:
`http://icking-music-archive.org/software/musixtex/musixdoc.pdf`
Paket ntheorem:
`http://mirror.ctan.org/macros/latex/contrib/ntheorem/ntheorem.pdf`
Paket parskip:
`http://mirror.ctan.org/macros/latex/contrib/parskip/parskip-doc.pdf`
Paket pst-uml:
`http://mirror.ctan.org/graphics/pstricks/contrib/pst-uml/pst-uml-doc.pdf`
Paket relsize:
`http://mirror.ctan.org/macros/latex/contrib/relsize/relsize-doc.pdf`
Paket shapepar:
`http://mirror.ctan.org/macros/latex/contrib/shapepar/shapepar.pdf`
Paket soul:
`http://mirror.ctan.org/macros/latex/contrib/soul/soul.pdf`
Paket stmaryrd:
`http://mirror.ctan.org/fonts/stmaryrd/stmaryrd.pdf`
Paket subfig:
`http://mirror.ctan.org/macros/latex/contrib/subfig/subfig.pdf`
Paket subfloat:
`http://mirror.ctan.org/macros/latex/contrib/subfloat/subfloat.pdf`
Paket ulem:
`http://mirror.ctan.org/macros/latex/contrib/ulem/ulem.pdf`
Paket undertilde:
`http://mirror.ctan.org/macros/latex/contrib/undertilde/undertilde.pdf`

Paket varioref:
http://mirror.ctan.org/macros/latex/required/tools/varioref.pdf
Paket wrapfig:
http://mirror.ctan.org/macros/latex/contrib/wrapfig/wrapfig-doc.pdf
Paket xy:
http://mirror.ctan.org/macros/generic/diagrams/xypic/xy-3.8.6/doc/xyrefer.pdf

A.3 Kollisionen von Paketen

Das Paket musixtex kollidiert mit den Grundpaketen in den Befehlen `\breve` und `\cup`. Abhilfe kann man sich schaffen, indem man das Paket musixtex kopiert und z. B. `\breve` in `\Breve` und `\cup` in `\cuP` umbenennt. (Achtung: `\Cup` wird schon von einem anderen Paket definiert!)

Das Paket musixtex kollidiert mit dem Paket xy im Befehl `\curve`. Abhilfe kann man sich schaffen, indem man das Paket musixtex kopiert und z. B. `\curve` in `\Curve` umbenennt.

A.4 Dateien eines LaTeX-Projekts

LaTeX legt eine ganze Reihe von Dateien mit dem Namen der compilierten Hauptdatei und mit unterschiedlicher Erweiterung an. Diese sind:

- aux: Erstellt und gelesen von latex/pdflatex, steht für „auxiliary“ und ist eine Hilfsdatei, die vor allem die Querverweise organisiert, und wenn `\includeonly` benutzt wird, auch die Zähler.
- dvi: Erstellt von latex, steht für „device independent“ und ist das Gerät-unabhängige Ausgabeformat, was immer als erstes erstellt wird, sofern es nicht PDF ist.
- ent: Erstellt und gelesen von latex/pdflatex, steht für „endnotes“ und organisiert die Endnoten.
- idx: Erstellt von latex/pdflatex, gelesen von makeindex, steht für „index“ und enthält die Einträge fürs Stichwortverzeichnis in der Rohform, insbesondere unsortiert.
- ilg: Erstellt von makeindex, steht für „index logfile“ und enthält Informationen über den Ablauf dieses Programms.
- ind: Erstellt von makeindex, gelesen von latex/pdflatex, steht für „index“ und enthält die Einträge fürs Stichwortverzeichnis in aufbereiteter Form, d. h. zusammengefasst und sortiert.
- lof: Erstellt und gelesen von latex/pdflatex, steht für „list of figures“ und organisiert das Abbildungsverzeichnis.
- log: Erstellt von latex/pdflatex, steht für „logfile“ und enthält Informationen über den Ablauf dieses Programms.
- lot: Erstellt und gelesen von latex/pdflatex, steht für „list of tables“ und organisiert das Tabellenverzeichnis.
- mx1: Erstellt von latex/pdflatex, gelesen von musixflx, steht für „musics 1“ und ist die Eingabedatei für die Blocksatzformatierung für Notenlinien.
- mx2: Erstellt von musixflx, gelesen von latex/pdflatex, steht für „musics 2“ und ist die Ausgabedatei für die Blocksatzformatierung für Notenlinien.

- pdf: Erstellt von pdflatex aus tex oder von ps2pdf aus ps, steht für „portable document format“ und ist das bekannte PDF-Ausgabeformat.
- ps: Erstellt von dvips aus dvi, steht für „postscript“ und ist das Postscript-Ausgabeformat.
- tex: Erstellt von uns, gelesen von latex/pdflatex, steht für „latex“ und ist die Eingabe-Hauptdatei.
- thm: Erstellt und gelesen von latex/pdflatex, steht für „theorems“ und organisiert das Verzeichnis der Theoreme.
- toc: Erstellt und gelesen von latex/pdflatex, steht für „table of contents“ und organisiert das Inhaltsverzeichnis.

A.5 Seitenaufbau

In Abbildung A.1 ist der Aufbau einer Seite mit allen Längen und deren Standardeinstellungen für ein article-Dokument in 11pt dargestellt. Die Darstellung ist auf 70 % skaliert.

A.6 Ein- und zweibuchstabige Befehle

Die folgende Liste enthält alle Namen, die man nicht für eigene Befehle, Längen und Umgebungen benutzen kann, weil sie in diversen Paketen schon definiert sind. Grundsätzlich sollte man keine einbuchstabigen Namen an eigene Befehle vergeben.

<table>
<tr><td>!</td><td>]</td><td>do</td><td>ln</td><td>sb</td></tr>
<tr><td>"</td><td>^</td><td>dp</td><td>lq</td><td>sc</td></tr>
<tr><td>#</td><td>_</td><td>dq</td><td>mp</td><td>sf</td></tr>
<tr><td>$</td><td>`</td><td>em</td><td>mu</td><td>sl</td></tr>
<tr><td>%</td><td>{</td><td>fi</td><td>ne</td><td>so</td></tr>
<tr><td>&</td><td>|</td><td>ge</td><td>NG</td><td>sp</td></tr>
<tr><td>'</td><td>}</td><td>gg</td><td>ng</td><td>SS</td></tr>
<tr><td>(</td><td>~</td><td>H</td><td>ni</td><td>ss</td></tr>
<tr><td>)</td><td>a</td><td>hl</td><td>nu</td><td>st</td></tr>
<tr><td>*</td><td>AA</td><td>ht</td><td>O</td><td>t</td></tr>
<tr><td>+</td><td>aa</td><td>i</td><td>o</td><td>TH</td></tr>
<tr><td>,</td><td>AE</td><td>if</td><td>OE</td><td>th</td></tr>
<tr><td>-</td><td>ae</td><td>IJ</td><td>oe</td><td>to</td></tr>
<tr><td>.</td><td>ar</td><td>ij</td><td>or</td><td>tt</td></tr>
<tr><td>/</td><td>b</td><td>Im</td><td>P</td><td>u</td></tr>
<tr><td>1</td><td>bf</td><td>in</td><td>Pi</td><td>ul</td></tr>
<tr><td>2</td><td>c</td><td>it</td><td>pi</td><td>v</td></tr>
<tr><td>:</td><td>CD</td><td>j</td><td>pm</td><td>wd</td></tr>
<tr><td>;</td><td>cr</td><td>k</td><td>Pr</td><td>wp</td></tr>
<tr><td>=</td><td>d</td><td>L</td><td>r</td><td>wr</td></tr>
<tr><td>></td><td>DH</td><td>l</td><td>Re</td><td>Xi</td></tr>
<tr><td>@</td><td>dh</td><td>le</td><td>rm</td><td>xi</td></tr>
<tr><td>[</td><td>DJ</td><td>lg</td><td>rq</td><td>Xy</td></tr>
<tr><td>\</td><td>dj</td><td>ll</td><td>S</td><td>xy</td></tr>
</table>

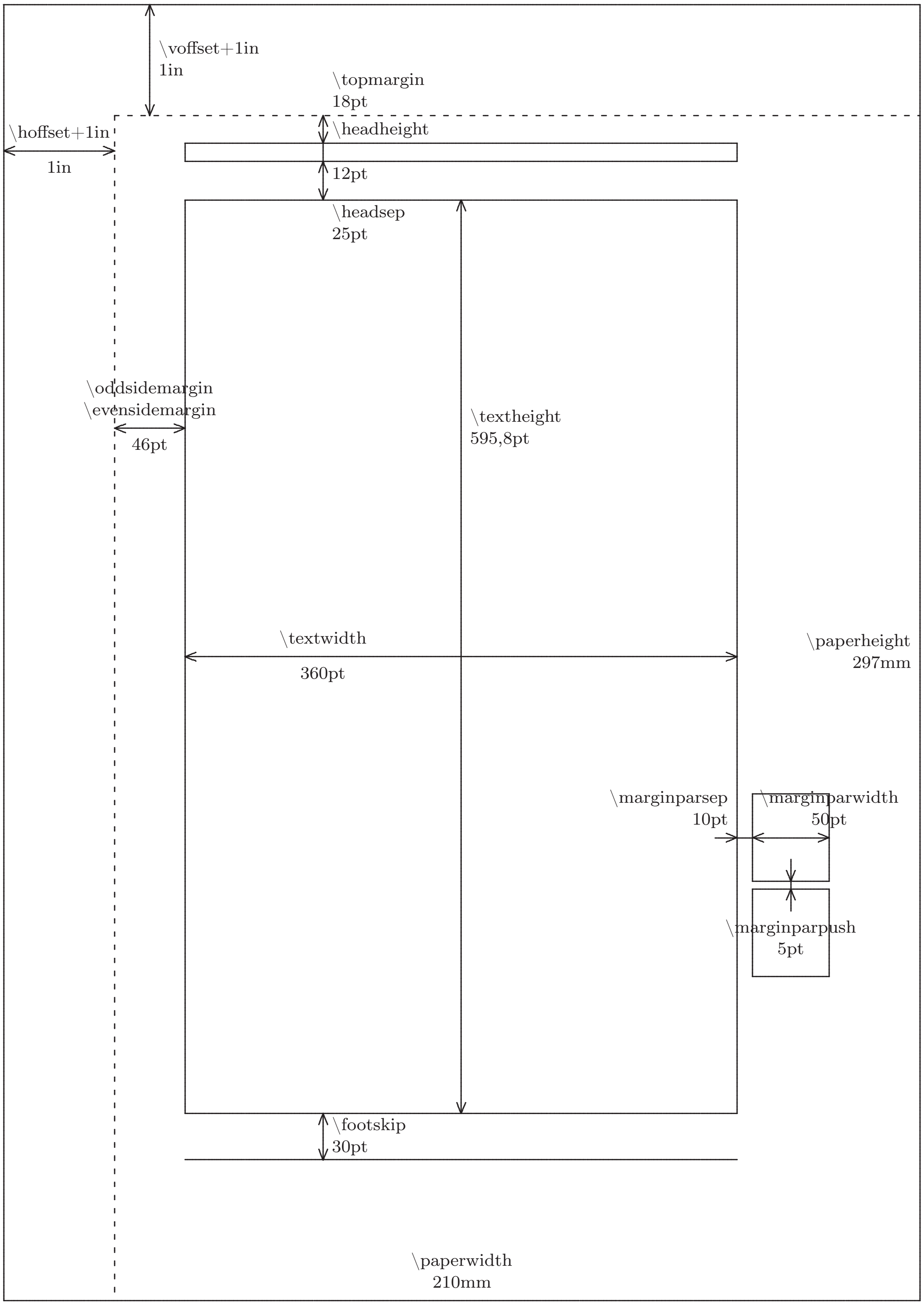


Abbildung A.1: Aufbau einer Seite

Anhang B

Liste der mathematischen Symbole

Das hochgestellte zweibuchstabige Kürzel gibt das Paket an:
[am]: amsmath
[as]: amssymb
[st]: stmaryrd

B.1 Buchstaben und Ziffern

0	`0`
1	`1`
2	`2`
3	`3`
4	`4`
5	`5`
6	`6`
7	`7`
8	`8`
9	`9`
A	`A`
a	`a`
α	`\alpha`
B	`B`
b	`b`
β	`\beta`
C	`C`
c	`c`
χ	`\chi`
D	`D`
d	`d`
Δ	`\Delta`
δ	`\delta`
E	`E`
e	`e`
ℓ	`\ell`
ϵ	`\epsilon`
η	`\eta`
F	`F`
f	`f`
G	`G`
g	`g`
Γ	`\Gamma`
γ	`\gamma`
H	`H`
h	`h`
$\hbar$	`\hbar`
$\hslash$	`\hslash`
I	`I`
i	`i`
$\imath$	`\imath`
ι	`\iota`
J	`J`
j	`j`
$\jmath$	`\jmath`
K	`K`
k	`k`
κ	`\kappa`
L	`L`
l	`l`
Λ	`\Lambda`
λ	`\lambda`
M	`M`
m	`m`
μ	`\mu`
N	`N`
n	`n`
ν	`\nu`
O	`O`
o	`o`
Ω	`\Omega`
ω	`\omega`
P	`P`
p	`p`
Φ	`\Phi`
ϕ	`\phi`
Π	`\Pi`
π	`\pi`
Ψ	`\Psi`
ψ	`\psi`
Q	`Q`
q	`q`
R	`R`
r	`r`
ρ	`\rho`
S	`S`
s	`s`
Σ	`\Sigma`
σ	`\sigma`
T	`T`
t	`t`
τ	`\tau`
Θ	`\Theta`
θ	`\theta`

U	`U`	φ	`\varphi`	w	`w`
u	`u`	$\varPi$	`\varPi`[as]	X	`X`
Υ	`\Upsilon`	ϖ	`\varpi`	x	`x`
υ	`\upsilon`	$\varPsi$	`\varPsi`[as]	Ξ	`\Xi`
V	`V`	ϱ	`\varrho`	ξ	`\xi`
v	`v`	$\varSigma$	`\varSigma`[as]	Y	`Y`
ε	`\varepsilon`	ς	`\varsigma`	y	`y`
$\varDelta$	`\varDelta`[as]	$\varTheta$	`\varTheta`[as]	Z	`Z`
$\varGamma$	`\varGamma`[as]	ϑ	`\vartheta`	z	`z`
$\varLambda$	`\varLambda`[as]	$\varUpsilon$	`\varUpsilon`[as]	ζ	`\zeta`
$\varOmega$	`\varOmega`[as]	$\varXi$	`\varXi`[as]		
$\varPhi$	`\varPhi`[as]	W	`W`		

B.2 Abstände und Interpunktionen

$a\!b$	`\!`	$a\:b$	`\:`	$a~b$	`~`
$,$	`,`	$;$	`;`	$a\qquad b$	`\qquad`
$a\,b$	`\,`	$a\;b$	`\;`	$a\quad b$	`\quad`
$.$	`.`	$a\>b$	`\>`		

B.3 Verknüpfende Symbole

$*$	`*`	$\bullet$	`\bullet`	$\leftslice$	`\leftslice`[st]
$+$	`+`	$\Cap$	`\Cap`[as]	$\leftthreetimes$	`\leftthreetimes`[as]
$-$	`-`	$\cap$	`\cap`	$\lor$	`\lor`
$/$	`/`	$\cdot$	`\cdot`	$\ltimes$	`\ltimes`[as]
$:$	`:`	$\centerdot$	`\centerdot`[as]	$\merge$	`\merge`[st]
$\amalg$	`\amalg`	$\circ$	`\circ`	$\minuso$	`\minuso`[st]
$\ast$	`\ast`	$\circledast$	`\circledast`[as]	$\moo$	`\moo`[st]
$\baro$	`\baro`[st]	$\circledcirc$	`\circledcirc`[as]	$\mp$	`\mp`
$\barwedge$	`\barwedge`[as]	$\circleddash$	`\circleddash`[as]	$\nplus$	`\nplus`[st]
$\bbslash$	`\bbslash`[st]	$\Cup$	`\Cup`[as]	$\obar$	`\obar`[st]
$\bigcirc$	`\bigcirc`	$\cup$	`\cup`	$\oblong$	`\oblong`[st]
$\bigtriangledown$	`\bigtriangledown`	$\curlyvee$	`\curlyvee`[as]	$\obslash$	`\obslash`[st]
$\bigtriangleup$	`\bigtriangleup`	$\curlywedge$	`\curlywedge`[as]	$\odot$	`\odot`
$\binampersand$	`\binampersand`[st]	$\dagger$	`\dagger`	$\ogreaterthan$	`\ogreaterthan`[st]
$\bindnasrepma$	`\bindnasrepma`[st]	$\ddagger$	`\ddagger`	$\olessthan$	`\olessthan`[st]
$\boxast$	`\boxast`[st]	$\diamond$	`\diamond`	$\ominus$	`\ominus`
$\boxbar$	`\boxbar`[st]	$\div$	`\div`	$\oplus$	`\oplus`
$\boxbox$	`\boxbox`[st]	$\divideontimes$	`\divideontimes`[as]	$\oslash$	`\oslash`
$\boxbslash$	`\boxbslash`[st]	$\dotplus$	`\dotplus`[as]	$\otimes$	`\otimes`
$\boxcircle$	`\boxcircle`[st]	$\doublebarwedge$	`\doublebarwedge`[as]	$\ovee$	`\ovee`[st]
$\boxdot$	`\boxdot`[as]	$\fatbslash$	`\fatbslash`[st]	$\owedge$	`\owedge`[st]
$\boxempty$	`\boxempty`[st]	$\fatsemi$	`\fatsemi`[st]	$\pm$	`\pm`
$\boxminus$	`\boxminus`[as]	$\fatslash$	`\fatslash`[st]	$\rightslice$	`\rightslice`[st]
$\boxplus$	`\boxplus`[as]	$\intercal$	`\intercal`[as]	$\rightthreetimes$	`\rightthreetimes`[as]
$\boxslash$	`\boxslash`[st]	$\interleave$	`\interleave`[st]	$\rtimes$	`\rtimes`[as]
$\boxtimes$	`\boxtimes`[as]	$\land$	`\land`	$\setminus$	`\setminus`

$\smallsetminus$	`\smallsetminus`[as]	$\varoast$	`\varoast`[st]	$\varovee$	`\varovee`[st]
$\sqcap$	`\sqcap`	$\varobar$	`\varobar`[st]	$\varowedge$	`\varowedge`[st]
$\sqcup$	`\sqcup`	$\varobslash$	`\varobslash`[st]	$\vartimes$	`\vartimes`[st]
$\sslash$	`\sslash`[st]	$\varocircle$	`\varocircle`[st]	$\vee$	`\vee`
$\star$	`\star`	$\varodot$	`\varodot`[st]	$\veebar$	`\veebar`[as]
$\talloblong$	`\talloblong`[st]	$\varogreaterthan$	`\varogreaterthan`[st]	$\wedge$	`\wedge`
$\times$	`\times`	$\varolessthan$	`\varolessthan`[st]	$\wr$	`\wr`
$\uplus$	`\uplus`	$\varominus$	`\varominus`[st]	$\Ydown$	`\Ydown`[st]
$\varbigcirc$	`\varbigcirc`[st]	$\varoplus$	`\varoplus`[st]	$\Yleft$	`\Yleft`[st]
$\varcurlyvee$	`\varcurlyvee`[st]	$\varoslash$	`\varoslash`[st]	$\Yright$	`\Yright`[st]
$\varcurlywedge$	`\varcurlywedge`[st]	$\varotimes$	`\varotimes`[st]	$\Yup$	`\Yup`[st]

B.4 Vergleichende Symbole

$<$	`<`	$\lll$	`\lll`[as]	$\precnapprox$	`\precnapprox`[as]
$=$	`=`	$\lnapprox$	`\lnapprox`[as]	$\precnsim$	`\precnsim`[as]
$>$	`>`	$\lneq$	`\lneq`[as]	$\precsim$	`\precsim`[as]
$\approx$	`\approx`	$\lneqq$	`\lneqq`[as]	$\propto$	`\propto`
$\asymp$	`\asymp`	$\lnsim$	`\lnsim`[as]	$\rhd$	`\rhd`[as]
$\cong$	`\cong`	$\mid$	`\mid`	$\sim$	`\sim`
$\equiv$	`\equiv`	$\ncong$	`\ncong`[as]	$\simeq$	`\simeq`
$\frown$	`\frown`	$\ne$	`\ne`	$\smile$	`\smile`
$\ge$	`\ge`	$\neq$	`\neq`	$\sqsubset$	`\sqsubset`
$\geq$	`\geq`	$\ngeq$	`\ngeq`[as]	$\sqsubseteq$	`\sqsubseteq`
$\geqq$	`\geqq`[as]	$\ngeqslant$	`\ngeqslant`[as]	$\sqsupset$	`\sqsupset`
$\geqslant$	`\geqslant`[as]	$\ngtr$	`\ngtr`[as]	$\sqsupseteq$	`\sqsupseteq`
$\gg$	`\gg`	$\ni$	`\ni`	$\Subset$	`\Subset`[as]
$\ggg$	`\ggg`[as]	$\nleq$	`\nleq`[as]	$\subset$	`\subset`
$\gnapprox$	`\gnapprox`[as]	$\nleqslant$	`\nleqslant`[as]	$\subseteq$	`\subseteq`
$\gneq$	`\gneq`[as]	$\nless$	`\nless`[as]	$\subseteqq$	`\subseteqq`[as]
$\gneqq$	`\gneqq`[as]	$\nmid$	`\nmid`[as]	$\subsetneq$	`\subsetneq`[as]
$\gnsim$	`\gnsim`[as]	$\notin$	`\notin`	$\subsetneqq$	`\subsetneqq`[as]
$\gtrapprox$	`\gtrapprox`[as]	$\nparallel$	`\nparallel`[as]	$\succ$	`\succ`
$\gtreqless$	`\gtreqless`[as]	$\nprec$	`\nprec`[as]	$\succapprox$	`\succapprox`[as]
$\gtreqqless$	`\gtreqqless`[as]	$\npreceq$	`\npreceq`[as]	$\succcurlyeq$	`\succcurlyeq`[as]
$\gtrless$	`\gtrless`[as]	$\nsim$	`\nsim`[as]	$\succeq$	`\succeq`
$\gtrsim$	`\gtrsim`[as]	$\nsubseteq$	`\nsubseteq`[as]	$\succnapprox$	`\succnapprox`[as]
$\in$	`\in`	$\nsubseteqq$	`\nsubseteqq`[as]	$\succnsim$	`\succnsim`[as]
$\le$	`\le`	$\nsucc$	`\nsucc`[as]	$\succsim$	`\succsim`[as]
$\leq$	`\leq`	$\nsucceq$	`\nsucceq`[as]	$\Supset$	`\Supset`[as]
$\leqq$	`\leqq`[as]	$\nsupseteq$	`\nsupseteq`[as]	$\supset$	`\supset`
$\leqslant$	`\leqslant`[as]	$\nsupseteqq$	`\nsupseteqq`[as]	$\supseteq$	`\supseteq`
$\lessapprox$	`\lessapprox`[as]	$\owns$	`\owns`	$\supseteqq$	`\supseteqq`[as]
$\lesseqgtr$	`\lesseqgtr`[as]	$\parallel$	`\parallel`	$\supsetneq$	`\supsetneq`[as]
$\lesseqqgtr$	`\lesseqqgtr`[as]	$\perp$	`\perp`	$\supsetneqq$	`\supsetneqq`[as]
$\lessgtr$	`\lessgtr`[as]	$\prec$	`\prec`	$\triangleleft$	`\triangleleft`
$\lesssim$	`\lesssim`[as]	$\precapprox$	`\precapprox`[as]	$\triangleright$	`\triangleright`
$\lhd$	`\lhd`[as]	$\preccurlyeq$	`\preccurlyeq`[as]	$\unlhd$	`\unlhd`[as]
$\ll$	`\ll`	$\preceq$	`\preceq`	$\unrhd$	`\unrhd`[as]

B.5 Pfeile

$\circlearrowleft$	`\circlearrowleft`[as]	$\Leftrightarrow$	`\Leftrightarrow`	$\Rightarrow$	`\Rightarrow`
$\circlearrowright$	`\circlearrowright`[as]	$\leftrightarrow$	`\leftrightarrow`	$\rightarrow$	`\rightarrow`
$\curlyveedownarrow$	`\curlyveedownarrow`[st]	$\leftrightarrows$	`\leftrightarrows`[as]	$\rightarrowtail$	`\rightarrowtail`[as]
$\curlyveeuparrow$	`\curlyveeuparrow`[st]	$\leftrightharpoons$	`\leftrightharpoons`[as]	$\rightarrowtriangle$	`\rightarrowtriangle`[st]
$\curlywedgedownarrow$	`\curlywedgedownarrow`[st]	$\leftrightsquigarrow$	`\leftrightsquigarrow`[as]	$\rightharpoondown$	`\rightharpoondown`
$\curlywedgeuparrow$	`\curlywedgeuparrow`[st]	$\Lleftarrow$	`\Lleftarrow`[as]	$\rightharpoonup$	`\rightharpoonup`
$\curvearrowleft$	`\curvearrowleft`[as]	$\Longleftarrow$	`\Longleftarrow`	$\rightleftarrows$	`\rightleftarrows`[as]
$\curvearrowright$	`\curvearrowright`[as]	$\longleftarrow$	`\longleftarrow`	$\rightleftharpoons$	`\rightleftharpoons`
$\dashleftarrow$	`\dashleftarrow`[as]	$\Longleftrightarrow$	`\Longleftrightarrow`	$\rightrightarrows$	`\rightrightarrows`[as]
$\dashrightarrow$	`\dashrightarrow`[as]	$\longleftrightarrow$	`\longleftrightarrow`	$\rightsquigarrow$	`\rightsquigarrow`[as]
$\Downarrow$	`\Downarrow`	$\Longmapsfrom$	`\Longmapsfrom`[st]	$\Rrightarrow$	`\Rrightarrow`[as]
$\downarrow$	`\downarrow`	$\longmapsfrom$	`\longmapsfrom`[st]	$\searrow$	`\searrow`
$\downdownarrows$	`\downdownarrows`[as]	$\Longmapsto$	`\Longmapsto`[st]	$\ssearrow$	`\ssearrow`[st]
$\downharpoonleft$	`\downharpoonleft`[as]	$\longmapsto$	`\longmapsto`	$\sswarrow$	`\sswarrow`[st]
$\downharpoonright$	`\downharpoonright`[as]	$\Longrightarrow$	`\Longrightarrow`	$\swarrow$	`\swarrow`
$\gets$	`\gets`	$\longrightarrow$	`\longrightarrow`	$\to$	`\to`
$\hookleftarrow$	`\hookleftarrow`	$\looparrowleft$	`\looparrowleft`[as]	$\twoheadleftarrow$	`\twoheadleftarrow`[as]
$\hookrightarrow$	`\hookrightarrow`	$\looparrowright$	`\looparrowright`[as]	$\twoheadrightarrow$	`\twoheadrightarrow`[as]
$\iff$	`\iff`	$\Mapsfrom$	`\Mapsfrom`[st]	$\Uparrow$	`\Uparrow`
$\Leftarrow$	`\Leftarrow`	$\mapsfrom$	`\mapsfrom`[st]	$\uparrow$	`\uparrow`
$\leftarrow$	`\leftarrow`	$\Mapsto$	`\Mapsto`[st]	$\Updownarrow$	`\Updownarrow`
$\leftarrowtail$	`\leftarrowtail`[as]	$\mapsto$	`\mapsto`	$\updownarrow$	`\updownarrow`
$\leftarrowtriangle$	`\leftarrowtriangle`[st]	$\nearrow$	`\nearrow`	$\upharpoonleft$	`\upharpoonleft`[as]
$\leftharpoondown$	`\leftharpoondown`	$\nnearrow$	`\nnearrow`[st]	$\upharpoonright$	`\upharpoonright`[as]
$\leftharpoonup$	`\leftharpoonup`	$\nnwarrow$	`\nnwarrow`[st]	$\upuparrows$	`\upuparrows`[as]
$\leftleftarrows$	`\leftleftarrows`[as]	$\nwarrow$	`\nwarrow`		

B.6 Große Symbole

$\bigbox$	`\bigbox`[st]	$\bigotimes$	`\bigotimes`	$\coprod$	`\coprod`
$\bigcap$	`\bigcap`	$\bigparallel$	`\bigparallel`[st]	$\idotsint$	`\idotsint`[am]
$\bigcup$	`\bigcup`	$\bigsqcap$	`\bigsqcap`[st]	$\iiiint$	`\iiiint`[am]
$\bigcurlyvee$	`\bigcurlyvee`[st]	$\bigsqcup$	`\bigsqcup`	$\iiint$	`\iiint`[am]
$\bigcurlywedge$	`\bigcurlywedge`[st]	$\bigtriangledown$	`\bigtriangledown`[st]	$\iint$	`\iint`[am]
$\biginterleave$	`\biginterleave`[st]	$\bigtriangleup$	`\bigtriangleup`[st]	$\int$	`\int`
$\bignplus$	`\bignplus`[st]	$\biguplus$	`\biguplus`	$\oint$	`\oint`
$\bigodot$	`\bigodot`	$\bigvee$	`\bigvee`	$\prod$	`\prod`
$\bigoplus$	`\bigoplus`	$\bigwedge$	`\bigwedge`	$\sum$	`\sum`

B.7 Klammern

$($	`(`	$\{$	`\{`	$\Downarrow$	`\Downarrow`
$)$	`)`	$\vert$	`|`	$\downarrow$	`\downarrow`
$/$	`/`	$\Vert$	`|`	$\langle$	`\langle`
$[$	`[`	$\}$	`\}`	$\lceil$	`\lceil`
$]$	`]`	$\backslash$	`\backslash`	$\lfloor$	`\lfloor`

⟦	\llbracket[st]	⌉	\rceil	∣	\rvert
⌈⌈	\llceil[st]	⌋	\rfloor	⇑	\Uparrow
⌊⌊	\llfloor[st]	⟧	\rrbracket[st]	↑	\uparrow
⦅	\llparenthesis[st]	⌉⌉	\rrceil[st]	⇕	\Updownarrow
‖	\lVert	⌋⌋	\rrfloor[st]	↕	\updownarrow
∣	\lvert	⦆	\rrparenthesis[st]	‖	\Vert
⟩	\rangle	‖	\rVert	∣	\vert

B.8 Sonstige Symbole

!	!	†	\dagger	∡	\measuredangle[as]
”	"	‡	\ddagger	℧	\mho[as]
#	\#	♢	\diamondsuit	∇	\nabla
$	\$	∅	\emptyset	♮	\natural
%	\%	∃	\exists	¬	\neg
&	\&	♭	\flat	∄	\nexists[as]
’	'	∀	\forall	¶	\P
?	?	♡	\heartsuit	∂	\partial
@	@	ℑ	\Im	ℜ	\Re
_	_	∞	\infty	§	\S
‘	`	↯	\lightning[st]	♯	\sharp
ℵ	\aleph	¬	\lnot	♠	\spadesuit
⨿	\amalg	$	\mathdollar	∢	\sphericalangle[as]
∠	\angle	…	\mathellipsis	□	\square[as]
⊥	\bot	¶	\mathparagraph	√	\surd
♣	\clubsuit	§	\mathsection	⊤	\top
:	\colon	£	\mathsterling	∅	\varnothing[as]
∁	\complement[as]	_	\mathunderscore	℘	\wp

Tabellenverzeichnis

Abbildungsverzeichnis

Stichwortverzeichnis

LaTeX-Verzeichnis

Zähler in geneigter Serifenschrift
Pakete in aufrechter serifenloser Schrift
Paketoptionen (auch für \documentclass) in geneigter serifenloser Schrift
{Umgebungen} in aufrechter Serifenschrift mit geschweiften Klammern
\Befehle in aufrechter Serifenschrift mit Backslash
Längen in geneigter Serifenschrift mit Backslash