

# Ein Beispieldokument für L<sup>A</sup>T<sub>E</sub>X

Stavros Kousidis

16. Januar 2008

## Inhaltsverzeichnis

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>Text und Mathematische Umgebung</b>      | <b>1</b>  |
| <b>2</b> | <b>Tabellen, Aufzählungen und ähnliches</b> | <b>2</b>  |
| 2.1      | Tabellen . . . . .                          | 2         |
| 2.2      | Gleichungen . . . . .                       | 2         |
| 2.3      | Fallunterscheidungen . . . . .              | 3         |
| 2.4      | Aufzählungen . . . . .                      | 4         |
| <b>3</b> | <b>Definitionen, Lemmata, Theoreme etc.</b> | <b>6</b>  |
| <b>4</b> | <b>Kommutative Diagramme</b>                | <b>8</b>  |
| <b>5</b> | <b>Algorithmen</b>                          | <b>9</b>  |
| <b>6</b> | <b>Kleinigkeiten</b>                        | <b>11</b> |
| 6.1      | Summenzeichen und Indizes . . . . .         | 11        |
| 6.2      | Text einrücken unterdrücken . . . . .       | 11        |
| <b>7</b> | <b>Links</b>                                | <b>12</b> |



# 1 Text und Mathematische Umgebung

Um mathematische Formeln im Fließtext einzubinden stehen in  $\text{\LaTeX}$  die Steuersymbole  $\langle$  und  $\rangle$  zur Verfügung. Damit wird der mathematische Teil des Satzes umklammert und  $\text{\LaTeX}$  angewiesen diesen gesondert zu formatieren. z.B. erzeugt

Der Satz des Pythagoras besagt, dass in einem rechtwinkligen Dreieck die Katheten  $\langle a \rangle, \langle b \rangle$  und die Hypotenuse  $\langle c \rangle$  die Gleichung  $\langle a^2 + b^2 = c^2 \rangle$  erfüllen.

die folgende Ausgabe

Der Satz des Pythagoras besagt, dass in einem rechtwinkligen Dreieck die Katheten  $a, b$  und die Hypotenuse  $c$  die Gleichung  $a^2 + b^2 = c^2$  erfüllen.

Darüberhinaus kann man mit den Steuersymbolen  $\langle$  und  $\rangle$  eine Zentrierung des geklammerten Ausdrucks erreichen. Diese Steuersymbole werden auch für die `array`-Umgebung verwendet, auf die im Folgenden eingegangen wird.

## 2 Tabellen, Aufzählungen und ähnliches

### 2.1 Tabellen

Zur Anordnung mehrerer Elemente in Tabellenform, kann man folgende Umgebung nutzen.

```
\[
\begin{array}{Spalten}
... & ... & ... \\
... & ... & ...
\end{array}
\]
```

Als „Spalten“ gibt man eine Zeichenfolge aus den drei Buchstaben  $\{l, c, r\}$  an. Die Anzahl der Buchstaben entspricht der Anzahl der Spalten, wobei  $l$  für links,  $c$  für zentriert und  $r$  für rechts Ausrichten innerhalb der jeweiligen Spalte steht. Die Steuerungssymbole  $\[$  und  $\]$  sorgen dafür, dass das Feld abgesetzt in einer neuen Zeile zentriert erscheint. Innerhalb der Umgebung trennt man einzelne Spalten mit  $\&$  und eine neue Zeile wird mit  $\backslash\backslash$  erzeugt. z.B. ergibt der folgende Quelltext

```
\[
\begin{array}{cccl}
f: & \mathbb{R} & \longrightarrow & \mathbb{R}^2 \\
& t & \longmapsto & (t^2, t^3)
\end{array}
\]
```

dieses Ergebnis

$$\begin{array}{lcl} f: \mathbb{R} & \longrightarrow & \mathbb{R}^2 \\ & t & \longmapsto (t^2, t^3) \end{array}$$

Prinzipiell läßt sich mit einer `array`-Umgebung alles tabellarisch darstellbare formatieren. Jedoch gibt es für einige spezielle Anforderungen weitere ähnliche Umgebungen, auf die wir im folgenden eingehen.

### 2.2 Gleichungen

Gleichungen können zwar auch mit der `array`-Umgebung formatiert werden, allerdings steht zusätzlich die folgende spezielle Umgebung zur Verfügung.

```
\begin{equation}
...
\end{equation}
```

Hier wird der Quelltext

```
\begin{equation}
  x^2 + y^2 &= & z^2
\end{equation}
```

wie folgt umgesetzt

$$x^2 + y^2 = z^2 \tag{1}$$

Die Steuerungssymbole `\[` und `\]`, sowie `&` und `\\` sind bei der `equation`-Umgebung nicht zu gebrauchen. Außerdem sieht man, dass diese Umgebung die Gleichung nummeriert.

Möchte man mehrere Gleichungen untereinander schreiben, so kann man die folgende Umgebung nutzen.

```
\begin{eqnarray}
  ... && & & \\
  ... && & & \\
\end{eqnarray}
```

Damit ergibt der Quelltext

```
\begin{eqnarray}
  x^2 + y^2 &= & z^2 \\
  x^3 - y^2 &= & 0
\end{eqnarray}
```

die folgende Formatierung.

$$x^2 + y^2 = z^2 \tag{2}$$

$$x^3 - y^2 = 0 \tag{3}$$

Man sieht, dass die Gleichungen insgesamt fortlaufend nummeriert werden. Der Übersichtlichkeit halber kann es jedoch Sinn machen, nicht jede Gleichung mit einer Nummer zu versehen. Daher sollte man bei jeder abgesetzten Gleichungen individuell überlegen, ob man diese mit einer `array`-Umgebung oder `equation`- bzw. `eqnarray`-Umgebung formatiert.

## 2.3 Fallunterscheidungen

Fallunterscheidungen können elegant mit der `case`-Umgebung dargestellt werden.

```

\[
Ausdruck =
\begin{cases}
... & \& ... \\
... & \& ...
\end{cases}
\]

```

z.B. ergibt der Quelltext

```

\[
X =
\begin{cases}
D(p,q) \cup J(p,q) & \& \mbox{falls } \$p < q\$ \\
D(p,q) & \& \mbox{falls } \$p > q\$
\end{cases}
\]

```

die folgende Ausgabe

$$X = \begin{cases} D(p, q) \cup J(p, q) & \text{falls } p < q \\ D(p, q) & \text{falls } p > q \end{cases}$$

## 2.4 Aufzählungen

Aufzählungen lassen sich mit der `enumerate`-Umgebung realisieren. Jeder einzelne Punkt in der Aufzählung wird mit dem Befehl `\item` eingeleitet.

```

\begin{enumerate}
\item ...
\item ...
...
\end{enumerate}

```

Das folgende Beispiel

```

\begin{enumerate}
\item Dies ist der erste Punkt
\item Jetzt kommt der zweite
\end{enumerate}

```

erzeugt dann die folgende Ausgabe

1. Dies ist der erste Punkt
2. Jetzt kommt der zweite

Aufzählungen nummerieren standardmäßig mit arabischen Ziffern. Möchte man aber mit römischen Ziffern oder Buchstaben nummerieren, so muß man vor dem ersten `\item`-Befehl folgendes ausführen

```
\renewcommand{\labelenumi}{Nummerierung}
```

wobei wir für `Nummerierung` die Befehle

```
\arabic{enumi}  
\roman{enumi}  
\alph{enumi}
```

setzen können, die entsprechend  $1, 2, \dots$  bzw.  $i, ii, \dots$  und  $a, b, \dots$  erzeugen. Zusätzlich kann man diese Befehle Klammern und mit Punkten versehen, um die Ausgabe ein wenig zu „optimieren“. z.B. erzeugt `(\roman{enumi})` dann  $(i)$ ,  $(ii)$ ,  $\dots$ . Insgesamt betrachtet erzeugt der Quelltext

```
\begin{enumerate}  
  \renewcommand{\labelenumi}{(\roman{enumi}).}  
  \item Erster Punkt  
  \item Zweiter Punkt  
  \renewcommand{\labelenumi}{\arabic{enumi}.}  
\end{enumerate}
```

dann die folgende Ausgabe

- (i). Erster Punkt
- (ii). Zweiter Punkt

Dabei sollte man darauf achten, am Ende wieder die Standardnummerierung einzustellen, um Unstimmigkeiten innerhalb des gesamten  $\text{\LaTeX}$ -Dokuments zu vermeiden. Dies bewirkt der letzte `\renewcommand` im obigen Beispiel.

### 3 Definitionen, Lemmata, Theoreme etc.

Für Definition, Propositionen, Theoreme, Lemmata, Korollare, Vermutungen, Notationen, Bemerkungen und Beispiele gibt es vorgefertigte Umgebungen, die in der Hauptdatei definiert werden können. In der vorliegenden Hauptdatei ist dies der folgende Abschnitt.

```
\theoremstyle{theorem}
\newtheorem{thm}[subsection]{Theorem}
\newtheorem{lem}[thm]{Lemma}
\newtheorem{cor}[thm]{Korollar}
\newtheorem{prp}[thm]{Proposition}
\newtheorem{cnj}[thm]{Vermutung}

\theoremstyle{definition}
\newtheorem{dfn}[thm]{Definition}
\newtheorem{ntn}[thm]{Notation}
\newtheorem{rem}[thm]{Bemerkung}
\newtheorem{exl}[thm]{Beispiel}

\theoremstyle{remark}
\newtheorem*{prf}{Beweis}
```

Damit kann man mit `\begin{Umgebung}` und `\end{Umgebung}` eine Definition etc. formatieren, wobei

$$\text{Umgebung} \in \{\text{thm, lem, prp, cnj, dfn, ntn, rem, exl, prf}\}$$

z.B. erzeugt

```
\begin{lem}
  Dies ist ein Lemma.
\end{lem}

\begin{prf}
  Gilt einfach.
\end{prf}
```

folgende Ausgabe

**Lemma 3.0.1.** *Dies ist ein Lemma.*

*Beweis.* Gilt einfach.

Zur Nummerierung dieser Umgebungen sollte noch etwas gesagt werden. Der optionale Parameter `[subsection]` bewirkt, dass diese Umgebungen fortlaufend auf `subsection`-Ebene nummeriert werden. Befinden wir uns

in der 3. Section und hierin wieder in der 2. Subsection, so erhält das 4. Lemma die Nummer 3.2.4. Der optionale Parameter `[thm]` bewirkt, dass alle Umgebungen gemeinsam fortlaufend nummeriert werden und nicht nur die Definitionen, Lemmata etc. jeweils für sich gesehen.

## 4 Kommutative Diagramme

In  $\LaTeX$  kann man sehr elegant kommutative Diagramme mit dem Paket `Xy-pic` erstellen. Dieses bindet man folgendermaßen ein.

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

Anschließend steht einem die `xy`-Umgebung zur Verfügung.

```
\[
\begin{xy}
\xymatrix{
... & \& ... \\
... & \& ...
}
\end{xy}
\]
```

D.h. der Quelltext

```
\[
\begin{xy}
\xymatrix{
A \ar[r]^f \ar[d]_i & B \ar[d]_j \\
C \ar[r]_g & D
}
\end{xy}
\]
```

generiert folgendes klassisches Quadrat

$$\begin{array}{ccc} A & \xrightarrow{f} & B \\ i \downarrow & & \downarrow j \\ C & \xrightarrow{g} & D \end{array}$$

Dieses Paket ist sehr umfangreich, daher gehe ich hier nicht näher darauf ein. Im Internet findet man sehr viele und auch sehr gute Dokumentationen zu diesem Paket und den Möglichkeiten, die man damit hat.

## 5 Algorithmen

Die Pakete `algorithmic` und `algorithm` bieten Funktionalität, um einen Algorithmus zu formatieren. Hier ein Beispiel

```
\floatname{algorithm}{Algorithmus}
\begin{algorithm}
  \caption{Würfel-Methode}
  \begin{algorithmic}
    \STATE wähle  $\phi$ ,  $\alpha$  geeignet
    \FOR  $\{i:=1 \text{ to } n\}$ 
      \STATE  $j:=1$ 
      \WHILE $\{(\{j \leq d \text{ und } \left|p_{i,j}-q_j\right| \leq \frac{\alpha}{2}\})\}$ 
        \STATE  $j:=j+1$ 
        \IF $\{j:=d+1\}$ 
          \STATE  $p:=p_i$ ;  $\alpha=2d_{\max}(p,q)$ 
        \ENDIF
      \ENDWHILE
    \ENDFOR
    \IF $\{p \text{ ist definiert}\}$ 
      \STATE return  $p$  als Nearest-Neighbour
    \ELSE
      \STATE Brute-Force Methode
    \ENDIF
  \end{algorithmic}
\end{algorithm}
```

und seine Ausgabe

---

**Algorithmus 1** Würfel-Methode

---

```
wähle  $\phi, \alpha$  geeignet
for  $i := 1$  to  $n$  do
   $j := 1$ 
  while ( $j \leq d$  und  $|p_{i,j} - q_j| \leq \frac{\alpha}{2}$ ) do
     $j := j + 1$ 
    if  $j := d + 1$  then
       $p = p_i; \alpha = 2d_{max}(p, q)$ 
    end if
  end while
end for
if  $p$  ist definiert then
  return  $p$  als Nearest-Neighbour
else
  Brute-Force Methode
end if
```

---

## 6 Kleinigkeiten

### 6.1 Summenzeichen und Indizes

Das Summenzeichen orientiert seine oberen und unteren Indizes hinter dem Summenzeichen, wie das Beispiel

```
\[
\begin{array}{rcl}
\sum_{i=1}^n i & = & \frac{1}{2} n (n+1)
\end{array}
\]
```

hier zeigt

$$\sum_{i=1}^n i = \frac{1}{2}n(n+1)$$

Man kann die oberen und unteren Indizes aber auch unter dem Summenzeichen anordnen, wenn man zusätzlich den Befehl `\limits` hinter dem Summenzeichen verwendet. D.h. man schreibt `\sum\limits` anstatt `\sum`. Wir modifizieren unser o.g. Beispiel dementsprechend

```
\[
\begin{array}{rcl}
\sum\limits_{i=1}^n i & = & \frac{1}{2} n (n+1)
\end{array}
\]
```

und erhalten

$$\sum_{i=1}^n i = \frac{1}{2}n(n+1)$$

### 6.2 Text einrücken unterdrücken

Möchte man das einrücken des Textes, z.B. nach den Steuersymbolen `\[` und `\]`, vermeiden, steht einem der Befehl `\noindent` zur Verfügung. Diesen setzt man vor den eigentlichen Text. D.h.

```
\[
\begin{array}{Spalten}
...
\end{array}
\]
\noindent Jetzt geht es uneingerückt im Text weiter...
```

Man kann dieses Verhalten auch global mit dem Befehl `\parindent0mm` ausschalten, wobei `0mm` der Wert der Einrücktiefe ist. Hat man dies gemacht und möchte dann doch explizit einrücken, benutzt man `\indent`.

## 7 Links

Es gibt eine Unmenge an Literatur im Internet zu  $\text{\LaTeX}$  und den jeweiligen Paketen. Als Beispiel seien folgende Seiten genannt.

$\text{\LaTeX}$  im Allgemeinen:

<http://theoal.sys.uea.ac.uk/~nlct/latex/novices/index.html>

<http://www.weinelt.de/latex/index.html>

<http://www.math.uiuc.edu/~hildebr/tex/>

<http://www.ctan.org>

<http://de.wikibooks.org/wiki/Latex>

Zum Package Xy-pic:

<http://www.tug.org/applications/Xy-pic/Xy-pic.html>

<http://www.guntherkrauss.de/computer/tex/diagramme.html>