

ISO-31-konformer Formelsatz in \LaTeX Version 0.7

Moritz Nadler

Letzte Revision: 7. August 2010

Inhaltsverzeichnis

0 Über diesen Artikel	1
1 Einleitung	2
2 Woher die Regeln kommen	2
3 Die internationalen Regeln für den Formelsatz und ihre Umsetzung mit \LaTeX	2
3.1 Variablen, Parameter und Funktionen	2
3.2 Funktionen mit festem Namen	3
3.3 Einheiten	4
3.4 Konstanten	6
3.5 Indizes	6
3.6 Matrizen und Vektoren	7
3.7 Mathematische Operatoren	8
3.8 Chemische Elemente und Teilchen	9
4 Gründe für die Standardformatierung und die Schwächen von $\LaTeX 2_{\epsilon}$	10
5 Zusammenfassung der Regeln	10
6 Was beim Wechsel der Schriftart zu beachten ist	11
7 Vorgestellte Pakete	12
Literatur	13

0 Über diesen Artikel

Die neueste Version dieses Artikels kann man immer unter moritz-nadler.de/formelsatz.pdf im Internet herunterladen. Die zugehörige \LaTeX -Quelldatei entsprechend unter moritz-nadler.de/formelsatz.tex

nadler.de/formelsatz.tex. Meine E-Mail-Adresse ist moritz_nadler@gmx.de. Ich freue mich über jeden Hinweis auf Fehler – egal wie groß oder klein.

1 Einleitung

Für den Formelsatz gibt es leider keine allgemeingültigen Regeln. So wurden und werden Formeln in verschiedenen Ländern nach jeweils eigener Tradition unterschiedlich gesetzt. \LaTeX setzt Formeln nach der anglo-amerikanischen Tradition. Diese weicht in einigen Punkten von der deutschen Tradition, wie sie zum Beispiel vom Deutschen Institut für Normung (DIN) beschrieben wird, aber auch von internationalen Empfehlungen, wie der Norm ISO 31, ab. In diesem Artikel versuche ich, die internationalen Regeln nach ISO 31 für den Formelsatz vorzustellen und zu erklären, wie man sie in \LaTeX umsetzt. Dabei versuche ich mit so wenig zusätzlichen Paketen wie möglich auszukommen, also möglichst nur standard Latexbefehle zu verwenden. Das ist nicht immer sinnvoll und oft auch nicht möglich. In diesen Fällen erwähne ich zusätzlich die entsprechenden Pakete, gehe aber bis auf ein Beispiel nicht genauer auf sie ein. Dafür sind die jeweiligen Anleitungen zuständig.

Es ist noch wichtig anzumerken, dass die Befehle zur Formatierung von Formeln, die hier vorgestellt werden, zum Teil nur mit der \TeX -Standardschriftart „Computer Modern“ (oder neuere Varianten dieser Schrift wie „Latin Modern“) funktionieren. Welche Probleme mit anderen Schriftarten auftreten können und wie man diese angeht, habe ich in Kapitel 6 zusammengefasst. Diese Kapitel ist auch für Leute interessant, die nie selbst auf andere Schriftarten wechseln würden, da manche beliebte Pakete wie beamer nicht „Computer Modern“ als Voreinstellung benutzen.

2 Woher die Regeln kommen

Zwar gibt es, wie in der Einleitung erwähnt, offizielle Standards, aber diese sind leider meistens nicht frei erhältlich, sondern müssen von den jeweiligen Organisationen gekauft werden. Die offizielle SI-Broschüre [1] gibt es frei im Internet. Ein gute Zusammenfassung der Empfehlungen der ISO und SI Gremien liefert das sogenannte „Green Book“ [2] herausgegeben von der International Union of Pure and Applied Chemistry oder der Artikel „Typesetting mathematics for science technology according to ISO 31 XI“ [3] von Claudio Beccari. Alle Regeln, die ich im folgenden vorstelle, habe ich diesen drei Dokumenten entnommen.

3 Die internationalen Regeln für den Formelsatz und ihre Umsetzung mit \LaTeX

3.1 Variablen, Parameter und Funktionen

Buchstaben, die für Variablen, Parameter oder Funktionen stehen, werden kursiv gesetzt. Weil \LaTeX das im Mathemodus automatisch macht, ist diese Regel leicht zu befolgen. Zwei Beispiele:

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

$$f(x) = x^2$$

Weil man den Malpunkt meistens weglässt, ist es ratsam nur einzelne Buchstaben zu benutzen, um Variablen oder Parameter zu benennen, damit es zu keinen Verwechslungen kommt. In machen Bereichen haben sich allerdings Namen für gängige Parameter etabliert, die aus mehreren Buchstaben bestehen. In der Hydrodynamik sind das zum Beispiel die Reynoldszahl *Re* oder die Mach-Zahl *Ma*. Für diese Parameter gibt es keine Ausnahmen, auch sie müssen kursiv gesetzt werden. Wer besonderen Wert auf gute Typografie legt, kann mit dem Befehl `\mathit{Ma}` zwei-buchstabile Symbole – hier die Mach-Zahl – enger zusammenschreiben, um sie optisch von einem Produkt abzuheben. Vergleiche dazu

$$Ma = F \quad \text{mit}$$

$$Ma = 100.$$

`\mathit` erzeugt eine kursive Formatierung, wie sie in einem Fließtext vorkommt. Die normale Formatierung im Mathemodus ist zwar auch kursiv, allerdings sind die Buchstaben etwas breiter und haben einen größeren Abstand, um deutlich zu machen, dass es einzelne Symbole sind, die multipliziert werden.

Weil es in der amerikanischen Formelsatztradition so üblich ist, setzt \LaTeX griechische Großbuchstaben immer aufrecht. Die ISO-Regeln sehen keine besondere Behandlung von griechischen Buchstaben vor. Das bedeutet, dass man, wenn man sie für Variablen, Parameter oder Funktionen verwendet, eine kursive Formatierung erzwingen muss. Das geht mit dem eben erwähnten Befehl `\mathit`. Als Beispiel Funktionen mit griechischen Buchstaben richtig gesetzt:

$$\Psi(x, t) = \psi(x)\phi(t) \quad \text{Falsch ist dagegen: } \Psi(x, t) = \psi(x)\phi(t)$$

Leider funktioniert `\mathit` angewandt auf griechische Großbuchstaben praktisch nur mit der \LaTeX -Standardschriftart „Computer Modern“. Siehe Kapitel 6 für bessere Lösungen.

3.2 Funktionen mit festem Namen

Funktion, die einen festen Namen tragen, werden aufrecht gesetzt. \LaTeX kennt viele Funktionsnamen bereits, wie zum Beispiel $\sin(x)$, $\exp(x)$ oder $\log(x)$. Um sie zu verwenden, muss man lediglich ihren Namen nach einem `\` schreiben. Zum Beispiel erzeugt `\exp (x)`

$$\exp(x).$$

Kennt \LaTeX einen Funktionsnamen, den man verwenden möchte nicht, kann man ihn mit dem Befehl `\operatorname` von \LaTeX genauso formatieren lassen, wie die eingebauten Namen. Da `\operatorname` ein recht langer Befehl ist kann es günstiger sein `\DeclareMathOperator` zu benutzen, wenn man denselben selbstdefinierten Funktionsnamen öfters verwenden will. Schreibt man zum Beispiel `\DeclareMathOperator{\sgn}{sgn}`

in die Headerdatei hat man einen neuen Befehl nämlich `sgn` zur Verfügung. `\sgn (x)` erzeugt dann:

$$\operatorname{sgn}(x)$$

So hat man sich die Signumfunktion definiert. `\DeclareMathOperator` ist Teil des Pakets `amsmath` und steht daher nur zur Verfügung, wenn man dieses eingebunden hat. Mit dem Ausdruck `\newcommand{sgn}{\operatorname{sgn}}` erreicht man das Gleiche ohne ein zusätzliches Paket, allerdings ist `\DeclareMathOperator{sgn}{sgn}` flexibler. Mehr dazu in Kapitel 3.7.

Die Regel, dass man Funktionen mit festem Namen aufrecht setzt, gilt auch für Funktionen, deren Namen nur aus einem Buchstaben besteht. Bekannte Beispiele sind die Gammafunktion

$$\Gamma(x)$$

und die Diracsche Deltafunktion

$$\delta(x).$$

Nun gibt es mit den griechischen Buchstaben in \LaTeX ein Problem: „Computer Modern“ enthält keine aufrechten griechischen Kleinbuchstaben außer μ . Deshalb macht `\mathrm` angewandt auf griechische Kleinbuchstaben einfach nichts. Um andere kleine aufrechte griechische Buchstaben bekommen zu können, ist ein zusätzliches Paket nötig: `upgreek`. Hat man dieses mit `\usepackage{upgreek}` eingebunden, muss man um einen aufrechten griechischen Buchstaben zu bekommen einfach ein `up` vor den jeweiligen Buchstabenname schreiben. `\updelta (x)` erzeugt also die richtig formatierte Deltafunktion $\delta(x)$. Die Buchstaben aus `upgreek` sind allerdings der Schriftart „Euler“ entnommen worden und passen deshalb nicht ganz zu „Computer Modern“; diese Lösung ist also eher eine Art Hack. Man muss für sich selbst entscheiden, was einem wichtiger ist: Standardkonformität oder eine konsistente Schrift im Dokument. In Kapitel 6 wird ebenfalls auf dieses Problem eingegangen.

3.3 Einheiten

Einheiten und ihre SI-Präfixe schreibt man aufrecht, immer! Hier kann man sich auch nicht mit verschiedenen Traditionen rausreden – die gab es bei Einheiten nie. Zwischen einer Zahl und ihrer Einheit wird ein Leerzeichen gesetzt. Wenn es die benutzte Software erlaubt, sollte man dieses Leerzeichen kleiner als das normale machen. In \LaTeX ist das natürlich möglich. Außerdem sollte man nie zwischen Wert und Einheit brechen. Beispiel:

$$r = 3 \text{ cm}$$

Völliger Unsinn ist:

$$r = 3cm$$

Erreicht wird die aufrechte Formatierung mit dem schon erwähnten `\mathrm`. Das „kleine Leerzeichen“ wird im sowohl im Mathematikmodus als auch im normalen Text durch ein `\,` erzeugt. \LaTeX bricht außerdem nie zwischen Dingen, die durch ein `\,-`Leerzeichen getrennt sind. Das obige Beispiel in \LaTeX -Code:

```
\[
  r = 3 \, \mathrm{cm}
\]
```

Natürlich muss auch das SI-Präfix für 10^{-6} also μ aufrecht gesetzt werden. Das ist mit dem Befehl `\textmu` aus dem Paket `textcomp` möglich. Im Mathemodus muss man `\textmu` noch ein `\text` voranstellen, sonst funktioniert es nicht. Weil `\mathrm` griechische Buchstaben ignoriert, sieht man oft selbst in wissenschaftlichen Zeitschriften Ausdrücke wie

$$x = 52 \mu\text{m}.$$

Richtig ist natürlich

$$x = 52 \mu\text{m}.$$

Das μ aus `textcomp` passt im Gegensatz zu den Buchstaben aus `upgreek` (z. B.: μ) genau zu „Computer Modern“ und passt sich bei einem Wechsel der Schriftart automatisch an, falls die neue Schriftart über ein aufrechtes μ verfügt.

Noch ein weiteres Beispiel, das den Unterschied zwischen den normalen – im Mathemodus mit `\` erzeugtem Leerzeichen – und dem „kleinen Leerzeichen“, dass mit `\,` erzeugt wird, verdeutlicht:

$$\begin{aligned} l &= 17 \text{ km} \\ &= 17 \text{ km} \end{aligned}$$

Eine Sonderstellung hat das Gradzeichen $^\circ$. Wird es als Einheit für Winkel verwendet, steht es direkt nach der Zahl: $\alpha = 360^\circ = 2\pi$. Ist es dagegen Teil eines Einheitennamens wie „Grad Celsius“ oder „Grad Fahrenheit“ steht es direkt an dem Einheitenbuchstaben, und nach der Zahl steht weiterhin ein kleines Leerzeichen: $23^\circ\text{C} = 73.4^\circ\text{F}$. Der hochgestellte kleine Kreis wird durch den L^AT_EX-Code `^\circ` erzeugt. Der Code der zwei Beispiele dieses Absatzes ist `\alpha = 360^\circ = 2\uppi` und `\$23\,\,^\circ\mathrm{C} = 73.4\,\,^\circ\mathrm{F}`.

Anstatt sich das Celsiuszeichen selbst zusammenzubasteln, kann man auch den Befehl `\textcelsius` aus dem Paket `textcomp` nutzen. Allerdings sieht dieses „fertige“ Celsiuszeichen etwas anders aus als das, das mit `^\circ` erzeugt wird; man sollte sie deshalb nicht beide in einem Text verwenden ($^\circ\text{C} \neq \text{C}^\circ$). Außerdem muss man dem `textcelsius`-Symbol innerhalb der Matheumgebung ein `\text` voranstellen, sonst funktioniert es dort nicht.

Wenn man in einem Text viele Werte mit Einheiten benutzt, kann es sinnvoll sein das Paket `siunitx` zu benutzen, das das richtige Formatieren von Zahlen und Einheiten stark vereinfacht. So erzeugt `\$v=\SI{10000.5}{m/s}` $v = 10\,000.5 \text{ m/s}$, wobei zwischen Zahl und Einheit der richtige Abstand gesetzt und die Zahl automatisch in Dreierblöcke eingeteilt wird. Außerdem kann man mit Paketoptionen die Gestalt des Bruches wählen und Punkt gegen Komma austauschen, unabhängig vom in der Tex-Datei benutzen Zeichen. Des Weiteren vereinfacht sich auch die Eingabe von Winkeln: aus `\ang{1;2;3}` wird $1^\circ 2' 3''$.

3.4 Konstanten

Mathematische Konstanten schreibt man aufrecht, naturwissenschaftliche und technische Konstanten schreibt man kursiv.

Die häufigsten mathematischen Konstanten sind natürlich die Zahlen selber; \LaTeX setzt sie automatisch aufrecht. Weitere wichtige mathematische Konstanten sind die imaginäre Einheit $i = \sqrt{-1}$, die Kreiszahl $\pi \approx 3.14$ und die Eulerzahl $e \approx 2.72$. Die Eulersche Formel richtig gesetzt ist daher

$$e^{i\pi} = -1$$

und *nicht*

$$e^{i\pi} = -1.$$

Auch hier benötigt man wieder den Befehl `\mathrm` um im Mathemodus Buchstaben aufrecht zu setzen und den Befehl `\uppi` aus dem Paket `upgreek` für das aufrechte π .

Wenn man oft i und e in Formeln braucht kann es etwas mühselig sein jedes mal ein extra `\mathrm` zu schreiben. Hier bietet es sich an im Header mit dem Befehl `\newcommand{\e}{\mathrm e}` ein Abkürzung zu definieren um jetzt für ein aufrechtes e im Mathemodus nur noch `\e` schreiben zu müssen. Da es den Befehl `\i` schon gibt muss man ihn mit dem Befehl `\renewcommand` überschreiben um das Gleiche für die imaginäre Einheit i zu erreichen. Natürlich darf man sich mit `\renewcommand` nichts überschreiben, von dem man nicht sicher weiß, dass man es nicht benutzt. Im Zweifel sollte man lieber einen neuen Befehl erfinden.

Weil man physikalische Konstanten kursiv schreibt, sind für sie keine eigenen Formatierungsbefehle im Mathemodus nötig. Beispiele wären die Lichtgeschwindigkeit im Vakuum

$$c_0 \approx 3 \times 10^8 \frac{\text{m}}{\text{s}}$$

und die elektrische Feldkonstante im Vakuum

$$\epsilon_0 = \frac{1}{\mu_0 c_0^2} \approx 8.85 \times 10^{-12} \frac{\text{As}}{\text{Vm}}.$$

3.5 Indizes

Oft muss man Größen mit tiefgestellten Indizes versehen, um sie von einander zu unterscheiden. Dabei gilt, dass Indizes, die für Variablen oder physikalische Größen stehen, genau wie diese kursiv gesetzt werden und Indizes, die Abkürzungen oder Bezeichnungen sind, aufrecht gesetzt werden. Beispiele sind das Bohrsche Magneton

$$\mu_B = \frac{e\hbar}{2m_e},$$

die spezifische Wärmekapazität bei konstantem Druck beziehungsweise bei konstantem Volumen

$$c_P > c_V$$

und die relative magnetische Permeabilität

$$\mu_r.$$

Das B in μ_B wird aufrecht geschrieben, weil es eine Abkürzung für den Namen Bohr ist. P und V bei der allgemeinen Beziehung zwischen den Wärmekapazitäten stehen für die physikalischen Größen Druck und Volumen, also werden sie kursiv geschrieben. Das r in μ_r schreibt man aufrecht, weil es eine Abkürzung für relativ ist. Eine Anmerkung noch zur Definition des Bohrschen Magnetons: Dort taucht e einmal als die physikalische Konstante „Elementarladung“ auf und wird als solche natürlich kursiv geschrieben und einmal als Index an der Masse m_e und wird dort aufrecht gesetzt, weil es die Abkürzung für das Wort Elektron ist.

Noch zwei Beispiele aus der Mathematik:

$$a_n = \frac{1}{n^2}$$
$$x_{\max} = 7$$

In der Folge a_n schreibt man das n kursiv, weil es ein Laufindex, also eine Variable ist. Bei x_{\max} ist max eine Abkürzung für maximal oder Maximum und wird wie auch schon das B bei μ_B aufrecht gesetzt.

Wie schon in den vorherigen Abschnitten wird die aufrechte Formatierung durch den Befehl `\mathrm` erzeugt.

3.6 Matrizen und Vektoren

Um Matrizen und Vektoren zu kennzeichnen gibt es keine festen Regeln sondern nur Empfehlungen. Für echte Mathematiker ist es nicht nötig, diese extra auszuzeichnen, da es sich ja nur um Variablen wie andere auch handelt; sie gehören einfach nur einer anderen Menge als zum Beispiel \mathbb{R} an. Und da ein Mathematiker sowieso bei jeder neuen Variable dazuschreiben sollte, welcher Menge sie angehört, bleibt alles eindeutig. Die Eigenwertgleichung hat dann die Gestalt:

$$Av_n = \lambda_n v_n \quad \text{mit } A \in \mathbb{C}^{N \times N}, \quad v \in \mathbb{C}^N, \quad \lambda \in \mathbb{C}.$$

Gerade in der Physik ist es aber oft sehr hilfreich, Matrizen und Vektoren klar von skalaren Größen zu unterscheiden. In Deutschland ist es üblich, Vektoren durch einen Pfeil über dem Buchstaben zu markieren, wie bei \vec{v} . Das wird durch den Befehl `\vec` erreicht. Für Matrizen ist es üblich, Großbuchstaben zu verwenden.

Die Empfehlung der internationalen Gremien ist, für Vektoren fette und kursive Kleinbuchstaben und für Matrizen fette und kursive Großbuchstaben zu verwenden. Die Eigenwertgleichung schreibt sich dann als:

$$\mathbf{A}\mathbf{v}_n = \lambda_n \mathbf{v}_n$$

Für die Formatierung fett und kursiv im Mathemodus braucht man ein zusätzliches Paket `bm` (steht für bold math). Nach dessen Einbindung steht dann der Befehl `\bm` zur

Verfügung, der jedes beliebige Zeichen im Mathemodus fett macht, ohne sonst etwas zu ändern. Das obige Beispiel wurde durch den Code

```
\[
\bm{Av}_n = \lambda_n \bm{v}_n
\]
```

erzeugt. Man kann sich mit `\bm` aber auch alles andere fett machen lassen zum Beispiel das Unendlichzeichen $\infty \neq \infty$.

Es gibt auch den Befehl `\pmb`, der einen ähnlichen Zweck hat wie `\bm`. Er sollte aber nicht verwendet werden, da er älter ist und deutlich schlechter aussehende Formatierungen liefert.

Der \LaTeX Befehl für fette Schrift im Mathemodus `\mathbf` erzeugt immer aufrechte Buchstaben auch wenn er mit `\mathit` kombiniert wird und ist daher ungeeignet. Die Verwendung von `\mathbf` führt zu:

$$\mathbf{Av}_n = \lambda_n \mathbf{v}_n$$

Diese Darstellung von Vektoren findet man auch oft in Büchern und Papers. Sie sollte aber vermieden werden, da man so völlig grundlos die Grundregel „Variablen schreibt man kursiv“ missachtet. Vektoren sind auch nur Variablen, und egal wie man sie hervorhebt, sollen sie auf jeden Fall kursiv bleiben wie die skalaren Variablen auch.

3.7 Mathematische Operatoren

Mathematische Operatoren schreibt man aufrecht. Wenn man so will, sind Operatoren ja auch nur Funktionen, also Zuordnungsvorschriften mit festem Namen oder einem festgelegten Symbol. Alle Befehle, die man dazu braucht, wurden schon in den vorherigen Abschnitten, vor allem in Kapitel 3.2 vorgestellt. Trotzdem noch einmal zur Wiederholung:

Kennt \LaTeX den Operatornamen muss man nur diesen nach einem `\` eingeben wie zum Beispiel bei Limes `\lim`

$$\lim_{x \rightarrow \infty} f(x) = 0,$$

Nabla `\nabla`

$$\mathbf{F} = -\nabla E_{\text{pot}}$$

oder Determinante `\det`

$$\det \mathbf{A} = 4.$$

Kennt \LaTeX einen Operator, der aus mehreren Buchstaben besteht, nicht, so benutzt man den Befehl `\operatorname` oder `\DeclareMathOperator` wie zum Beispiel bei Gradient

$$\mathbf{F} = -\operatorname{grad} E_{\text{pot}}$$

oder bei den Operatoren Imaginärteil und Realteil

$$z = x + iy \iff \operatorname{Re} z = x \wedge \operatorname{Im} z = y.$$

`\DeclareMathOperator` hat gegenüber `\operatorname` den Vorteil, dass man durch einen `*` bei Superskripts ein Verhalten wie beim Limes erzwingen kann. Nach einem `\DeclareMathOperator*{\meinlim}{meinlim}` erzeugt `\meinlim_{ x \rightarrow \infty}`

$$\operatorname{meinlim}_{x \rightarrow \infty}.$$

Besteht der Operator nur aus einem Buchstaben so ist es meistens besser `\mathrm` zu benutzen wie beim Transponieroperator für Matrizen

$$\mathbf{B} = \mathbf{A}^T$$

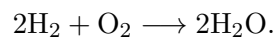
oder dem Differenzialoperator

$$\int x^2 dx = \frac{1}{3}x^3$$

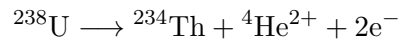
$$\frac{d^n}{dx^n} f(x).$$

3.8 Chemische Elemente und Teilchen

Die Zeichen für chemische Elemente oder Teilchen werden immer aufrecht gesetzt, egal ob im Text oder in Reaktionsgleichungen. Wie in den vorhergehenden Kapiteln geht das mit `\mathrm`. Ein Beispiel ist die Knallgasreaktion:



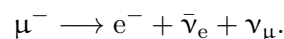
Will man das genaue Isotop eines Elements benennen braucht man hoch- und tiefgestellte Zahlen auch vor einem Buchstaben, dazu muss man ein `{}` vor das entsprechende Symbol schreiben und daran die hoch- und tiefgestellten Zahlen anhängen. Das nächste Beispiel mit dem α -Zerfall von $^{238}\mathrm{U}$



sieht als Code so aus:

```
\[
{}^{238} \mathrm{U}
\longrightarrow {}^{234} \mathrm{Th} + {}^4 \mathrm{He}^{2+} + 2\mathrm{e}^-
\]
```

Natürlich schreibt man auch die Teilchen, die mit griechischen Buchstaben benannt sind, aufrecht. Hierbei ist das schon oft genannte Paket `upgreek` sehr hilfreich. Beispiele sind das neutrale Pion π^0 oder das Photon γ . Nach dem genannten sollte es auch klar sein, dass es die α -Strahlung, die β -Strahlung und die γ -Strahlung heißt, und **nicht** wie man oft liebt die α -, β - oder γ -Strahlung. Noch ein Beispiel aus der Teilchenphysik. Ein möglicher Zerfallskanal des Myons ist



Wenn man wie bei Neutrinos die Teilchen von den Antiteilchen nicht durch ihre Ladung unterscheiden kann (weil sie keine haben), ist es üblich einen Querstrich über dem Teilchenzeichen zu machen. Der Strich wird in \LaTeX durch den Befehl `\bar` erzeugt.

Spätestens nach drei längeren chemischen Formeln werden die hier beschriebenen Formatierungen sehr mühselig und kosten viel Tipparbeit. Das Paket `mhchem` erleichtert das Setzen von chemischen Formeln enorm. Es stellt den Befehl `\ce` zur Verfügung mit dem sich das Setzen der Knallgasreaktion auf `\ce{2 H2 + O2 -> 2 H2O}` vereinfacht, was $2\text{H}_2 + \text{O}_2 \rightarrow 2\text{H}_2\text{O}$ erzeugt, und das Setzen des Zerfalls von Uran auf `\ce{^{238}U -> ^{234}Th + ^4He^{2+} + 2 e-}` vereinfacht, was $^{238}\text{U} \rightarrow ^{234}\text{Th} + ^4\text{He}^{2+} + 2\text{e}^-$ erzeugt.

4 Gründe für die Standardformatierung und die Schwächen von $\LaTeX 2_\epsilon$

Es gibt vor allem zwei Aspekte, die ISO-konformes Formelsetzen in \LaTeX schwieriger machen, als es sein könnte. Zum einen folgt Standardformatierung in \LaTeX wie schon erwähnt der angelsächsischen Tradition des Formelsatzes und nicht den internationalen Normen. Ein Beispiel dafür ist die nach den ISO-Regeln „falsche“ Standardformatierung griechischer Großbuchstaben: aufrecht anstatt kursiv. Zum anderen ist \TeX sehr alt, und es wurden am Anfang seiner Entwicklung einige Kompromisse an die geringen Kapazitäten der Computer der frühen 80er gemacht. Und weil \TeX gar nicht mehr und \LaTeX nur sehr langsam weiterentwickelt wird, wirken sich diese heute noch aus. So lassen sich die Formatierungsbefehle für den Mathemodus nicht kombinieren, und es sind Zusatzpakete wie `bm` nötig, um bestimmte Formatierungen zu ermöglichen. Auch sind manche dieser Befehle nicht allgemein verwendbar; so erzeugt `\mathit` angewandt auf griechische Großbuchstaben Unsinn, wenn nicht „Computer Modern“ verwendet wird.

Die größte Schwäche von \TeX in Bezug auf standardkonformen Formelsatz ist sicher das Fehlen von aufrechten griechischen Kleinbuchstaben in „Computer Modern“ (bis auf μ). Knuth sah wahrscheinlich auf Grund der angelsächsischen Tradition, griechische Kleinbuchstaben unabhängig von ihrer Verwendung immer kursiv zu setzen, keine Notwendigkeit dafür.

5 Zusammenfassung der Regeln

In Formeln setzt man *kursiv*:

- Variablen, z. B.: x
- Funktionen, z. B.: $f(x)$
- Physikalische Konstanten, z. B.: c_0
- Indizes, die Variablen oder physikalische Größen sind, z. B.: $a_{i,j}$ oder c_V

Und **aufrecht** setzt man:

- Funktionen mit festem Namen, z. B.: $\sin(x)$ oder $\Gamma(x)$
- Mathematische Konstanten, z. B.: π , i oder e
- Einheiten und deren Vorsätze, z. B.: $\lambda = 0.56 \mu\text{m}$
- Indizes, die für Namen oder Bezeichner stehen, z. B.: x_{max} oder μ_B
- Chemische Elemente und Teilchen, z. B.: H_2O
- Mathematische Operatoren, z. B.: $A^T = B$

6 Was beim Wechsel der Schriftart zu beachten ist

Es gibt vor allem drei Dinge, die bei einem Wechsel der Schriftart oder der Dokumentklasse zu ungewollten Ergebnissen führen können:

1. Der in diesem Artikel häufig genutzte Befehl `\mathrm`
2. Der Befehl `\mathit`, wenn er auf griechische Großbuchstaben angewandt wird
3. Die griechischen Buchstaben aus dem Paket `upgreek`

Das Problem mit `\mathrm` ist, dass er, wie jeder Befehl, von einem Paketautor undefiniert werden kann und man sich somit nicht darauf verlassen kann, ob er die gewünschte Formatierung erzeugt. Bei vielen Paketen schaltet `\mathrm` auf Großbuchstaben mit Serifen um, auch wenn die eigentlich gewählte Schriftart serifenlos ist, so wie der Befehl `\mathsf` meistens einen serifenlosen Buchstaben erzeugt, auch wenn die Matheschriftart Serifen hat. Eine Lösung dieses Problems ist sich ein kleines Makro zu definieren, das man zum Beispiel `\mathup` nennen könnte und das dann als Synonym für `\mathrm`, `\mathsf` oder `\textup` dient, je nach dem was bei der vorhandenen Kombination aus Schriftart und Dokumentklasse das gewünschte Ergebnis liefert.

`\textup` sollte immer funktionieren, wenn die Schriftart für den Mathemodus und den normalen Text identisch sind, was ja sowieso besser ist. Wenn man aus irgendeinem Grund doch verschiedene Schriftarten im normalen Text und im Mathemodus hat, sollte `\mathrm` bei Matheschriften mit Serifen im Normalfall richtig sein und `\mathsf` bei welchen ohne Serifen.

Manchmal liest man auch, dass man aufrechte Formatierung im Mathemodus mit dem Befehl `\text` (gehört zu `amsmath`) erzeugen soll. `\text` ist aber keine gute Wahl, weil es innerhalb des Mathemodus eine Formatierung erzeugt, die vom umgebenden Text benutzt wird. Das bedeutet, dass Buchstaben in einer Formel, die mit `\text` ausgezeichnet sind, sich automatisch mitverändern, wenn der umgebende Absatz auf kursiv umgestellt wird.

Punkt 2 ist dagegen sehr leicht zu umgehen: Wenn man kursive griechische Großbuchstaben möchte, sollte man immer Befehle wie zum Beispiel `\varGamma` aus dem Paket `amsmath` anstatt `\mathit \Gamma` benutzen, weil die Makros aus `amsmath` eigentlich bei jeder Schriftart, die kursive griechische Großbuchstaben hat, funktionieren; `\mathit` funktioniert dagegen nur bei „Computer Modern“.

Punkt 3 ist eigentlich kein Problem, das speziell beim Wechsel der Schriftart auftritt, sondern es besteht immer. Die Buchstaben, die `upgreek` liefert, stammen aus der Schriftart „Euler“, und je stärker sich die Schriftart des Dokuments von Euler unterscheidet, desto „schlimmer“ stechen die griechischen Buchstaben aus `upgreek` heraus. Man muss sich also überlegen, was einem wichtiger ist: Standardkonformität oder konsistentes Schriftbild.

Die „richtige“ Lösung des Problems wäre es, eine Schriftart zu benutzen, die einen *vollständigen* Satz an aufrechten und kursiven griechischen Buchstaben mitbringt. Leider gibt nur sehr wenige kostenlose Schriftarten für \LaTeX , die das tun – die \LaTeX -Standard-schriftart „Computer Modern“ hat auch keine aufrechten griechischen Kleinbuchstaben. Ein positives Beispiel wäre „Minion Pro“. Die Installation dieser Schriftart ist allerdings sehr aufwendig und sie muss bei Zeichen wie \mathbb{R} oder \mathbb{N} auf „Computer Modern“ zurückgreifen. Das heißt, dass man wieder eine Mischung von Schriftarten nur bei anderen Symbolen hat.

7 Vorgestellte Pakete

In diesem Abschnitt werden nochmal alle im Artikel erwähnte Pakete genannt und beschrieben:

- `upgreek` [Information auf CTAN](#)
liefert aufrechte griechische Buchstaben, wie man sie für die Kreiszahl π oder Teilchennamen braucht. Die Buchstaben aus `upgreek` sind der Schriftart „Euler“ entnommen. Man mischt also immer zwei Schriftarten, wenn man `upgreek` benutzt. Und je stärker der Unterschied zwischen der gewählten Schrift und Euler ist, desto unpassender wirken die `upgreek` Buchstaben im Dokument. Daher ist die Verwendung von `upgreek` eher ein „Hack“ oder „Workaround“, aber nicht eine wirklich gute Lösung.
- `bm` [Information auf CTAN](#)
ermöglicht die fette Hervorhebung von Variablen und Symbolen im Mathemodus, ohne die standardkonforme kursive Formatierung aufzugeben – was `\mathbf` nicht kann. Es ist besser als `\pmb` oder `\boldsymbol`.
- `amsmath` [Information auf CTAN](#)
wird so häufig eingebunden, dass viele gar nicht mehr wissen, ob ein Befehl reines \LaTeX ist oder aus diesem Paket stammt. Aus diesem sehr umfangreichen Paket wurde hier `\DeclareMathOperator{}{}` als flexiblere Alternative zu `\newcommand{}{\operatorname{}}` verwendet; außerdem wurde auf die Makros für kursive griechische Großbuchstaben aufmerksam gemacht. Diese liefern nach einem Wechsel der Mathematikschriftart immernoch die richtigen Buchstaben, wogegen `\mathit` Γ dann zum Beispiel nicht mehr funktioniert. Auch der Befehl `\text`, der Buchstaben in Formeln wie den umgebenden Text formatiert, ist Teil dieses Pakets.

- `textcomp` [Information auf CTAN](#)
liefert zusätzliche Symbole wie °C oder μ passend zur gewählten Schriftart. Daher ist das μ aus diesem Paket im Allgemeinen dem μ aus `upgreek` vorzuziehen. Will man die Symbole aus diesem Paket im Mathemodus nutzen, muss man ihnen ein `\text` voranstellen.
- `siunitx` [Information auf CTAN](#)
vereinfacht das setzen von Zahlen, Winkeln und Einheiten. Wenn vorher `textcomp` geladen wurde, wird automatisch das „fertige“ Celsius-Symbol und das bessere μ benutzt.
- `mhchem` [Information auf CTAN](#)
vereinfacht stark das aufwendige Setzen von Summenformeln, Reaktionsgleichungen und Isotopen.

Literatur

- [1] Bureau international des poids et mesures (BIPM) „SI brochure (8th edition)“
http://www.bipm.org/en/si/si_brochure
- [2] International Union of Pure and Applied Chemistry (IUPAC) (Herausgeber) „Quantities, Units and Symbols in Physical Chemistry 3rd Edition“ 2007
ISBN-13 978 0 85404 433 7
Die vollständige zweite Ausgabe ist frei unter
http://www.iupac.org/publications/books/gbook/green_book_2ed.pdf erhältlich.
- [3] Claudio Beccari „Typesetting mathematics for science technology according to ISO 31 XI“ TUGboat, Volume 18 1997, No. 1
<http://www.tug.org/TUGboat/Articles/tb18-1/tb54becc.pdf>