Wissenschaftliches Arbeiten in CSE

Prof. Dr. Karsten Urban,
Dipl. Math. Katharina Becker-Steinberger,
Dipl. Math. oec. Klaus Stolle
Institut für Numerische Mathematik, Universität Ulm

Ulm, 31.10.2012

Klammern

- Etliche Varianten, z.B.
  - runde Klammern `(...)` mittels ()
  - eckige Klammern `[...]` mittels []
  - geschwungene Klammern `{...}` mittels `\{...\}`
  - Absolutbetrag `|...|` mittels | oder `\vert ...\vert`
  - Norm ||...|| mittels `\|...\|`

Klammern

- Etliche Varianten, z.B.
  - runde Klammern `(...)` mittels ()
  - eckige Klammern `[...]` mittels []
  - geschwungene Klammern `{...}` mittels `\{...\}`
  - Absolutbetrag `|...|` mittels | oder `\vert ...\vert`
  - Norm ||...|| mittels `\|...\|`
- größere Größe der Klammern händisch wählbar
  - Präfix `\big, \Big, \bigg, \Bigg` vor Klammer
  - z.B. `\big( (x+1)(x-1)\big)^2= (x^2-1)^2`

    $\Longrightarrow \big((x+1)(x-1)\big)^2 = (x^2-1)^2$

Klammern

- Etliche Varianten, z.B.
  - runde Klammern `(...)` mittels ()
  - eckige Klammern `[...]` mittels []
  - geschwungene Klammern `{...}` mittels `\{...\}`
  - Absolutbetrag `|...|` mittels | oder `\vert ...\vert`
  - Norm ||...|| mittels `\|...\|`
- größere Größe der Klammern händisch wählbar
  - Präfix `\big, \Big, \bigg, \Bigg` vor Klammer
  - z.B. `\big( (x+1)(x-1)\big)^2= (x^2-1)^2`

    $\Longrightarrow \big((x+1)(x-1)\big)^2 = (x^2-1)^2$
- oder Größe automatisch von LaTeX wählbar
  - Präfix `\left, \right` vor Klammer
    - jeder `\left` braucht ein `\right`
    - ggf. nur `\right`, falls nur links Klammer sein soll

Klammern

- Etliche Varianten, z.B.
  - runde Klammern `(...)` mittels ()
  - eckige Klammern `[...]` mittels []
  - geschwungene Klammern `{...}` mittels `\{...\}`
  - Absolutbetrag `|...|` mittels | oder `\vert ...\vert`
  - Norm ||...|| mittels `\|...\|`
- größere Größe der Klammern händisch wählbar
  - Präfix `\big, \Big, \bigg, \Bigg` vor Klammer
  - z.B. `\big( (x+1)(x-1)\big)^2= (x^2-1)^2`

    $\Longrightarrow \big((x+1)(x-1)\big)^2 = (x^2-1)^2$
- oder Größe automatisch von LaTeX wählbar
  - Präfix `\left, \right` vor Klammer
    - jeder `\left` braucht ein `\right`
    - ggf. nur `\right`, falls nur links Klammer sein soll

Mathematische Sonderzeichen

- De facto alles vorhanden (Pakete einbinden!)
  - `\usepackage{latexsym}` `\usepackage{amssymb}`

Exponenten und Indizes

- `$a^x+y\neq a^{x+y}$` $\Longrightarrow$ $a^x + y \neq a^{x+y}$
- `x_{\ell+1}:= x_{\ell}+x_{\ell-1} $` $\Longrightarrow$ $x_{\ell+1} := x_\ell + x_{\ell-1}$

Exponenten und Indizes

- `$a^x+y\neq a^{x+y}$` $\Longrightarrow$ $a^x + y \neq a^{x+y}$
- `x_{\ell+1}:= x_{\ell}+x_{\ell-1} $` $\Longrightarrow$ $x_{\ell+1} := x_\ell + x_{\ell-1}$

Brüche und Wurzeln

- `$\frac{1}{n+1}\neq \frac{1}{n(n+1)}$` $\Longrightarrow$ $\frac{1}{n+1} \neq \frac{1}{n(n+1)}$
- `$\frac{\partial f}{\partial x_j}$` $\Longrightarrow$ $\frac{\partial f}{\partial x_j}$
- `$(\sqrt{x})^{1/3} = x^{1/6}=\sqrt[6]{x}$` $\Longrightarrow$ $(\sqrt{x})^{1/3} = x^{1/6} = \sqrt[6]{x}$

Exponenten und Indizes

- `$a^x+y\neq a^{x+y}$` $\Longrightarrow$ $a^x + y \neq a^{x+y}$
- `x_{\ell+1}:= x_{\ell}+x_{\ell-1} $` $\Longrightarrow$ $x_{\ell+1} := x_\ell + x_{\ell-1}$

Brüche und Wurzeln

- `$\frac{1}{n+1}\neq \frac{1}{n(n+1)}$` $\Longrightarrow$ $\frac{1}{n+1} \neq \frac{1}{n(n+1)}$
- `$\frac{\partial f}{\partial x_j}$` $\Longrightarrow$ $\frac{\partial f}{\partial x_j}$
- `$(\sqrt{x})^{1/3} = x^{1/6}=\sqrt[6]{x}$` $\Longrightarrow$ $(\sqrt{x})^{1/3} = x^{1/6} = \sqrt[6]{x}$

Mengen

- `$y\in \{ f(x)\, | \, x>0\}$` $\Longrightarrow$ $y \in \{f(x) \,|\, x > 0\}$
- `\in` $\in$, `\ni` $\ni$, `\cup` $\cup$, `\bigcup` $\bigcup$, `\cap` $\cap$, `\bigcap` $\bigcap$
- `\backslash` $\backslash$
- `\subset` $\subset$, `\subseteq` $\subseteq$, `\subsetneqq` $\subsetneqq$
- `\supset` $\supset$, `\supseteq` $\supseteq$, `\supsetneqq` $\supsetneqq$

Exponenten und Indizes

- `$a^x+y\neq a^{x+y}$` $\Longrightarrow$ $a^x + y \neq a^{x+y}$
- `x_{\ell+1}:= x_{\ell}+x_{\ell-1} $` $\Longrightarrow$ $x_{\ell+1} := x_\ell + x_{\ell-1}$

Brüche und Wurzeln

- `$\frac{1}{n+1}\neq \frac{1}{n(n+1)}$` $\Longrightarrow$ $\frac{1}{n+1} \neq \frac{1}{n(n+1)}$
- `$\frac{\partial f}{\partial x_j}$` $\Longrightarrow$ $\frac{\partial f}{\partial x_j}$
- `$(\sqrt{x})^{1/3} = x^{1/6}=\sqrt[6]{x}$` $\Longrightarrow$ $(\sqrt{x})^{1/3} = x^{1/6} = \sqrt[6]{x}$

Mengen

- `$y\in \{ f(x)\, | \, x>0\}$` $\Longrightarrow$ $y \in \{f(x) \,|\, x > 0\}$
- `\in` $\in$, `\ni` $\ni$, `\cup` $\cup$, `\bigcup` $\bigcup$, `\cap` $\cap$, `\bigcap` $\bigcap$
- `\backslash` $\backslash$
- `\subset` $\subset$, `\subseteq` $\subseteq$, `\subsetneqq` $\subsetneqq$
- `\supset` $\supset$, `\supseteq` $\supseteq$, `\supsetneqq` $\supsetneqq$

Gleichheit und Ungleichheit

- `=`, `<`, `>`, `\neq` $\neq$, `\le` $\leq$, `\lneqq` $\lneqq$,`\ge` $\geq$, `\qneqq` $\gneqq$

Mathematische Funktionen

- `\exp, \log, \ln, \arg`
- Trigonometrische Funktionen z.B. `\sin, \arccos, \sinh`
- `\sup, \max, \inf, \min`
- `\lim, \limsup, \limif`
  - `$\lim_{x\to 0}\frac{\sin x}{x}=1$`
    $\lim_{x\to 0} \frac{\sin x}{x} = 1$
  - `$\lim\limits{x\to 0}\frac{\sin x}{x}=1$`
    $\lim x \to 0 \frac{\sin x}{x} = 1$
  - `$\displaystyle \lim_{x\to 0}\frac{\sin x}{x}=1$`
    $$\lim_{x\to 0} \frac{\sin x}{x} = 1$$

Summe, Produkt, Integral

Code-Auszug (LaTeX)

```
$\sum_{j=1}^n j= \frac{n(n+1)}{2}$ bzw.

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


\vspace{1cm}



$\prod_{j=1}^{\infty} j=1\cdot 2\cdot 3\cdots$ bzw.

$$\prod_{j=1}^{\infty} j=1\cdot 2\cdot 3\cdots$$


\vspace{1cm}



$\int_0^{\pi/2}\cos(x)\, dx =1$ bzw.

$$\int_0^{\pi/2}\cos(x)\, dx =1$$
```

Ausgabe-Datei (PDF)

$\sum_{j=1}^n j = \frac{n(n+1)}{2}$ bzw.

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

$\prod_{j=1}^{\infty} j = 1 \cdot 2 \cdot 3 \cdots$ bzw.

$$\prod_{j=1}^{\infty} j = 1 \cdot 2 \cdot 3 \cdots$$

$\int_0^{\pi/2} \cos(x)\, dx = 1$ bzw.

$$\int_0^{\pi/2} \cos(x)\, dx = 1$$

Kalligraphische Großbuchstaben

- `$\mathcal{A}$` $\mathcal{A}$, `$\mathcal{B}$` $\mathcal{B}$, `$\mathcal{C}$` $\mathcal{C}$ etc.

Kalligraphische Großbuchstaben

- `$\mathcal{A}$` $\mathcal{A}$, `$\mathcal{B}$` $\mathcal{B}$, `$\mathcal{C}$` $\mathcal{C}$ etc.

Griechische Symbole

- `$\alpha$` α, `$\beta$` β, `$\gamma$` γ, `$\delta$` δ, `$\epsilon$` ϵ etc.
- `$\Gamma$` Γ, `$\Delta$` Δ

Kalligraphische Großbuchstaben

- `$\mathcal{A}$` $\mathcal{A}$, `$\mathcal{B}$` $\mathcal{B}$, `$\mathcal{C}$` $\mathcal{C}$ etc.

Griechische Symbole

- `$\alpha$` α, `$\beta$` β, `$\gamma$` γ, `$\delta$` δ, `$\epsilon$` ϵ etc.
- `$\Gamma$` Γ, `$\Delta$` Δ

Logische Quantoren

- `\forall\, x>0:\quad x^2>0` $\Longrightarrow$ $\forall\, x > 0 : \quad x^2 > 0$
- `\forall\, T\mbox{ Topf } \exists\, D \mbox{ Deckel }` $\Longrightarrow$ $\forall\, T$ Topf $\exists\, D$ Deckel

Kalligraphische Großbuchstaben

- `$\mathcal{A}$` $\mathcal{A}$, `$\mathcal{B}$` $\mathcal{B}$, `$\mathcal{C}$` $\mathcal{C}$ etc.

Griechische Symbole

- `$\alpha$` α, `$\beta$` β, `$\gamma$` γ, `$\delta$` δ, `$\epsilon$` ϵ etc.
- `$\Gamma$` Γ, `$\Delta$` Δ

Logische Quantoren

- `\forall\, x>0:\quad x^2>0` $\Longrightarrow$ $\forall\, x > 0 : \quad x^2 > 0$
- `\forall\, T\mbox{ Topf } \exists\, D \mbox{ Deckel }` $\Longrightarrow$ $\forall\, T$ Topf $\exists\, D$ Deckel

Blackboard-Großbuchstaben

- `\usepackage{amssymb}` erforderlich!
- `\mathbb{N}` $\mathbb{N}$, `\mathbb{Z}` $\mathbb{Z}$, `\mathbb{R}` $\mathbb{R}$, `\mathbb{C}` $\mathbb{C}$ etc.

Vektoren & Matrizen

Code-Auszug (LaTeX)

```
X=
\left(
     \begin{array}{ccc}
            x_{11} & x_{12} & \ldots\\
            x_{21} & x_{22} &\ldots\\
            \vdots  & \vdots  & \ldots
     \end{array}
\right)
```

Ausgabe-Datei (PDF)

$$X = \left(\begin{array}{ccc} x_{11} & x_{12} & \ldots \\ x_{21} & x_{22} & \ldots \\ \vdots & \vdots & \ldots \end{array} \right)$$

- `array`-Umgebung für Matrizen und Vektoren (= Matrix mit einer Spalte)
  - beliebig viele Zeilen
    - Zeilenumbruch jeweils mit `\\`
  - Anzahl Spalten + Ausrichtung muss angegeben werden hier: 3 Spalten mit mittiger Ausrichtung: `{ccc}`
    - Ausrichtung: mittig `{c}`, links `{l}`, rechts `{rrr}`

Vektoren & Matrizen

Code-Auszug (LaTeX)

```
X=
\left(
     \begin{array}{ccc}
            x_{11} & x_{12} & \ldots\\
            x_{21} & x_{22} &\ldots\\
            \vdots  & \vdots  & \ldots
     \end{array}
\right)
```

Ausgabe-Datei (PDF)

$$X = \left(\begin{array}{ccc} x_{11} & x_{12} & \ldots \\ x_{21} & x_{22} & \ldots \\ \vdots & \vdots & \ldots \end{array} \right)$$

- `array`-Umgebung für Matrizen und Vektoren (= Matrix mit einer Spalte)
  - beliebig viele Zeilen
    - Zeilenumbruch jeweils mit `\\`
  - Anzahl Spalten + Ausrichtung muss angegeben werden hier: 3 Spalten mit mittiger Ausrichtung: `{ccc}`
    - Ausrichtung: mittig `{c}`, links `{l}`, rechts `{rrr}`
- `array`-Umgebung ist ein Teil einer mathematischen Formel
  - z.B. `$...$`, `equation`-Umgebung

Vektoren & Matrizen

Code-Auszug (LaTeX)

```
X=
\left(
     \begin{array}{ccc}
            x_{11} & x_{12} & \ldots\\
            x_{21} & x_{22} &\ldots\\
            \vdots  & \vdots  & \ldots
     \end{array}
\right)
```

Ausgabe-Datei (PDF)

$$X = \left(\begin{array}{ccc} x_{11} & x_{12} & \ldots \\ x_{21} & x_{22} & \ldots \\ \vdots & \vdots & \ldots \end{array} \right)$$

- `array`-Umgebung für Matrizen und Vektoren (= Matrix mit einer Spalte)
  - beliebig viele Zeilen
    - Zeilenumbruch jeweils mit `\\`
  - Anzahl Spalten + Ausrichtung muss angegeben werden hier: 3 Spalten mit mittiger Ausrichtung: `{ccc}`
    - Ausrichtung: mittig `{c}`, links `{l}`, rechts `{rrr}`
- `array`-Umgebung ist ein Teil einer mathematischen Formel
  - z.B. `$...$`, `equation`-Umgebung
- `array`-Umgebung auch für Fallunterscheidung
  - Verwende `\left\{` mit `\right`

Referenzen

- In mathematischen Absätzen gibt es häufig Referenzen
  - auf Formeln, z.B. siehe Formel (2.7)
  - auf Seiten, z.B. in Formel (2.7) auf Seite 10
  - auf Bilder, z.B. siehe Abbildung 2.3
  - auf Abschnitte, z.B. siehe Kapitel 3
  - auf Sätze, z.B. siehe Satz 2.4

Referenzen

- In mathematischen Absätzen gibt es häufig Referenzen
  - auf Formeln, z.B. siehe Formel (2.7)
  - auf Seiten, z.B. in Formel (2.7) auf Seite 10
  - auf Bilder, z.B. siehe Abbildung 2.3
  - auf Abschnitte, z.B. siehe Kapitel 3
  - auf Sätze, z.B. siehe Satz 2.4
- **Wichtig:** Referenzen werden in LaTeX nicht hart kodiert!

Referenzen

- In mathematischen Absätzen gibt es häufig Referenzen
  - auf Formeln, z.B. siehe Formel (2.7)
  - auf Seiten, z.B. in Formel (2.7) auf Seite 10
  - auf Bilder, z.B. siehe Abbildung 2.3
  - auf Abschnitte, z.B. siehe Kapitel 3
  - auf Sätze, z.B. siehe Satz 2.4
- **Wichtig:** Referenzen werden in LaTeX nicht hart kodiert!
- **Hilfreich:** `\usepackage{showkeys}` zeigt Referenzen & Label an
  - zum Schreiben des Dokuments sinnvoll

Referenzen

- In mathematischen Absätzen gibt es häufig Referenzen
  - auf Formeln, z.B. siehe Formel (2.7)
  - auf Seiten, z.B. in Formel (2.7) auf Seite 10
  - auf Bilder, z.B. siehe Abbildung 2.3
  - auf Abschnitte, z.B. siehe Kapitel 3
  - auf Sätze, z.B. siehe Satz 2.4
- **Wichtig:** Referenzen werden in LaTeX nicht hart kodiert!
- **Hilfreich:** `\usepackage{showkeys}` zeigt Referenzen & Label an
  - zum Schreiben des Dokuments sinnvoll

Vorgehen

1. Voraussetzung: Man setzt Label
   - durch `\label{name}`
   - LaTeX verknüpft intern das Label `name` mit zuletzt vorausgegangener Zähler-Auswertung

Referenzen

- In mathematischen Absätzen gibt es häufig Referenzen
  - auf Formeln, z.B. siehe Formel (2.7)
  - auf Seiten, z.B. in Formel (2.7) auf Seite 10
  - auf Bilder, z.B. siehe Abbildung 2.3
  - auf Abschnitte, z.B. siehe Kapitel 3
  - auf Sätze, z.B. siehe Satz 2.4
- **Wichtig:** Referenzen werden in LaTeX nicht hart kodiert!
- **Hilfreich:** `\usepackage{showkeys}` zeigt Referenzen & Label an
  - zum Schreiben des Dokuments sinnvoll

Vorgehen

1. Voraussetzung: Man setzt Label
   - durch `\label{name}`
   - LaTeX verknüpft intern das Label `name` mit zuletzt vorausgegangener Zähler-Auswertung
2. Im Text Referenzen einfügen durch
   - `\ref{name}`: nur Zählerausgabe
   - `\eqref{ref}`: Zählerausgabe für Gleichung
     - benötigt `\usepackage{amsmath}`
   - `\pageref{name}`: Ausgabe der Seitenzahl

LaTeX-Warnungen

- LaTeX speichert Labels in AUX-Datei

LaTeX-Warnungen

- LaTeX speichert Labels in AUX-Datei
- LaTeXerkennt, falls Referenzen neu sind
  - LOG-File endet in diesem Fall mit
    `LaTeX Warning: Label(s) may have changed.`
    `Return to get cross-references right`
  - Dann: LaTeX-File noch ein mal kompilieren
- LaTeX erkennt, falls Label doppelt benutzt wird
  - `LaTeX Warning: Label 'X' multiply defined.`
  - LOG-File endet in diesem Fall mit
    `LaTeX Warning: There were multiply-defined labels.`
- LaTeX gibt Warnung, falls Label unbekannt ist
  - `LaTeX Warning: Reference 'X' on page XX.`
    `undefined on input line XXX`
  - LOG-File endet in diesem Fall mit
    `LaTeX Warning: There were undefined references.`

Definieren von Makros

- Definition eines neuen Makros mittels
  - `\newcommand{\name}[anz]{definition}`

Definieren von Makros

- Definition eines neuen Makros mittels
  - `\newcommand{\name}[anz]{definition}`
- Obligatorisch sind
  - Name des Makros `name`
  - Befehlsfolge des Makros `definition`

Definieren von Makros

- Definition eines neuen Makros mittels
  - `\newcommand{\name}[anz]{definition}`
- Obligatorisch sind
  - Name des Makros `name`
  - Befehlsfolge des Makros `definition`
- Optional ist Anzahl `anz` der obligatorischen Parameter des Makros
  - Fehlt `anz` so ist `\name` parameterlos
  - maximal 9 Parameter, intern `#1, ...#9`

Definieren von Makros

- Definition eines neuen Makros mittels
  - `\newcommand{\name}[anz]{definition}`
- Obligatorisch sind
  - Name des Makros `name`
  - Befehlsfolge des Makros `definition`
- Optional ist Anzahl `anz` der obligatorischen Parameter des Makros
  - Fehlt `anz` so ist `\name` parameterlos
  - maximal 9 Parameter, intern `#1, ...#9`
- Beispiele:
  - `\newcommand{\R}{\mathbb{R}}`
    - Aufruf: `\R`
    - Ausgabe: $\mathbb{R}$
  - `\newcommand{\norm}[1]{\left\| #1\right\|}`
    - Aufruf: `\norm{f}`
    - Ausgabe: $\|f\|$
  - `\newcommand{\set}[2]{\big\{ #1\,\big| \, #2\right\} }`
    - Aufruf: `\set{x\in\R}{f(x)=0}`
    - Ausgabe: $\big\{x \in \mathbb{R} \,\big|\, f(x) = 0\big\}$

Definieren von Makros

- Definition eines neuen Makros mittels
  - `\newcommand{\name}[anz]{definition}`
- Obligatorisch sind
  - Name des Makros `name`
  - Befehlsfolge des Makros `definition`
- Optional ist Anzahl `anz` der obligatorischen Parameter des Makros
  - Fehlt `anz` so ist `\name` parameterlos
  - maximal 9 Parameter, intern `#1, ...#9`
- Beispiele:
  - `\newcommand{\R}{\mathbb{R}}`
    - Aufruf: `\R`
    - Ausgabe: $\mathbb{R}$
  - `\newcommand{\norm}[1]{\left\| #1\right\|}`
    - Aufruf: `\norm{f}`
    - Ausgabe: $\|f\|$
  - `\newcommand{\set}[2]{\big\{ #1\,\big| \, #2\right\} }`
    - Aufruf: `\set{x\in\R}{f(x)=0}`
    - Ausgabe: $\big\{x \in \mathbb{R} \,\big|\, f(x) = 0\big\}$
- LaTeX passt auf, ob Makro-Name bereits vergeben ist
  - `! LaTeX Error: Command XXX already defined.`
  - Altes Überschreiben mittels `\renewcommand`
    - Parameter/Verwendung wie bei `\newcommand`

Warum Makros?

- Lesbarkeit des Codes, insbesondere Mathematischer Formeln
  - `\big\{x\in\mathbb{R}\, \big| \, f(x)=0\big\}`
    versus
  - `\set{x\in\R}{f(x)=0}`

Warum Makros?

- Lesbarkeit des Codes, insbesondere Mathematischer Formeln
  - `\big\{x\in\mathbb{R}\, \big| \, f(x)=0\big\}` versus
  - `\set{x\in\R}{f(x)=0}`
- Code wird kürzer & übersichtlicher

Warum Makros?

- Lesbarkeit des Codes, insbesondere Mathematischer Formeln
  - `\big\{x\in\mathbb{R}\, \big| \, f(x)=0\big\}` versus
  - `\set{x\in\R}{f(x)=0}`
- Code wird kürzer & übersichtlicher
- Vermeidung von Tippfehlern

Warum Makros?

- Lesbarkeit des Codes, insbesondere Mathematischer Formeln
  - `\big\{x\in\mathbb{R}\, \big| \, f(x)=0\big\}` versus
  - `\set{x\in\R}{f(x)=0}`
- Code wird kürzer & übersichtlicher
- Vermeidung von Tippfehlern
- Wiederverwendbarkeit in weiteren Dokumenten

Warum Makros?

- Lesbarkeit des Codes, insbesondere Mathematischer Formeln
  - `\big\{x\in\mathbb{R}\, \big| \, f(x)=0\big\}` versus
  - `\set{x\in\R}{f(x)=0}`
- Code wird kürzer & übersichtlicher
- Vermeidung von Tippfehlern
- Wiederverwendbarkeit in weiteren Dokumenten
- einfache Anpassung von math. Notation
  - z.B. $\left\{x \in \mathbb{R} \,\middle|\, f(x) = 0\right\}$ vs. $\left\{x \in \mathbb{R} : f(x) = 0\right\}$

Warum Makros?

- Lesbarkeit des Codes, insbesondere Mathematischer Formeln
  - `\big\{x\in\mathbb{R}\, \big| \, f(x)=0\big\}` versus
  - `\set{x\in\R}{f(x)=0}`
- Code wird kürzer & übersichtlicher
- Vermeidung von Tippfehlern
- Wiederverwendbarkeit in weiteren Dokumenten
- einfache Anpassung von math. Notation
  - z.B. $\left\{x \in \mathbb{R} \,\middle|\, f(x) = 0\right\}$ vs. $\left\{x \in \mathbb{R} : f(x) = 0\right\}$
- Umstellung der Notation im gesamten Dokument durch Änderung einer Zeile

Was sollte man bei der Definition von Makros beachten?

- Sprechende Namen für Makros verwenden
  - `\set , \norm, \scalarproduct`

Was sollte man bei der Definition von Makros beachten?

- Sprechende Namen für Makros verwenden
  - `\set , \norm, \scalarproduct`
- Kurze Namen nur für reine Zeichen z.B.
  - `\bbN, \bbZ, \bbR` etc. für mathbb-Symbole $\mathbb{N}, \mathbb{Z}, \mathbb{R}$
  - `\cA, \cB, \cC` etc. für mathcal-Symbole $\mathcal{A}, \mathcal{B}, \mathcal{C}$
  - `\x, \y, \z` etc. für Vektoren $\mathbb{N}, \mathbb{Z}, \mathbb{R}$

Was sollte man bei der Definition von Makros beachten?

- Sprechende Namen für Makros verwenden
  - `\set , \norm, \scalarproduct`
- Kurze Namen nur für reine Zeichen z.B.
  - `\bbN, \bbZ, \bbR` etc. für mathbb-Symbole $\mathbb{N}, \mathbb{Z}, \mathbb{R}$
  - `\cA, \cB, \cC` etc. für mathcal-Symbole $\mathcal{A}, \mathcal{B}, \mathcal{C}$
  - `\x, \y, \z` etc. für Vektoren $\mathbb{N}, \mathbb{Z}, \mathbb{R}$
- Keine Makros zur puren Abkürzung von Tipparbeit
  - Solchen Code kann man später nicht mehr lesen!

Makros mit optionalen Parametern

- Man kann Makros definieren deren erster Parameter optional ist
  - `\newcommand{\name}[anz][default1]{definition}`

Makros mit optionalen Parametern

- Man kann Makros definieren deren erster Parameter optional ist
  - `\newcommand{\name}[anz][default1]{definition}`
  - `name`, `anz`, `definition` wie bisher

Makros mit optionalen Parametern

- Man kann Makros definieren deren erster Parameter optional ist
  - `\newcommand{\name}[anz][default1]{definition}`
  - `name`, `anz`, `definition` wie bisher
  - Parameter `#1` ist optional
    - Übergabe in eckigen Klammern `[parameter1]`
    - Wert `default1`, falls nicht gegeben

Makros mit optionalen Parametern

- Man kann Makros definieren deren erster Parameter optional ist
  - `\newcommand{\name}[anz][default1]{definition}`
  - `name`, `anz`, `definition` wie bisher
  - Parameter `#1` ist optional
    - Übergabe in eckigen Klammern `[parameter1]`
    - Wert `default1`, falls nicht gegeben
  - Parameter `#2, ...#anz` sind obligatorisch
    - Übergabe in Klammern `{parameter}`

Makros mit optionalen Parametern

- Man kann Makros definieren deren erster Parameter optional ist
  - `\newcommand{\name}[anz][default1]{definition}`
  - `name`, `anz`, `definition` wie bisher
  - Parameter `#1` ist optional
    - Übergabe in eckigen Klammern `[parameter1]`
    - Wert `default1`, falls nicht gegeben
  - Parameter `#2, ...#anz` sind obligatorisch
    - Übergabe in Klammern `{parameter}`

- Beispiel:
  - `\newcommand{\norm}[2][] {\left \| #2\right\|_{#1}}`

Makros mit optionalen Parametern

- Man kann Makros definieren deren erster Parameter optional ist
  - `\newcommand{\name}[anz][default1]{definition}`
  - `name`, `anz`, `definition` wie bisher
  - Parameter `#1` ist optional
    - Übergabe in eckigen Klammern `[parameter1]`
    - Wert `default1`, falls nicht gegeben
  - Parameter `#2, ...#anz` sind obligatorisch
    - Übergabe in Klammern `{parameter}`

- Beispiel:
  - `\newcommand{\norm}[2][] {\left \| #2\right\|_{#1}}`
    - Aufruf: `\norm[L^2(\Omega)]{f}` Ausgabe: $\|f\|_{L^2(\Omega)}$

Makros mit optionalen Parametern

- Man kann Makros definieren deren erster Parameter optional ist
  - `\newcommand{\name}[anz][default1]{definition}`
  - `name`, `anz`, `definition` wie bisher
  - Parameter `#1` ist optional
    - Übergabe in eckigen Klammern `[parameter1]`
    - Wert `default1`, falls nicht gegeben
  - Parameter `#2, ...#anz` sind obligatorisch
    - Übergabe in Klammern `{parameter}`

- Beispiel:
  - `\newcommand{\norm}[2][] {\left \| #2\right\|_{#1}}`
    - Aufruf: `\norm[L^2(\Omega)]{f}` Ausgabe: $\|f\|_{L^2(\Omega)}$
    - Aufruf: `\norm{f}` Ausgabe: $\|f\|$

Makros mit optionalen Parametern

- Man kann Makros definieren deren erster Parameter optional ist
  - `\newcommand{\name}[anz][default1]{definition}`
  - `name`, `anz`, `definition` wie bisher
  - Parameter `#1` ist optional
    - Übergabe in eckigen Klammern `[parameter1]`
    - Wert `default1`, falls nicht gegeben
  - Parameter `#2, ...#anz` sind obligatorisch
    - Übergabe in Klammern `{parameter}`

- Beispiel:
  - `\newcommand{\norm}[2][] {\left \| #2\right\|_{#1}}`
    - Aufruf: `\norm[L^2(\Omega)]{f}` Ausgabe: $\|f\|_{L^2(\Omega)}$
    - Aufruf: `\norm{f}` Ausgabe: $\|f\|$
  - `\newcommand{\set}[3][\Big] {#1 \{\ #2, #1\|, #3#1\}}`

Makros mit optionalen Parametern

- Man kann Makros definieren deren erster Parameter optional ist
  - `\newcommand{\name}[anz][default1]{definition}`
  - `name`, `anz`, `definition` wie bisher
  - Parameter `#1` ist optional
    - Übergabe in eckigen Klammern `[parameter1]`
    - Wert `default1`, falls nicht gegeben
  - Parameter `#2, ...#anz` sind obligatorisch
    - Übergabe in Klammern `{parameter}`

- Beispiel:
  - `\newcommand{\norm}[2][] {\left \| #2\right\|_{#1}}`
    - Aufruf: `\norm[L^2(\Omega)]{f}` Ausgabe: $\|f\|_{L^2(\Omega)}$
    - Aufruf: `\norm{f}` Ausgabe: $\|f\|$
  - `\newcommand{\set}[3][\Big] {#1 \{\ #2, #1\|, #3#1\}}`
    - Aufruf: `\set{x\in\R}{f(x)=0}` Ausgabe: $\{x \in \mathbb{R} | f(x) = 0\}$

Makros mit optionalen Parametern

- Man kann Makros definieren deren erster Parameter optional ist
  - `\newcommand{\name}[anz][default1]{definition}`
  - `name`, `anz`, `definition` wie bisher
  - Parameter `#1` ist optional
    - Übergabe in eckigen Klammern `[parameter1]`
    - Wert `default1`, falls nicht gegeben
  - Parameter `#2, ...#anz` sind obligatorisch
    - Übergabe in Klammern `{parameter}`

- Beispiel:
  - `\newcommand{\norm}[2][] {\left \| #2\right\|_{#1}}`
    - Aufruf: `\norm[L^2(\Omega)]{f}` Ausgabe: $\|f\|_{L^2(\Omega)}$
    - Aufruf: `\norm{f}` Ausgabe: $\|f\|$
  - `\newcommand{\set}[3][\Big] {#1 \{\ #2, #1\|, #3#1\}}`
    - Aufruf: `\set{x\in\R}{f(x)=0}` Ausgabe: $\{x \in \mathbb{R} | f(x) = 0\}$
    - Aufruf: `\set[\Big]{x\in\R}{f(x)=0}\norm{f}` Ausgabe: $\left\{x \in \mathbb{R} | f(x) = 0\right\}$

Vordefinierte Zähler

- Abhängig von der Dokument-Klasse gibt es Zähler für die Gliederung
  - `chapter, section, subsection` etc.

Vordefinierte Zähler

- Abhängig von der Dokument-Klasse gibt es Zähler für die Gliederung
  - `chapter, section, subsection` etc.
- Weitere Zähler sind
  - `page, equation, figure, table`

Vordefinierte Zähler

- Abhängig von der Dokument-Klasse gibt es Zähler für die Gliederung
  - `chapter, section, subsection` etc.
- Weitere Zähler sind
  - `page, equation, figure, table`
- Auswertung eines Zählers
  - `\arabic{counter}` = 1, 2, 3, 4 etc
  - `\roman{counter}` = i, ii, iii, iv etc.
  - `\Roman{counter}` = I, II, III, IV etc.
  - `\alpha{counter}` = a, b, c, d (counter $\leq$ 26)
  - `\Alpha{counter}` = A, B, C, D (counter $\leq$ 26)

Vordefinierte Zähler

- Abhängig von der Dokument-Klasse gibt es Zähler für die Gliederung
  - `chapter, section, subsection` etc.
- Weitere Zähler sind
  - `page, equation, figure, table`
- Auswertung eines Zählers
  - `\arabic{counter}` = 1, 2, 3, 4 etc
  - `\roman{counter}` = i, ii, iii, iv etc.
  - `\Roman{counter}` = I, II, III, IV etc.
  - `\alpha{counter}` = a, b, c, d (counter $\leq$ 26)
  - `\Alpha{counter}` = A, B, C, D (counter $\leq$ 26)

Vordefinierte Zähler

- Abhängig von der Dokument-Klasse gibt es Zähler für die Gliederung
  - `chapter, section, subsection` etc.
- Weitere Zähler sind
  - `page, equation, figure, table`
- Auswertung eines Zählers
  - `\arabic{counter}` = 1, 2, 3, 4 etc
  - `\roman{counter}` = i, ii, iii, iv etc.
  - `\Roman{counter}` = I, II, III, IV etc.
  - `\alpha{counter}` = a, b, c, d (counter ≤ 26)
  - `\Alpha{counter}` = A, B, C, D (counter ≤ 26)
- Zu jedem `counter` gehört ein Ausgabebefehl `\thecounter`, der u.a. von `\ref` aufgerufen wird
- Beispiel: Nummerierung der Gleichung mit Kapitel + Abschnitt + Formel `\renewcommand{\theequation}{\arabic{chapter}.\arabic{section}. \arabic{equation}}`
- Wertzuweisung eines Zählers
  - `\setcounter{counter}{zahl}`
- Zähler um 1 erhöhen & referenzierbar machen
  - `\refstepcounter{counter}`

Eigene Zähler definieren

- `\newcounter{\newcounter}[oldcounter]`

Eigene Zähler definieren

- `\newcounter{\newcounter}[oldcounter]`
- Falls obligatorischer Parameter `oldcounter` angegeben wird, wird `newcounter` automatisch durch `\refstepcounter{oldcounter}` auf 0 gesetzt

Eigene Zähler definieren

- `\newcounter{\newcounter}[oldcounter]`
- Falls obligatorischer Parameter `oldcounter` angegeben wird, wird `newcounter` automatisch durch `\refstepcounter{oldcounter}` auf 0 gesetzt
- Beispiel: Sätze kapitelweise nummerieren
  - Satz 1.1, Satz 1.2, ... Satz 2.1 etc.

Eigene Zähler definieren

- `\newcounter{\newcounter}[oldcounter]`
- Falls obligatorischer Parameter `oldcounter` angegeben wird, wird `newcounter` automatisch durch `\refstepcounter{oldcounter}` auf 0 gesetzt
- Beispiel: Sätze kapitelweise nummerieren
  - Satz 1.1, Satz 1.2, ... Satz 2.1 etc.
- Ausgabe des Zählers festlegen
  - `\renewcommand{\thenewcounter}{...}`

Eigene Zähler definieren

- `\newcounter{\newcounter}[oldcounter]`
- Falls obligatorischer Parameter `oldcounter` angegeben wird, wird `newcounter` automatisch durch `\refstepcounter{oldcounter}` auf 0 gesetzt
- Beispiel: Sätze kapitelweise nummerieren
  - Satz 1.1, Satz 1.2, ... Satz 2.1 etc.
- Ausgabe des Zählers festlegen
  - `\renewcommand{\thenewcounter}{...}`

Quelldatei (LaTeX)

```
1  % zaeler.tex
2  \documentclass[a4paper,12pt]{report}
3  \usepackage{fullpage}
4
5  \newcounter{const}
6  \renewcommand{\theconst}{\arabic{const}}
7
8  \newcommand{\newconst}[1]{%
9  \refstepcounter{const}%
10 C_{\theconst}\label{const:#1}%
11 }
12 \newcommand{\const}[1]{\ref{const:#1}}
13
14 \begin{document}
15 Seien $\newconst{2},\newconst{1}>0$
16 und es gelte $\const{2}\le \const{1}$
17 \end{document}
```

Ausgabe-Datei (PDF)

Seien $C_1, C_2 > 0$ und es gelte $1 \leq 2$

- Beispiel 14: Zaeler.tex

Vordefinierte Zähler bearbeiten

- Standardmäßig zählt `equation` bei der Dokument-Klasse `article` global

Vordefinierte Zähler bearbeiten

- Standardmäßig zählt `equation` bei der Dokument-Klasse `article` global
- Standardmäßig zählt `equation` bei der Dokument-Klasse `report` oder `book` kapitelweise

Vordefinierte Zähler bearbeiten

- Standardmäßig zählt `equation` bei der Dokument-Klasse `article` global
- Standardmäßig zählt `equation` bei der Dokument-Klasse `report` oder `book` kapitelweise
- Neu-Definition der Zählerabhängigkeit zum Zurücksetzen auf Null mittels `\numberwithin[format]{counter}{recounter}`

Vordefinierte Zähler bearbeiten

- Standardmäßig zählt `equation` bei der Dokument-Klasse `article` global
- Standardmäßig zählt `equation` bei der Dokument-Klasse `report` oder `book` kapitelweise
- Neu-Definition der Zählerabhängigkeit zum Zurücksetzen auf Null mittels `\numberwithin[format]{counter}{recounter}`
  - format = `\arabic`, `\roman`, `\alpha` etc.
    - Standard ist `\arabic`

Vordefinierte Zähler bearbeiten

- Standardmäßig zählt `equation` bei der Dokument-Klasse `article` global
- Standardmäßig zählt `equation` bei der Dokument-Klasse `report` oder `book` kapitelweise
- Neu-Definition der Zählerabhängigkeit zum Zurücksetzen auf Null mittels `\numberwithin[format]{counter}{recounter}`
  - format = `\arabic`, `\roman`, `\alpha` etc.
    - Standard ist `\arabic`
  - z.B. `\numberwithin{equation}{section}`
    - Nummerierung = `\thesection.\arabic{equation}`
    - Erste Formel in neuer Section hat nun stets Nummer 1

Vordefinierte Zähler bearbeiten

- Standardmäßig zählt `equation` bei der Dokument-Klasse `article` global
- Standardmäßig zählt `equation` bei der Dokument-Klasse `report` oder `book` kapitelweise
- Neu-Definition der Zählerabhängigkeit zum Zurücksetzen auf Null mittels `\numberwithin[format]{counter}{recounter}`
  - format = `\arabic`, `\roman`, `\alpha` etc.
    - Standard ist `\arabic`
  - z.B. `\numberwithin{equation}{section}`
    - Nummerierung = `\thesection.\arabic{equation}`
    - Erste Formel in neuer Section hat nun stets Nummer 1
  - benötigt `\usepackage{amsmath}`

Weitere Text-Umgebungen

- Wir kennen bereits `center, flushright, flushleft`-Umgebung

Weitere Text-Umgebungen

- Wir kennen bereits `center, flushright, flushleft`-Umgebung
- für Zitate: `quote`-Umgebung

Weitere Text-Umgebungen

- Wir kennen bereits `center, flushright, flushleft`-Umgebung
- für Zitate: `quote`-Umgebung
- als ob Schreibmaschine `verbatim`-Umgebung

Weitere Text-Umgebungen

- Wir kennen bereits `center, flushright, flushleft`-Umgebung
- für Zitate: `quote`-Umgebung
- als ob Schreibmaschine `verbatim`-Umgebung
- Aufzählungen: `itemize`-Umgebung
  - jeder Punkt wird mit `\item` eingeleitet
  - optional `\item[zeichen]` für anderes Symbol

Weitere Text-Umgebungen

- Wir kennen bereits `center, flushright, flushleft`-Umgebung
- für Zitate: `quote`-Umgebung
- als ob Schreibmaschine `verbatim`-Umgebung
- Aufzählungen: `itemize`-Umgebung
  - jeder Punkt wird mit `\item` eingeleitet
  - optional `\item[zeichen]` für anderes Symbol
- Nummerierte Aufzählung: `enumerate`-Umgebung
  - jeder Punkt wird mit `\item` eingeleitet
  - Art der Aufzählung über Zähler manipulierbar
    - `enumi`
    - `enumii, enumiii, enumiv` bei geschachtelten `enumerate`-Umgebungen
  - `\usepackage{enumerate}` hat mehr Funktionalität
    - Erweiterung der `enumerate`-Umgebungen um optionale Layout-Parameter

Warum Umgebungen?

- Viele Objekte in mathematischen Texten sollten dasselbe Layout haben
  - z.B. Sätze, Definitionen, Beweise

Warum Umgebungen?

- Viele Objekte in mathematischen Texten sollten dasselbe Layout haben
  - z.B. Sätze, Definitionen, Beweise
- Umgebungen trennen Inhalt und Layout
  - Code wird lesbarer
  - Layout wird leichter veränderbar

Warum Umgebungen?

- Viele Objekte in mathematischen Texten sollten dasselbe Layout haben
  - z.B. Sätze, Definitionen, Beweise
- Umgebungen trennen Inhalt und Layout
  - Code wird lesbarer
  - Layout wird leichter veränderbar

Definition einer Umgebung

- `\newenvironment{name}[anz]{defbegin}{defend}`

Warum Umgebungen?

- Viele Objekte in mathematischen Texten sollten dasselbe Layout haben
  - z.B. Sätze, Definitionen, Beweise
- Umgebungen trennen Inhalt und Layout
  - Code wird lesbarer
  - Layout wird leichter veränderbar

Definition einer Umgebung

- `\newenvironment{name}[anz]{defbegin}{defend}`
- `name, anz` wie bei `\newcommand`

Warum Umgebungen?

- Viele Objekte in mathematischen Texten sollten dasselbe Layout haben
  - z.B. Sätze, Definitionen, Beweise
- Umgebungen trennen Inhalt und Layout
  - Code wird lesbarer
  - Layout wird leichter veränderbar

Definition einer Umgebung

- `\newenvironment{name}[anz]{defbegin}{defend}`
- `name`, `anz` wie bei `\newcommand`
- `defbegin` = Was löst `\begin{name}` aus?
- `defend` = Was löst `\end{name}` aus?

Warum Umgebungen?

- Viele Objekte in mathematischen Texten sollten dasselbe Layout haben
  - z.B. Sätze, Definitionen, Beweise
- Umgebungen trennen Inhalt und Layout
  - Code wird lesbarer
  - Layout wird leichter veränderbar

Definition einer Umgebung

- `\newenvironment{name}[anz]{defbegin}{defend}`
- `name`, `anz` wie bei `\newcommand`
- `defbegin` = Was löst `\begin{name}` aus?
- `defend` = Was löst `\end{name}` aus?
- `\renewenvironment` analog zu `\renewcommand`
- Beispiel:
  - `\newenvironment{proof}{\textbf{Beweis.}}{\hfill\textbf{q.e.d.}}`
  - Eingabe: `\begin{proof} ...\end{proof}` Ausgabe: **Beweis.** ... **q.e.d.**

Wichtige mathematische Pakete

- `amsmath` = Umgebungen, Befehle
  - im Folgenden `\usepackage{amsmath}` notwendig!
- `amsth` = Theorem-Umgebungen
- `amsfonts`, `amssymb` = Schriftarten + Symbole

Wichtige mathematische Pakete

- `amsmath` = Umgebungen, Befehle
  - im Folgenden `\usepackage{amsmath}` notwendig!
- `amsth` = Theorem-Umgebungen
- `amsfonts`, `amssymb` = Schriftarten + Symbole

Praktische Umgebungen

- `matrix`-Umgebung für Vektoren und Matrizen
  - bequemer als `array`-Umgebung, weil man die Anzahl der Spalten nicht angeben muss
  - ansonsten gleiche Syntax
    - zeilenweise Angabe
    - & für neue Spalte
    - \\ für neue Zeile

Wichtige mathematische Pakete

- `amsmath` = Umgebungen, Befehle
  - im Folgenden `\usepackage{amsmath}` notwendig!
- `amsth` = Theorem-Umgebungen
- `amsfonts`, `amssymb` = Schriftarten + Symbole

Praktische Umgebungen

- `matrix`-Umgebung für Vektoren und Matrizen
  - bequemer als `array`-Umgebung, weil man die Anzahl der Spalten nicht angeben muss
  - ansonsten gleiche Syntax
    - zeilenweise Angabe
    - & für neue Spalte
    - \\ für neue Zeile
- `pmatrix`-Umgebung
  - =`\left(\begin{matrix}...\end{matrix}\right)`

Wichtige mathematische Pakete

- `amsmath` = Umgebungen, Befehle
  - im Folgenden `\usepackage{amsmath}` notwendig!
- `amsth` = Theorem-Umgebungen
- `amsfonts`, `amssymb` = Schriftarten + Symbole

Praktische Umgebungen

- `matrix`-Umgebung für Vektoren und Matrizen
  - bequemer als `array`-Umgebung, weil man die Anzahl der Spalten nicht angeben muss
  - ansonsten gleiche Syntax
    - zeilenweise Angabe
    - & für neue Spalte
    - \\ für neue Zeile
- `pmatrix`-Umgebung
  - =`\left(\begin{matrix}...\end{matrix}\right)`
- `case`-Umgebung
  - =`\left(\begin{array}[ll]...\end{array}\right`

Wichtige mathematische Pakete

- `amsmath` = Umgebungen, Befehle
  - im Folgenden `\usepackage{amsmath}` notwendig!
- `amsth` = Theorem-Umgebungen
- `amsfonts`, `amssymb` = Schriftarten + Symbole

Praktische Umgebungen

- `matrix`-Umgebung für Vektoren und Matrizen
  - bequemer als `array`-Umgebung, weil man die Anzahl der Spalten nicht angeben muss
  - ansonsten gleiche Syntax
    - zeilenweise Angabe
    - & für neue Spalte
    - \\ für neue Zeile
- `pmatrix`-Umgebung
  - =`\left(\begin{matrix}...\end{matrix}\right)`
- `case`-Umgebung
  - =`\left(\begin{array}[ll]...\end{array}\right`
- Beispiel 15: `array`, `pmatrix` und `binom`

Die `align`-Umgebung

- Flexibler als `equation`- und `eqnarray`-Umgebung

Die `align`-Umgebung

- Flexibler als `equation`- und `eqnarray`-Umgebung
- Mit (`align`) und ohne (`align*`) Formelnummer

Die `align`-Umgebung

- Flexibler als `equation`- und `eqnarray`-Umgebung
- Mit (`align`) und ohne (`align*`) Formelnummer
- Erlaubt mehrzeilige Formeln, Zeilenumbruch `\\`

Die `align`-Umgebung

- Flexibler als `equation`- und `eqnarray`-Umgebung
- Mit (`align`) und ohne (`align*`) Formelnummer
- Erlaubt mehrzeilige Formeln, Zeilenumbruch \\
- Ordnet tabellarisch an
  - & neue Spalte
  - Spalten abwechselnd links/rechts ausgerichtet
  - Spaltenpaar links/rechts bildet jeweils Gruppe ohne Anstand

Die `align`-Umgebung

- Flexibler als `equation`- und `eqnarray`-Umgebung
- Mit (`align`) und ohne (`align*`) Formelnummer
- Erlaubt mehrzeilige Formeln, Zeilenumbruch `\\`
- Ordnet tabellarisch an
  - & neue Spalte
  - Spalten abwechselnd links/rechts ausgerichtet
  - Spaltenpaar links/rechts bildet jeweils Gruppe ohne Anstand
- `\tag{text}` ersetzt Formelnummer durch Text `\notag` unterdrückt Ausgabe der Formelnummer
  - falls nur manche Zeilen einer mehrzeiligen Formel Nummer haben sollen

Die `align`-Umgebung

- Flexibler als `equation`- und `eqnarray`-Umgebung
- Mit (`align`) und ohne (`align*`) Formelnummer
- Erlaubt mehrzeilige Formeln, Zeilenumbruch `\\`
- Ordnet tabellarisch an
  - & neue Spalte
  - Spalten abwechselnd links/rechts ausgerichtet
  - Spaltenpaar links/rechts bildet jeweils Gruppe ohne Anstand
- `\tag{text}` ersetzt Formelnummer durch Text `\notag` unterdrückt Ausgabe der Formelnummer
  - falls nur manche Zeilen einer mehrzeiligen Formel Nummer haben sollen
- In Verbindung mit `split`-Umgebung kann man Formelnummern mehrzeiliger Formeln vertikal zentrieren

Die `align`-Umgebung

- Flexibler als `equation`- und `eqnarray`-Umgebung
- Mit (`align`) und ohne (`align*`) Formelnummer
- Erlaubt mehrzeilige Formeln, Zeilenumbruch `\\`
- Ordnet tabellarisch an
  - & neue Spalte
  - Spalten abwechselnd links/rechts ausgerichtet
  - Spaltenpaar links/rechts bildet jeweils Gruppe ohne Anstand
- `\tag{text}` ersetzt Formelnummer durch Text `\notag` unterdrückt Ausgabe der Formelnummer
  - falls nur manche Zeilen einer mehrzeiligen Formel Nummer haben sollen
- In Verbindung mit `split`-Umgebung kann man Formelnummern mehrzeiliger Formeln vertikal zentrieren
- Beispiel 16: `align`-Umgebung

Mathematisch Sätze

- Umgebung für math. Sätze etc. können leicht(!) erstellt werden, d.h. `\newenvironment` hier unnötig!

Mathematisch Sätze

- Umgebung für math. Sätze etc. können leicht(!) erstellt werden, d.h. `\newenvironment` hier unnötig!
- `\newtheorem{name}[counter]{text}[supercounter]`
  - Obligatorisch:
    - `name`: Name der neuen Umgebung
    - `text`: Überschrift der neuen Umgebung, z.B. Satz, Lemma
  - Optional:
    - `counter`: falls kein neuer Zähler angelegt werden soll, sondern vorhandener mitbenutzt wird
    - `supercounter`: spezifiziert übergeordneten Zähler, z.B. `section`: wenn Section erhöht, wird `counter` auf 0 gesetzt
    - gleiche Funktion wie `\numberwithin`

Mathematisch Sätze

- Umgebung für math. Sätze etc. können leicht(!) erstellt werden, d.h. `\newenvironment` hier unnötig!
- `\newtheorem{name}[counter]{text}[supercounter]`
  - Obligatorisch:
    - `name`: Name der neuen Umgebung
    - `text`: Überschrift der neuen Umgebung, z.B. Satz, Lemma
  - Optional:
    - `counter`: falls kein neuer Zähler angelegt werden soll, sondern vorhandener mitbenutzt wird
    - `supercounter`: spezifiziert übergeordneten Zähler, z.B. `section`: wenn Section erhöht, wird `counter` auf 0 gesetzt
    - gleiche Funktion wie `\numberwithin`
- Beispiele:
  - `\newtheorem{satz}{Satz}[section]`
    - Satz-Umgebung
    - Zähler zählt in jeder section neu

Mathematisch Sätze

- Umgebung für math. Sätze etc. können leicht(!) erstellt werden, d.h. `\newenvironment` hier unnötig!
- `\newtheorem{name}[counter]{text}[supercounter]`
  - Obligatorisch:
    - `name`: Name der neuen Umgebung
    - `text`: Überschrift der neuen Umgebung, z.B. Satz, Lemma
  - Optional:
    - `counter`: falls kein neuer Zähler angelegt werden soll, sondern vorhandener mitbenutzt wird
    - `supercounter`: spezifiziert übergeordneten Zähler, z.B. `section`: wenn Section erhöht, wird `counter` auf 0 gesetzt
    - gleiche Funktion wie `\numberwithin`
- Beispiele:
  - `\newtheorem{satz}{Satz}[section]`
    - Satz-Umgebung
    - Zähler zählt in jeder section neu
  - `\newtheorem{lemma}{Satz}[Lemma]`
    - Satz & Lemma werden gemeinsam nummeriert

Mathematisch Sätze

- Umgebung für math. Sätze etc. können leicht(!) erstellt werden, d.h. `\newenvironment` hier unnötig!
- `\newtheorem{name}[counter]{text}[supercounter]`
  - Obligatorisch:
    - `name`: Name der neuen Umgebung
    - `text`: Überschrift der neuen Umgebung, z.B. Satz, Lemma
  - Optional:
    - `counter`: falls kein neuer Zähler angelegt werden soll, sondern vorhandener mitbenutzt wird
    - `supercounter`: spezifiziert übergeordneten Zähler, z.B. `section`: wenn Section erhöht, wird `counter` auf 0 gesetzt
    - gleiche Funktion wie `\numberwithin`
- Beispiele:
  - `\newtheorem{satz}{Satz}[section]`
    - Satz-Umgebung
    - Zähler zählt in jeder section neu
  - `\newtheorem{lemma}{Satz}[Lemma]`
    - Satz & Lemma werden gemeinsam nummeriert
  - `\newtheorem{bemerkung}{Bemerkung}[section]`
    - Bemerkungen werden unabhängig nummeriert
    - Zähler zählt in jeder Section neu

Mathematisch Sätze

- Umgebung für math. Sätze etc. können leicht(!) erstellt werden, d.h. `\newenvironment` hier unnötig!
- `\newtheorem{name}[counter]{text}[supercounter]`
  - Obligatorisch:
    - `name`: Name der neuen Umgebung
    - `text`: Überschrift der neuen Umgebung, z.B. Satz, Lemma
  - Optional:
    - `counter`: falls kein neuer Zähler angelegt werden soll, sondern vorhandener mitbenutzt wird
    - `supercounter`: spezifiziert übergeordneten Zähler, z.B. `section`: wenn Section erhöht, wird `counter` auf 0 gesetzt
    - gleiche Funktion wie `\numberwithin`
- Beispiele:
  - `\newtheorem{satz}{Satz}[section]`
    - Satz-Umgebung
    - Zähler zählt in jeder section neu
  - `\newtheorem{lemma}{Satz}[Lemma]`
    - Satz & Lemma werden gemeinsam nummeriert
  - `\newtheorem{bemerkung}{Bemerkung}[section]`
    - Bemerkungen werden unabhängig nummeriert
    - Zähler zählt in jeder Section neu
- Beispiel 17: newtheorem.tex

Die `tabbing`-Umgebung

- Zur Spaltenweisen Ausrichtung von Text

Die `tabbing`-Umgebung

- Zur Spaltenweisen Ausrichtung von Text
- \= Markierung setzen
- `\kill` Zeile nicht ausgeben
  - für Definitionszeile
- \> Textposition auf nächste Markierung setzen

Die tabbing-Umgebung

- Zur Spaltenweisen Ausrichtung von Text
- \= Markierung setzen
- \kill Zeile nicht ausgeben
  - für Definitionszeile
- \> Textposition auf nächste Markierung setzen

Quelldatei (LaTeX)

```
% tabbing.tex
\documentclass[a4paper,12pt]{report}
\usepackage{fullpage}
\usepackage[ngerman]{babel}
\usepackage[applemac]{inputenc}

\begin{document}
\noindent Jetzt kommt eine \texttt{tabbing}-Umgebung:
\begin{tabbing}
% Definition der Tabulator-Stops
\hspace*{25mm} \= \hspace*{3cm} \= \hspace*{5cm} \= \kill
% Der ausgerichtete Text
Spalte 1 \> Spalte 2 \> Spalte 3 \> Spalte 4\\
Text A   \> Text B   \> Text C             \\
         \> Weiter   \> so        \> !!
\end{tabbing}
\end{document}
```

Ausgabe-Datei (PDF)

Spalte 1	Spalte 2	Spalte 3	Spalte 4
Text A	Text B	Text C	
	Weiter	so	!!

Die `tabular`-Umgebung

- Benutzung wie `array`-Umgebung
  - Anzahl Spalten angeben & Ausrichtung
    - mittig `{c}`, links `{l}`, rechts `{r}`
    - Blocksatz mit fester Spaltenbreite `p{Breite}`
  - vertikale Trennlinie mit Pipe | angeben
    - oder eigene Trennlinie mit
  - Zeilenumbruch mit \\
  - horizontale Trennlinie mit `\hline`
- kann Trennlinien auch in `array`-Umgebung nutzen

Die `tabular`-Umgebung

- Benutzung wie `array`-Umgebung
  - Anzahl Spalten angeben & Ausrichtung
    - mittig `{c}`, links `{l}`, rechts `{r}`
    - Blocksatz mit fester Spaltenbreite `p{Breite}`
  - vertikale Trennlinie mit Pipe | angeben
    - oder eigene Trennlinie mit
  - Zeilenumbruch mit `\\`
  - horizontale Trennlinie mit `\hline`
- kann Trennlinien auch in `array`-Umgebung nutzen
- Verwende `\cline{von-bis}`, falls horizontale Linie nur Spalte `von` bis `bis`

Die `tabular`-Umgebung

- Benutzung wie `array`-Umgebung
  - Anzahl Spalten angeben & Ausrichtung
    - mittig `{c}`, links `{l}`, rechts `{r}`
    - Blocksatz mit fester Spaltenbreite `p{Breite}`
  - vertikale Trennlinie mit Pipe | angeben
    - oder eigene Trennlinie mit
  - Zeilenumbruch mit `\\`
  - horizontale Trennlinie mit `\hline`
- kann Trennlinien auch in `array`-Umgebung nutzen
- Verwende `\cline{von-bis}`, falls horizontale Linie nur Spalte `von` bis `bis`
- Verwende `\multicolumn{anz}{style}{text}` für Eintrag `text` über mehrere Spalten
  - `anz` = Anzahl der betroffenen Spalten
  - `style` = analog zu `tabular`-Style, z.B. `{|c|}`

Die tabular-Umgebung

Quelldatei (LaTeX)

```
% tabular.tex
\documentclass[a4paper,12pt]{report}
\usepackage{fullpage}
\usepackage[ngerman]{babel}
\usepackage[applemac]{inputenc}

\begin{document}
\begin{tabular}{|l| |c| |c| | r|}
\hline
links & mittig & rechts & rechts\\
\hline\hline
1 & 2 & 3 & 4\\
5 & 6 & 7 & 8\\
\hline
\end{tabular}
\end{document}
```

Ausgabe-Datei (PDF)

links	mittig	rechts	rechts
1	2	3	4
5	6	7	8

- Beispiel 18: tabular.tex

Die `table`-Umgebung

- I.d.R. soll Tabelle nicht Teil von Text sein, sondern herausgehoben mit Unterschrift und Nummer
  - verwende `table`-Umgebung

Die `table`-Umgebung

- I.d.R. soll Tabelle nicht Teil von Text sein, sondern herausgehoben mit Unterschrift und Nummer
  - verwende `table`-Umgebung
- `\caption` gibt der Tabelle eine Unterschrift

Die `table`-Umgebung

- I.d.R. soll Tabelle nicht Teil von Text sein, sondern herausgehoben mit Unterschrift und Nummer
  - verwende `table`-Umgebung
- `\caption` gibt der Tabelle eine Unterschrift
- `table`-Umgebung erzeugt eine so genanntes float object
  - wird von LaTeX automatisch platziert
  - wird intern in Liste eingetragen und sobald als möglich gesetzt
    - First-In-First-Out Prinzip
    - `\clearpage` arbeitet Float-Liste ab, danach Seitenumbruch (`\newpage` = neue Seite)

Die `table`-Umgebung

- I.d.R. soll Tabelle nicht Teil von Text sein, sondern herausgehoben mit Unterschrift und Nummer
  - verwende `table`-Umgebung
- `\caption` gibt der Tabelle eine Unterschrift
- `table`-Umgebung erzeugt eine so genanntes float object
  - wird von LaTeX automatisch platziert
  - wird intern in Liste eingetragen und sobald als möglich gesetzt
    - First-In-First-Out Prinzip
    - `\clearpage` arbeitet Float-Liste ab, danach Seitenumbruch (`\newpage` = neue Seite)
- Präferenz für Platzierung kann optional als Liste angegeben werden
  - z.B. `\begin{table}[!thpb]`
    - `!` = egal, ob es vernünftig scheint
    - `t` = top
    - `h` = here
    - `t` = page = Extraseite nur mit floats
    - `b` = bottom
  - wird in der angegebenen Reihenfolge von LaTeX in Erwägung gezogen

Die `table`-Umgebung

- I.d.R. soll Tabelle nicht Teil von Text sein, sondern herausgehoben mit Unterschrift und Nummer
  - verwende `table`-Umgebung
- `\caption` gibt der Tabelle eine Unterschrift
- `table`-Umgebung erzeugt eine so genanntes float object
  - wird von LaTeX automatisch platziert
  - wird intern in Liste eingetragen und sobald als möglich gesetzt
    - First-In-First-Out Prinzip
    - `\clearpage` arbeitet Float-Liste ab, danach Seitenumbruch (`\newpage` = neue Seite)
- Präferenz für Platzierung kann optional als Liste angegeben werden
  - z.B. `\begin{table}[!thpb]`
    - `!` = egal, ob es vernünftig scheint
    - `t` = top
    - `h` = here
    - `t` = page = Extraseite nur mit floats
    - `b` = bottom
  - wird in der angegebenen Reihenfolge von LaTeX in Erwägung gezogen
- erzeugt Tabellen-Verzeichnis
  - Einträge werden aus `\caption{...}` übernommen
    - erstes `latex name.tex` erzeugt `name.lot`
    - zweites `latex name.tex` bindet Verzeichnis ein
  - Falls Unterschrift zu lang ist, Kurztitel festlegen
    - `\caption[kurztitel]{unterschrift}\verb`

Bilder einbinden

- Voraussetzung: `\usepackage[dvips]{graphicx}`
- Einbinden: `\includegraphics[options]{filename}`
  - Optionale Parameter sind:
    - `width = unit`: Breite festlegen (& ggf. skalieren)
    - `height = unit`: Höhe festlegen (& ggf. skalieren)
    - `scale = num`: Bild skalieren
    - `angle = unit`: Bild drehen (math. pos. Grad)

Quelldatei (LaTeX)

```
%includegraphics.tex
\documentclass[a4paper,12pt]{report}
\usepackage{fullpage}
\usepackage[ngerman]{babel}
\usepackage[applemac]{inputenc}
\usepackage[dvips]{graphicx}

\begin{document}
\includegraphics[width=0.75\textwidth]{Grafiken/UniLogo.pdf}

\includegraphics[width=0.75\textwidth,angle=25]{Grafiken/UniLog
\includegraphics[width=3cm, angle=-25]{Grafiken/TeXLogo.pdf}
\end{document}
```

Ausgabe-Datei (PDF)

Die figure-Umgebung

- Verwendung von figure analog zu table
- \listoffigures erzeugt Abbildungsverzeichnis
  - erzeugt Datei name.lof

Quelldatei (LaTeX)

```
%figure.tex
\documentclass[a4paper,12pt]{report}
\usepackage{fullpage}
\usepackage[ngerman]{babel}
\usepackage[applemac]{inputenc}
\usepackage[dvips]{graphicx}

\begin{document}
\listoffigures
\clearpage
\begin{figure}[t]
\begin{center}
\includegraphics[width=0.75\textwidth]{Grafiken/UniLogo.pdf}
\caption{Logo der Universität Ulm}
\label{fig::bsp1}
\end{center}
\end{figure}
\begin{figure}[t]
\begin{center}
\includegraphics[width=0.4\textwidth]{Grafiken/UniLogo.pdf}
\includegraphics[width=0.4\textwidth]{Grafiken/TeXLogo.pdf}
\caption{Links:Logo der Universität Ulm,
rechts: \TeX-\LaTeX-Logo}
\label{fig::bsp2}
\end{center}
\end{figure}
\end{document}
```

Ausgabe-Datei (PDF)

Abbildung: Logo der Universität Ulm

Abbildung: Links: Logo der Universität Ulm, rechts: TEX-LATEX-Logo

Export von Bildern aus Matlab

Matlab

- `print` druckt alle Figure aus
- `print(opt1, opt2, ..., name)` erzeugt File name
  - optionale Strings opt geben an z.B. Auflösung: '-r200' =200dpi (Std. 150dpi) z.B. Dateityp:
    - '-deps' = EPS s/w
    - '-depsc' = EPS farbig
    - '-djpeg90' = JPG, Qualität 90
    - '-dpdf' = PDF

```
% demoprint.m
x = -6:.01:6
y = exp(-x.^2);
z = x.^2/30

plot(x,y, 'b--')
hold on
plot(x,z, 'r')
text(0,1.05, 'exp(0)=1')
hold off

legend('exp(-x^2)', 'x^2/30'))
xlabel('Intervall [-6, 6])
ylabel('Funktionswerte')
title('Ein kleines Beispiel')

print('-r600',-depsc', 'demoprint.eps')
print('-r600',-djpg', 'demoprint.jpg')
```

TEX-LATEX

- Ersetze Matlab-Text durch LATEX-Text: `\usepackage{psfrag}`
- `\psfrag{ps}[posTeX][posPS][scale][angle]{tex}`
  - ersetzt Text `ps` in eps/ps-File durch `tex` in LATEX
  - Bezugspunkte `posTeX` & `posPS` im `ps`- & `tex`-Text
    - horizontal: (ℓ, r, c), vertikal (t, b, c, B) (Std. ℓB)
    - optional: Skalierung und Winkel
  - vor `\includegraphics{filename}`

```
% demoprint.tex
\documentclass[a4paper,12pt]{article}
\usepackage{fullpage}
\usepackage[ngerman]{babel}
\usepackage[applemac]{inputenc}
\usepackage[dvips]{graphicx}

\begin{document}
\begin{figure}[t]
\begin{center}
\includegraphics[width=3cm]{demoprint.eps}
\caption{Wir binden \texttt{demoprint.eps} e
\label{fig::bsp1}
\end{center}
\end{figure}
\end{document}
```

Wissenschaftliches Arbeiten

- In offiziellen mathematischen Dokumenten muss der Autor Quellen angeben
  - im Literaturverzeichnis am Ende
    - vollständige Liste aller verwendeter Hilfen
  - im Fliesstext genaue Angaben
    - woher Ergebnisse, Ideen oder Beweise übernommen wurden
    - ob Teile wörtlich übernommen wurden

- Eigenleistung des Autors muss klar werden
  - z.B. einheitliche Darstellung eines Stoffs aus mehreren Quellen
    - genaue Angabe: Was stammt woher?
  - z.B. zusammenfassende Darstellung eines Stoffs
  - z.B. eigene Beweisidee, aber bekanntes Resultat
  - eigenes Resultat & eigener Beweis

- Im Extremfall: Vorwurf des Plagiats
  - juristisches Nachspiel (z.B. Exmatrikulation!)
  - Aberkennung akademischer Titel

Literatursuche

- http://www.uni-ulm.de/einrichtungen/kiz/bibliothek/literatursuche.html
  - Bibliothekskatalog (Bücher und Zeitschriften der Uni Ulm)
- http://rzblx1.uni-regensburg.de/ezeit/
  - elektronische Zeitschriftenbibliothek mit Links zu Online-Jounals (inkl. Ampel-Darstellung)
- http://www.zentralblatt-math.org/zmath/de
  - bibliographische Datenbank math. Veröffentlichungen
  - eingeschränkter Zugang innerhalb der Uni-Ulm
- http://www.ams.org/mathscient
  - bibliographische Datenbank math. Veröffentlichungen
  - Abkürzungsverzeichnis für Zeitschriften
  - freier Zugang innerhalb der Uni Ulm

Literaturverzeichnis anlegen

- `thebibliography`-Umgebung:
  - startet mit `\begin{thebibliography}{string}`
    - `string` gibt nur maximale Länge von Markern an
  - Einträge mittels `\bibitem[marker]{label}`
    - `label`: definiert Label zum zitieren
    - `marker`: optional, gibt Kennung für Eintrag
    - falls `marker` fehlt, wird Nummer zugewiesen

- Zitieren im Text mittels
  - `\cite[string]{referenz}`
    - `referenz` = `label` vom `\bibitem`
    - `string`: optional, wird zusätzlich ausgegeben, z.B. expliziter Verweis auf einen Satz
  - Listen `\cite{ref1, ref2, ...}` sind erlaubt

Grundsätzliches

- Einträge im Literaturverzeichnis einheitlich!
  - alle Vornamen abkürzen oder ausschreiben
  - gleiches Layout für alle Einträge
  - insb. einheitliche Groß-Kleinschreibung
  - am Ende jedes Eintrags ein Punkt oder nicht
- gewisse Sortierung
  - alphabetisch nach Erstautor (Autorenreihenfolge nicht ändern!)
  - chronologisch nach Veröffentlichungsjahr
  - chronologisch nach Reihenfolge des Zitierens
- korrekte Abkürzungen bei Zeitschriften (http://www.ams.org/mathscient)

Beispiel zum Literaturverzeichnis

```
% literatur.tex
\documentclass[a4paper,12pt]{report}
\usepackage{fullpage}
\usepackage[appelmac]{inputenc}
\usepackage[ngerman]{babel}


\begin{thebibliography}{2}
\addcontentsline{toc}{chapter}{Literaturverzeichnis}
\bibitem{Alberty:1999}{\sc J. Alberty, C. Carstensen, S. A. Funken}:
Remarks around 50 lines of Matlab:
short finite element implementation, Numerical Algorithms 20
(1999), 117-137.

\bibitem{Alt:1980}{\sc H.W. Alt}:
\emph{Lineare Funktionalanalysis},
Springer, Berlin, 1980.

\bibitem{Alt:2003}{\sc H.W. Alt}:
Partielle Differentialgleichungen, Vorlesungsskript WS 2002/03 - SoSe 2003,
Institut f\"ur Angewandte Mathematik, Universität Bonn.

\bibitem{Axelsson:1984}{\sc O. Axelsson, V. A. Barker}:
\emph{Finite Element Solution of Boundary Value Problems},
Academic Press, Orlando, 1984.

\bibitem{Baumgarel:1985}{\sc H. Baumgärtel}:
\emph{Analytic Pertubation Theory for Matrices and Operators},
Birkhäuser, Bosten, 1985.

\end{thebibliography}
```

Weitere Fragen?

Katharina.Becker-Steinberger@uni-ulm.de