

Kapitelunterteilung

L^AT_EX erlaubt die Unterteilung des Textes in Kapitel, Abschnitte,...

```
\section{Titel}
\subsection{Unter-Titel}
\subsubsection{Unter-Unter-Titel}
\paragraph{Unter-Unter-Unter-Titel}
\subparagraph{Unter-Unter-Unter-Unter-Titel}
```

1. Titel

1.1 Unter-Titel

1.1.1 Unter-Unter-Titel

1.1.1.1 Unter-Unter-Unter-Titel

1.1.1.1.1 Unter-Unter-Unter-Unter-Titel

Titel und Inhaltsverzeichnis

- **Titelseite** — Eine Titelseite (bei der `article`-Klasse ein Titel über der ersten Seite) kann man mit folgenden Befehlen erstellen:

```
\title{...}
\author{...}
\date{...} ← optional
\maketitle
```

- **Inhaltsverzeichnis** — Das Erzeugen eines Inhaltsverzeichnisses erfolgt mit dem Befehl

```
\tableofcontents
```

Eine Aufstellung der Titel, die mit den “sectioning”-Befehlen erzeugt wurden, wird an der Stelle dieses Befehls eingefügt. Hierzu sind zwei (u.U. sogar drei) L^AT_EX-Läufe erforderlich.

Kapitelunterteilung

Bemerkungen zu den “sectioning”-Befehlen:

- Die Nummerierung erfolgt automatisch.
- Bei den Klassen `report` oder `book` wird “oberhalb” von `\section{...}` noch der Befehl `\chapter{...}` eingefügt.
- Es ist auch möglich, den Text mit dem Befehl `\part{...}` in Teile zu untergliedern. In jedem Teil beginnt die Nummerierung der Kapitel, Abschnitte, ... von Neuem.
- Der Befehl `\appendix` ändert die Nummerierung der obersten Ebene (`\chapter` bzw. `\section`) von Nummern in Buchstaben. I.a. wird das zur Bezeichnung der Anhänge genutzt.
- Mit einem nachgestellten `*` wird die Nummerierung unterdrückt: `\subsection*{Titel}`

Bsp.: Inhaltsverzeichnis

Inhaltsverzeichnis

1 Einführung	7
1.1 Dirichlet-Charaktere und L -Reihen	7
1.2 Grundlagen	14
1.3 Der Mittelwertsatz	16
2 Beweis des Mittelwertsatzes	19
2.1 Verschiebung des Integrationsweges	19
2.2 Anwendung der Funktionalgleichung	20

...

Ein Beispiel

Setzen wir all dies zu einem Beispiel zusammen:

```
\documentclass[12pt]{article}
\usepackage[latin1]{inputenc}
\usepackage{ngerman}
\usepackage{a4wide}
\begin{document}
\title{Aufgabe 84\Holomorphe Funktionen mit kompaktem Träger}
\author{Karl Nemo}
\date{Heute}
\maketitle
\tableofcontents
\section{Einleitung}
\subsection{Grundlagen}
...
\appendix
\section{Literaturverzeichnis}
...
\end{document}
```

Verweise

Ein großer Vorteil von L^AT_EX ist die automatische Nummerierung von Abschnitten etc. und der Umgang damit. Dies funktioniert wie die Nummerierung von Formeln:

Dies `\label{verweis}` steht in Abschnitt `\ref{verweis}` auf Seite `\pageref{verweis}`.

Dies steht in Abschnitt 6 auf Seite 4.

Auch hier sind zwei oder drei L^AT_EX-Läufe erforderlich, um die korrekten Verweise zu erhalten. L^AT_EX warnt bei Bedarf mit dem Hinweis

LaTeX Warning: Label(s) may have changed. Rerun to get cross-references right.

Aufgabe 84 Holomorphe Funktionen mit kompaktem Träger

Karl Nemo

Heute

Inhaltsverzeichnis

1 Einleitung	1
1.1 Grundlagen	1
A Literaturverzeichnis	1

1 Einleitung

1.1 Grundlagen

...

A Literaturverzeichnis

...

Fußnoten

Fußnoten sind recht einfach mit dem Befehl `\footnote{...}` zu erzeugen. Sie werden innerhalb des Dokuments automatisch durchnummeriert:

Fußnoten wirken `\footnote{zumindest auf den naiven Leser}` immer sehr wissenschaftlich, man kann es aber übertreiben.

Fußnoten wirken² immer sehr wissenschaftlich, man kann es aber übertreiben.

²zumindest auf den naiven Leser

Umgebungen

Umgebungen (`\begin{...}\end{...}`) haben wir bereits kennengelernt:

- `\begin{equation}... \end{equation}`
- `\begin{eqnarray}... \end{eqnarray}`
- `\begin{cases}... \end{cases}`
- `\begin{pmatrix}... \end{pmatrix}` und andere...

Sie werden aber auch benutzt, um besondere Textgestaltungen zu setzen:

- zentrierter Text
- verbatim-Texte
- Listen
- Tabellen

verbatim-Umgebung

\LaTeX setzt Texte gewöhnlich selbstständig, kann aber auch Text so setzen, wie er eingegeben wurde und ignoriert dabei \LaTeX -Befehle:

```
\begin{verbatim}
  Jede \LaTeX-Datei beginnt mit einem
  Befehl wie \documentclass[12pt]{article}.
\end{verbatim}
```

```
Jede \LaTeX-Datei beginnt mit einem
Befehl wie \documentclass[12pt]{article}.
```

Auch der Zeilenumbruch erfolgt hier nicht automatisch.

Zentrierter, links- oder rechtsbündiger Text

Ein Beispiel für zentrierten Text:

Gewöhnlich wird Text im Blocksatz gesetzt, es ist aber auch möglich,

```
\begin{center}
  den Text \\
  zentriert zu setzen.
\end{center}
```

Gewöhnlich wird Text im Blocksatz gesetzt, es ist aber auch möglich,

```
den Text
zentriert zu setzen.
```

Analog gibt es die Umgebungen `flushleft` und `flushright` für links- bzw. rechtsbündigen Satz:

verbatim-Befehl

Es müssen nicht immer ganze Absätze verbatim gesetzt werden:

Ähnlich zur `\verb+\verbatim+`-Umgebung gibt es den Befehl `\verb+\verb+`, der kurze Ausdrücke verbatim setzt.

Ähnlich zur `verbatim`-Umgebung gibt es den Befehl `\verb`, der kurze Ausdrücke verbatim setzt.

Der Text wird verbatim gesetzt, der zwischen den “+”-Zeichen steht. Als “Klammer”-Zeichen (hier “+”) kann jedes Zeichen außer Buchstaben, * und Leerstellen dienen (sofern es natürlich nicht im verbatim zu setzenden Text vorkommt).

Listen

Es gibt in \LaTeX drei Arten von Listen:

- gewöhnliche Listen
- nummerierte Listen
- benannte Listen

Gesetzt wurde diese “gewöhnliche” Liste mit:

```
\begin{itemize}
  \item gewöhnliche Listen
  \item nummerierte Listen
  \item benannte Listen
\end{itemize}
```

Verschachtelte Listen

Listen können natürlich auch ineinander verschachtelt werden:

```
\begin{itemize}
\item Erste Ebene
  \begin{itemize}
    \item Zweite Ebene
      \begin{itemize}
        \item Dritte Ebene
      \end{itemize}
    \end{itemize}
  \end{itemize}
\end{itemize}
```

- Erste Ebene
 - Zweite Ebene
 - * Dritte Ebene

Nummerierte Listen

Ganz analog funktioniert die `enumerate`-Umgebung, hier werden die einzelnen Punkte aber automatisch nummeriert:

```
\begin{enumerate}
  \item gewöhnliche Listen
  \item nummerierte Listen
  \item benannte Listen
\end{enumerate}
```

1. gewöhnliche Listen
2. nummerierte Listen
3. benannte Listen

Verschachtelte Listen (2)

Ganz analog geht dies auch mit nummerierten Listen:

```
\begin{enumerate}
\item Erste Ebene
  \begin{enumerate}
    \item Zweite Ebene
      \begin{enumerate}
        \item Dritte Ebene
      \end{enumerate}
    \end{enumerate}
  \end{enumerate}
\end{enumerate}
```

1. Erste Ebene
 - (a) Zweite Ebene
 - i. Dritte Ebene

Benannte Listen

Eine Besonderheit ist die benannte Umgebung:

```
\begin{description}
\item[Pochieren] ist Garen in heißer Flüssigkeit,
\item[Dämpfen] gilt als besonders schonend.
\item[Braten] schmeckt jedoch oft besser.
\end{description}
```

Pochieren ist Garen in heißer Flüssigkeit,
Dämpfen gilt als besonders schonend.
Braten schmeckt jedoch oft besser.

Tabellen

```
\begin{tabular}{rlc}
9:03 & Tigerenten Club & (ab 7 Jahren) \\
10:50 & Löwenzahn & (ab 6 Jahren) \\
11:20 & Die Schatzinsel & (ab 10 Jahren)
\end{tabular}
```

9:03	Tigerenten Club	(ab 7 Jahren)
10:50	Löwenzahn	(ab 6 Jahren)
11:20	Die Schatzinsel	(ab 10 Jahren)

Da L^AT_EX Tabellen nicht auf einer neuen Zeile beginnt, sollten Sie stets in einem eigenen Absatz oder in einer `center`-Umgebung benutzt werden.

Tabellen

Spezielle Tabellen haben wir bereits z.B. in der `matrix`-Umgebung kennengelernt. Allgemein werden sie mit

```
\begin{tabular}{ Format } ... \end{tabular}
```

erzeugt. Dabei enthält *Format* für jede Spalte ein Zeichen:

- `l` linksbündige Spalte
- `r` rechtsbündige Spalte
- `c` zentrierte Spalte

Die Spalten werden dann mit `&` getrennt, die Zeilen mit `\\` (kein `\\` nach der letzten Zeile!). Ein Beispiel:

```
\begin{tabular}{rlc}
9:03 & Tigerenten Club & (ab 7 Jahren) \\
10:50 & Löwenzahn & (ab 6 Jahren) \\
11:20 & Die Schatzinsel & (ab 10 Jahren)
\end{tabular}
```

Linien in Tabellen

Horizontale Linien zwischen den Zeilen einer Tabelle können mit `\hline` erzeugt werden, vertikale Linien werden beim *Format* mit “|” dargestellt:

```
\begin{tabular}{|r|lc|}
\hline
9:03 & Tigerenten Club & (ab 7 Jahren) \\
... & \\
\hline
\end{tabular}
```

9:03	Tigerenten Club	(ab 7 Jahren)
10:50	Löwenzahn	(ab 6 Jahren)
11:20	Die Schatzinsel	(ab 10 Jahren)

Z.B. mit `\cline{2-4}` wird nur eine horizontale Linie von Spalte 2 bis 4 gezogen.

multicolumn

Mit dem Befehl `\multicolumn{n}{Format}{Inhalt}` kann man n Spalten mit einer Spalte im angegebenen *Format* überspannen:

```
\begin{tabular}{rlc}
  9:03 & Tigerenten Club & (ab 7 Jahren) \\
  10:50 & \multicolumn{2}{c}{--- Pause ---} \\
  11:20 & Die Schatzinsel & (ab 10 Jahren)
\end{tabular}
```

9:03	Tigerenten Club	(ab 7 Jahren)
10:50	— Pause —	
11:20	Die Schatzinsel	(ab 10 Jahren)

Hervorhebungen und Zeichensätze

Wichtige Begriffe sollte man *hervorheben*, damit der Leser sie besser erkennt.

Gesetzt wurde dies mit:

Wichtige Begriffe sollte man `\emph{hervorheben}`, damit der Leser sie besser erkennt.

Es ist auch möglich, bestimmte Zeichensätze direkt auszuwählen:

<code>\textrm{normal}</code>	normal
<code>\textbf{fett}</code>	fett
<code>\textit{kursiv}</code>	<i>kursiv</i>
<code>\texttt{teletype}</code>	teletype
<code>\textsf{sans serif}</code>	sans serif
<code>\textsc{Kapitale}</code>	KAPITALE

p{...} und @{...}

Zwei weitere Format-Zeichen (neben `lrc` und `|`):

- `p{2.5cm}` fügt eine Spalte für 2.5cm breite Absätze ein. Erlaubte Maßeinheiten z.B.: mm, cm, in

Bsp.: `\begin{tabular}{rp{2cm}c}`

9:03	Tigerenten	(ab 7 Jahren)
	Club	
11:20	Die Schatz-	(ab 10 Jahren)
	insel	

- `@{xyz}` fügt eine Spalte ein, die mit dem Text `xyz` gefüllt wird. Der Abstand zwischen den einzelnen Spalten wird dann weggelassen.

Bsp.: `\begin{tabular}{cr@{.}l}`

		12.345
+	0.5	
=		12.845

Schriftgrößen

Wird eine bestimmte Schriftgröße gewählt, gilt dies innerhalb eines “Blockes”, der mit `{` und `}` umschlossen wird:

Dies ist ein `\huge` kleines Beispiel.

Dies ist ein `kleines` Beispiel.

<code>\tiny</code>	Text	<code>\scriptsize</code>	Text
<code>\footnotesize</code>	Text	<code>\small</code>	Text
<code>\normalsize</code>	Text	<code>\large</code>	Text
<code>\Large</code>	Text	<code>\LARGE</code>	Text
<code>\huge</code>	Text	<code>\Huge</code>	Text

documentclass-Optionen

Im Wesentlichen wird das Layout des Textes durch seine **Klasse** bestimmt: `article`, `report`, `book`,...

Einfache Feineinstellungen des Layouts erfolgen über **Optionen** zum `\documentclass[...]{...}`-Befehl:

- `10pt/11pt/12pt`: benutzte Basisschriftgröße
- `fleqn`: Formeln werden linksbündig statt zentriert gesetzt
- `leqno`: Formelnummern stehen auf der linken Seite statt rechts
- `titlepage/notitlepage`: es wird eine/keine eigene Titelseite gesetzt
- `onecolumn/twocolumn`: ein-/zweispaltiger Satz
- `oneside/twoside`: ein-/zweiseitiger Satz
- `landscape`: Seiten werden im Querformat gesetzt

Weitere Layout-Pakete

Neben z.B. den `a4`-Paketen gibt es weitere Pakete, die Einstellungen des Layouts erlauben. Einige wenige Beispiele:

- `\usepackage[...]{geometry}` erlaubt die genaue Einstellung aller vier Ränder. Mögliche Optionen z.B.:
`left=6cm, top=2.5cm, bottom=2.5cm, right=2.5cm`
- `\usepackage{setspace}` definiert drei Befehle zur Einstellung des Zeilenabstands:
 - ▶ `\singlespacing` (default) einfacher Zeilenabstand
 - ▶ `\onehalfspacing` eineinhalb-facher Abstand
 - ▶ `\doublespacing` doppelter Abstand

Kopf und Fuß

Seitenkopf und -fuß können durch den Befehl `\pagestyle{...}` eingestellt werden. Zur Auswahl stehen

- `\pagestyle{plain}` (Default) Nur eine Seitenzahl am Fuß der Seite.
- `\pagestyle{empty}` Kopf und Fuß bleiben leer.
- `\pagestyle{headings}` Aktuelle Kapitelüberschrift und Seitenzahl im Kopf der Seite.

`\pagestyle` sollte im Kopf der L^AT_EX-Datei, also vor `\begin{document}` stehen!

Mit `\thispagestyle{...}` läßt sich der Stil nur für die “aktuelle” Seite ändern.

Längen

Im Prinzip ist es möglich, alle Abstände etc. zu verändern. Eine (sehr kleine!) Auswahl dieser “Längen” (“dimen”):

<code>\baselineskip</code>	Höhe einer Zeile
<code>\parskip</code>	vertikaler Abstand zwischen Absätzen
<code>\parindent</code>	Breite der Einrückung am Anfang eines Absatzes
<code>\textwidth</code>	Breite einer Zeile
<code>\textheight</code>	Höhe des Textes auf einer Seite

Eine genaue Darstellung der jeweils aktuellen Längen, die das Layout einer Seite bestimmen liefert der Befehl `\layout` aus dem `layout`-Paket.

Maßeinheiten

Die wichtigsten Maßeinheiten für Längen sind:

mm	Millimeter	_3
cm	Centimeter (1cm=10mm)	_____
in	Zoll (1in=25.4mm)	_____
pt	“Punkt” (72.27pt=1in, 1pt≈0.35mm)	·
em	Breite eines “M”	—
ex	Höhe eines “x”	—

Auch die Längen selbst können als Maßeinheiten dienen:

$2\backslash\text{baselineskip}$	Doppelter Zeilenabstand	_____
$0.1\backslash\text{textwidth}$	$\frac{1}{10}$ Zeilenlänge	_____

³Die hier gezeigten Längen entsprechen nicht den wahren Größen, sondern zeigen nur die Verhältnisse der Größen zueinander.

Bilder einbinden

Möchte man Bilder in den Text einbinden, stellt das `graphicx`-Paket nützliche Befehle zur Verfügung:

```
\usepackage{graphicx}
```

```
...
```

Das Polynom $f(x)=\frac{1}{2}x^3-x$:

```
\begin{center}
```

```
\includegraphics{polynom.jpg}
```

```
\end{center}
```

```
...
```

Die Bilddateien sollten im gleichen Verzeichnis wie die `tex`-Dateien gespeichert sein.

Umgang mit Längen

Die Veränderung der Längen erfolgt mit folgenden Befehlen:

- `\setlength{\parindent}{0pt}`:
Die Länge wird auf die angegebene Größe gesetzt.
- `\addtolength{\textheight}{1\baselineskip}`:
Die angegebene Größe wird zur Länge hinzuaddiert.
- `\settoheight{\parindent}{Text}`:
Die Länge wird auf die Breite von “Text” gesetzt.

Analog zu `\settoheight` gibt es `\settodepth` (Höhe des Textes über der Grundlinie) und `\settoheight` (“Unterlänge” des Textes: wie weit reicht er unter die Grundlinie der Zeile).

Bildformate

Wichtig: Nicht jedes Bildformat kann benutzt werden.

- `pdflatex` (L^AT_EX, welches PDF-Dateien erstellt) verarbeitet das JPEG-Format (`*.jpg`), welches für Fotos sehr gut, für “technische” Zeichnungen aber weniger geeignet ist.
- Das “normale” L^AT_EX, welches `dvi`-Dateien erzeugt, verarbeitet das PostScript-Format (`*.eps`).

Das obige Beispiel funktioniert (JPEG-Datei!) also nur mit `pdflatex`. Erstellt man eine `dvi`-Datei als Ausgabe, müsste der Befehl

```
\includegraphics{polynom.eps}
```

lauten und eine PostScript-datei einbinden.

Abbildungen

Eingebundene Grafiken können innerhalb des Textes (z.B. in einer `center`-Umgebung) benutzt werden. Hierbei treten oft Probleme mit einem optisch unbefriedigenden Seitenumbruch auf.

\LaTeX übernimmt die automatische Positionierung von Abbildungen:

```
\begin{figure}
  \begin{center}
    \includegraphics{polynom.jpg}
  \end{center}
  \caption{Das Polynom  $f(x)=\frac{1}{2}x^3-x$ }
\end{figure}
```

Die Abbildung wird samt Beschriftung an einer “geeigneten” Stelle dargestellt. Mittels `\label` und `\ref` oder `\pageref` kann auf diese Abbildung verwiesen werden.

Eigene Umgebungen

Analog können Umgebungen definiert werden:

```
\newenvironment{name}[n]{ParameterA}{ParameterB}
```

`ParameterA` wird dabei jeweils am Anfang der Umgebung (`\begin{name}`) und `ParameterB` am Ende (`\end{name}`) eingefügt.

Möchte man Befehle oder Umgebungen neu definieren, die es bereits gibt, müssen die Anweisungen `\renewcommand` bzw. `\renewenvironment` benutzt werden.

Eigene Befehle

Die einfachste Art, \LaTeX zu erweitern, ist die Definition eigener Befehle:

```
\newcommand{\eps}{\varepsilon}
```

Allgemein definiert `\newcommand{\name}[n]{...}` einen Befehl `\name` mit `n` Parametern. Diese können innerhalb der Definition mit `#1, #2, ..., #n` angesprochen werden:

```
\newcommand{\vektor}[2]{%
  {\left( {#1}_1, \ldots, {#1}_{#2} \right)}
```

`\vektor{x}{10}` liefert damit (x_1, \dots, x_{10}) .

`input` und `include`

Bei einem umfangreichen Text ist es übersichtlicher, diesen in mehrere Dateien zu unterteilen. Diese können dann von einer anderen \LaTeX -Datei eingeladen werden:

```
\input{kapitel1.tex}
```

Hiermit wird die Datei `kapitel1.tex` an die Stelle dieses Befehls geladen.

Eine Alternative ist der Befehl `include{kapitel1}` (ohne `.tex!`). Hierbei wird dann aber stets eine neue Seite begonnen. Es kann in diesem Fall gesteuert werden, welche Dateien eingeladen werden (diese Anweisung muss im Kopf der Datei stehen):

```
\includeonly{kapitel1,kapitel3}
```