Semantic Enrichment of Mathematics via 'tooltips'*

Ross Moore

Macquarie University, Sydney, Australia
ross.moore@mq.edu.au

Abstract. A package mathsem for pdf-LaTeX implements a way to provide semantic meaning to symbols, without adding a large syntactical burden to the specification of a mathematical expression. It uses a concept of 'active comment', allowing the '%' character at the beginning of a new line to become an active token under highly-controlled circumstances. With a strictly defined syntax, words to express the semantic meaning of a variable ('x' say) can be associated with each occurrence of 'x' in the expression following. The words become content of a tooltip, that 'pops-up' by the symbol in a PDF document. The idea extends to:

1. allow multiple instances of the same symbol have distinct meanings;
2. attach semantics to macro-names as well as character symbols;
3. allow nested tooltip rectangles, for sub-expressions;
4. assign defaults to be attached to symbols and macros, at either global or local levels, to maintain consistency of meaning within extended portions of a document.

It is planned to use the same syntactical constructions to provide words for spoken 'alternative text', in the context of fully-tagged, accessible, mathematical content within PDF documents.

Thanks to Michael Kohlhase for ideas suggesting such a package.

1 Introduction

Mathematical notation has been refined over centuries into a very succinct syntax, particularly for input to computer programs, where input such as `\(a + b\)` using LaTeX, produces the typeset visual form: $a + b$. — Call this *Example A*. However, such an expression contains no real *semantics* describing what the mathematics actually represents. How can an author provide such information conveniently, without compromising the high-quality visual representation?

To address this we take a lead from computer software interfaces, where use of 'tooltips', small windows containing a short textual description that pop-up, has become common-place. An extra benefit of tooltips is that their content is, according to the PDF Reference specifications [1,3], intended to be vocalised (e.g., by Adobe Reader's 'Read Out Loud') as a replacement for the content which lies under the tooltip's Button annotation, or by other means. Thus throughout this

* The final publication is available at link.springer.com as M. Kerber et al. (Eds.): CICM 2015, LNAI 9150, pp. 1–5, 2015. DOI: 10.1007/978-3-319-20615-8

paper the term 'semantics' could be taken to mean a 'well-structured, meaningful, vocal or extra rendition' of mathematical content, understandable without the need to see the visual form of the expression.

Several LaTeX packages[1] support tool-tips, with all employing rather verbose coding. For example, in a linguistics context, one might want something like $S + O$ using coding[2] as follows — call this *Example B*: (Test it[3] yourself.)

```
\pdftooltip{S}{S,Subject}\pdftooltip{+}{plus,Verb}\pdftooltip{O}{O,Object}
```

This puts quite a heavy burden on a document's author to keep both the visual form of the mathematics, and associated semantics, fully correct during editing.

This kind of 'low-level' semantic enrichment of mathematical content is different to, but need not be incompatible with, using LaTeX to semantically enrich documents for a 'Semantic Web' as described in [2]. These use the sTeX collection of LaTeX macro packages [6]; see *Example F* below for relevant remarks.

2 Adding semantics to math environments

To help alleviate the burden, the mathsem package[4] employs a concept of 'active comment' (or 'semantic comment'). This allows extra words to be introduced into the TeX-based processing and stored for later use. These can appear as text in a tooltip, or be used in other ways such as 'alternative text' for vocalisation purposes. We use the term 'tagging' to refer to addition of such information, with symbols and expressions becoming 'tagged'. Without the package, the comments are simply ignored, thus giving the correct visual form with no edits required.

A series of examples follow, in sequence with *A* and *B* above. Each introduces concepts and explains syntax used in the package. *Example D* shows how user-defined macros overcome an inherent difficulty. This opens up great flexibility allowing semantics to be applied to not just individual symbols, but subexpressions and the environment as a whole; see *Examples E* and *F*.

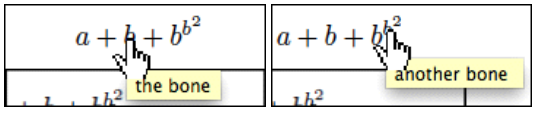
Example C. At the simplest level, meanings of symbols are passed using TeX comments, without interrupting the coding of the mathematical expression.

```
\(
%$semantics
% a $ the dog
% b $ the bone; another bone
% + $ eats
% 2 $ many bones
%$endsemantics
a + b + b^{b^{2}}
% other comments ignored
\)
```

Spacing and vertical positioning of symbols should be unaffected.

Non-tagged output: $a + b + b^{b^2}$

Tagged output: $a + b + b^{b^2}$



[1] pdfcomment, fancytooltips, cooltooltips. Click for links to CTAN.
[2] This uses the \pdftooltip command from the pdfcomment package [7].
[3] Not all PDF readers support tooltips; Adobe Reader® is recommended.
[4] ... downloadable from https://rutherglen.science.mq.edu.au/~maths/CICM/.

TEXnical Note. This works using TEX's *category-code* mechanism [8] to adjust the roles of '`%`' and end-of-line, so lines beginning with '`%`' no longer need be ignored. Such 'semantic comment' lines are parsed to identify characters to which semantics can be attached. Valid syntax is lines of the following forms:

`%` ⟨*token*⟩ `$` ⟨*semantics*⟩ ⟨*line-end*⟩
`%` `$`⟨*keyword*⟩ ⟨*line-end*⟩

with spaces allowed either side of the ⟨*token*⟩ and '`$`' delimiter, chosen since '`$`' can never occur within a math-environment. The ⟨*semantics*⟩ can specify a list, delimited with '`;`' (semi-colon), of text snippets to be used with successive instances of the same character with the final one persisting for continued use. (See the line for '`b`' in *Example C*). The `$semantics` keyword is optional, reminding an author of the purpose of semantic comments which follow.

After reading a semantic comment and storing its ⟨*semantics*⟩, TEX 'looks ahead'[5] to see whether the next line also starts with '`%`'. If so, this is scanned for a semantic comment; otherwise the scope terminates. Any line beginning with an active '`%`' followed by nothing, '`$endsemantics`', an unrecognised keyword or other violation of the strict syntax, is treated as an ordinary comment; the scope for active '`%`'s terminates. Next, any found ⟨*token*⟩ characters are 'activated', by having their `\catcode` set to 13, before actual mathematical content starts. Such 'active' characters behave as a single-letter name for a macro, which expands to produce the tooltip before placing an (unactive) character into the math-list for normal processing. A single-digit exponent must be enclosed within braces (e.g., `..^{b^{2}}` in *Example C*), since the '`2`' now expands into more than just a single character being superscripted. This active nature of characters is carefully controlled; in particular, being restricted to just a single math-environment.

Example D. Use of active characters, as outlined in the *TEXnical Note*, has one drawback; but there is an easy work-around. When a letter variable, 'a' say, is activated for semantic enrichment, this letter cannot be used in any macro names employed directly within the same math-environment.

```
\(
%$semantics
% + $ added to
% a$ complex number a; a; a conjugate
%$endsemantics
\Re a = \tfrac12(a + \bar{a})
\)
```

The result can be surprising, producing either wrong output or a TEX error, or both: $\Re a = ac12(a + \underline{a}ra)$

```
! Undefined control sequence.
l.331 \Re a = \tfr
                ac12(a + \bar{a})
```

A warning message also is issued; *viz.*

```
LaTeX Warning: Command \b invalid in math mode on input line 331.
```

The reason is that the letter '`a`' cannot be used within macro names, as it no longer has the category code of a letter when it is later used as part of the mathematics. User-defined macros provide the solution as in the next example.

[5] using TEX's `\futurelet` primitive command

Example E.

```
\DeclareRobustCommand{\HALFOF}%
  {\tfrac12}%
\DeclareRobustCommand
   {\ACONJUGATE}{\bar{a}}%
\(
%$semantics
%\Re $ real part of
% a $ a complex number a ; a
% = $ equals
%\HALFOF $ half of
% ( $ open bracket
% + $ added to
%\ACONJUGATE$ its complex conjugate
% ) $ close bracket
%$endsemantics
 \Re a = \HALFOF (a + \ACONJUGATE)
\)
```

Existing macro names can also be enriched with semantics, provided the macro-name does not use any letters that are to be made active; in this example `\Re` is 'safe', but `\tfrac` and `\bar` are not, because of the active letter '`a`'. An 'unsafe' macro is protected within the expansion of a previously defined 'safe' macro. Since most usual TEX and LATEX commands use only lowercase letters to form the name, we employ just uppercase letters in the new names used here.

Non-tagged: $\Re a = \frac{1}{2}(a + \bar{a})$

Tagged: $\Re a = \frac{1}{2}(a + \bar{a})$

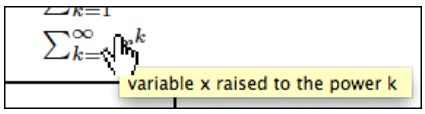
Being already active, the existing macro expansion is first saved under an internal name. Just as with active characters, a new expansion creates the tooltip before placing the actual mathematics using this saved expansion. Macro names themselves are best chosen to correspond to the semantic meaning, as in [2,5,6]. `\DeclareRobustCommand` allows for better log-messages and safe captions, etc.

Example F. This next example is based upon Example 1 in the STEX documentation [6], using macros `\CsumLimits` and `\Cpower` (incorrectly stated as `\Cexp` in [6]), as defined for STEX.

```
{\DeclareRobustCommand{\InfinSumXexpK}{\CsumLimits{k}1\infty{\XtotheK}}%
\DeclareRobustCommand{\XtotheK}{\Cpower{\varx}{\vark}}%
\DeclareRobustCommand{\vark}{k}%
\DeclareRobustCommand{\varx}{x}%
\(
%$semantics
% \vark $ bound variable k
% \varx $ free variable x
% \infty $ infinity
%\XtotheK $ variable x raised to the power k
%\InfinSumXexpK $ sum with k from 1 to infinity, of x to the power k
\InfinSumXexpK
\)}%
```

Non-tagged: $\sum_{k=1}^{\infty} x^k$

Tagged: $\sum_{k=1}^{\infty} x^k$



To indicate the available flexibility, this example uses the character 'k' both tagged, via `\vark`, and untagged in the lower limit of the sum. The upper limit of '∞' is tagged, whereas the lower limit '1' is not — but could have been. Being provided internally by the macro `\CSumLimits`, it is not possible to tag the '=' (equals sign). This would be possible if STEX had used a macro, `\EqualsSign` say, in its definition for `\CSumLimits`.

Example G. Frequently the same semantics will need to be attached to a variable or expression, being used in multiple environments within a paragraph or larger sectional unit within a publication. The following declares 'default' semantics

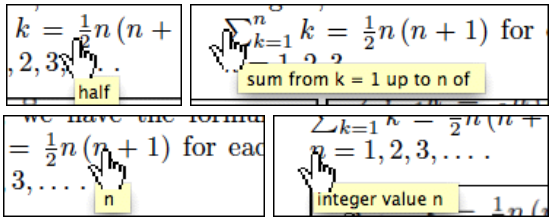
`\DeclareMathSemantics{`⟨*token*⟩`}{`⟨*semantics*⟩`}`

where ⟨*token*⟩ may be a single character or macro. The ⟨*semantics*⟩ then apply to instances of ⟨*token*⟩ inside all math-environments within the current TEX grouping. However, 'semantic comments' are still needed to activate characters.

```
{\DeclareMathSemantics{n}{integer value n }%
 \DeclareRobustCommand{\HALF}{\tfrac12}%
 \DeclareMathSemantics{\HALF}{half}%
 \DeclareRobustCommand{\INTEGERsumToN}{\sum_{k=1}^{n}}%
Summing consecutive positive integers, we have the formula
\(
% \INTEGERsumToN $ sum from k = 1 up to n of
% \HALF $;
% k $ bound variable k
% = $ equals
% ( $ open bracket
% n $ n
% + $ plus
% 1 $ unity
% ) $ close bracket
%
 \INTEGERsumToN{k} = \HALF n\,(n+1)
\) for each \(
% n $;
% = $ taking values
% \ldots $ and so on
%
 n = 1, 2, 3, \ldots\
\).
}%
```

Non-tagged: Summing consecutive positive integers, we have the formula $\sum_{k=1}^{n} k = \frac{1}{2} n\,(n+1)$ for each $n = 1, 2, 3, \ldots$.

Tagged: Summing consecutive positive integers, we have the formula $\sum_{k=1}^{n} k = \frac{1}{2} n\,(n+1)$ for each $n = 1, 2, 3, \ldots$.



Empty list elements, *viz.* '`% \HALF $;`' and '`% n $;`', specify the default to be used as semantics. On the RHS of the summation equality, the instances of n use semantics defined by '`% n $ n`' to override the default.

Defaults may be read from a file to apply to whole sections. Then local definitions can be used to override, or set a few specific new defaults to apply within a limited scope. Authors can then still use 'active comments' in the most specific instances that the default has not given the most appropriate meaning. This aspect is most important with machine-generated code, having final verification by a human, to provide the most relevant semantics within a particular context.

3 Possible future developments

Adding support for the aligned environments defined in the $\mathcal{AMS}$-math is a top priority for further development of the mathsem package. With 'active comments'

being a viable way to include extra (semantic) information into the processing of math-environments, one can envisage other places to use this; e.g., chemical formulæ, chess layouts, contexts where single letters represent complicated objects. More keywords can be implemented; e.g., to add meaningful /Alt text for PDF tagging (using `/Formula<</Alt(...)...>> BDC`) of complete mathematical formulae, in documents conforming to the PDF/UA [4] standard. All formulæ must have alternative text for the benefit of screen-readers and other assistive technology. Code for *Example B* might then start as follows.

```
\(
% $all
% "S + O" representing "Subject, Verb, Object"
% $semantics
```

Such tagging of structure with /Alt text applies to "Tagged PDF" generally, as in the author's earlier work [9,10,11]. Those methods need an interface to allow a document's author to supply words to override built-in defaults, where the meaning of a mathematical symbol can be dependent on context.

References

1. Adobe Systems Inc.; PDF Reference 1.7, November 2006. Also available as [3]. http://www.adobe.com/devnet/pdf/pdf_reference.html.
2. Groza, Tudor, Handschuh, Siegfried, Möller, Knud, Decker, Stefan: SALT – Semantically Annotated LaTeX for Scientific Publications. In Franconi, Enrico, Kifer, Michael, May, Wolfgang (eds.) ESWC 2007. LNCS vol. 4519, pp. 518–532. Springer, Heidelberg (2007)
3. ISO 32000-1:2008; Document management — Portable document format (PDF 1.7); Technical Committee ISO/TC 171/SC 2, July 2008
4. ISO 14289-1:2012; Document management applications – Electronic document file format enhancement for accessibility – Part 1: Use of ISO 32000-1 (PDF/UA-1); Technical Committee ISO/TC 171/SC 2. Corrected version. December 2014
5. Kohlhase M.: Using LaTeX as a Semantic Markup Format, Mathematics in Computer Science (2:2) pp. 279–304. Birkhäuser (2008). https://svn.kwarc.info/repos/stex/doc/mcs08/stex.pdf
6. Kohlhase M.: STeX: Semantic Markup in TeX/LaTeX. Documentation of the STeX packages. https://trac.kwarc.info/sTeX/export/2430/trunk/sty/stex.pdf
7. Kleber, J.: pdfcomment — A user-friendly interface to pdf annotations. LaTeX package; available from CTAN; version 2.3a, September 2012
8. D.E. Knuth: The TeXbook. Addison Wesley (1984, 1986 and later revisions)
9. Moore, R.R.: Ongoing efforts to generate "tagged PDF" using pdfTeX, Muni Press, 2009. Reprinted as: TUGboat, Vol. 30, No. 2 (2009)
10. Moore, R.R.: Tagged Mathematics in PDFs for Accessibility and other purposes, in *CICM-WS-WiP 2013*. CEUR Workshops Proceedings. Vol-1010, paper-01.pdf (2013)
11. Moore, Ross: PDF/A–3u as an archival format for Accessible mathematics. In Watt, Stephen M., Davenport, James H., Sexton, Alan P., Sojka, Petr, Urban, Josef (eds.) CICM 2014. LNCS, vol. 8543, pp. 184–199. Springer, Heidelberg (2014)