A Short Guide to Nath

M. Marvan

3 January 2002

§1. Annotation. Nath is a LaTeX style to separate presentation and content in mathematical typography. The style delivers a particular context-dependent presentation on the basis of a rather coarse context-independent notation. Although essentially backward compatible with LaTeX, Nath aims at producing traditional math typography even from sources devoid of aesthetic ambitions. Its name is derived from "*na*tural ma*th* notation" (see [4]).

§2. License. Nath is a free software distributed under the terms of the GNU General Public License, see `http://www.gnu.org/copyleft/gpl.html`.

§3. Usage. To install Nath, put the `nath.sty` file into the TeX input directory. A LaTeX 2.09 document may start like

```
\documentstyle[nath]{article}
```

Under LaTeX 2_ε, the effect is achieved with

```
\documentclass{article}
\usepackage{nath}
```

Nath does not introduce any new fonts. See §28 for combining Nath and other LaTeX styles.

§4. Local options. A few Nath options may be set in the body of a document. The command `\nathstyle` accepts a list of arguments of the form '*name*=*value*' or '*name*'; the latter having the same meaning as '*name*=`on`'. Currently supported options are `geometry` (see §10), `tensors` (see §18), `leqno` (see §22), and `silent` (see §5).

§5. Errors and warnings. Nath errors are visualized by ▮ (or whatever is `\natherrormark`) placed where the error manifests itself (which may look misplaced). Unlike errors, Nath warnings appear only in the `log` file and do so only if the local option (see §4) `silent` is set to `on`.

Be aware that once admissible constructions may produce TeX errors now. E.g., superfluous braces may be harmful in math formulas except around macro arguments. Therefore, `{` and `}` should be used just where something (a sub- or superscript, a numerator, a denominator, and similar) begins or ends.

§6. Math modes. Nath uses two distinct math modes. The single dollar sign `$` invokes the *in-line* mode. The double dollar sign `$$` as well as other math environments invoke the *display* mode.

Observe the difference: `$(1 + \frac xy)^2$` typesets as $(1 + x/y)^2$, while

```
$$
(1 + \frac xy)^2
$$
```

typesets as

$$\left(1 + \frac{x}{y}\right)^2,$$

even though the notation is one and the same.

Commands `\inline` and `\displayed` force either mode on a subexpression. Sub- and superscripts are normally typeset in in-line mode; but

```
$$
(\sum_{i=1}^n x_i^p)^{\displayed{\frac 1p}}
$$
```

produces the *display* mode in the *script* size:

$$\left(\sum_{i=1}^n x_i^p\right)^{\frac{1}{p}}.$$

Never leave delimiters un`\displayed` in these cases.

The four math style switches of TeX newly refer only to the *size* of math expressions: `\scriptstyle` and `\scriptscriptstyle` to the script and second-level-script size of the *current* size; `\textstyle` is void; whereas `\displaystyle` has a special meaning in the context of the principle of smallest fences (see §8).

§7. Fractions. Fractions indicate division in a very broad sense (cf. $\partial f/\partial x$) and may occur in three shapes:

$$\text{built-up } \frac{A}{B}, \qquad \text{piece } \tfrac{1}{2}, \qquad \text{solidus } A/B.$$

Nath provides a single universal command `\frac` (besides of the obvious slash, '`/`'). The resulting shape is determined by special algorithms (see [4]).

§8. Displayed fractions. Non-numeric fractions come out as built up. According to what we call the *principle of smallest fences*, numeric fractions are typeset built up if and only if this does not extend any paired delimiters. E.g.,

```
$$
(\frac 12 + x)(\frac 12 + \frac 1x)
$$
```

results in

$$(\tfrac{1}{2}+x)\left(\frac{1}{2}+\frac{1}{x}\right).$$

One can circumvent the rule in two possible ways.

(i) In order to force a built-up fraction, place `\displaystyle` anywhere within the nearest pair of delimiters. E.g.,

$$\left(\frac{1}{2}+x\right)\left(\frac{1}{2}+\frac{1}{x}\right)$$

results from

```
$$
(\frac 12 + x\displaystyle)(\frac 12 + \frac 1x)
$$
```

(ii) In order to force a case fraction, insert an extra pair of invisible delimiters. E.g.,

$$\int x\,dx = \tfrac{1}{2}\,x^2$$

results from

```
$$
\int x\,dx = \left. \frac12 x^2 \right.
$$
```

Compound fractions have their numerator and denominator in display mode:

$$\frac{1+\dfrac{x}{y}}{1-\dfrac{x}{y}}.$$

One can, of course, force the in-line mode. Namely,

```
$$
\frac{\inline{1 + \frac xy}}{\inline{1 - \frac xy}}
$$
```

or, even better,

```
\newcommand\ifrac[2]{\frac{\inline{#1}}{\inline{#2}}}
$$
\ifrac{1 + \frac xy}{1 - \frac xy}
$$
```

(cf. §26) typesets as

$$\frac{1+x/y}{1-x/y}.$$

§9. In-line fractions. A `\frac` with numeric arguments results in a case fraction, such as the Bernoulli number $B_{12} = -\tfrac{691}{2730}$. Otherwise we get a solidus fraction and parentheses are added whenever needed for preservation of the mathematical meaning. E.g.,

```
$\frac{\frac ab}{\frac cd}$
```

produces $(a/b)/(c/d)$.

Examples below present one and the same expression in display and in-line mode. Roughly speaking, Nath assumes that binary operations other than slash have less binding power than the slash,

$$\frac{a+b}{c+d} \qquad (a+b)/(c+d),$$

$$\frac{\frac{a\cdot b}{c}\cdot d}{c\cdot d} \qquad ((a\cdot b)/c\cdot d)/(c\cdot d),$$

$$x+\frac{a}{b} \qquad x+a/b.$$

In particular, this rule applies to the binary operations of commutative algebra:

$$\frac{A}{B}\otimes\frac{C}{D} \qquad A/B\otimes C/D,$$

$$\frac{A\otimes B}{C\otimes D} \qquad (A\otimes B)/(C\otimes D),$$

even though existing tradition may be different in this particular case. On the other side, *juxtaposition* has more binding power than the slash:

$$\frac{a}{b}\frac{c}{d} \qquad (a/b)(c/d),$$

$$\frac{\partial}{\partial x}\frac{f}{g} \qquad (\partial/\partial x)(f/g),$$

$$d\frac{u}{v} \qquad d(u/v),$$

$$\frac{\partial^3 f}{\partial x\,\partial y^2} \qquad \partial^3 f/\partial x\,\partial y^2,$$

$$\frac{a}{bc} \qquad a/bc.$$

Nath only avoids inserting parentheses between a fraction and a numeric coefficient, e.g.,

$$-\frac{u}{v}+2\frac{u}{v}-\frac{1}{2}\frac{a}{b} \qquad -u/v+2u/v-\tfrac{1}{2}a/b,$$

unless there is a danger of confusion, e.g.,

$$2\frac{\pm u}{v} \qquad\qquad 2(\pm u/v).$$

In case of loose juxtaposition between operator and its argument, there is no obvious winner, thus

$$\frac{\sin x}{2} + \sin\frac{x}{2} \qquad\qquad (\sin x)/2 + \sin(x/2).$$

Of course, no parentheses will be inserted when they are already present in one or another form:

$$A\left[\frac{u}{v}\right]^2 \qquad\qquad A[u/v]^2,$$

$$\frac{(x,y)}{\lVert x\rVert\,\lVert y\rVert} \qquad\qquad (x,y)/\lVert x\rVert\,\lVert y\rVert$$

(the last example uses `\lVert x \rVert \, \lVert y \rVert` in the denominator).

Grouping prevents Nath from adding parentheses around the whole fraction: `$a{\frac bc}$` typesets as ab/c, otherwise as $a(b/c)$. To be on the safe side, avoid superfluous braces in math formulas (cf. §5).

To disable parentheses around the numerator or denominator, a pair of invisible parentheses is needed: `$\frac{\left.\sin x\right.}{\cos x}$` typesets as $\sin x/\cos x$, otherwise as $(\sin x)/\cos x$.

An important remark is due. Professional typographers generally follow the rule that 'a/bc means a divided by bc.' Still some mathematicians (especially those with a programming background) argue that if juxtaposition denotes multiplication, then a/bc means $a/b\cdot c$, which is $(a/b)\cdot c$ by the commonly accepted rules of precedence. However, ab and $a\cdot b$ are different notations and it is the notation what matters in typography. Yet the AIP style manual [1] is cautious enough to say just: "do not write $1/3x$ unless you mean $1/(3x)$." Altogether, notation a/bc is considered ambiguous by a nonignorable part of the mathematical community. Then, at least, the choices made by Nath are known, traditional, and easy to remember.

And, of course, it is never unwise to display difficult fractions.

§10. Delimiters.

TeX's `\left` and `\right` produce rather poor results, especially when overused or underused. Under natural notation, every fence is a left or right delimiter by its very nature, and delimiters do their best to match the material enclosed:

$$\frac{M}{\left(1-\frac{x_1+\cdots+x_n+pZ}{r}\right)\left(1-p\frac{\frac{\partial Z}{\partial x_2}+\cdots+\frac{\partial Z}{\partial x_n}}{\rho}\right)}.$$

For matching purposes, every Nath mathematical object is assigned an auxiliary height and depth; sub- and superscripts as well as accents do not contribute to these dimensions, hence “small parts” may exceed the fences:

$$(\widetilde{P} - \widetilde{Q})\Bigl(1 + \prod_{i=1}^{\lfloor\sqrt{n}\rfloor} p_i\Bigr)^2.$$

Needless to say, line breaks are allowed between delimiters. E.g.,

$$\sin 2nx = 2n\cos x\Biggl[\sin x \\ \qquad + \sum_{k=1}^{n}(-4)^k\frac{(n^2-1^2)(n^2-2^2)\cdots(n^2-k^2)}{(2k-1)!}\sin^{2k-1}x\Biggr]$$

results from the simple

```
$$
\sin 2nx = 2n \cos x [\sin x \\
\qquad + \sum_{k = 1}^n (-4)^k
 \frac{(n^2 - 1^2)(n^2 - 2^2) \dots (n^2 - k^2)}{(2k - 1)!}
 \sin^{2k - 1} x]
$$.
```

The modifiers `\left` and `\right` still must be used with symmetric delimiters (e.g., vertical lines $|$ and $\|$) or when intended to override the natural disposition (e.g., `\left]`). The newly introduced modifiers `\double` and `\triple` create double and triple delimiters. E.g., `$\double[u_1,\dots,u_n\double]$` produces $[\![u_1,\dots,u_n]\!]$.

The *middle delimiters*, such as `\mid` and `\middle|` for $|$, `\Mid` and `\double|` for $\|$, and `\triple|` for $|||$, have the size of the nearest outer pair of delimiters. For example:

$$\Bigl\{(x_i)\in R^\infty\Bigm|\sum_{i=1}^{\infty}x_i^2 = 1\Bigr\}.$$

With nested delimiters, there are two ways to ensure that outer delimiters come out bigger than inner ones. In display mode this is controlled by a count `\delimgrowth`. Setting the `\delimgrowth` to n makes (approx.) every nth delimiter bigger. One should set `\delimgrowth=1` when a display contains many vertical bars (and insert extra `\,` between adjacent right and left bars).

In in-line mode, the *command* `\big` has the effect that the next entered level of delimiters is set in big size (in the sense of plain TeX). It is not necessary that the `\big` is immediately followed by a delimiter; and `\bigg` is an abbreviation for `\big\big`. For instance, `$\Delta\big \frac 1{f(x)}$` produces $\Delta\bigl(1/f(x)\bigr)$; in this way one can enlarge implicit delimiters such as those induced by the command `\frac`. It is an error to place a `\big` within delimiters that are not

Left delimiters		Right delimiters	
`(`	$($	`)`	$)$
`[`,`\lbrack`	$[$	`]`,`\rbrack`	$]$
`\{`, `\lbrace`	$\{$	`\}`, `\rbrace`	$\}$
`<`, `\langle`	$\langle$	`>`, `\rangle`	$\rangle$
`\lfloor`	$\lfloor$	`\rfloor`	$\rfloor$
`\lceil`	$\lceil$	`\rceil`	$\rceil$
`\lvert`, `\left|`	$\vert$	`\rvert`, `\right|`	$\vert$
`\lBrack`, `\double[`	$\llbracket$	`\rBrack`, `\double]`	$\rrbracket$
`\lAngle`, `\double<`	$\langle\langle$	`\rAngle`, `\double>`	$\rangle\rangle$
`\lFloor`	$\lfloor\lfloor$	`\rFloor`	$\rfloor\rfloor$
`\lCeil`	$\lceil\lceil$	`\rCeil`	$\rceil\rceil$
`\lVert`, `\ldouble|`	$\Vert$	`\rvert`, `\rdouble|`	$\Vert$
`\triple[`	$[[[$	`\triple]`	$]]]$
`\triple<`	$\langle\langle\langle$	`\triple>`	$\rangle\rangle\rangle$
`\ltriple|`	$\vert\vert\vert$	`\rtriple|`	$\vert\vert\vert$

Table 1: Paired delimiters

big themselves. Unbalanced delimiters may be present in an in-line formula (as is usual in tensor calculus — cf. §18), but then cannot be resized.

Table 1 lists paired delimiters. To enable `<` and `>` as a notation for angle braces, one must set `\nathstyle{geometry}` (this misusage of notation is common in geometry and math physics). As symbols of ordering, $<$ and $>$ can be always accessed through '`\lt`' and '`\gt`'.

While in math modes, brackets `[`, `]` never denote optional arguments. This helps to avoid common LaTeX misinterpretations, as with `\\[`. On the other side, *grouping* interspersed with delimiters — once harmless — is a serious defect now (cf. §5). E.g., `({x})` derails TeX if used in display mode.

§11. Operators. Nath typsets `\lambda\mathop{\rm id} - g` as

$$\lambda \operatorname{id} - g,$$

whereas TeX would put uneven spacing around the minus sign: $\lambda \mathop{\rm id} -g$, erroneously considering the minus sign a unary operator (by [3, rule 5 on p. 442]).

In subscripts of big operators, `\\` is allowed and starts a new line, e.g.,

```
$$
\sum_{i,j \in K \\ i \ne j} a_{ij}
$$
```

prints as

$$\sum_{\substack{i,j \in K \\ i \ne j}} a_{ij}.$$

Within math, the exclamation mark `!` alone ensures suitable spacing around factorials: `C^n_k = \frac{n!}{(n - k)!k!}` typesets as $C_k^n = n!/(n-k)!\,k!$ or

$$C_k^n = \frac{n!}{(n-k)!\,k!}.$$

May be doubled: $(2n)!! = n!\,2^n$.

Finally, integral signs stick one to another unless something else intervenes:

```
$$
\int\int\int_M dV.
$$
```

produces

$$\iiint_M dV.$$

§12. Abbreviations. According to typographic tradition, names of variables that are abbreviations should be typeset in roman, for which Nath offers a handy notation: abbreviations are letter strings starting from the back quote '`''. E.g., `$`e^{\pi`i}$` and `$`ad_x y$` typeset as $\mathrm{e}^{\pi\mathrm{i}} = -1$ and $\mathrm{ad}_x\,y$, respectively.

Strings containing more than one letter, such as `` `span``, become math operators. Until now they must have been declared in advance with some additional care to avoid conflicts (`\span` is a TEX primitive). Some more examples:

$$\begin{aligned}
&H' = H'_{\mathrm{symm}} + H'_{\mathrm{antisymm}},\\
&\bar f = f|_{\operatorname{int} U},\\
&a = \mathrm{const}_1\,,\\
&G = \mathrm{SO}(n).
\end{aligned}$$

§13. Roots. Nath's `\sqrt` differs in several aspects. Firstly, its vertical size never depends on the presence of subscripts:

$$\sqrt{a} + \sqrt{a_j}\,.$$

Secondly, nested `\sqrt`'s are aligned at the top:

$$\cos\frac{\pi}{10} = \frac{1}{4}\sqrt{10 + 2\sqrt{5}}\,.$$

(Compare it with the TEX's

$$\cos\frac{\pi}{10} = \frac{1}{4}\sqrt{10 + 2\sqrt{5}}.)$$

Thirdly, no optional arguments are allowed. LATEX's `\sqrt[3]{x}` must be replaced with `\root{3}{x}` to produce $\sqrt[3]{x}\,$.

§14. Special symbols. Nath introduces `\vin` and `\niv` as names of the important symbols '⅃' and '⅂' not included in any standard math font.

Arrows `\to`, `\ot`, `\otto`, and `\mapsto` are expandable and descriptable via sub- and superscripts. Thus,

```
$$
A \to^f_{\text{isomorphism}} B, \qquad a \mapsto^f a'
$$
```

gives

$$A \xrightarrow[\text{isomorphism}]{f} B, \qquad a \xmapsto{f} a'.$$

The command `\adot` denotes the centered dot to be used a an argument placeholder, as in $f(\cdot)$ or $g(\cdot,\cdot)$.

§15. Horizontal braces. The upper and lower horizontal braces are created with `\underbrace{`*expression*`}_{`*label*`}` and `\overbrace{`*expression*`}_{`*label*`}`, respectively. For instance,

```
$$
f^n(x) = \underbrace{f(f(\dots f(}_{n \text{ times}}x) \dots))
$$
```

results in

$$f^n(x) = \underbrace{f\Big(f(\cdots f(}_{n \text{ times}}x)\cdots)\Big)$$

Observe that the construction does not interfere with the displayed mode of delimiters.

§16. Accents. Hat, tilde, and bar accents are extensible and grow wider with the size of the accented material:

$$\hat{a} + \widehat{ab} + \widehat{abc}.$$

When these accents outreach their limit of extensibility, they take the superscript position:

$$(a + b + c)^{\wedge}.$$

A sequence of accents goes from top to down or from right to left. For instance, `\hat\bar a + \hat\bar{ab} + \hat\bar{abc}` gives

$$\hat{\bar{a}} + \widehat{\overline{ab}} + \widehat{\overline{abc}},$$

whereas `\hat\bar{a + b + c}` typesets as

$$(a + b + c)^{-\wedge}.$$

All kinds of things may happen if braces intervene as in `\bar{\bar{ab}}`.

Let us note that `\bar` is not arbitrarily extensible, unlike `\overline`. For instance, `\hat{\overline{a + b + c}}` gives $\overline{a+b+c}^{\wedge}$ (over- and underlines and arrows are *not* accents). Over a single character, there is no limit on the number and type of accents in the sequence; e.g.,

$$\hat{\ddot{\tilde{W}}}$$

results from `\hat\ddot\tilde W`. Over an expression, a non-extensible accent, like `\dot`, makes others non-extensible as well. Thus, `\hat{ab} + \dot{ab} + \dot\hat{ab} + \hat\dot{ab}` gives

$$\widehat{ab} + (ab)^{\bullet} + (ab)^{\wedge\bullet} + (ab)^{\bullet\wedge}.$$

§17. Arrays. Entries are typeset in display mode:

$$\begin{vmatrix} x & 1 \\ 1 & \dfrac{1}{x} \end{vmatrix} = 0.$$

Moreover, arrays grow smaller when used in sub- and superscripts:

$$\mathrm{e}^{\begin{pmatrix} a & b \\ c & d \end{pmatrix}}.$$

A `matrix` environment differs from `array` in that it does not have any preamble. As a special case, `\binom{`*m*`}{`*n*`}` creates the binomial coefficient $\binom{m}{n}$.

§18. Tensors. With `\nathstyle{tensors}`, first-level sub- and superscripts to ordinary symbols occupy predetermined positions. Thus,

$$A^{[k} B^{l]}_{(k} C_{l)}$$

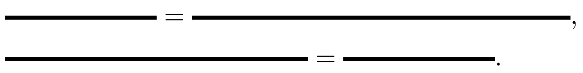
results from

```
\nathstyle{tensors=on}
$$
A^{[k} B^{l]}_{(k} C_{l)}
$$
```

(unbalanced delimiters are allowed in in-line style).

§19. Displayed formulas. Displayed formulas are indented by `\mathindent` of default value of 4 pc. With `\mathindent` set to a negative length, displayed formulas are centered. Formulas enclosed between double dollars `$$` are unnumbered. Alternatively one may enclose them between `\[` and `\]`. Ends of lines (any formula may be multiline) are marked with `\\`. Nath does not support automatic line breaks (as does the Downes style [2]).

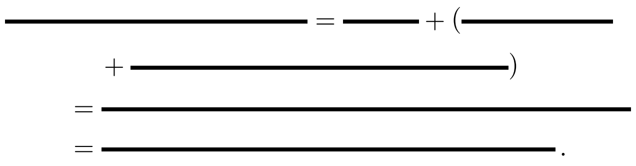
E.g., `$$ stuff = stuff , \\ stuff = stuff . $$` typesets as a left-aligned multiline formula (the punctuation is important, see §24):



To achieve finer arrangements, one may begin every continuation line with a number of `\quad`'s; e.g., two in front of a binary relation, three in front of a binary operation:

```
$$
stuff = stuff + (stuff \\
\qqquad + stuff) \\
\qquad = stuff \\
\qquad = stuff .
$$
```

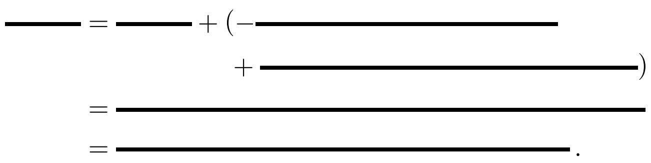
gives



§20. Walls. Walls represent a simple and convenient tool to achieve better visual appearance of complex displayed equations. The syntax is `\wall` *stuff* `\\` *stuff* `\\` ⋯ `\\` *stuff* `\return`, and can be arbitrarily nested. The `\wall` makes every next line to start at the "wall" until removed by `\return`. For instance,

```
$$
stuff
\wall = stuff + (\wall - stuff \\
                         + stuff)
                    \return
       = stuff \\
       = stuff.
\return
$$
```

gives

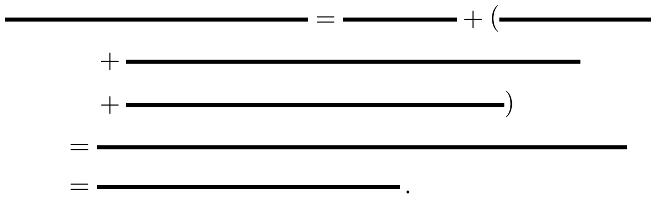


The typical placement of `\wall` is in front of a relation symbol or immediately after an opening delimiter anywhere in the left half of a formula.

A simple alternative is `\padded{`A`}`, which prefixes each subsequent line with A until stopped by `\return`. Typically, A is a kern:

```
$$
\padded\qquad \padded\quad stuff = stuff + (stuff \\
                                            + stuff \\
                                            + stuff)
          \return
          = stuff \\
          = stuff
\return
$$
```

gives



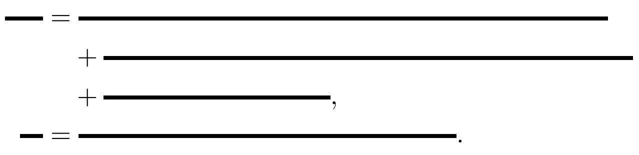
With short formulas it may be easier to prefix each line with explicit `\quad`'s as we did in §19.

See §24 on the interplay between walls and punctuation.

§21. Alignments. Unfortunately, display mode of delimiters interferes badly with alignments unless every cell is balanced (as is, e.g., with matrices). The recommended solution is to fill the cells with balanced wall/return blocks. E.g.,

```
\begin{eqnarray*}
stuff &=& \wall stuff \\
            + stuff \\
            + stuff,
      \return
\\
stuff &=& stuff
\end{eqnarray*}
```

produces



Walls save `&`'s and ensure vertical centering of the equation numbers (see §22).

§22. Equation numbering. A formula enclosed between `\begin{equation}` and `\end{equation}` obtains a single number (the value of `\theequation`) on the right. Putting the command `\numbered` inside of an unnumbered formula has the same effect:

```
$$
stuff. \numbered
$$
```

results in

$$\rule{6cm}{1pt}. \tag{1}$$

Alternatively, `\eqno{`A`}` makes A the equation number.

In emergency, the equation number goes one line below the formula:

$$\rule{8cm}{1pt}$$
$$\tag{2}$$

We already know that any formula may be multiline. If so, the equation number is centered:

$$\begin{gathered}\rule{6cm}{1pt},\\ \rule{5.5cm}{1pt}.\end{gathered} \tag{3}$$

To have centered numbers within the `eqnarray` environment, use wall/return blocks as described in §21 (but then the equation numbers may be overwritten with the formula content without warning).

There is also the `eqns` environment, which puts a number on each line:

$$\rule{6cm}{1pt}, \tag{4}$$
$$\rule{5.5cm}{1pt}. \tag{5}$$

It also uses larger and breakable interline space. Multiline blocks then may be created by using the walls (§20).

Equation numbering is normally determined by `\theequation`. The environment `subabc` introduces a subordinate numbering by letters,

$$A = B, \tag{6a}$$

no matter how many numbered equations are enclosed,

$$C = D. \tag{6b}$$

This output was obtained from

```
\begin{subabc}
\begin{equation}
A = B, \label{A}
\end{equation}
no matter how many numbered equations are enclosed,
\begin{equation}
C = D. \label{C}
\end{equation}
\end{subabc}
```

After `\end{subabc}`, the original numbering mode is restored:

$$E = F. \tag{7}$$

Every numbered equation should be referred to somewhere, hence it should have a label — a warning (§5) is issued if it does not.

To put equation numbers on the left, call either the documentstyle option `leqno` or the local option `\nathstyle{leqno}`.

§23. Items. Lay typographers tend to overuse list environments. Rather than list items, numbered statements so often encountered in theorems and definitions may be alternatively formatted as numbered paragraphs. Nath's command `\paritem{`*item label*`}` starts a numbered paragraph and may occur even within a displayed formula. Our next example demonstrates this:

The following statements on a real function f are equivalent:
(i) f is continuous;

(ii) $$f(\lim_{i\to\infty} x_i) = \lim_{i\to\infty} f(x_i)$$

for every converging sequence x_i.

In a left-numbered formula, `\paritem` supersedes the numbering and a warning is issued.

§24. Punctuation. Nath provides a simple tool to encourage line breaks after punctuation in in-line mode. Namely, `\␣` denotes a breakable space no matter where it is used. Therefore, `$a = b,\ c = d$` will break after the comma, $a = b$, $c = d$, rather than after the '$=$' sign. The inclination to break is measured by `\punctpenalty` (if a positive integer less than 10000).

Three dots are denoted by `\dots`. In some contexts, their proper place is at the level of math axis, e.g., $a_1+\cdots+a_n$. Nath uses a very simple rule — the dots are not raised if and only if they follow a comma or a semicolon. Accordingly, we have $a_1, \dots, a_n$ and $a_1; \dots; a_n$.

Punctuation after displayed formulas is important for recognizing continuing lines. Without punctuation, what seems to be a system of equations

$$U_x = AU$$
$$-\, U_y = BU$$

may well be a chain of them:

$$U_x = AU - U_y = BU.$$

To disambiguate your notation, be sure to insert comma (or semicolon or full stop) at the end of each line that is not continued:

$$U_x = AU,$$
$$-U_y = BU.$$

(Observe that the minus sign starting the second line is typeset closer to U — becomes a unary operator.)

§25. Spacing. Nath's displayed formulas use frozen spacing (TEX's "skips" and "glues" neither stretch nor shrink). While it is seldom useful to stretch a displayed formula, one may wish to shrink formulas too wide to fit between the margins. Within the `tight` environment, displayed formulas occupy slightly less horizontal space. E.g.,

$$\sin^6 x = -\tfrac{1}{32}\cos 6x + \tfrac{3}{16}\cos 4x - \tfrac{15}{32}\cos 2x + \tfrac{5}{16}$$

becomes

$$\sin^6 x = -\tfrac{1}{32}\cos 6x{+}\tfrac{3}{16}\cos 4x{-}\tfrac{15}{32}\cos 2x{+}\tfrac{5}{16}$$

if written as

```
\begin{tight}
$$
\sin^6 x =
 -\frac 1{32} \cos 6x + \frac 3{16} \cos 4x
 - \frac{15}{32} \cos 2x + \frac 5{16}
$$
\end{tight}
```

Striving for safe defaults, Nath sets even interword spaces in text. TEXperts may wish to call `\nonfrenchspacing` (see [3, p. 74]) to achieve a century-old look.

§26. User definitions. Feel free to introduce your own commands by using `\newcommand` or `\def`. We already gave a useful example of `\ifrac` in §8.

Here is another example: A first-order partial derivative suitable for all math modes and sizes can be introduced via

```
\newcommand\pd[2]{\frac{\partial#1}{\partial#2}}
```

We then have $\bigl((\partial f/\partial x)(\partial g/\partial y)\bigr)^2$ or $\mathrm{e}^{((\partial f/\partial x)(\partial g/\partial y))^2}$ or

$$\left(\frac{\partial f}{\partial x}\frac{\partial g}{\partial y}\right)^2$$

from one and the same `(\pd f x \pd g y)^2`.

The price is that fragile commands occurring inside in-line math may have to be protected (any in-line mode material must be considered a "moving argument"). Nath commands are robust by design and need no `\protect`ing. When encountering a mysterious error, such as "undefined command `\wrapfrac@`," fragile commands are to be blamed. Besides `\protect`, Nath offers `\makerobust`, a command that takes an already assigned control sequence as argument and makes it robust.

§27. Efficiency. Nath helps to prevent wasting human work on something that can be done by computer. On average, LATEX runs about three times slower with Nath than without it, depending on the complexity of math formulas.

§28. Other packages. Nath is not guaranteed to be compatible with other LaTeX packages. However, some combinations turn out to be safe and useful. For example, when starting a LaTeX 2.09 document with

```
\documentstyle[amssymb,nath]{article}
```

or a LaTeX 2_ε document with

```
\documentclass{article}
\usepackage{amssymb,nath}
```

one invokes `amssymb`, a component of the famous $\mathcal{AMS}$-LaTeX package from the American Mathematical Society, thereby introducing a wider range of mathematical symbols. Users can also enable text mode `amsmath` commands by starting a LaTeX 2_ε document with

```
\usepackage{amsmath,nath}
```

(math mode commands must be those of Nath).

§29. Commands of enhanced functionality. A number of math commands have been redefined; `\old{`*command*`}` often provides access to what `\`*command* was before Nath redefined it (see the source code of this guide for examples).

Here is the list of all enhanced and newly introduced commands:

`\␣`	a breakable space in math (§25)
`\\`	see §11 and §19
`\abbreviation`	a long form of `‘` in math (§12)
`\adot`	argument placeholder (§14)
`\arraycolsep`	macro, formerly a dimension register (§17)
`\big`	making inline delimiters bigger (§10)
`\bigg`	same as `\big\big` (§10)
`\biggg`	same as `\big\big\big` (§10)
`\biggl`	same as `\big\big\left`
`\bigl`	same as `\big\left`
`\binom`	binomial coefficient (§17)
`\delimgrowth`	see §10
`\displayed`	forcing displayed math mode (§6)
`\double`	doubling a delimiter (§10)
`\eqno`	equation number (§22)
`\natherrormark`	a mark to visualize nath errors (§5)
`\factorial`	long form of `!` in math (§11)
`\fbox`	making frame around a subformula
`\frac`	fraction (§7)
`\gt`	greater than sign (§10)
`\hat`	attaching hat accent (§16)
`\inline`	forcing in-line math mode (§6)
`\int`	integral sign (§11)

`\langle`	left angle bracket (§10)
`\lAngle`	left double angle bracket (§10)
`\lbrace`	left brace (§10)
`\lbrack`	left bracket (§10)
`\lBrack`	left double bracket (§10)
`\lceil`	left ceiling bracket (§10)
`\lCeil`	left double ceiling bracket (§10)
`\ldouble`	left doubling (§10)
`\left`	left modifier (§10)
`\lfloor`	left floor bracket (§10)
`\lFloor`	left double floor bracket (§10)
`\lnull`	left invisible fence (§10)
`\lt`	less than sign (§10)
`\ltriple`	left tripling (§10)
`\lvert`	left vertical line (§10)
`\lVert`	left double vertical line (§10)
`\mapsto`	sizeable ‘$\mapsto$’ (§14)
`\mathop`	see §11
`\mathstrut`	see [3]
`\mid`	middle vertical line (§10)
`\Mid`	middle double vertical line (§10)
`\middle`	middle modifier (§10)
`\Nath`	logo
`\nathstyle`	local options (§4)
`\niv`	the symbol ‘∟’ (§14)
`\nonumber`	suppresses equation number (§22)
`\numbered`	forces equation number (§22)
`\old`	see the beginning of this section
`\ot`	sizeable left arrow (§14)
`\otto`	sizeable left-right arrow (§14)
`\overbrace`	horizontal braces over unbalanced math material (§15)
`\overleftarrow`	left arrow over an expression
`\overleftrightarrow`	left-right arrow over an expression
`\overline`	overline an expression (§16)
`\overrightarrow`	right arrow over an expression
`\padded`	like a wall, with every next line padded (§20)
`\paritem`	numbered statement (§23)
`\punctpenalty`	penalty inserted after punctuation in math (§24)
`\quad`	1em space (§19)
`\qquad`	2em space (§19)
`\qqquad`	3em space (§19)
`\rangle`	right angle bracket (§10)
`\rAngle`	right double angle bracket (§10)
`\rbrace`	right brace (§10)
`\rbrack`	right bracket (§10)

`\rBrack`	right double bracket (§10)
`\rceil`	right ceiling bracket (§10)
`\rCeil`	right double ceiling bracket (§10)
`\rdouble`	right doubling (§10)
`\return`	ends `\wall` and `\padded` (§20)
`\right`	right modifier (§10)
`\rfloor`	right floor bracket (§10)
`\rFloor`	right double floor bracket (§10)
`\rnull`	right invisible fence (§10)
`\root`	arbitrary root (§13)
`\rtriple`	right tripling (§10)
`\rvert`	right vertical line (§10)
`\rVert`	right double vertical line (§10)
`\scriptscriptstyle`	setting size to second next level script size
`\scriptstyle`	setting size to next level script size
`\sqrt`	square root (§13)
`\stackrel`	as in LaTeX
`\text`	text within math
`\tilde`	attaching tilde accent (§16)
`\to`	sizeable right arrow (§14)
`\triple`	tripling a delimiter (§10)
`\underbrace`	horizontal braces under unbalanced math material (§15)
`\underleftarrow`	left arrow under an expression
`\underleftrightarrow`	left-right arrow under an expression
`\underline`	underline an expression
`\underrightarrow`	right arrow under an expression
`\vin`	the symbol '⌟' (§14)
`\wall`	begin a wall/return block (§20)

Redefined and new environments:

`array`	see §17
`cases`	as in TeX
`eqnsabc`	`eqns` within `subabc`
`eqnarray`	as in LaTeX
`eqnarray*`	as in LaTeX
`eqnarrayabc`	`eqnarray` within `subabc`
`eqns`	a pile of equations (§22)
`equation`	as in LaTeX
`matrix`	see §17
`subabc`	subnumbering by letters (§22)
`tight`	tighter spacing (§25)

The following characters are active, retaining their previous meaning: `$`, `^`, `_`. Other characters become active in math mode:

`(`	see §10

`)`	see §10
`[`	see §10
`]`	see §10
`<`	see §10
`>`	see §10
`,`	see §24
`;`	see §24
`!`	see §11
`` ` ``	see §12

Commands that became obsolete are still preserved in reduced form for backward compatibility:

`\Big`	ignored
`\Bigg`	ignored
`\Biggl`	same as `\left`
`\biggm`	same as `\middle`
`\Biggm`	same as `\middle`
`\biggr`	same as `\right`
`\Biggr`	same as `\right`
`\Bigl`	same as `\left`
`\bigm`	same as `\middle`
`\Bigm`	same as `\middle`
`\bigr`	same as `\right`
`\Bigr`	same as `\right`
`\mathchoice`	`useless and discouraged`
`\mathpalette`	`useless and discouraged`
`\textstyle`	ignored

The following TEX commands are disabled:

`\atop`
`\over`
`\choose`

The following LATEX environment is disabled:

`math`

New ifs (correspond to local options):

`\ifgeometry`	see §10
`\ifleqno`	see §22
`\ifsilent`	see §5
`\iftensors`	see §18

New dimension registers:

`\arraycolsepdim`	former `\arraycolsep`
`\displaylineskiplimit`	
`\mathindent`	see §19
`\mex`	a prorated `ex`
`\paritemwd`	see §23

New skips (self-explanatory):

```
\displaybaselineskip
\displaylineskip
\interdisplayskip
\intereqnsskip
\beloweqnsskip
```

New boxes:

`\sizebox`	delimiters match it (§10)

Moreover, Nath takes box and token registers on the fly.

§30. Final remarks. Nath is a scientific software intended to assist and ease the process of scientific publication. By disburdening the encoding of mathematics, Nath tries to uphold TEX's position as a language suitable for both scientific and typographic purposes — especially if alternatives are still elusive.

Nath is provided as it is; only bug reports and serious discussion should go to `M.Marvan@math.slu.cz`.

References

[1] *AIP Style Manual*, 4th edition (Amer. Inst. Physics, New York, 1990).

[2] M. Downes, Breaking equations, *TUGboat* 18 (1997) 182–194.

[3] D.E. Knuth, *The TEXbook* (Addison Wesley, Reading, 1984).

[4] M. Marvan, Natural TEX notation in mathematics, in: Proc. Conf. *EuroTEX 2001*, Kerkrade, 23–27 September 2001; online `www.ntg.nl/eurotex/proceedings.html`.