The lips LaTeX package
Chicago Manual text ellipses (Frankenstein's lips)

Matt Swift <swift@alum.mit.edu>

Version: 2.1 Date: 2001/08/31
Documentation revision: 2001/08/31

Abstract

The `\lips` command generates text ellipses that are closer to what *Chicago Manual of Style* suggests than what `\dots` produces. It does the right thing in most circumstances, and so is easy to use.

Contents

source	result
`Hello\lips.␣And`	Hello. . . . And
`Hello␣\lips.␣And`	Hello. . . . And
`Hello.\lips␣And`	Hello. . . . And
`Hello.␣\lips␣And`	Hello. . . . And
`one,\lips,␣three`	one, . . . , three
`one,␣\lips,␣three`	one, . . . , three
`Hello\lips␣and`	Hello . . . and
`Hello␣\lips␣and`	Hello . . . and
`Hello!\lips␣And`	Hello! . . . And
`Hello!␣\lips␣And`	Hello! . . . And

Figure 1: Examples of correct usage of `\lips`.

Part I
Discussion

1 Text Ellipses

`\lips` The macro `\lips` should be used for all text ellipses, that is, ellipses not part of a mathematical expression.

Chicago Manual does not acknowledge different spacings after different marks of punctuation, distinguish interword from intersentence space, or give rules about where to break a line near an ellipsis. Given a context where such spaces do vary and rules for linebreaks can be given, I arrived at the following rules for handling them around ellipses.

1. Pre-, post-, and intra-ellipsis space is normal intra-word space, which should be 3-to-em. In LaTeX there is also some glue (it's a "rubber space").
2. When non-period punctuation p follows `\lips`, p is followed by the space that normally follows p.
3. A period following `\lips` is treated as if the period preceded `\lips`. Conceptually, an ellipsis never precedes a period, so conceptually there is only one case; but it is convenient to be able to type either `\lips.` or `.\lips`.
4. Lines are a tiny bit breakable before ellipses and a tiny bit breakable after ellipses that are not followed by punctuation.

In figure 1 are examples of correct usage and in figure 2 is a comparison of `\lips` to `\dots`. You can substitute any of the characters `!"()*+,-/:;=?@[]' ‘|<>~{}` for the exclamation point in the examples. There can be additional contiguous spaces after `\lips`, just like after any command name. There can also be contiguous spaces before `\lips`.

2 Line breaking considerations

`\LPNobreakList` When `\lips` is followed by certain punctuation (`,:;?!)']}/` by default), we never

source	result
`Hello\dots.␣And`	Hello.... And
`Hello\lips.␣And`	Hello. . . . And
`Hello␣\dots.␣And`	Hello .... And
`Hello␣\lips.␣And`	Hello. . . . And
`Hello.\dots␣And`	Hello.... And
`Hello.\lips␣And`	Hello. . . . And
`Hello.␣\dots␣And`	Hello. ... And
`Hello.␣\lips␣And`	Hello. . . . And
`one,\dots,␣three`	one,..., three
`one,\lips,␣three`	one, . . . , three
`one,␣\dots,␣three`	one, ..., three
`one,␣\lips,␣three`	one, . . . , three
`Hello\dots␣and`	Hello...and
`Hello\lips␣and`	Hello . . . and
`Hello␣\dots␣and`	Hello ...and
`Hello␣\lips␣and`	Hello . . . and
`Hello!\dots␣And`	Hello!...And
`Hello!\lips␣And`	Hello! . . . And
`Hello!␣\dots␣And`	Hello! ...And
`Hello!␣\lips␣And`	Hello! . . . And

Figure 2: Comparison of `\lips` with `\dots`.

want to break a line after the ellipsis and before that punctuation. For the other punctuation, such as open parenthesis and open quote, we want normally breakable space following the ellipsis. `\LPNobreakList` is a list of those characters before which `\lips` should *never* break a line. To force unbreakable space following `\lips`, follow `\lips` with a tie (`~`).

3 Limitations

`\lips` does the right thing after `\par`, but after `\indent` or `\noindent` there is an extra preceding space (see the final section on testing for an example). Suggestions on how to handle this problem are welcome.

Part II
Implementation

4 Version control

\fileinfo
\DoXUsepackagE
\HaveECitationS
\fileversion
\filedate
\docdate
\PPOptArg

These definitions must be the first ones in the file.

```
\def\fileinfo{Chicago Manual text ellipses (Frankenstein's lips)}
\def\DoXPackageS {}
\def\initelyHavECitationS {}
\def\fileversion{v2.1}
\def\filedate{2001/08/31}
\def\docdate{2001/08/31}
\edef\PPOptArg {%
  \filedate\space \fileversion\space \fileinfo
}
```

If we're loading this file from a `\ProcessDTXFile` command (see the *compsci* package), then `\JusTLoaDInformatioN` will be defined; othewise we assume it is not (that's why the FunkY NamE).

If we're loading from `\ProcessDTXFile`, we want to load the packages listed in `\DoXPackageS` (needed to typeset the documentation for this file) and then bail out. Otherwise, we're using this file in a normal way as a package, so do nothing. `\DoXPackageS`, if there are any, are declared in the `dtx` file, and, if you're reading the typeset documentation of this package, would appear just above. (It's OK to call `\usepackage` with an empty argument or `\relax`, by the way.)

```
\makeatletter% A special comment to help create bst files.  Don't change!
\@ifundefined{JusTLoaDInformatioN} {%
  }{% ELSE (we know the compsci package is already loaded, too)
  \UndefineCS\JusTLoaDInformatioN
  \SaveDoXVarS
  \eExpand\csname DoXPackageS\endcsname\In {%use \csname in case it's undefined
    \usepackage{#1}%
  }%
  \RestoreDoXVarS
  \makeatother
  \endinput
}% A special comment to help create bst files.  Don't change!
```

Now we check for LaTeX2e and declare the LaTeX package.

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{lips}[\PPOptArg]
```

5 The macro

\Lips
\BracketedLips
\lips
\olips

If we're in math mode, raise an error. If we're in vertical mode, leave it and do an ellipsis not breakable at the end. If we're in horizontal mode, call `\lp@lips@hmode`.

Normally I would use `\NewRobustCommand`, but this seems not enough reason to require the *moredefs* package. We reserve the name `\lips` by hand. In the future, this package may require *moredefs*.

```
\newcommand*\Lips {}
```

```
\let\Lips\relax
\DeclareRobustCommand\Lips {%
  \ifmmode
    \def\sc@t@a {%
      \PackageError{lips}
        {Don't use \protect\lips \space in math mode; use \protect\dots.}\@ehc
    }%
  \else
    \ifvmode
      \def\sc@t@a {%
        \leavevmode
        .\nobreak\ .\nobreak\ .\nobreak\ %
      }%
    \else
      \let\sc@t@a\lp@lips@hmode
    \fi
  \fi
  \sc@t@a
}
\newcommand*\BracketedLips {}
\let\BracketedLips\relax
\DeclareRobustCommand\BracketedLips {%
  \ifmmode
    \def\sc@t@a {%
      \PackageError{lips}
        {Don't use \protect\lips \space in math mode; use \protect\dots.}\@ehc
    }%
  \else
    \ifvmode
      \def\sc@t@a {%
        \leavevmode
        [.\nobreak\ .\nobreak\ .]\nobreak\ %
      }%
    \else
      \let\sc@t@a\lp@blips@hmode
    \fi
  \fi
  \sc@t@a
}
```

To do: *need the relax to avoid mistaking as optarg in some contexts?*

In MLA style, that is, with the `mla` option, `\lips` gives you brackets, meaning you To get no brackets, use `\olips` for "original lips," that is, ellipses in the original document.

To do: *Better dox here. Are options case sensitive? If so, make MLA=mla.*

```
\newlet\lips\Lips
\newlet\olips\lips
\DeclareOption{mla} {%
  \let\lips\BracketedLips
}
\ProcessOptions
```

`\lp@lips@hmode` `\lp@blips@hmode` `\GobbleIgnoreSpaces`

The horizontal mode case. Then we remove the last skip on the horizontal list. If the next character is a period, we want to act as if the period preceded `\lips`, so

we plonk down a period, a minimally-breakable space, then an ellipsis ending with normally-breakable space. This keeps the ellipsis on the same line as the sentence it follows if at all possible, but allows it to spill onto the next line if there's no other way to avoid an overfull line. Then we want to gobble up the period that's about to come, and ignore any spaces after it, since we've already inserted proper space.

If the next character is not a period, we output a minimally-breakable space and an ellipsis. To decide whether to follow it with unbreakable or normally-breakable space, we need to know more about the next character than just that it is not a period, so we `\let` it to scratch `a` and call `\lp@lips@check@punct`.

To do: *document different logic with bracketed lips; how the heck to handle greater inter-sentence space?*

```
\newcommand\lp@lips@hmode {%
  \unskip
  \@ifnextchar . {%
      .\penalty9999\ .\nobreak\ .\nobreak\ .\ %
      \GobbleIgnoreSpaces
    }{% ELSE
    \penalty9999\ .\nobreak\ .\nobreak\ .%
    \futurelet\sc@t@a\lp@lips@check@punct
  }%
}
% hmm, have to keep the check for period i guess because i don't want it in
% nobreaklist?  because the break/nobreak is different?  because it's more efficient?
\newcommand\lp@blips@hmode {%
  \unskip
  \@ifnextchar . {%
      \penalty9999\ [.\nobreak\ .\nobreak\ .]%
    }{% ELSE
    \penalty9999\ [.\nobreak\ .\nobreak\ .]%
    \futurelet\sc@t@a\lp@lips@check@punct
  }%
}
\newcommand\GobbleIgnoreSpaces [1] {%
  \ignorespaces
}
```

`\LPNobreakList`
`\lp@<punctuation>`

We'd like to use `\ifcat.\sc@t@a` for this, but that won't work when we get a command sequence in scratch `a`. Bummer. Do I really need to define all this punctuation now? It protects against something becoming an active char or otherwise changing catcodes after the *lips* package is loaded.

```
\newcommand*\lp@bang {}
\newcommand*\lp@rparen {}
\newcommand*\lp@comma {}
\newcommand*\lp@slash {}
\newcommand*\lp@colon {}
\newcommand*\lp@semic {}
\newcommand*\lp@q {}
\newcommand*\lp@rbrack {}
\newcommand*\lp@rquote {}
\newcommand*\lp@tilde {}

```

```
\let\lp@bang=!
\let\lp@rparen=)
\let\lp@comma=,
\let\lp@slash=/
\let\lp@colon=:
\let\lp@semic=;
\let\lp@q=?
\let\lp@rbrack=]
\let\lp@rquote='
\let\lp@tilde=~

% \egroup

```

I'm making an attempt to put these in order of expected frequency. A doublequote could be at the beginning or end of a quotation, so half the time you'll have to correct by adding `{}`, so let's leave it out so there's one less to check.

```
\newcommand \LPNobreakList {%
  \lp@comma \lp@colon  \lp@semic
  \lp@q \lp@bang
  \lp@rparen \lp@rquote \lp@rbrack \egroup
  \lp@slash
}
```

`\lp@lips@check@punct` Add nonbreakable space if scratch `a` is in `\LPNobreakList`; add normally-breakable space otherwise. Add no space at all if scratch `a` is a tie—the tie will add nonbreakable space. We use just the guts of `\@tfor` for efficiency.

```
\newcommand\lp@lips@check@punct {%
  \ifx\sc@t@a\lp@tilde
  \else
    \@tempswatrue
    \let\@fortmp\PunctList
    \expandafter\@tforloop\LPNobreakList\@nil\@nil\@@\reserved@a {%
      \ifx\sc@t@a\reserved@a
\@tempswafalse
\@break@tfor
      \fi
     }%
    \if@tempswa
      \ %
    \else
      \nobreak\ %
    \fi
  \fi
}
```

You might want to compare the LaTeX version (as of 95/12/01). `\fontdimen3` is the interword stretchability, not interword space.

```
\DeclareTextCommandDefault{\textellipsis}{%
  .\kern\fontdimen3\font
  .\kern\fontdimen3\font
  .\kern\fontdimen3\font
}
```

```
\def\mathellipsis{\mathinner{\ldotp\ldotp\ldotp}}
\DeclareRobustCommand{\dots}{%
  \ifmmode\mathellipsis\else\textellipsis\fi
}
\let\ldots\dots
```

Part III
Testing

6 Ante-\lips

This is some testing. We need some text here as the first paragraph after the `\section`.

:. . . .
: . . . text.
: . . .)
;. . . .
; . . . text.
; . . .)
!. . . .
! . . . text.
! . . .)
t. . . .
t . . . text.
t . . .)
T. . . .
T . . . text.
T . . .)
,. . . .
, . . . text.
, . . .)
. . . and (`\par \lips and`)
and (`\par and`)
. . . (`\par\indent \lips and`)
and (`\par\indent and`)
. . . and (`\par\noindent \lips and`)
and (`\par\noindent and`)
And. . . .
(`And.\lips \par`)
And.
(`And. \par`)

7 Post-\lips

`\lips` before a tie: One two three. . . . tiedtolips (I doubt you'd want to do this.)
`\lips` before), e.g., punctuation transparent to spacefactors:
(One two three. . . .) This follows the) with interword space, to compare.
(One two three. . . .) This should follow the) with intersentence space.
(One two three. . . .) This should follow the) with intersentence space.
`\lips` before shortverb: . . . `shortverb verbatim stuff`
`\lips` before `\textsf`: . . . sansserif

`\lips` before \\: . . .
second line of a `center` environment

`\lips` before `\#`: . . . #
`\lips` before `<`: . . . ¡
`\lips` before an abbrev: . . . `Frankenstein`
`\lips` before `\cite`: . . . (University of Chicago Press 1993)
`\lips` before `$`: . . . $f = ma$
`\lips` before `&`: . . . second column of a `tabular`

References

University of Chicago Press. 1993. *The Chicago Manual of Style*. 14th ed. Chicago: University of Chicago Press.

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.