TEX, LATEX and math

Enrico Gregorio

Abstract

We discuss some aspects of mathematical typesetting: choice of symbols, code abstraction, fine details. Relationships between math typesetting and international standards are examined. A final section on typesetting of numbers and units reports on some recent developments in the field.

1 Introduction

Readers may well know that TEX was born out of Knuth's discomfort after having seen the proofs of the new edition of the first volume of his *magnum opus* "The Art of Computer Programming".

Many papers have been written by Knuth himself and by others on the topic of math typesetting. Here I'd like to present some personal ideas on the subject, coming from almost thirty years of experience in mathematical typesetting. I'll also present some recent developments and new tricks made available with expl3.

2 A very short lead-in to math in TEX

Every TEX guru knows that TEX is (almost) always in one of three *modes*:

- horizontal mode,
- vertical mode,
- math mode.

Except for circumstances when TEX is in no mode at all (when writing to external files, for the curious).

Each mode comes into two flavors, but here we are interested only in math mode. Knuth calls the two flavors 'math mode' and 'display math mode'. In order to better distinguish between them, I'll call the former 'inline math mode', so the unadorned 'math mode' will denote both.

There are subtle, well, not so much so, differences between the two flavors; beginners are most impressed by $\sum_{k=1}^{n} k^2 = \frac{1}{3}n\left(n+\frac{1}{2}\right)(n+1)$ that suddenly becomes

$$\sum_{k=1}^{n} k^2 = \frac{1}{3}n\left(n+\frac{1}{2}\right)(n+1)$$

when displayed and a very common question is 'how do I get the limits above and below the summation symbol and real fractions, not that smallish replacement symbol?'

Like all of us, I've been a beginner myself; I discovered `\limits` and abused it. *Penitenziagite*, would have said Salvatore in "Il nome della rosa" (Umberto Eco's novel; the English title is "The name of the Rose"). Now I'm no longer a beginner and *know* why `\limits` should not be used; and let's not talk about the dreaded `\displaystyle` that is sometimes suggested to newbies. The proper way is just `\sum`.

To the contrary, beginners are usually much less impressed by the wrong typesetting in

$$A\backslash B = \{x|x \in A, x \notin B\}$$

but they are likely to shrug and move on, if they ever note it. Sometimes they see something's wrong and 'fix' the vertical bar by using `\,|\,`, that's still wrong. Why is it wrong? The spacing is too small, of course, but there's more to the problem: two appearances of such a construction in the same document is a sin similar to what I describe to young basketball referees: "whoever calls a double foul during their career has called one too many". The correct answer is: first of all define a macro for the object, for instance,

```
\newcommand{\suchthat}{\,|\,}
```

(I'm talking LATEX, plain TEX users can translate). In case one asks, if `a+b` appears twice or more in a document there's no need to make a macro out of it; the separator in the set builder notation is a single conceptual object and so it *must* be typed by a single command.

About the spacing, one should realize that the reverse bar is a binary operation symbol and the vertical bar is a relation symbol. Both are already defined in all flavors of TEX and they are, respectively, `\setminus` and `\mid`, but it's still convenient and logically sound to define `\suchthat`, because `\mid` is a 'generic' name:

```
\newcommand{\suchthat}{\mid}
...
A \setminus B=\{x \suchthat
  x\in A, x\notin B\}
```

will typeset as

$$A \setminus B = \{x \mid x \in A, x \notin B\}$$

while this is the version with the thin spaces:

$$A \setminus B = \{x \,|\, x \in A, x \notin B\}$$

Compare closely the spaces around the vertical bar.

I'm not saying the last realization should be rejected as awfully wrong: personal judgment is always welcome when typography is concerned, after having studied the alternatives and common practice. Above all, consistency throughout a document is a must. I had to edit a paper where the separator was a bar or a colon or a semicolon, depending on which

First published in *ArsTEXnica* 28 (Oct 2019), pp. 47–57. Reprinted with permission.

of the three authors had typed the formula. Defining `\suchthat` allows for delaying any decision about what symbol to use until the last minute. More on set builder notation later.

The TEXbook lists several symbol names, some have semantics attached to them, like `\setminus`, and others don't, like `\mid` or `\otimes`.[1] Why is that? Some symbols have essentially a single use case, others appear in different branches of mathematics with different meanings. Everybody loves `\lhd` and `\unlhd`, right? The symbols typeset as $\lhd$ and $\unlhd$ respectively. I believe to have seen once what the names should suggest, but I forgot it. The symbols are common in group theory, where they denote 'normal subgroup': it's heartily recommended to group theorists to define a meaningful command for them. Oh, I was almost forgetting! Those are not defined as relation symbols in LaTeX, so a savvy group theorist will type in the document preamble

```
% normal subgroup
\newcommand{\ns}{\mathrel\lhd}
\newcommand{\nseq}{\mathrel\unlhd}
% subnormal subgroup
\newcommand{\sns}{\ns\ns}
```

The symbols are not among the core ones designed by Knuth. They first appeared in a symbol font distributed along with LaTeX; possibly Lamport used them for his own papers as binary operators and the classification stuck. They were later included in `amssymb` (`\vartriangleleft`, which is a relation).

What should an author do? The case of normal subgroups is clear: I surely wouldn't litter my paper with `\mathrel\lhd` each time I want to mention normal subgroups. However, suppose a paper frequently uses Euler's *totient function*, which has the well established tradition of being denoted by φ (the open version of phi). Is it better to use `\varphi` or to define `\euphi`? The latter. Imagine that upon receiving proofs, the author realizes that *all* instances of `\varphi` print out ϕ, because the publisher uses a font that lacks the proper symbol. With `\euphi` it is a matter of doing a redefinition, probably borrowing the open phi from another font. We don't know when the instruction `\let\varphi=\phi` is performed, but using `\euphi` makes this irrelevant.

An important exception: in the abstract there should be *no* use of personal macros. It should be able to typeset with a 'naked' version of LaTeX: it's very common nowadays that the abstract is fed to some web page that maybe uses MathML, MathJax or similar device for handing the text to browsers.

[1] Generally LaTeX kept the same names.

Going back to the normal subgroup symbol, one should know that every math symbol belongs to a class and there are seven of them:

- class 0, ordinary symbols;
- class 1, operators;
- class 2, binary operations;
- class 3, binary relations;
- class 4, opening symbols;
- class 5, closing symbols;
- class 6, punctuation.

TeX will set the spacing between symbols according to well defined rules. This is not the place to discuss them fully, see [4]. Any object, as long as it is legal in math mode, can be defined to behave as if it belongs in one of the above classes by typing it as the argument to

```
\mathord \mathop \mathbin \mathrel
\mathopen \mathclose \mathpunct
```

For instance, the symbol for the determinant is internally carried out (in LaTeX) by something like

```
\mathop{\operator@font det}\nolimits
```

but there is a higher level interface available for declaring new symbols like this; for instance, one does

```
\DeclareMathOperator{\adj}{adj}
```

in order to introduce a symbol for the adjugate matrix. A one-shot operator can be typeset in the document by

```
\operatorname{adj}
```

The *-version of both commands makes for a symbol that carries limits above and below in display math mode, on the side when inline.

The unfortunately common perversion of denoting open intervals like $]a,b[$ needs input such as

```
\mathopen]a,b\mathclose[
```

One can easily spot that something is wrong when just using `]a,b[` by looking at the difference between the two instances below

$$x \in]a,b[\qquad x \in]a,b[$$

In my calculus notes I type `\interval[o]{a,b}`, so I can decide to be a perv by just changing a few lines in the definition. An open interval will be typeset as $(a \mathinner{..} b)$, but I'm not bound in any way: I can go back to the comma again by just changing a line. Also, I like to write upper unbounded intervals like $(a \mathinner{..} \to)$, but I use `\pinf` for the arrow, so I can make it to be typeset ∞ by acting on a single line, should I change my mind.

Upon entering math mode, TeX will construct a *math list* consisting of *math atoms*, each of which

$$
\begin{cases} a^6 + 2a^3b^3 + b^6 = q^2 \\ 4a^3b^3 = -\frac{4}{27}\,p^3 \end{cases}
\begin{cases} a^3 + b^3 = -q \\ a^6 - 2a^3b^3 + b^6 = q^2 + \frac{4}{27}\,p^3 \end{cases}
\begin{cases} a^3 + b^3 = -q \\ (a^3 - b^3)^2 = q^2 + \frac{4}{27}\,p^3 \end{cases}
\begin{cases} a^3 + b^3 = -q \\ a^3 - b^3 = \sqrt{q^2 + \frac{4}{27}\,p^3} \end{cases}
$$

$$
\left\{\begin{gathered} a^6 + 2a^3b^3 + b^6 = q^2 \\ 4a^3b^3 = -\frac{4}{27}\,p^3 \end{gathered}\right.
\left\{\begin{gathered} a^3 + b^3 = -q \\ a^6 - 2a^3b^3 + b^6 = q^2 + \frac{4}{27}\,p^3 \end{gathered}\right.
\left\{\begin{gathered} a^3 + b^3 = -q \\ (a^3 - b^3)^2 = q^2 + \frac{4}{27}\,p^3 \end{gathered}\right.
\begin{cases} a^3 + b^3 = -q \\ a^3 - b^3 = \sqrt{q^2 + \frac{4}{27}\,p^3} \end{cases}
$$

$$
\left\{\begin{aligned} a^6 + 2a^3b^3 + b^6 &= q^2 \\ 4a^3b^3 &= -\frac{4}{27}\,p^3 \end{aligned}\right.
\left\{\begin{aligned} a^3 + b^3 &= -q \\ a^6 - 2a^3b^3 + b^6 &= q^2 + \frac{4}{27}\,p^3 \end{aligned}\right.
\left\{\begin{aligned} a^3 + b^3 &= -q \\ (a^3 - b^3)^2 &= q^2 + \frac{4}{27}\,p^3 \end{aligned}\right.
\left\{\begin{aligned} a^3 + b^3 &= -q \\ a^3 - b^3 &= \sqrt{q^2 + \frac{4}{27}\,p^3} \end{aligned}\right.
$$

Figure 1: Three ways of laying out the derivation of Cardano's formula

$$
\left\{\begin{aligned} a^6 + 2a^3b^3 + b^6 &= q^2 \\ 4a^3b^3 &= -\frac{4}{27}\,p^3 \end{aligned}\right.
\left\{\begin{aligned} a^3 + b^3 &= -q \\ a^6 - 2a^3b^3 + b^6 &= q^2 + \frac{4}{27}\,p^3 \end{aligned}\right.
\left\{\begin{aligned} a^3 + b^3 &= -q \\ (a^3 - b^3)^2 &= q^2 + \frac{4}{27}\,p^3 \end{aligned}\right.
\left\{\begin{aligned} a^3 + b^3 &= -q \\ a^3 - b^3 &= \sqrt{q^2 + \frac{4}{27}\,p^3} \end{aligned}\right.
$$

Figure 2: One of the worst alignments I can conceive

has a *nucleus*, a *subscript field* and a *superscript* field. When exiting from math mode, the math list will be transformed into a horizontal list according to the (complex) rules described in Appendix G of *The TEXbook*. These rules add spaces, as said before, but also take care of the bidimensionality of math formulas: superscripts, subscripts, fractions, accents, radicals, extensible delimiters and many more aspects.

Had Knuth been into theoretical physics, he probably would have added also "prescripts" for isotopes and staggered multiple subscripts and superscripts for tensors. Unfortunately he hasn't. See later for more on this topic.

3 Fine points of mathematics typing

The title is the same as chapter 18 in *The TEXbook*. Of course I won't go through Knuth's words. Since I'm talking LATEX and math, I assume that amsmath is loaded: no serious math typesetting can be done without it.

A point that's not touched upon in the TEXbook is 'when, really, consecutive equations should be aligned and where'. Browsing TEX.StackExchange reveals several examples of bad alignments.

A prominent example is a derivation of Cardano's formula[2] which I won't give the code for, but just three realizations that you can see in figure 1.

I often use the style "the good, the bad, and the ugly". There is actually an even uglier way, which is what the questioner was asking for, see figure 2.

What's the problem? The equals signs are not really related to one another. The *pairs of formulas* are related, but the fact that they are equations is almost irrelevant. Mixing ragged right and ragged left in one and the same paragraph (or display) makes for very hard reading. I'd instead be more generous with vertical spacing between the various braces and I have no doubt whatsoever that the leftmost realization is our Clint Eastwood. Look for holes in the typeset output and remove them.

Another example can be seen in figure 3.[3] You can judge for yourself which is the best way to present the display. My opinion is that the equals signs in the second column pair are not related to each other, so they're not to be aligned.

Linear systems are an exception, because their matrix-like structure is more important than holes. I recommend the wonderful systeme package by Christian Tellechea [11]. No doubt there are other exceptions: typography, and mathematical typography in particular, is a craft that doesn't obey mechanical rules. A thin space may open up symbols and make them easier to read, adding a pair of parentheses may clear up an ambiguity, removing unnecessary

[2] https://tex.stackexchange.com/questions/193581
[3] https://tex.stackexchange.com/questions/500472

$$\begin{pmatrix} A_\mu \\ \rho^*_\mu \end{pmatrix} \to \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} A_\mu \\ \rho^*_\mu \end{pmatrix}, \qquad \tan\theta = \frac{g_{el}}{g_*} \tag{1}$$

$$\begin{pmatrix} \psi_L \\ \chi_L \end{pmatrix} \to \begin{pmatrix} \cos\varphi_{\psi_L} & -\sin\varphi_{\psi_L} \\ \sin\varphi_{\psi_L} & \cos\varphi_{\psi_L} \end{pmatrix} \begin{pmatrix} \psi_L \\ \chi_L \end{pmatrix}, \qquad \tan\varphi_{\psi_L} = \frac{\Delta}{m} \tag{2}$$

$$\begin{pmatrix} \tilde\psi_R \\ \tilde\chi_R \end{pmatrix} \to \begin{pmatrix} \cos\varphi_{\tilde\psi_R} & -\sin\varphi_{\tilde\psi_R} \\ \sin\varphi_{\tilde\psi_R} & \cos\varphi_{\tilde\psi_R} \end{pmatrix} \begin{pmatrix} \tilde\psi_R \\ \tilde\chi_R \end{pmatrix}, \qquad \tan\varphi_{\tilde\psi_R} = \frac{\tilde\Delta}{\tilde m} \tag{3}$$

$$\begin{pmatrix} A_\mu \\ \rho^*_\mu \end{pmatrix} \to \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} A_\mu \\ \rho^*_\mu \end{pmatrix}, \qquad \tan\theta = \frac{g_{el}}{g_*} \tag{4}$$

$$\begin{pmatrix} \psi_L \\ \chi_L \end{pmatrix} \to \begin{pmatrix} \cos\varphi_{\psi_L} & -\sin\varphi_{\psi_L} \\ \sin\varphi_{\psi_L} & \cos\varphi_{\psi_L} \end{pmatrix} \begin{pmatrix} \psi_L \\ \chi_L \end{pmatrix}, \qquad \tan\varphi_{\psi_L} = \frac{\Delta}{m} \tag{5}$$

$$\begin{pmatrix} \tilde\psi_R \\ \tilde\chi_R \end{pmatrix} \to \begin{pmatrix} \cos\varphi_{\tilde\psi_R} & -\sin\varphi_{\tilde\psi_R} \\ \sin\varphi_{\tilde\psi_R} & \cos\varphi_{\tilde\psi_R} \end{pmatrix} \begin{pmatrix} \tilde\psi_R \\ \tilde\chi_R \end{pmatrix}, \qquad \tan\varphi_{\tilde\psi_R} = \frac{\tilde\Delta}{\tilde m} \tag{6}$$

Figure 3: Two similar alignments

parentheses may improve the quality of a formula. Compare top and bottom line

$$a\frac{f(x+h)-f(x)}{h}+b\frac{g(x+h)-g(x)}{h}$$
$$a\,\frac{f(x+h)-f(x)}{h}+b\,\frac{g(x+h)-g(x)}{h}$$

and decide which one looks better. In my notes I used the bottom one when working the proof of linearity of the derivative. If I talk about "the function $g(z) = \sqrt{z-1}$", I add a thin space before ending inline math mode:

```
``the function $g(z)=\sqrt{z-1}\,$''
```

in order to avoid the clash between the vinculum and the quotes in "the function $g(z) = \sqrt{z-1}$". Try with a parenthesis after the radical to see another case: $(1+\sqrt{2})^{-1}$ versus $(1+\sqrt{2}\,)^{-1}$. In the latter case a thin space has been added.

Going to *very* fine details: does anybody notice the differences below? Consider the formulas

$$\log|x| \neq \log|x| \tag{7}$$
$$|\sin x| \neq |\sin x| \tag{8}$$
$$\|\operatorname{adj} A\| \neq \|\operatorname{adj} A\| \tag{9}$$

where the questionable typesetting is on the left. While the top left *could* be a typographic choice (so long as it is consistent), the other formulas in the left-hand sides are definitely wrong.

The mathtools package provides a very good facility for handling these cases, namely

```
\DeclarePairedDelimiter{\abs}{|}{|}
```

that allows to type `\abs{\sin x}` and forget about the dreaded thin space, which can also be avoided by

```
\lvert\sin x\rvert
```

Which style to choose is a matter of personal preference and habit. I recommend not to abuse the facility: reserve it for *functions* such as absolute values, norms and similar objects. Don't exploit it for parenthesized expressions: something like

```
\paren{a+b}\paren{a-b}=a^2-b^2
```

hinders input reading and would print the same as `(a+b)(a-b)=a^2-b^2`. True, one could do

```
\paren[\big]{a\paren{b+c}}
```

but is this really more legible than

```
\bigl( a (b + c) \bigr)
```

that keeps the usual mathematical structure? That is, assuming `\big` size is really necessary, which it isn't in the particular case.

Since I mentioned trigonometric functions, look at

$$\sqrt{\sin x}+\sqrt{\cos x}+\sqrt{\tan x}$$

and explain what's going wrong. Yes, the tittle makes the difference! It makes 'sin' higher than 'cos' and moves up the radical sign; similarly with 'tan'. In my trigonometry notes I have

```
\let\cos\undefined
\DeclareMathOperator{\cos}
  {cos\vphantom{i}}
\let\tan\undefined
\DeclareMathOperator{\tan}
  {tan\vphantom{i}}
```

with which the above formula would become

$$\sqrt{\sin x}+\sqrt{\cos x}+\sqrt{\tan x}$$

Radicals often need fine control in order to get them aligned with each other. Some appropriate trick involving `\vphantom` or `\smash` can fix things up:

$$\sqrt{x}+\sqrt{y}\neq\sqrt{x}+\sqrt{y}$$

Again, left is the questionable output; the formula on the right has been input as

```
\sqrt{x}+\sqrt{\smash[b]{y}}
```

The alternative

```
\sqrt{\mathstrut x}+\sqrt{\mathstrut y}
```

doesn't seem as attractive: $\sqrt{x}+\sqrt{y}$. Radicals would need a full chapter, so I'll stop here, except for one last thing: add a thin space when a radical is followed by a fence; similarly, add a thin space when a big operator (summation, product, integral) in display math mode is preceded by a fence and its limits are wide. Example:

$$\left(\sum_{k=1}^{n} a_k\right) \neq \left(\sum_{k=1}^{n} a_k\right)$$

4 Upright or italic?

Rivers of (electronic) ink have been spilled trying to answer the question. Actually it cannot be answered: mathematicians and engineers agree to disagree. Physicists disagree with each other.

Part of the question is: should constants be typeset in an upright font or not? The ISO 80000-2:2009 standard prescribes upright; adhering to this standard is mandatory in some technology and commercial fields. This is a good thing: people reading a technical report or manual will have no doubt about the meaning of a symbol.[4] While I strongly disagree with several decisions of the ISO standard, on mathematical grounds, I accept the underlying philosophy towards uniformity in the technical fields. Surely I appreciate its ban on the mathematically wrong $\sin^{-1}$ and similar: the standard has disputable aspects, but it's never wrong from a mathematical point of view.

On the other hand, many mathematicians are traditionalists and prefer italics for constants such as e (the Euler number) and i (the imaginary unit). Euler and Gauss used italics for the latter, and I'm among those who don't dare to challenge their authority. Of course, I know that mathematical notation has changed along time. I'd not use Cayley's original notation for matrices[5] because a better notation has developed. I follow the practice of setting standard function names in upright type (sine, cosine, logarithm and so on) even when ancient mathematicians didn't.

However, such decrees as 'symbols for vectors should be bold italic serif lowercase, for matrices should be bold italic serif uppercase, for tensors should be bold italic sans serif uppercase' make me smile: as a mathematician, I know that vectors, matrices and tensors are not different objects from a mathematical point of view. Matrices admit an easier two-dimensional representation: this is the 'big' difference.

For pedagogical reasons, I might use distinctive typesetting for vectors and matrices in a students' textbook. In a research paper or graduate level book I'd probably not make any distinction, if not mandated by clarity. In this case I'd explain the notation choices at the beginning of the paper or book.

A very fine book by J. Dieudonné [2], in the English edition by Academic Press, uses

- $\mathbf{R}$ or $\mathbf{C}$ for number sets,
- X for manifolds, E for vector bundles,
- A for vector space operators,
- $\mathrm{T}_x\mathrm{X}$ or $\mathrm{T}_x(f)$ for the tangent space or linear mapping,
- $d_x\mathbf{f}$ or $d_x f$ for the differential at x of a mapping (vector valued or scalar valued),
- $\boldsymbol{Z}$ for tensor fields,

and several other conventions that are consistently followed across the book and the series. The book starts off with a nine page long notation section. The same notation is used in the original French version.

However, it happens that book translations use different conventions from the original. It is the case for W. Rudin's 'Real and Complex Analysis' [9] where the differential 'd' is in italics, whereas it's upright in the Italian translation published by Bollati-Boringhieri [10]. I disagree with the publisher: maybe the editorial preference is for the upright 'd', but the author's style should be preserved as much as possible.

Not a big deal, one could think. No, this reflects on the *meaning* of the differential 'd'. There are several arguments in favor or against italics; my feeling is that most pure mathematicians prefer italics.

By the way, how to input the symbol in such a way that the convention can be changed at will? The simplest and more effective way is to define

```
\newcommand{\diff}{\mathop{}\!d}
```

(or `\mathrm{d}` if one *must* have the abomination).

I believe I learned this from Claudio Beccari through a `comp.text.tex` post. The code was credited to him in the paper [5],[6] but I'm not sure about the real source of this code pearl. Claudio Beccari had earlier proposed much more complicated code [1], namely

[4] A problem with ISO standards is that they have to be *bought*; the one we're talking about prices 158 CHF, about 143 € or $160 at the current exchange rate.

[5] `https://tex.stackexchange.com/q/487643`

[6] The paper is also available in English [6].

```
\makeatletter
\providecommand*{\diff}{%
  \@ifnextchar^{\DIfF}{\DIfF^{}}%
}
\makeatother
\def\DIfF^#1{%
  \mathop{\mathrm{\mathstrut d}}%
  \nolimits^{#1}%
  \gobblespace
}
\def\gobblespace{%
  \futurelet\diffarg\opspace
}
\def\opspace{%
  \let\DiffSpace\!%
  \ifx\diffarg(%
    \let\DiffSpace\relax
  \else
    \ifx\diffarg[%
      \let\DiffSpace\relax
    \else
      \ifx\diffarg\{%
        \let\DiffSpace\relax
      \fi
    \fi
  \fi
  \DiffSpace
}
```

What's the idea in the complicated definition? Look whether a superscript follows; if it doesn't, add a dummy one. Well, this is already wrong, because it adds `\scriptspace` unconditionally. After that, the next token is examined: if it is a fence, then don't add `\!`, because a `\mathop` is followed by a fence with no thin space; in case an ordinary symbol follows, the `\mathop` would add a thin space, which is removed by `\!`. Well, try it with `\diff\bigl(x+y\bigr)`. Next try the simpler definition and see! Where's the trick? The *empty* `\mathop` is followed by an ordinary symbol, the 'd'; we just need to remove the excess thin space! The thin space preceding the empty `\mathop` is inserted automatically by TEX following the rules. Thus we can define

```
\newcommand{\tder}[2]
  {\frac{\diff #1}{\diff #2}}
```

without worrying that spurious spaces may creep in. Instead

```
\iint\limits_{D} f(x,y) \diff x \diff y
```

will typeset as needed

$$\iint\limits_{D} f(x,y)\,dx\,dy$$

For differential forms

$$f(x,y)\,dx \wedge dy$$

the spacing will be automatically right.

The same paper by Claudio [1] proposes commands for the constants, namely

```
% The number 'e'
\providecommand*{\eu}
  {\ensuremath{\mathrm{e}}}
% The imaginary unit
\providecommand*{\iu}
  {\ensuremath{\mathrm{j}}}
```

I strongly disagree with proposing `\ensuremath`; referring in the text to the Euler's number by

```
We use \eu\ to denote...
```

is by no means easier and clearer than

```
We use $\eu$ to denote...
```

One keystroke more? So what? That's a mathematical symbol so it ought to be typed in math mode, just like when we talk about the variable x. *My* definition would be

```
\newcommand{\eu}{\mathord{e}}
```

so typing `\eu` outside of math mode would raise an error. Change to `\mathrm` if you prefer upright type.

During the preparation of this paper, I examined the toptesi bundle, to find

```
\providecommand{\eu}{%
  \ensuremath{%
    {\mathop{\mathrm{e}}\nolimits}%
  }%
}
```

This is disputable in several respects:

- `\ensuremath` serves no real purpose;
- `\nolimits` can be safely omitted, because the `\mathop{...}` bit is followed by `}`, so surely there are no limits to take into account;
- `\mathop` itself is redundant, because the whole thing is braced, so it is treated as an ordinary symbol.

Oh, wait! No, `\mathop` is actually wrong! Consider the following code:

```
\documentclass{standalone}
\usepackage{amsmath}
\newcommand{\euA}{\mathrm{e}}
\newcommand{\euB}{%
  \ensuremath{%
    {\mathop{\mathrm{e}}\nolimits}%
  }%
}
\begin{document}
$2\euA\euB$
\end{document}
```

The output is shown in figure 4. Do you see the problem? A single character in the argument to `\mathop` is raised or lowered so that it extends the same distance above and below the math axis.

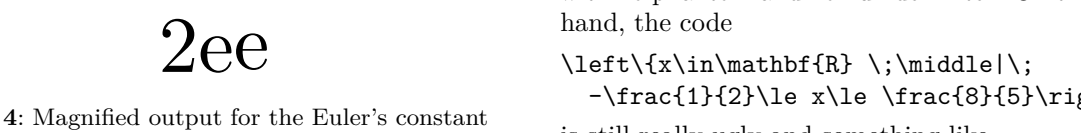

Figure 4: Magnified output for the Euler's constant problem

Now that we're on the spot, how to define a better `\tder` macro also supporting higher order derivatives? The first attempt,

```
\newcommand{\tder}[3][]{%
  \frac{\diff^{#1}#2}{\diff #3^{#1}}%
}
```

has a flaw: it unconditionally adds `\scriptspace` to both the numerator and denominator. If I measure the width of `\tder{f}{t}` in display math mode, with the standard fonts and document class, I get `14.07712pt`; the version without the dummy exponents has width `11.91045pt`. More than two points! With the upright 'd', the difference would be half a point. And the visual result shows more:

$$\frac{df}{dt} \neq \frac{df}{dt} \qquad \frac{\mathrm{d}f}{\mathrm{d}t} \neq \frac{\mathrm{d}f}{\mathrm{d}t}$$

Yes, we need to avoid the dummy superscript, also with the upright 'd', although the difference is less noticeable: we want perfect output, don't we? And we want macros that allow users to choose their own preferred 'd'. One could test whether the argument is empty, but there's a better way with xparse:

```
\NewDocumentCommand{\tder}{s o m m}{%
  \IfBooleanTF{#1}{\dfrac}{\frac}%
    {\diff\IfValueT{#2}{^{#2}}#3}% num
    {\diff #4\IfValueT{#2}{^{#2}}}% den
}
```

The *-version delivers `\dfrac` (just in case one needs it), otherwise `\frac` is used. The numerator and the denominator add the exponent only if the optional argument is specifically used. Thus `\tder{f}{t}` will not add a dummy exponent.

5 Sets, bras and kets

A short note on the title. Physicists have a sense of humor: a well-established notation for inner products is $\langle x \mid y \rangle$, called a "bracket". A mathematician would denote the linear or semilinear forms induced by the bracket as $\langle x \mid - \rangle$ and $\langle - \mid y \rangle$. Physicists, instead, use $\langle x|$ for the former and $|y\rangle$ for the latter, calling them "bra" and "ket".

For several years, LaTeX has been requiring e-TeX extensions, among which `\middle` is a very useful one. For instance, we can typeset

$$\left\{ x \in \mathbf{R} \;\middle|\; -\frac{1}{2} \le x \le \frac{8}{5} \right\}$$

with no phantom and no null delimiter. On the other hand, the code

```
\left\{x\in\mathbf{R} \;\middle|\;
  -\frac{1}{2}\le x\le \frac{8}{5}\right\}
```

is still really ugly and something like

```
\set*{x\in\mathbf{R}\suchthat
  -\frac{1}{2}\le x\le \frac{8}{5}}
```

would be much nicer. We call xparse and expl3 to the rescue!

```
\documentclass[varwidth]{standalone}
\usepackage{amsmath}
\usepackage{xparse}

\ExplSyntaxOn
\NewDocumentCommand{\set}{som}
 {
  % limit the scope for \suchthat
  \group_begin:
  \cs_set_protected:Npn \suchthat
   {
    \tl_use:N \l__egreg_set_st_tl
   }
  \IfBooleanTF{#1}
   {
    \egreg_set_auto:n { #3 }
   }
   {
    \egreg_set_fixed:nn { #2 } { #3 }
   }
  \group_end:
 }

\tl_new:N \l__egreg_set_st_tl

\cs_new_protected:Nn \__egreg_set_st:n
 {
  \tl_set:Nn \l__egreg_set_st_tl { #1 }
 }

\cs_new_protected:Nn \egreg_set_auto:n
 {
  \__egreg_set_st:n
   {
    \nonscript\;
    \middle\vert
    \nonscript\;
   }
  \left\{ #1 \right\}
 }

\cs_new_protected:Nn \egreg_set_fixed:nn
 {
  \tl_if_novalue:nTF { #1 }
```

```
   {
    \__egreg_set_st:n { \mid }
    \lbrace #2 \rbrace
   }
   {
    \__egreg_set_st:n
     { \mathrel{#1\vert} }
    \mathopen{#1\lbrace}
    #2
    \mathclose{#1\rbrace}
   }
 }
\ExplSyntaxOff

\begin{document}

$\set{a,b,c}\cup\set[\big]{a,b,c}$

$\set{x\suchthat a<x<b}$

$\set[\Big]{x\suchthat a<x<b}$

$\set*{x\suchthat \dfrac{1}{2}<x<3}$
\end{document}
```

The idea is to use a syntax familiar from mathtools' `\DeclarePairedDelimiter`. The output is in figure 5.

$$\{a,b,c\}\cup\{a,b,c\}$$
$$\{x \mid a<x<b\}$$
$$\left\{x \;\middle|\; a<x<b\right\}$$
$$\left\{x \;\middle|\; \frac{1}{2}<x<3\right\}$$

Figure 5: Examples of set notation

In *The TEXbook*, Knuth recommends to add thin spaces when the set builder notation contains a bar, that is, it is not just a list of elements. I disagree, but how could it be implemented? It's possible to look for the presence of `\suchthat` at the outer level and, in this case, to add the thin spaces at either end; nested sets would examine their own contents for the presence at the outer level.

A full implementation would also feature the choice for the delimiter as a preamble setting. I leave this as an exercise for whoever wants to make a package out of this code.

There is some duplication in the code below, but it's unavoidable. The reason is that using an `O{}` specifier for the optional argument would allow `\mathclose{#2\rbrace}` and no case distinction. However, one can see the difference if a subscript is added

```
\rbrace_{1}   \mathclose{\rbrace}_{1}
```
$\rbrace_1$ $\mathclose{\rbrace}_1$

Different coding is possible, though. It would not be difficult to allow `|` instead of `\suchthat`. Look at how the macros for bras and kets can be defined.

```
\documentclass[varwidth]{standalone}
\usepackage{amsmath}
\usepackage{xparse}

\NewDocumentCommand{\bra}{som}{%
  \IfBooleanTF{#1}
    {\left\langle #3 \right|}
    {%
     \IfNoValueTF{#2}
      {\langle#3\mathclose|}
      {\mathopen{#2\langle}#3\mathclose{#2|}}%
    }
}
\NewDocumentCommand{\ket}{som}{%
  \IfBooleanTF{#1}
    {\left| #3 \right\rangle}
  ,  {%
     \IfNoValueTF{#2}
      {\mathopen|#3\rangle}
      {\mathopen{#2|}#3\mathclose{#2\rangle}}%
    }
}

\NewDocumentCommand{\braket}{som}{%
  \IfBooleanTF{#1}
   {\extensiblebraket{#3}}
   {\fixedbraket{#2}{#3}}%
}

\ExplSyntaxOn
\NewDocumentCommand{\extensiblebraket}{m}
 {
  \group_begin:
  \char_set_active_eq:nN { `| } \egreg_bar_auto:
  \mathcode`|="8000 \scan_stop:
  \left\langle
  #1
  \right\rangle
  \group_end:
 }

\NewDocumentCommand{\fixedbraket}{mm}
 {
  \group_begin:
  \char_set_active_eq:nN
    { `| }    % active char is |
    \egreg_bar_fixed: % equal to
  \mathcode`|="8000 \scan_stop:
  \IfNoValueTF{#1}
   { \egreg_braket:n { #2 } }
   { \egreg_braket:nn { #1 } { #2 } }
  \group_end:
 }

\cs_new_protected:Nn \egreg_bar_auto:
```

```
 {
  \nonscript\,\middle\vert\nonscript\,
 }
\cs_new_protected:Nn \egreg_bar_fixed:
 {
  \mathinner{\egreg_size: \vert}
 }
\cs_new_protected:Nn \egreg_braket:n
 {
  \cs_set_protected:Nn \egreg_size: { }
  \langle #1 \rangle
 }
\cs_new_protected:Nn \egreg_braket:nn
 {
  \cs_set_protected:Nn \egreg_size: { #1 }
  \mathopen{\egreg_size: \langle}
  #2
  \mathclose{\egreg_size: \rangle}
 }
\ExplSyntaxOff

\begin{document}

$\bra{x}\quad\ket{x}$

$\braket{x|y}$

$\braket[\Big]{x|y|z}$

$\braket*{a|b}$

$\braket[\Big]{a|b|\dfrac{c}{d}}$

$\braket*{a|b|\dfrac{c}{d}}$
\end{document}
```

I'll not comment the code, except for mentioning how easy is to define the value of a character when it will be made active (math active, in this case).

$$\langle x| \quad |x\rangle$$
$$\langle x \,|\, y\rangle$$
$$\left\langle x \,\middle|\, y \,\middle|\, z \right\rangle$$
$$\langle a \,|\, b\rangle$$
$$\left\langle a \,\middle|\, b \,\middle|\, \frac{c}{d} \right\rangle$$
$$\left\langle a \,\middle|\, b \,\middle|\, \frac{c}{d} \right\rangle$$

Figure 6: Examples of bras and kets

6 Numbers and units

How should numbers be typed in the LaTeX document? Knuth himself once acknowledged that his usual practice is not very good and realized it when writing "Concrete Mathematics" [3], were numbers are typeset with the Euler font when they're used in their mathematical meaning (and not, say, as page markers).

When a number appears in text and is mentioned as a mathematical object is should be input inside a math formula:

```
a vector space of dimension~$5$
```

But what about large numbers that need to be split in smaller units for readability? For instance, can you spell out 7400043022221 without first counting how many digits the number has? Isn't 7 400 043 022 221 easier to parse? Possibly not for an American who's more accustomed to 7,400,043,022,221 (and probably would be at stake when people talks about meters and liters).

Now let's suppose your scientific paper has several tables with numeric data and you're not sure about the editorial policy of the journal to which you're submitting it. Will the journal require American style or prefer thin spaces for grouping digits?

Table 1: Tables with different formatting options for numbers (Source: Mr Leporello, private communication)

Nation	Number
Italy	640 375
Germany	231 803
France	100 002
Turkey	91 329
Spain	1 003 000

Nation	Number
Italy	640,375
Germany	231,803
France	100,002
Turkey	91,329
Spain	1,003,000

Let's consider the two tables in table 1. They are typeset with exactly the same input, namely

```
\begin{tabular}{
  @{}
  l
  S[table-format=7.0]
  @{}
}
\toprule
Nation  & {Number} \\
\midrule
Italy   &  640375 \\
Germany &  231803 \\
France  &  100002 \\
Turkey  &   91329 \\
Spain   & 1003000 \\
\bottomrule
\end{tabular}
```

and it's siunitx [12] doing all the magic. Of course there is a catch: just before the second copy of the table I added

```
\sisetup{group-separator={,}}
```

I could have added the option also in the bracketed argument to the `S` column, which is one of the facilities made available by the package. Similarly, the big number above has been typeset first with `\num{7400043022221}` and then with

```
\num[group-separator={,}]{7400043022221}
```

The default for the package is to use a thin space as a group separator between digits. An `S` column basically applies `\num` to every entry, but also aligns them at the decimal separator. In the case of our Leporello table, all entries are integers, so they're right aligned.

If an entry belonging to an `S` column is braced, it will be ignored as far as number alignment is concerned and centered on the total width of the column (options are available for left or right alignment). This is obviously needed in the header.

With another option we can easily scale down the figures:

Nation	Number
Italy	640×10^3
Germany	232×10^3
France	100×10^3
Turkey	91.3×10^3
Spain	1.00×10^6

This is achieved with the options

```
\sisetup{
  round-mode=figures,
  round-precision=3,
  scientific-notation=engineering
}
```

and by changing the column specifier to

```
S[table-format=3.2e1]
```

which directs to reserve space for three digits in the integer part, two in the mantissa and one in the exponent. The table body in the input has not changed in any way.

One might write an entire large chapter of *The LaTeX Companion* about siunitx. Some time ago, Joseph Wright took up the job of making a successor package to SIunit adding some features along the way. For version 2 he had the idea of exploiting expl3 which not only allowed for *many* more features and facilities, but made him enter the LaTeX team.[7] He's into chemistry, and tables with numeric data are his staple food.

The main purpose of the package is of course typesetting numbers with their SI unit according to the guidelines of the Bureau International des Poids et Mesures (BIPM). This is also part of the ISO standard mentioned before:

```
\SI{1}{\newton} is defined as
\SI{1}{\kilogram\meter\per\second\squared}
```

will typeset "1 N is defined as $1\,\mathrm{kg\,m\,s^{-2}}$". However, if we prefer slashes instead of negative exponents, we can add to the preamble

```
\sisetup{per-mode=symbol}
```

and the same text will now typeset as "1 N is defined as $1\,\mathrm{kg\,m/s^2}$". The mode can also be changed on a local basis with an optional argument to `\SI`.

All SI units and prefixes are supported:

```
\SI{5}{\tera\meter} \SI{2}{\pico\farad}
```

yields 5 Tm and 2 pF. One can also print just a unit with `\si`: the unit for energy is the J which is the same as $\mathrm{kg\,m^2\,s^{-2}}$.

Going on with our fictional scientist who's uncertain where her breakthrough paper will be published, decimal numbers might require the period as separator, or the comma; in scientific notation, there could be the $\times 10^n$ part or En might be asked for. How to do it? Not to mention uncertainty! Let's take as an example the rest mass of the electron

$9.109\,383\,701\,5(28) \times 10^{-31}\,\mathrm{kg}$

$9.1093837015(28) \times 10^{-31}\,\mathrm{kg}$

$9{,}109\,383\,701\,5(28) \times 10^{-31}\,\mathrm{kg}$

$(9.109\,383\,701\,5 \pm 0.000\,000\,002\,8) \times 10^{-31}\,\mathrm{kg}$

$9.109\,383\,701\,5 \times 10^{-31}\,\mathrm{kg}$

$9.109\,383\,701\,5(28)\mathrm{E}{-31}\,\mathrm{kg}$

The first line has been input with

```
\SI{9.1093837015(28)E-31}{\kilogram}
```

and the following lines by adding an option

```
\SI[⟨option⟩]{9.1093837015(28)E-31}
  {\kilogram}
```

The used options are, in order,

```
output-decimal-marker={,}
group-digits=integer
separate-uncertainty
omit-uncertainty
output-exponent-marker=\mathrm{E}
```

and they can be combined to get the desired effect *without changing the code in the document* if the settings are done with `\sisetup` in the document preamble. When inputting numbers, one can use spaces and either a decimal period or decimal comma. The first mandatory argument to `\SI` behaves the same as the mandatory argument to `\num`, so I'll use the latter:

[7] It seems that understanding and propagating expl3 opens a straight way to the team.

```
\num{12345.678}
\num{12345,678}
\num{12 345.678}
```

will print the same

12 345.678 12 345.678 12 345.678

Very few things are hardwired in siunitx: one can instruct it to ignore something, for instance. Suppose you have a set of numbers with comma separators for groups: the big number already used might occur in the source file as `7,400,043,022,221`. Set (globally or locally) the `input-ignore` option and we can remove the comma from the possible decimal separators:

```
\num[
  input-ignore={,},
  input-decimal-markers={.}
]{7,400,043,022,221}
```

and you'll get

7 400 043 022 221

The package is not limited to 'standard numbers': it also copes with angles, time and complex numbers. For instance, we can type

```
\ang{30.24}
\ang{30;12;44.375}
\num{3-4i}
```

to get

$30.24°,\ 30°12′44.375″,\ 3 - 4\mathrm{i}$

Oh, dear! An upright 'i'! Let's fix it with

```
\sisetup{
  output-complex-root=\mathnormal{i}
}
```

(one could also tell it to use 'j', of course) and get

$3 - 4i$

Phew! Yes, the package obviously adheres to the ISO standard, but it's very customizable.

7 Further reading

Twenty-two years have passed from the seminal paper by Claudio Beccari: we have seen great progress in the field of math typesetting, in particular towards the uniformity that's necessary in technical and commercial reports.

There are other packages that can be tried for the purpose of compliance to the ISO 80000-2:2009 standard. I would mention isomath by Günter Milde [7] and also unicode-math by Will Robertson [8] that provided facilities to the purpose; the former is for legacy `pdflatex`, the latter for XeLaTeX and LuaLaTeX.

References

[1] C. Beccari. Typesetting mathematics for science and technology according to ISO 31/XI. *TUGboat* 18(1), 1997. https://tug.org/TUGboat/tb18-1/tb54becc.pdf

[2] J. Dieudonné. *Treatise on Analysis. Vol. III.* Academic Press, New York-London, 1972. Translated from the French by I. G. MacDonald, Pure and Applied Mathematics, Vol. 10-III.

[3] R. L. Graham, D. E. Knuth, and O. Patashnik. *Concrete Mathematics.* Addison-Wesley Publishing Company, Advanced Book Program, Reading, MA, 1989. A foundation for computer science.

[4] E. Gregorio. Simboli matematici in TeX e LaTeX. *ArsTEXnica* 8:7–24, Ottobre 2009. https://www.guitex.org/home/numero-8

[5] M. Guiggiani and L. F. Mori. Consigli su come *non* maltrattare le formule matematiche. *ArsTEXnica* 5:5–14, Aprile 2008. https://www.guitex.org/home/numero-5

[6] M. Guiggiani and L. F. Mori. Suggestions on how *not* to mishandle mathematical formulae. *TUGboat* 29(2), 2008. https://tug.org/TUGboat/tb29-2/tb92guiggiani.pdf

[7] G. Milde. isomath — mathematical style for science and technology, 2012. Version 0.6.1. https://ctan.org/pkg/isomath

[8] W. Robertson. Experimental Unicode mathematical typesetting: The unicode-math package, 2019. Version 0.8o. https://ctan.org/pkg/unicode-math

[9] W. Rudin. *Real and Complex Analysis.* McGraw-Hill Book Co., New York–Toronto, Ont.–London, 1966.

[10] W. Rudin. *Analisi Reale e Complessa.* Bollati Boringhieri, 1974.

[11] C. Tellechea. L'extension pour TeX et LaTeX systeme, 2019. Version 0.32. https://ctan.org/pkg/systeme

[12] J. Wright. siunitx — A comprehensive (SI) units package, 2018. Version 2.7. https://ctan.org/pkg/siunitx

◇ Enrico Gregorio
Dipartimento di Informatica,
Università di Verona
enrico dot gregorio at univr dot it