ZfS–Kurs „LaTeX für Naturwissenschaftler“

Karin Halupczok
Email: Karin.Halupczok@math.uni-freiburg.de

SoSe 2009

Dritte Kurssitzung

http://home.mathematik.uni-freiburg.de/halupczok/latex.html

Der Mathematikmodus

Mathematikformeln einsetzen

Mit dem Symbol $ wird eine Mathematikformel umschlossen und in die aktuelle Textzeile eingesetzt, Bsp.: $2\sum_{k=1}^{n} k = n(n+1)$ wird erzeugt durch `$2\sum_{k=1}^{n} k = n(n+1)$`

Mathematikformeln einsetzen

Mit dem Symbol $ wird eine Mathematikformel umschlossen und in die aktuelle Textzeile eingesetzt, Bsp.: $2\sum_{k=1}^{n} k = n(n+1)$ wird erzeugt durch `$2\sum_{k=1}^{n} k = n(n+1)$`

Mit der Umgebung `\[ ...\]` (auch möglich mit $$ ... $$) wird die Formel abgesetzt, Bsp.:

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

wird erzeugt durch

```
\[
  2\sum_{k=1}^{n} k = n(n+1)
\]
```

Mathematikformeln einsetzen

Mit dem Symbol $ wird eine Mathematikformel umschlossen und in die aktuelle Textzeile eingesetzt, Bsp.: $2\sum_{k=1}^{n} k = n(n+1)$ wird erzeugt durch `$2\sum_{k=1}^{n} k = n(n+1)$`

Mit der Umgebung `\[ ...\]` (auch möglich mit $$ ... $$) wird die Formel abgesetzt, Bsp.:

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

wird erzeugt durch

```
\[
  2\sum_{k=1}^{n} k = n(n+1)
\]
```

Abstände in Formeln (falls nötig): `\! \, \: \; \enspace \quad \qquad`

Große Symbole

Das Summen-Sumbol $\sum$ wird durch `\sum` erzeugt,
das Produkt-Symbol $\prod$ durch `\prod`,
ein Integral-Symbol $\int$ durch `\int`,
die Indizierung erfolgt so wie im vorigen Beispiel.

Große Symbole

Das Summen-Sumbol $\sum$ wird durch `\sum` erzeugt,
das Produkt-Symbol $\prod$ durch `\prod`,
ein Integral-Symbol $\int$ durch `\int`,
die Indizierung erfolgt so wie im vorigen Beispiel.

Ansonsten erzeugt `_{}` einen Index, und `^{}` einen Exponenten.

Große Symbole

Das Summen-Sumbol $\sum$ wird durch `\sum` erzeugt,
das Produkt-Symbol $\prod$ durch `\prod`,
ein Integral-Symbol $\int$ durch `\int`,
die Indizierung erfolgt so wie im vorigen Beispiel.

Ansonsten erzeugt `_{}` einen Index, und `^{}` einen Exponenten.

Für mehrzeilige Indizes nimmt man `\substack`:

$$\sum_{\substack{p\leq N \\ a|p}} N_p$$

wird erzeugt durch
`\[ \sum_{\substack{p\leq N \\ a|p}} N_{p} \]`.

Große Symbole

Das Summen-Sumbol $\sum$ wird durch `\sum` erzeugt,
das Produkt-Symbol $\prod$ durch `\prod`,
ein Integral-Symbol $\int$ durch `\int`,
die Indizierung erfolgt so wie im vorigen Beispiel.

Ansonsten erzeugt `_{}` einen Index, und `^{}` einen Exponenten.

Für mehrzeilige Indizes nimmt man `\substack`:

$$\sum_{\substack{p\leq N \\ a|p}} N_p$$

wird erzeugt durch
`\[ \sum_{\substack{p\leq N \\ a|p}} N_{p} \]`.

Wurzel-Konstrukte entstehen mit `\sqrt`, z.B. $\sqrt{n}$ durch `\sqrt{n}`
oder $\sqrt[3]{n}$ durch `\sqrt[3]{n}`.

Mathematische Symbole

Viele viele weitere Symbole werden einfach per Befehl gesetzt, z. B.:

$\geq$ \geq	$\leq$ \leq	$\in$ \in	$\ni$ \ni
$\subseteq$ \subseteq	$\supseteq$ \supseteq	$\ll$ \ll	$\gg$ \gg
$\equiv$ \equiv	$\approx$ \approx	$\sim$ \sim	$\cong$ \cong
$\to$ \to	$\mapsto$ \mapsto	$\Rightarrow$ \Rightarrow	$\Leftarrow$ \Leftarrow

Mathematische Symbole

Viele viele weitere Symbole werden einfach per Befehl gesetzt, z. B.:

$\geq$ `\geq`	$\leq$ `\leq`	$\in$ `\in`	$\ni$ `\ni`
$\subseteq$ `\subseteq`	$\supseteq$ `\supseteq`	$\ll$ `\ll`	$\gg$ `\gg`
$\equiv$ `\equiv`	$\approx$ `\approx`	$\sim$ `\sim`	$\cong$ `\cong`
$\to$ `\to`	$\mapsto$ `\mapsto`	$\Rightarrow$ `\Rightarrow`	$\Leftarrow$ `\Leftarrow`

Viele dieser Symbole sind nur im Zusammenhang mit den Paketen `amsmath` und `amssymb` verfügbar. Diese sollte man unbedingt mitladen, wenn man viel mit dem Mathematikmodus arbeitet.

Mathematische Symbole

Viele viele weitere Symbole werden einfach per Befehl gesetzt, z. B.:

$\geq$ \geq	$\leq$ \leq	$\in$ \in	$\ni$ \ni
$\subseteq$ \subseteq	$\supseteq$ \supseteq	$\ll$ \ll	$\gg$ \gg
$\equiv$ \equiv	$\approx$ \approx	$\sim$ \sim	$\cong$ \cong
$\to$ \to	$\mapsto$ \mapsto	$\Rightarrow$ \Rightarrow	$\Leftarrow$ \Leftarrow

Viele dieser Symbole sind nur im Zusammenhang mit den Paketen `amsmath` und `amssymb` verfügbar. Diese sollte man unbedingt mitladen, wenn man viel mit dem Mathematikmodus arbeitet.

Auch griechische Buchstaben werden „mit Namen“ angesprochen, z. B. α \alpha, β \beta, γ \gamma, Γ \Gamma, δ \delta usw.
Das Zeichen für's partielle Ableiten: ∂ \partial

Mathematische Symbole

Viele viele weitere Symbole werden einfach per Befehl gesetzt, z. B.:

$\geq$ \geq	$\leq$ \leq	$\in$ \in	$\ni$ \ni
$\subseteq$ \subseteq	$\supseteq$ \supseteq	$\ll$ \ll	$\gg$ \gg
$\equiv$ \equiv	$\approx$ \approx	$\sim$ \sim	$\cong$ \cong
$\to$ \to	$\mapsto$ \mapsto	$\Rightarrow$ \Rightarrow	$\Leftarrow$ \Leftarrow

Viele dieser Symbole sind nur im Zusammenhang mit den Paketen `amsmath` und `amssymb` verfügbar. Diese sollte man unbedingt mitladen, wenn man viel mit dem Mathematikmodus arbeitet.

Auch griechische Buchstaben werden „mit Namen“ angesprochen, z. B. α \alpha, β \beta, γ \gamma, Γ \Gamma, δ \delta usw.
Das Zeichen für's partielle Ableiten: ∂ \partial

Liste mathematischer Symbole (ab S. 20): www.ctan.org/tex-archive/info/symbols/comprehensive/symbols-a4.pdf

Operatornamen

Hin und wieder muß man Operatornamen selbst definieren, z. B. grad für einen Grad, etwa den eines Polynoms p. Dies geht mit `$\operatorname{grad} p$` und ergibt $\operatorname{grad} p$.

Operatornamen

Hin und wieder muß man Operatornamen selbst definieren, z. B. grad für einen Grad, etwa den eines Polynoms p. Dies geht mit `$\operatorname{grad} p$` und ergibt grad p.

Damit man auch einfach `$\grad p$` schreiben kann, macht man in der Präambel die folgende Definition:
`\newcommand{\grad}{\operatorname{grad}}`

Operatornamen

Hin und wieder muß man Operatornamen selbst definieren, z. B. grad für einen Grad, etwa den eines Polynoms p. Dies geht mit `$\operatorname{grad} p$` und ergibt grad p.

Damit man auch einfach `$\grad p$` schreiben kann, macht man in der Präambel die folgende Definition:
`\newcommand{\grad}{\operatorname{grad}}`

Um für die reellen Zahlen $\mathbb{R}$ einfach `$\R$` schreiben zu können, definiert man in der Präambel den Befehl
`\newcommand{\R}{\mathbb{R}}`

Mit den anderen Zahlbereichen geht das natürlich genauso.

Operatornamen

Hin und wieder muß man Operatornamen selbst definieren, z. B. grad für einen Grad, etwa den eines Polynoms p. Dies geht mit `$\operatorname{grad} p$` und ergibt grad p.

Damit man auch einfach `$\grad p$` schreiben kann, macht man in der Präambel die folgende Definition:
`\newcommand{\grad}{\operatorname{grad}}`

Um für die reellen Zahlen $\mathbb{R}$ einfach `$\R$` schreiben zu können, definiert man in der Präambel den Befehl
`\newcommand{\R}{\mathbb{R}}`

Mit den anderen Zahlbereichen geht das natürlich genauso.

Für Fettdruck und Kalligraphie-Buchstaben im Mathemodus nimmt man `\mathbf` und `\mathcal`.

Formelteile hervorheben

Unterstreichung: `\underline{Text oder Formel}` ergibt
$\underline{\text{Text oder Formel}}$, auch im Textmodus

Formelteile hervorheben

Unterstreichung: `\underline{Text oder Formel}` ergibt $\underline{\text{Text oder Formel}}$, auch im Textmodus

Überstreichung: `\overline`, für komplex Konjugiertes, z. B. `$\overline{f(x)}+g(x)$` ergibt $\overline{f(x)} + g(x)$.

Formelteile hervorheben

Unterstreichung: `\underline{Text oder Formel}` ergibt
$\underline{\text{Text oder Formel}}$, auch im Textmodus

Überstreichung: `\overline`, für komplex Konjugiertes, z. B.
`$\overline{f(x)}+g(x)$` ergibt $\overline{f(x)} + g(x)$.

Unterschreibung: `\underset`, z. B. `$f(x)\underset{(1)}{\leq} g(x)$` erzeugt $f(x) \underset{(1)}{\leq} g(x)$

Ebenso läßt sich `\overset` für Überschreibung verwenden.

Formelteile hervorheben

Unterstreichung: `\underline{Text oder Formel}` ergibt
$\underline{\text{Text oder Formel}}$, auch im Textmodus

Überstreichung: `\overline`, für komplex Konjugiertes, z. B.
`$\overline{f(x)}+g(x)$` ergibt $\overline{f(x)}+g(x)$.

Unterschreibung: `\underset`, z. B. `$f(x)\underset{(1)}{\leq} g(x)$` erzeugt $f(x) \underset{(1)}{\leq} g(x)$

Ebenso läßt sich `\overset` für Überschreibung verwenden.

Unterklammerung: `\underbrace`, z. B.
`$\underbrace{f(x)}+g(x)$` erzeugt $\underbrace{f(x)}+g(x)$.

Etwas unter die Klammer setzt man mit
`\underbrace{Formel}_{drunter}`.

Ebenso erzeugt `\overbrace` eine Überklammerung.

Fallunterscheidungskonstrukte

Die cases-Umgebung erzeugt eine Fallunterscheidung, z. B.

$$f(m) = \begin{cases} g(m), & \text{falls } m \text{ gerade,} \\ h(m), & \text{falls } m \text{ ungerade.} \end{cases}$$

Fallunterscheidungskonstrukte

Die cases-Umgebung erzeugt eine Fallunterscheidung, z. B.

$$f(m) = \begin{cases} g(m), & \text{falls } m \text{ gerade,} \\ h(m), & \text{falls } m \text{ ungerade.} \end{cases}$$

wird erzeugt durch

```
\[
  f(m) =
 \begin{cases}
     g(m), & \text{falls } m \text{ gerade}, \\
     h(m), & \text{falls } m \text{ ungerade.}
 \end{cases}
\]
```

Fallunterscheidungskonstrukte

Die cases-Umgebung erzeugt eine Fallunterscheidung, z. B.

$$f(m) = \begin{cases} g(m), & \text{falls } m \text{ gerade,} \\ h(m), & \text{falls } m \text{ ungerade.} \end{cases}$$

wird erzeugt durch

```
\[
  f(m) =
 \begin{cases}
     g(m), & \text{falls } m \text{ gerade}, \\
     h(m), & \text{falls } m \text{ ungerade.}
 \end{cases}
\]
```

Mit `\text{}` läßt sich Text in einer Formel einfügen. LaTeX achtet dabei auf Leerzeichen.

Klammern

Klammern in verschiedenen Größen: $(), \big(\big), \Big(\Big), \bigg(\bigg), \Bigg(\Bigg)$

Klammern

Klammern in verschiedenen Größen: $(), \bigl(\bigr), \Bigl(\Bigr), \biggl(\biggr), \Biggl(\Biggr)$

diese werden erzeugt von
`$(),\bigl(\bigr),\Bigl(\Bigr),\biggl(\biggr),\Biggl(\Biggr)$`, automatische Größe mit `\left` und `\right`:
Z. B.: `\left(` für eine linke Klammer mit automatisch richtiger Größe (aber nicht immer schön!).

Klammern

Klammern in verschiedenen Größen: $(), \bigl(\bigr), \Bigl(\Bigr), \biggl(\biggr), \Biggl(\Biggr)$

diese werden erzeugt von
`$(),\bigl(\bigr),\Bigl(\Bigr),\biggl(\biggr),\Biggl(\Biggr)$`, automatische Größe mit `\left` und `\right`:
Z. B.: `\left(` für eine linke Klammer mit automatisch richtiger Größe (aber nicht immer schön!).

Weitere Klammern sind $\{\}$ (werden erzeugt von `$\{\}$`) sowie die Norm-Symbole $|$ (mit `$|$`) und $\|$ (mit `$\|$`), die mit den `\big...`- und `\Big...`-Befehlen wie mit () verwendet werden können.

Klammersymbole und Matrizen

Brüche werden mit `\frac` erzeugt, also z. B. $\frac{1}{2}$ durch `$\frac{1}{2}$`.

Klammersymbole und Matrizen

Brüche werden mit `\frac` erzeugt, also z. B. $\frac{1}{2}$ durch `$\frac{1}{2}$`.

Binomialkoeffizienten werden mit `\binom` erzeugt, also z. B. $\binom{n}{k}$ durch `$\binom{n}{k}$`.

Klammersymbole und Matrizen

Brüche werden mit `\frac` erzeugt, also z. B. $\frac{1}{2}$ durch `$\frac{1}{2}$`.

Binomialkoeffizienten werden mit `\binom` erzeugt, also z. B. $\binom{n}{k}$ durch `$\binom{n}{k}$`.

Eine Matrix mit runden Klammern erzeugt man mit dem Befehl `\pmatrix` so:

```
\[ \begin{pmatrix} 0 & 1 \\ -1 & i \end{pmatrix} \]
```

erzeugt die Matrix

$$\begin{pmatrix} 0 & 1 \\ -1 & i \end{pmatrix}$$

Klammersymbole und Matrizen

Brüche werden mit `\frac` erzeugt, also z. B. $\frac{1}{2}$ durch `$\frac{1}{2}$`.

Binomialkoeffizienten werden mit `\binom` erzeugt, also z. B. $\binom{n}{k}$ durch `$\binom{n}{k}$`.

Eine Matrix mit runden Klammern erzeugt man mit dem Befehl `\pmatrix` so:

```
\[ \begin{pmatrix} 0 & 1 \\ -1 & i \end{pmatrix} \]
```

erzeugt die Matrix

$$\begin{pmatrix} 0 & 1 \\ -1 & i \end{pmatrix}$$

Weitere Matrix-Umgebungen mit anderen Begrenzungen (d. h. nichtrunde Klammern) sind `matrix`, `bmatrix`, `vmatrix`, `Vmatrix`

Formelumgebungen

Das Paket `amsmath` stellt eine Menge verschiedener Formelumgebungen zur Verfügung.

Formelumgebungen

Das Paket `amsmath` stellt eine Menge verschiedener Formelumgebungen zur Verfügung.

Eine davon ist die Umgebung `equation`, sie setzt eine abgesetzte Formel, genauso wie `\[ \]`. Sie wird jedoch automatisch numeriert, es sei denn, man verwendet die Umgebung `equation*`, welche die Numerierung unterdrückt.

Formelumgebungen

Das Paket `amsmath` stellt eine Menge verschiedener Formelumgebungen zur Verfügung.

Eine davon ist die Umgebung `equation`, sie setzt eine abgesetzte Formel, genauso wie `\[ \]`. Sie wird jedoch automatisch numeriert, es sei denn, man verwendet die Umgebung `equation*`, welche die Numerierung unterdrückt.

Mit der `gather`-Umgebung kann man mehrzeilige Formeln ohne Ausrichtung zueinander setzen:

$$a^2 + b^2 = c^2$$

$$a, b, c \neq 0$$

Formelumgebungen

Das Paket `amsmath` stellt eine Menge verschiedener Formelumgebungen zur Verfügung.

Eine davon ist die Umgebung `equation`, sie setzt eine abgesetzte Formel, genauso wie `\[ \]`. Sie wird jedoch automatisch numeriert, es sei denn, man verwendet die Umgebung `equation*`, welche die Numerierung unterdrückt.

Mit der `gather`-Umgebung kann man mehrzeilige Formeln ohne Ausrichtung zueinander setzen:

$$a^2 + b^2 = c^2$$
$$a, b, c \neq 0$$

wird erzeugt von

```
\begin{gather*}
   a^{2}+b^{2}=c^{2}\\
   a,b,c\neq 0
\end{gather*}
```

Formelumgebungen: align und alignat

Eine weitere wichtige Umgebung ist `align`. Diese richtet mehrzeilige Formeln an einem Symbol untereinander aus, das mit `&` markiert wird. Das können pro Zeile auch mehrere sein.
Die *-Variante unterdrückt die Formelnumerierung. Bsp.:

Formelumgebungen: align und alignat

Eine weitere wichtige Umgebung ist `align`. Diese richtet mehrzeilige Formeln an einem Symbol untereinander aus, das mit `&` markiert wird. Das können pro Zeile auch mehrere sein.
Die *-Variante unterdrückt die Formelnumerierung. Bsp.:

$$a^2 + b^2 = c^2$$
$$x^3 + y^3 = z^3 \quad (*)$$

```
\begin{align*}
    a^{2}+b^{2} &= c^{2} \\
    x^{3}+y^{3} &= z^{3} \tag{*}
\end{align*}
```

Formelumgebungen: align und alignat

Eine weitere wichtige Umgebung ist `align`. Diese richtet mehrzeilige Formeln an einem Symbol untereinander aus, das mit `&` markiert wird. Das können pro Zeile auch mehrere sein.
Die *-Variante unterdrückt die Formelnumerierung. Bsp.:

$$a^2 + b^2 = c^2$$
$$x^3 + y^3 = z^3 \quad (*)$$

```
\begin{align*}
    a^{2}+b^{2} &= c^{2} \\
    x^{3}+y^{3} &= z^{3} \tag{*}
\end{align*}
```

Die Umgebung `alignat` richtet an mehreren Stellen aus, die mit dem Zeichen `&` markiert werden. Dann muß die Anzahl dieser Ausrichtungszeichen in jeder Zeile gleich sein.

Formelumgebungen: align und alignat

Eine weitere wichtige Umgebung ist `align`. Diese richtet mehrzeilige Formeln an einem Symbol untereinander aus, das mit `&` markiert wird. Das können pro Zeile auch mehrere sein.
Die *-Variante unterdrückt die Formelnumerierung. Bsp.:

$$a^2 + b^2 = c^2$$
$$x^3 + y^3 = z^3 \quad (*)$$

```
\begin{align*}
    a^{2}+b^{2} &= c^{2} \\
    x^{3}+y^{3} &= z^{3} \tag{*}
\end{align*}
```

Die Umgebung `alignat` richtet an mehreren Stellen aus, die mit dem Zeichen `&` markiert werden. Dann muß die Anzahl dieser Ausrichtungszeichen in jeder Zeile gleich sein.

Soll in einer *-Umgebung ohne Numerierung doch ein Numerierungssymbol hinein, geht das wie hier mit `\tag{}` am Ende der gewünschten Formelzeile.

Abstände und Theoreme

In einer `align(at)-` oder `gather`-Umgebung erzeugt `\\[1cm]` anstatt `\\` einen Abstand von 1cm zwischen den Zeilen der Formel.

Abstände und Theoreme

In einer `align(at)-` oder `gather`-Umgebung erzeugt `\\[1cm]` anstatt `\\` einen Abstand von 1cm zwischen den Zeilen der Formel.

Ferner stellt das Paket `amsthm` eine Fülle an Theorem-, Satz- und Lemma-Umgebungen zur Verfügung, die man sich selbst auch umdefinieren kann.

Abstände und Theoreme

In einer `align(at)-` oder `gather`-Umgebung erzeugt `\\[1cm]` anstatt `\\` einen Abstand von 1cm zwischen den Zeilen der Formel.

Ferner stellt das Paket `amsthm` eine Fülle an Theorem-, Satz- und Lemma-Umgebungen zur Verfügung, die man sich selbst auch umdefinieren kann.

Anwendungsbeispiel:

Theorem
The real function $f(x)$ is positive for large x.

wird erzeugt von

```
\begin{theorem}
  The real function $f(x)$ is positive for large $x$.
\end{theorem}
```