

Parallel typesetting for critical editions: the **ledpar** package*

Peter Wilson
Herries Press[†]
Maïeul Rouquette[‡]

Abstract

The **ledmac** package, which is based on the PLAIN T_EX set of **EDMAC** macros, has been used for some time for typesetting critical editions. The **ledpar** package is an extension to **ledmac** which enables texts and their critical apparatus to be typeset in parallel, either in two columns or on pairs of facing pages.

Contents

1	Introduction	3
2	The ledpar package	4
2.1	General	4
3	Parallel columns	5
4	Facing pages	5
5	Left and right texts	6
6	Numbering text lines	8
7	Verse	9
8	Implementation overview	11
9	Preliminaries	11
9.1	Messages	12
10	Sectioning commands	12

*This file (**ledpar.dtx**) has version number v0.3c, last revised 2011/06/16.

[†]herries dot press at earthlink dot net

[‡]maieul at maieul dot net

11 Line counting	15
11.1 Choosing the system of lineation	15
11.2 Line-number counters and lists	17
11.3 Reading the line-list file	18
11.4 Commands within the line-list file	19
11.5 Writing to the line-list file	27
12 Marking text for notes	30
13 Parallel environments	31
14 Paragraph decomposition and reassembly	33
14.1 Boxes, counters, <code>\pstart</code> and <code>\pend</code>	33
14.2 Processing one line	35
14.3 Line and page number computation	38
14.4 Line number printing	39
14.5 Add insertions to the vertical list	41
14.6 Penalties	41
14.7 Printing leftover notes	42
15 Footnotes	42
15.1 Outer-level footnote commands	42
15.2 Normal footnote formatting	46
16 Cross referencing	46
17 Side notes	48
18 Familiar footnotes	49
19 Verse	50
20 Naming macros	51
21 Counts and boxes for parallel texts	52
22 Fixing babel	53
23 Parallel columns	55
24 Parallel pages	58
25 The End	65
A Examples	66
A.1 Parallel column example	74
A.2 Example parallel facing pages	76
A.3 Example poetry on parallel facing pages	82

<i>List of Figures</i>	3
References	87
Index	87
Change History	95

List of Figures

1	Output from <code>villon.tex</code>	67
2	Left page output from <code>djd17nov.tex</code>	68
3	Right page output from <code>djd17nov.tex</code>	69
4	First left page output from <code>djdpoems.tex</code>	70
5	First right page output from <code>djdpoems.tex</code>	71
6	Second left page output from <code>djdpoems.tex</code>	72
7	Second right page output from <code>djdpoems.tex</code>	73

1 Introduction

The EDMAC macros [LW90] for typesetting critical editions of texts have been available for use with TeX for some years. Since EDMAC became available there had been a small but constant demand for a version of EDMAC that could be used with LaTeX. The `ledmac` package was introduced in 2003 in an attempt to satisfy that request.

Some critical editions contain texts in more than one form, such as a set of verses in one language and their translations in another. In such cases there is a desire to be able to typeset the two texts, together with any critical apparatus, in parallel. The `ledpar` package is an extension to `ledmac` that enables two texts and their apparatus to be set in parallel, either in two columns or on pairs of facing pages.

The package has to try and coerce TeX into paths it was not designed for. Use of the package, therefore, may produce some surprising results.

This manual contains a general description of how to use `ledpar` starting in section 2; the complete source code for the package, with extensive documentation (in sections 8 through 25); and an Index to the source code. As `ledpar` is an adjunct to `ledmac` I assume that you have read the `ledmac` manual. Also `ledpar` requires `ledmac` to be used, preferably at least version 0.6 (2004/12/10). You do not need to read the source code for this package in order to use it but doing so may help to answer any questions you might have. On a first reading, I suggest that you should skip anything after the general documentation in sections 2 until 8, unless you are particularly interested in the innards of `ledpar`.

2 The *ledpar* package

A file may mix *numbered* and *unnumbered* text. Numbered text is printed with marginal line numbers and can include footnotes and endnotes that are referenced to those line numbers: this is how you'll want to print the text that you're editing. Unnumbered text is not printed with line numbers, and you can't use *ledmac*'s note commands with it: this is appropriate for introductions and other material added by the editor around the edited text.

The *ledpar* package lets you typeset two *numbered* texts in parallel. This can be done either as setting the 'Leftside' and 'Rightside' texts in two columns or on facing pages. In the paired pages case footnotes are placed at the bottom of the page on which they are called out — that is, footnotes belonging to the left are set at the foot of a left (even numbered) page, and those for right texts are at the bottom of the relevant right (odd numbered) page. However, in the columnar case, all footnotes are set at the bottom left of the page on which they are called out — they are not set below the relevant column. The line numbering schemes need not be the same for the two texts.

2.1 General

ledmac essentially puts each chunk of numbered text (the text within a `\pstart` ... `\pend`) into a box and then following the `\pend` extracts the text line by line from the box to number and print it. More precisely, the text is first put into the the box as though it was being typeset as normal onto a page and any notes are stored without being typeset. Then each typeset line is extracted from the box and any notes for that line are recalled. The line, with any notes, is then output for printing, possibly with a line number attached. Effectively, all the text is typeset and then afterwards all the notes are typeset.

ledpar similarly puts the left and right chunks into boxes but can't immediately output the text after a `\pend` — it has to wait until after both the left and right texts have been collected before it can start processing. This means that several boxes are required and possibly TeX has to store a lot of text in its memory; both the number of potential boxes and memory are limited. If TeX's memory is overfilled the recourse is to reduce the amount of text stored before printing.

`\maxchunks` It is possible to have multiple chunks in the left and right texts before printing them. The macro `\maxchunks{<num>}` specifies the maximum number of chunks within the left or right texts. This is initially set as:

`\maxchunks{10}`

meaning that there can be up to 10 chunks in the left text and up to 10 chunks in the right text, requiring a total of 20 boxes. If you need more chunks then you can increase `\maxchunks`.

TeX has a limited number of boxes; if you get an error message along the lines of 'no room for a new box', then decrease the number. A chunk also requires a counter so you may get a message along the lines 'no room for a new count', which may be resolved by reducing `\maxchunks`.

On the other hand, if you get a *ledmac* error message along the lines: 'Too

many `\pstart` without printing. Some text will be lost.’ then you will have to either increase `\maxchunks` or use the parallel printing commands (`\Columns` or `\Pages`) more frequently.

When typesetting verse using `\syntax`, each line is treated as a chunk, so be warned that if you are setting parallel verses you might have to increase `\maxchunks` much more than it appears at first sight.

In general, `ledmac` is a TeX resource hog, and `ledpar` only makes things worse in this respect.

3 Parallel columns

pairs Numbered text that is to be set in columns must be within a **pairs** environment. Within the environment the text for the lefthand and righthand columns is placed within the **Leftside** and **Rightside** environments, respectively; these are described in more detail below in section 5.

\Columns The command `\Columns` typesets the texts in the previous pair of **Leftside** and **Rightside** environments. The general scheme for parallel columns looks like this:

```
\begin{pairs}
\begin{Leftside} ... \end{Leftside}
\begin{Rightside} ... \end{Rightside}
\Columns
\begin{Leftside} ... \end{Leftside}
...
\Columns
\end{pairs}
```

There is no required pagebreak before or after the columns.

\Lcolwidth The lengths `\Lcolwidth` and `\Rcolwidth` are the widths of the left and right columns, respectively. By default, these are:

```
\setlength{\Lcolwidth}{0.45\textwidth}
\setlength{\Rcolwidth}{0.45\textwidth}
```

They may be adjusted if one text tends to be ‘bulkier’ than the other.

\columnrulewidth The macro `\columnseparator` is called between each left/right pair of lines. By default it inserts a vertical rule of width `\columnrulewidth`. As this is initially defined to be 0pt the rule is invisible. For a visible rule between the columns you could try:

```
\setlength{\columnrulewidth}{0.4pt}
```

You can also modify `\columnseparator` if you want more control.

4 Facing pages

pages Numbered text that is to be set on facing pages must be within a **pages** environ-

ment. Within the environment the text for the lefthand and righthand pages is placed within the **Leftside** and **Rightside** environments, respectively.

\Pages The command **\Pages** typesets the texts in the previous pair of **Leftside** and **Rightside** environments. The general scheme for parallel pages looks like this:

```
\begin{pages}
\begin{Leftside} ... \end{Leftside}
\begin{Rightside} ... \end{Rightside}
\Pages
\begin{Leftside} ... \end{Leftside}
...
\Pages
\end{pages}
```

The **Leftside** text is set on lefthand (even numbered) pages and the **Rightside** text is set on righthand (odd numbered) pages. Each **\Pages** command starts a new even numbered page. After parallel typesetting is finished, a new page is started.

\Lcolwidth Within the **pages** environment the lengths **\Lcolwidth** and **\Rcolwidth** are the widths of the left and right pages, respectively. By default, these are set to the normal textwidth for the document, but can be changed within the environment if necessary.

\goalfraction When doing parallel pages **ledpar** has to guess where TeX is going to put pagebreaks and hopefully get there first in order to put the pair of texts on their proper pages. When it thinks that the fraction **\goalfraction** of a page has been filled, it finishes that page and starts on the other side's text. The definition is:

```
\newcommand*{\goalfraction}{0.9}
```

If you think you can get more on a page, increase this. On the other hand, if some left text overflows onto an odd numbered page or some right text onto an even page, try reducing it, for instance by:

```
\renewcommand*{\goalfraction}{0.8}
```

5 Left and right texts

Parallel texts are divided into **Leftside** and **Rightside**. The form of the contents of these two are independent of whether they will be set in columns or pages.

Leftside The left text is put within the **Leftside** environment and the right text likewise in the **Rightside** environment. The number of **Leftside** and **Rightside** environments must be the same.

\firstlinenum Within these environments you can designate the line numbering scheme(s) to be used. The **ledmac** package originally used counters for specifying the numbering scheme; now both **ledmac**¹ and the **ledpar** package use macros instead. **\linenumincrement** Following **\firstlinenum{<num>}** the first line number will be *<num>*, and following **\linenumincrement{<num>}** only every *<num>*th line will have a printed

¹when used with **ledpatch** v0.2 or greater.

number. Using these macros inside the `Leftside` and `Rightside` environments gives you independent control over the left and right numbering schemes. The `\firstsublinenum` and `\sublinenumincrement` macros correspondingly set the numbering scheme for sublines.

`\pstart` In a serial (non-parallel) mode, each numbered paragraph, or chunk, is contained between the `\pstart` and `\pend` macros, and the paragraph is output when the `\pend` macro occurs. The situation is somewhat different with parallel typesetting as the left text (contained within `\pstart` and `\pend` groups within the `Leftside` environment) has to be set in parallel with the right text (contained within its own `\pstart` and `\pend` groups within the corresponding `Rightside` environment) the `\pend` macros cannot immediately initiate any typesetting — this has to be controlled by the `\Columns` or `\Pages` macros. Several chunks may be specified within a `Leftside` or `Rightside` environment. A multi-chunk text then looks like:

```
\begin{...side}
  % \beginnumbering
  \pstart first chunk \pend
  \pstart  second chunk \pend
  ...
  \pstart  last chunk \pend
  % \endnumbering
\end{...side}
```

Numbering, via `\beginnumbering` and `\endnumbering`, may extend across several `Leftside` or `Rightside` environments. Remember, though, that the Left/Right sides are effectively independent of each other.

Generally speaking, controls like `\firstlinenum` or `\linenummargin` apply to sequential and left texts. To effect right texts only they have to be within a `Rightside` environment.

If you are using the `babel` package with different languages (via, say, `\selectlanguage`) for the left and right texts it is particularly important to select the appropriate language within the `Leftside` and `Rightside` environments. The initial language selected for the right text is the `babel` package's default. Also, it is the *last* `\selectlanguage` in a side that controls the language used in any notes for that side when they get printed. If you are using multilingual notes then it is probably safest to explicitly specify the language(s) for each note rather than relying on the language selection for the side. The right side language is also applied to the right side line numbers.

Corresponding left and right sides must have the same number of paragraph chunks — if there are four on the left there must be four on the right, even if some are empty. The start of each pair of left and right chunks are aligned horizontally on the page. The ends may come at different positions — if one chunk is shorter than the other then blank lines are output on the shorter side until the end of the longer chunk is reached.

6 Numbering text lines

`\beginnumbering` Each section of numbered text must be preceded by `\beginnumbering` and followed by `\endnumbering`, like:

```
\beginnumbering
<text>
\endnumbering
```

These have to be separately specified within `Leftside` and `Rightside` environments.

The `\beginnumbering` macro resets the line number to zero, reads an auxiliary file called `<jobname>.nn` (where `<jobname>` is the name of the main input file for this job, and `nn` is 1 for the first numbered section, 2 for the second section, and so on), and then creates a new version of this auxiliary file to collect information during this run. Separate auxiliary files are maintained for right hand texts and these are named `<jobname>.nnR`, using the ‘R’ to distinguish them from the left hand and serial (non-parallel) texts.

`\memorydump` The command `\memorydump` effectively performs an `\endnumbering` immediately followed by a `\beginnumbering` while not restarting the numbering sequence. This has the effect of clearing TeX’s memory of previous texts and any associated notes, allowing longer apparent streams of parallel texts. The command should be applied to both left and right texts, and after making sure that all previous notes have been output. For example, along the lines of:

```
\begin{Leftside}
  \beginnumbering
  ...
\end{Leftside}
\begin{Rightside}
  \beginnumbering
  ...
\end{Rightside}
\Pages
\begin{Leftside}
  \memorydump
  ...
\end{Leftside}
\begin{Rightside}
  \memorydump
  ...
```

`\Rlineflag` The value of `\Rlineflag` is appended to the line numbers of the right texts. Its default definition is:

```
\newcommand*{\Rlineflag}{R}
```

This may be useful for parallel columns but for parallel pages it might be more appropriate to redefine it as:

```
\renewcommand*{\Rlineflag}{.}
```

`\printlinesR` The `\printlines` macro is ordinarily used to print the line number referred to by `\ledsavedprintlines`.

ences for critical footnotes. For footnotes from right side texts a special version is supplied, called `\printlinesR`, which incorporates `\Rlineflag`. (The macro `\ledsavedprintlines` is a copy of the original `\printlines`, just in case ...). As provided, the package makes no use of `\printlinesR` but you may find it useful. For example, if you only use the B footnote series in righthand texts then you may wish to flag any line numbers in those footnotes with the value of `\Rlineflag`. You could do this by putting the following code in your preamble:

```
\let\oldBfootfmt\Bfootfmt
\renewcommand{\Bfootfmt}[3]{%
  \let\printlines\printlinesR
  \oldBfootfmt{#1}{#2}{#3}}
```

7 Verse

If you are typesetting verse with `ledmac` you can use the `\stanza` construct, and you can also use this in right or left parallel texts. In this case each verse line is a chunk which has two implications. (1) you can unexpectedly exceed the `\maxchunks` limit or the overall limit on the number of boxes, and (2) left and right verse lines are matched, which may not be desirable if one side requires more print lines for verse lines than the other does.

`astanza` `ledpar` provides an `astanza` environment which you can use instead of `\stanza` (simply replace `\stanza` by `\begin{astanza}` and add `\end{astanza}` after the ending `\&`). Within the `astanza` environment each verse line is treated as a paragraph, so there must be no blank lines in the environment otherwise there will be some extraneous vertical spacing.

If you get an error message along the lines of ‘Missing number, treated as zero `\sza000`’ it is because you have forgotten to use `\setstanzaindents` to set the stanza indents.

`\skipnumbering` The command `\skipnumbering` when inserted in a line of parallel text causes the numbering of that particular line to be skipped. This can useful if you are putting some kind of marker (even if it is only a blank line) between stanzas. Remember, parallel texts must be numbered and this provides a way to slip in an ‘unnumbered’ line.

The `astanza` environment forms a chunk but you may want to have more than one stanza within the chunk. Here are a couple of ways of doing that with a blank line between each internal stanza, and with each stanza numbered. First some preliminary definitions:

```
\newcommand*{\stanzanum}[2][\stanzaindentbase]{%
  \hskip -#1\llap{\textbf{#2}}\hskip #1\ignorespaces}
\newcommand{\interstanza}{\par\mbox{}\skipnumbering}
```

And now for two stanzas in one. In this first example the line numbering repeats for each stanza.

```

\setstanzaindents{1,0,1,0,1,0,1,0,1,0,1}
\begin{pairs}
\begin{Leftside}
\firstlinenum{2}
\linenumincrement{1}
\beginnumbering
\begin{astanza}
\stanzanum{1} First in first stanza &
                Second in first stanza &
                Second in first stanza &
                Third in first stanza &
                Fourth in first stanza &

\interstanza
\setline{2}\stanzanum{2} First in second stanza &
                Second in second stanza &
                Second in second stanza &
                Third in second stanza &
                Fourth in second stanza \&

\end{astanza}
...

```

And here is a slightly different way of doing the same thing, but with the line numbering being continuous.

```

\setstanzaindents{1,0,1,0,1,0,0,1,0,1,0,1}
\begin{pairs}
\begin{Leftside}
\firstlinenum{2}
\linenumincrement{1}
\beginnumbering
\begin{astanza}
\stanzanum{1} First in first stanza &
                Second in first stanza &
                Second in first stanza &
                Third in first stanza &
                Fourth in first stanza &

\strut &
\stanzanum{2}\advanceline{-1} First in second stanza &
                Second in second stanza &
                Second in second stanza &
                Third in second stanza &
                Fourth in second stanza \&

\end{astanza}
...

```

8 Implementation overview

TeX is designed to process a single stream of text, which may include footnotes, tables, and so on. It just keeps converting its input into a stream typeset pages. It was not designed for typesetting two texts in parallel, where it has to alternate from one to the other. Further, TeX essentially processes its input one paragraph at a time — it is very difficult to get at the ‘internals’ of a paragraph such as the individual lines in case you want to number them or put some mark at the start or end of the lines.

`ledmac` solves the problem of line numbering by putting the paragraph in type-set form into a box, and then extracting the lines one by one from the box for TeX to put them onto the page with the appropriate page breaks. Most of the `ledmac` code is concerned with handling this box and its contents.

`ledpar`’s solution to the problem of parallel texts is to put the two texts into separate boxes, and then appropriately extract the pairs of lines from the boxes. This involves duplicating much of the original box code for an extra right text box. The other, smaller, part of the code is concerned with coordinating the line extractions from the boxes.

The package code is presented in roughly in the same order as in `ledmac`.

9 Preliminaries

Announce the name and version of the package, which is targetted for LaTeX2e. The package also requires the `ledmac` package, preferably at least version 0.6 (2004/12/10).

```

1 (*code)
2 \NeedsTeXFormat{LaTeX2e}
3 \ProvidesPackage{ledpar}[2011/06/16 v0.3c ledmac extension for parallel texts]
4
```

As noted above, much of the code is a duplication of the original `ledmac` code to handle the extra box(es) for the right hand side text, and sometimes for the left hand side as well. In order to distinguish I use ‘R’ or ‘L’ in the names of macros for the right and left code. The specifics of ‘L’ and ‘R’ are normally hidden from the user by letting the `Leftside` and `Rightside` environments set things up appropriately.

```

\ifl@dpairing \ifl@dpairing is set TRUE if we are processing parallel texts and \ifl@dpaging
\ifl@dpaging is also set TRUE if we are doing parallel pages. \ifledRcol is set TRUE if we
\ifledRcol are doing the right hand text. \ifl@dpairing is defined in ledmac.

5 \l@dpairingfalse
6 \newif\ifl@dpaging
7 \l@dpagingfalse
8 \newif\ifledRcol
9 \ledRcolfalse
```

`\Lcolwidth` The widths of the left and right parallel columns (or pages).
`\Rcolwidth` 10 `\newdimen\Lcolwidth`
 11 `\Lcolwidth=0.45\textwidth`
 12 `\newdimen\Rcolwidth`
 13 `\Rcolwidth=0.45\textwidth`
 14

9.1 Messages

All the error and warning messages are collected here as macros.

```
\led@err@TooManyPstarts
15 \newcommand*\led@err@TooManyPstarts}{%
16 \ledmac@error{Too many \string\pstart\space without printing.
17 \qquad Some text will be lost}\@ehc}}

\led@err@BadLeftRightPstarts
18 \newcommand*\led@err@BadLeftRightPstarts}[2]{%
19 \ledmac@error{The numbers of left (#1) and right (#2)
20 \string\pstart s do not match}\@ehc}}

\led@err@LeftOnRightPage
\led@err@RightOnLeftPage
21 \newcommand*\led@err@LeftOnRightPage}{%
22 \ledmac@error{The left page has ended on a right page}\@ehc}}
23 \newcommand*\led@err@RightOnLeftPage}{%
24 \ledmac@error{The right page has ended on a left page}\@ehc}}
```

10 Sectioning commands

`\section@numR` This is the right side equivalent of `\section@num`.
 Each section will read and write an associated ‘line-list file’, containing information used to do the numbering. Normally the file will be called `<jobname>.nn`, where `nn` is the section number. However, for right side texts the file is called `<jobname>.nnR`. The `\extensionchars` applies to the right side files just as it does to the normal files.

```
25 \newcount\section@numR
26 \section@numR=\z@
```

`\ifnumberingR` The `\ifnumberingR` flag is set to `true` if we’re within a right text numbered section.

```
27 \newif\ifnumberingR
```

`\ifpst@rtedL` `\ifpst@rtedL` is set `FALSE` at the start of left side numbering, and similarly for `\ifpst@rtedR`. `\ifpst@rtedL` is defined in `ledmac`.

```
28 \pst@rtedLfalse
29 \newif\ifpst@rtedR
30 \pst@rtedRfalse
31
```

`\beginnumbering` For parallel processing the original `\beginnumbering` is extended to zero `\l@dnumpstartsL` — the number of chunks to be processed. It also sets `\ifpst@rtedL` to FALSE.

```

32 \providecommand*\beginnumbering{%
33   \ifnumbering
34     \led@err@NumberingStarted
35   \endnumbering
36 \fi
37 \global\l@dnumpstartsL \z@
38 \global\pst@rtedLfalse
39 \global\numberingtrue
40 \global\advance\section@num \@ne
41 \initnumbering@reg
42 \message{Section \the\section@num}%
43 \line@list@stuff{\jobname.\extensionchars\the\section@num}%
44 \l@dend@stuff}

```

`\beginnumberingR` This is the right text equivalent of `\beginnumbering`, and begins a section of numbered text.

```

45 \newcommand*\beginnumberingR{%
46   \ifnumberingR
47     \led@err@NumberingStarted
48   \endnumberingR
49 \fi
50 \global\l@dnumpstartsR \z@
51 \global\pst@rtedRfalse
52 \global\numberingRtrue
53 \global\advance\section@numR \@ne
54 \global\absline@numR \z@
55 \global\line@numR \z@
56 \global\subline@numR \z@
57 \global\@lock \z@
58 \global\sub@lock \z@
59 \global\sublines@false
60 \global\let\next@page@numR=\relax
61 \global\let\sub@change=\relax
62 \message{Section \the\section@numR R }%
63 \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
64 \l@dend@stuff}
65

```

`\endnumbering` This is the left text version of the regular `\endnumbering` and must follow the last text for a left text numbered section. It sets `\ifpst@rtedL` to FALSE. It is fully defined in `ledmac`.

`\endnumberingR` This is the right text equivalent of `\endnumbering` and must follow the last text for a right text numbered section.

```

66 \def\endnumberingR{%
67   \ifnumberingR
68     \global\numberingRfalse

```

```

69 \normal@pars
70 \ifl@dpairing
71 \global\pst@rtedRfalse
72 \else
73 \ifx\insertlines@listR\empty\else
74 \global\noteschanged@true
75 \fi
76 \ifx\line@listR\empty\else
77 \global\noteschanged@true
78 \fi
79 \fi
80 \ifnoteschanged@
81 \led@mess@NotesChanged
82 \fi
83 \else
84 \led@err@NumberingNotStarted
85 \fi}
86

\pausenumberingR These are the right text equivalents of \pausenumbering and \resumenumbering.
\resumenumberingR
87 \newcommand*{\pausenumberingR}{%
88 \endnumberingR\global\numberingRtrue}
89 \newcommand*{\resumenumberingR}{%
90 \ifnumberingR
91 \global\pst@rtedRtrue
92 \global\advance\section@numR \@ne
93 \led@mess@SectionContinued{\the\section@numR R}%
94 \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
95 \l@dend@stuff
96 \else
97 \led@err@numberingShouldHaveStarted
98 \endnumberingR
99 \beginnumberingR
100 \fi}
101

\memorydumpL \memorydump is a shorthand for \pausenumbering\resumenumbering. This will
\memorydumpR clear the memorised stuff for the previous chunks while keeping the numbering
going.
102 \newcommand*{\memorydumpL}{%
103 \endnumbering
104 \numberingtrue
105 \global\pst@rtedLtrue
106 \global\advance\section@num \@ne
107 \led@mess@SectionContinued{\the\section@num}%
108 \line@list@stuff{\jobname.\extensionchars\the\section@num}%
109 \l@dend@stuff}
110 \newcommand*{\memorydumpR}{%
111 \endnumberingR

```

```

112 \numberingRtrue
113 \global\pst@rtedRtrue
114 \global\advance\section@numR \@one
115 \led@mess@SectionContinued{\the\section@numR R}%
116 \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
117 \l@dend@stuff}
118

```

11 Line counting

11.1 Choosing the system of lineation

Sometimes you want line numbers that start at 1 at the top of each page; other times you want line numbers that start at 1 at the start of each section and increase regardless of page breaks. `ledpar` lets you choose different schemes for the left and right texts.

`\ifbypage@R` The `\ifbypage@R` flag specifies the current lineation system for right texts: `false` for line-of-section, `true` for line-of-page. `ledpar` will use the line-of-section system unless instructed otherwise.

```

119 \newif\ifbypage@R
120 \bypage@Rfalse

```

`\lineationR` `\lineationR{<word>}` is the macro used to select the lineation system for right texts. Its argument is a string: either `page` or `section`.

```

121 \newcommand*{\lineationR}[1]{%
122 \ifnumberingR
123 \led@err@LineationInNumbered
124 \else
125 \def\@tempa{#1}\def\@tempb{page}%
126 \ifx\@tempa\@tempb
127 \global\bypage@Rtrue
128 \else
129 \def\@tempb{section}%
130 \ifx\@tempa\@tempb
131 \global\bypage@Rfalse
132 \else
133 \led@warn@BadLineation
134 \fi
135 \fi
136 \fi}}
137

```

`\linenummargin` You call `\linenummargin{<word>}` to specify which margin you want your right text's line numbers in; it takes one argument, a string. You can put the line numbers in the same margin on every page using `left` or `right`; or you can use `inner` or `outer` to get them in the inner or outer margins. You can change this

within a numbered section, but the change may not take effect just when you'd like; if it's done between paragraphs nothing surprising should happen.

For right texts the selection is recorded in the count `\line@marginR`, otherwise in the count `\line@margin`: 0 for left, 1 for right, 2 for outer, and 3 for inner.

```

138 \newcount\line@marginR
139 \renewcommand*{\linenummargin}[1]{%
140   \l@getline@margin{#1}%
141   \ifnum\l@dtmpcntb>\m@ne
142     \ifledRcol
143       \global\line@marginR=\l@dtmpcntb
144     \else
145       \global\line@margin=\l@dtmpcntb
146     \fi
147   \fi}}

```

By default put right text numbers at the right.

```

148 \line@marginR=\@ne
149

```

`\c@firstlinenumR` The following counters tell `ledmac` which right text lines should be printed with line numbers. `firstlinenum` is the number of the first line in each section that gets a number; `linenumincrement` is the difference between successive numbered lines. The initial values of these counters produce labels on lines 5, 10, 15, etc. `linenumincrement` must be at least 1.

```

150 \newcounter{firstlinenumR}
151 \setcounter{firstlinenumR}{5}
152 \newcounter{linenumincrementR}
153 \setcounter{linenumincrementR}{5}

```

`\c@firstsublinenumR` The following parameters are just like `firstlinenumR` and `linenumincrementR`,
`\c@sublinenumincrementR` but for sub-line numbers. `sublinenumincrementR` must be at least 1.

```

154 \newcounter{firstsublinenumR}
155 \setcounter{firstsublinenumR}{5}
156 \newcounter{sublinenumincrementR}
157 \setcounter{sublinenumincrementR}{5}
158

```

`\firstlinenum` These are the user's macros for changing (sub) line numbers. They are defined in `ledmac v0.7`, but just in case I have started by `\provideing` them.

```

\firstsublinenum 159 \providecommand*{\firstlinenum}{}
\linenumincrement 160 \providecommand*{\linenumincrement}{}
\firstsublinenum 161 \providecommand*{\firstsublinenum}{}
\sublinenumincrement 162 \providecommand*{\sublinenumincrement}{}
163 \renewcommand*{\firstlinenum}[1]{%
164   \ifledRcol \setcounter{firstlinenumR}{#1}%
165   \else      \setcounter{firstlinenumR}{#1}%
166   \fi}
167 \renewcommand*{\linenumincrement}[1]{%

```



```

168 \ifledRcol \setcounter{linenumincrementR}{#1}%
169 \else      \setcounter{linenumincrement}{#1}%
170 \fi}
171 \renewcommand*{\firstsublinenum}[1]{%
172 \ifledRcol \setcounter{firstsublinenumR}{#1}%
173 \else      \setcounter{firstsublinenum}{#1}%
174 \fi}
175 \renewcommand*{\sublinenumincrement}[1]{%
176 \ifledRcol \setcounter{sublinenumincrementR}{#1}%
177 \else      \setcounter{sublinenumincrement}{#1}%
178 \fi}
179

```

`\Rlineflag` This is appended to the line numbers of right text.

```

180 \newcommand*{\Rlineflag}{R}
181

```

`\linenumrepR` `\linenumrepR{<ctr>}` typesets the right line number `<ctr>`, and similarly `\sublinenumrepR` for subline numbers.

```

182 \newcommand*{\linenumrepR}[1]{\@arabic{#1}}
183 \newcommand*{\sublinenumrepR}[1]{\@arabic{#1}}
184

```

`\leftlinenumR` `\leftlinenumR` and `\rightlinenumR` are the macros that are called to print the right text's marginal line numbers. Much of the code for these is common and is maintained in `\l@dlinenumR`.

```

185 \newcommand*{\leftlinenumR}{%
186 \l@dlinenumR
187 \kern\linenumsep}
188 \newcommand*{\rightlinenumR}{%
189 \kern\linenumsep
190 \l@dlinenumR}
191 \newcommand*{\l@dlinenumR}{%
192 \numlabfont\linenumrepR{\line@numR}\Rlineflag%
193 \ifsublines@
194 \ifnum\subline@num>\z@
195 \unskip\fullstop\sublinenumrepR{\subline@numR}%
196 \fi
197 \fi}
198

```

11.2 Line-number counters and lists

We need another set of counters and lists for the right text, corresponding to those in `ledmac` for `regualr` or left text.

`\line@numR` The count `\line@numR` stores the line number that's used in the right text's marginal line numbering and in notes. The count `\subline@numR` stores a sub-line number that qualifies `\line@numR`. The count `\absline@numR` stores the absolute

number of lines since the start of the right text section: that is, the number we've actually printed, no matter what numbers we attached to them.

```
199 \newcount\line@numR
200 \newcount\subline@numR
201 \newcount\absline@numR
202
```

```
\line@listR Now we can define the list macros that will be created from the line-list file. They
\insertlines@listR are directly analagous to the left text ones. The full list of action codes and their
\actionlines@listR meanings is given in the ledmac manual.
\actions@listR Here are the commands to create these lists:
```

```
203 \list@create{\line@listR}
204 \list@create{\insertlines@listR}
205 \list@create{\actionlines@listR}
206 \list@create{\actions@listR}
207
```

```
\linesinpar@listL In order to synchronise left and right chunks in parallel processing we need to know
\linesinpar@listR how many lines are in each left and right text chunk, and the maximum of these
\maxlinesinpar@list for each pair of chunks.
```

```
208 \list@create{\linesinpar@listL}
209 \list@create{\linesinpar@listR}
210 \list@create{\maxlinesinpar@list}
211
```

```
\page@numR The right text page number.
```

```
212 \newcount\page@numR
213
```

11.3 Reading the line-list file

```
\read@linelist \read@linelist{<file>} is the control sequence that's called by \beginnumbering
(via \line@list@stuff) to open and process a line-list file; its argument is the
name of the file.
```

```
214 \renewcommand*{\read@linelist}[1]{%
```

We do do different things depending whether or not we are processing right text

```
215 \ifledRcol
216 \list@clear{\line@listR}%
217 \list@clear{\insertlines@listR}%
218 \list@clear{\actionlines@listR}%
219 \list@clear{\actions@listR}%
220 \list@clear{\linesinpar@listR}%
221 \list@clear{\linesonpage@listR}
222 \else
223 \list@clearing@reg
224 \list@clear{\linesinpar@listL}%
225 \list@clear{\linesonpage@listL}%
226 \fi
```

Make sure that the `\maxlinesinpar@list` is empty (otherwise things will be thrown out of kilter if there is any old stuff still hanging in there).

```
227 \list@clear{\maxlinesinpar@list}
```

Now get the file and interpret it.

```
228 \get@linelistfile{#1}%
```

```
229 \endgroup
```

When the reading is done, we're all through with the line-list file. All the information we needed from it will now be encoded in our list macros. Finally, we initialize the `\next@actionline` and `\next@action` macros, which specify where and what the next action to be taken is.

```
230 \ifledRcol
231   \global\page@numR=\m@ne
232   \ifx\actionlines@listR\empty
233     \gdef\next@actionlineR{1000000}%
234   \else
235     \gl@p\actionlines@listR\to\next@actionlineR
236     \gl@p\actions@listR\to\next@actionR
237   \fi
238 \else
239   \global\page@num=\m@ne
240   \ifx\actionlines@list\empty
241     \gdef\next@actionline{1000000}%
242   \else
243     \gl@p\actionlines@list\to\next@actionline
244     \gl@p\actions@list\to\next@action
245   \fi
246 \fi}
247
```

This version of `\read@linelist` creates list macros containing data for the entire section, so they could get rather large. The `\memorydump` macro is available if you run into macro memory limitations.

11.4 Commands within the line-list file

This section defines the commands that can appear within a line-list file, except for `\@lab` which is in a later section among the cross-referencing commands it is associated with.

The macros with `action` in their names contain all the code that modifies the action-code list.

\@l `\@l` does everything related to the start of a new line of numbered text. Exactly what it does depends on whether right text is being processed.

```
248 \renewcommand{\@l}[2]{%
```

```
249   \fix@page{#1}%
```

```
250   \ifledRcol
```

```
251     \ifx\l@dchset@num\relax \else
```

```

252     \advance\absline@numR \@ne
253     \set@line@action
254     \let\l@dchset@num=\relax
255     \advance\absline@numR \m@ne
256     \advance\line@numR \m@ne% % do we need this??
257     \fi
258     \advance\absline@numR \@ne

```

Increment the absolute line-number, and perform deferred actions relating to page starts and sub-lines.

```

259     \ifx\next@page@numR\relax \else
260     \page@action
261     \let\next@page@numR=\relax
262     \fi
263     \ifx\sub@change\relax \else
264     \ifnum\sub@change>\z@
265     \sublines@true
266     \else
267     \sublines@false
268     \fi
269     \sub@action
270     \let\sub@change=\relax
271     \fi

```

Fix the lock counters, if necessary. A value of 1 is advanced to 2; 3 advances to 0; other values are unchanged.

```

272     \ifcase\@lock
273     \or
274     \@lock \tw@
275     \or \or
276     \@lock \z@
277     \fi
278     \ifcase\sub@lock
279     \or
280     \sub@lock \tw@
281     \or \or
282     \sub@lock \z@
283     \fi

```

Now advance the visible line number, unless it's been locked.

```

284     \ifsublines@
285     \ifnum\sub@lock<\tw@
286     \advance\subline@numR \@ne
287     \fi
288     \else
289     \ifnum\@lock<\tw@
290     \advance\line@numR \@ne \subline@numR \z@
291     \fi
292     \fi
293     \else

```

And when we are not in right text

```
294 \l@reg
295 \fi}
296
```

\last@page@numR We have to adjust \fix@page to handle parallel texts.

```
\fix@page
297 \newcount\last@page@numR
298 \last@page@numR=-10000
299 \renewcommand*{\fix@page}[1]{%
300 \ifledRcol
301 \ifnum #1=\last@page@numR
302 \else
303 \ifbypage@R
304 \line@numR \z@ \subline@numR \z@
305 \fi
306 \page@numR=#1\relax
307 \last@page@numR=#1\relax
308 \def\next@page@numR{#1}%
309 \fi
310 \else
311 \ifnum #1=\last@page@num
312 \else
313 \ifbypage@
314 \line@num \z@ \subline@num \z@
315 \fi
316 \page@num=#1\relax
317 \last@page@num=#1\relax
318 \def\next@page@num{#1}%
319 \fi
320 \fi}
321
```

\@adv The \@adv{<num>} macro advances the current visible line number by the amount specified as its argument. This is used to implement \advanceline.

```
322 \renewcommand*{\@adv}[1]{%
323 \ifsublines@
324 \ifledRcol
325 \advance\subline@numR by #1\relax
326 \ifnum\subline@numR<\z@
327 \led@warn@BadAdvancelineSubline
328 \subline@numR \z@
329 \fi
330 \else
331 \advance\subline@num by #1\relax
332 \ifnum\subline@num<\z@
333 \led@warn@BadAdvancelineSubline
334 \subline@num \z@
335 \fi
336 \fi
```

```

337 \else
338   \ifledRcol
339     \advance\line@numR by #1\relax
340     \ifnum\line@numR<\z@
341       \led@warn@BadAdvancelineLine
342       \line@numR \z@
343     \fi
344   \else
345     \advance\line@num by #1\relax
346     \ifnum\line@num<\z@
347       \led@warn@BadAdvancelineLine
348       \line@num \z@
349     \fi
350   \fi
351 \fi
352 \set@line@action}
353

```

`\@set` The `\@set{<num>}` macro sets the current visible line number to the value specified as its argument. This is used to implement `\setline`.

```

354 \renewcommand*{\@set}[1]{%
355   \ifledRcol
356     \ifsublines@
357       \subline@numR=#1\relax
358     \else
359       \line@numR=#1\relax
360     \fi
361   \set@line@action
362 \else
363   \ifsublines@
364     \subline@num=#1\relax
365   \else
366     \line@num=#1\relax
367   \fi
368   \set@line@action
369 \fi}
370

```

`\l@d@set` The `\l@d@set{<num>}` macro sets the line number for the next `\pstart...` to the value specified as its argument. This is used to implement `\setlinenum`.

`\l@dchset@num` is a flag to the `\@l` macro. If it is not `\relax` then a linenum change is to be done.

```

371 \renewcommand*{\l@d@set}[1]{%
372   \ifledRcol
373     \line@numR=#1\relax
374     \advance\line@numR \@ne
375     \def\l@dchset@num{#1}
376   \else
377     \line@num=#1\relax

```

```

378 \advance\line@num \@ne
379 \def\l@dchset@num{#1}
380 \fi}
381 \let\l@dchset@num\relax
382

```

\page@action \page@action adds an entry to the action-code list to change the page number.

```

383 \renewcommand*{\page@action}{%
384 \ifledRcol
385 \xright@appenditem{\the\absline@numR}\to\actionlines@listR
386 \xright@appenditem{\next@page@numR}\to\actions@listR
387 \else
388 \xright@appenditem{\the\absline@num}\to\actionlines@list
389 \xright@appenditem{\next@page@num}\to\actions@list
390 \fi}

```

\set@line@action \set@line@action adds an entry to the action-code list to change the visible line number.

```

391 \renewcommand*{\set@line@action}{%
392 \ifledRcol
393 \xright@appenditem{\the\absline@numR}\to\actionlines@listR
394 \ifsublines@
395 \@l@tempcnta=-\subline@numR
396 \else
397 \@l@tempcnta=-\line@numR
398 \fi
399 \advance\@l@tempcnta by -5000\relax
400 \xright@appenditem{\the\@l@tempcnta}\to\actions@listR
401 \else
402 \xright@appenditem{\the\absline@num}\to\actionlines@list
403 \ifsublines@
404 \@l@tempcnta=-\subline@num
405 \else
406 \@l@tempcnta=-\line@num
407 \fi
408 \advance\@l@tempcnta by -5000\relax
409 \xright@appenditem{\the\@l@tempcnta}\to\actions@list
410 \fi}
411

```

\sub@action \sub@action adds an entry to the action-code list to turn sub-lineation on or off, according to the current value of the \ifsublines@ flag.

```

412 \renewcommand*{\sub@action}{%
413 \ifledRcol
414 \xright@appenditem{\the\absline@numR}\to\actionlines@listR
415 \ifsublines@
416 \xright@appenditem{-1001}\to\actions@listR
417 \else
418 \xright@appenditem{-1002}\to\actions@listR

```

```

419 \fi
420 \else
421 \xright@appenditem{\the\absline@num}\to\actionlines@list
422 \ifsublines@
423 \xright@appenditem{-1001}\to\actions@list
424 \else
425 \xright@appenditem{-1002}\to\actions@list
426 \fi
427 \fi}
428

```

`\do@lockon` `\lock@on` adds an entry to the action-code list to turn line number locking on. The current setting of the sub-lineation flag tells us whether this applies to line numbers or sub-line numbers.

```

429 \renewcommand*{\do@lockon}{%
430 \ifx\next\lock@off
431 \global\let\lock@off=\skip@lockoff
432 \else
433 \ifl@edRcol
434 \xright@appenditem{\the\absline@numR}\to\actionlines@listR
435 \ifsublines@
436 \xright@appenditem{-1005}\to\actions@listR
437 \ifcase\sub@lock
438 \sub@lock \@ne
439 \else
440 \sub@lock \z@
441 \fi
442 \else
443 \xright@appenditem{-1003}\to\actions@listR
444 \ifcase\@lock
445 \@lock \@ne
446 \else
447 \@lock \z@
448 \fi
449 \fi
450 \else
451 \xright@appenditem{\the\absline@num}\to\actionlines@list
452 \ifsublines@
453 \xright@appenditem{-1005}\to\actions@list
454 \ifcase\sub@lock
455 \sub@lock \@ne
456 \else
457 \sub@lock \z@
458 \fi
459 \else
460 \xright@appenditem{-1003}\to\actions@list
461 \ifcase\@lock
462 \@lock \@ne
463 \else
464 \@lock \z@

```



```

465         \fi
466     \fi
467 \fi
468 \fi}

```

`\lock@off` `\lock@off` adds an entry to the action-code list to turn line number locking off.

```

\do@lockoff 469 \renewcommand*{\do@lockoff}{%
\skip@lockoff 470 \ifledRcol
471     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
472     \ifsublines@
473         \xright@appenditem{-1006}\to\actions@listR
474         \ifnum\sub@lock=\tw@
475             \sub@lock \thr@@
476         \else
477             \sub@lock \z@
478         \fi
479     \else
480         \xright@appenditem{-1004}\to\actions@listR
481         \ifnum\@lock=\tw@
482             \@lock \thr@@
483         \else
484             \@lock \z@
485         \fi
486     \fi
487 \else
488     \xright@appenditem{\the\absline@num}\to\actionlines@list
489     \ifsublines@
490         \xright@appenditem{-1006}\to\actions@list
491         \ifnum\sub@lock=\tw@
492             \sub@lock \thr@@
493         \else
494             \sub@lock \z@
495         \fi
496     \else
497         \xright@appenditem{-1004}\to\actions@list
498         \ifnum\@lock=\tw@
499             \@lock \thr@@
500         \else
501             \@lock \z@
502         \fi
503     \fi
504 \fi}
505 \global\let\lock@off=\do@lockoff
506

```

`\n@num` This macro implements the `\skipnumbering` command. It uses a new action code, namely 1007.

```

507 \providecommand*{\n@num}{}
508 \renewcommand*{\n@num}{%
509     \ifledRcol

```

```

510    \xright@appenditem{\the\absline@numR}\to\actionlines@listR
511    \xright@appenditem{-1007}\to\actions@listR
512  \else
513    \n@num@reg
514  \fi}
515

```

`\@ref` `\@ref` marks the start of a passage, for creation of a footnote reference. It takes `\insert@countR` two arguments:

- #1, the number of entries to add to `\insertlines@list` for this reference. This value for right text, here and within `\edtext`, which computes it and writes it to the line-list file, will be stored in the count `\insert@countR`.

```

516    \newcount\insert@countR

```

- #2, a sequence of other line-list-file commands, executed to determine the ending line-number. (This may also include other `\@ref` commands, corresponding to uses of `\edtext` within the first argument of another instance of `\edtext`.)

The first thing `\@ref` itself does is to add the specified number of items to the `\insertlines@list` list.

```

517 \renewcommand*{\@ref}[2]{%
518   \ifledRcol
519     \global\insert@countR=#1\relax
520     \loop\ifnum\insert@countR>\z@
521       \xright@appenditem{\the\absline@numR}\to\insertlines@listR
522       \global\advance\insert@countR \m@ne
523     \repeat

```

Next, process the second argument to determine the page and line numbers for the end of this lemma. We temporarily equate `\@ref` to a different macro that just executes its argument, so that nested `\@ref` commands are just skipped this time. Some other macros need to be temporarily redefined to suppress their action.

```

524   \begingroup
525     \let\@ref=\dummy@ref
526     \let\page@action=\relax
527     \let\sub@action=\relax
528     \let\set@line@action=\relax
529     \let\@lab=\relax
530     #2
531     \global\endpage@num=\page@numR
532     \global\endline@num=\line@numR
533     \global\endsubline@num=\subline@numR
534   \endgroup

```

Now store all the information about the location of the lemma's start and end in `\line@list`.

```

535    \xright@appenditem%

```

```

536      {\the\page@numR|\the\line@numR|}%
537      \ifsublines@ \the\subline@numR \else 0\fi|}%
538      \the\endpage@num|\the\endline@num|}%
539      \ifsublines@ \the\endsubline@num \else 0\fi}\to\line@listR

```

Finally, execute the second argument of `\@ref` again, to perform for real all the commands within it.

```

540      #2
541      \else

```

And when not in right text

```

542      \@ref@reg{#1}{#2}%
543      \fi}

```

`\@pend` `\@pend{<num>}` adds its argument to the `\linesinpar@listL` list, and analogously `\@pendR` for `\@pendR`. We start off with a `\providecommand` just in case an older version of `ledmac` is being used which does not define these macros.

```

544 \providecommand*{\@pend}[1]{}
545 \renewcommand*{\@pend}[1]{%
546   \xright@appenditem{#1}\to\linesinpar@listL}
547 \providecommand*{\@pendR}[1]{}
548 \renewcommand*{\@pendR}[1]{%
549   \xright@appenditem{#1}\to\linesinpar@listR}
550

```

`\@lopL` `\@lopL{<num>}` adds its argument to the `\linesonpage@listL` list, and analogously `\@lopR` for `\@lopR`. We start off with a `\providecommand` just in case an older version of `ledmac` is being used which does not define these macros.

```

551 \providecommand*{\@lopL}[1]{}
552 \renewcommand*{\@lopL}[1]{%
553   \xright@appenditem{#1}\to\linesonpage@listL}
554 \providecommand*{\@lopR}[1]{}
555 \renewcommand*{\@lopR}[1]{%
556   \xright@appenditem{#1}\to\linesonpage@listR}
557

```

11.5 Writing to the line-list file

We've now defined all the counters, lists, and commands involved in reading the line-list file at the start of a section. Now we'll cover the commands that `ledmac` uses within the text of a section to write commands out to the line-list.

`\linenum@outR` The file for right texts will be opened on output stream `\linenum@outR`.

```

558 \newwrite\linenum@outR

```

`\iffirst@linenum@out@R` Once any file is opened on this stream, we keep it open forever, or else switch to another file that we keep open.

```

\first@linenum@out@Rtrue
\first@linenum@out@Rfalse
559 \newif\iffirst@linenum@out@R
560 \first@linenum@out@Rtrue

```

`\line@list@stuffR` This is the right text version of the `\line@list@stuff{<file>}` macro. It is called by `\beginnumberingR` and performs all the line-list operations needed at the start of a section. Its argument is the name of the line-list file.

```

561 \newcommand*{\line@list@stuffR}[1]{%
562   \read@linelist{#1}%
563   \iffirst@linenum@out@R
564     \immediate\closeout\linenum@outR
565     \global\first@linenum@out@Rfalse
566     \immediate\openout\linenum@outR=#1
567   \else
568     \closeout\linenum@outR
569     \openout\linenum@outR=#1
570   \fi}
571

```

`\new@lineR` The `\new@lineR` macro sends the `\@l` command to the right text line-list file, to mark the start of a new text line.

```

572 \newcommand*{\new@lineR}{%
573   \write\linenum@outR{\string\@l[\the\c@page][\thepage]}}

```

`\flag@start` We enclose a lemma marked by `\edtext` in `\flag@start` and `\flag@end`: these
`\flag@end` send the `\@ref` command to the line-list file.

```

574 \renewcommand*{\flag@start}{%
575   \ifledRcol
576     \edef\next{\write\linenum@outR{%
577       \string\@ref[\the\insert@countR][ ]}%
578     \next
579   \else
580     \edef\next{\write\linenum@outR{%
581       \string\@ref[\the\insert@countR][ ]}%
582     \next
583   \fi}
584 \renewcommand*{\flag@end}{%
585   \ifledRcol
586     \write\linenum@outR{[]}%
587   \else
588     \write\linenum@outR{[]}%
589   \fi}

```

`\startsub` `\startsub` and `\endsub` turn sub-lineation on and off, by writing appropriate
`\endsub` instructions to the line-list file.

```

590 \renewcommand*{\startsub}{\dimen0\lastskip
591   \ifdim\dimen0>0pt \unskip \fi
592   \ifledRcol \write\linenum@outR{\string\sub@on}%
593   \else      \write\linenum@outR{\string\sub@on}%
594   \fi
595   \ifdim\dimen0>0pt \hskip\dimen0 \fi}
596 \def\endsub{\dimen0\lastskip
597   \ifdim\dimen0>0pt \unskip \fi

```

```

598 \ifledRcol \write\linenum@outR{\string\sub@off}%
599 \else      \write\linenum@out{\string\sub@off}%
600 \fi
601 \ifdim\dimen0>0pt \hskip\dimen0 \fi}
602

```

\advanceline You can use `\advanceline{<num>}` in running text to advance the current visible line-number by a specified value, positive or negative.

```

603 \renewcommand*{\advanceline}[1]{%
604 \ifledRcol \write\linenum@outR{\string\@adv[#1]}%
605 \else      \write\linenum@out{\string\@adv[#1]}%
606 \fi}

```

\setline You can use `\setline{<num>}` in running text (i.e., within `\pstart... \pend`) to set the current visible line-number to a specified positive value.

```

607 \renewcommand*{\setline}[1]{%
608 \ifnum#1<\z@
609 \led@warn@BadSetline
610 \else
611 \ifledRcol \write\linenum@outR{\string\@set[#1]}%
612 \else      \write\linenum@out{\string\@set[#1]}%
613 \fi
614 \fi}

```

\setlinenum You can use `\setlinenum{<num>}` before a `\pstart` to set the visible line-number to a specified positive value. It writes a `\l@d@set` command to the line-list file.

```

615 \renewcommand*{\setlinenum}[1]{%
616 \ifnum#1<\z@
617 \led@warn@BadSetlinenum
618 \else
619 \ifledRcol \write\linenum@outR{\string\l@d@set[#1]}
620 \else      \write\linenum@out{\string\l@d@set[#1]} \fi
621 \fi}
622

```

\startlock You can use `\startlock` or `\endlock` in running text to start or end line number locking at the current line. They decide whether line numbers or sub-line numbers are affected, depending on the current state of the sub-lineation flags.

```

623 \renewcommand*{\startlock}{%
624 \ifledRcol \write\linenum@outR{\string\lock@on}%
625 \else      \write\linenum@out{\string\lock@on}%
626 \fi}
627 \def\endlock{%
628 \ifledRcol \write\linenum@outR{\string\lock@off}%
629 \else      \write\linenum@out{\string\lock@off}%
630 \fi}
631

```

`\skipnumbering` In numbered text, `\skipnumbering` in a line will suspend the numbering for that particular line. That is, line numbers are unchanged and no line number will be printed.

```
632 \renewcommand*{\skipnumbering}{%
633   \ifledRcol \write\linenum@outR{\string\n@num}%
634             \advanceline{-1}%
635   \else
636     \skipnumbering@reg
637   \fi}
638
```

12 Marking text for notes

The `\edtext` (or `\critext`) macro is used to create all footnotes and endnotes, as well as to print the portion of the main text to which a given note or notes is keyed. The idea is to have that lemma appear only once in the `.tex` file: all instances of it in the main text and in the notes are copied from that one appearance.

`\critext` requires two arguments. At any point within numbered text, you use it by saying:

```
\critext{#1}#2/
```

Similarly `\edtext` requires the same two arguments but you use it by saying:

```
\edtext{#1}{#2}
```

`\critext` Now we begin `\critext` itself.

We slightly modify the original to make accomodation for when right text is being processed.

```
639 \long\def\critext#1#2/{\leavevmode
640   \begingroup
641     \no@expands
642     \xdef\@tag{#1}%
643     \set@line
644     \ifledRcol \global\insert@countR \z@
645     \else      \global\insert@count \z@ \fi
646     \ignorespaces #2\relax
647     \flag@start
648   \endgroup
649   \showlemma{#1}%
650   \ifx\end@lemmas\empty \else
651     \gl@p\end@lemmas\to\x@lemma
652     \x@lemma
653     \global\let\x@lemma=\relax
654   \fi
655   \flag@end}
```

`\edtext` And similarly for `\edtext`.

```

656 \renewcommand{\edtext}[2]{\leavevmode
657   \begingroup
658     \noexpands
659     \xdef\@tag{#1}%
660     \set@line
661     \ifledRcol \global\insert@countR \z@
662     \else      \global\insert@count \z@ \fi
663     \ignorespaces #2\relax
664     \flag@start
665   \endgroup
666   \showlemma{#1}%
667   \ifx\end@lemmas\empty \else
668     \glp\end@lemmas\to\x@lemma
669     \x@lemma
670     \global\let\x@lemma=\relax
671   \fi
672   \flag@end}
673
```

`\set@line` The `\set@line` macro is called by `\edtext` to put the line-reference field and font specifier for the current block of text into `\l@d@nums`.

```

674 \renewcommand*{\set@line}{%
675   \ifledRcol
676     \ifx\line@listR\empty
677       \global\noteschanged@true
678       \xdef\l@d@nums{000|000|000|000|000|000|\edfont@info}%
679     \else
680       \glp\line@listR\to\@tempb
681       \xdef\l@d@nums{\@tempb|\edfont@info}%
682       \global\let\@tempb=\undefined
683     \fi
684   \else
685     \ifx\line@list\empty
686       \global\noteschanged@true
687       \xdef\l@d@nums{000|000|000|000|000|000|\edfont@info}%
688     \else
689       \glp\line@list\to\@tempb
690       \xdef\l@d@nums{\@tempb|\edfont@info}%
691       \global\let\@tempb=\undefined
692     \fi
693   \fi}
694
```

13 Parallel environments

The initial set up for parallel processing is deceptively simple.

pairs The **pairs** environment is for parallel columns and the **pages** environment for parallel pages.

```
695 \newenvironment{pairs}{%
696   \l@dpairingtrue
697   \l@dpagingfalse
698 }{%
699   \l@dpairingfalse
700 }
```

The **pages** environment additionally sets the ‘column’ widths to the `\textwidth` (as known at the time the package is called).

```
701 \newenvironment{pages}{%
702   \l@dpairingtrue
703   \l@dpagingtrue
704   \setlength{\Lcolwidth}{\textwidth}%
705   \setlength{\Rcolwidth}{\textwidth}%
706 }{%
707   \l@dpairingfalse
708   \l@dpagingfalse
709 }
710
```

Leftside Within the **pairs** and **pages** environments the left and right hand texts are within **Leftside** and **Rightside** environments, respectively. The **Leftside** environment is simple, indicating that right text is not within its purview and using some particular macros.

```
711 \newenvironment{Leftside}{%
712   \ledRcolfalse
713   \let\pstart\pstartL
714   \let\pend\pendL
715   \let\memorydump\memorydumpL
716   \Leftsidehook
717 }{\Leftsidehookend}
```

`\Leftsidehook` Hooks into the start and end of the **Leftside** and **Rightside** environments. These
`\Leftsidehookend` are initially empty.

```
\Rightsidehook 718 \newcommand*{\Leftsidehook}{}
\Rightsidehookend 719 \newcommand*{\Leftsidehookend}{}
720 \newcommand*{\Rightsidehook}{}
721 \newcommand*{\Rightsidehookend}{}
722
```

Rightside The **Rightside** environment is only slightly more complicated than the **Leftside**. Apart from indicating that right text is being provided it ensures that the right right text code will be used.

```
723 \newenvironment{Rightside}{%
724   \ledRcoltrue
725   \let\beginnumbering\beginnumberingR
726   \let\endnumbering\endnumberingR
```



```

727 \let\pausenumbering\pausenumberingR
728 \let\resumenumbering\resumenumberingR
729 \let\memorydump\memorydumpR
730 \let\pstart\pstartR
731 \let\pend\pendR
732 \let\lineation\lineationR
733 \Rightsidehook
734 }{
735 \ledRcolfalse
736 \Rightsidehookend
737 }
738

```

14 Paragraph decomposition and reassembly

In order to be able to count the lines of text and affix line numbers, we add an extra stage of processing for each paragraph. We send the paragraph into a box register, rather than straight onto the vertical list, and when the paragraph ends we slice the paragraph into its component lines; to each line we add any notes or line numbers, add a command to write to the line-list, and then at last send the line to the vertical list. This section contains all the code for this processing.

14.1 Boxes, counters, \pstart and \pend

\num@linesR Here are numbers and flags that are used internally in the course of the paragraph decomposition.
\one@lineR
\par@lineR

When we first form the paragraph, it goes into a box register, **\l@dLcolrawbox** or **\l@dRcolrawbox** for right text, instead of onto the current vertical list. The **\ifnumberedpar@** flag will be **true** while a paragraph is being processed in that way. **\num@lines(R)** will store the number of lines in the paragraph when it's complete. When we chop it up into lines, each line in turn goes into the **\one@line** or **\one@lineR** register, and **\par@line(R)** will be the number of that line within the paragraph.

```

739 \newcount\num@linesR
740 \newbox\one@lineR
741 \newcount\par@lineR

```

\pstartL **\pstart** starts the paragraph by clearing the **\inserts@list** list and other relevant variables, and then arranges for the subsequent text to go into the appropriate box. **\pstart** needs to appear at the start of every paragraph that's to be numbered.

Beware: everything that occurs between **\pstart** and **\pend** is happening within a group; definitions must be global if you want them to survive past the end of the paragraph.

We have to have specific left and right **\pstart** when parallel processing; among other things because of potential changes in the linewidth.

```

742 \newcommand*{\pstartL}{\ifnumbering \else
743   \led@err@PstartNotNumbered
744   \beginnumbering
745   \fi
746   \ifnumberedpar@
747   \led@err@PstartInPstart
748   \pend
749   \fi

```

If this is the first \pstart in a numbered section, clear any inserts and set \ifpst@rtedL to FALSE.

```

750   \ifpst@rtedL\else
751     \list@clear{\inserts@list}%
752     \global\let\next@insert=\empty
753     \global\pst@rtedLtrue
754   \fi
755   \begingroup\normal@pars

```

When parallel processing we check that we haven't exceeded the maximum number of chunks. In any event we grab a box for the forthcoming text.

```

756   \global\advance\l@dnumpstartsL \@ne
757   \ifnum\l@dnumpstartsL>\l@dc@maxchunks
758     \led@err@TooManyPstarts
759     \global\l@dnumpstartsL=\l@dc@maxchunks
760   \fi
761   \global\setnamebox{\l@dLcolrawbox\the\l@dnumpstartsL}=\vbox\bgroup%
762     \hsize=\Lcolwidth
763   \numberedpar@true}

```

```

764 \newcommand*{\pstartR}{\ifnumberingR \else
765   \led@err@PstartNotNumbered
766   \beginnumberingR
767   \fi
768   \ifnumberedpar@
769   \led@err@PstartInPstart
770   \pendR
771   \fi
772   \ifpst@rtedR\else
773     \list@clear{\inserts@listR}%
774     \global\let\next@insertR=\empty
775     \global\pst@rtedRtrue
776   \fi
777   \begingroup\normal@pars
778   \global\advance\l@dnumpstartsR \@ne
779   \ifnum\l@dnumpstartsR>\l@dc@maxchunks
780     \led@err@TooManyPstarts
781     \global\l@dnumpstartsR=\l@dc@maxchunks
782   \fi
783   \global\setnamebox{\l@dRcolrawbox\the\l@dnumpstartsR}=\vbox\bgroup%
784     \hsize=\Rcolwidth
785   \numberedpar@true}

```

`\pendL` `\pend` must be used to end a numbered paragraph. Again we need a version that knows about left parallel texts.

```

786 \newcommand*{\pendL}{\ifnumbering \else
787   \led@err@PendNotNumbered
788 \fi
789 \ifnumberedpar@ \else
790   \led@err@PendNoPstart
791 \fi

```

We set all the usual interline penalties to zero and then immediately call `\endgraf` to end the paragraph; this ensures that there'll be no large interline penalties to prevent us from slicing the paragraph into pieces. These penalties revert to the values that you set when the group for the `\vbox` ends.

```

792 \l@dzeropenalties
793 \endgraf\global\num@lines=\prevgraf\egroup
794 \global\par@line=0

```

End the group that was begun in the `\pstart`.

```

795 \endgroup
796 \ignorespaces}
797

```

`\pendR` The version of `\pend` needed for right texts.

```

798 \newcommand*{\pendR}{\ifnumberingR \else
799   \led@err@PendNotNumbered
800 \fi
801 \ifnumberedpar@ \else
802   \led@err@PendNoPstart
803 \fi
804 \l@dzeropenalties
805 \endgraf\global\num@linesR=\prevgraf\egroup
806 \global\par@lineR=0
807 \endgroup
808 \ignorespaces}
809

```

14.2 Processing one line

For parallel texts we have to be able to process left and right lines independently. For sequential text we happily use the original `\do@line`. Otherwise ...

`\l@dleftbox` A line of left text will be put in the box `\l@dleftbox`, and analagously for a line
`\l@drightbox` of right text.

```

810 \newbox\l@dleftbox
811 \newbox\l@drightbox
812

```

`\countLline` We need to know the number of lines processed.

`\countRline` 813 \newcount\countLline

```

814 \countLline \z@
815 \newcount\countRline
816 \countRline \z@
817

\@donereallinesL We need to know the number of ‘real’ lines output (i.e., those that have been input
\@donetotallinesL by the user), and the total lines output (which includes any blank lines output for
\@donereallinesR synchronisation).
\@donetotallinesR 818 \newcount\@donereallinesL
819 \newcount\@donetotallinesL
820 \newcount\@donereallinesR
821 \newcount\@donetotallinesR
822

\do@lineL The \do@lineL macro is called to do all the processing for a single line of left text.

823 \newcommand*{\do@lineL}{%
824 \advance\countLline \@ne

If the current \l@dLcolrawbox box is not empty it contains the remaining unpro-
cessed lines of the chunk, so pull one line off the top. \vbadness must be cranked
up to suppress Underfull vbox errors from \vsplit; \splittopskip will be in-
serted at the top of \one@line, so we zero it. (This skip will appear in the final
vertical list, just before every \baselineskip.)

825 \ifvbox\namebox{\l@dLcolrawbox\the\l@dpscl}%
826 {\vbadness=10000 \splittopskip=0pt

Insert the \do@lineLhook and null the \...d@ta, which may later hold line
numbers, here. They will get defined within \affixline@num. Similarly for
\l@dcsnotetext for the text of a sidenote.

827 \do@lineLhook
828 \l@emptyd@ta

829 \global\setbox\one@line=\vsplit\namebox{\l@dLcolrawbox\the\l@dpscl}
830 to\baselineskip}%

\one@line comes out of \vsplit as a vbox; we now convert it to an hbox.
831 \unvbox\one@line \global\setbox\one@line=\lastbox

Calculate the line and page number for this line.
832 \getline@num

Now we’ll put the line into a box of the appropriate width, with a line number
attached if necessary.

833 \setbox\l@leftbox
834 \hb@xt@ \l@colwidth{%

Add line numbers, inserts, sidenotes, etc.

835 \affixline@num
836 \l@dld@ta
837 \add@inserts
838 \affixside@note

```

Now stick everything into a set of boxes.

```
839 \l@dlsn@te% left side note
840 {\ledllfill\hb@xt@ \wd\one@line{\new@line\unhbox\one@line}\ledrlfill\l@drd@ta
841 \l@drsn@te% right side note
842 }}}
```

Penalties get stripped off by this slicing process; the following macro puts them back in as the last step and then we increment the number of ‘real’ (text), and total lines done.

```
843 \add@penaltiesL
844 \global\advance\@donereallinesL\@ne
845 \global\advance\@donetotallinesL\@ne
```

Otherwise, the current `\l@dLcolrawbox` has been emptied, so just generate an empty box and increment the total lines.

```
846 \else
847 \setbox\l@leftbox \hb@xt@ \Lcolwidth{\hspace*\Lcolwidth}}%
848 \global\advance\@donetotallinesL\@ne
849 \fi}
850
```

`\do@lineLhook` Hooks, initially empty, into the respective `\do@line(L/R)` macros.

```
\do@lineRhook 851 \newcommand*{\do@lineLhook}{}
852 \newcommand*{\do@lineRhook}{}
853
```

`\do@lineR` The `\do@lineR` macro is called to do all the processing for a single line of right text.

```
854 \newcommand*{\do@lineR}{%
855 \advance\countRline \@ne
856 \ifvbox\namebox{\l@dRcolrawbox\the\l@dpscR}%
857 {\vbadness=10000 \splittopskip=0pt
858 \do@lineRhook
859 \l@emptyd@ta
860 \global\setbox\one@lineR=\vsplit\namebox{\l@dRcolrawbox\the\l@dpscR}
861 to\baselineskip}%
862 \unvbox\one@lineR \global\setbox\one@lineR=\lastbox
863 \getline@numR
864 \setbox\l@drightbox
865 \hb@xt@ \Rcolwidth{%
866 \affixline@numR
867 \l@dld@ta
868 \add@insertsR
869 \affixside@noteR
870 \l@dlsn@te% left side note
871 {\ledllfill\hb@xt@ \wd\one@lineR{\new@lineR\unhbox\one@lineR}\ledrlfill\l@drd@ta
872 \l@drsn@te% right side note
873 }}}}
874 \add@penaltiesR
875 \global\advance\@donereallinesR \@ne
```

```

876 \global\advance\@donetotallinesR \@ne
877 \else
878 \setbox\l@drightbox \hb@xt@ \Rcolwidth{\hspace*{\Rcolwidth}}
879 \global\advance\@donetotallinesR \@ne
880 \fi}
881

```

14.3 Line and page number computation

`\getline@numR` The `\getline@numR` macro determines the page and line numbers for the right text line we're about to send to the vertical list.

```

882 \newcommand*{\getline@numR}{%
883 \global\advance\absline@numR \@ne
884 \do@actionsR
885 \do@ballastR
886 \ifsublines@
887 \ifnum\sub@lock<\tw@
888 \global\advance\subline@numR \@ne
889 \fi
890 \else
891 \ifnum\@lock<\tw@
892 \global\advance\line@numR \@ne
893 \global\subline@numR=\z@
894 \fi
895 \fi}
896

```

`\do@ballastR` The real work in the line macros above is done in `\do@actions`, but before we plunge into that, let's get `\do@ballastR` out of the way.

```

897 \newcommand*{\do@ballastR}{\global\ballast@count=\z@
898 \begingroup
899 \advance\absline@numR \@ne
900 \ifnum\next@actionlineR=\absline@numR
901 \ifnum\next@actionR>-1001
902 \global\advance\ballast@count by -\c@ballast
903 \fi
904 \fi
905 \endgroup}

```

`\do@actionsR` The `\do@actionsR` macro looks at the list of actions to take at particular right text absolute line numbers, and does everything that's specified for the current line.

It may call itself recursively and we use tail recursion, via `\do@actions@nextR` for this.

```

906 \newcommand*{\do@actionsR}{%
907 \global\let\do@actions@nextR=\relax
908 \@l@dttempcntb=\absline@numR
909 \ifnum\@l@dttempcntb<\next@actionlineR\else
910 \ifnum\next@actionR>-1001

```

```

911     \global\page@numR=\next@actionR
912     \ifbypage@R
913       \global\line@numR=\z@ \global\subline@numR=\z@
914     \fi
915   \else
916     \ifnum\next@actionR<-4999
917       \@l@dttempcnta=-\next@actionR
918       \advance\@l@dttempcnta by -5001
919       \ifsublines@
920         \global\subline@numR=\@l@dttempcnta
921       \else
922         \global\line@numR=\@l@dttempcnta
923       \fi
924     \else
925       \@l@dttempcnta=-\next@actionR
926       \advance\@l@dttempcnta by -1000
927       \do@actions@fixedcode
928     \fi
929   \fi
930   \ifx\actionlines@listR\empty
931     \gdef\next@actionlineR{1000000}%
932   \else
933     \gl@p\actionlines@listR\to\next@actionlineR
934     \gl@p\actions@listR\to\next@actionR
935     \global\let\do@actions@nextR=\do@actionsR
936   \fi
937 \fi
938 \do@actions@nextR}
939
```

14.4 Line number printing

`\affixline@numR` `\affixline@numR` is the right text version of the `\affixline@num` macro.

```

940 \newcommand*{\affixline@numR}{%
941 \ifl@dskipnumber
942   \global\l@dskipnumberfalse
943 \else
944   \ifsublines@
945     \@l@dttempcntb=\subline@numR
946     \ifnum\subline@numR>\c@firstsublinenumR
947       \@l@dttempcnta=\subline@numR
948       \advance\@l@dttempcnta by-\c@firstsublinenumR
949       \divide\@l@dttempcnta by\c@sublinenumincrementR
950       \multiply\@l@dttempcnta by\c@sublinenumincrementR
951       \advance\@l@dttempcnta by\c@firstsublinenumR
952     \else
953       \@l@dttempcnta=\c@firstsublinenumR
954     \fi
955     \ch@cksub@l@ck

```

```

956 \else
957   \@l@tempcntb=\line@numR
958   \ifx\linenumberlist\empty
959     \ifnum\line@numR>\c@firstlinenumR
960       \@l@tempcnta=\line@numR
961       \advance\@l@tempcnta by-\c@firstlinenumR
962       \divide\@l@tempcnta by\c@linenumincrementR
963       \multiply\@l@tempcnta by\c@linenumincrementR
964       \advance\@l@tempcnta by\c@firstlinenumR
965     \else
966       \@l@tempcnta=\c@firstlinenumR
967     \fi
968   \else
969     \@l@tempcnta=\line@numR
970     \edef\rem@inder{\linenumberlist,\number\line@numR,}%
971     \edef\sc@n@list{\def\noexpand\sc@n@list
972       ###1,\number\@l@tempcnta,###2|{\def\noexpand\rem@inder{###2}}}%
973     \sc@n@list\expandafter\sc@n@list\rem@inder|
974     \ifx\rem@inder\empty\advance\@l@tempcnta\@ne\fi
975   \fi
976   \ch@ck@l@ck
977 \fi
978 \ifnum\@l@tempcnta=\@l@tempcntb
979 \if@twocolumn
980   \if@firstcolumn
981     \gdef\l@dld@ta{\llap{\leftlinenumR}}}%
982   \else
983     \gdef\l@drd@ta{\rlap{\rightlinenumR}}}%
984   \fi
985 \else
986   \@l@tempcntb=\line@marginR
987   \ifnum\@l@tempcntb>\@ne
988     \advance\@l@tempcntb by\page@numR
989   \fi
990   \ifodd\@l@tempcntb
991     \gdef\l@drd@ta{\rlap{\rightlinenumR}}}%
992   \else
993     \gdef\l@dld@ta{\llap{\leftlinenumR}}}%
994   \fi
995 \fi
996 \else
997 %%    #1%
998 \fi
999 \f@x@l@cks
1000 \fi}
1001

```


14.5 Add insertions to the vertical list

`\inserts@listR` `\inserts@listR` is the list macro that contains the inserts that we save up for one right text paragraph.

```
1002 \list@create{\inserts@listR}
```

`\add@insertsR` The right text version.

```
\add@inserts@nextR 1003 \newcommand*{\add@insertsR}{%
1004   \global\let\add@inserts@nextR=\relax
1005   \ifx\inserts@listR\empty \else
1006     \ifx\next@insertR\empty
1007       \ifx\insertlines@listR\empty
1008         \global\noteschanged@true
1009         \gdef\next@insertR{100000}%
1010       \else
1011         \gl@p\insertlines@listR\to\next@insertR
1012       \fi
1013     \fi
1014     \ifnum\next@insertR=\absline@numR
1015       \gl@p\inserts@listR\to\@insertR
1016       \@insertR
1017       \global\let\@insertR=\undefined
1018       \global\let\next@insertR=\empty
1019       \global\let\add@inserts@nextR=\add@insertsR
1020     \fi
1021   \fi
1022   \add@inserts@nextR}
1023
```

14.6 Penalties

`\add@penaltiesL` `\add@penaltiesL` is the last macro used by `\do@lineL`. It adds up the club, widow, and interline penalties, and puts a single penalty of the appropriate size back into the paragraph; these penalties get removed by the `\vsplit` operation. `\displaywidowpenalty` and `\brokenpenalty` are not restored, since we have no easy way to find out where we should insert them.

In the code below, which is a virtual copy of the original `\add@penalties`, `\num@lines` is the number of lines in the whole paragraph, and `\par@line` is the line we're working on at the moment. The count `\@l@dttempcnta` is used to calculate and accumulate the penalty; it is initially set to the value of `\ballast@count`, which has been worked out in `\do@ballast`. Finally, the penalty is checked to see that it doesn't go below `-10000`.

```
\newcommand*{\add@penaltiesR}{\@l@dttempcnta=\ballast@count
\ifnum\num@linesR>\@ne
\global\advance\par@lineR \@ne
\ifnum\par@lineR=\@ne
\advance\@l@dttempcnta by \clubpenalty
\fi}
```

```

\@l@dttempcntb=\par@lineR \advance\@l@dttempcntb \@ne
\ifnum\@l@dttempcntb=\num@linesR
  \advance\@l@dttempcnta by \widowpenalty
\fi
\ifnum\par@lineR<\num@linesR
  \advance\@l@dttempcnta by \interlinepenalty
\fi
\fi
\ifnum\@l@dttempcnta=\z@
  \relax
\else
  \ifnum\@l@dttempcnta>-10000
    \penalty\@l@dttempcnta
  \else
    \penalty -10000
  \fi
\fi}

```

This is for a single chunk. However, as we are probably dealing with several chunks at a time, the above is not really relevant. I think that it is likely with parallel text that there is no real need to add back any penalties; even if there was, they would have to match across the left and right lines. So, I end up with the following.

```

1024 \newcommand*{\add@penaltiesL}{-}
1025 \newcommand*{\add@penaltiesR}{-}
1026

```

14.7 Printing leftover notes

`\flush@notesR` The `\flush@notesR` macro is called after the entire right text has been sliced up and sent on to the vertical list.

```

1027 \newcommand*{\flush@notesR}{%
1028   \@xloop
1029   \ifx\inserts@listR\empty \else
1030     \gl@p\inserts@listR\to\@insertR
1031     \@insertR
1032     \global\let\@insertR=\undefined
1033   \repeat}
1034

```

15 Footnotes

15.1 Outer-level footnote commands

`\Afootnote` The outer-level footnote commands will look familiar: they're just called `\Afootnote`, `\Bfootnote`, etc., instead of plain `\footnote`. What they do, however, is quite different, since they have to operate in conjunction with `\edtext` when numbering is in effect.

If we're within a line-numbered paragraph, then, we tack this note onto the `\inserts@list` list, and increment the deferred-page-bottom-note counter.

```

1035 \renewcommand*{\Afootnote}[1]{%
1036   \ifnumberedpar@
1037     \ifledRcol
1038       \xright@appenditem{\noexpand\Afootnote{A}%
1039         {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1040       \global\advance\insert@countR \@ne
1041     \else
1042       \xright@appenditem{\noexpand\Afootnote{A}%
1043         {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1044       \global\advance\insert@count \@ne
1045     \fi

```

Within free text, there's no need to put off making the insertion for this note. No line numbers are available, so this isn't generally that useful; but you might want to use it to get around some limitation of `ledmac`.

```

1046 \else
1047   \vAfootnote{A}{\l@d@nums}{\@tag}{#1}}%
1048 \fi\ignorespaces}

```

`\Bfootnote` We need similar commands for the other footnote series.

```

\Bfootnote 1049 \renewcommand*{\Bfootnote}[1]{%
\Bfootnote 1050   \ifnumberedpar@
\Bfootnote 1051     \ifledRcol
1052       \xright@appenditem{\noexpand\Bfootnote{B}%
1053         {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1054       \global\advance\insert@countR \@ne
1055     \else
1056       \xright@appenditem{\noexpand\Bfootnote{B}%
1057         {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1058       \global\advance\insert@count \@ne
1059     \fi
1060   \else
1061     \vBfootnote{B}{\l@d@nums}{\@tag}{#1}}%
1062   \fi\ignorespaces}

1063 \renewcommand*{\Cfootnote}[1]{%
1064   \ifnumberedpar@
1065     \ifledRcol
1066       \xright@appenditem{\noexpand\Cfootnote{C}%
1067         {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1068       \global\advance\insert@countR \@ne
1069     \else
1070       \xright@appenditem{\noexpand\Cfootnote{C}%
1071         {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1072       \global\advance\insert@count \@ne
1073     \fi
1074   \else
1075     \vCfootnote{C}{\l@d@nums}{\@tag}{#1}}%

```

```

1076 \fi\ignorespaces}
1077 \renewcommand*{\Dfootnote}[1]{%
1078 \ifnumberedpar@
1079 \ifledRcol
1080 \xright@appenditem{\noexpand\VDfootnote{D}%
1081 {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1082 \global\advance\insert@countR \@ne
1083 \else
1084 \xright@appenditem{\noexpand\VDfootnote{D}%
1085 {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1086 \global\advance\insert@count \@ne
1087 \fi
1088 \else
1089 \VDfootnote{D}{{0|0|0|0|0|0|0|0}}{#1}}%
1090 \fi\ignorespaces}
1091 \renewcommand*{\Efootnote}[1]{%
1092 \ifnumberedpar@
1093 \ifledRcol
1094 \xright@appenditem{\noexpand\VEfootnote{E}%
1095 {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1096 \global\advance\insert@countR \@ne
1097 \else
1098 \xright@appenditem{\noexpand\VEfootnote{E}%
1099 {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1100 \global\advance\insert@count \@ne
1101 \fi
1102 \else
1103 \VEfootnote{E}{{0|0|0|0|0|0|0|0}}{#1}}%
1104 \fi\ignorespaces}
1105

```

`\mpAfootnote` For footnotes in minipages and the like, we need a similar series of commands.

```

\mpBfootnote 1106 \renewcommand*{\mpAfootnote}[1]{%
\mpCfootnote 1107 \ifnumberedpar@
\mpDfootnote 1108 \ifledRcol
\mpEfootnote 1109 \xright@appenditem{\noexpand\mpvAfootnote{A}%
1110 {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1111 \global\advance\insert@countR \@ne
1112 \else
1113 \xright@appenditem{\noexpand\mpvAfootnote{A}%
1114 {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1115 \global\advance\insert@count \@ne
1116 \fi
1117 \else
1118 \mpvAfootnote{A}{{0|0|0|0|0|0|0|0}}{#1}}%
1119 \fi\ignorespaces}
1120 \renewcommand*{\mpBfootnote}[1]{%
1121 \ifnumberedpar@

```

```

1122 \ifledRcol
1123   \xright@appenditem{\noexpand\mpvBfootnote{B}%
1124                       {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1125   \global\advance\insert@countR \@ne
1126 \else
1127   \xright@appenditem{\noexpand\mpvBfootnote{B}%
1128                       {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1129   \global\advance\insert@count \@ne
1130 \fi
1131 \else
1132   \mpvBfootnote{B}{\{0|0|0|0|0|0|0|0\}}{#1}}%
1133 \fi\ignorespaces}
1134 \renewcommand*{\mpCfootnote}[1]{%
1135   \ifnumberedpar@
1136   \ifledRcol
1137     \xright@appenditem{\noexpand\mpvCfootnote{C}%
1138                         {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1139     \global\advance\insert@countR \@ne
1140   \else
1141     \xright@appenditem{\noexpand\mpvCfootnote{C}%
1142                         {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1143     \global\advance\insert@count \@ne
1144   \fi
1145   \else
1146     \mpvCfootnote{C}{\{0|0|0|0|0|0|0|0\}}{#1}}%
1147   \fi\ignorespaces}
1148 \renewcommand*{\mpDfootnote}[1]{%
1149   \ifnumberedpar@
1150   \ifledRcol
1151     \xright@appenditem{\noexpand\mpvDfootnote{D}%
1152                         {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1153     \global\advance\insert@countR \@ne
1154   \else
1155     \xright@appenditem{\noexpand\mpvDfootnote{D}%
1156                         {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1157     \global\advance\insert@count \@ne
1158   \fi
1159   \else
1160     \mpvDfootnote{D}{\{0|0|0|0|0|0|0|0\}}{#1}}%
1161   \fi\ignorespaces}
1162 \renewcommand*{\mpEfootnote}[1]{%
1163   \ifnumberedpar@
1164   \ifledRcol
1165     \xright@appenditem{\noexpand\mpvEfootnote{E}%
1166                         {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1167     \global\advance\insert@countR \@ne
1168   \else
1169     \xright@appenditem{\noexpand\mpvEfootnote{E}%
1170                         {\l@d@nums}{\@tag}{#1}}\to\inserts@list

```

```

1171 \global\advance\insert@count \@ne
1172 \fi
1173 \else
1174 \mpvEfootnote{E}{\{0|0|0|0|0|0|0\}\{#1\}}%
1175 \fi\ignorespaces}

```

15.2 Normal footnote formatting

The `\printlines` macro prints the line numbers for a note—which, in the general case, is a rather complicated task. The seven parameters of the argument are the line numbers as stored in `\l@d@nums`, in the form described on page ??: the starting page, line, and sub-line numbers, followed by the ending page, line, and sub-line numbers, and then the font specifier for the lemma.

```

\printlinesR This is the right text version of \printlines and takes account of \Rlineflag.
\ledsavedprintlines Just in case, \ledsavedprintlines is a copy of the original \printlines.
                Just a reminder of the arguments:
\printlinesR   #1      | #2 | #3 | #4 | #5 | #6 | #7
\printlinesR start-page | line | subline | end-page | line | subline | font
1176 \def\printlinesR#1|#2|#3|#4|#5|#6|#7|{\begingroup
1177 \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
1178 \ifl@d@pnum #1\fullstop\fi
1179 \ifledplinenum \linenumr@p{#2}\Rlineflag\else \symlinenum\fi
1180 \ifl@d@ssub \fullstop \sublinenumr@p{#3}\fi
1181 \ifl@d@dash \endashchar\fi
1182 \ifl@d@pnum #4\fullstop\fi
1183 \ifl@d@elin \linenumr@p{#5}\Rlineflag\fi
1184 \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumr@p{#6}\fi
1185 \endgroup}
1186
1187 \let\ledsavedprintlines\printlines
1188

```

16 Cross referencing

`\labelref@listR` Set up a new list, `\labelref@listR`, to hold the page, line and sub-line numbers for each label in right text.

```

1189 \list@create{\labelref@listR}
1190

```

`\edlabel` The `\edlabel` command first writes a `\@lab` macro to the `\linenum@out` file. It then checks to see that the `\labelref@list` actually has something in it (if not, it creates a dummy entry), and pops the next value for the current label, storing it in `\label@refs`. Finally it defines the label to be `\empty` so that any future check will turn up the fact that it has been used.

```

1191 \renewcommand*{\edlabel}[1]{\@bsphack
1192 \ifledRcol

```

```

1193 \write\linenum@outR{\string\@lab}%
1194 \ifx\labelref@listR\empty
1195 \xdef\label@refs{\zz@@@}%
1196 \else
1197 \gl@p\labelref@listR\to\label@refs
1198 \fi
1199 \protected@write\@auxout{%
1200 {\string\l@dmake@labelsR\space\thepage|\label@refs|{#1}}%
1201 \else
1202 \write\linenum@out{\string\@lab}%
1203 \ifx\labelref@list\empty
1204 \xdef\label@refs{\zz@@@}%
1205 \else
1206 \gl@p\labelref@list\to\label@refs
1207 \fi
1208 \fi
1209 \protected@write\@auxout{%
1210 {\string\l@dmake@labels\space\thepage|\label@refs|{#1}}%
1211 \@esphack}
1212

```

`\l@dmake@labelsR` This is the right text version of `\l@dmake@labels`, taking account of `\Rlineflag`.

```

1213 \def\l@dmake@labelsR#1|#2|#3|#4{%
1214 \expandafter\ifx\csname the@label#4\endcsname \relax\else
1215 \led@warn@DuplicateLabel{#4}%
1216 \fi
1217 \expandafter\gdef\csname the@label#4\endcsname{#1|#2\Rlineflag|#3}%
1218 \ignorespaces}
1219 \AtBeginDocument{%
1220 \def\l@dmake@labelsR#1|#2|#3|#4{%
1221 }
1222

```

`\@lab` The `\@lab` command, which appears in the `\linenum@out` file, appends the current values of page, line and sub-line to the `\labelref@list`. These values are defined by the earlier `\@page`, `\@l`, and the `\sub@on` and `\sub@off` commands appearing in the `\linenum@out` file.

```

1223 \renewcommand*{\@lab}{%
1224 \ifledRcol
1225 \xright@appenditem{\linenumr@p{\line@numR}}{|%
1226 \ifsublines@ \sublinenumr@p{\subline@numR}\else 0\fi}%
1227 \to\labelref@listR
1228 \else
1229 \xright@appenditem{\linenumr@p{\line@num}}{|%
1230 \ifsublines@ \sublinenumr@p{\subline@num}\else 0\fi}%
1231 \to\labelref@list
1232 \fi}
1233

```

17 Side notes

Regular `\marginpars` do not work inside numbered text — they don't produce any note but do put an extra unnumbered blank line into the text.

`\sidenote@marginR` Specifies which margin sidenotes can be in.

```
\sidenotemargin 1234 \newcount\sidenote@marginR
                  1235 \renewcommand*{\sidenotemargin}[1]{%
                  1236   \l@dgetsidenote@margin{#1}%
                  1237   \ifnum\@l@dttempcntb>\m@ne
                  1238     \ifledRcol
                  1239       \global\sidenote@marginR=\@l@dttempcntb
                  1240     \else
                  1241       \global\sidenote@margin=\@l@dttempcntb
                  1242     \fi
                  1243   \fi}}
                  1244 \sidenotemargin{right}
                  1245 \global\sidenote@margin=\@ne
                  1246
```

`\l@dlsnote` The ‘footnotes’ for left, right, and moveable sidenotes. The whole scheme is reminiscent of the critical footnotes code.

```
\l@dcsnote 1247 \renewcommand*{\l@dlsnote}[1]{%
            1248   \ifnumberedpar@
            1249     \ifledRcol
            1250       \xright@appenditem{\noexpand\l@dlsnote{\l@d@nums}{\@tag}{#1}}%
            1251                           \to\inserts@listR
            1252       \global\advance\insert@countR \@ne
            1253     \else
            1254       \xright@appenditem{\noexpand\l@dlsnote{\l@d@nums}{\@tag}{#1}}%
            1255                           \to\inserts@list
            1256       \global\advance\insert@count \@ne
            1257     \fi
            1258   \fi\ignorespaces}
            1259 \renewcommand*{\l@drsnote}[1]{%
            1260   \ifnumberedpar@
            1261     \ifledRcol
            1262       \xright@appenditem{\noexpand\l@drsnote{\l@d@nums}{\@tag}{#1}}%
            1263                           \to\inserts@listR
            1264       \global\advance\insert@countR \@ne
            1265     \else
            1266       \xright@appenditem{\noexpand\l@drsnote{\l@d@nums}{\@tag}{#1}}%
            1267                           \to\inserts@list
            1268       \global\advance\insert@count \@ne
            1269     \fi
            1270   \fi\ignorespaces}
            1271 \renewcommand*{\l@dcsnote}[1]{%
            1272   \ifnumberedpar@
            1273     \ifledRcol
            1274       \xright@appenditem{\noexpand\l@dcsnote{\l@d@nums}{\@tag}{#1}}%
```



```

1275             \to\inserts@listR
1276     \global\advance\insert@countR \@ne
1277     \else
1278       \xright@appenditem{\noexpand\l@dcsnote{\l@d@nums}{\@tag}{#1}}}%
1279       \to\inserts@list
1280     \global\advance\insert@count \@ne
1281     \fi
1282 \fi\ignorespaces}
1283

```

`\affixside@noteR` The right text version of `\affixside@note`.

```

1284 \newcommand*{\affixside@noteR}{%
1285   \gdef\@templ@d{%
1286     \ifx\@templ@d\l@dcsnote\text \else
1287       \if@twocolumn
1288         \if@firstcolumn
1289           \setl@dlp@rbox{}{\l@dcsnote\text}%
1290         \else
1291           \setl@drp@rbox{}{\l@dcsnote\text}%
1292         \fi
1293       \else
1294         \@l@tempcntb=\sidenote@marginR
1295         \ifnum\@l@tempcntb>\@ne
1296           \ifl@dpaging
1297             \advance\@l@tempcntb by\@ne
1298           \else
1299             \advance\@l@tempcntb by\page@num
1300           \fi
1301         \fi
1302         \ifodd\@l@tempcntb
1303           \setl@drp@rbox{}{\l@dcsnote\text}%
1304         \else
1305           \setl@dlp@rbox{}{\l@dcsnote\text}%
1306         \fi
1307       \fi
1308     \fi}
1309

```

18 Familiar footnotes

`\l@dbfnote` `\l@dbfnote` adds the footnote to the insert list, and `\vl@dbfnote` calls the original `\@footnotetext`.

```

1310 \renewcommand{\l@dbfnote}[1]{%
1311   \ifnumberedpar@
1312     \ifledRcol
1313       \xright@appenditem{\noexpand\vl@dbfnote{#1}{\@thefnmark}}%
1314       \to\inserts@listR
1315     \global\advance\insert@countR \@ne

```

```

1316     \else
1317         \xright@appenditem{\noexpand\vl@dbfnote{#{1}}{\@thefnmark}}%
1318                                 \to\inserts@list
1319         \global\advance\insert@count \@ne
1320     \fi
1321 \fi\ignorespaces}
1322
\normalbfnoteX
1323 \renewcommand{\normalbfnoteX}[2]{%
1324     \ifnumberedpar@
1325         \ifledRcol
1326             \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}{\@nameuse{thefootnote#1}}}%
1327                                     \to\inserts@listR
1328             \global\advance\insert@countR \@ne
1329         \else
1330             \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}{\@nameuse{thefootnote#1}}}%
1331                                     \to\inserts@list
1332             \global\advance\insert@count \@ne
1333         \fi
1334     \fi\ignorespaces}
1335

```

19 Verse

Before we can define the main stanza macros we need to be able to save and reset the category code for &. To save the current value we use `\next` from the `\loop` macro.

```

1336 \chardef\next=\catcode'\&
1337 \catcode'\&=\active
1338

```

astanza This is roughly an environmental form of `\stanza`, which treats its stanza-like contents as a single chunk.

```

1339 \newenvironment{astanza}{%
1340     \startstanzahook
1341     \catcode'\&=\active
1342     \global\stanza@count\@ne
1343     \ifnum\usernamecount{sza@0@}=\z@
1344         \let\stanza@hang\relax
1345     \let\endlock\relax
1346     \else
1347     %%% \interlinepenalty\@M % this screws things up, but I don't know why
1348     \rightskip\z@ plus 1fil\relax
1349     \fi
1350     \ifnum\usernamecount{szp@0@}=\z@
1351         \let\sza@penalty\relax
1352     \fi

```

```

1353 \def&{%
1354   \endlock\mbox{}}%
1355   \sza@penalty
1356   \global\advance\stanza@count\@ne
1357   \@astanza@line}%
1358 \def\&{%
1359   \endlock\mbox{}}
1360   \pend
1361   \endstanzaextra}%
1362 \pstart
1363 \@astanza@line
1364 }{}
1365

```

`\@astanza@line` This gets put at the start of each line in the environment. It sets up the paragraph style — each line is treated as a paragraph.

```

1366 \newcommand*{\@astanza@line}{%
1367   \parindent=\csname sza@\number\stanza@count @\endcsname\stanzaindentbase
1368   \par
1369   \stanza@hang%\mbox{}}%
1370   \ignorespaces}
1371

```

Lastly reset the modified category codes.

```

1372 \catcode'\&=\next
1373

```

20 Naming macros

The LaTeX kernel provides `\@namedef` and `\@namuse` for defining and using macros that may have non-letters in their names. We need something similar here as we are going to need and use some numbered boxes and counters.

`\newnamebox` A set of macros for creating and using ‘named’ boxes; the macros are called after `\setnamebox` the regular box macros, but including the string ‘name’.

```

\unhnamebox 1374 \providecommand*{\newnamebox}[1]{%
\unvnamebox 1375   \expandafter\newbox\csname #1\endcsname}
\namebox 1376 \providecommand*{\setnamebox}[1]{%
1377   \expandafter\setbox\csname #1\endcsname}
1378 \providecommand*{\unhnamebox}[1]{%
1379   \expandafter\unhbox\csname #1\endcsname}
1380 \providecommand*{\unvnamebox}[1]{%
1381   \expandafter\unvbox\csname #1\endcsname}
1382 \providecommand*{\namebox}[1]{%
1383   \csname #1\endcsname}
1384

```

`\newnamecount` Macros for creating and using ‘named’ counts.
`\usenamecount`

```

1385 \providecommand*\newnamecount}[1]{%
1386   \expandafter\newcount\csname #1\endcsname}
1387 \providecommand*\usenamecount}[1]{%
1388   \csname #1\endcsname}
1389

```

21 Counts and boxes for parallel texts

In sequential text, each chunk (that enclosed by `\pstart ... \pend`) is put into a box called `\raw@text` and then immediately printed, resulting in the box being emptied and ready for the next chunk. For parallel processing multiple boxes are needed as printing is delayed. We also need extra counters for various things.

`\maxchunks` The maximum number of chunk pairs before printing has to be called for. The `\l@dc@maxchunks` default is 10 chunk pairs.

```

1390 \newcount\l@dc@maxchunks
1391 \newcommand*\maxchunks}[1]{\l@dc@maxchunks=#1}
1392 \maxchunks{10}
1393

```

`\l@dnumpstartsL` The numbers of left and right chunks. `\l@dnumpstartsL` is defined in `ledmac`.

```

\l@dnumpstartsR 1394 \newcount\l@dnumpstartsR
1395

```

`\l@pscL` A couple of scratch counts for use in left and right texts, respectively.

```

\l@pscR 1396 \newcount\l@dpscL
1397 \newcount\l@dpscR
1398

```

`\l@dsetuprawboxes` This macro creates `\maxchunks` pairs of boxes for left and right chunks. The boxes are called `\l@dLcolrawbox1`, `\l@dLcolrawbox2`, etc.

```

1399 \newcommand*\l@dsetuprawboxes){%
1400   \@l@tempcntb=\l@dc@maxchunks
1401   \loop\ifnum\@l@tempcntb>\z@
1402     \newnamebox{\l@dLcolrawbox\the\@l@tempcntb}
1403     \newnamebox{\l@dRcolrawbox\the\@l@tempcntb}
1404     \advance\@l@tempcntb \m@ne
1405   \repeat}
1406

```

`\l@dsetupmaxlinecounts` To be able to synchronise left and right texts we need to know the maximum number of text lines there are in each pair of chunks. `\l@dsetupmaxlinecounts` creates `\maxchunks` new counts called `\l@dmaxlinesinpar1`, etc., and `\l@dzeromaxlinecounts` zeroes all of them.

```

1407 \newcommand*\l@dsetupmaxlinecounts){%
1408   \@l@tempcntb=\l@dc@maxchunks
1409   \loop\ifnum\@l@tempcntb>\z@

```

```

1410 \newnamecount{l@maxlinesinpar\the\@l@tempcntb}
1411 \advance\@l@tempcntb \m@ne
1412 \repeat}
1413 \newcommand*{\l@deromaxlinecounts}{%
1414 \begingroup
1415 \l@tempcntb=\l@dc@maxchunks
1416 \loop\ifnum\@l@tempcntb>\z@
1417 \global\usenamecount{l@maxlinesinpar\the\@l@tempcntb}=\z@
1418 \advance\@l@tempcntb \m@ne
1419 \repeat
1420 \endgroup}
1421

```

Make sure that all these are set up. This has to be done after the user has had an opportunity to change `\maxchunks`.

```

1422 \AtBeginDocument{%
1423 \l@setuprawboxes
1424 \l@setupmaxlinecounts
1425 \l@deromaxlinecounts
1426 \l@dumpstartsL=\z@
1427 \l@dumpstartsR=\z@
1428 \l@dpscl=\z@
1429 \l@dpscr=\z@}
1430

```

22 Fixing babel

With parallel texts there is the possibility that the two sides might use different languages via `babel`. On the other hand, `babel` might not be called at all (even though it might be already built into the format).

With the normal sequential text each line is initially typeset in the current language environment, and then it is output at which time its attachments are typeset (in the same language environment. In the parallel case lines are typeset in their current language but an attachment might be typeset outside the language environment of its line if the left and right side languages are different. To counter this, we have to make sure that the correct language is used at the proper times.

```

\ifl@dusedbabel A flag for checking if babel has been used as a package.
\l@dusedbabelfalse 1431 \newif\ifl@dusedbabel
\l@dusedbabeltrue 1432 \l@dusedbabelfalse

\ifl@dsamelang A flag for checking if the same babel language has been used for both the left and
\l@dsamelangfalse right texts.
\l@dsamelangtrue 1433 \newif\ifl@dsamelang
1434 \l@dsamelangtrue

```

`\l@dcchecklang` I'm going to use `\theledlanguageL` and `\theledlanguageR` to hold the names of the languages used for the left and right texts. This macro sets `\ifl@dsamelang` TRUE if they are the same, otherwise it sets it FALSE.

```

1435 \newcommand*{\l@dcchecklang}{%
1436   \l@dsamelangfalse
1437   \edef\@tempa{\theledlanguageL}\edef\@tempb{\theledlanguageR}%
1438   \ifx\@tempa\@tempb
1439     \l@dsamelangtrue
1440   \fi}
1441
```

`\l@dbbl@set@language` In babel the macro `\bbl@set@language{<lang>}` does the work when the language `<lang>` is changed via `\selectlanguage`. Unfortunately for me, if it is given an argument in the form of a control sequence it strips off the `\` character rather than expanding the command. I need a version that accepts an argument in the form `\lang` without it stripping the `\`.

```

1442 \newcommand*{\l@dbbl@set@language}[1]{%
1443   \edef\language{#1}%
1444   \select@language{\language}%
1445   \if@filesw
1446     \protected@write\@auxout{}\string\select@language{\language}%
1447     \addtocontents{toc}{\string\select@language{\language}%
1448     \addtocontents{lof}{\string\select@language{\language}%
1449     \addtocontents{lot}{\string\select@language{\language}%
1450   \fi}
1451
```

The rest of the setup has to be postponed until the end of the preamble when we know if `babel` has been used or not. However, for now assume that it has not been used.

`\selectlanguage` `\selectlanguage` is a babel command. `\theledlanguageL` and `\theledlanguageR`
`\l@duselanguage` are the names of the languages of the left and right texts. `\l@duselanguage` is
`\theledlanguageL` similar to `\selectlanguage`.

```

\theledlanguageR 1452 \providecommand{\selectlanguage}[1]{%
1453 \newcommand*{\l@duselanguage}[1]{%
1454 \gdef\theledlanguageL{}
1455 \gdef\theledlanguageR{}
1456
```

Now do the `babel` fix or `polyglossia`, if necessary.

```

1457 \AtBeginDocument{%
1458   \@ifundefined{xpg@main@language}{%
1459     \@ifundefined{bbl@main@language}{%
```

Either `babel` has not been used or it has been used with no specified language.

```

1460   \l@dusedbabelfalse
1461   \renewcommand*{\selectlanguage}[1]{}}%
```

Here we deal with the case where babel has been used. `\selectlanguage` has to be redefined to use our version of `\bbl@set@language` and to store the left or right language.

```

1462 \l@dusedbabeltrue
1463 \let\l@doldselectlanguage\selectlanguage
1464 \let\l@doldbbl@set@language\bbl@set@language
1465 \let\bbl@set@language\l@dbbl@set@language
1466 \renewcommand{\selectlanguage}[1]{%
1467   \l@doldselectlanguage{#1}%
1468   \ifledRcol \gdef\theledlanguageR{#1}%
1469   \else      \gdef\theledlanguageL{#1}%
1470   \fi}

```

`\l@duselanguage` simply calls the original `\selectlanguage` so that `\theledlanguageL` and `\theledlanguageR` are unaltered.

```

1471 \renewcommand*{\l@duselanguage}[1]{%
1472   \l@doldselectlanguage{#1}}

```

Lastly, initialise the left and right languages to the current babel one.

```

1473 \gdef\theledlanguageL{\bbl@main@language}%
1474 \gdef\theledlanguageR{\bbl@main@language}%
1475 }%
1476 }

```

If on Polyglossia

```

1477 { \apptocmd{\xpg@set@language}{%
1478   \ifledRcol \gdef\theledlanguageR{#1}%
1479   \else      \gdef\theledlanguageL{#1}%
1480   \fi}%
1481   \let\l@duselanguage\xpg@set@language
1482   \gdef\theledlanguageL{\xpg@main@language}%
1483   \gdef\theledlanguageR{\xpg@main@language}%
1484 % \end{macrocode}
1485 % That's it.
1486 % \begin{macrocode}
1487 }}

```

23 Parallel columns

`\Columns` The `\Columns` command results in the previous Left and Right texts being typeset in matching columns. There should be equal numbers of chunks in the left and right texts.

```

1488 \newcommand*{\Columns}{%
1489   \ifnum\l@dnumpstartsL=\l@dnumpstartsR\else
1490     \led@err@BadLeftRightPstarts{\the\l@dnumpstartsL}{\the\l@dnumpstartsR}%
1491   \fi

```

Start a group and zero counters, etc.

```

1492 \begingroup

```

```

1493 \l@dzeroopenalties
1494 \endgraf\global\num@lines=\prevgraf
1495 \global\num@linesR=\prevgraf
1496 \global\par@line=\z@
1497 \global\par@lineR=\z@
1498 \global\l@dpscL=\z@
1499 \global\l@dpscR=\z@

```

Check if there are chunks to be processed, and process them two by two (left and right pairs).

```

1500 \check@pstarts
1501 \loop\if@pstarts

```

Increment `\l@dpscL` and `\l@dpscR` which here count the numbers of left and right chunks.

```

1502 \global\advance\l@dpscL \@ne
1503 \global\advance\l@dpscR \@ne

```

Check if there is text yet to be processed in at least one of the two current chunks, and also whether the left and right languages are the same

```

1504 \checkraw@text
1505 \l@dchecklang
1506 { \loop\ifaraw@text

```

Grab the next pair of left and right text lines and output them, swapping languages if they differ

```

1507 \ifl@dsamelang
1508 \do@lineL
1509 \do@lineR
1510 \else
1511 \l@duselanguage{\theledlanguageL}%
1512 \do@lineL
1513 \l@duselanguage{\theledlanguageR}%
1514 \do@lineR
1515 \fi
1516 \hb@xt@ \hsize{%
1517 \unhbox\l@leftbox
1518 \hfill \columnseparator \hfill
1519 \unhbox\l@rightbox
1520 }%
1521 \checkraw@text
1522 \repeat}

```

Having completed a pair of chunks, write the number of lines in each chunk to the respective section files.

```

1523 \@writelinesinparL
1524 \@writelinesinparR
1525 \check@pstarts
1526 \repeat

```

Having output all chunks, make sure all notes have been output, then zero counts ready for the next set of texts.


```

1527 \flush@notes
1528 \flush@notesR
1529 \endgroup

1530 \global\l@dpscL=\z@
1531 \global\l@dpscR=\z@
1532 \global\l@dnumpestartsL=\z@
1533 \global\l@dnumpestartsR=\z@
1534 \ignorespaces}
1535

```

`\columnseparator` The separator between line pairs in parallel columns is in the form of a vertical rule extending a little below the baseline and with a height slightly greater than the `\baselineskip`. The width of the rule is `\columnrulewidth` (initially 0pt so the rule is invisible).

```

1536 \newcommand*{\columnseparator}{%
1537 \smash{\rule[-0.2\baselineskip]{\columnrulewidth}{1.05\baselineskip}}}
1538 \newdimen\columnrulewidth
1539 \columnrulewidth=\z@
1540

```

`\if@pstarts` `\check@pstarts` returns `\@pstartstrue` if there are any unprocessed chunks.

```

\@pstartstrue 1541 \newif\if@pstarts
\@pstartstrue 1542 \newcommand*{\check@pstarts}{%
\check@pstarts 1543 \if@pstartstrue
1544 \ifnum\l@dnumpestartsL>\l@dpscL
1545 \if@pstartstrue
1546 \else
1547 \ifnum\l@dnumpestartsR>\l@dpscR
1548 \if@pstartstrue
1549 \fi
1550 \fi}
1551

```

`\ifaraw@text` `\checkraw@text` checks whether the current Left or Right box is void or not. If `\araw@texttrue` one or other is not void it sets `\araw@texttrue`, otherwise both are void and it `\araw@textfalse` sets `\araw@textfalse`.

```

\checkraw@text 1552 \newif\ifaraw@text
1553 \araw@textfalse
1554 \newcommand*{\checkraw@text}{%
1555 \araw@textfalse
1556 \ifvbox\namebox{\l@dLcolrawbox\the\l@dpscL}
1557 \araw@texttrue
1558 \else
1559 \ifvbox\namebox{\l@dRcolrawbox\the\l@dpscR}
1560 \araw@texttrue
1561 \fi
1562 \fi}
1563

```

`\@writelinesinparL` These write the number of text lines in a chunk to the section files, and then
`\@writelinesinparR` afterwards zero the counter.

```

1564 \newcommand*{\@writelinesinparL}{%
1565   \edef\next{%
1566     \write\linenum@out{\string\@pend[\the\@donereallinesL]}}%
1567   \next
1568   \global\@donereallinesL \z@}
1569 \newcommand*{\@writelinesinparR}{%
1570   \edef\next{%
1571     \write\linenum@outR{\string\@pendR[\the\@donereallinesR]}}%
1572   \next
1573   \global\@donereallinesR \z@}
1574
```

24 Parallel pages

This is considerably more complicated than parallel columns.

`\numpagelinesL` Counts for the number of lines on a left or right page, and the smaller of the
`\numpagelinesR` number of lines on a pair of facing pages.

```

\l@dminpagelines 1575 \newcount\numpagelinesL
                  1576 \newcount\numpagelinesR
                  1577 \newcount\l@dminpagelines
                  1578
```

`\Pages` The `\Pages` command results in the previous Left and Right texts being typeset on matching facing pages. There should be equal numbers of chunks in the left and right texts.

```

1579 \newcommand*{\Pages}{%
1580   \typeout{}
1581   \typeout{***** PAGES *****}
1582   \ifnum\l@dnumpstartsL=\l@dnumpstartsR\else
1583     \led@err@BadLeftRightPstarts{\the\l@dnumpstartsL}{\the\l@dnumpstartsR}%
1584   \fi
```

Get onto an empty even (left) page, then initialise counters, etc.

```

1585 \cleartol@evenpage
1586 \begingroup
1587   \l@dzeropenalties
1588   \endgraf\global\num@lines=\prevgraf
1589   \global\num@linesR=\prevgraf
1590   \global\par@line=\z@
1591   \global\par@lineR=\z@
1592   \global\l@dpscL=\z@
1593   \global\l@dpscR=\z@
1594   \writtenlinesLfalse
1595   \writtenlinesRfalse
```

Check if there are chunks to be processed.

```
1596      \check@pstarts
1597      \loop\if@pstarts
```

Loop over the number of chunks, incrementing the chunk counts (\l@dpscL and \l@dpscR are chunk (box) counts.)

```
1598      \global\advance\l@dpscL \@ne
1599      \global\advance\l@dpscR \@ne
```

Calculate the maximum number of real text lines in the chunk pair, storing the result in the relevant \l@dmmaxlinesinpar.

```
1600      \getlinesfromparlistL
1601      \getlinesfromparlistR
1602      \l@dcalc@maxoftwo{\@cs@linesinparL}{\@cs@linesinparR}%
1603      {\usernamecount{l@dmmaxlinesinpar\the\l@dpscL}}%
1604      \check@pstarts
1605      \repeat
```

Zero the counts again, ready for the next bit.

```
1606      \global\l@dpscL=\z@
1607      \global\l@dpscR=\z@
```

Get the number of lines on the first pair of pages and store the minimum in \l@dmminpagelines.

```
1608      \getlinesfrompagelistL
1609      \getlinesfrompagelistR
1610      \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
1611      {\l@dmminpagelines}%
```

Now we start processing the left and right chunks (\l@dpscL and \l@dpscR count the left and right chunks), starting with the first pair.

```
1612      \check@pstarts
1613      \if@pstarts
```

Increment the chunk counts to get the first pair.

```
1614      \global\advance\l@dpscL \@ne
1615      \global\advance\l@dpscR \@ne
```

We haven't processed any lines from these chunks yet, so zero the respective line counts.

```
1616      \global\@donereallinesL=\z@
1617      \global\@donetotallinesL=\z@
1618      \global\@donereallinesR=\z@
1619      \global\@donetotallinesR=\z@
```

Start a loop over the boxes (chunks).

```
1620      \checkraw@text
1621 %      \begingroup
1622 {      \loop\ifaraw@text
```

See if there is more that can be done for the left page and set up the left language.

```

1623      \checkpageL
1624      \l@duselanguage{\theledlanguageL}%
1625      %%%
1626      \begingroup
1627      \loop\ifl@dsamepage

```

Process the next (left) text line, adding it to the page.

```

1627      \do@lineL
1628      \advance\numpagelinesL \@ne
1629      \hb@xt@ \hsize{\ledstrutL\unhbox\l@dleftbox}%

```

Perhaps we have to move to the next (left) box. Check if we have got all we can onto the page. If not, repeat for the next line.

```

1630      \get@nextboxL
1631      \checkpageL
1632      \repeat

```

That (left) page has been filled. Output the number of real lines on the page — if the page break is because the page has been filled with lines, use the actual number, otherwise the page has been ended early in order to synchronise with the facing page so use an impossibly large number.

```

1633      \ifl@dpagfull
1634      \@writelinesonpageL{\the\numpagelinesL}%
1635      \else
1636      \@writelinesonpageL{1000}%
1637      \fi

```

Zero the left page lines count and clear the page to get onto the facing (odd, right) page.

```

1638      \numpagelinesL \z@
1639      \clearl@dleftpage }%

```

Now do the same for the right text.

```

1640      \checkpageR
1641      \l@duselanguage{\theledlanguageR}%
1642      {
1643      \loop\ifl@dsamepage
1644      \do@lineR
1645      \advance\numpagelinesR \@ne
1646      \hb@xt@ \hsize{\ledstrutR\unhbox\l@drighthbox}%
1647      \get@nextboxR
1648      \checkpageR
1649      \repeat
1650      \ifl@dpagfull
1651      \@writelinesonpageR{\the\numpagelinesR}%
1652      \else
1653      \@writelinesonpageR{1000}%
1654      \fi
1655      \numpagelinesR=\z@

```

The page is full, so move onto the next (left, odd) page and repeat left text processing.

```

1655      \clearl@drightpage}

More to do? If there is we have to get the number of lines for the next pair of
pages before starting to output them.

1656      \checkraw@text
1657      \ifaraw@text
1658      \getlinesfrompagelistL
1659      \getlinesfrompagelistR
1660      \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
1661                      {\l@dminpagelines}%
1662      \fi
1663      \repeat}

```

We have now output the text from all the chunks.

```

1664      \fi

Make sure that there are no inserts hanging around.

1665      \flush@notes
1666      \flush@notesR
1667      \endgroup

```

Zero counts ready for the next set of left/right text chunks.

```

1668      \global\l@dpscL=\z@
1669      \global\l@dpscR=\z@
1670      \global\l@dnumpestartsL=\z@
1671      \global\l@dnumpestartsR=\z@
1672      \ignorespaces}
1673

```

`\ledstrutL` Struts inserted into left and right text lines.

```

\ledstrutR 1674 \newcommand*{\ledstrutL}{\strut}
1675 \newcommand*{\ledstrutR}{\strut}
1676

```

`\cleartoevenpage` `\cleartoevenpage`, which is defined in the memoir class, is like `\clear(double)page` except that we end up on an even page. `\cleartol@devenpage` is similar except that it first checks to see if it is already on an empty page. `\clearl@dleftpage` and `\clearl@drightpage` get us onto an odd and even page, respectively, checking that we end up on the immediately next page.

```

1677 \providecommand{\cleartoevenpage}[1][\@empty]{%
1678   \clearpage
1679   \ifodd\c@page\hbox{#1}\clearpage\fi}
1680 \newcommand*{\cleartol@devenpage}{%
1681   \ifdim\pagetotal<\topskip% on an empty page
1682     \else
1683       \clearpage
1684     \fi
1685   \ifodd\c@page\hbox{#1}\clearpage\fi}

```

```

1686 \newcommand*{\clearl@leftpage}{%
1687   \clearpage
1688   \ifodd\c@page\else
1689     \led@err@LeftOnRightPage
1690     \hbox{}%
1691   \cleardoublepage
1692   \fi}
1693 \newcommand*{\clearl@rightpage}{%
1694   \clearpage
1695   \ifodd\c@page
1696     \led@err@RightOnLeftPage
1697     \hbox{}%
1698   \cleartoevenpage
1699   \fi}
1700

```

`\getlinesfromparlistL` `\getlinesfromparlistL` gets the next entry from the `\linesinpar@listL` and `\@cs@linesinparL` puts it into `\@cs@linesinparL`; if the list is empty, it sets `\@cs@linesinparL` to 0. Similarly for `\getlinesfromparlistR`.

```

\@cs@linesinparR 1701 \newcommand*{\getlinesfromparlistL}{%
1702   \ifx\linesinpar@listL\empty
1703     \gdef\@cs@linesinparL{0}%
1704   \else
1705     \gl@p\linesinpar@listL\to\@cs@linesinparL
1706   \fi}
1707 \newcommand*{\getlinesfromparlistR}{%
1708   \ifx\linesinpar@listR\empty
1709     \gdef\@cs@linesinparR{0}%
1710   \else
1711     \gl@p\linesinpar@listR\to\@cs@linesinparR
1712   \fi}
1713

```

`\getlinesfrompagelistL` `\getlinesfrompagelistL` gets the next entry from the `\linesonpage@listL` and `\@cs@linesonpageL` puts it into `\@cs@linesonpageL`; if the list is empty, it sets `\@cs@linesonpageL` to 1000. Similarly for `\getlinesfrompagelistR`.

```

\@cs@linesonpageR 1714 \newcommand*{\getlinesfrompagelistL}{%
1715   \ifx\linesonpage@listL\empty
1716     \gdef\@cs@linesonpageL{1000}%
1717   \else
1718     \gl@p\linesonpage@listL\to\@cs@linesonpageL
1719   \fi}
1720 \newcommand*{\getlinesfrompagelistR}{%
1721   \ifx\linesonpage@listR\empty
1722     \gdef\@cs@linesonpageR{1000}%
1723   \else
1724     \gl@p\linesonpage@listR\to\@cs@linesonpageR
1725   \fi}
1726

```

`\@writelinesonpageL` These macros output the number of lines on a page to the section file in the form
`\@writelinesonpageR` of `\@lopL` or `\@lopR` macros.

```

1727 \newcommand*{\@writelinesonpageL}[1]{%
1728   \edef\next{\write\linenum@out{\string\@lopL{#1}}}%
1729   \next}
1730 \newcommand*{\@writelinesonpageR}[1]{%
1731   \edef\next{\write\linenum@outR{\string\@lopR{#1}}}%
1732   \next}
1733

```

`\l@dcalc@maxoftwo` `\l@dcalc@maxoftwo{<num>}{<num>}{<count>}` sets `<count>` to the maximum of
`\l@dcalc@minoftwo` the two `<num>`.

Similarly `\l@dcalc@minoftwo{<num>}{<num>}{<count>}` sets `<count>` to the minimum of the two `<num>`.

```

1734 \newcommand*{\l@dcalc@maxoftwo}[3]{%
1735   \ifnum #2>#1\relax
1736     #3=#2\relax
1737   \else
1738     #3=#1\relax
1739   \fi}
1740 \newcommand*{\l@dcalc@minoftwo}[3]{%
1741   \ifnum #2<#1\relax
1742     #3=#2\relax
1743   \else
1744     #3=#1\relax
1745   \fi}
1746

```

`\ifl@dsamepage` `\checkpageL` tests if the space and lines already taken on the page by text and foot-
`\l@dsamepagetrue` notes is less than the constraints. If so, then `\ifl@dpagetrue` is set FALSE and
`\l@dsamepagefalse` `\ifl@dsamepage` is set TRUE. If the page is spatially full then `\ifl@dpagetrue`
`\ifl@dpagetrue` is set TRUE and `\ifl@dsamepage` is set FALSE. If it is not spatially full but
`\l@dpagetrue` the maximum number of lines have been output then both `\ifl@dpagetrue` and
`\l@dpagetruefalse` `\ifl@dsamepage` are set FALSE.

```

\checkpageL 1747 \newif\ifl@dsamepage
\checkpageR 1748   \l@dsamepagetrue
1749   \newif\ifl@dpagetrue
1750   \newcommand*{\checkpageL}{%
1751     \l@dpagetrue
1752     \l@dsamepagetrue
1753     \check@goal
1754     \ifdim\pagetotal<\ledthegoal
1755       \ifnum\numpagelinesL<\l@dsamepagelines
1756         \else
1757           \l@dsamepagefalse
1758           \l@dpagetruefalse
1759         \fi
1760       \else

```

```

1761 \l@dsamepagefalse
1762 \l@dpagfulltrue
1763 \fi}
1764 \newcommand*{\checkpageR}{%
1765 \l@dpagfulltrue
1766 \l@dsamepagetrue
1767 \check@goal
1768 \ifdim\pagetotal<\ledthegoal
1769 \ifnum\numpagelinesR<\l@dminpagelines
1770 \else
1771 \l@dsamepagefalse
1772 \l@dpagfullfalse
1773 \fi
1774 \else
1775 \l@dsamepagefalse
1776 \l@dpagfulltrue
1777 \fi}
1778

```

`\ledthegoal` `\ledthegoal` is the amount of space allowed to taken by text and footnotes on a page before a forced pagebreak. This can be controlled via `\goalfraction`.
`\check@goal` `\ledthegoal` is calculated via `\check@goal`.

```

1779 \newdimen\ledthegoal
1780 \newcommand*{\goalfraction}{0.9}
1781 \newcommand*{\check@goal}{%
1782 \ledthegoal=\goalfraction\pagegoal}
1783

```

`\ifwrittenlinesL` Booleans for whether line data has been written to the section file.

```

\ifwrittenlinesL 1784 \newif\ifwrittenlinesL
1785 \newif\ifwrittenlinesR
1786

```

`\get@nextboxL` If the current box is not empty (i.e., still contains some lines) nothing is done.
`\get@nextboxR` Otherwise if and only if a synchronisation point is reached the next box is started.

```

1787 \newcommand*{\get@nextboxL}{%
1788 \ifvbox\namebox{\l@dLcolrawbox\the\l@dpscL}% box is not empty
    The current box is not empty; do nothing.
1789 \else%
    box is empty
    The box is empty; check if enough lines (real and blank) have been output.
1790 \ifnum\usenamecount{\l@dmaxlinesinpar\the\l@dpscL}>\@donetotallinesL
1791 \else
    Sufficient lines have been output.
1792 \ifwrittenlinesL
1793 \else

```


Write out the number of lines done, and set the boolean so this is only done once.

```

1794      \@writelinesinparL
1795      \writtenlinesLtrue
1796      \fi
1797      \ifnum\l@dnumpstartsL>\l@dpscL

    There are still unprocessed boxes. Recalculate the maximum number of lines
    needed, and move onto the next box (by incrementing \l@dpscL).

1798      \writtenlinesLfalse
1799      \l@dcalc@maxoftwo{\the\usernamecount{l@dmaxlinesinpar\the\l@dpscL}}%
1800                          {\the\@donetotallinesL}%
1801                          {\usernamecount{l@dmaxlinesinpar\the\l@dpscL}}%
1802      \global\@donetotallinesL \z@
1803      \global\advance\l@dpscL \@ne
1804      \fi
1805      \fi
1806      \fi}

1807 \newcommand*{\get@nextboxR}{%
1808   \ifvbox\namebox{l@dRcolrawbox\the\l@dpscR}% box is not empty
1809   \else% box is empty
1810     \ifnum\usernamecount{l@dmaxlinesinpar\the\l@dpscR}>\@donetotallinesR
1811     \else
1812       \ifwrittenlinesR
1813       \else
1814         \@writelinesinparR
1815         \writtenlinesRtrue
1816       \fi
1817       \ifnum\l@dnumpstartsR>\l@dpscR
1818       \writtenlinesRfalse
1819       \l@dcalc@maxoftwo{\the\usernamecount{l@dmaxlinesinpar\the\l@dpscR}}%
1820                           {\the\@donetotallinesR}%
1821                           {\usernamecount{l@dmaxlinesinpar\the\l@dpscR}}%
1822       \global\@donetotallinesR \z@
1823       \global\advance\l@dpscR \@ne
1824     \fi
1825     \fi
1826   \fi}
1827
```

25 The End

This is the end of the package code. But before we finish, enable a patch file (if there is one) to be read.

```

1828 \InputIfFileExists{ledparpatch.sty}
1829
1830 \</code>

```

A Examples

This section presents some sample documents.

The figures are from processed versions of the files. Having latexed a file I used DVIPS to get Encapsulated PostScript, then the `epstopdf` script to get a PDF version as well, for example:

```
> latex villon
> latex villon
> latex villon
> dvips -E -o villon.eps villon % produces villon.eps
> epstopdf villon.eps          % produces villon.pdf
```

For a multipage example, DVIPS has an option to output a range of pages (`-p` for the first and `-l` (letter l) for the last). For instance, to output a single page, say page 2:

```
> latex djd17nov
> latex djd17nov
> latex djd17nov
> dvips -E -p2 -l2 -o djd17novL.eps djd17nov % produces djd17novL.eps
> epstopdf djd17novL.eps                    % produces djd17novL.pdf
```

For those who aren't fascinated by LaTeX code, I show the all the typeset results first, then the code that produced them.

I thought that limericks were peculiarly English, but this appears not to be the case. As with most limericks this one is by Anonymous.

	Il y avait un jeune homme de Dijon,	There was a young man of Dijon,	1
2	Qui n'avait que peu de religion.	Who had only a little religion,	
	Il dit: 'Quant à moi,	He said: 'As for me,	3
4	Je déteste tous les trois,	I detest all the three,	
	Le Père, et le Fils, et le Pigeon.'	The Father, the Son, and the Pigeon.'	5

The following is verse LXXIII of François Villon's *Le Testament* (The Testament), composed in 1461.

	Dieu mercy et Tacque Thibault,	Thanks to God — and to Tacque Thibaud	
2	Qui tant d'eaue froid m'a fait boire,	Who made me drink so much cold water,	2r
	Mis en bas lieu, non pas en hault,	Put me underground instead of higher up	
4	Mengier d'angoisse maints poire,	And made me eat such bitter fruit,	4r
	Enferré ... Quant j'en ay memoire,	In chains ... When I think of this,	
6	Je Prie pour luy <i>et reliqua</i> ,	I pray for him— <i>et reliqua</i> ;	6r
	Que Dieu luy doint, et voire, voire!	May God grant him (yes, by God)	
8	Ce que je pense ... <i>et cetera</i> .	What I think ... <i>et cetera</i> .	8r

The translation and notes are by Anthony Bonner, *The Complete Works of François Villon*, published by Bantam Books in 1960.

4 poire d'angoisse] This has a triple meaning: literally it is the fruit of the choke pear, figuratively it means 'bitter fruit', and it also refers to a torture instrument.
6 *et reliqua*] and so on

1r Tacque Thibaud] A favourite of Jean, Duc de Berry and loathed for his exactions and debauchery. Villon uses his name as an insulting nickname for Thibaud d'Auxigny, the Bishop of Orléans.

2r cold water] Can either refer to the normal prison diet of bread and water or to a common medieval torture which involved forced drinking of cold water.

Figure 1: Output from `villon.tex`.

1 De ecclesia S. Stephani Novimagensi

Nobilis itaque comes Otto imperio et dominio Novimagensi sibi, ut praefer-
tur, impignoratis et commissis proinde praeesse cupiens, anno LIII superius
descripto, mense Iunio, una cum iudice, scabinis ceterisque civibus civitatis
Novimagensis, pro ipsius et inhabitantium in ea necessitate, commodo et utili-
tate, ut ecclesia eius parochialis extra civitatem sita destrueretur et infra muros
transferretur ac de novo construeretur, a reverendo patre domino Conrado de
Hofsteden, archiepiscopo Coloniensi, licentiam, et a venerabilibus dominis de-
cano et capitulo sanctorum Apostolorum Coloniensi, ipsius ecclesiae ab antiquo
veris et pacificis patronis, consensum, citra tamen praeiudicium, damnum aut
gravamen iurium et bonorum eorundem, impetravit.

Et exinde liberum locum eiusdem civitatis qui dicitur Hundisburg, de praeli-
bati Wilhelmi Romanorum regis, ipsius fundi domini, consensu, ad aedifican-
dum et consecrandum ecclesiam et coemeterium, eisdem decano et capitulo de
expresso eiusdem civitatis assensu libera contradiderunt voluntate, obligantes
se ipsi comes et civitas dictis decano et capitulo, quod in recompensationem
illius areae infra castrum et portam, quae fuit dos ecclesiae, in qua plebanus
habitare solebat—quae tunc per novum fossatum civitatis est destructa—aliam
aream competentem et ecclesiae novae, ut praefertur, aedificandae satis con-
tiguam, ipsi plebano darent et assignarent. Et desuper apud dictam ecclesiam
sanctorum Apostolorum est littera sigillis ipsorum Ottonis comitis et civitatis
Novimagensis sigillata.

// One additional line to show synchronization. //

3 p. 227 R 4 p. 97 N 6 p. 129 D 12 f. 72v M 13 p. 228 R 20 p. 130 D

2 proinde] primum D 5 ecclesia eius] ecclesia D: eius eius H extra civitatem om. H
infra] intra D 6 transferretur] transferreretur NH 7 Hofsteden] Hoffstede D: Hoffsteden
H Coloniensi] Colononiensi H dominis] viris H 8 Coloniensi] Coloniae H 10 iurium]
virium D 11 liberum] librum H qui] quae D Hundisburg] Hundisburch D: Hundisbrug
HMN: Hunsdisbrug R 12 regis] imperatoris D 13 et consecrandum om. H eisdem]
eiusdem D 15 comes] comites D dictis om. H 17 tunc] nunc H 18 ut...aedificandae
om. H 18–19 contiguam] contiguum M 19 apud om. H 20 est] et H littera] litteram
H 21 Novimagensis] Novimagii D sigillata] sigillis communita H

6–7 William is confusing two charters that are five years apart. Permission from St. Apostles’
Church in Cologne had been obtained as early as 1249. Cf. Sloet, *Oorkondenboek* nr. 707
(14 November 1249): “...nos devotionis tue precibus annuentes, ut ipsam ecclesiam faciens
demoliri transferas in locum alium competentem, tibi auctoritate presentium indulgemus...”
11–19 Cf. Sloet, *Oorkondenboek* nr. 762 (June 1254)

1 St. Stephen's Church in Nijmegen

After the noble count Otto had taken in pledge the power over Nijmegen,¹ like
 I have written above, he wanted to protect the town. So in June 1254 he and 1254
 the judge, the sheriffs and other citizens of Nijmegen obtained permission to
 demolish the parish church that lay outside the town walls,² to move it inside
 5 the walls and to rebuild it new. This operation was necessary and useful both for
 Otto himself and for the inhabitants of the town. The reverend father Conrad of
 Hochstaden, archbishop of Cologne,³ gave his permission. So did the reverend
 dean and canons of the chapter of St. Apostles' in Cologne, who had long⁴ been
 the true and benevolent patrons of the church—but they did not allow Otto to
 10 do anything without their knowledge, nor to infringe their rights, nor to damage
 their property.

And so the count and the town voluntarily gave an open space in town called
 Hundisburg, which was owned by the aforementioned king William, to the dean
 and chapter of St. Apostles' in order to build and consecrate a church and grave-
 15 yard. King William approved and the town of Nijmegen explicitly expressed its
 assent. A new ditch was dug on property of the church near the castle and the
 harbour,⁵ causing the demolition of the presbytery. In compensation, the count
 and citizens committed themselves to giving the parish priest another suitable
 space close enough to the new church that was about to be built. A letter about
 20 these transactions, with the seals of count Otto and the town of Nijmegen, is
 kept at St. Apostles' church.⁶

// One additional line to show synchronization. //

¹In 1247 William II (1227–1256) count of Holland needed money to fight his way to Aachen to be crowned King of the Holy Roman Empire. He gave the town of Nijmegen in pledge to Otto II (1229–1271) count of Guelders.

²Since the early seventh century old St. Stephen's church had been located close to the castle, at today's Kelfkensbos square. Traces of the church and the presbytery were found during excavations in 1998–1999.

³Conrad of Hochstaden († 1261) was archbishop of Cologne in 1238–1261. Nijmegen belonged to the archdiocese of Cologne until 1559.

⁴They probably became the patrons when the chapter was established in the early eleventh century. About the church and the chapter, see Gottfried Stracke, *Köln: St. Aposteln*, Stadtspuren – Denkmäler in Köln, vol. 19, Köln: J. P. Bachem, 1992.

⁵Nowadays, the exact location of the medieval ditch—and of two Roman ones—can be seen in the pavement of Kelfkensbos square.

⁶The original letter is lost. A 15th century transcription of it is kept at the Historisches Archiv der Stadt Köln (HASTK).

1 Arma gravi numero violentaque bella parabam
 2 edere, materiā conveniente modis.
 3 Par erat inferior versus—risisse Cupido
 4 dicitur atque unum surripuisse pedem.

 5 “Quis tibi, saeve puer, dedit hoc in carmina iuris?
 6 Pieridum vates, non tua turba sumus.
 7 Quid si praecripiat flavae Vēnus arma Minervae,
 8 ventilet accensas flava Minerva faces?

 9 Quis probet in silvis Cererem regnare iugosis,
 10 lege pharetratae Virginis arva coli?
 11 Crinibus insignem quis acuta cuspide Phoebum

 12 instruat, Aoniam Marte movente lyram?

6 sumus] note lost 11 acuta] acutā (abl. abs.)

Figure 4: First left page output from `djdpoems.tex`.

¹I.e., the even lines, which were hexameters (with six feet) became pentameters (with five feet).

²Muses

³Ceres was the Roman goddess of the harvest.

⁴By '*Virgo*' ('Virgin') Ovid means Diana, the Roman goddess of the hunt.

⁵Lines 7R–12R show some paradoxical situations that would occur if the gods didn't stay with their own business.

12R Aonian] Mount Parnassus, where the Muses live, is located in Aonia.

1 Arma gravi numero violentaque bella parabam
 2 edere, materiā conveniente modis.
 3 Par erat inferior versus—risisse Cupido
 4 dicitur atque unum surripuisse pedem.

 5 “Quis tibi, saeve puer, dedit hoc in carmina iuris?
 6 Pieridum vates, non tua turba sumus.
 7 Quid si praecripiat flavae Vēnus arma Minervae,
 8 ventilet accensas flava Minerva faces?

 9 Quis probet in silvis Cererem regnare iugosis,
 10 lege pharetratae Virginis arva coli?
 11 Crinibus insignem quis acuta cuspide Phoebum
 12 instruat, Aoniam Marte movente lyram?

6 sumus] note lost 11 acuta] acutā (abl. abs.)

Figure 6: Second left page output from `djdpoe.ms.tex`.

⁶I.e., the even lines, which were hexameters (with six feet) became pentameters (with five feet).

⁷Muses

⁸Ceres was the Roman goddess of the harvest.

⁹By '*Virgo*' ('Virgin') Ovid means Diana, the Roman goddess of the hunt.

¹⁰Lines 7R–12R show some paradoxical situations that would occur if the gods didn't stay with their own business.

12R Aonian] Mount Parnassus, where the Muses live, is located in Aonia.

A.1 Parallel column example

This made-up example, `villon.tex`, is included to show parallel columns and how they can be interspersed in regular text. The verses are set using the `\stanza` construct, where each verse line is a chunk. The code is given below and the result is shown in Figure 1.

```

1831 (*villon)
1832 %% villon.tex Example parallel columns
1833 \documentclass{article}
1834 \addtolength{\textheight}{-10\baselineskip}
1835 \usepackage{ledmac,ledpar}
1836 %% Use r instead of R to flag right text line numbers
1837 \renewcommand{\Rlineflag}{r}
1838 %% Use the flag in the notes
1839 \let\oldBfootfmt\Bfootfmt
1840 \renewcommand{\Bfootfmt}[3]{%
1841   \let\printlines\printlinesR
1842   \oldBfootfmt{#1}{#2}{#3}}
1843 \begin{document}
1844
1845 I thought that limericks were peculiarly English, but this appears not
1846 to be the case. As with most limericks this one is by Anonymous.
1847
1848 \vspace*{\baselineskip}
1849
1850 \begin{pairs}
1851 %% no indentation
1852 \setstanzaindents{0,0,0,0,0,0,0,0}
1853 %% no number flag
1854 \renewcommand{\Rlineflag}{}
1855 %% draw a rule and widen the columns
1856 \setlength{\columnrulewidth}{0.4pt}
1857 \setlength{\Lcolwidth}{0.46\textwidth}
1858 \setlength{\Rcolwidth}{\Lcolwidth}
1859
1860 \begin{Leftside}
1861 %% set left text line numbering sequence
1862 \firstlinenum{2}
1863 \linenumincrement{2}
1864 \linenummargin{left}
1865 \beginnumbering
1866 \stanza
1867 Il y avait un jeune homme de Dijon, &
1868 Qui n'avait que peu de religion. &
1869 Il dit: 'Quant \'{a} moi, &
1870 Je d\'{e}teste tous les trois, &
1871 Le P\'{e}re, et le Fils, et le Pigeon.' \&
1872 \endnumbering
1873 \end{Leftside}

```

```

1874
1875 \begin{Rightside}
1876 %% different right text line numbering sequence
1877 \firstlinenum{1}
1878 \linenumincrement{2}
1879 \linenummargin{right}
1880 \beginnumbering
1881 \stanza
1882 There was a young man of Dijon, &
1883 Who had only a little religion, &
1884 He said: 'As for me, &
1885 I detest all the three, &
1886 The Father, the Son, and the Pigeon.' \&
1887 \endnumbering
1888 \end{Rightside}
1889
1890 \Columns
1891 \end{pairs}
1892
1893 \vspace*{\baselineskip}
1894
1895     The following is verse \textsc{lxixiii} of Fran\c{c}ois Villon's
1896 \textit{Le Testament} (The Testament), composed in 1461.
1897
1898 %% Allow for hanging indentation for long lines
1899 \setstanzaindents{1,0,0,0,0,0,0,0}
1900 %% Columns wider than the default
1901 \setlength{\Lcolwidth}{0.46\textwidth}
1902 \setlength{\Rcolwidth}{\Lcolwidth}
1903 \vspace*{\baselineskip}
1904
1905 \begin{pairs}
1906 \begin{Leftside}
1907 \firstlinenum{2}
1908 \linenumincrement{2}
1909 \linenummargin{left}
1910 \beginnumbering
1911 \stanza
1912 Dieu mercy et Tacque Thibault, &
1913 Qui tant d'eaue froid m'a fait boire, &
1914 Mis en bas lieu, non pas en hault, &
1915 Mengier d'angoisse maints \edtext{poire}{\lemma{poire d'angoisse}}%
1916 \Afootnote{This has a triple meaning: literally it is the fruit of the
1917  choke pear,
1918  figuratively it means 'bitter fruit', and it also refers to a torture
1919  instrument.}}, &
1920 Enferr'\{e} \ldots Quant j'en ay memoire, &
1921 Je Prie pour luy \edtext{\textit{et reliqua}}{\Afootnote{and so on}}, &
1922 Que Dieu luy doint, et voire, voire! &
1923 Ce que je pense \ldots \textit{et cetera}. \&

```

```

1924 \endnumbering
1925 \end{Leftside}
1926
1927 \begin{Rightside}
1928 \firstlinenum{2}
1929 \linenumincrement{2}
1930 \linenummargin{right}
1931 \beginnumbering
1932 \stanza
1933 Thanks to God --- and to \edtext{Tacque Thibaud}{%
1934   \Bfootnote{A favourite of Jean, Duc de Berry and loathed for his exactions
1935     and debauchery. Villon uses his name as an insulting nickname for
1936     Thibaud d'Auxigny, the Bishop of Orl'\{e}ans.}} &
1937 Who made me drink so much \edtext{cold water}{%
1938   \Bfootnote{Can either refer to the normal prison diet of bread and
1939     water or to a common medieval torture which involved forced drinking
1940     of cold water.}}, &
1941 Put me underground instead of higher up &
1942 And made me eat such bitter fruit, &
1943 In chains \ldots When I think of this, &
1944 I pray for him---\textit{et reliqua;} &
1945 May God grant him (yes, by God) &
1946 What I think \ldots \textit{et cetera}. \&
1947 \endnumbering
1948 \end{Rightside}
1949
1950 \Columns
1951 \end{pairs}
1952
1953 \vspace*{\baselineskip}
1954
1955   The translation and notes are by Anthony Bonner,
1956 \textit{The Complete Works of Fran\c{c}ois Villon}, published by
1957 Bantam Books in 1960.
1958
1959 \end{document}
1960
1961 </villon>

```

A.2 Example parallel facing pages

This example, illustrated in Figures 2 and 3, was provided in November 2004 by Dirk-Jan Dekker of the Department of Medieval History at Radboud University, Nijmegen.

```

1962 <*djd17nov>
1963 %%% This is djd17nov.tex, a sample critical text edition
1964 %%% written in LaTeX2e with the ledmac and ledpar packages.
1965 %%% (c) 2003--2004 by Dr. Dirk-Jan Dekker,

```

```

1966 %%% Radboud University, Nijmegen (The Netherlands)
1967 %%% (PRW) Modified slightly by PRW to fit the ledpar manual
1968
1969 \documentclass[10pt, letterpaper, twoside]{article}
1970 \usepackage[latin,english]{babel}
1971 \usepackage{makeidx}
1972 \usepackage{ledmac,ledpar}
1973 \lineation{section}
1974 \linenummargin{inner}
1975 \sidenotemargin{outer}
1976
1977 \makeindex
1978
1979 \renewcommand{\notenumfont}{\footnotesize}
1980 \newcommand{\notetextfont}{\footnotesize}
1981
1982 %\let\Afootnoterule=\relax
1983 \let\Bfootnoterule=\relax
1984 \let\Cfootnoterule=\relax
1985
1986 \addtolength{\skip\Afootins}{1.5mm}
1987 %\addtolength{\skip\Bfootins}{1.5mm}
1988 %\addtolength{\skip\Cfootins}{1.5mm}
1989
1990 \makeatletter
1991
1992 \renewcommand*{\para@vfootnote}[2]{%
1993   \insert\csname #1footins\endcsname
1994   \bgroup
1995     \notefontsetup
1996     \interlinepenalty=\interfootnotelinepenalty
1997     \floatingpenalty=\@MM
1998     \splittopskip=\ht\strutbox \splitmaxdepth=\dp\strutbox
1999     \leftskip=\z@skip \rightskip=\z@skip
2000     \l@dparsfootspec #2\ledplinenumtrue%           new from here
2001     \ifnum\@nameuse{previous@#1@number}=\l@dparsestartline\relax
2002       \ledplinenumfalse
2003       \fi
2004       \ifnum\previous@page=\l@dparsestartpage\relax
2005       \else \ledplinenumtrue \fi
2006       \ifnum\l@dparsestartline=\l@dparseendline\relax
2007       \else \ledplinenumtrue \fi
2008       \expandafter\xdef\csname previous@#1@number\endcsname{\l@dparsestartline}%
2009       \xdef\previous@page{\l@dparsestartpage}%           to here
2010       \setbox0=\vbox{\hsize=\maxdimen
2011         \noindent\csname #1footfmt\endcsname#2}%
2012       \setbox0=\hbox{\unvbox0}%
2013       \dp0=0pt
2014       \ht0=\csname #1footfudgefactor\endcsname\wd0
2015       \box0

```

```

2016     \penalty0
2017   \egroup
2018 }
2019
2020 \newcommand*{\previous@A@number}{-1}
2021 \newcommand*{\previous@B@number}{-1}
2022 \newcommand*{\previous@C@number}{-1}
2023 \newcommand*{\previous@page}{-1}
2024
2025 \newcommand{\abb}[1]{#1%
2026     \let\rbracket\nobrak\relax}
2027 \newcommand{\nobrak}{\textnormal{}}
2028 \newcommand{\morenoexpands}{%
2029     \let\abb=0%
2030 }
2031
2032 \newcommand{\Aparafootfmt}[3]{%
2033     \ledsetnormalparstuff
2034     \scriptsize
2035     \notenumfont\printlines#1\enspace
2036 %   \lemmafонт#1|#2\enskip
2037     \notetextfont
2038     #3\penalty-10\hskip 1em plus 4em minus.4em\relax}
2039
2040 \newcommand{\Bparafootfmt}[3]{%
2041     \ledsetnormalparstuff
2042     \scriptsize
2043     \notenumfont\printlines#1|%
2044     \ifledplinenum
2045     \enspace
2046     \else
2047     {\hskip 0em plus 0em minus .3em}%
2048     \fi
2049     \select@lemmafонт#1|#2\rbracket\enskip
2050     \notetextfont
2051     #3\penalty-10\hskip 1em plus 4em minus.4em\relax }
2052
2053 \newcommand{\Cparafootfmt}[3]{%
2054     \ledsetnormalparstuff
2055     \scriptsize
2056     \notenumfont\printlines#1\enspace
2057 %   \lemmafонт#1|#2\enskip
2058     \notetextfont
2059     #3\penalty-10\hskip 1em plus 4em minus.4em\relax}
2060
2061 \makeatother
2062
2063 \footparagraph{A}
2064 \footparagraph{B}
2065 \footparagraph{C}

```

```

2066
2067 \let\Afootfmt=\Aparafootfmt
2068 \let\Bfootfmt=\Bparafootfmt
2069 \let\Cfootfmt=\Cparafootfmt
2070
2071 \renewcommand*{\Rlineflag}{}
2072
2073 \emergencystretch40pt
2074
2075 \author{Guillelmus de Berchen}
2076 \title{Chronicon Geldriae}
2077 \date{}
2078 \hyphenation{archi-epi-sco-po Huns-dis-brug li-be-ra No-vi-ma-gen-si}
2079 \begin{document}
2080 \begin{pages}
2081 \begin{Leftside}
2082 \beginnumbering\pstart
2083 \selectlanguage{latin}
2084 \section{De ecclesia S. Stephani Novimagensi}
2085
2086 \noindent\setline{1}
2087 Nobilis itaque comes Otto\protect\edindex{Otto II of Guelders}
2088 imperio et dominio Novimagensi sibi, ut praeferetur, impignoratis
2089 et commissis
2090 \edtext{proinde}{\Bfootnote{primum D}} praeesse cupiens, anno
2091 \textsc{liiii} superius descripto, mense
2092 \u\edtext{}{\Afootnote{p.\ 227~R}}nio, una cum iudice, scabinis ceterisque
2093 civibus civitatis Novimagensis, pro ipsius et inhabitantium in ea
2094 necessitate,\edtext{}{\Afootnote{p.\ 97~N}} commodo et utilitate,
2095 ut \edtext{ecclesia eius}{\Bfootnote{ecclesia D: eius eius H}} parochialis
2096 \edtext{\abb{extra civitatem}}{\Bfootnote{\textit{om.}~H}} sita
2097 destrueretur et \edtext{infra}{\Bfootnote{intra D}} muros
2098 \edtext{transfer\edtext{}{\Afootnote{p.\ 129~D}}retur}%
2099 {\Bfootnote{transferreretur NH}}
2100 ac de novo construeretur,
2101 \edtext{a reverendo patre domino
2102 Conrado\protect\edindex{Conrad of Hochstaden} de
2103 \edtext{Hofsteden}{\Bfootnote{Hoffstede D: Hoffsteden H}}, archiepiscopo
2104 \edtext{Coloniensi}{\Bfootnote{Colononiensi H}}, licentiam}%
2105 {\Cfootnote{William is confusing two charters that are five years
2106 apart. Permission from St.\ Apostles' Church in Cologne had been
2107 obtained as early as 1249. Cf.\
2108 Sloet\protect\index{Sloet van de Beele, L.A.J.W.},
2109 \textit{Oorkondenboek} nr.\ 707 (14 November 1249):
2110 ‘‘\ldots}nos devotionis tue precibus annuentes, ut ipsam ecclesiam
2111 faciens demoliri transferas in locum alium competentem, tibi
2112 auctoritate presentium indulgemus\ldots’’}}, et a venerabilibus
2113 \edtext{dominis}{\Bfootnote{viris H}} decano et capitulo sanctorum
2114 Apostolorum\protect\edindex{St. Apostles' (Cologne)}
2115 \edtext{Coloniensi}{\Bfootnote{Coloniae H}}, ipsius ecclesiae ab

```

```

2116 antiquo veris et pacificis patronis, consensum, citra tamen
2117 praeiudicium, damnum aut gravamen \edtext{iurium}{\Bfootnote{virium D}}
2118 et bonorum eorundem, impetravit.
2119 \pend
2120
2121 \pstart
2122 \edtext{Et exinde \edtext{liberum}{\Bfootnote{librum H}}
2123 locum eiusdem civitatis
2124 \edtext{qui}{\Bfootnote{quae D}} dicitur
2125 \edtext{Hundisburg}{\Bfootnote{Hundisburch D: Hundisbrug HMN:
2126 Hunsdisbrug R}}\protect\edindex{Hundisburg},
2127 de praelibati Wilhelmi\protect\edindex{William II of Holland} Romanorum
2128 \edtext{regis}{\Bfootnote{imperatoris D}}, ipsius fundi
2129 do\edtext{}{\Afootnote{f.\ 72v~M}}mini, consensu, ad aedificandum
2130 \edtext{\abb{et consecrandum}}{\Bfootnote{\textit{om.}\ H}}
2131 ecclesi\edtext{}{\Afootnote{p.\ 228~R}}am et coemeterium,
2132 \edtext{eisdem}{\Bfootnote{eiusdem D}} decano et capitulo de expresso
2133 eiusdem civitatis assensu libera contradiderunt voluntate, obligantes
2134 se ipsi \edtext{comes}{\Bfootnote{comites D}} et civitas
2135 \edtext{\abb{dictis}}{\Bfootnote{\textit{om.}\ H}} decano et capitulo,
2136 quod in recompensationem illius areae infra castrum et portam, quae
2137 fuit dos ecclesiae, in qua plebanus habitare solebat---quae
2138 \edtext{tunc}{\Bfootnote{nunc H}} per novum fossatum civitatis est
2139 destructa---aliam aream competentem et ecclesiae novae,
2140 \edtext{ut praeferitur, aedificandae}{\%
2141 \lemma{\abb{ut\ldots aedificandae}}{\Bfootnote{\textit{om.}\ H}} satis
2142 \edtext{contiguam}{\Bfootnote{contiguum M}}, ipsi plebano darent et
2143 assignarent.}{\Cfootnote{Cf.\ Sloet, \textit{Oorkondenboek} nr.\ 762
2144 (June 1254)}} Et desuper
2145 \edtext{\abb{apud}}{\Bfootnote{\textit{om.}\ H}} dictam ecclesiam
2146 sanctorum Apostolorum \edtext{est}{\Bfootnote{et H}}
2147 \edtext{littera}{\Bfootnote{litteram H}} sigillis ipsorum
2148 Ottonis\edtext{}{\Afootnote{p.\ 130~D}} comitis et civitatis
2149 \edtext{Novimagensis}{\Bfootnote{Novimagii D}}
2150 \edtext{sigillata}{\Bfootnote{sigillis communita H}}.
2151 \pend
2152
2153 \pstart
2154 // One additional line to show synchronization. //
2155 \pend
2156 \endnumbering
2157 \end{Leftside}
2158
2159 \begin{Rightside}
2160 \sidenotemargin{right}\selectlanguage{english}
2161 \beginnumbering
2162 \pstart
2163 \addtocounter{section}{-1}%
2164 \leavevmode\section{St.\ Stephen's Church in Nijmegen}
2165

```


2166 \noindent\setline{1}%
 2167 After the noble count Otto had taken in pledge the power over
 2168 Nijmegen,\footnote{In 1247 William II\protect\index{William II of Holland}
 2169 (1227--1256) count of Holland needed money to fight his way to
 2170 Aachen\protect\index{Aachen} to be crowned King of the Holy Roman
 2171 Empire. He gave the town of Nijmegen in pledge to Otto
 2172 II\protect\index{Otto II of Guelders} (1229--1271) count of Guelders.}
 2173 like I have written above, he wanted to protect the town. So in June
 2174 1254\ledsidenote{1254} he and the judge, the sheriffs and other
 2175 citizens of Nijmegen obtained permission to demolish the parish
 2176 church that lay outside the town walls,\footnote{Since the early
 2177 seventh century old St.\ Stephen's church had been located close
 2178 to the castle, at today's
 2179 Kelfkensbos\protect\index{Kelfkensbos (Nijmegen)} square.
 2180 Traces of the church and the presbytery were found during excavations
 2181 in 1998--1999.} to move it inside the walls and to rebuild it new.
 2182 This operation was necessary and useful both for Otto himself and
 2183 for the inhabitants of the town. The reverend father Conrad of
 2184 Hochstaden, archbishop of
 2185 Cologne,\footnote{Conrad of Hochstaden ({\textdag} 1261) was
 2186 archbishop of Cologne in 1238--1261. Nijmegen belonged to the
 2187 archdiocese of Cologne until 1559.} gave his permission. So did the
 2188 reverend dean and canons of the chapter of St.\
 2189 Apostles'\protect\index{St. Apostles' (Cologne)} in Cologne, who had
 2190 long\footnote{They probably became the patrons when the chapter was
 2191 established in the early eleventh century. About the church and the
 2192 chapter, see Gottfried Stracke\protect\index{Stracke, G.},
 2193 \textit{K}\{o\}ln:\ St.\ Aposteln}, Stadtpuren -- Denkm\{a\}ler in
 2194 K\{o\}ln, vol.\ 19, K\{o\}ln: J.\,P.\ Bachem, 1992.} been the true
 2195 and benevolent patrons of the church---but they did not allow Otto
 2196 to do anything without their knowledge, nor to infringe their rights,
 2197 nor to damage their property.
 2198 \pend
 2199
 2200 \pstart
 2201 And so the count and the town voluntarily gave an open space in town
 2202 called Hundisburg, which was owned by the aforementioned king William,
 2203 to the dean and chapter of St.\ Apostles' in order to build and
 2204 consecrate a church and graveyard. King William approved and the
 2205 town of Nijmegen explicitly expressed its assent. A new ditch was dug
 2206 on property of the church near the castle and the
 2207 harbour,\footnote{Nowadays, the exact location of the medieval
 2208 ditch---and of two Roman ones---can be seen in the pavement of
 2209 Kelfkensbos\protect\index{Kelfkensbos (Nijmegen)} square.} causing
 2210 the demolition of the presbytery. In compensation, the count and
 2211 citizens committed themselves to giving the parish priest another
 2212 suitable space close enough to the new church that was about to be
 2213 built. A letter about these transactions, with the seals of count
 2214 Otto and the town of Nijmegen, is kept at St.\ Apostles'
 2215 church.\footnote{The original letter is lost. A 15th century

```

2216 transcription of it is kept at the Historisches Archiv der
2217 Stadt K\{"o}ln (HASTK).}
2218 \pend
2219
2220 \pstart
2221 // One additional line to show synchronization. //
2222 \pend
2223 \endnumbering
2224 \end{Rightside}
2225 \Pages
2226 \end{pages}
2227
2228 %%%%%%%%%%%
2229 \printindex
2230 \end{document}
2231 %%%%%%%%%%%
2232
2233 </djd17nov>

```

A.3 Example poetry on parallel facing pages

This example, illustrated in Figures 4 to 7, was originally provided in November 2004 by Dirk-Jan Dekker for an earlier version of `ledpar`. I have updated it, and also extended it to show the difference between the `\stanza` command and the `astanza` environment. `\stanza` is used for the first pair of pages and `astanza` for the second pair. Note the definition of `\endstanzaextra` to give a short line after each stanza.

```

2234 (*djdpoems)
2235 %% djdpoems.tex  example parallel verses on facing pages
2236 \documentclass{article}
2237 \usepackage{ledmac,ledpar}
2238 \addtolength{\textheight}{-15\baselineskip}
2239
2240 \maxchunks{24} % default value = 10
2241 \setstanzaindent{6,0,1,0,1}
2242
2243 \newcommand{\longdash}{-----}
2244
2245 \footparagraph{A} % for left pages
2246 \footparagraph{B} % for right pages
2247 \firstlinenum{1}
2248 \linenumincrement{1}
2249
2250 \let\oldBfootfmt\Bfootfmt
2251 \renewcommand{\Bfootfmt}[3]{%
2252   \let\printlines\printlinesR
2253   \oldBfootfmt{#1}{#2}{#3}}

```

```

2254
2255 \begin{document}
2256
2257 \newcommand{\interstanza}{\pstart\centering\longdash\skipnumbering\pend}
2258
2259 \begin{pages}
2260 \begin{Leftside}
2261 \def\endstanzaextra{\interstanza}
2262 \beginnumbering
2263
2264 \stanza
2265 Arma gravi numero violentaque bella parabam &
2266   edere, materi\={a} conveniente modis. &
2267 Par erat inferior versus---risisse Cupido &
2268 dicitur atque unum surripuisse pedem. \&
2269
2270 \stanza
2271 ‘Quis tibi, saeve puer, dedit hoc in carmina iuris? &
2272 Pieridum vates, non tua turba \edtext{sumus}{\Afootnote{note lost}}. &
2273 Quid si praeripiat flavae V\{e}nus arma Minervae, &
2274 ventilet accensas flava Minerva faces? \&
2275
2276 \stanza
2277 Quis probet in silvis Cererem regnare iugosis, &
2278 lege pharetratae Virginis arva coli? &
2279 Crinibus insignem quis \edtext{acuta}{\Afootnote{acut\={a} (abl.\ abs.)}}
2280 cuspidе Phoebum &
2281 instruat, Aoniam Marte movente lyram? \&
2282 \endnumbering
2283 \end{Leftside}
2284
2285 \begin{Rightside}
2286 \def\endstanzaextra{\interstanza}
2287 \beginnumbering
2288 \firstlinenum{1}
2289 \linenumincrement{1}
2290 \setstanzaindents{6,0,1,0,1,0}
2291
2292 \stanza
2293 I was preparing to sing of weapons and violent wars, &
2294 in heavy numbers, with the subject matter suited to the verse measure. &
2295 The even lines were as long as the odd ones, but Cupid laughed, &
2296 they said, and he stole away one foot.\footnote{I.e., the even lines,
2297 which were hexameters (with six feet) became pentameters
2298 (with five feet).} \&
2299
2300 \stanza
2301 ‘O cruel boy, who gave you the right over poetry? &
2302 We poets belong to the Pierides,\footnote{Muses} we are not your folk. &
2303 \edlabel{beginparadox}What if Venus should seize away the arms of

```

```

2304 Minerva with the golden hair, &
2305 if Minerva with the golden hair should fan alight the kindled torch
2306 of love? \&
2307
2308 \stanza
2309 Who would approve of Ceres\footnote{Ceres was the Roman goddess of
2310 the harvest.} reigning on the woodland ridges, &
2311 and of land tilled under the law of the Maid with the
2312 quiver\footnote{By '\textit{Virgo}' ('Virgin') Ovid means Diana, the
2313 Roman goddess of the hunt.}? &
2314 Who would provide Phoebus with his beautiful hair with a sharp-pointed
2315 spear, &
2316 while Mars stirs the \edtext{Aonian}{\Bfootnote{Mount Parnassus,
2317 where the Muses live, is located in Aonia.}}
2318 lyre?\edlabel{endparadox}\footnote{Lines
2319 \xlineref{beginparadox}--\xlineref{endparadox} show some paradoxical
2320 situations that would occur if the gods didn't stay with their own
2321 business.} \&
2322 \endnumbering
2323 \end{Rightside}
2324
2325 \Pages
2326 \end{pages}
2327
2328 \begin{pages}
2329 \begin{Leftside}
2330 \def\endstanzaextra{\interstanza}
2331 \beginnumbering
2332
2333 \begin{astanza}
2334 Arma gravi numero violentaque bella parabam &
2335 edere, materi\={a} conveniente modis. &
2336 Par erat inferior versus---risisse Cupido &
2337 dicitur atque unum surripuisse pedem. \&
2338 \end{astanza}
2339
2340 \begin{astanza}
2341 'Quis tibi, saeve puer, dedit hoc in carmina iuris? &
2342 Pieridum vates, non tua turba \edtext{sumus}{\Afootnote{note lost}}. &
2343 Quid si praeripiat flavae V\ufe{nus arma Minervae, &
2344 ventilet accensas flava Minerva faces? \&
2345 \end{astanza}
2346
2347 \begin{astanza}
2348 Quis probet in silvis Cererem regnare iugosis, &
2349 lege pharetratae Virginis arva coli? &
2350 Crinibus insignem quis \edtext{acuta}{\Afootnote{acut\={a} (abl.\ abs.)}}
2351 cuspidem Phoebum &
2352 instruat, Aoniam Marte movente lyram? \&
2353 \end{astanza}

```

```

2354
2355 \endnumbering
2356 \end{Leftside}
2357
2358 \begin{Rightside}
2359 \def\endstanzaextra{\interstanza}
2360 \beginnumbering
2361 \firstlinenum{1}
2362 \linenumincrement{1}
2363 \setstanzaindent{6,0,1,0,1,0}
2364
2365 \begin{astanza}
2366 I was preparing to sing of weapons and violent wars, &
2367 in heavy numbers, with the subject matter suited to the verse measure. &
2368 The even lines were as long as the odd ones, but Cupid laughed, &
2369 they said, and he stole away one foot.\footnote{I.e., the even lines,
2370 which were hexameters (with six feet) became pentameters
2371 (with five feet).} \&
2372 \end{astanza}
2373
2374 \begin{astanza}
2375 ‘‘O cruel boy, who gave you the right over poetry? &
2376 We poets belong to the Pierides,\footnote{Muses} we are not your folk. &
2377 \edlabel{beginparadox}What if Venus should seize away the arms of
2378 Minerva with the golden hair, &
2379 if Minerva with the golden hair should fan alight the kindled torch
2380 of love? \&
2381 \end{astanza}
2382
2383 \begin{astanza}
2384 Who would approve of Ceres\footnote{Ceres was the Roman goddess of the
2385 harvest.} reigning on the woodland ridges, &
2386 and of land tilled under the law of the Maid with the
2387 quiver\footnote{By ‘\textit{Virgo}’ (‘Virgin’) Ovid means Diana,
2388 the Roman goddess of the hunt.}? &
2389 Who would provide Phoebus with his beautiful hair with a sharp-pointed
2390 spear, &
2391 while Mars stirs the \edtext{Aonian}{\Bfootnote{Mount Parnassus, where
2392 the Muses live, is located in Aonia.}}
2393 lyre?\edlabel{endparadox}\footnote{Lines
2394 \xlineref{beginparadox}--\xlineref{endparadox} show some paradoxical
2395 situations that would occur if the gods didn’t stay with their
2396 own business.} \&
2397 \end{astanza}
2398
2399 \endnumbering
2400 \end{Rightside}
2401
2402 \Pages
2403 \end{pages}

```

```
2404
2405 \end{document}
2406
2407 </djdpoems>
```

References

- [LW90] John Lavagnino and Dominik Wujastyk. ‘An overview of EDMAC: a PLAIN TeX format for critical editions’. *TUGboat*, **11**, 4, pp. 623–643, November 1990. (Code available from CTAN in `macros/plain/contrib/edmac`)
- [Wil02] Peter Wilson. *The memoir class for configurable typesetting*. November 2002. (Available from CTAN in `macros/latex/contrib/memoir`)
- [Wil04] Peter Wilson. *ledmac A presumptuous attempt to port EDMAC, TABMAC and EDSTANZA to LaTeX*. December 2004. (Available from CTAN in `macros/latex/contrib/ledmac`)

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.

Symbols	
<code>\&</code> . . .	1336, 1337, 1341, 1358, 1372, 1871, 1886, 1923, 1946, 2268, 2274, 2281, 2298, 2306, 2321, 2337, 2344, 2352, 2371, 2380, 2396
<code>\@M</code>	1347
<code>\@MM</code>	1997
<code>\@adv</code>	<u>322</u> , 604, 605
<code>\@arabic</code>	182, 183
<code>\@astanza@line</code>	1357, 1363, <u>1366</u>
<code>\@auxout</code>	1199, 1209, 1446
<code>\@cs@linesinparL</code>	1602, <u>1701</u>
<code>\@cs@linesinparR</code>	1602, <u>1701</u>
<code>\@cs@linesonpageL</code>	1610, 1660, <u>1714</u>
<code>\@cs@linesonpageR</code>	1610, 1660, <u>1714</u>
<code>\@donereallinesL</code> <u>818</u> , 844, 1566, 1568, 1616
<code>\@donereallinesR</code> <u>818</u> , 875, 1571, 1573, 1618
<code>\@donetotallinesL</code>	<u>818</u> , 845, 848, 1617, 1790, 1800, 1802
<code>\@donetotallinesR</code>	<u>818</u> , 876, 879, 1619, 1810, 1820, 1822
<code>\@insertR</code>	1015–1017, 1030–1032
<code>\@l</code>	<u>248</u> , 573
<code>\@l@dttempcnta</code>	395, 397, 399, 400, 404, 406, 408, 409, 917, 918, 920, 922, 925, 926, 947–951, 953, 960–964, 966, 969, 972, 974, 978
<code>\@l@dttempcntb</code>	141, 143, 145, 908, 909, 945, 957, 978, 986–988, 990, 1237, 1239, 1241, 1294, 1295, 1297, 1299, 1302, 1400–1404, 1408–1411, 1415–1418
<code>\@l@reg</code>	294
<code>\@lab</code>	529, 1193, 1202, <u>1223</u>
<code>\@lock</code>	57, 272, 274, 276, 289, 444, 445, 447, 461, 462, 464, 481, 482, 484, 498, 499, 501, 891
<code>\@lopL</code>	<u>551</u> , 1728
<code>\@lopR</code>	<u>551</u> , 1731
<code>\@nameuse</code>	1326, 1330, 2001
<code>\@pend</code>	<u>544</u> , 1566
<code>\@pendR</code>	<u>544</u> , 1571
<code>\@pstartsfalse</code>	<u>1541</u>
<code>\@pstartstrue</code>	<u>1541</u>
<code>\@ref</code>	<u>516</u> , 577, 581
<code>\@ref@reg</code>	542

`\@set` 354, 611, 612
`\@tag` 642, 659,
 1039, 1043, 1053, 1057, 1067,
 1071, 1081, 1085, 1095, 1099,
 1110, 1114, 1124, 1128, 1138,
 1142, 1152, 1156, 1166, 1170,
 1250, 1254, 1262, 1266, 1274, 1278
`\@temp` 1437
`\@templod` 1285, 1286
`\@writelinesinparL` .. 1523, 1564, 1794
`\@writelinesinparR` .. 1524, 1564, 1814
`\@writelinesonpageL` . 1634, 1636, 1727
`\@writelinesonpageR` . 1650, 1652, 1727
`\@xloop` 1028
`\affixside@noteR` 869, 1284
`\Afootfmt` 2067
`\Afootins` 1986
`\Afootnote` 1035, 1916,
 1921, 2092, 2094, 2098, 2129,
 2131, 2148, 2272, 2279, 2342, 2350
`\Afootnoterule` 1982
`\Aparafootfmt` 2032, 2067
`\apptocmd` 1477
`\araw@textfalse` 1552
`\araw@texttrue` 1552
`astanza (environment)` 7, 1339
`\AtBeginDocument` ... 1219, 1422, 1457
`\author` 2075

B

`_` 2092, 2094, 2098, 2106, 2107, 2109,
 2129–2131, 2135, 2141, 2143,
 2145, 2148, 2164, 2177, 2188,
 2193, 2194, 2203, 2214, 2279, 2350
A
`\abb` 2025,
 2029, 2096, 2130, 2135, 2141, 2145
`\absline@num` .. 388, 402, 421, 451, 488
`\absline@numR` ... 54, 199, 252, 255,
 258, 385, 393, 414, 434, 471,
 510, 521, 883, 899, 900, 908, 1014
`\actionlines@list`
 . 240, 243, 388, 402, 421, 451, 488
`\actionlines@listR`
 203, 218, 232, 235, 385,
 393, 414, 434, 471, 510, 930, 933
`\actions@list` 244, 389,
 409, 423, 425, 453, 460, 490, 497
`\actions@listR`
 . 203, 219, 236, 386, 400, 416,
 418, 436, 443, 473, 480, 511, 934
`\add@inserts` 837
`\add@inserts@nextR` 1003
`\add@insertsR` 868, 1003
`\add@penaltiesL` 843, 1024
`\add@penaltiesR` 874, 1024
`\addtocontents` 1447–1449
`\addtocounter` 2163
`\addtolength` ... 1834, 1986–1988, 2238
`\advanceline` 603, 634
`\affixline@num` 835
`\affixline@numR` 866, 940
`\affixside@note` 838
`\ballast@count` 897, 902
`\bbl@main@language` 1473, 1474
`\bbl@set@language` 1464, 1465
`\beginnumbering` 6, 32, 725,
 744, 1865, 1880, 1910, 1931,
 2082, 2161, 2262, 2287, 2331, 2360
`\beginnumberingR` 45, 99, 725, 766
`\Bfootfmt` 1839, 1840, 2068, 2250, 2251
`\Bfootins` 1987
`\Bfootnote` 1049, 1934, 1938,
 2090, 2095–2097, 2099, 2103,
 2104, 2113, 2115, 2117, 2122,
 2124, 2125, 2128, 2130, 2132,
 2134, 2135, 2138, 2141, 2142,
 2145–2147, 2149, 2150, 2316, 2391
`\Bfootnoterule` 1983
`\box` 2015
`\Bparafootfmt` 2040, 2068
`\bypage@Rfalse` 119, 131
`\bypage@Rtrue` 119, 127

C

`\c@ballast` 902
`\c@firstlinenumR` 150, 959, 961, 964, 966
`\c@firstsublinenumR`
 154, 946, 948, 951, 953
`\c@linenumincrementR` .. 150, 962, 963
`\c@page` ... 573, 1679, 1685, 1688, 1695
`\c@sublinenumincrementR` 154, 949, 950
`\centering` 2257
`\Cfootfmt` 2069
`\Cfootins` 1988
`\Cfootnote` 1049, 2105, 2143
`\Cfootnoterule` 1984

- `\ch@ck@l@ck` 976
`\ch@cksub@l@ck` 955
`\chardef` 1336
`\check@goal` 1753, 1767, 1779
`\check@pstarts`
 1500, 1525, 1541, 1596, 1604, 1612
`\checkpageL` 1623, 1631, 1747
`\checkpageR` 1640, 1647, 1747
`\checkraw@text`
 1504, 1521, 1552, 1620, 1656
`\clear@doublepage` 1691
`\clearl@dleftpage` 1639, 1677
`\clearl@drightpage` 1655, 1677
`\clear@toevenpage` 1677
`\clear@tol@evenpage` 1585, 1677
`\closeout` 564, 568
`\columnrulewidth` 3, 1536, 1856
`\Columns` 3, 1488, 1890, 1950
`\columnseparator` 3, 1518, 1536
`\countLline` 813, 824
`\countRline` 813, 855
`\Cparaf@otfmt` 2053, 2069
`\critext` 639
- D**
- `\date` 2077
Dekker, Dirk-Jan 76, 82
`\Dfootnote` 1049
`\dimen` 590, 591, 595–597, 601
`\divide` 949, 962
`\do@actions@fixedcode` 927
`\do@actions@nextR` 906
`\do@actionsR` 884, 906
`\do@ballastR` 885, 897
`\do@lineL` 823, 1508, 1512, 1627
`\do@lineLhook` 827, 851
`\do@lineR` 854, 1509, 1514, 1643
`\do@lineRhook` 851, 858
`\do@lockoff` 469
`\do@lockon` 429
`\documentclass` 1833, 1969, 2236
`\dp` 1998, 2013
`\dummy@ref` 525
- E**
- `\edfont@info` 678, 681, 687, 690
`\edindex` . 2087, 2102, 2114, 2126, 2127
`\edlabel` . 1191, 2303, 2318, 2377, 2393
`\edtext` 656, 1915,
 1921, 1933, 1937, 2090, 2092,
 2094–2098, 2101, 2103, 2104,
 2113, 2115, 2117, 2122, 2124,
 2125, 2128–2132, 2134, 2135,
 2138, 2140, 2142, 2145–2150,
 2272, 2279, 2316, 2342, 2350, 2391
`\Efootnote` 1049
`\emergencystretch` 2073
`\empty` .. 73, 76, 232, 240, 650, 667,
 676, 685, 752, 774, 930, 958,
 974, 1005–1007, 1018, 1029,
 1194, 1203, 1702, 1708, 1715, 1721
`\end@lemmas` 650, 651, 667, 668
`\endashchar` 1181
`\endgraf` 793, 805, 1494, 1588
`\endline@num` 532, 538
`\endlock` 623, 1345, 1354, 1359
`\endnumbering` 6, 35, 66, 103,
 726, 1872, 1887, 1924, 1947,
 2156, 2223, 2282, 2322, 2355, 2399
`\endnumberingR` 48, 66, 88, 98, 111, 726
`\endpage@num` 531, 538
`\endstanzaextra`
 1361, 2261, 2286, 2330, 2359
`\endsub` 590
`\endsubline@num` 533, 539
`\enskip` 2036, 2049, 2057
`\enspace` 2035, 2045, 2056
environments:
 `astanza` 7, 1339
 `Leftside` 5, 711
 `pages` 4, 695
 `pairs` 3, 695
 `Rightside` 5, 723
`\extensionchars` .. 43, 63, 94, 108, 116
- F**
- `\f@x@l@cks` 999
`\first@linenum@out@Rfalse` .. 559, 565
`\first@linenum@out@Rtrue` 559
`\firstlinenum` 5, 159, 1862,
 1877, 1907, 1928, 2247, 2288, 2361
`\firstsublinenum` 5, 159
`\fix@page` 249, 297
`\flag@end` 574, 655, 672
`\flag@start` 574, 647, 664
`\floatingpenalty` 1997
`\flush@notes` 1527, 1665
`\flush@notesR` 1027, 1528, 1666
`\footnote`
 . 2168, 2176, 2185, 2190, 2207,

- 2215, 2296, 2302, 2309, 2312,
2318, 2369, 2376, 2384, 2387, 2393
`\footnotesize` 1979, 1980
`\footparagraph` . 2063–2065, 2245, 2246
`\fullstop` . 195, 1178, 1180, 1182, 1184
- G**
- `\get@linelistfile` 228
`\get@nextboxL` 1630, 1787
`\get@nextboxR` 1646, 1787
`\getline@num` 832
`\getline@numR` 863, 882
`\getlinesfrompagelistL`
 1608, 1658, 1714
`\getlinesfrompagelistR`
 1609, 1659, 1714
`\getlinesfromparlistL` ... 1600, 1701
`\getlinesfromparlistR` ... 1601, 1701
`\gl@p` 235, 236, 243, 244, 651, 668, 680,
 689, 933, 934, 1011, 1015, 1030,
 1197, 1206, 1705, 1711, 1718, 1724
`\goalfraction` 4, 1779
- H**
- `\hb@xt@` 834, 840,
 847, 865, 871, 878, 1516, 1629, 1645
`\hsize` . 762, 784, 1516, 1629, 1645, 2010
`\hyphenation` 2078
- I**
- `\if@filesw` 1445
`\if@firstcolumn` 980, 1288
`\if@pstarts` 1501, 1541, 1597, 1613
`\ifaraw@text` ... 1506, 1552, 1622, 1657
`\ifbypage@` 313
`\ifbypage@R` 119, 303, 912
`\ifdim` 591, 595, 597, 601, 1681, 1754, 1768
`\iffirst@linenum@out@R` 559, 563
`\ifl@d@dash` 1181
`\ifl@d@elin` 1183, 1184
`\ifl@d@esl` 1184
`\ifl@d@pnum` 1178, 1182
`\ifl@d@ssub` 1180
`\ifl@dpagefull` 1633, 1649, 1747
`\ifl@dpadding` 5, 1296
`\ifl@dpairing` 5, 70
`\ifl@dsamelang` 1433, 1507
`\ifl@dsamepage` 1626, 1642, 1747
`\ifl@dskipnumber` 941
`\ifl@dusedbabel` 1431
`\ifledplinenum` 1179, 2044
`\ifledRcol` 5, 142, 164,
 168, 172, 176, 215, 230, 250,
 300, 324, 338, 355, 372, 384,
 392, 413, 433, 470, 509, 518,
 575, 585, 592, 598, 604, 611,
 619, 624, 628, 633, 644, 661,
 675, 1037, 1051, 1065, 1079,
 1093, 1108, 1122, 1136, 1150,
 1164, 1192, 1224, 1238, 1249,
 1261, 1273, 1312, 1325, 1468, 1478
`\ifnoteschanged@` 80
`\ifnumberedpar@` ... 746, 768, 789,
 801, 1036, 1050, 1064, 1078,
 1092, 1107, 1121, 1135, 1149,
 1163, 1248, 1260, 1272, 1311, 1324
`\ifnumbering` 33, 742, 786
`\ifnumberingR`
 27, 46, 67, 90, 122, 764, 798
`\ifodd` 990, 1302, 1679, 1685, 1688, 1695
`\ifpst@rtedL` 28, 750
`\ifpst@rtedR` 28, 772
`\ifsublines@` 193,
 284, 323, 356, 363, 394, 403,
 415, 422, 435, 452, 472, 489,
 537, 539, 886, 919, 944, 1226, 1230
`\ifvbox` 825, 856, 1556, 1559, 1788, 1808
`\ifwrittenlinesL` 1784, 1792
`\ifwrittenlinesR` 1785, 1812
`\initnumbering@reg` 41
`\InputIfFileExists` 1828
`\insert` 1993
`\insert@count` 515, 581, 645,
 662, 1044, 1058, 1072, 1086,
 1100, 1115, 1129, 1143, 1157,
 1171, 1256, 1268, 1280, 1319, 1332
`\insert@countR` 516, 577, 644,
 661, 1040, 1054, 1068, 1082,
 1096, 1111, 1125, 1139, 1153,
 1167, 1252, 1264, 1276, 1315, 1328
`\insertlines@listR`
 73, 203, 217, 521, 1007, 1011
`\inserts@list`
 .. 751, 1043, 1057, 1071, 1085,
 1099, 1114, 1128, 1142, 1156,
 1170, 1255, 1267, 1279, 1318, 1331
`\inserts@listR`
 .. 773, 1002, 1005, 1015, 1029,
 1030, 1039, 1053, 1067, 1081,

- 1095, 1110, 1124, 1138, 1152,
1166, 1251, 1263, 1275, 1314, 1327
`\interfootnotelinepenalty` 1996
`\interlinepenalty` 1347, 1996
`\interstanza`
. 2257, 2261, 2286, 2330, 2359
- L**
- `\l@d@nums` 678, 681, 687, 690,
1039, 1043, 1053, 1057, 1067,
1071, 1081, 1085, 1095, 1099,
1110, 1114, 1124, 1128, 1138,
1142, 1152, 1156, 1166, 1170,
1250, 1254, 1262, 1266, 1274, 1278
`\l@d@set` 371, 619, 620
`\l@dbbl@set@language` 1442, 1465
`\l@dbfnote` 1310
`\l@dc@maxchunks` 757, 759,
779, 781, 1390, 1400, 1408, 1415
`\l@dcalc@maxoftwo`
. 1602, 1734, 1799, 1819
`\l@dcalc@minoftwo` 1610, 1660, 1734
`\l@dchecklang` 1435, 1505
`\l@dchset@num` 251, 254, 371
`\l@dcsnote` 1247
`\l@dcsnotetext`
. 1286, 1289, 1291, 1303, 1305
`\l@demptyd@ta` 828, 859
`\l@dend@stuff` 44, 64, 95, 109, 117
`\l@dgetline@margin` 140
`\l@dgetsidenote@margin` 1236
`\l@dld@ta` 836, 867, 981, 993
`\l@dleftbox` 810, 833, 847, 1517, 1629
`\l@dlinenumR` 185
`\l@dlsn@te` 839, 870
`\l@dlsnote` 1247
`\l@dmake@labels` 1210
`\l@dmake@labelsR` 1200, 1213
`\l@dminpagelines`
. 1575, 1611, 1661, 1755, 1769
`\l@dnumpstartsL` 37, 756, 757, 759,
761, 1394, 1426, 1489, 1490,
1532, 1544, 1582, 1583, 1670, 1797
`\l@dnumpstartsR` 50, 778, 779, 781,
783, 1394, 1427, 1489, 1490,
1533, 1547, 1582, 1583, 1671, 1817
`\l@doldbbl@set@language` 1464
`\l@doldselectlanguage` 1463, 1467, 1472
`\l@dpagfullfalse` 1747
`\l@dpagfulltrue` 1747
`\l@dpagingfalse` 7, 697, 708
`\l@dpagingtrue` 703
`\l@dpairingfalse` 5, 699, 707
`\l@dpairingtrue` 696, 702
`\l@dparsedendline` 2006
`\l@dparsedstartline` 2001, 2006, 2008
`\l@dparsedstartpage` 2004, 2009
`\l@dparsefootspec` 2000
`\l@dpscl` 825, 829, 1396, 1428, 1498,
1502, 1530, 1544, 1556, 1592,
1598, 1603, 1606, 1614, 1668,
1788, 1790, 1797, 1799, 1801, 1803
`\l@dpscr` 856, 860, 1397, 1429,
1499, 1503, 1531, 1547, 1559,
1593, 1599, 1607, 1615, 1669,
1808, 1810, 1817, 1819, 1821, 1823
`\l@drd@ta` 840, 871, 983, 991
`\l@drightbox` 810, 864, 878, 1519, 1645
`\l@drsn@te` 841, 872
`\l@drsnote` 1247
`\l@dsamelangfalse` 1433, 1436
`\l@dsamelangtrue` 1433, 1439
`\l@dsamepagefalse` 1747
`\l@dsamepagetrue` 1747
`\l@dsetupmaxlinecounts` 1407, 1424
`\l@dsetuprawboxes` 1399, 1423
`\l@dskipnumberfalse` 942
`\l@dusedbabelfalse` 1431, 1460
`\l@dusedbabeltrue` 1431, 1462
`\l@duselanguage`
. 1452, 1511, 1513, 1624, 1641
`\l@dzeromaxlinecounts` 1407, 1425
`\l@dzeropenalties` 792, 804, 1493, 1587
`\l@pscl` 1396
`\l@pscr` 1396
`\label@refs`
. 1195, 1197, 1200, 1204, 1206, 1210
`\labelref@list` 1203, 1206, 1231
`\labelref@listR` 1189, 1194, 1197, 1227
`\language` 1443, 1444, 1446–1449
`\last@page@num` 311, 317
`\last@page@numR` 297
`\lastbox` 831, 862
`\lastskip` 590, 596
`\lcolwidth` 3, 4, 10, 704, 762,
834, 847, 1857, 1858, 1901, 1902
`\ldots` 1920,
1923, 1943, 1946, 2110, 2112, 2141
`\led@err@BadLeftRightPstarts`
. 18, 1490, 1583

- \led@err@LeftOnRightPage ... [21](#), 1689
 - \led@err@LineationInNumbered ... 123
 - \led@err@NumberingNotStarted ... 84
 - \led@err@NumberingShouldHaveStarted
..... 97
 - \led@err@NumberingStarted 34, 47
 - \led@err@PendNoPstart 790, 802
 - \led@err@PendNotNumbered ... 787, 799
 - \led@err@PstartInPstart ... 747, 769
 - \led@err@PstartNotNumbered . 743, 765
 - \led@err@RightOnLeftPage ... [21](#), 1696
 - \led@err@TooManyPstarts [15](#), 758, 780
 - \led@mess@NotesChanged 81
 - \led@mess@SectionContinued
..... 93, 107, 115
 - \led@warn@BadAdvancelineLine 341, 347
 - \led@warn@BadAdvancelineSubline .
..... 327, 333
 - \led@warn@BadLineation 133
 - \led@warn@BadSetline 609
 - \led@warn@BadSetlinenum 617
 - \led@warn@DuplicateLabel 1215
 - \ledllfill 840, 871
 - \ledmac@error 16, 19, 22, 24
 - \ledplinenumfalse 2002
 - \ledplinenumtrue ... 2000, 2005, 2007
 - \ledRcolfalse 9, 712, 735
 - \ledRcoltrue 724
 - \ledrlfill 840, 871
 - \ledsavedprintlines [7](#), [1176](#)
 - \ledsetnormalparstuff 2033, 2041, 2054
 - \ledsidenote 2174
 - \ledstrutL 1629, [1674](#)
 - \ledstrutR 1645, [1674](#)
 - \ledthegoal 1754, 1768, [1779](#)
 - \leftlinenumR [185](#), 981, 993
 - Leftside (environment) [5](#), [711](#)
 - \Leftsidehook 716, [718](#)
 - \Leftsidehookend 717, [718](#)
 - \lemma 1915, 2141
 - \lemmafont 2036, 2057
 - \line@list 685, 689
 - \line@list@stuff 43, 108
 - \line@list@stuffR ... 63, 94, 116, [561](#)
 - \line@listR . 76, [203](#), 216, 539, 676, 680
 - \line@margin 145
 - \line@marginR [138](#), 986
 - \line@num 314, 345,
346, 348, 366, 377, 378, 406, 1229
 - \line@numR . 55, 192, [199](#), 256, 290,
304, 339, 340, 342, 359, 373,
374, 397, 532, 536, 892, 913,
922, 957, 959, 960, 969, 970, 1225
 - \lineation 732, 1973
 - \lineationR [121](#), 732
 - \linenum@out 580, 588, 593, 599, 605,
612, 620, 625, 629, 1202, 1566, 1728
 - \linenum@outR
. [558](#), 564, 566, 568, 569, 573,
576, 586, 592, 598, 604, 611,
619, 624, 628, 633, 1193, 1571, 1731
 - \linenumberlist 958, 970
 - \linenumincrement .. [5](#), [159](#), 1863,
1878, 1908, 1929, 2248, 2289, 2362
 - \linenummargin
[138](#), 1864, 1879, 1909, 1930, 1974
 - \linenumr@p 1179, 1183, 1225, 1229
 - \linenumrepR [182](#), 192
 - \linenumsep 187, 189
 - \linesinpar@listL
..... [208](#), 224, 546, 1702, 1705
 - \linesinpar@listR
..... [208](#), 220, 549, 1708, 1711
 - \linesonpage@listL 225, 553, 1715, 1718
 - \linesonpage@listR 221, 556, 1721, 1724
 - \list@clear
. 216–221, 224, 225, 227, 751, 773
 - \list@clearing@reg 223
 - \list@create
... 203–206, 208–210, 1002, 1189
 - \lock@off 430, 431, [469](#), 628, 629
 - \lock@on 624, 625
 - \longdash 2243, 2257
- M**
- \maxchunks [2](#), [1390](#), 2240
 - \maxdimen 2010
 - \maxlinesinpar@list [208](#), 227
 - \memorydump 6, 715, 729
 - \memorydumpL [102](#), 715
 - \memorydumpR [102](#), 729
 - \message 42, 62
 - \morenoexpands 2028
 - \mpAfootnote [1106](#)
 - \mpBfootnote [1106](#)
 - \mpCfootnote [1106](#)
 - \mpDfootnote [1106](#)
 - \mpEfootnote [1106](#)
 - \mpvAfootnote 1109, 1113, 1118

\mpvBfootnote 1123, 1127, 1132
 \mpvCfootnote 1137, 1141, 1146
 \mpvDfootnote 1151, 1155, 1160
 \mpvEfootnote 1165, 1169, 1174
 \multiply 950, 963

N

\n@num 507, 633
 \n@num@reg 513
 \namebox 825, 829, 856,
 860, 1374, 1556, 1559, 1788, 1808
 \NeedsTeXFormat 2
 \new@line 840
 \new@lineR 572, 871
 \newbox 740, 810, 811, 1375
 \newcounter 150, 152, 154, 156
 \newif 6,
 8, 27, 29, 119, 559, 1431, 1433,
 1541, 1552, 1747, 1749, 1784, 1785
 \newnamebox 1374, 1402, 1403
 \newnamecount 1385, 1410
 \newwrite 558
 \next@action 244
 \next@actionline 241, 243
 \next@actionlineR
 233, 235, 900, 909, 931, 933
 \next@actionR 236,
 901, 910, 911, 916, 917, 925, 934
 \next@insert 752
 \next@insertR
 774, 1006, 1009, 1011, 1014, 1018
 \next@page@num 318, 389
 \next@page@numR 60, 259, 261, 308, 386
 \noexpands 641, 658
 \nobrak 2026, 2027
 \noindent 2011, 2086, 2166
 \normal@pars 69, 755, 777
 \normalbfnoteX 1323
 \notefontsetup 1995
 \notenumfont 1979, 2035, 2043, 2056
 \noteschanged@true
 74, 77, 677, 686, 1008
 \notetextfont 1980, 2037, 2050, 2058
 \num@lines 793, 1494, 1588
 \num@linesR 739, 805, 1495, 1589
 \numberedpar@true 763, 785
 \numberingRfalse 68
 \numberingRtrue 52, 88, 112
 \numberingtrue 39, 104
 \numlabfont 192

\numpagelinesL
 1575, 1628, 1634, 1638, 1755
 \numpagelinesR
 1575, 1644, 1650, 1654, 1769

O

\oldBfootfmt 1839, 1842, 2250, 2253
 \one@line 829, 831, 840
 \one@lineR 739, 860, 862, 871
 \openout 566, 569

P

\page@action 260, 383, 526
 \page@num 239, 316, 1299
 \page@numR
 212, 231, 306, 531, 536, 911, 988
 \pagegoal 1782
 \Pages 4, 1579, 2225, 2325, 2402
 pages (environment) 4, 695
 \pagetotal 1681, 1754, 1768
 pairs (environment) 3, 695
 \par@line 794, 1496, 1590
 \par@lineR 739, 806, 1497, 1591
 \para@vfootnote 1992
 \pausenumbering 727
 \pausenumberingR 87, 727
 \pend 5, 714, 731, 748, 1360, 2119,
 2151, 2155, 2198, 2218, 2222, 2257
 \pendL 714, 786
 \pendR 731, 770, 798
 \prevgraf
 793, 805, 1494, 1495, 1588, 1589
 \previous@A@number 2020
 \previous@B@number 2021
 \previous@C@number 2022
 \previous@page 2004, 2009, 2023
 \printindex 2229
 \printlines
 1187, 1841, 2035, 2043, 2056, 2252
 \printlinesR 7, 1176, 1841, 2252
 \protected@write 1199, 1209, 1446
 \ProvidesPackage 3
 \pst@rtedLfalse 28, 38
 \pst@rtedLtrue 105, 753
 \pst@rtedRfalse 30, 51, 71
 \pst@rtedRtrue 91, 113, 775
 \pstart 5, 16, 20, 713, 730, 1362, 2082,
 2121, 2153, 2162, 2200, 2220, 2257
 \pstartL 713, 742
 \pstartR 730, 742

R

\rbracket 2026, 2049
 \Rcolwidth 3, 4,
 10, 705, 784, 865, 878, 1858, 1902
 \read@linelist 214, 562
 \rem@inder 970, 972–974
 \resumenumbering 728
 \resumenumberingR 87, 728
 \rightlinenumR 185, 983, 991
 Rightside (environment) 5, 723
 \Rightsidehook 718, 733
 \Rightsidehookend 718, 736
 \rlap 983, 991
 \Rlineflag 7, 180, 192,
 1179, 1183, 1217, 1837, 1854, 2071
 \rule 1537

S

\sc@n@list 971, 973
 \section@num 40, 42, 43, 106–108
 \section@numR
 ... 25, 53, 62, 63, 92–94, 114–116
 \select@language ... 1444, 1446–1449
 \select@lemmafont 2049
 \selectlanguage 1452, 2083, 2160
 \set@line 643, 660, 674
 \set@line@action
 ... 253, 352, 361, 368, 391, 528
 \setl@dlp@rbox 1289, 1305
 \setl@drp@rbox 1291, 1303
 \setline 607, 2086, 2166
 \setlinenum 615
 \setnamebox 761, 783, 1374
 \setprintlines 1177
 \setstanzaindents
 ... 1852, 1899, 2241, 2290, 2363
 \showlemma 649, 666
 \sidenote@margin 1241, 1245
 \sidenote@marginR 1234, 1294
 \sidenotemargin 1234, 1975, 2160
 \skip 1986–1988
 \skip@lockoff 431, 469
 \skipnumbering 7, 632, 2257
 \skipnumbering@reg 636
 \smash 1537
 \splitmaxdepth 1998
 \splittopskip 826, 857, 1998
 \stanza ... 1866, 1881, 1911, 1932,
 2264, 2270, 2276, 2292, 2300, 2308
 \stanza@count 1342, 1356, 1367

\stanza@chang 1344, 1369
 \stanzaindentbase 1367
 \startlock 623
 \startstanzahook 1340
 \startsub 590
 \sub@action 269, 412, 527
 \sub@change 61, 263, 264, 270
 \sub@lock .. 58, 278, 280, 282, 285,
 437, 438, 440, 454, 455, 457,
 474, 475, 477, 491, 492, 494, 887
 \sub@off 598, 599
 \sub@on 592, 593
 \subline@num 194,
 314, 331, 332, 334, 364, 404, 1230
 \subline@numR 56,
 195, 199, 286, 290, 304, 325,
 326, 328, 357, 395, 533, 537,
 888, 893, 913, 920, 945–947, 1226
 \sublinenumincrement 5, 159
 \sublinenumr@p . 1180, 1184, 1226, 1230
 \sublinenumrepR 182, 195
 \sublines@false 59, 267
 \sublines@true 265
 \symlinenum 1179
 \sza@penalty 1351, 1355

T

\textdagger 2185
 \textheight 1834, 2238
 \textit 1896, 1921, 1923, 1944, 1946,
 1956, 2096, 2109, 2130, 2135,
 2141, 2143, 2145, 2193, 2312, 2387
 \textnormal 2027
 \textsc 1895, 2091
 \textwidth 11, 13, 704, 705, 1857, 1901
 \theledlanguageL 1437, 1452, 1511, 1624
 \theledlanguageR 1437, 1452, 1513, 1641
 \thepage 573, 1200, 1210
 \thr@@ 475, 482, 492, 499
 \title 2076
 \topskip 1681

U

\unhbox 840,
 871, 1379, 1517, 1519, 1629, 1645
 \unhnamebox 1374
 \unvbox 831, 862, 1381
 \unvnamebox 1374
 \unvxh 2012

<code>\usernamecount</code>	<code>\writtenlinesRtrue</code>
. 1343, 1350, <u>1385</u> , 1417, 1603,	1815
1790, 1799, 1801, 1810, 1819, 1821	
<code>\usepackage</code> 1835, 1970–1972, 2237	
V	
<code>\vAfootnote</code>	1038, 1042, 1047
<code>\vbadness</code>	826, 857
<code>\vbfnoteX</code>	1326, 1330
<code>\vBfootnote</code>	1052, 1056, 1061
<code>\vbox</code>	761, 783, 2010
<code>\vCfootnote</code>	1066, 1070, 1075
<code>\vDfootnote</code>	1080, 1084, 1089
<code>\vEfootnote</code>	1094, 1098, 1103
<code>\vl@dbfnote</code>	1313, 1317
<code>\vl@dcsnote</code>	1274, 1278
<code>\vl@dlsnote</code>	1250, 1254
<code>\vl@drsnote</code>	1262, 1266
<code>\vsplit</code>	829, 860
W	
<code>\wd</code>	840, 871, 2014
<code>\writtenlinesLfalse</code>	1594, 1798
<code>\writtenlinesLtrue</code>	1795
<code>\writtenlinesRfalse</code>	1595, 1818
X	
<code>\x@lemma</code>	651–653, 668–670
<code>\xlineref</code>	2319, 2394
<code>\xpg@main@language</code>	1482, 1483
<code>\xpg@set@language</code>	1477, 1481
<code>\xright@appenditem</code>	
.....	385, 386, 388, 389, 393,
400, 402, 409, 414, 416, 418,	
421, 423, 425, 434, 436, 443,	
451, 453, 460, 471, 473, 480,	
488, 490, 497, 510, 511, 521,	
535, 546, 549, 553, 556, 1038,	
1042, 1052, 1056, 1066, 1070,	
1080, 1084, 1094, 1098, 1109,	
1113, 1123, 1127, 1137, 1141,	
1151, 1155, 1165, 1169, 1225,	
1229, 1250, 1254, 1262, 1266,	
1274, 1278, 1313, 1317, 1326, 1330	
Z	
<code>\z@skip</code>	1999
<code>\zz@@@</code>	1195, 1204

Change History

v0.1	<code>\do@actions@nextR:</code> Used
General: First public release	1
v0.2	<code>\do@actions@fixedcode</code> in
General: Added section of babel re-	<code>\do@actionsR</code>
lated code	37
Fix babel problems	1
<code>\Columns:</code> Added <code>\l@dchecklang</code>	<code>\do@lineL:</code> Added <code>\do@lineLhook</code>
and <code>\l@duselanguage</code> to	to <code>\do@lineL</code>
<code>\Columns</code>	35
<code>\Pages:</code> Added <code>\l@duselanguage</code>	Simplified <code>\do@lineL</code> by using
to <code>\Pages</code>	macros for some common code
59	35
v0.3	<code>\do@lineR:</code> Changed <code>\do@lineR</code>
General: Reorganize for ledarab ..	similarly to <code>\do@lineL</code>
<code>\affixline@numR:</code> Changed	36
<code>\affixline@numR</code> to match new	<code>\do@lineRhook:</code> Added <code>\do@lineLhook</code>
ledmac	and <code>\do@lineRhook</code>
38	36
	<code>Leftside:</code> Added hooks into Left-
	side environment
	31
	<code>\flag@end:</code> Removed extraneous
	spaces from <code>\flag@end</code>
	27
	<code>\ifledRcol:</code> Moved <code>\ifl@dpairing</code>

to ledmac	10	\Rightsidehook and \Rightsidehookend	
\ifpst@rtedR: Moved \ifpst@rtedL		31
to ledmac	11	\sublinenumrepR: Added	
\l@dlinenumR: Simplified		\linenumrepR and \sublinenumrepR	
\leftlinenumR and \rightlinenumR		16
by introducing \l@dlinenumR	16	v0.3a	
\l@dnumpstartsR: Moved		General: Minor \linenummargin	
\l@dnumpstartsL to ledmac .	51	fix	1
\ledsavedprintlines: Simpli-		\line@marginR: Don't just	
fied \printlinesR by using		set \line@marginR in	
\setprintlines	45	\linenummargin	15
\ledstrutR: Added \ledtrutL and		v0.3b	
\ledstrutR	60	General: Improved parallel page	
\normalbfnoteX: Removed		balancing	1
extraneous spaces from		\Pages: Added \l@dminpagelines	
\normalbfnoteX	49	calculation for succeeding page	
\Pages: Added \ledstrutL to		pairs	60
\Pages	59	v0.3c	
Added \ledstrutR to \Pages .	59	General: Compatibilty with Poly-	
\Rightsidehookend: Added		glossia	1
\Leftsidehook, \Leftsidehookend,			