

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.

Please, for all bug's report, open a ticket on <https://github.com/maieul/ledmac/issues/>

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

*This file (**ledpar.dtx**) has version number v0.6, last revised 2011/08/22.

[†]herries dot press at earthlink dot net

[‡]maieul at maieul dot net

10 Sectioning commands	12
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	29
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	37
14.4 Line number printing	39
14.5 Add insertions to the vertical list	42
14.6 Penalties	42
14.7 Printing leftover notes	43
15 Footnotes	44
15.1 Outer-level footnote commands	44
15.2 Normal footnote formatting	47
16 Cross referencing	47
17 Side notes	49
18 Familiar footnotes	50
19 Verse	51
20 Naming macros	52
21 Counts and boxes for parallel texts	53
22 Fixing babel	54
23 Parallel columns	56
24 Parallel pages	59
25 The End	66

<i>List of Figures</i>	3
------------------------	---

A Examples	67
A.1 Parallel column example	76
A.2 Example parallel facing pages	78
A.3 Example poetry on parallel facing pages	84
References	89
Index	89
Change History	97

List of Figures

1	Output from <code>villon.tex</code>	69
2	Left page output from <code>djd17nov.tex</code>	70
3	Right page output from <code>djd17nov.tex</code>	71
4	First left page output from <code>djdpoems.tex</code>	72
5	First right page output from <code>djdpoems.tex</code>	73
6	Second left page output from <code>djdpoems.tex</code>	74
7	Second right page output from <code>djdpoems.tex</code>	75

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 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 **\Rcolwidth** 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 **\columnseparator** 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}[2011/08/22 v0.6 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\@lockR \z@
57 \global\sub@lockR \z@
58 \global\sublines@false
59 \global\let\next@page@numR\relax
60 \global\let\sub@change\relax
61 \message{Section \the\section@numR R}%
62 \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
63 \l@dend@stuff}
64

```

`\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.

```

65 \def\endnumberingR{%
66   \ifnumberingR
67     \global\numberingRfalse
68     \normal@pars

```

```

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

```

`\pausenumberingR` These are the right text equivalents of `\pausenumbering` and `\resumenumbering`.

```

\resumenumberingR 86 \newcommand*{\pausenumberingR}{%
87 \endnumberingR\global\numberingRtrue}
88 \newcommand*{\resumenumberingR}{%
89 \ifnumberingR
90 \global\pst@rtedRtrue
91 \global\advance\section@numR \@ne
92 \led@mess@SectionContinued{\the\section@numR R}%
93 \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
94 \l@dend@stuff
95 \else
96 \led@err@numberingShouldHaveStarted
97 \endnumberingR
98 \beginnumberingR
99 \fi}
100

```

`\memorydumpL` `\memorydump` is a shorthand for `\pausenumbering\resumenumbering`. This will clear the memorised stuff for the previous chunks while keeping the numbering going.

```

101 \newcommand*{\memorydumpL}{%
102 \endnumbering
103 \numberingtrue
104 \global\pst@rtedLtrue
105 \global\advance\section@num \@ne
106 \led@mess@SectionContinued{\the\section@num}%
107 \line@list@stuff{\jobname.\extensionchars\the\section@num}%
108 \l@dend@stuff}
109 \newcommand*{\memorydumpR}{%
110 \endnumberingR
111 \numberingRtrue

```

```

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

```

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.

```

118 \newif\ifbypage@R
119 \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`.

```

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

```

`\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.

`\line@marginR`

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.

```

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

```

By default put right text numbers at the right.

```

147 \line@marginR=\@ne
148

```

`\c@firstlinenumR` The following counters tell ledmac which right text lines should be printed with
`\c@linenumincrementR` 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.

```

149 \newcounter{firstlinenumR}
150 \setcounter{firstlinenumR}{5}
151 \newcounter{linenumincrementR}
152 \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.

```

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

```

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

```

\firstsublinenum 158 \providecommand*{\firstlinenum}{}
\sublinenumincrement 159 \providecommand*{\linenumincrement}{}
160 \providecommand*{\firstsublinenum}{}
161 \providecommand*{\sublinenumincrement}{}
162 \renewcommand*{\firstlinenum}[1]{%
163   \ifledRcol \setcounter{firstlinenumR}{#1}%
164   \else      \setcounter{firstlinenumR}{#1}%
165   \fi}
166 \renewcommand*{\linenumincrement}[1]{%
167   \ifledRcol \setcounter{linenumincrementR}{#1}%
168   \else      \setcounter{linenumincrementR}{#1}%
169   \fi}

```



```

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

```

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

```

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

```

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

```

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

```

`\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`.

```

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

```

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.

```

198 \newcount\line@numR
199 \newcount\subline@numR
200 \newcount\absline@numR
201

```

`\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:

```

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

```

`\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.

```

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

```

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

```

211 \newcount\page@numR
212

```

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.

```

213 \renewcommand*\read@linelist[1]{%

```

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

```

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

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

Now get the file and interpret it.

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

```
228 \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.

```
229 \ifledRcol
```

```
230 \global\page@numR=\m@ne
```

```
231 \ifx\actionlines@listR\empty
```

```
232 \gdef\next@actionlineR{1000000}%
```

```
233 \else
```

```
234 \glp\actionlines@listR\to\next@actionlineR
```

```
235 \glp\actions@listR\to\next@actionR
```

```
236 \fi
```

```
237 \else
```

```
238 \global\page@num=\m@ne
```

```
239 \ifx\actionlines@list\empty
```

```
240 \gdef\next@actionline{1000000}%
```

```
241 \else
```

```
242 \glp\actionlines@list\to\next@actionline
```

```
243 \glp\actions@list\to\next@action
```

```
244 \fi
```

```
245 \fi}
```

```
246
```

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@regR` `\@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.

```
247 \newcommand{\@l@regR}{%
```

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

```
249 \advance\absline@numR \@ne
```

```
250 \set@line@action
```

```

251 \let\l@dchset@num\relax
252 \advance\absline@numR \m@ne
253 \advance\line@numR \m@ne% % do we need this?
254 \fi
255 \advance\absline@numR \@ne
256 \ifx\next@page@numR\relax \else
257 \page@action
258 \let\next@page@numR\relax
259 \fi
260 \ifx\sub@change\relax \else
261 \ifnum\sub@change>\z@
262 \sublines@true
263 \else
264 \sublines@false
265 \fi
266 \sub@action
267 \let\sub@change\relax
268 \fi
269 \ifcase\@lockR
270 \or
271 \@lockR \tw@
272 \or\or
273 \@lockR \z@
274 \fi
275 \ifcase\sub@lockR
276 \or
277 \sub@lockR \tw@
278 \or\or
279 \sub@lockR \z@
280 \fi
281 \ifsublines@
282 \ifnum\sub@lockR<\tw@
283 \advance\subline@numR \@ne
284 \fi
285 \else
286 \ifnum\@lockR<\tw@
287 \advance\line@numR \@ne \subline@numR \z@
288 \fi
289 \fi}
290
291 \renewcommand*{\@l}[2]{%
292 \fix@page{#1}%
293 \ifledRcol
294 \@l@regR
295 \else
296 \@l@reg
297 \fi}
298

```

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

```

\fix@page
299 \newcount\last@page@numR
300 \last@page@numR=-10000
301 \renewcommand*{\fix@page}[1]{%
302   \ifledRcol
303     \ifnum #1=\last@page@numR
304     \else
305       \ifbypage@R
306         \line@numR \z@ \subline@numR \z@
307       \fi
308       \page@numR=#1\relax
309       \last@page@numR=#1\relax
310       \def\next@page@numR{#1}%
311     \fi
312   \else
313     \ifnum #1=\last@page@num
314     \else
315       \ifbypage@
316         \line@num \z@ \subline@num \z@
317       \fi
318       \page@num=#1\relax
319       \last@page@num=#1\relax
320       \def\next@page@num{#1}%
321     \fi
322   \fi}
323

```

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

```

324 \renewcommand*{\@adv}[1]{%
325   \ifsublines@
326   \ifledRcol
327     \advance\subline@numR by #1\relax
328     \ifnum\subline@numR<\z@
329       \led@warn@BadAdvancelineSubline
330       \subline@numR \z@
331     \fi
332   \else
333     \advance\subline@num by #1\relax
334     \ifnum\subline@num<\z@
335       \led@warn@BadAdvancelineSubline
336       \subline@num \z@
337     \fi
338   \fi
339 \else
340   \ifledRcol
341     \advance\line@numR by #1\relax
342     \ifnum\line@numR<\z@
343       \led@warn@BadAdvancelineLine

```

```

344     \line@numR \z@
345     \fi
346   \else
347     \advance\line@num by #1\relax
348     \ifnum\line@num<\z@
349       \led@warn@BadAdvancelineLine
350       \line@num \z@
351     \fi
352   \fi
353 \fi
354 \set@line@action}
355

```

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

```

356 \renewcommand*{\@set}[1]{%
357   \ifledRcol
358     \ifsublines@
359       \subline@numR=#1\relax
360     \else
361       \line@numR=#1\relax
362     \fi
363   \set@line@action
364 \else
365   \ifsublines@
366     \subline@num=#1\relax
367   \else
368     \line@num=#1\relax
369   \fi
370   \set@line@action
371 \fi}
372

```

`\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.

```

373 \renewcommand*{\l@d@set}[1]{%
374   \ifledRcol
375     \line@numR=#1\relax
376     \advance\line@numR \@ne
377     \def\l@dchset@num{#1}
378   \else
379     \line@num=#1\relax
380     \advance\line@num \@ne
381     \def\l@dchset@num{#1}
382   \fi}
383 \let\l@dchset@num\relax
384

```

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

```

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

```

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

```

393 \renewcommand*{\set@line@action}{%
394   \ifledRcol
395     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
396     \ifsublines@
397       \@l@dttempcnta=-\subline@numR
398     \else
399       \@l@dttempcnta=-\line@numR
400     \fi
401     \advance\@l@dttempcnta by -5000\relax
402     \xright@appenditem{\the\@l@dttempcnta}\to\actions@listR
403   \else
404     \xright@appenditem{\the\absline@num}\to\actionlines@list
405     \ifsublines@
406       \@l@dttempcnta=-\subline@num
407     \else
408       \@l@dttempcnta=-\line@num
409     \fi
410     \advance\@l@dttempcnta by -5000\relax
411     \xright@appenditem{\the\@l@dttempcnta}\to\actions@list
412   \fi}
413

```

`\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.

```

414 \renewcommand*{\sub@action}{%
415   \ifledRcol
416     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
417     \ifsublines@
418       \xright@appenditem{-1001}\to\actions@listR
419     \else
420       \xright@appenditem{-1002}\to\actions@listR
421     \fi
422   \else
423     \xright@appenditem{\the\absline@num}\to\actionlines@list
424     \ifsublines@
425       \xright@appenditem{-1001}\to\actions@list
426     \else

```

```

427     \xright@appenditem{-1002}\to\actions@list
428     \fi
429 \fi}
430

```

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

```

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

```

`\lock@off` `\lock@off` adds an entry to the action-code list to turn line number locking off.
`\do@lockoff` 466
`\do@lockoffR` 467
`\skip@lockoff` 468 `\newcommand{\do@lockoffR}{%`
469 `\xright@appenditem{\the\absline@numR}\to\actionlines@listR`


```

470 \ifsublines@
471   \xright@appenditem{-1006}\to\actions@listR
472   \ifnum\sub@lockR=\tw@
473     \sub@lockR \thr@@
474   \else
475     \sub@lockR \z@
476   \fi
477 \else
478   \xright@appenditem{-1004}\to\actions@listR
479   \ifnum\@lockR=\tw@
480     \@lockR \thr@@
481   \else
482     \@lockR \z@
483   \fi
484 \fi}
485
486 \renewcommand*{\do@lockoff}{%
487   \ifledRcol
488     \do@lockoffR
489   \else
490     \do@lockoffL
491   \fi}
492 \global\let\lock@off=\do@lockoff
493

```

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

```

494 \providecommand*{\n@num}{-}
495 \renewcommand*{\n@num}{%
496   \ifledRcol
497     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
498     \xright@appenditem{-1007}\to\actions@listR
499   \else
500     \n@num@reg
501   \fi}
502

```

`\@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`.

```

503   \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.

```

504 \renewcommand*{\@ref}[2]{%
505   \ifledRcol
506     \global\insert@countR=#1\relax
507     \loop\ifnum\insert@countR>\z@
508       \xright@appenditem{\the\absline@numR}\to\insertlines@listR
509       \global\advance\insert@countR \m@ne
510     \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.

```

511 \begingroup
512   \let\@ref=\dummy@ref
513   \let\page@action=\relax
514   \let\sub@action=\relax
515   \let\set@line@action=\relax
516   \let\@lab=\relax
517   #2
518   \global\endpage@num=\page@numR
519   \global\endline@num=\line@numR
520   \global\endsubline@num=\subline@numR
521 \endgroup

```

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

```

522   \xright@appenditem%
523     {\the\page@numR|\the\line@numR|%
524     \ifsublines@ \the\subline@numR \else 0\fi|%
525     \the\endpage@num|\the\endline@num|%
526     \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.

```

527   #2
528   \else

```

And when not in right text

```

529     \@ref@reg{#1}{#2}%
530   \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.

```

531 \providecommand*{\@pend}[1]{%
532 \renewcommand*{\@pend}[1]{%
533   \xright@appenditem{#1}\to\linesinpar@listL}

```

```

534 \providecommand*\@pendR}[1]{}
535 \renewcommand*\@pendR}[1]{%
536   \xright@appenditem{#1}\to\linesinpar@listR}
537

```

`\@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.

```

538 \providecommand*\@lopL}[1]{}
539 \renewcommand*\@lopL}[1]{%
540   \xright@appenditem{#1}\to\linesonpage@listL}
541 \providecommand*\@lopR}[1]{}
542 \renewcommand*\@lopR}[1]{%
543   \xright@appenditem{#1}\to\linesonpage@listR}
544

```

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`.

```

545 \newwrite\linenum@outR

```

`\iffirst@linenum@outR` 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
546 \newif\iffirst@linenum@out@R
547   \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.

```

548 \newcommand*\line@list@stuffR}[1]{%
549   \read@linelist{#1}%
550   \iffirst@linenum@out@R
551     \immediate\closeout\linenum@outR
552     \global\first@linenum@out@Rfalse
553     \immediate\openout\linenum@outR=#1
554   \else
555     \closeout\linenum@outR
556     \openout\linenum@outR=#1
557   \fi}
558

```

`\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.

```

559 \newcommand*\new@lineR}{%
560   \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.

```
561 \renewcommand*{\flag@start}{%
562   \ifledRcol
563     \edef\next{\write\linenum@outR{%
564               \string\@ref[\the\insert@countR] []}%
565     \next
566   \else
567     \edef\next{\write\linenum@out{%
568               \string\@ref[\the\insert@count] []}%
569     \next
570   \fi}
571 \renewcommand*{\flag@end}{%
572   \ifledRcol
573     \write\linenum@outR{}}%
574   \else
575     \write\linenum@out{}}%
576   \fi}
```

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

```
577 \renewcommand*{\startsub}{\dimen0\lastskip
578   \ifdim\dimen0>0pt \unskip \fi
579   \ifledRcol \write\linenum@outR{\string\sub@on}%
580   \else      \write\linenum@out{\string\sub@on}%
581   \fi
582   \ifdim\dimen0>0pt \hskip\dimen0 \fi}
583 \def\endsub{\dimen0\lastskip
584   \ifdim\dimen0>0pt \unskip \fi
585   \ifledRcol \write\linenum@outR{\string\sub@off}%
586   \else      \write\linenum@out{\string\sub@off}%
587   \fi
588   \ifdim\dimen0>0pt \hskip\dimen0 \fi}
589
```

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

```
590 \renewcommand*{\advanceline}[1]{%
591   \ifledRcol \write\linenum@outR{\string\@adv[#1]}%
592   \else      \write\linenum@out{\string\@adv[#1]}%
593   \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.

```
594 \renewcommand*{\setline}[1]{%
595   \ifnum#1<\z@
596     \led@warn@BadSetline
597   \else
598     \ifledRcol \write\linenum@outR{\string\@set[#1]}%
```

```

599     \else          \write\linenum@out{\string\@set[#1]}%
600     \fi
601 \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.

```

602 \renewcommand*{\setlinenum}[1]{%
603   \ifnum#1<\z@
604     \led@warn@BadSetlinenum
605   \else
606     \ifledRcol \write\linenum@outR{\string\l@d@set[#1]}
607     \else      \write\linenum@out{\string\l@d@set[#1]} \fi
608   \fi}
609

```

\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.

```

\endlock
610 \renewcommand*{\startlock}{%
611   \ifledRcol \write\linenum@outR{\string\lock@on}%
612   \else      \write\linenum@out{\string\lock@on}%
613   \fi}
614 \def\endlock{%
615   \ifledRcol \write\linenum@outR{\string\lock@off}%
616   \else      \write\linenum@out{\string\lock@off}%
617   \fi}
618

```

\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.

```

619 \renewcommand*{\skipnumbering}{%
620   \ifledRcol \write\linenum@outR{\string\n@num}%
621             \advanceline{-1}%
622   \else
623     \skipnumbering@reg
624   \fi}
625

```

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.

```
626 \long\def\critext#1#2/{\leavevmode
627   \begingroup
628     \no@expands
629     \xdef\@tag{#1}%
630     \set@line
631     \ifledRcol \global\insert@countR \z@
632     \else      \global\insert@count \z@ \fi
633     \ignorespaces #2\relax
634     \flag@start
635   \endgroup
636   \showlemma{#1}%
637   \ifx\end@lemmas\empty \else
638     \gl@p\end@lemmas\to\x@lemma
639     \x@lemma
640     \global\let\x@lemma=\relax
641   \fi
642   \flag@end}
```

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

```
643 \renewcommand{\edtext}[2]{\leavevmode
644   \begingroup
645     \no@expands
646     \xdef\@tag{#1}%
647     \set@line
648     \ifledRcol \global\insert@countR \z@
649     \else      \global\insert@count \z@ \fi
650     \ignorespaces #2\relax
651     \flag@start
652   \endgroup
653   \showlemma{#1}%
654   \ifx\end@lemmas\empty \else
655     \gl@p\end@lemmas\to\x@lemma
656     \x@lemma
657     \global\let\x@lemma=\relax
658   \fi
659   \flag@end}
660
```

`\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`.

```

661 \renewcommand*{\set@line}{%
662   \ifledRcol
663     \ifx\line@listR\empty
664       \global\noteschanged@true
665       \xdef\l@d@nums{000|000|000|000|000|000|000|000|\edfont@info}%
666     \else
667       \gl@p\line@listR\to\@tempb
668       \xdef\l@d@nums{\@tempb|\edfont@info}%
669       \global\let\@tempb=\undefined
670     \fi
671   \else
672     \ifx\line@list\empty
673       \global\noteschanged@true
674       \xdef\l@d@nums{000|000|000|000|000|000|000|000|\edfont@info}%
675     \else
676       \gl@p\line@list\to\@tempb
677       \xdef\l@d@nums{\@tempb|\edfont@info}%
678       \global\let\@tempb=\undefined
679     \fi
680   \fi}
681
```

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.

```

chapterinpages 682 \newenvironment{pairs}{%
683   \l@dpairingtrue
684   \l@dpaddingfalse
685 }{%
686   \l@dpairingfalse
687 }
```

The `pages` environment additionally sets the ‘column’ widths to the `\textwidth` (as known at the time the package is called). As in this environnement there is two text in parallels in 2 pages, chapter have not to start in a left page. So the `\chapter` command is redefined to make no test about clearing page.

```

688 \newenvironment{pages}{%
689   \let\oldchapter\chapter
690   \let\chapter\chapterinpages
691   \l@dpairingtrue
692   \l@dpaddingtrue
693   \setlength{\Lcolwidth}{\textwidth}%
694   \setlength{\Rcolwidth}{\textwidth}%

```

```

695 }{%
696   \l@dpairingfalse
697   \l@dpagingfalse
698   \let\chapter\oldchapter
699 }
700 \newcommand{\chapterinpages}{\thispagestyle{plain}%
701                               \global\@topnum\z@
702                               \@afterindentfalse
703                               \secdef\@chapter\@schapter}
704

```

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.

```

705 \newenvironment{Leftside}{%
706   \ledRcolfalse
707   \let\pstart\pstartL
708   \let\pend\pendL
709   \let\memorydump\memorydumpL
710   \Leftsidehook
711 }{\Leftsidehookend}

```

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

```

\Rightsidehook 712 \newcommand*{\Leftsidehook}{}
\Rightsidehookend 713 \newcommand*{\Leftsidehookend}{}
714 \newcommand*{\Rightsidehook}{}
715 \newcommand*{\Rightsidehookend}{}
716

```

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.

```

717 \newenvironment{Rightside}{%
718   \ledRcoltrue
719   \let\beginnumbering\beginnumberingR
720   \let\endnumbering\endnumberingR
721   \let\pausenumbering\pausenumberingR
722   \let\resumenumbering\resumenumberingR
723   \let\memorydump\memorydumpR
724   \let\pstart\pstartR
725   \let\pend\pendR
726   \let\lineation\lineationR
727   \Rightsidehook
728 }{%
729   \ledRcolfalse
730   \Rightsidehookend
731 }
732

```


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.

```
733 \newcount\num@linesR
734 \newbox\one@lineR
735 \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.
`\pstartR`

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.

```
736 \newcommand*{\pstartL}{
737 \if@nobreak
738 \let\@oldnobreak\@nobreaktrue
739 \else
740 \let\@oldnobreak\@nobreakfalse
741 \fi
742 \@nobreaktrue
743 \ifnumbering \else
744 \led@err@PstartNotNumbered
745 \beginnumbering
746 \fi
747 \ifnumberedpar@
748 \led@err@PstartInPstart
749 \pend
```

750 \fi

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

```
751 \ifpst@rtedL\else
752   \list@clear{\inserts@list}%
753   \global\let\next@insert=\empty
754   \global\pst@rtedLtrue
755 \fi
756 \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.

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

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

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

```
794 \newcommand*{\pendL}{\ifnumbering \else
795   \led@err@PendNotNumbered
796 \fi
797 \ifnumberedpar@ \else
798   \led@err@PendNoPstart
799 \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.

```
800 \l@dzeropenalties
801 \endgraf\global\num@lines=\prevgraf\egroup
802 \global\par@line=0
```

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

```
803 \endgroup
804 \ignorespaces
805 \@oldnbreak}
806
```

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

```
807 \newcommand*{\pendR}{\ifnumberingR \else
808   \led@err@PendNotNumbered
809 \fi
810 \ifnumberedpar@ \else
811   \led@err@PendNoPstart
812 \fi
813 \l@dzeropenalties
814 \endgraf\global\num@linesR=\prevgraf\egroup
815 \global\par@lineR=0
816 \endgroup
817 \ignorespaces
818 \@oldnbreak}
819
```

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 of right text.

```
820 \newbox\l@dleftbox
821 \newbox\l@drightbox
822
```

```

\countLline We need to know the number of lines processed.
\countRline 823 \newcount\countLline
            824 \countLline \z@
            825 \newcount\countRline
            826 \countRline \z@
            827

\@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 828 \newcount\@donereallinesL
                829 \newcount\@donetotallinesL
                830 \newcount\@donereallinesR
                831 \newcount\@donetotallinesR
                832

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

            833 \newcommand*{\do@lineL}{%
            834 \advance\countLline \@ne
            835 \ifvbox\namebox{1@dLcolrawbox\the\l@dpscL}%
            836 {\vbadness=10000
            837 \splittopskip=\z@
            838 \do@lineLhook
            839 \l@emptyd@ta
            840 \global\setbox\one@line=\vsplit\namebox{1@dLcolrawbox\the\l@dpscL}
            841 to\baselineskip}%
            842 \unvbox\one@line \global\setbox\one@line=\lastbox
            843 \getline@num
            844 \setbox\l@dleftbox
            845 \hb@xt@ \Lcolwidth{%
            846 \affixline@num
            847 \l@dld@ta
            848 \add@inserts
            849 \affixside@note
            850 \l@dlsn@te
            851 {\ledllfill\hb@xt@ \wd\one@line{\new@line\l@dunhbox@line{\one@line}}\ledrlfill\l@drd@
            852 \l@drsn@te
            853 }}%
            854 \add@penaltiesL
            855 \global\advance\@donereallinesL\@ne
            856 \global\advance\@donetotallinesL\@ne
            857 \else
            858 \setbox\l@dleftbox \hb@xt@ \Lcolwidth{\hspace*\Lcolwidth}}%
            859 \global\advance\@donetotallinesL\@ne
            860 \fi}
            861
            862

```

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

```
\do@lineRhook 863 \newcommand*{\do@lineLhook}{%
864 \newcommand*{\do@lineRhook}{%
865
```

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

```
866 \newcommand*{\do@lineR}{%
867   \advance\countRline \@ne
868   \ifvbox\namebox{\l@dRcolrawbox\the\l@dpscR}%
869   {\vbadness=10000
870    \splittopskip=\z@
871    \do@lineRhook
872    \l@demptyd@ta
873    \global\setbox\one@lineR=\vsplit\namebox{\l@dRcolrawbox\the\l@dpscR}
874    to\baselineskip}%
875   \unvbox\one@lineR \global\setbox\one@lineR=\lastbox
876   \getline@numR
877   \setbox\l@dtrightbox
878   \hb@xt@ \Rcolwidth{%
879     \affixline@numR
880     \l@dld@ta
881     \add@insertsR
882     \affixside@noteR
883     \l@dlsn@te
884     {\ledllfill\hb@xt@ \wd\one@lineR{\new@lineR\l@dunhbox@line{\one@lineR}}\ledrlfill\l@d@rd@ta%
885     \l@d@rsn@te
886   }}%
887   \add@penaltiesR
888   \global\advance\@donereallinesR\@ne
889   \global\advance\@donetotallinesR\@ne
890 \else
891   \setbox\l@dtrightbox \hb@xt@ \Rcolwidth{\hspace*{\Rcolwidth}}
892   \global\advance\@donetotallinesR\@ne
893 \fi}
894
895
```

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.

```
896 \newcommand*{\getline@numR}{%
897   \global\advance\absline@numR \@ne
898   \do@actionsR
899   \do@ballastR
900   \ifsublines@
901     \ifnum\sub@lockR<\tw@
902       \global\advance\subline@numR \@ne
```

```

903 \fi
904 \else
905 \ifnum\@lockR<\tw@
906 \global\advance\line@numR \@ne
907 \global\subline@numR \z@
908 \fi
909 \fi}
910
911

```

`\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.

```

912 \newcommand*{\do@ballastR}{\global\ballast@count=\z@
913 \begingroup
914 \advance\absline@numR \@ne
915 \ifnum\next@actionlineR=\absline@numR
916 \ifnum\next@actionR>-1001
917 \global\advance\ballast@count by -\c@ballast
918 \fi
919 \fi
920 \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.

`\do@actions@fixedcodeR`

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

```

921 \newcommand*{\do@actions@fixedcodeR}{%
922 \ifcase\@l@dttempcnta%
923 \or% % 1001
924 \global\sublines@true
925 \or% % 1002
926 \global\sublines@false
927 \or% % 1003
928 \global\@lockR=\@ne
929 \or% % 1004
930 \ifnum\@lockR=\tw@
931 \global\@lockR=\thr@@
932 \else
933 \global\@lockR=\z@
934 \fi
935 \or% % 1005
936 \global\sub@lockR=\@ne
937 \or% % 1006
938 \ifnum\sub@lockR=\tw@
939 \global\sub@lockR=\thr@@
940 \else
941 \global\sub@lockR=\z@
942 \fi

```

```

943 \or% % 1007
944 \l@dskipnumbertrue
945 \else
946 \led@warn@BadAction
947 \fi}
948
949
950 \newcommand*{\do@actionsR}{%
951 \global\let\do@actions@nextR=\relax
952 \@l@dttempcntb=\absline@numR
953 \ifnum\@l@dttempcntb<\next@actionlineR\else
954 \ifnum\next@actionR>-1001\relax
955 \global\page@numR=\next@actionR
956 \ifbypage@R
957 \global\line@numR \z@ \global\subline@numR \z@
958 \fi
959 \else
960 \ifnum\next@actionR<-4999\relax % 9/05 added relax here
961 \@l@dttempcnta=-\next@actionR
962 \advance\@l@dttempcnta by -5001\relax
963 \ifsublines@
964 \global\subline@numR=\@l@dttempcnta
965 \else
966 \global\line@numR=\@l@dttempcnta
967 \fi
968 \else
969 \@l@dttempcnta=-\next@actionR
970 \advance\@l@dttempcnta by -1000\relax
971 \do@actions@fixedcodeR
972 \fi
973 \fi
974 \ifx\actionlines@listR\empty
975 \gdef\next@actionlineR{1000000}%
976 \else
977 \gl@p\actionlines@listR\to\next@actionlineR
978 \gl@p\actions@listR\to\next@actionR
979 \global\let\do@actions@nextR=\do@actionsR
980 \fi
981 \fi
982 \do@actions@nextR}
983

```

14.4 Line number printing

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

```

\ch@cksub@l@ckR 984
\ch@ck@l@ckR 985 \providecommand*{\l@dcalcnm}[3]{%
\fx@l@cksR 986 \ifnum #1 > #2\relax
\affixline@numR 987 \@l@dttempcnta = #1\relax

```

```

988 \advance\@l@tempcnta by -#2\relax
989 \divide\@l@tempcnta by #3\relax
990 \multiply\@l@tempcnta by #3\relax
991 \advance\@l@tempcnta by #2\relax
992 \else
993 \@l@tempcnta=#2\relax
994 \fi}
995
996 \newcommand*{\ch@cksub@l@ckR}{%
997 \ifcase\sub@lockR
998 \or
999 \ifnum\sublock@disp=\@ne
1000 \@l@tempcntb \z@ \@l@tempcnta \@ne
1001 \fi
1002 \or
1003 \ifnum\sublock@disp=\tw@
1004 \else
1005 \@l@tempcntb \z@ \@l@tempcnta \@ne
1006 \fi
1007 \or
1008 \ifnum\sublock@disp=\z@
1009 \@l@tempcntb \z@ \@l@tempcnta \@ne
1010 \fi
1011 \fi}
1012
1013 \newcommand*{\ch@ck@l@ckR}{%
1014 \ifcase\@lockR
1015 \or
1016 \ifnum\lock@disp=\@ne
1017 \@l@tempcntb \z@ \@l@tempcnta \@ne
1018 \fi
1019 \or
1020 \ifnum\lock@disp=\tw@
1021 \else
1022 \@l@tempcntb \z@ \@l@tempcnta \@ne
1023 \fi
1024 \or
1025 \ifnum\lock@disp=\z@
1026 \@l@tempcntb \z@ \@l@tempcnta \@ne
1027 \fi
1028 \fi}
1029
1030 \newcommand*{\f@x@l@cksR}{%
1031 \ifcase\@lockR
1032 \or
1033 \global\@lockR \tw@
1034 \or \or
1035 \global\@lockR \z@
1036 \fi
1037 \ifcase\sub@lockR

```



```

1038 \or
1039   \global\sub@lockR \tw@
1040 \or \or
1041   \global\sub@lockR \z@
1042 \fi}
1043
1044
1045 \newcommand*{\affixline@numR}{%
1046 \ifl@dskipnumber
1047   \global\l@dskipnumberfalse
1048 \else
1049   \ifsublines@
1050     \@l@tempcntb=\subline@numR
1051     \l@dcalcnnum{\subline@numR}{\c@firstsublinenumR}{\c@sublinenumincrementR}%
1052     \ch@cksub@lockR
1053   \else
1054     \@l@tempcntb=\line@numR
1055     \ifx\linenumberlist\empty
1056       \l@dcalcnnum{\line@numR}{\c@firstlinenumR}{\c@linenumincrementR}%
1057     \else
1058       \@l@tempcnta=\line@numR
1059       \edef\rem@inder{\linenumberlist,number\line@numR,%
1060         \edef\sc@n@list{\def\noexpand\sc@n@list
1061           ###1,number\@l@tempcnta,###2|{\def\noexpand\rem@inder{###2}}}%
1062         \sc@n@list\expandafter\sc@n@list\rem@inder|
1063         \ifx\rem@inder\empty\advance\@l@tempcnta\@ne\fi
1064       \fi
1065       \ch@ck@l@ckR
1066     \fi
1067     \ifnum\@l@tempcnta=\@l@tempcntb
1068       \if@twocolumn
1069         \if@firstcolumn
1070           \gdef\l@dld@ta{\llap{\leftlinenumR}}}%
1071         \else
1072           \gdef\l@drd@ta{\rlap{\rightlinenumR}}}%
1073         \fi
1074       \else
1075         \@l@tempcntb=\line@marginR
1076         \ifnum\@l@tempcntb>\@ne
1077           \advance\@l@tempcntb by\page@numR
1078         \fi
1079         \ifodd\@l@tempcntb
1080           \gdef\l@drd@ta{\rlap{\rightlinenumR}}}%
1081         \else
1082           \gdef\l@dld@ta{\llap{\leftlinenumR}}}%
1083         \fi
1084       \fi
1085     \fi
1086     \f@x@l@ckR
1087 \fi}

```

1088

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.

1089 `\list@create{\inserts@listR}`

`\add@insertsR` The right text version.

```

\add@inserts@nextR 1090 \newcommand*{\add@insertsR}{%
1091   \global\let\add@inserts@nextR=\relax
1092   \ifx\inserts@listR\empty \else
1093     \ifx\next@insertR\empty
1094       \ifx\insertlines@listR\empty
1095         \global\noteschanged@true
1096         \gdef\next@insertR{100000}%
1097       \else
1098         \gl@p\insertlines@listR\to\next@insertR
1099       \fi
1100     \fi
1101     \ifnum\next@insertR=\absline@numR
1102       \gl@p\inserts@listR\to\@insertR
1103     \@insertR
1104     \global\let\@insertR=\undefined
1105     \global\let\next@insertR=\empty
1106     \global\let\add@inserts@nextR=\add@insertsR
1107   \fi
1108 \fi
1109 \add@inserts@nextR}
1110
```

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@tempcnta by \clubpenalty
\fi
\@l@tempcntb=\par@lineR \advance\@l@tempcntb \@ne
\ifnum\@l@tempcntb=\num@linesR
  \advance\@l@tempcnta by \widowpenalty
\fi
\ifnum\par@lineR<\num@linesR
  \advance\@l@tempcnta by \interlinepenalty
\fi
\fi
\ifnum\@l@tempcnta=\z@
  \relax
\else
  \ifnum\@l@tempcnta>-10000
    \penalty\@l@tempcnta
  \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.

```

1111 \newcommand*\add@penaltiesL{}\fi
1112 \newcommand*\add@penaltiesR{}\fi
1113

```

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.

```

1114 \newcommand*\flush@notesR}{%
1115   \@xloop
1116   \ifx\inserts@listR\empty \else
1117     \gl@p\inserts@listR\to\@insertR
1118     \@insertR
1119     \global\let\@insertR=\undefined
1120   \repeat}
1121

```

15.1 Outer-level footnote commands

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.

```

\Cfootnote 1136 \renewcommand*{\Bfootnote}[1]{%
\Bfootnote 1137   \ifnumberedpar@
\Efootnote 1138     \ifledRcol
1139       \xright@appenditem{\noexpand\vBfootnote{B}%
1140         {{\l@d@nums}{\@tag}{#1}}}\to\inserts@listR
1141       \global\advance\insert@countR \@ne
1142     \else
1143       \xright@appenditem{\noexpand\vBfootnote{B}%
1144         {{\l@d@nums}{\@tag}{#1}}}\to\inserts@list
1145       \global\advance\insert@count \@ne
1146     \fi
1147   \else
1148     \vBfootnote{B}{{0|0|0|0|0|0|0}{#1}}%
1149   \fi\ignorespaces}

1150 \renewcommand*{\Cfootnote}[1]{%
1151   \ifnumberedpar@
1152     \ifledRcol
1153       \xright@appenditem{\noexpand\vCfootnote{C}%
1154         {{\l@d@nums}{\@tag}{#1}}}\to\inserts@listR

```

```

1155     \global\advance\insert@countR \@ne
1156   \else
1157     \xright@appenditem{\noexpand\vcfootnote{C}%
1158                       {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1159     \global\advance\insert@count \@ne
1160   \fi
1161 \else
1162   \vcfootnote{C}{\l@d@nums}{\@tag}{#1}%
1163 \fi\ignorespaces}

1164 \renewcommand*{\Dfootnote}[1]{%
1165   \ifnumberedpar@
1166     \ifledRcol
1167       \xright@appenditem{\noexpand\vdfootnote{D}%
1168                         {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1169       \global\advance\insert@countR \@ne
1170     \else
1171       \xright@appenditem{\noexpand\vdfootnote{D}%
1172                         {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1173       \global\advance\insert@count \@ne
1174     \fi
1175   \else
1176     \vdfootnote{D}{\l@d@nums}{\@tag}{#1}%
1177   \fi\ignorespaces}

1178 \renewcommand*{\Efootnote}[1]{%
1179   \ifnumberedpar@
1180     \ifledRcol
1181       \xright@appenditem{\noexpand\vefootnote{E}%
1182                         {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1183       \global\advance\insert@countR \@ne
1184     \else
1185       \xright@appenditem{\noexpand\vefootnote{E}%
1186                         {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1187       \global\advance\insert@count \@ne
1188     \fi
1189   \else
1190     \vefootnote{E}{\l@d@nums}{\@tag}{#1}%
1191   \fi\ignorespaces}
1192

```

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

```

\mpBfootnote 1193 \renewcommand*{\mpAfootnote}[1]{%
\mpCfootnote 1194   \ifnumberedpar@
\mpDfootnote 1195     \ifledRcol
\mpEfootnote 1196       \xright@appenditem{\noexpand\mpvAfootnote{A}%
1197                         {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1198       \global\advance\insert@countR \@ne
1199     \else
1200       \xright@appenditem{\noexpand\mpvAfootnote{A}%
1201                         {\l@d@nums}{\@tag}{#1}}\to\inserts@list

```

```

1202 \global\advance\insert@count \@ne
1203 \fi
1204 \else
1205 \mpvAfootnote{A}{{0|0|0|0|0|0|0|0}}-{\#1}}%
1206 \fi\ignorespaces}

1207 \renewcommand*{\mpBfootnote}[1]{%
1208 \ifnumberedpar@
1209 \ifledRcol
1210 \xright@appenditem{\noexpand\mpvBfootnote{B}%
1211 {{\l@d@nums}{\@tag}-{\#1}}}\to\inserts@listR
1212 \global\advance\insert@countR \@ne
1213 \else
1214 \xright@appenditem{\noexpand\mpvBfootnote{B}%
1215 {{\l@d@nums}{\@tag}-{\#1}}}\to\inserts@list
1216 \global\advance\insert@count \@ne
1217 \fi
1218 \else
1219 \mpvBfootnote{B}{{0|0|0|0|0|0|0|0}}-{\#1}}%
1220 \fi\ignorespaces}

1221 \renewcommand*{\mpCfootnote}[1]{%
1222 \ifnumberedpar@
1223 \ifledRcol
1224 \xright@appenditem{\noexpand\mpvCfootnote{C}%
1225 {{\l@d@nums}{\@tag}-{\#1}}}\to\inserts@listR
1226 \global\advance\insert@countR \@ne
1227 \else
1228 \xright@appenditem{\noexpand\mpvCfootnote{C}%
1229 {{\l@d@nums}{\@tag}-{\#1}}}\to\inserts@list
1230 \global\advance\insert@count \@ne
1231 \fi
1232 \else
1233 \mpvCfootnote{C}{{0|0|0|0|0|0|0|0}}-{\#1}}%
1234 \fi\ignorespaces}

1235 \renewcommand*{\mpDfootnote}[1]{%
1236 \ifnumberedpar@
1237 \ifledRcol
1238 \xright@appenditem{\noexpand\mpvDfootnote{D}%
1239 {{\l@d@nums}{\@tag}-{\#1}}}\to\inserts@listR
1240 \global\advance\insert@countR \@ne
1241 \else
1242 \xright@appenditem{\noexpand\mpvDfootnote{D}%
1243 {{\l@d@nums}{\@tag}-{\#1}}}\to\inserts@list
1244 \global\advance\insert@count \@ne
1245 \fi
1246 \else
1247 \mpvDfootnote{D}{{0|0|0|0|0|0|0|0}}-{\#1}}%
1248 \fi\ignorespaces}

1249 \renewcommand*{\mpEfootnote}[1]{%

```

```

1250 \ifnumberedpar@
1251 \ifledRcol
1252   \xright@appenditem{\noexpand\mpvEfootnote{E}%
1253                      {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1254   \global\advance\insert@countR \@ne
1255 \else
1256   \xright@appenditem{\noexpand\mpvEfootnote{E}%
1257                      {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1258   \global\advance\insert@count \@ne
1259 \fi
1260 \else
1261   \mpvEfootnote{E}{\l@d@nums}{\@tag}{#1}%
1262 \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:

<code>\printlinesR</code>	<code>#1</code>	<code> </code>	<code>#2</code>	<code> </code>	<code>#3</code>	<code> </code>	<code>#4</code>	<code> </code>	<code>#5</code>	<code> </code>	<code>#6</code>	<code> </code>	<code>#7</code>
<code>\printlinesR</code>	start-page		line		subline		end-page		line		subline		font

```

1263 \def\printlinesR#1|#2|#3|#4|#5|#6|#7|{\begingroup
1264   \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
1265   \ifl@d@pnum #1\fullstop\fi
1266   \ifledplinenum \linenumr@p{#2}\Rlineflag\else \symplinenum\fi
1267   \ifl@d@ssub \fullstop \sublinenumr@p{#3}\fi
1268   \ifl@d@dash \endashchar\fi
1269   \ifl@d@pnum #4\fullstop\fi
1270   \ifl@d@elin \linenumr@p{#5}\Rlineflag\fi
1271   \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumr@p{#6}\fi
1272 \endgroup}
1273
1274 \let\ledsavedprintlines\printlines
1275

```

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.

```

1276 \list@create{\labelref@listR}
1277

```

`\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.

```

1278 \renewcommand*{\edlabel}[1]{\@bsphack
1279   \ifledRcol
1280     \write\linenum@outR{\string\@lab}%
1281     \ifx\labelref@list\empty
1282       \xdef\label@refs{\zz@@@}%
1283     \else
1284       \gl@p\labelref@listR\to\label@refs
1285     \fi
1286     \protected@write\@auxout{%
1287       {\string\l@dmake@labelsR\space\thepage|\label@refs|{#1}}%
1288     \else
1289       \write\linenum@out{\string\@lab}%
1290       \ifx\labelref@list\empty
1291         \xdef\label@refs{\zz@@@}%
1292       \else
1293         \gl@p\labelref@list\to\label@refs
1294       \fi
1295     \fi
1296     \protected@write\@auxout{%
1297       {\string\l@dmake@labelsR\space\thepage|\label@refs|{#1}}%
1298     \@esphack}
1299

```

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

```

1300 \def\l@dmake@labelsR#1|#2|#3|#4{%
1301   \expandafter\ifx\csname the@label#4\endcsname \relax\else
1302     \led@warn@DuplicateLabel{#4}%
1303   \fi
1304   \expandafter\gdef\csname the@label#4\endcsname{#1|#2\Rlineflag|#3}%
1305   \ignorespaces}
1306 \AtBeginDocument{%
1307   \def\l@dmake@labelsR#1|#2|#3|#4{%
1308 }
1309

```

`\@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.

```

1310 \renewcommand*{\@lab}{%
1311   \ifledRcol
1312     \xright@appenditem{\linenumr@p{\line@numR}}|%
1313     \ifsublines@ \sublinenumr@p{\subline@numR}\else 0\fi}%

```



```

1314     \to\labelref@listR
1315 \else
1316   \xright@appenditem{\linenumr@p{\line@num}}{|%
1317     \ifsublines@ \sublinenumr@p{\subline@num}\else 0\fi}%
1318   \to\labelref@list
1319 \fi}
1320

```

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 1321 \newcount\sidenote@marginR
1322 \renewcommand*{\sidenotemargin}[1]{\%
1323   \l@dgetsidenote@margin{#1}%
1324   \ifnum\@l@dttempcntb>\m@ne
1325     \ifledRcol
1326       \global\sidenote@marginR=\@l@dttempcntb
1327     \else
1328       \global\sidenote@margin=\@l@dttempcntb
1329     \fi
1330   \fi}}
1331 \sidenotemargin{right}
1332 \global\sidenote@margin=\@ne
1333

```

`\l@dlsnote` The ‘footnotes’ for left, right, and moveable sidenotes. The whole scheme is rem-

`\l@drsnote` iniscent of the critical footnotes code.

```

\l@dcsnote 1334 \renewcommand*{\l@dlsnote}[1]{%
1335   \ifnumberedpar@
1336     \ifledRcol
1337       \xright@appenditem{\noexpand\l@dlsnote{#1}}{|%
1338         \to\inserts@listR
1339       \global\advance\insert@countR \@ne
1340     \else
1341       \xright@appenditem{\noexpand\l@dlsnote{#1}}{|%
1342         \to\inserts@list
1343       \global\advance\insert@count \@ne
1344     \fi
1345   \fi\ignorespaces}
1346 \renewcommand*{\l@drsnote}[1]{%
1347   \ifnumberedpar@
1348     \ifledRcol
1349       \xright@appenditem{\noexpand\l@drsnote{#1}}{|%
1350         \to\inserts@listR
1351       \global\advance\insert@countR \@ne
1352     \else

```

```

1353 \xright@appenditem{\noexpand\vl@drsnote{#1}}%
1354 \to\inserts@list
1355 \global\advance\insert@count \@ne
1356 \fi
1357 \fi\ignorespaces}
1358 \renewcommand*{\l@dcnote}[1]{%
1359 \ifnumberedpar@
1360 \ifledRcol
1361 \xright@appenditem{\noexpand\vl@dcnote{#1}}%
1362 \to\inserts@listR
1363 \global\advance\insert@countR \@ne
1364 \else
1365 \xright@appenditem{\noexpand\vl@dcnote{#1}}%
1366 \to\inserts@list
1367 \global\advance\insert@count \@ne
1368 \fi
1369 \fi\ignorespaces}
1370

```

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

```

1371 \newcommand*{\affixside@noteR}{%
1372 \gdef\@templ@d{}%
1373 \ifx\@templ@d\l@dcnotetext \else
1374 \if@twocolumn
1375 \if@firstcolumn
1376 \setl@dlp@rbox{\l@dcnotetext}%
1377 \else
1378 \setl@drp@rbox{\l@dcnotetext}%
1379 \fi
1380 \else
1381 \@l@tempcntb=\sidenote@marginR
1382 \ifnum\@l@tempcntb>\@ne
1383 \advance\@l@tempcntb by\page@num
1384 \fi
1385 \ifodd\@l@tempcntb
1386 \setl@drp@rbox{\l@dcnotetext}%
1387 \else
1388 \setl@dlp@rbox{\l@dcnotetext}%
1389 \fi
1390 \fi
1391 \fi}
1392

```

18 Familiar footnotes

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

```

1393 \renewcommand{\l@dbfnote}[1]{%

```

```

1394 \ifnumberedpar@
1395   \ifledRcol
1396     \xright@appenditem{\noexpand\vl@dbfnote{{#1}}{\@thefnmark}}%
1397       \to\inserts@listR
1398     \global\advance\insert@countR \@ne
1399   \else
1400     \xright@appenditem{\noexpand\vl@dbfnote{{#1}}{\@thefnmark}}%
1401       \to\inserts@list
1402     \global\advance\insert@count \@ne
1403   \fi
1404 \fi\ignorespaces}
1405

```

`\normalbfnoteX`

```

1406 \renewcommand{\normalbfnoteX}[2]{%
1407   \ifnumberedpar@
1408     \ifledRcol
1409       \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}{\@nameuse{thefootnote#1}}}%
1410         \to\inserts@listR
1411       \global\advance\insert@countR \@ne
1412     \else
1413       \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}{\@nameuse{thefootnote#1}}}%
1414         \to\inserts@list
1415       \global\advance\insert@count \@ne
1416     \fi
1417 \fi\ignorespaces}
1418

```

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.

```

1419 \chardef\next=\catcode'\&
1420 \catcode'\&=\active
1421

```

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

```

1422 \newenvironment{astanza}{%
1423   \startstanzahook
1424   \catcode'\&=\active
1425   \global\stanza@count\@ne
1426   \ifnum\usernamecount{sza@00}=\z@
1427     \let\stanza@hang\relax
1428     \let\endlock\relax
1429   \else
1430   %% \interlinepenalty\@M % this screws things up, but I don't know why

```

```

1431 \rightskip\z@ plus 1fil\relax
1432 \fi
1433 \ifnum\usernamecount{szp000}=\z@
1434 \let\sza@penalty\relax
1435 \fi
1436 \def&{%
1437 \endlock\mbox{}}%
1438 \sza@penalty
1439 \global\advance\stanza@count\@ne
1440 \@astanza@line}%
1441 \def\&{%
1442 \endlock\mbox{}}
1443 \pend
1444 \endstanzaextra}%
1445 \pstart
1446 \@astanza@line
1447 }{}
1448

```

`\@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.

```

1449 \newcommand*{\@astanza@line}{%
1450 \parindent=\csname szp@number\stanza@count @\endcsname\stanzaindentbase
1451 \par
1452 \stanza@hang%\mbox{}}%
1453 \ignorespaces}
1454

```

Lastly reset the modified category codes.

```

1455 \catcode'\&=\next
1456

```

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 1457 \providecommand*{\newnamebox}[1]{%
\unvnamebox 1458 \expandafter\newbox\csname #1\endcsname}
\namebox 1459 \providecommand*{\setnamebox}[1]{%
1460 \expandafter\setbox\csname #1\endcsname}
1461 \providecommand*{\unhnamebox}[1]{%
1462 \expandafter\unhbox\csname #1\endcsname}
1463 \providecommand*{\unvnamebox}[1]{%
1464 \expandafter\unvbox\csname #1\endcsname}

```

```

1465 \providecommand*\namebox}[1]{%
1466     \csname #1\endcsname}
1467

```

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

```

\usenamecount 1468 \providecommand*\newnamecount}[1]{%
1469     \expandafter\newcount\csname #1\endcsname}
1470 \providecommand*\usenamecount}[1]{%
1471     \csname #1\endcsname}
1472

```

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 default is 10 chunk pairs.

```

1473 \newcount\l@dc@maxchunks
1474 \newcommand*\maxchunks}[1]{\l@dc@maxchunks=#1}
1475 \maxchunks{10}
1476

```

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

```

\l@dnumpstartsR 1477 \newcount\l@dnumpstartsR
1478

```

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

```

\l@psclR 1479 \newcount\l@dp scl
1480 \newcount\l@dp scl
1481

```

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

```

1482 \newcommand*\l@dsetuprawboxes){%
1483     \@l@dttempcntb=\l@dc@maxchunks
1484     \loop\ifnum\@l@dttempcntb>\z@
1485         \newnamebox{\l@dLcolrawbox\the\@l@dttempcntb}
1486         \newnamebox{\l@dRcolrawbox\the\@l@dttempcntb}
1487         \advance\@l@dttempcntb \m@ne
1488     \repeat}
1489

```

`\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@dmmaxlinesinpar1`, etc., and `\l@dzeromaxlinecounts` zeroes all of them.

```

1490 \newcommand*{\l@dsetupmaxlinecounts}{%
1491   \@l@dttempcntb=\l@dc@maxchunks
1492   \loop\ifnum\@l@dttempcntb>\z@
1493     \newnamecount{\l@dmmaxlinesinpar\the\@l@dttempcntb}
1494     \advance\@l@dttempcntb \m@ne
1495   \repeat}
1496 \newcommand*{\l@dzeromaxlinecounts}{%
1497   \begingroup
1498   \@l@dttempcntb=\l@dc@maxchunks
1499   \loop\ifnum\@l@dttempcntb>\z@
1500     \global\usenamecount{\l@dmmaxlinesinpar\the\@l@dttempcntb}=\z@
1501     \advance\@l@dttempcntb \m@ne
1502   \repeat
1503   \endgroup}
1504
```

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

```

1505 \AtBeginDocument{%
1506   \l@dsetuppraxboxes
1507   \l@dsetupmaxlinecounts
1508   \l@dzeromaxlinecounts
1509   \l@dnumstartsl=\z@
1510   \l@dnumstartsr=\z@
1511   \l@dpscl=\z@
1512   \l@dpscr=\z@}
1513
```

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 1514 \newif\ifl@dusedbabel
\l@dusedbabeltrue 1515 \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
```

```

1516 \newif\ifl@dsamelang
1517   \l@dsamelangtrue

```

`\l@dsamelang` 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.

```

1518 \newcommand*{\l@dsamelang}{%
1519   \l@dsamelangfalse
1520   \edef\@tempa{\theledlanguageL}\edef\@tempb{\theledlanguageR}%
1521   \ifx\@tempa\@tempb
1522     \l@dsamelangtrue
1523   \fi}
1524

```

`\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 `\`.

```

1525 \newcommand*{\l@dbbl@set@language}[1]{%
1526   \edef\language{#1}%
1527   \select@language{\language}%
1528   \if@filesw
1529     \protected@write\@auxout{\string\select@language{\language}}%
1530     \addtocontents{toc}{\string\select@language{\language}}%
1531     \addtocontents{lof}{\string\select@language{\language}}%
1532     \addtocontents{lot}{\string\select@language{\language}}%
1533   \fi}
1534

```

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 1535 \providecommand{\selectlanguage}[1]{%
1536 \newcommand*{\l@duselanguage}[1]{%
1537   \gdef\theledlanguageL{}
1538   \gdef\theledlanguageR{}
1539

```

Now do the babel fix or polyglossia, if necessary.

```

1540 \AtBeginDocument{%
1541   \ifundefined{xpg@main@language}{%
1542     \ifundefined{bbl@main@language}{%

```

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

```

1543 \l@dusedbabelfalse
1544 \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.

```

1545 \l@dusedbabeltrue
1546 \let\l@doldselectlanguage\selectlanguage
1547 \let\l@doldbbl@set@language\bbl@set@language
1548 \let\bbl@set@language\l@dbbl@set@language
1549 \renewcommand{\selectlanguage}[1]{%
1550   \l@doldselectlanguage{#1}%
1551   \ifledRcol \gdef\theledlanguageR{#1}%
1552   \else      \gdef\theledlanguageL{#1}%
1553   \fi}

```

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

```

1554 \renewcommand*{\l@duselanguage}[1]{%
1555   \l@doldselectlanguage{#1}}

```

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

```

1556 \gdef\theledlanguageL{\bbl@main@language}%
1557 \gdef\theledlanguageR{\bbl@main@language}%
1558 }%
1559 }

```

If on Polyglossia

```

1560 { \apptocmd{\xpg@set@language}{%
1561   \ifledRcol \gdef\theledlanguageR{#1}%
1562   \else      \gdef\theledlanguageL{#1}%
1563   \fi}%
1564   \let\l@duselanguage\xpg@set@language
1565   \gdef\theledlanguageL{\xpg@main@language}%
1566   \gdef\theledlanguageR{\xpg@main@language}%
1567 % \end{macrocode}
1568 % That's it.
1569 % \begin{macrocode}
1570 }}

```

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.

```

1571 \newcommand*{\Columns}{%
1572   \ifnum\l@dnumpstartsL=\l@dnumpstartsR\else
1573     \led@err@BadLeftRightPstarts{\the\l@dnumpstartsL}{\the\l@dnumpstartsR}%
1574   \fi

```


Start a group and zero counters, etc.

```

1575 \begingroup
1576 \l@dzeroopenalties
1577 \endgraf\global\num@lines=\prevgraf
1578 \global\num@linesR=\prevgraf
1579 \global\par@line=\z@
1580 \global\par@lineR=\z@
1581 \global\l@dpscL=\z@
1582 \global\l@dpscR=\z@

```

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

```

1583 \check@pstarts
1584 \loop\if@pstarts

```

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

```

1585 \global\advance\l@dpscL \@ne
1586 \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

```

1587 \checkraw@text
1588 \l@dcchecklang
1589 { \loop\if@raw@text

```

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

```

1590 \ifl@dsamelang
1591 \do@lineL
1592 \do@lineR
1593 \else
1594 \l@duselanguage{\theledlanguageL}%
1595 \do@lineL
1596 \l@duselanguage{\theledlanguageR}%
1597 \do@lineR
1598 \fi
1599 \hb@xt@ \hsize{%
1600 \unhbox\l@leftbox
1601 \hfill \columnseparator \hfill
1602 \unhbox\l@rightbox
1603 }%
1604 \checkraw@text
1605 \repeat}

```

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

```

1606 \@writelinesinparL
1607 \@writelinesinparR
1608 \check@pstarts
1609 \repeat

```

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

```

1610 \flush@notes
1611 \flush@notesR
1612 \endgroup

1613 \global\l@dpscL=\z@
1614 \global\l@dpscR=\z@
1615 \global\l@dnumstartsL=\z@
1616 \global\l@dnumstartsR=\z@
1617 \ignorespaces}
1618

```

`\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).

```

1619 \newcommand*{\columnseparator}{%
1620 \smash{\rule[-0.2\baselineskip]{\columnrulewidth}{1.05\baselineskip}}}
1621 \newdimen\columnrulewidth
1622 \columnrulewidth=\z@
1623

```

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

```

\@pstartstrue 1624 \newif\if@pstarts
\@pstartsfalse 1625 \newcommand*{\check@pstarts}{%
\check@pstarts 1626 \@pstartsfalse
1627 \ifnum\l@dnumstartsL>\l@dpscL
1628 \@pstartstrue
1629 \else
1630 \ifnum\l@dnumstartsR>\l@dpscR
1631 \@pstartstrue
1632 \fi
1633 \fi}
1634

```

`\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 sets `\araw@textfalse`.

```

\checkraw@text 1635 \newif\ifaraw@text
1636 \araw@textfalse
1637 \newcommand*{\checkraw@text}{%
1638 \araw@textfalse
1639 \ifvbox\namebox{\l@dLcolrawbox\the\l@dpscL}
1640 \araw@texttrue
1641 \else
1642 \ifvbox\namebox{\l@dRcolrawbox\the\l@dpscR}
1643 \araw@texttrue
1644 \fi

```

```

1645 \fi}
1646

```

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

```

1647 \newcommand*{\@writelinesinparL}{%
1648 \edef\next{%
1649 \write\linenum@out{\string\@pend[\the\@donereallinesL]}}%
1650 \next
1651 \global\@donereallinesL \z@}
1652 \newcommand*{\@writelinesinparR}{%
1653 \edef\next{%
1654 \write\linenum@outR{\string\@pendR[\the\@donereallinesR]}}%
1655 \next
1656 \global\@donereallinesR \z@}
1657

```

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` 1658 \newcount\numpagelinesL
1659 \newcount\numpagelinesR
1660 \newcount\l@dminpagelines
1661

`\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.

```

1662 \newcommand*{\Pages}{%
1663 \typeout{}
1664 \typeout{***** PAGES *****}
1665 \ifnum\l@dnumpstartsL=\l@dnumpstartsR\else
1666 \led@err@BadLeftRightPstarts{\the\l@dnumpstartsL}{\the\l@dnumpstartsR}%
1667 \fi

```

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

```

1668 \clearto\l@devenpage
1669 \begingroup
1670 \l@dzeropenalties
1671 \endgraf\global\num@lines=\prevgraf
1672 \global\num@linesR=\prevgraf
1673 \global\par@line=\z@
1674 \global\par@lineR=\z@
1675 \global\l@dpscL=\z@
1676 \global\l@dpscR=\z@

```

```

1677     \writtenlinesLfalse
1678     \writtenlinesRfalse

    Check if there are chunks to be processed.
1679     \check@pstarts
1680     \loop\if@pstarts

    Loop over the number of chunks, incrementing the chunk counts (\l@dpscL and
    \l@dpscR are chunk (box) counts.)
1681     \global\advance\l@dpscL \@ne
1682     \global\advance\l@dpscR \@ne

    Calculate the maximum number of real text lines in the chunk pair, storing the
    result in the relevant \l@dmaxlinesinpar.
1683     \getlinesfromparlistL
1684     \getlinesfromparlistR
1685     \l@dcalc@maxoftwo{\@cs@linesinparL}{\@cs@linesinparR}%
1686     {\@cs@linesinparR}%
1687     \check@pstarts
1688     \repeat

    Zero the counts again, ready for the next bit.
1689     \global\l@dpscL=\z@
1690     \global\l@dpscR=\z@

    Get the number of lines on the first pair of pages and store the minimum in
    \l@dminpagelines.
1691     \getlinesfrompagelistL
1692     \getlinesfrompagelistR
1693     \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
1694     {\@cs@linesonpageR}%

    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.
1695     \check@pstarts
1696     \if@pstarts

    Increment the chunk counts to get the first pair.
1697     \global\advance\l@dpscL \@ne
1698     \global\advance\l@dpscR \@ne

    We haven't processed any lines from these chunks yet, so zero the respective line
    counts.
1699     \global\@donereallinesL=\z@
1700     \global\@donetotallinesL=\z@
1701     \global\@donereallinesR=\z@
1702     \global\@donetotallinesR=\z@

    Start a loop over the boxes (chunks).
1703     \checkraw@text
1704 %     \begingroup
1705 {     \loop\if@raw@text

```

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

```

1706      \checkpageL
1707      \l@duselanguage{\theledlanguageL}%
1708      %%
1709      \beginngroup
1709      {
1709      \loop\ifl@dsamepage

```

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

```

1710      \do@lineL
1711      \advance\numpagelinesL \@ne
1712      \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.

```

1713      \get@nextboxL
1714      \checkpageL
1715      \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.

```

1716      \ifl@dpagfull
1717      \@writelinesonpageL{\the\numpagelinesL}%
1718      \else
1719      \@writelinesonpageL{1000}%
1720      \fi

```

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

```

1721      \numpagelinesL \z@
1722      \clearl@dleftpage }%

```

Now do the same for the right text.

```

1723      \checkpageR
1724      \l@duselanguage{\theledlanguageR}%
1725      {
1725      \loop\ifl@dsamepage
1726      \do@lineR
1727      \advance\numpagelinesR \@ne
1728      \hb@xt@ \hsize{\ledstrutR\unhbox\l@drightbox}%
1729      \get@nextboxR
1730      \checkpageR
1731      \repeat
1732      \ifl@dpagfull
1733      \@writelinesonpageR{\the\numpagelinesR}%
1734      \else
1735      \@writelinesonpageR{1000}%
1736      \fi
1737      \numpagelinesR=\z@

```

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

```
1738      \clearl@drihtpage}
```

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.

```
1739      \checkraw@text
1740      \ifaraw@text
1741          \getlinesfrompagelistL
1742          \getlinesfrompagelistR
1743          \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
1744                      {\l@dminpagelines}%
1745      \fi
1746      \repeat}
```

We have now output the text from all the chunks.

```
1747      \fi
```

Make sure that there are no inserts hanging around.

```
1748      \flush@notes
1749      \flush@notesR
1750      \endgroup
```

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

```
1751      \global\l@dpscL=\z@
1752      \global\l@dpscR=\z@
1753      \global\l@dnumpestartsL=\z@
1754      \global\l@dnumpestartsR=\z@
1755      \ignorespaces}
1756
```

`\ledstrutL` Struts inserted into leftand right text lines.

```
\ledstrutR 1757 \newcommand*{\ledstrutL}{\strut}
1758 \newcommand*{\ledstrutR}{\strut}
1759
```

`\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@drihtpage` get us onto an odd and even page, respectively, checking that we end up on the immediately next page.

```
1760 \providecommand{\cleartoevenpage}[1][\@empty]{%
1761     \clearpage
1762     \ifodd\c@page\hbox{#1}\clearpage\fi}
1763 \newcommand*{\cleartol@devenpage}{%
1764     \ifdim\pagetotal<\topskip% on an empty page
1765     \else
1766     \clearpage
1767     \fi
1768     \ifodd\c@page\hbox{#1}\clearpage\fi}
```

```

1769 \newcommand*{\clearl@dleftpage}{%
1770   \clearpage
1771   \ifodd\c@page\else
1772     \led@err@LeftOnRightPage
1773     \hbox{}%
1774   \cleardoublepage
1775   \fi}
1776 \newcommand*{\clearl@drightpage}{%
1777   \clearpage
1778   \ifodd\c@page
1779     \led@err@RightOnLeftPage
1780     \hbox{}%
1781   \cleartoevenpage
1782   \fi}
1783

```

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

```

\@cs@linesinparR 1784 \newcommand*{\getlinesfromparlistL}{%
1785   \ifx\linesinpar@listL\empty
1786     \gdef\@cs@linesinparL{0}%
1787   \else
1788     \gl@p\linesinpar@listL\to\@cs@linesinparL
1789   \fi}
1790 \newcommand*{\getlinesfromparlistR}{%
1791   \ifx\linesinpar@listR\empty
1792     \gdef\@cs@linesinparR{0}%
1793   \else
1794     \gl@p\linesinpar@listR\to\@cs@linesinparR
1795   \fi}
1796

```

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

```

\@cs@linesonpageR 1797 \newcommand*{\getlinesfrompagelistL}{%
1798   \ifx\linesonpage@listL\empty
1799     \gdef\@cs@linesonpageL{1000}%
1800   \else
1801     \gl@p\linesonpage@listL\to\@cs@linesonpageL
1802   \fi}
1803 \newcommand*{\getlinesfrompagelistR}{%
1804   \ifx\linesonpage@listR\empty
1805     \gdef\@cs@linesonpageR{1000}%
1806   \else
1807     \gl@p\linesonpage@listR\to\@cs@linesonpageR
1808   \fi}
1809

```

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

```
1810 \newcommand*{\@writelinesonpageL}[1]{%
1811   \edef\next{\write\linenum@out{\string\@lopL{#1}}}%
1812   \next}
1813 \newcommand*{\@writelinesonpageR}[1]{%
1814   \edef\next{\write\linenum@outR{\string\@lopR{#1}}}%
1815   \next}
1816
```

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

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

```
1817 \newcommand*{\l@dcalc@maxoftwo}[3]{%
1818   \ifnum #2>#1\relax
1819     #3=#2\relax
1820   \else
1821     #3=#1\relax
1822   \fi}
1823 \newcommand*{\l@dcalc@minoftwo}[3]{%
1824   \ifnum #2<#1\relax
1825     #3=#2\relax
1826   \else
1827     #3=#1\relax
1828   \fi}
1829
```

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

```
\checkpageL 1830 \newif\ifl@dsamepage
\checkpageR 1831   \l@dsamepagetrue
1832 \newif\ifl@dpagfull
1833 \newcommand*{\checkpageL}{%
1834   \l@dpagfulltrue
1835   \l@dsamepagetrue
1836   \check@goal
1837   \ifdim\pagetotal<\ledthegoal
1838     \ifnum\numpagelinesL<\l@dminpagelines
1839       \else
1840         \l@dsamepagefalse
1841         \l@dpagfullfalse
1842       \fi
1843     \else
```



```

1844 \l@dsamepagefalse
1845 \l@dpagelltrue
1846 \fi}
1847 \newcommand*{\checkpageR}{%
1848 \l@dpagelltrue
1849 \l@dsamepagetrue
1850 \check@goal
1851 \ifdim\pagetotal<\ledthegoal
1852 \ifnum\numpagelinesR<\l@dminpagelines
1853 \else
1854 \l@dsamepagefalse
1855 \l@dpagellfalse
1856 \fi
1857 \else
1858 \l@dsamepagefalse
1859 \l@dpagelltrue
1860 \fi}
1861

```

`\ledthegoal` `\ledthegoal` is the amount of space allowed to be 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`.

```

1862 \newdimen\ledthegoal
1863 \newcommand*{\goalfraction}{0.9}
1864 \newcommand*{\check@goal}{%
1865 \ledthegoal=\goalfraction\pagegoal}
1866

```

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

```

\ifwrittenlinesL 1867 \newif\ifwrittenlinesL
1868 \newif\ifwrittenlinesR
1869

```

`\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.

```

1870 \newcommand*{\get@nextboxL}{%
1871 \ifvbox\namebox{\l@dLcolrawbox\the\l@dpscl}% box is not empty

    The current box is not empty; do nothing.

1872 \else%                                box is empty

    The box is empty; check if enough lines (real and blank) have been output.

1873 \ifnum\usenamecount{\l@dmaxlinesinpar\the\l@dpscl}>\@donetotallinesL
1874 \else

    Sufficient lines have been output.

1875 \ifwrittenlinesL
1876 \else

```

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

```

1877      \@writelinesinparL
1878      \writtenlinesLtrue
1879      \fi
1880      \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).

1881      \writtenlinesLfalse
1882      \l@dcalc@maxoftwo{\the\usernamecount{l@dmaxlinesinpar\the\l@dpscL}}%
1883                      {\the\@donetotallinesL}%
1884                      {\usernamecount{l@dmaxlinesinpar\the\l@dpscL}}%
1885      \global\@donetotallinesL \z@
1886      \global\advance\l@dpscL \@ne
1887      \fi
1888      \fi
1889      \fi}

1890 \newcommand*{\get@nextboxR}{%
1891   \ifvbox\namebox{l@dRcolrawbox\the\l@dpscR}% box is not empty
1892   \else% box is empty
1893     \ifnum\usernamecount{l@dmaxlinesinpar\the\l@dpscR}>\@donetotallinesR
1894     \else
1895       \ifwrittenlinesR
1896       \else
1897         \@writelinesinparR
1898         \writtenlinesRtrue
1899       \fi
1900       \ifnum\l@dnumpstartsR>\l@dpscR
1901       \writtenlinesRfalse
1902       \l@dcalc@maxoftwo{\the\usernamecount{l@dmaxlinesinpar\the\l@dpscR}}%
1903                       {\the\@donetotallinesR}%
1904                       {\usernamecount{l@dmaxlinesinpar\the\l@dpscR}}%
1905       \global\@donetotallinesR \z@
1906       \global\advance\l@dpscR \@ne
1907     \fi
1908   \fi
1909 \fi}
1910
```

25 The End

i/code_i

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 praeripiat 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 cuspidē 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.

```

1911 (*villon)
1912 %%% villon.tex Example parallel columns
1913 \documentclass{article}
1914 \addtolength{\textheight}{-10\baselineskip}
1915 \usepackage{ledmac,ledpar}
1916 %% Use r instead of R to flag right text line numbers
1917 \renewcommand{\Rlineflag}{r}
1918 %% Use the flag in the notes
1919 \let\oldBfootfmt\Bfootfmt
1920 \renewcommand{\Bfootfmt}[3]{%
1921   \let\printlines\printlinesR
1922   \oldBfootfmt{#1}{#2}{#3}}
1923 \begin{document}
1924
1925 I thought that limericks were peculiarly English, but this appears not
1926 to be the case. As with most limericks this one is by Anonymous.
1927
1928 \vspace*{\baselineskip}
1929
1930 \begin{pairs}
1931 %% no indentation
1932 \setstanzaindents{0,0,0,0,0,0,0,0,0}
1933 %% no number flag
1934 \renewcommand{\Rlineflag}{}
1935 %% draw a rule and widen the columns
1936 \setlength{\columnrulewidth}{0.4pt}
1937 \setlength{\Lcolwidth}{0.46\textwidth}
1938 \setlength{\Rcolwidth}{\Lcolwidth}
1939
1940 \begin{Leftside}
1941 %% set left text line numbering sequence
1942 \firstlinenum{2}
1943 \linenumincrement{2}
1944 \linenummargin{left}
1945 \beginnumbering
1946 \stanza
1947 Il y avait un jeune homme de Dijon, &
1948 Qui n'avait que peu de religion. &
1949 Il dit: 'Quant \'{a} moi, &
1950 Je d'\{e\}teste tous les trois, &
1951 Le P\{e\}re, et le Fils, et le Pigeon.' \&
1952 \endnumbering
1953 \end{Leftside}

```

```

1954
1955 \begin{Rightside}
1956 %% different right text line numbering sequence
1957 \firstlinenum{1}
1958 \linenumincrement{2}
1959 \linenummargin{right}
1960 \beginnumbering
1961 \stanza
1962 There was a young man of Dijon, &
1963 Who had only a little religion, &
1964 He said: 'As for me, &
1965 I detest all the three, &
1966 The Father, the Son, and the Pigeon.' \&
1967 \endnumbering
1968 \end{Rightside}
1969
1970 \Columns
1971 \end{pairs}
1972
1973 \vspace*{\baselineskip}
1974
1975     The following is verse \textsc{lxixiii} of Fran\c{c}ois Villon's
1976 \textit{Le Testament} (The Testament), composed in 1461.
1977
1978 %% Allow for hanging indentation for long lines
1979 \setstanzaindents{1,0,0,0,0,0,0,0}
1980 %% Columns wider than the default
1981 \setlength{\Lcolwidth}{0.46\textwidth}
1982 \setlength{\Rcolwidth}{\Lcolwidth}
1983 \vspace*{\baselineskip}
1984
1985 \begin{pairs}
1986 \begin{Leftside}
1987 \firstlinenum{2}
1988 \linenumincrement{2}
1989 \linenummargin{left}
1990 \beginnumbering
1991 \stanza
1992 Dieu mercy et Tacque Thibault, &
1993 Qui tant d'eaue froid m'a fait boire, &
1994 Mis en bas lieu, non pas en hault, &
1995 Mengier d'angoisse maints \edtext{poire}{\lemma{poire d'angoisse}}%
1996 \Afootnote{This has a triple meaning: literally it is the fruit of the
1997 choke pear,
1998 figuratively it means 'bitter fruit', and it also refers to a torture
1999 instrument.}}, &
2000 Enferr'\{e} \ldots Quant j'en ay memoire, &
2001 Je Prie pour luy \edtext{\textit{et reliqua}}{\Afootnote{and so on}}, &
2002 Que Dieu luy doint, et voire, voire! &
2003 Ce que je pense \ldots \textit{et cetera}. \&

```

```

2004 \endnumbering
2005 \end{Leftside}
2006
2007 \begin{Rightside}
2008 \firstlinenum{2}
2009 \linenumincrement{2}
2010 \linenummargin{right}
2011 \beginnumbering
2012 \stanza
2013 Thanks to God --- and to \edtext{Tacque Thibaud}{%
2014   \Bfootnote{A favourite of Jean, Duc de Berry and loathed for his exactions
2015     and debauchery. Villon uses his name as an insulting nickname for
2016     Thibaud d'Auxigny, the Bishop of Orl\{'e}ans.}} &
2017 Who made me drink so much \edtext{cold water}{%
2018   \Bfootnote{Can either refer to the normal prison diet of bread and
2019     water or to a common medieval torture which involved forced drinking
2020     of cold water.}}, &
2021 Put me underground instead of higher up &
2022 And made me eat such bitter fruit, &
2023 In chains \ldots When I think of this, &
2024 I pray for him---\textit{et reliqua;} &
2025 May God grant him (yes, by God) &
2026 What I think \ldots \textit{et cetera}. \&
2027 \endnumbering
2028 \end{Rightside}
2029
2030 \Columns
2031 \end{pairs}
2032
2033 \vspace*{\baselineskip}
2034
2035   The translation and notes are by Anthony Bonner,
2036 \textit{The Complete Works of Fran\c{c}ois Villon}, published by
2037 Bantam Books in 1960.
2038
2039 \end{document}
2040
2041 </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.

```

2042 <*djd17nov>
2043 %%% This is djd17nov.tex, a sample critical text edition
2044 %%% written in LaTeX2e with the ledmac and ledpar packages.
2045 %%% (c) 2003--2004 by Dr. Dirk-Jan Dekker,

```

```

2046 %% Radboud University, Nijmegen (The Netherlands)
2047 %% (PRW) Modified slightly by PRW to fit the ledpar manual
2048
2049 \documentclass[10pt, letterpaper, twoside]{article}
2050 \usepackage[latin,english]{babel}
2051 \usepackage{makeidx}
2052 \usepackage{ledmac,ledpar}
2053 \lineation{section}
2054 \linenummargin{inner}
2055 \sidenotemargin{outer}
2056
2057 \makeindex
2058
2059 \renewcommand{\notenumfont}{\footnotesize}
2060 \newcommand{\notetextfont}{\footnotesize}
2061
2062 %\let\Afootnoterule=\relax
2063 \let\Bfootnoterule=\relax
2064 \let\Cfootnoterule=\relax
2065
2066 \addtolength{\skip\Afootins}{1.5mm}
2067 %\addtolength{\skip\Bfootins}{1.5mm}
2068 %\addtolength{\skip\Cfootins}{1.5mm}
2069
2070 \makeatletter
2071
2072 \renewcommand*{\para@vfootnote}[2]{%
2073   \insert\csname #1footins\endcsname
2074   \bgroup
2075     \notefontsetup
2076     \interlinepenalty=\interfootnotelinepenalty
2077     \floatingpenalty=\@MM
2078     \splittopskip=\ht\strutbox \splitmaxdepth=\dp\strutbox
2079     \leftskip=\z@skip \rightskip=\z@skip
2080     \l@dparsfootsspec #2\ledplinenumtrue%           new from here
2081     \ifnum\@nameuse{previous@#1@number}=\l@dparsestartline\relax
2082       \ledplinenumfalse
2083     \fi
2084     \ifnum\previous@page=\l@dparsestartpage\relax
2085     \else \ledplinenumtrue \fi
2086     \ifnum\l@dparsestartline=\l@dparseendline\relax
2087     \else \ledplinenumtrue \fi
2088     \expandafter\xdef\csname previous@#1@number\endcsname{\l@dparsestartline}%
2089     \xdef\previous@page{\l@dparsestartpage}%           to here
2090     \setbox0=\vbox{\hsize=\maxdimen
2091       \noindent\csname #1footfmt\endcsname#2}%
2092     \setbox0=\hbox{\unvvh0}%
2093     \dp0=0pt
2094     \ht0=\csname #1footfudgefactor\endcsname\wd0
2095     \box0

```

```

2096 \penalty0
2097 \egroup
2098 }
2099
2100 \newcommand*{\previous@A@number}{-1}
2101 \newcommand*{\previous@B@number}{-1}
2102 \newcommand*{\previous@C@number}{-1}
2103 \newcommand*{\previous@page}{-1}
2104
2105 \newcommand{\abb}[1]{#1%
2106 \let\rbracket\nobrak\relax}
2107 \newcommand{\nobrak}{\textnormal{}}
2108 \newcommand{\morenoexpands}{%
2109 \let\abb=0%
2110 }
2111
2112 \newcommand{\Aparafootfmt}[3]{%
2113 \ledsetnormalparstuff
2114 \scriptsize
2115 \notenumfont\printlines#1\enspace
2116 % \lemmafont#1/#2\enskip
2117 \notetextfont
2118 #3\penalty-10\hskip 1em plus 4em minus.4em\relax}
2119
2120 \newcommand{\Bparafootfmt}[3]{%
2121 \ledsetnormalparstuff
2122 \scriptsize
2123 \notenumfont\printlines#1/%
2124 \ifledplinenum
2125 \enspace
2126 \else
2127 {\hskip 0em plus 0em minus .3em}%
2128 \fi
2129 \select@lemmafont#1/#2\rbracket\enskip
2130 \notetextfont
2131 #3\penalty-10\hskip 1em plus 4em minus.4em\relax }
2132
2133 \newcommand{\Cparafootfmt}[3]{%
2134 \ledsetnormalparstuff
2135 \scriptsize
2136 \notenumfont\printlines#1\enspace
2137 % \lemmafont#1/#2\enskip
2138 \notetextfont
2139 #3\penalty-10\hskip 1em plus 4em minus.4em\relax}
2140
2141 \makeatother
2142
2143 \footparagraph{A}
2144 \footparagraph{B}
2145 \footparagraph{C}

```



```

2146
2147 \let\Afootfmt=\Aparafootfmt
2148 \let\Bfootfmt=\Bparafootfmt
2149 \let\Cfootfmt=\Cparafootfmt
2150
2151 \renewcommand*{\Rlineflag}{}
2152
2153 \emergencystretch40pt
2154
2155 \author{Guillelmus de Berchen}
2156 \title{Chronicon Geldriae}
2157 \date{}
2158 \hyphenation{archi-epi-sco-po Huns-dis-brug li-be-ra No-vi-ma-gen-si}
2159 \begin{document}
2160 \begin{pages}
2161 \begin{Leftside}
2162 \beginnumbering\pstart
2163 \selectlanguage{latin}
2164 \section{De ecclesia S. Stephani Novimagensi}
2165
2166 \noindent\setline{1}
2167 Nobilis itaque comes Otto\protect\edindex{Otto II of Guelders}
2168 imperio et dominio Novimagensi sibi, ut praefertur, impignoratis
2169 et commissis
2170 \edtext{proinde}{\Bfootnote{primum D}} praeesse cupiens, anno
2171 \textsc{liiii} superius descripto, mense
2172 \textit{Iu}\edtext{}{\Afootnote{p.\ 227~R}}nio, una cum iudice, scabinis ceterisque
2173 civibus civitatis Novimagensis, pro ipsius et inhabitantium in ea
2174 necessitate,\edtext{}{\Afootnote{p.\ 97~N}} commodo et utilitate,
2175 ut \edtext{ecclesia eius}{\Bfootnote{ecclesia D: eius eius H}} parochialis
2176 \edtext{\abb{extra civitatem}}{\Bfootnote{\textit{om.}~H}} sita
2177 destrueretur et \edtext{infra}{\Bfootnote{intra D}} muros
2178 \edtext{transfer}\edtext{}{\Afootnote{p.\ 129~D}}retur}%
2179 {\Bfootnote{transferreretur NH}}
2180 ac de novo construeretur,
2181 \edtext{a reverendo patre domino
2182 Conrado\protect\edindex{Conrad of Hochstaden} de
2183 \edtext{Hofsteden}{\Bfootnote{Hoffstede D: Hoffsteden H}}, archiepiscopo
2184 \edtext{Coloniensi}{\Bfootnote{Colononiensi H}}, licentiam}%
2185 {\Cfootnote{William is confusing two charters that are five years
2186 apart. Permission from St.\ Apostles' Church in Cologne had been
2187 obtained as early as 1249. Cf.\
2188 Sloet\protect\index{Sloet van de Beele, L.A.J.W.},
2189 \textit{Oorkondenboek} nr.\ 707 (14 November 1249):
2190 ‘‘\ldots}nos devotionis tue precibus annuentes, ut ipsam ecclesiam
2191 faciens demoliri transferas in locum alium competentem, tibi
2192 auctoritate presentium indulgemus\ldots’’}}, et a venerabilibus
2193 \edtext{dominis}{\Bfootnote{viris H}} decano et capitulo sanctorum
2194 Apostolorum\protect\edindex{St. Apostles' (Cologne)}
2195 \edtext{Coloniensi}{\Bfootnote{Coloniae H}}, ipsius ecclesiae ab

```

```

2196 antiquo veris et pacificis patronis, consensum, citra tamen
2197 praeiudicium, damnum aut gravamen \edtext{iurium}{\Bfootnote{virium D}}
2198 et bonorum eorundem, impetravit.
2199 \pend
2200
2201 \pstart
2202 \edtext{Et exinde \edtext{liberum}{\Bfootnote{librum H}}}
2203 locum eiusdem civitatis
2204 \edtext{qui}{\Bfootnote{quae D}} dicitur
2205 \edtext{Hundisburg}{\Bfootnote{Hundisburch D: Hundisbrug HMN:
2206 Hunsdisbrug R}}\protect\edindex{Hundisburg},
2207 de praelibati Wilhelmi\protect\edindex{William II of Holland} Romanorum
2208 \edtext{regis}{\Bfootnote{imperatoris D}}, ipsius fundi
2209 do\edtext{}{\Afootnote{f.\ 72v~M}}mini, consensu, ad aedificandum
2210 \edtext{\abb{et consecrandum}}{\Bfootnote{\textit{om.}\ H}}
2211 ecclesi\edtext{}{\Afootnote{p.\ 228~R}}am et coemeterium,
2212 \edtext{eisdem}{\Bfootnote{eiusdem D}} decano et capitulo de expresso
2213 eiusdem civitatis assensu libera contradiderunt voluntate, obligantes
2214 se ipsi \edtext{comes}{\Bfootnote{comites D}} et civitas
2215 \edtext{\abb{dictis}}{\Bfootnote{\textit{om.}\ H}} decano et capitulo,
2216 quod in recompensationem illius areae infra castrum et portam, quae
2217 fuit dos ecclesiae, in qua plebanus habitare solebat---quae
2218 \edtext{tunc}{\Bfootnote{nunc H}} per novum fossatum civitatis est
2219 destructa---aliam aream competentem et ecclesiae novae,
2220 \edtext{ut praefertur, aedificandae}{%
2221 \lemma{\abb{ut\ldots aedificandae}}{\Bfootnote{\textit{om.}\ H}} satis
2222 \edtext{contiguam}{\Bfootnote{contiguum M}}, ipsi plebano darent et
2223 assignarent.}{\Cfootnote{Cf.\ Sloet, \textit{Oorkondenboek} nr.\ 762
2224 (June 1254)}} Et desuper
2225 \edtext{\abb{apud}}{\Bfootnote{\textit{om.}\ H}} dictam ecclesiam
2226 sanctorum Apostolorum \edtext{est}{\Bfootnote{et H}}
2227 \edtext{littera}{\Bfootnote{litteram H}} sigillis ipsorum
2228 Ottonis\edtext{}{\Afootnote{p.\ 130~D}} comitis et civitatis
2229 \edtext{Novimagensis}{\Bfootnote{Novimagii D}}
2230 \edtext{sigillata}{\Bfootnote{sigillis communita H}}.
2231 \pend
2232
2233 \pstart
2234 // One additional line to show synchronization. //
2235 \pend
2236 \endnumbering
2237 \end{Leftside}
2238
2239 \begin{Rightside}
2240 \sidenotemargin{right}\selectlanguage{english}
2241 \beginnumbering
2242 \pstart
2243 \addtocounter{section}{-1}%
2244 \leavevmode\section{St.\ Stephen's Church in Nijmegen}
2245

```

2246 \noindent\setline{1}%
 2247 After the noble count Otto had taken in pledge the power over
 2248 Nijmegen,\footnote{In 1247 William II\protect\index{William II of Holland}
 2249 (1227--1256) count of Holland needed money to fight his way to
 2250 Aachen\protect\index{Aachen} to be crowned King of the Holy Roman
 2251 Empire. He gave the town of Nijmegen in pledge to Otto
 2252 II\protect\index{Otto II of Guelders} (1229--1271) count of Guelders.}
 2253 like I have written above, he wanted to protect the town. So in June
 2254 1254\ledsidenote{1254} he and the judge, the sheriffs and other
 2255 citizens of Nijmegen obtained permission to demolish the parish
 2256 church that lay outside the town walls,\footnote{Since the early
 2257 seventh century old St.\ Stephen's church had been located close
 2258 to the castle, at today's
 2259 Kelfkensbos\protect\index{Kelfkensbos (Nijmegen)}} square.
 2260 Traces of the church and the presbytery were found during excavations
 2261 in 1998--1999.} to move it inside the walls and to rebuild it new.
 2262 This operation was necessary and useful both for Otto himself and
 2263 for the inhabitants of the town. The reverend father Conrad of
 2264 Hochstaden, archbishop of
 2265 Cologne,\footnote{Conrad of Hochstaden ({\textdag} 1261) was
 2266 archbishop of Cologne in 1238--1261. Nijmegen belonged to the
 2267 archdiocese of Cologne until 1559.} gave his permission. So did the
 2268 reverend dean and canons of the chapter of St.\
 2269 Apostles'\protect\index{St. Apostles' (Cologne)} in Cologne, who had
 2270 long\footnote{They probably became the patrons when the chapter was
 2271 established in the early eleventh century. About the church and the
 2272 chapter, see Gottfried Stracke\protect\index{Stracke, G.},
 2273 \textit{K}\{"o}ln: \ St.\ Aposteln}, Stadtspuren -- Denkm\{"a}ler in
 2274 K\{"o}ln, vol.\ 19, K\{"o}ln: J.\,P.\ Bachem, 1992.} been the true
 2275 and benevolent patrons of the church---but they did not allow Otto
 2276 to do anything without their knowledge, nor to infringe their rights,
 2277 nor to damage their property.
 2278 \pend
 2279
 2280 \pstart
 2281 And so the count and the town voluntarily gave an open space in town
 2282 called Hundisburg, which was owned by the aforementioned king William,
 2283 to the dean and chapter of St.\ Apostles' in order to build and
 2284 consecrate a church and graveyard. King William approved and the
 2285 town of Nijmegen explicitly expressed its assent. A new ditch was dug
 2286 on property of the church near the castle and the
 2287 harbour,\footnote{Nowadays, the exact location of the medieval
 2288 ditch---and of two Roman ones---can be seen in the pavement of
 2289 Kelfkensbos\protect\index{Kelfkensbos (Nijmegen)}} square.} causing
 2290 the demolition of the presbytery. In compensation, the count and
 2291 citizens committed themselves to giving the parish priest another
 2292 suitable space close enough to the new church that was about to be
 2293 built. A letter about these transactions, with the seals of count
 2294 Otto and the town of Nijmegen, is kept at St.\ Apostles'
 2295 church.\footnote{The original letter is lost. A 15th century

```

2296 transcription of it is kept at the Historisches Archiv der
2297 Stadt K\{"o}ln (HASTK).}
2298 \pend
2299
2300 \pstart
2301 // One additional line to show synchronization. //
2302 \pend
2303 \endnumbering
2304 \end{Rightside}
2305 \Pages
2306 \end{pages}
2307
2308 %%%%%%%%%%%
2309 \printindex
2310 \end{document}
2311 %%%%%%%%%%%
2312
2313 \</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.

```

2314 (*djdpoems)
2315 %% djdpoems.tex  example parallel verses on facing pages
2316 \documentclass{article}
2317 \usepackage{ledmac,ledpar}
2318 \addtolength{\textheight}{-15\baselineskip}
2319
2320 \maxchunks{24} % default value = 10
2321 \setstanzaindents{6,0,1,0,1}
2322
2323 \newcommand{\longdash}{-----}
2324
2325 \footparagraph{A} % for left pages
2326 \footparagraph{B} % for right pages
2327 \firstlinenum{1}
2328 \linenumincrement{1}
2329
2330 \let\oldBfootfmt\Bfootfmt
2331 \renewcommand{\Bfootfmt}[3]{%
2332   \let\printlines\printlinesR
2333   \oldBfootfmt{#1}{#2}{#3}}

```

```

2334
2335 \begin{document}
2336
2337 \newcommand{\interstanza}{\pstart\centering\longdash\skipnumbering\pend}
2338
2339 \begin{pages}
2340 \begin{Leftside}
2341 \def\endstanzaextra{\interstanza}
2342 \beginnumbering
2343
2344 \stanza
2345 Arma gravi numero violentaque bella parabam &
2346 edere, materi\={a} conveniente modis. &
2347 Par erat inferior versus---risisse Cupido &
2348 dicitur atque unum surripuisse pedem. \&
2349
2350 \stanza
2351 ‘Quis tibi, saeve puer, dedit hoc in carmina iuris? &
2352 Pieridum vates, non tua turba \edtext{sumus}{\Afootnote{note lost}}. &
2353 Quid si praeripiat flavae V\{e}nus arma Minervae, &
2354 ventilet accensas flava Minerva faces? \&
2355
2356 \stanza
2357 Quis probet in silvis Cererem regnare iugosis, &
2358 lege pharetratae Virginis arva coli? &
2359 Crinibus insignem quis \edtext{acuta}{\Afootnote{acut\={a} (abl.\ abs.)}}
2360 cuspidе Phoebum &
2361 instruat, Aoniam Marte movente lyram? \&
2362 \endnumbering
2363 \end{Leftside}
2364
2365 \begin{Rightside}
2366 \def\endstanzaextra{\interstanza}
2367 \beginnumbering
2368 \firstlinenum{1}
2369 \linenumincrement{1}
2370 \setstanzaindents{6,0,1,0,1,0}
2371
2372 \stanza
2373 I was preparing to sing of weapons and violent wars, &
2374 in heavy numbers, with the subject matter suited to the verse measure. &
2375 The even lines were as long as the odd ones, but Cupid laughed, &
2376 they said, and he stole away one foot.\footnote{I.e., the even lines,
2377 which were hexameters (with six feet) became pentameters
2378 (with five feet).} \&
2379
2380 \stanza
2381 ‘O cruel boy, who gave you the right over poetry? &
2382 We poets belong to the Pierides,\footnote{Muses} we are not your folk. &
2383 \edlabel{beginparadox}What if Venus should seize away the arms of

```

```

2384 Minerva with the golden hair, &
2385 if Minerva with the golden hair should fan alight the kindled torch
2386 of love? \&
2387
2388 \stanza
2389 Who would approve of Ceres\footnote{Ceres was the Roman goddess of
2390 the harvest.} reigning on the woodland ridges, &
2391 and of land tilled under the law of the Maid with the
2392 quiver\footnote{By '\textit{Virgo}' ('Virgin') Ovid means Diana, the
2393 Roman goddess of the hunt.}? &
2394 Who would provide Phoebus with his beautiful hair with a sharp-pointed
2395 spear, &
2396 while Mars stirs the \edtext{Aonian}{\Bfootnote{Mount Parnassus,
2397 where the Muses live, is located in Aonia.}}
2398 lyre?\edlabel{endparadox}\footnote{Lines
2399 \xlineref{beginparadox}--\xlineref{endparadox} show some paradoxical
2400 situations that would occur if the gods didn't stay with their own
2401 business.} \&
2402 \endnumbering
2403 \end{Rightside}
2404
2405 \Pages
2406 \end{pages}
2407
2408 \begin{pages}
2409 \begin{Leftside}
2410 \def\endstanzaextra{\interstanza}
2411 \beginnumbering
2412
2413 \begin{astanza}
2414 Arma gravi numero violentaque bella parabam &
2415 edere, materi\={a} conveniente modis. &
2416 Par erat inferior versus---risisse Cupido &
2417 dicitur atque unum surripuisse pedem. \&
2418 \end{astanza}
2419
2420 \begin{astanza}
2421 'Quis tibi, saeve puer, dedit hoc in carmina iuris? &
2422 Pieridum vates, non tua turba \edtext{sumus}{\Afootnote{note lost}}. &
2423 Quid si praeripiat flavae V\ufe}nus arma Minervae, &
2424 ventilet accensas flava Minerva faces? \&
2425 \end{astanza}
2426
2427 \begin{astanza}
2428 Quis probet in silvis Cererem regnare iugosis, &
2429 lege pharetratae Virginis arva coli? &
2430 Crinibus insignem quis \edtext{acuta}{\Afootnote{acut\={a} (abl.\ abs.)}}
2431 cuspidem Phoebum &
2432 instruat, Aoniam Marte movente lyram? \&
2433 \end{astanza}

```

```

2434
2435 \endnumbering
2436 \end{Leftside}
2437
2438 \begin{Rightside}
2439 \def\endstanzaextra{\interstanza}
2440 \beginnumbering
2441 \firstlinenum{1}
2442 \linenumincrement{1}
2443 \setstanzaindents{6,0,1,0,1,0}
2444
2445 \begin{astanza}
2446 I was preparing to sing of weapons and violent wars, &
2447 in heavy numbers, with the subject matter suited to the verse measure. &
2448 The even lines were as long as the odd ones, but Cupid laughed, &
2449 they said, and he stole away one foot.\footnote{I.e., the even lines,
2450 which were hexameters (with six feet) became pentameters
2451 (with five feet).} \&
2452 \end{astanza}
2453
2454 \begin{astanza}
2455 ‘‘O cruel boy, who gave you the right over poetry? &
2456 We poets belong to the Pierides,\footnote{Muses} we are not your folk. &
2457 \edlabel{beginparadox}What if Venus should seize away the arms of
2458 Minerva with the golden hair, &
2459 if Minerva with the golden hair should fan alight the kindled torch
2460 of love? \&
2461 \end{astanza}
2462
2463 \begin{astanza}
2464 Who would approve of Ceres\footnote{Ceres was the Roman goddess of the
2465 harvest.} reigning on the woodland ridges, &
2466 and of land tilled under the law of the Maid with the
2467 quiver\footnote{By ‘\textit{Virgo}’ (‘Virgin’) Ovid means Diana,
2468 the Roman goddess of the hunt.}? &
2469 Who would provide Phoebus with his beautiful hair with a sharp-pointed
2470 spear, &
2471 while Mars stirs the \edtext{Aonian}{\Bfootnote{Mount Parnassus, where
2472 the Muses live, is located in Aonia.}}
2473 lyre?\edlabel{endparadox}\footnote{Lines
2474 \xlineref{beginparadox}--\xlineref{endparadox} show some paradoxical
2475 situations that would occur if the gods didn’t stay with their
2476 own business.} \&
2477 \end{astanza}
2478
2479 \endnumbering
2480 \end{Rightside}
2481
2482 \Pages
2483 \end{pages}

```

```
2484
2485 \end{document}
2486
2487 </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> . . . 1419, 1420, 1424, 1441, 1455, 1951, 1966, 2003, 2026, 2348, 2354, 2361, 2378, 2386, 2401, 2417, 2424, 2432, 2451, 2460, 2476	<code>\@donetotallinesR</code> 828, 889, 892, 1702, 1893, 1903, 1905
<code>\@M</code> 1430	<code>\@insertR</code> 1102–1104, 1117–1119
<code>\@MM</code> 2077	<code>\@l</code> 247, 560
<code>\@adv</code> <u>324</u> , 591, 592	<code>\@l@dttempcnta</code> 397, 399, 401, 402, 406, 408, 410, 411, 922, 961, 962, 964, 966, 969, 970, 987–991, 993, 1000, 1005, 1009, 1017, 1022, 1026, 1058, 1061, 1063, 1067
<code>\@afterindentfalse</code> 702	<code>\@l@dttempcntb</code> 140, 142, 144, 952, 953, 1000, 1005, 1009, 1017, 1022, 1026, 1050, 1054, 1067, 1075–1077, 1079, 1324, 1326, 1328, 1381–1383, 1385, 1483–1487, 1491–1494, 1498–1501
<code>\@arabic</code> 181, 182	<code>\@l@reg</code> 296
<code>\@astanza@line</code> 1440, 1446, <u>1449</u>	<code>\@l@regR</code> <u>247</u>
<code>\@auxout</code> 1286, 1296, 1529	<code>\@lab</code> 516, 1280, 1289, <u>1310</u>
<code>\@chapter</code> 703	<code>\@lockR</code> 56, 269, 271, 273, 286, 431, 447, 448, 450, 451, 479, 480, 482, 905, 928, 930, 931, 933, 1014, 1031, 1033, 1035
<code>\@cs@linesinparL</code> 1685, <u>1784</u>	<code>\@lopL</code> <u>538</u> , 1811
<code>\@cs@linesinparR</code> 1685, <u>1784</u>	<code>\@lopR</code> <u>538</u> , 1814
<code>\@cs@linesonpageL</code> . . 1693, 1743, <u>1797</u>	
<code>\@cs@linesonpageR</code> . . 1693, 1743, <u>1797</u>	
<code>\@donereallinesL</code> 828, 855, 1649, 1651, 1699	
<code>\@donereallinesR</code> 828, 888, 1654, 1656, 1701	
<code>\@donetotallinesL</code> 828, 856, 859, 1700, 1873, 1883, 1885	

- \@nameuse 1409, 1413, 2081
 - \@nobreakfalse 740, 769
 - \@nobreaktrue 738, 742, 767, 771
 - \@oldnobreak 738, 740, 767, 769, 805, 818
 - \@pend 531, 1649
 - \@pendR 531, 1654
 - \@pstartsfalse 1624
 - \@pstartstrue 1624
 - \@ref 503, 564, 568
 - \@ref@reg 529
 - \@schapter 703
 - \@set 356, 598, 599
 - \@tag 629, 646, 1126, 1130, 1140, 1144,
1154, 1158, 1168, 1172, 1182,
1186, 1197, 1201, 1211, 1215,
1225, 1229, 1239, 1243, 1253, 1257
 - \@temp 1520
 - \@templ@d 1372, 1373
 - \@writelinesinparL .. 1606, 1647, 1877
 - \@writelinesinparR .. 1607, 1647, 1897
 - \@writelinesonpageL . 1717, 1719, 1810
 - \@writelinesonpageR . 1733, 1735, 1810
 - \@xloop 1115
 - \add@penaltiesR 887, 1111
 - \addtocontents 1530–1532
 - \addtocounter 2243
 - \addtolength ... 1914, 2066–2068, 2318
 - \advanceline 590, 621
 - \affixline@num 846
 - \affixline@numR 879, 984
 - \affixside@note 849
 - \affixside@noteR 882, 1371
 - \Afootfmt 2147
 - \Afootins 2066
 - \Afootnote 1122, 1996,
2001, 2172, 2174, 2178, 2209,
2211, 2228, 2352, 2359, 2422, 2430
 - \Afootnoterule 2062
 - \Aparafootfmt 2112, 2147
 - \apptocmd 1560
 - \araw@textfalse 1635
 - \araw@texttrue 1635
 - astanza (environment) 7, 1422
 - \AtBeginDocument ... 1306, 1505, 1540
 - \author 2155
- B**
- \ballast@count 912, 917
 - \bbl@main@language 1556, 1557
 - \bbl@set@language 1547, 1548
 - \beginnumbering 6, 32, 719,
745, 1945, 1960, 1990, 2011,
2162, 2241, 2342, 2367, 2411, 2440
 - \beginnumberingR 45, 98, 719, 774
 - \Bfootfmt 1919, 1920, 2148, 2330, 2331
 - \Bfootins 2067
 - \Bfootnote 1136, 2014, 2018,
2170, 2175–2177, 2179, 2183,
2184, 2193, 2195, 2197, 2202,
2204, 2205, 2208, 2210, 2212,
2214, 2215, 2218, 2221, 2222,
2225–2227, 2229, 2230, 2396, 2471
 - \Bfootnoterule 2063
 - \box 2095
 - \Bparafootfmt 2120, 2148
 - \bypage@Rfalse 118, 130
 - \bypage@Rtrue 118, 126
- C**
- \c@ballast 917
 - \c@firstlinenumR 149, 1056
 - \c@firstsublinenumR 153, 1051
 - \c@linenumincrementR 149, 1056
- A**
- \abb 2105,
2109, 2176, 2210, 2215, 2221, 2225
 - \absline@num 390, 404, 423
 - \absline@numR ... 54, 198, 249, 252,
255, 387, 395, 416, 435, 469,
497, 508, 897, 914, 915, 952, 1101
 - \actionlines@list
..... 239, 242, 390, 404, 423
 - \actionlines@listR
..... 202, 217, 231, 234, 387,
395, 416, 435, 469, 497, 974, 977
 - \actions@list . 243, 391, 411, 425, 427
 - \actions@listR
..... 202, 218, 235, 388, 402, 418,
420, 437, 446, 471, 478, 498, 978
 - \add@inserts 848
 - \add@inserts@nextR 1090
 - \add@insertsR 881, 1090
 - \add@penaltiesL 854, 1111

- `\c@page` . . . 560, 1762, 1768, 1771, 1778
`\c@sublinenumincrementR` . . 153, 1051
`\centering` 2337
`\Cfootfmt` 2149
`\Cfootins` 2068
`\Cfootnote` 1136, 2185, 2223
`\Cfootnoterule` 2064
`\ch@ck@l@ckR` 984
`\ch@cksub@l@ckR` 984
`\ch@cksub@lockR` 1052
`\chapter` 689, 690, 698
`\chapterinpages` 682, 690, 700
`\chardef` 1419
`\check@goal` 1836, 1850, 1862
`\check@pstarts`
 1583, 1608, 1624, 1679, 1687, 1695
`\checkpageL` 1706, 1714, 1830
`\checkpageR` 1723, 1730, 1830
`\checkraw@text`
 1587, 1604, 1635, 1703, 1739
`\cleardoublepage` 1774
`\clearl@dleftpage` 1722, 1760
`\clearl@drighpage` 1738, 1760
`\cleartoevenpage` 1760
`\cleartol@devenpage` 1668, 1760
`\closeout` 551, 555
`\columnrulewidth` 4, 1619, 1936
`\Columns` 3, 1571, 1970, 2030
`\columnseparator` 4, 1601, 1619
`\countLline` 823, 834
`\countRline` 823, 867
`\Cparafootfmt` 2133, 2149
`\critext` 626
- D**
- `\date` 2157
Dekker, Dirk-Jan 76, 82
`\Dfootnote` 1136
`\dimen` 577, 578, 582–584, 588
`\divide` 989
`\do@actions@fixedcodeR` 921
`\do@actions@nextR` 921
`\do@actionsR` 898, 921
`\do@ballastR` 899, 912
`\do@lineL` 833, 1591, 1595, 1710
`\do@lineLhook` 838, 863
`\do@lineR` 866, 1592, 1597, 1726
`\do@lineRhook` 863, 871
`\do@lockoff` 466
`\do@lockoffL` 490
- `\do@lockoffR` 466
`\do@lockon` 431
`\do@lockonL` 463
`\do@lockonR` 431
`\documentclass` 1913, 2049, 2316
`\dp` 2078, 2093
`\dummy@ref` 512
- E**
- `\edfont@info` 665, 668, 674, 677
`\edindex` 2167, 2182, 2194, 2206, 2207
`\edlabel` 1278, 2383, 2398, 2457, 2473
`\edtext` 643, 1995,
 2001, 2013, 2017, 2170, 2172,
 2174–2178, 2181, 2183, 2184,
 2193, 2195, 2197, 2202, 2204,
 2205, 2208–2212, 2214, 2215,
 2218, 2220, 2222, 2225–2230,
 2352, 2359, 2396, 2422, 2430, 2471
`\Efootnote` 1136
`\emergencystretch` 2153
`\empty` 72, 75, 231, 239, 637, 654,
 663, 672, 753, 782, 974, 1055,
 1063, 1092–1094, 1105, 1116,
 1281, 1290, 1785, 1791, 1798, 1804
`\end@lemmas` 637, 638, 654, 655
`\endashchar` 1268
`\endgraf` 801, 814, 1577, 1671
`\endline@num` 519, 525
`\endlock` 610, 1428, 1437, 1442
`\endnumbering` 6, 35, 65, 102,
 720, 1952, 1967, 2004, 2027,
 2236, 2303, 2362, 2402, 2435, 2479
`\endnumberingR` 48, 65, 87, 97, 110, 720
`\endpage@num` 518, 525
`\endstanzaextra`
 1444, 2341, 2366, 2410, 2439
`\endsub` 577
`\endsubline@num` 520, 526
`\enskip` 2116, 2129, 2137
`\enspace` 2115, 2125, 2136
environments:
 `astanza` 7, 1422
 `Leftside` 5, 705
 `pages` 4, 682
 `pairs` 3, 682
 `Rightside` 5, 717
`\extensionchars` 43, 62, 93, 107, 115

F

`\f@x@l@cksR` 984
`\first@linenum@out@Rfalse` .. 546, 552
`\first@linenum@out@Rtrue` 546
`\firstlinenum` 5, 158, 1942,
 1957, 1987, 2008, 2327, 2368, 2441
`\firstsublinenum` 5, 158
`\fix@page` 292, 299
`\flag@end` 561, 642, 659
`\flag@start` 561, 634, 651
`\floatingpenalty` 2077
`\flush@notes` 1610, 1748
`\flush@notesR` 1114, 1611, 1749
`\footnote`
 . 2248, 2256, 2265, 2270, 2287,
 2295, 2376, 2382, 2389, 2392,
 2398, 2449, 2456, 2464, 2467, 2473
`\footnotesize` 2059, 2060
`\footparagraph` . 2143–2145, 2325, 2326
`\fullstop` . 194, 1265, 1267, 1269, 1271

G

`\get@linelistfile` 227
`\get@nextboxL` 1713, 1870
`\get@nextboxR` 1729, 1870
`\getline@num` 843
`\getline@numR` 876, 896
`\getlinesfrompagelistL`
 1691, 1741, 1797
`\getlinesfrompagelistR`
 1692, 1742, 1797
`\getlinesfromparlistL` ... 1683, 1784
`\getlinesfromparlistR` ... 1684, 1784
`\gl@p` 234, 235, 242, 243, 638, 655, 667,
 676, 977, 978, 1098, 1102, 1117,
 1284, 1293, 1788, 1794, 1801, 1807
`\goalfraction` 4, 1862

H

`\hb@xt@` 845, 851,
 858, 878, 884, 891, 1599, 1712, 1728
`\hsize` . 763, 792, 1599, 1712, 1728, 2090
`\hyphenation` 2158

I

`\if@filesW` 1528
`\if@firstcolumn` 1069, 1375
`\if@nobreak` 737, 766
`\if@pstarts` 1584, 1624, 1680, 1696
`\ifaraw@text` ... 1589, 1635, 1705, 1740

`\ifbypage@` 315
`\ifbypage@R` 118, 305, 956
`\ifdim` 578, 582, 584, 588, 1764, 1837, 1851
`\iffirst@linenum@out@R` 546, 550
`\ifl@d@dash` 1268
`\ifl@d@elin` 1270, 1271
`\ifl@d@esl` 1271
`\ifl@d@pnum` 1265, 1269
`\ifl@d@ssub` 1267
`\ifl@dpagfull` 1716, 1732, 1830
`\ifl@dpaging` 5
`\ifl@dpairing` 5, 69
`\ifl@dsamelang` 1516, 1590
`\ifl@dsamepage` 1709, 1725, 1830
`\ifl@dskipnumber` 1046
`\ifl@dusedbabel` 1514
`\ifledplinenum` 1266, 2124
`\ifledRcol` 5, 141, 163,
 167, 171, 175, 214, 229, 293,
 302, 326, 340, 357, 374, 386,
 394, 415, 460, 487, 496, 505,
 562, 572, 579, 585, 591, 598,
 606, 611, 615, 620, 631, 648,
 662, 1124, 1138, 1152, 1166,
 1180, 1195, 1209, 1223, 1237,
 1251, 1279, 1311, 1325, 1336,
 1348, 1360, 1395, 1408, 1551, 1561
`\ifnoteschanged@` 79
`\ifnumberedpar@` ... 747, 776, 797,
 810, 1123, 1137, 1151, 1165,
 1179, 1194, 1208, 1222, 1236,
 1250, 1335, 1347, 1359, 1394, 1407
`\ifnumbering` 33, 743, 794
`\ifnumberingR`
 27, 46, 66, 89, 121, 772, 807
`\ifodd` 1079, 1385, 1762, 1768, 1771, 1778
`\ifpst@rtedL` 28, 751
`\ifpst@rtedR` 28, 780
`\ifsublines@` 192, 281, 325, 358, 365,
 396, 405, 417, 424, 436, 470,
 524, 526, 900, 963, 1049, 1313, 1317
`\ifvbox` 835, 868, 1639, 1642, 1871, 1891
`\ifwrittenlinesL` 1867, 1875
`\ifwrittenlinesR` 1868, 1895
`\initnumbering@reg` 41
`\insert` 2073
`\insert@count` 502, 568, 632,
 649, 1131, 1145, 1159, 1173,
 1187, 1202, 1216, 1230, 1244,
 1258, 1343, 1355, 1367, 1402, 1415

- \insert@countR 503, 564, 631,
648, 1127, 1141, 1155, 1169,
1183, 1198, 1212, 1226, 1240,
1254, 1339, 1351, 1363, 1398, 1411
- \insertlines@listR
- 72, 202, 216, 508, 1094, 1098
- \inserts@list
- . . 752, 1130, 1144, 1158, 1172,
1186, 1201, 1215, 1229, 1243,
1257, 1342, 1354, 1366, 1401, 1414
- \inserts@listR
- . . 781, 1089, 1092, 1102, 1116,
1117, 1126, 1140, 1154, 1168,
1182, 1197, 1211, 1225, 1239,
1253, 1338, 1350, 1362, 1397, 1410
- \interfootnotelinepenalty 2076
- \interlinepenalty 1430, 2076
- \interstanza
- 2337, 2341, 2366, 2410, 2439
- L**
- \l@d@nums 665, 668, 674,
677, 1126, 1130, 1140, 1144,
1154, 1158, 1168, 1172, 1182,
1186, 1197, 1201, 1211, 1215,
1225, 1229, 1239, 1243, 1253, 1257
- \l@d@set 373, 606, 607
- \l@d@bbl@set@language 1525, 1548
- \l@d@bfnote 1393
- \l@d@maxchunks 758, 760,
787, 789, 1473, 1483, 1491, 1498
- \l@d@calc@maxoftwo
- 1685, 1817, 1882, 1902
- \l@d@calc@minoftwo . . 1693, 1743, 1817
- \l@d@calcnun 984
- \l@d@checklang 1518, 1588
- \l@d@chset@num 248, 251, 373
- \l@d@csnote 1334
- \l@d@csnotetext
- 1373, 1376, 1378, 1386, 1388
- \l@d@emptyd@ta 839, 872
- \l@d@dend@stuff 44, 63, 94, 108, 116
- \l@d@getline@margin 139
- \l@d@getsidenote@margin 1323
- \l@d@dld@ta 847, 880, 1070, 1082
- \l@d@leftbox . . 820, 844, 858, 1600, 1712
- \l@d@linenumR 184
- \l@d@l@sn@te 850, 883
- \l@d@l@snote 1334
- \l@d@make@labels 1297
- \l@d@make@labelsR 1287, 1300
- \l@d@minpagelines
- 1658, 1694, 1744, 1838, 1852
- \l@d@numpstartsL . 37, 757, 758, 760,
762, 1477, 1509, 1572, 1573,
1615, 1627, 1665, 1666, 1753, 1880
- \l@d@numpstartsR . 50, 786, 787, 789,
791, 1477, 1510, 1572, 1573,
1616, 1630, 1665, 1666, 1754, 1900
- \l@d@doldbbl@set@language 1547
- \l@d@doldselectlanguage 1546, 1550, 1555
- \l@d@pagefullfalse 1830
- \l@d@pagefulltrue 1830
- \l@d@pagingfalse 7, 684, 697
- \l@d@pagingtrue 692
- \l@d@pairingfalse 5, 686, 696
- \l@d@pairingtrue 683, 691
- \l@d@parsedendline 2086
- \l@d@parsedstartline . 2081, 2086, 2088
- \l@d@parsedstartpage 2084, 2089
- \l@d@parsefootspec 2080
- \l@d@pscL 835, 840, 1479, 1511, 1581,
1585, 1613, 1627, 1639, 1675,
1681, 1686, 1689, 1697, 1751,
1871, 1873, 1880, 1882, 1884, 1886
- \l@d@pscR 868, 873, 1480, 1512,
1582, 1586, 1614, 1630, 1642,
1676, 1682, 1690, 1698, 1752,
1891, 1893, 1900, 1902, 1904, 1906
- \l@d@rd@ta 851, 884, 1072, 1080
- \l@d@rightbox . 820, 877, 891, 1602, 1728
- \l@d@rsn@te 852, 885
- \l@d@rsnote 1334
- \l@d@samelangfalse 1516, 1519
- \l@d@samelangtrue 1516, 1522
- \l@d@samepagefalse 1830
- \l@d@samepagetrue 1830
- \l@d@setupmaxlinecounts . . 1490, 1507
- \l@d@setuprawboxes 1482, 1506
- \l@d@skipnumberfalse 1047
- \l@d@skipnumbertrue 944
- \l@d@unhbox@line 851, 884
- \l@d@usedbabelfalse 1514, 1543
- \l@d@usedbabeltrue 1514, 1545
- \l@d@uselanguage
- 1535, 1594, 1596, 1707, 1724
- \l@d@zeromaxlinecounts . . 1490, 1508
- \l@d@zeropenalties 800, 813, 1576, 1670
- \l@pscL 1479
- \l@pscR 1479

- \label@refs 1282, 1284, 1287, 1291, 1293, 1297
- \labelref@list 1290, 1293, 1318
- \labelref@listR 1276, 1281, 1284, 1314
- \language name .. 1526, 1527, 1529–1532
- \last@page@num 313, 319
- \last@page@numR 299
- \lastbox 842, 875
- \lastskip 577, 583
- \Lcolwidth 3, 4, 10, 693, 763, 845, 858, 1937, 1938, 1981, 1982
- \ldots 2000, 2003, 2023, 2026, 2190, 2192, 2221
- \led@err@BadLeftRightPstarts ... 18, 1573, 1666
- \led@err@LeftOnRightPage ... 21, 1772
- \led@err@LineationInNumbered ... 122
- \led@err@NumberingNotStarted ... 83
- \led@err@NumberingShouldHaveStarted 96
- \led@err@NumberingStarted 34, 47
- \led@err@PendNoPstart 798, 811
- \led@err@PendNotNumbered ... 795, 808
- \led@err@PstartInPstart ... 748, 777
- \led@err@PstartNotNumbered . 744, 773
- \led@err@RightOnLeftPage ... 21, 1779
- \led@err@TooManyPstarts 15, 759, 788
- \led@mess@NotesChanged 80
- \led@mess@SectionContinued 92, 106, 114
- \led@warn@BadAction 946
- \led@warn@BadAdvancelineLine 343, 349
- \led@warn@BadAdvancelineSubline . 329, 335
- \led@warn@BadLineation 132
- \led@warn@BadSetline 596
- \led@warn@BadSetlinenum 604
- \led@warn@DuplicateLabel 1302
- \ledllfill 851, 884
- \ledmac@error 16, 19, 22, 24
- \ledplinenumfalse 2082
- \ledplinenumtrue ... 2080, 2085, 2087
- \ledRcolfalse 9, 706, 729
- \ledRcoltrue 718
- \ledrlfill 851, 884
- \ledsavedprintlines 7, 1263
- \ledsetnormalparstuff 2113, 2121, 2134
- \ledsidenote 2254
- \ledstrutL 1712, 1757
- \ledstrutR 1728, 1757
- \ledthegoal 1837, 1851, 1862
- \leftlinenumR 184, 1070, 1082
- Leftside (environment) 5, 705
- \Leftsidehook 710, 712
- \Leftsidehookend 711, 712
- \lemma 1995, 2221
- \lemmafnt 2116, 2137
- \line@list 672, 676
- \line@list@stuff 43, 107
- \line@list@stuffR ... 62, 93, 115, 548
- \line@listR . 75, 202, 215, 526, 663, 667
- \line@margin 144
- \line@marginR 137, 1075
- \line@num 316, 347, 348, 350, 368, 379, 380, 408, 1316
- \line@numR . 55, 191, 198, 253, 287, 306, 341, 342, 344, 361, 375, 376, 399, 519, 523, 906, 957, 966, 1054, 1056, 1058, 1059, 1312
- \lineation 726, 2053
- \lineationR 120, 726
- \linenum@out 567, 575, 580, 586, 592, 599, 607, 612, 616, 1289, 1649, 1811
- \linenum@outR 545, 551, 553, 555, 556, 560, 563, 573, 579, 585, 591, 598, 606, 611, 615, 620, 1280, 1654, 1814
- \linenumberlist 1055, 1059
- \linenumincrement .. 5, 158, 1943, 1958, 1988, 2009, 2328, 2369, 2442
- \linenummargin 137, 1944, 1959, 1989, 2010, 2054
- \linenumr@p 1266, 1270, 1312, 1316
- \linenumrepR 181, 191
- \linenumsep 186, 188
- \linesinpar@listL 207, 223, 533, 1785, 1788
- \linesinpar@listR 207, 219, 536, 1791, 1794
- \linesonpage@listL 224, 540, 1798, 1801
- \linesonpage@listR 220, 543, 1804, 1807
- \list@clear 215–220, 223, 224, 226, 752, 781
- \list@clearing@reg 222
- \list@create 202–205, 207–209, 1089, 1276
- \lock@disp 1016, 1020, 1025
- \lock@off 457, 458, 466, 615, 616
- \lock@on 611, 612
- \longdash 2323, 2337

M

`\maxchunks` 3, 1473, 2320
`\maxdimen` 2090
`\maxlinesinpar@list` 207, 226
`\memorydump` 6, 709, 723
`\memorydumpL` 101, 709
`\memorydumpR` 101, 723
`\message` 42, 61
`\morenoexpands` 2108
`\mpAfootnote` 1193
`\mpBfootnote` 1193
`\mpCfootnote` 1193
`\mpDfootnote` 1193
`\mpEfootnote` 1193
`\mpvAfootnote` 1196, 1200, 1205
`\mpvBfootnote` 1210, 1214, 1219
`\mpvCfootnote` 1224, 1228, 1233
`\mpvDfootnote` 1238, 1242, 1247
`\mpvEfootnote` 1252, 1256, 1261
`\multiply` 990

N

`\n@num` 494, 620
`\n@num@reg` 500
`\namebox` 835, 840, 868,
 873, 1457, 1639, 1642, 1871, 1891
`\NeedsTeXFormat` 2
`\new@line` 851
`\new@lineR` 559, 884
`\newbox` 734, 820, 821, 1458
`\newcounter` 149, 151, 153, 155
`\newif` 6,
 8, 27, 29, 118, 546, 1514, 1516,
 1624, 1635, 1830, 1832, 1867, 1868
`\newnamebox` 1457, 1485, 1486
`\newnamecount` 1468, 1493
`\newwrite` 545
`\next@action` 243
`\next@actionline` 240, 242
`\next@actionlineR`
 232, 234, 915, 953, 975, 977
`\next@actionR` 235,
 916, 954, 955, 960, 961, 969, 978
`\next@insert` 753
`\next@insertR`
 782, 1093, 1096, 1098, 1101, 1105
`\next@page@num` 320, 391
`\next@page@numR` 59, 256, 258, 310, 388
`\no@expands` 628, 645
`\nobrak` 2106, 2107

`\noindent` 2091, 2166, 2246
`\normal@pars` 68, 756, 785
`\normalbfnoteX` 1406
`\notefontsetup` 2075
`\notenumfont` ... 2059, 2115, 2123, 2136
`\noteschanged@true`
 73, 76, 664, 673, 1095
`\notetextfont` .. 2060, 2117, 2130, 2138
`\num@lines` 801, 1577, 1671
`\num@linesR` 733, 814, 1578, 1672
`\numberedpar@true` 764, 793
`\numberingRfalse` 67
`\numberingRtrue` 52, 87, 111
`\numberingtrue` 39, 103
`\numlabfont` 191
`\numpagelinesL`
 1658, 1711, 1717, 1721, 1838
`\numpagelinesR`
 1658, 1727, 1733, 1737, 1852

O

`\oldBfootfmt` ... 1919, 1922, 2330, 2333
`\oldchapter` 689, 698
`\one@line` 840, 842, 851
`\one@lineR` 733, 873, 875, 884
`\openout` 553, 556

P

`\page@action` 257, 385, 513
`\page@num` 238, 318, 1383
`\page@numR`
 211, 230, 308, 518, 523, 955, 1077
`\pagegoal` 1865
`\Pages` 4, 1662, 2305, 2405, 2482
`pages` (environment) 4, 682
`\pagetotal` 1764, 1837, 1851
`pairs` (environment) 3, 682
`\par@line` 802, 1579, 1673
`\par@lineR` 733, 815, 1580, 1674
`\para@vfootnote` 2072
`\pausenumbering` 721
`\pausenumberingR` 86, 721
`\pend` .. 5, 708, 725, 749, 1443, 2199,
 2231, 2235, 2278, 2298, 2302, 2337
`\pendL` 708, 794
`\pendR` 725, 778, 807
`\prevgraf`
 801, 814, 1577, 1578, 1671, 1672
`\previous@A@number` 2100
`\previous@B@number` 2101

- \previous@C@number 2102
 - \previous@page 2084, 2089, 2103
 - \printindex 2309
 - \printlines
 - 1274, 1921, 2115, 2123, 2136, 2332
 - \printlinesR 7, 1263, 1921, 2332
 - \protected@write ... 1286, 1296, 1529
 - \ProvidesPackage 3
 - \pst@rtedLfalse 28, 38
 - \pst@rtedLtrue 104, 754
 - \pst@rtedRfalse 30, 51, 70
 - \pst@rtedRtrue 90, 112, 783
 - \pstart 5, 16, 20, 707, 724, 1445, 2162,
 - 2201, 2233, 2242, 2280, 2300, 2337
 - \pstartL 707, 736
 - \pstartR 724, 736
- R**
- \rbracket 2106, 2129
 - \Rcolwidth 3, 4,
 - 10, 694, 792, 878, 891, 1938, 1982
 - \read@linelist 213, 549
 - \rem@inder 1059, 1061–1063
 - \resumenumbering 722
 - \resumenumberingR 86, 722
 - \rightlinenumR 184, 1072, 1080
 - Rightside (environment) 5, 717
 - \Rightsidehook 712, 727
 - \Rightsidehookend 712, 730
 - \rlap 1072, 1080
 - \Rlineflag 7, 179, 191,
 - 1266, 1270, 1304, 1917, 1934, 2151
 - \rule 1620
- S**
- \sc@n@list 1060, 1062
 - \secdef 703
 - \section@num 40, 42, 43, 105–107
 - \section@numR
 - ... 25, 53, 61, 62, 91–93, 113–115
 - \select@language ... 1527, 1529–1532
 - \select@lemmafont 2129
 - \selectlanguage 1535, 2163, 2240
 - \set@line 630, 647, 661
 - \set@line@action
 - ... 250, 354, 363, 370, 393, 515
 - \setl@dlp@rbox 1376, 1388
 - \setl@drp@rbox 1378, 1386
 - \setline 594, 2166, 2246
 - \setlinenum 602
 - \setnamebox 762, 791, 1457
 - \setprintlines 1264
 - \setstanzaindents
 - ... 1932, 1979, 2321, 2370, 2443
 - \showlemma 636, 653
 - \sidenote@margin 1328, 1332
 - \sidenote@marginR 1321, 1381
 - \sidenotemargin 1321, 2055, 2240
 - \skip 2066–2068
 - \skip@lockoff 458, 466
 - \skipnumbering 7, 619, 2337
 - \skipnumbering@reg 623
 - \smash 1620
 - \splitmaxdepth 2078
 - \splittopskip 837, 870, 2078
 - \stanza ... 1946, 1961, 1991, 2012,
 - 2344, 2350, 2356, 2372, 2380, 2388
 - \stanza@count 1425, 1439, 1450
 - \stanza@hang 1427, 1452
 - \stanzaindentbase 1450
 - \startlock 610
 - \startstanzahook 1423
 - \startsub 577
 - \sub@action 266, 414, 514
 - \sub@change 60, 260, 261, 267
 - \sub@lockR 57, 275, 277,
 - 279, 282, 432, 438, 439, 441,
 - 442, 472, 473, 475, 901, 936,
 - 938, 939, 941, 997, 1037, 1039, 1041
 - \sub@off 585, 586
 - \sub@on 579, 580
 - \subline@num 193,
 - 316, 333, 334, 336, 366, 406, 1317
 - \subline@numR
 - . 194, 198, 283, 287, 306, 327,
 - 328, 330, 359, 397, 520, 524,
 - 902, 907, 957, 964, 1050, 1051, 1313
 - \sublinenumincrement 5, 158
 - \sublinenumr@p . 1267, 1271, 1313, 1317
 - \sublinenumrepR 181, 194
 - \sublines@false 58, 264, 926
 - \sublines@true 262, 924
 - \sublock@disp 999, 1003, 1008
 - \symplinenum 1266
 - \sza@penalty 1434, 1438
- T**
- \textdagger 2265
 - \textheight 1914, 2318

<code>\textit</code>	1976, 2001, 2003, 2024, 2026, 2036, 2176, 2189, 2210, 2215, 2221, 2223, 2225, 2273, 2392, 2467	<code>\vl@dcnote</code>	1361, 1365
<code>\textnormal</code>	2107	<code>\vl@dlnote</code>	1337, 1341
<code>\textsc</code>	1975, 2171	<code>\vl@drnote</code>	1349, 1353
<code>\textwidth</code>	11, 13, 693, 694, 1937, 1981	<code>\vsplit</code>	840, 873
<code>\theledlanguageL</code>	1520, <u>1535</u> , 1594, 1707	W	
<code>\theledlanguageR</code>	1520, <u>1535</u> , 1596, 1724	<code>\wd</code>	851, 884, 2094
<code>\thepage</code>	560, 1287, 1297	<code>\writtenlinesLfalse</code>	1677, 1881
<code>\thr@@</code>	441, 450, 473, 480, 931, 939	<code>\writtenlinesLtrue</code>	1878
<code>\title</code>	2156	<code>\writtenlinesRfalse</code>	1678, 1901
<code>\topskip</code>	1764	<code>\writtenlinesRtrue</code>	1898
U		X	
<code>\unhbox</code>	1462, 1600, 1602, 1712, 1728	<code>\x@lemma</code>	638–640, 655–657
<code>\unhnamebox</code>	<u>1457</u>	<code>\xlineref</code>	2399, 2474
<code>\unvbox</code>	842, 875, 1464	<code>\xpg@main@language</code>	1565, 1566
<code>\unvnamebox</code>	<u>1457</u>	<code>\xpg@set@language</code>	1560, 1564
<code>\unvxh</code>	2092	<code>\xright@appenditem</code>	387, 388, 390, 391, 395, 402, 404, 411, 416, 418, 420, 423, 425, 427, 435, 437, 446, 469, 471, 478, 497, 498, 508, 522, 533, 536, 540, 543, 1125, 1129, 1139, 1143, 1153, 1157, 1167, 1171, 1181, 1185, 1196, 1200, 1210, 1214, 1224, 1228, 1238, 1242, 1252, 1256, 1312, 1316, 1337, 1341, 1349, 1353, 1361, 1365, 1396, 1400, 1409, 1413
<code>\usenamecount</code>	1426, 1433, <u>1468</u> , 1500, 1686, 1873, 1882, 1884, 1893, 1902, 1904	Z	
<code>\usepackage</code>	1915, 2050–2052, 2317	<code>\z@skip</code>	2079
V		<code>\zz@@@</code>	1282, 1291
<code>\vAfootnote</code>	1125, 1129, 1134		
<code>\vbadness</code>	836, 869		
<code>\vbfnoteX</code>	1409, 1413		
<code>\vBfootnote</code>	1139, 1143, 1148		
<code>\vbox</code>	762, 791, 2090		
<code>\vCfootnote</code>	1153, 1157, 1162		
<code>\vDfootnote</code>	1167, 1171, 1176		
<code>\vEfootnote</code>	1181, 1185, 1190		
<code>\vl@dbfnote</code>	1396, 1400		

Change History

v0.1	and <code>\l@duselanguage</code> to	
General: First public release	<code>\Columns</code>	56
v0.2	<code>\Pages</code> : Added <code>\l@duselanguage</code>	
General: Added section of babel re-	to <code>\Pages</code>	60
lated code		
Fix babel problems	v0.3	
<code>\Columns</code> : Added <code>\l@dchecklang</code>	General: Reorganize for ledarab	1
	<code>\affixline@numR</code> :	Changed

<code>\affixline@numR</code> to match new ledmac	38	Added <code>\ledstrutR</code> to <code>\Pages</code> .	60
<code>\do@actions@nextR</code> : Used <code>\do@actions@fixedcode</code> in <code>\do@actionsR</code>	37	<code>\Rightsidehookend</code> : Added <code>\Leftsidehook</code> , <code>\Leftsidehookend</code> , <code>\Rightsidehook</code> and <code>\Rightsidehookend</code>	31
<code>\do@lineL</code> : Added <code>\do@lineLhook</code> to <code>\do@lineL</code>	35	<code>\sublinenumrepR</code> : Added <code>\linenumrepR</code> and <code>\sublinenumrepR</code>	16
Simplified <code>\do@lineL</code> by using macros for some common code	35	v0.3a General: Minor <code>\linenummargin</code> fix	1
<code>\do@lineR</code> : Changed <code>\do@lineR</code> similarly to <code>\do@lineL</code>	36	<code>\line@marginR</code> : Don't just set <code>\line@marginR</code> in <code>\linenummargin</code>	15
<code>\do@lineRhook</code> : Added <code>\do@lineLhook</code> and <code>\do@lineRhook</code>	36	v0.3b General: Improved parallel page balancing	1
Leftside: Added hooks into Left- side environment	31	<code>\Pages</code> : Added <code>\l@dminpagelines</code> calculation for succeeding page pairs	61
<code>\flag@end</code> : Removed extraneous spaces from <code>\flag@end</code>	27	v0.3c General: Compatibilty with Poly- glossia	1
<code>\ifledRcol</code> : Moved <code>\ifl@dpairing</code> to ledmac	10	v0.4 General: No more ledparpatch. All patches are now in the main file.	1
<code>\ifpst@rtedR</code> : Moved <code>\ifpst@rtedL</code> to ledmac	11	v0.5 General: Corrections about <code>\section</code> and other titles in numbered sections	1
<code>\l@dlinenumR</code> : Simplified <code>\leftlinenumR</code> and <code>\rightlinenumR</code> by introducing <code>\l@dlinenumR</code>	16	v0.6 General: Be able to use <code>\chapter</code> in parallel pages.	1
<code>\l@dnumpstartsR</code> : Moved <code>\l@dnumpstartsL</code> to ledmac .	52		
<code>\ledsavedprintlines</code> : Simpli- fied <code>\printlinesR</code> by using <code>\setprintlines</code>	46		
<code>\ledstrutR</code> : Added <code>\ledtrutL</code> and <code>\ledstrutR</code>	61		
<code>\normalbfnoteX</code> : Removed extraneous spaces from <code>\normalbfnoteX</code>	50		
<code>\Pages</code> : Added <code>\ledstrutL</code> to <code>\Pages</code>	60		