

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

Peter Wilson  
Herries Press<sup>†</sup>  
Maïeul Rouquette<sup>‡</sup>

## Abstract

The **ledmac** package, which is based on the PLAIN T<sub>E</sub>X 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.

To report bugs, please go to **ledmac**'s GitHub page and click "New Issue": <https://github.com/maieul/ledmac/issues/>. You must open an account with github.com to access my page (maieul/ledmac). GitHub accounts are free for open-source users.

## Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>The <b>ledpar</b> package</b>	<b>4</b>
2.1	General . . . . .	4
<b>3</b>	<b>Parallel columns</b>	<b>5</b>
<b>4</b>	<b>Facing pages</b>	<b>5</b>
<b>5</b>	<b>Left and right texts</b>	<b>6</b>
<b>6</b>	<b>Numbering text lines and paragraphs</b>	<b>8</b>
<b>7</b>	<b>Verse</b>	<b>9</b>
<b>8</b>	<b>Implementation overview</b>	<b>12</b>

---

\*This file (**ledpar.dtx**) has version number v0.9.2, last revised 2011/11/08.

<sup>†</sup>herries dot press at earthlink dot net

<sup>‡</sup>maieul at maieul dot net

<b>9 Preliminaries</b>	<b>12</b>
9.1 Messages . . . . .	13
<b>10 Sectioning commands</b>	<b>13</b>
<b>11 Line counting</b>	<b>16</b>
11.1 Choosing the system of lineation . . . . .	16
11.2 Line-number counters and lists . . . . .	19
11.3 Reading the line-list file . . . . .	19
11.4 Commands within the line-list file . . . . .	20
11.5 Writing to the line-list file . . . . .	28
<b>12 Marking text for notes</b>	<b>31</b>
<b>13 Parallel environments</b>	<b>32</b>
<b>14 Paragraph decomposition and reassembly</b>	<b>34</b>
14.1 Boxes, counters, <code>\pstart</code> and <code>\pend</code> . . . . .	34
14.2 Processing one line . . . . .	37
14.3 Line and page number computation . . . . .	39
14.4 Line number printing . . . . .	42
14.5 Add insertions to the vertical list . . . . .	44
14.6 Penalties . . . . .	44
14.7 Printing leftover notes . . . . .	45
<b>15 Footnotes</b>	<b>46</b>
15.1 Outer-level footnote commands . . . . .	46
15.2 Normal footnote formatting . . . . .	49
<b>16 Cross referencing</b>	<b>49</b>
<b>17 Side notes</b>	<b>51</b>
<b>18 Familiar footnotes</b>	<b>53</b>
<b>19 Verse</b>	<b>53</b>
<b>20 Naming macros</b>	<b>55</b>
<b>21 Counts and boxes for parallel texts</b>	<b>55</b>
<b>22 Fixing babel</b>	<b>56</b>
<b>23 Parallel columns</b>	<b>59</b>
<b>24 Parallel pages</b>	<b>61</b>
<b>25 The End</b>	<b>69</b>

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

<b>A Examples</b>	<b>70</b>
A.1 Parallel column example . . . . .	78
A.2 Example parallel facing pages . . . . .	80
A.3 Example poetry on parallel facing pages . . . . .	86
<b>References</b>	<b>91</b>
<b>Index</b>	<b>91</b>
<b>Change History</b>	<b>100</b>

## List of Figures

1	Output from <code>villon.tex</code> . . . . .	71
2	Left page output from <code>djd17nov.tex</code> . . . . .	72
3	Right page output from <code>djd17nov.tex</code> . . . . .	73
4	First left page output from <code>djdpoems.tex</code> . . . . .	74
5	First right page output from <code>djdpoems.tex</code> . . . . .	75
6	Second left page output from <code>djdpoems.tex</code> . . . . .	76
7	Second right page output from <code>djdpoems.tex</code> . . . . .	77

## 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 0.10 (2011/08/22). You do not need to read the source code for this package in order to use it but doing so may help to answer any questions you might have. On a first reading, I suggest that you should skip anything after the general documentation in sections 2 until 8, unless you are particularly interested in the innards of **ledpar**.

## 2 The *ledpar* package

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

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

### 2.1 General

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

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

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

`\maxchunks{10}`

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

TeX has a limited number of boxes; if you get an error message along the lines of 'no room for a new box', then load the package *etex*, which needs *pdf<sub>l</sub>atex* or *x<sub>e</sub>l<sub>a</sub>tex*. If you `\maxchunks` is too little you can get a *ledmac* error message along the lines: 'Too many `\pstart` without printing. Some text will be lost.' then you will have to either increase `\maxchunks` or use the parallel printing commands

(`\Columns` or `\Pages`) more frequently.

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

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

### 3 Parallel columns

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

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

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

There is no required pagebreak before or after the columns.

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

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

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

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

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

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

### 4 Facing pages

**pages** Numbered text that is to be set on facing pages must be within a **pages** environment. 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.

**Rightside**      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**<sup>1</sup> and the **ledpar** package use macros instead. Following **\firstlinenum{<num>}** the first line number will be *<num>*, and following **\linenumincrement{<num>}** only every *<num>*th line will have a printed number. Using these macros inside the **Leftside** and **Rightside** environments gives you independent control over the left and right numbering schemes. The

---

<sup>1</sup>when used with **ledpatch** v0.2 or greater.

`\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 and paragraphs

`\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}}
```

It's possible to insert a number at every `\pstart` command. You must use the `\numberpstarttrue` command to have it. You can stop the numerotation with `\numberpstartfalse`. You can redefine the commands `\thepstartL` and `\thepstartR` to change style. The numbering restarts on each `\beginnumbering`

```
\numberpstarttrue
\numberpstartfalse
  \thepstartL
  \thepstartR
```

## 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}
  ...
```

`\hangingsymbol`      Like in `ledmac`, you could redefine the command `\hangingsymbol` to insert a character in each hanged line. If you use it, you must run `LATEX` two time. Example for the french typographie

```
\renewcommand{\hangingsymbol}{[,}
```

## 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 typeset 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 0.13 (2011/11/08).

```
1 (*code)
2 \NeedsTeXFormat{LaTeX2e}
3 \ProvidesPackage{ledpar}[2011/11/08 v0.9.2 ledmac extension for parallel texts]
4
```

With the option ‘`shiftedverses`’ a long verse on the left side (or in the right side) don’t make a blank on the corresponding verse, but the blank is put on the bottom of the page. Consequently, the verses on the parallel pages are shifted, but the shifted stop at every end of pages.

```
5 \newif\ifshiftedverses
6 \shiftedversesfalse
7 \DeclareOption{shiftedverses}{\shiftedversestrue}
8 \ProcessOptions
```

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.
9 \l@dpairingfalse
10 \newif\ifl@dpaging
11 \l@dpagingfalse
12 \ledRcolfalse

\Lcolwidth The widths of the left and right parallel columns (or pages).
\Rcolwidth 13 \newdimen\Lcolwidth
14 \Lcolwidth=0.45\textwidth
15 \newdimen\Rcolwidth
16 \Rcolwidth=0.45\textwidth
17

```

## 9.1 Messages

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

```

\led@err@TooManyPstarts
18 \newcommand*{\led@err@TooManyPstarts}{%
19 \ledmac@error{Too many \string\pstart\space without printing.
20 Some text will be lost}{\@ehc}}

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

\led@err@LeftOnRightPage
\led@err@RightOnLeftPage 24 \newcommand*{\led@err@LeftOnRightPage}{%
25 \ledmac@error{The left page has ended on a right page}{\@ehc}}
26 \newcommand*{\led@err@RightOnLeftPage}{%
27 \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.

```

28 \newcount\section@numR
29 \section@numR=\z@

```

`\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`.

```
30 \pst@rtedLfalse
31 \newif\ifpst@rtedR
32 \pst@rtedRfalse
33
```

`\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.

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

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

```
47 \newcommand*{\beginnumberingR}{%
48   \ifnumberingR
49     \led@err@NumberingStarted
50   \endnumberingR
51 \fi
52 \global\l@dnumpstartsR \z@
53 \global\pst@rtedRfalse
54 \global\numberingRtrue
55 \global\advance\section@numR \@ne
56 \global\absline@numR \z@
57 \global\line@numR \z@
58 \global\@lockR \z@
59 \global\sub@lockR \z@
60 \global\sublines@false
61 \global\let\next@page@numR\relax
62 \global\let\sub@change\relax
63 \message{Section \the\section@numR R }%
64 \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
65 \l@dend@stuff
66 \setcounter{pstartR}{1}
67 }
68
```

`\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.

```

69 \def\endnumberingR{%
70   \ifnumberingR
71     \global\numberingRfalse
72     \normal@pars
73     \ifl@dpairing
74       \global\pst@rtedRfalse
75     \else
76       \ifx\insertlines@listR\empty\else
77         \global\noteschanged@true
78       \fi
79       \ifx\line@listR\empty\else
80         \global\noteschanged@true
81       \fi
82     \fi
83     \ifnoteschanged@
84       \led@mess@NotesChanged
85     \fi
86   \else
87     \led@err@NumberingNotStarted
88   \fi}
89

```

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

```

90 \newcommand*{\pausenumberingR}{%
91   \endnumberingR\global\numberingRtrue}
92 \newcommand*{\resumenumberingR}{%
93   \ifnumberingR
94     \global\pst@rtedRtrue
95     \global\advance\section@numR \@ne
96     \led@mess@SectionContinued{\the\section@numR R}%
97     \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
98     \l@dend@stuff
99   \else
100     \led@err@numberingShouldHaveStarted
101     \endnumberingR
102     \beginnumberingR
103   \fi}
104

```

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

```

105 \newcommand*{\memorydumpL}{%
106   \endnumbering

```

```

107 \numberingtrue
108 \global\pst@rtedLtrue
109 \global\advance\section@num \@ne
110 \led@mess@SectionContinued{\the\section@num}%
111 \line@list@stuff{\jobname.\extensionchars\the\section@num}%
112 \l@dend@stuff}
113 \newcommand*\memorydumpR}{%
114 \endnumberingR
115 \numberingRtrue
116 \global\pst@rtedRtrue
117 \global\advance\section@numR \@ne
118 \led@mess@SectionContinued{\the\section@numR R}%
119 \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
120 \l@dend@stuff}
121

```

## 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.

```

122 \newif\ifbypage@R
123 \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`.

```

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

```



```
139 \fi}}
140
```

`\linenummargin` You call `\linenummargin{<word>}` to specify which margin you want your right text's line numbers in; it takes one argument, a string. You can put the line numbers in the same margin on every page using `left` or `right`; or you can use `inner` or `outer` to get them in the inner or outer margins. You can change this within a numbered section, but the change may not take effect just when you'd like; if it's done between paragraphs nothing surprising should happen.

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

```
141 \newcount\line@marginR
142 \renewcommand*{\linenummargin}[1]{%
143   \l@getline@margin{#1}%
144   \ifnum\@l@tempcntb>\m@ne
145     \ifledRcol
146       \global\line@marginR=\@l@tempcntb
147     \else
148       \global\line@margin=\@l@tempcntb
149     \fi
150 \fi}}
```

By default put right text numbers at the right.

```
151 \line@marginR=\@ne
152
```

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

```
153 \newcounter{firstlinenumR}
154 \setcounter{firstlinenumR}{5}
155 \newcounter{linenumincrementR}
156 \setcounter{linenumincrementR}{5}
```

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

```
157 \newcounter{firstsublinenumR}
158 \setcounter{firstsublinenumR}{5}
159 \newcounter{sublinenumincrementR}
160 \setcounter{sublinenumincrementR}{5}
161
```

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

```
\firstsublinenum 162 \providecommand*{\firstlinenum}{}
\sublinenumincrement 163 \providecommand*{\linenumincrement}{}

```

```

164 \providecommand*\firstsublinenum{}\}
165 \providecommand*\sublinenumincrement{}\}
166 \renewcommand*\firstlinenum}[1]{%
167   \ifledRcol \setcounter{firstlinenumR}{#1}%
168   \else      \setcounter{firstlinenum}{#1}%
169   \fi}
170 \renewcommand*\linenumincrement}[1]{%
171   \ifledRcol \setcounter{linenumincrementR}{#1}%
172   \else      \setcounter{linenumincrement}{#1}%
173   \fi}
174 \renewcommand*\firstsublinenum}[1]{%
175   \ifledRcol \setcounter{firstsublinenumR}{#1}%
176   \else      \setcounter{firstsublinenum}{#1}%
177   \fi}
178 \renewcommand*\sublinenumincrement}[1]{%
179   \ifledRcol \setcounter{sublinenumincrementR}{#1}%
180   \else      \setcounter{sublinenumincrement}{#1}%
181   \fi}
182

```

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

```

183 \newcommand*\Rlineflag{R}
184

```

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

```

185 \newcommand*\linenumrepR}[1]{\@arabic{#1}}
186 \newcommand*\sublinenumrepR}[1]{\@arabic{#1}}
187

```

`\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`.

```

188 \newcommand*\leftlinenumR{%
189   \l@dlinenumR
190   \kern\linenumsep}
191 \newcommand*\rightlinenumR{%
192   \kern\linenumsep
193   \l@dlinenumR}
194 \newcommand*\l@dlinenumR{%
195   \numlabfont\linenumrepR{\line@numR}\Rlineflag%
196   \ifsublines@
197     \ifnum\subline@num>\z@
198       \unskip\fullstop\sublinenumrepR{\subline@numR}%
199     \fi
200   \fi}
201

```

## 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.

```
202 \newcount\line@numR
203 \newcount\subline@numR
204 \newcount\absline@numR
205
```

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

`\actions@listR` Here are the commands to create these lists:

```
206 \list@create{\line@listR}
207 \list@create{\insertlines@listR}
208 \list@create{\actionlines@listR}
209 \list@create{\actions@listR}
210
```

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

```
211 \list@create{\linesinpar@listL}
212 \list@create{\linesinpar@listR}
213 \list@create{\maxlinesinpar@list}
214
```

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

```
215 \newcount\page@numR
216
```

## 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.

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

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

```
218 \ifledRcol
219 \list@clear{\line@listR}%
220 \list@clear{\insertlines@listR}%
```

```

221 \list@clear{\actionlines@listR}%
222 \list@clear{\actions@listR}%
223 \list@clear{\linesinpar@listR}%
224 \list@clear{\linesonpage@listR}
225 \else
226 \list@clearing@reg
227 \list@clear{\linesinpar@listL}%
228 \list@clear{\linesonpage@listL}%
229 \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).

```

230 \list@clear{\maxlinesinpar@list}

```

Now get the file and interpret it.

```

231 \get@linelistfile{#1}%
232 \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.

```

233 \ifledRcol
234 \global\page@numR=\m@ne
235 \ifx\actionlines@listR\empty
236 \gdef\next@actionlineR{1000000}%
237 \else
238 \gl@p\actionlines@listR\to\next@actionlineR
239 \gl@p\actions@listR\to\next@actionR
240 \fi
241 \else
242 \global\page@num=\m@ne
243 \ifx\actionlines@list\empty
244 \gdef\next@actionline{1000000}%
245 \else
246 \gl@p\actionlines@list\to\next@actionline
247 \gl@p\actions@list\to\next@action
248 \fi
249 \fi}
250

```

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  
`\@l` what it does depends on whether right text is being processed.

```

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

```

296 \fix@page{#1}%
297 \ifledRcol
298 \l@regR
299 \else
300 \l@reg
301 \fi}
302

```

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

```

\fix@page 303 \newcount\last@page@numR
304 \last@page@numR=-10000
305 \renewcommand*{\fix@page}[1]{%
306 \ifledRcol
307 \ifnum #1=\last@page@numR
308 \else
309 \ifbypage@R
310 \line@numR \z@ \subline@numR \z@
311 \fi
312 \page@numR=#1\relax
313 \last@page@numR=#1\relax
314 \def\next@page@numR{#1}%
315 \fi
316 \else
317 \ifnum #1=\last@page@num
318 \else
319 \ifbypage@
320 \line@num \z@ \subline@num \z@
321 \fi
322 \page@num=#1\relax
323 \last@page@num=#1\relax
324 \def\next@page@num{#1}%
325 \fi
326 \fi}
327

```

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

```

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

```

```

340     \subline@num \z@
341     \fi
342 \fi
343 \else
344     \ifledRcol
345         \advance\line@numR by #1\relax
346         \ifnum\line@numR<\z@
347             \led@warn@BadAdvancelineLine
348             \line@numR \z@
349         \fi
350     \else
351         \advance\line@num by #1\relax
352         \ifnum\line@num<\z@
353             \led@warn@BadAdvancelineLine
354             \line@num \z@
355         \fi
356     \fi
357 \fi
358 \set@line@action}
359

```

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

```

360 \renewcommand*{\@set}[1]{%
361     \ifledRcol
362         \ifsublines@
363             \subline@numR=#1\relax
364         \else
365             \line@numR=#1\relax
366         \fi
367         \set@line@action
368     \else
369         \ifsublines@
370             \subline@num=#1\relax
371         \else
372             \line@num=#1\relax
373         \fi
374         \set@line@action
375     \fi}
376

```

**\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.

```

377 \renewcommand*{\l@d@set}[1]{%
378     \ifledRcol
379         \line@numR=#1\relax
380         \advance\line@numR \@ne

```

```

381   \def\l@dchset@num{#1}
382   \else
383     \line@num=#1\relax
384     \advance\line@num \@ne
385     \def\l@dchset@num{#1}
386   \fi}
387 \let\l@dchset@num\relax
388

```

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

```

389 \renewcommand*{\page@action}{%
390   \ifl@edRcol
391     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
392     \xright@appenditem{\next@page@numR}\to\actions@listR
393   \else
394     \xright@appenditem{\the\absline@num}\to\actionlines@list
395     \xright@appenditem{\next@page@num}\to\actions@list
396   \fi}

```

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

```

397 \renewcommand*{\set@line@action}{%
398   \ifl@edRcol
399     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
400     \ifsublines@
401       \l@dtempcnta=-\subline@numR
402     \else
403       \l@dtempcnta=-\line@numR
404     \fi
405     \advance\l@dtempcnta by -5000\relax
406     \xright@appenditem{\the\l@dtempcnta}\to\actions@listR
407   \else
408     \xright@appenditem{\the\absline@num}\to\actionlines@list
409     \ifsublines@
410       \l@dtempcnta=-\subline@num
411     \else
412       \l@dtempcnta=-\line@num
413     \fi
414     \advance\l@dtempcnta by -5000\relax
415     \xright@appenditem{\the\l@dtempcnta}\to\actions@list
416   \fi}
417

```

`\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.

```

418 \renewcommand*{\sub@action}{%
419   \ifl@edRcol
420     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
421     \ifsublines@

```



```

422     \xright@appenditem{-1001}\to\actions@listR
423   \else
424     \xright@appenditem{-1002}\to\actions@listR
425   \fi
426 \else
427   \xright@appenditem{\the\absline@num}\to\actionlines@list
428   \ifsublines@
429     \xright@appenditem{-1001}\to\actions@list
430   \else
431     \xright@appenditem{-1002}\to\actions@list
432   \fi
433 \fi}
434

```

`\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.

```

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

```

```

468     \fi
469   \fi}

\lock@off \lock@off adds an entry to the action-code list to turn line number locking off.
\do@lockoff 470
\do@lockoffR 471
\skip@lockoff 472 \newcommand{\do@lockoffR}{%
473   \xright@appenditem{\the\absline@numR}\to\actionlines@listR
474   \ifsublines@
475     \xright@appenditem{-1006}\to\actions@listR
476     \ifnum\sub@lockR=\tw@
477       \sub@lockR \thr@@
478     \else
479       \sub@lockR \z@
480     \fi
481   \else
482     \xright@appenditem{-1004}\to\actions@listR
483     \ifnum\@lockR=\tw@
484       \@lockR \thr@@
485     \else
486       \@lockR \z@
487     \fi
488   \fi}
489
490 \renewcommand*{\do@lockoff}{%
491   \ifledRcol
492     \do@lockoffR
493   \else
494     \do@lockoffL
495   \fi}
496 \global\let\lock@off=\do@lockoff
497

```

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

```

498 \providecommand*{\n@num}{%
499 \renewcommand*{\n@num}{%
500   \ifledRcol
501     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
502     \xright@appenditem{-1007}\to\actions@listR
503   \else
504     \n@num@reg
505   \fi}
506

```

**\@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`.

```
507 \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.

```
508 \renewcommand*{\@ref}[2]{%
509 \ifledRcol
510 \global\insert@countR=#1\relax
511 \loop\ifnum\insert@countR>\z@
512 \xright@appenditem{\the\absline@numR}\to\insertlines@listR
513 \global\advance\insert@countR \m@ne
514 \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.

```
515 \begingroup
516 \let\@ref=\dummy@ref
517 \let\page@action=\relax
518 \let\sub@action=\relax
519 \let\set@line@action=\relax
520 \let\@lab=\relax
521 #2
522 \global\endpage@num=\page@numR
523 \global\endline@num=\line@numR
524 \global\endsubline@num=\subline@numR
525 \endgroup
```

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

```
526 \xright@appenditem%
527 {\the\page@numR|\the\line@numR}%
528 \ifsublines@ \the\subline@numR \else 0\fi}%
529 \the\endpage@num|\the\endline@num}%
530 \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.

```
531 #2
532 \else
```

And when not in right text

```
533 \@ref@reg{#1}{#2}%
534 \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.

```
535 \providecommand*\@pend}[1]{}
536 \renewcommand*\@pend}[1]{%
537   \xright@appenditem{#1}\to\linesinpar@listL}
538 \providecommand*\@pendR}[1]{}
539 \renewcommand*\@pendR}[1]{%
540   \xright@appenditem{#1}\to\linesinpar@listR}
541
```

`\@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.

```
542 \providecommand*\@lopL}[1]{}
543 \renewcommand*\@lopL}[1]{%
544   \xright@appenditem{#1}\to\linesonpage@listL}
545 \providecommand*\@lopR}[1]{}
546 \renewcommand*\@lopR}[1]{%
547   \xright@appenditem{#1}\to\linesonpage@listR}
548
```

## 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`.

```
549 \newwrite\linenum@outR
```

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

```
\first@linenum@out@Rfalse 550 \newif\iffirst@linenum@out@R
551 \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.

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

```
561 \fi}
562
```

`\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.

```
563 \newcommand*{\new@lineR}{%
564 \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.

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

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

```
581 \renewcommand*{\startsub}{\dimen0\lastskip
582 \ifdim\dimen0>0pt \unskip \fi
583 \ifledRcol \write\linenum@outR{\string\sub@on}%
584 \else \write\linenum@out{\string\sub@on}%
585 \fi
586 \ifdim\dimen0>0pt \hskip\dimen0 \fi}
587 \def\endsub{\dimen0\lastskip
588 \ifdim\dimen0>0pt \unskip \fi
589 \ifledRcol \write\linenum@outR{\string\sub@off}%
590 \else \write\linenum@out{\string\sub@off}%
591 \fi
592 \ifdim\dimen0>0pt \hskip\dimen0 \fi}
593
```

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

```
594 \renewcommand*{\advanceline}[1]{%
595 \ifledRcol \write\linenum@outR{\string\@adv[#1]}%
596 \else \write\linenum@out{\string\@adv[#1]}%
597 \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.

```
598 \renewcommand*{\setline}[1]{%
599   \ifnum#1<\z@
600     \led@warn@BadSetline
601   \else
602     \ifledRcol \write\linenum@outR{\string\@set[#1]}%
603   \else      \write\linenum@out{\string\@set[#1]}%
604   \fi
605 \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.

```
606 \renewcommand*{\setlinenum}[1]{%
607   \ifnum#1<\z@
608     \led@warn@BadSetlinenum
609   \else
610     \ifledRcol \write\linenum@outR{\string\l@d@set[#1]}
611   \else      \write\linenum@out{\string\l@d@set[#1]} \fi
612   \fi}
613
```

`\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.

```
614 \renewcommand*{\startlock}{%
615   \ifledRcol \write\linenum@outR{\string\lock@on}%
616   \else      \write\linenum@out{\string\lock@on}%
617   \fi}
618 \def\endlock{%
619   \ifledRcol \write\linenum@outR{\string\lock@off}%
620   \else      \write\linenum@out{\string\lock@off}%
621   \fi}
622
```

`\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.

```
623 \renewcommand*{\skipnumbering}{%
624   \ifledRcol \write\linenum@outR{\string\n@num}%
625             \advanceline{-1}%
626   \else
627     \skipnumbering@reg
628   \fi}
629
```

## 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.

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

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

```
647 \renewcommand{\edtext}[2]{\leavevmode
648   \begingroup
649     \no@expands
650     \xdef\@tag{#1}%
651     \set@line
652     \ifledRcol \global\insert@countR \z@
653     \else      \global\insert@count \z@ \fi
654     \ignorespaces #2\relax
655     \flag@start
656   \endgroup
657   \showlemma{#1}%
```

```

658 \ifx\end@lemmas\empty \else
659   \gl@p\end@lemmas\to\x@lemma
660   \x@lemma
661   \global\let\x@lemma=\relax
662 \fi
663 \flag@end}
664

```

`\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`.

```

665 \renewcommand*{\set@line}{%
666   \ifledRcol
667     \ifx\line@listR\empty
668       \global\noteschanged@true
669       \xdef\l@d@nums{000|000|000|000|000|000|\edfont@info}%
670     \else
671       \gl@p\line@listR\to\@tempb
672       \xdef\l@d@nums{\@tempb|\edfont@info}%
673       \global\let\@tempb=\undefined
674     \fi
675   \else
676     \ifx\line@list\empty
677       \global\noteschanged@true
678       \xdef\l@d@nums{000|000|000|000|000|000|\edfont@info}%
679     \else
680       \gl@p\line@list\to\@tempb
681       \xdef\l@d@nums{\@tempb|\edfont@info}%
682       \global\let\@tempb=\undefined
683     \fi
684   \fi}
685

```

## 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
pages    parallel pages.
chapterinpages 686 \newenvironment{pairs}{%}
687   \l@dpairingtrue
688   \l@dpagingfalse
689 }{%
690   \l@dpairingfalse
691 }

```

The `pages` environment additionally sets the ‘column’ widths to the `\textwidth` (as known at the time the package is called). In this environment, there are two text in parallel on 2 pages. To prevent chapters starting on a lefthand page, the `\chapter` command is redefined to not clear pages.



```

692 \newenvironment{pages}{%
693   \let\oldchapter\chapter
694   \let\chapter\chapterinpages
695   \l@ddpairingtrue
696   \l@ddpacingtrue
697   \setlength{\Lcolwidth}{\textwidth}%
698   \setlength{\Rcolwidth}{\textwidth}%
699 }{%
700   \l@ddpairingfalse
701   \l@ddpacingfalse
702   \let\chapter\oldchapter
703 }
704 \newcommand{\chapterinpages}{\thispagestyle{plain}%
705                               \global\@topnum\z@
706                               \@afterindentfalse
707                               \secdef\@chapter\@schapter}
708

```

**ifinstanzaL** These boolean tests are switched by the `\stanza` command, using either the left  
**ifinstanzaR** or right side.

```

709 \newif\ifinstanzaL
710 \newif\ifinstanzaR

```

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

```

711 \newenvironment{Leftside}{%
712   \ledRcolfalse
713   \let\beginnumbering\beginnumbering\setcounter{pstartL}{1}
714   \let\pstart\pstartL
715   \let\thepstart\thepstartL
716   \let\pend\pendL
717   \let\memorydump\memorydumpL
718   \Leftsidehook
719   \let\oldstanza\stanza
720   \renewcommand{\stanza}{\oldstanza\global\instanzaLtrue}
721 }{
722   \let\stanza\oldstanza
723   \Leftsidehookend}

```

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

```

\Rightsidehook 724 \newcommand*{\Leftsidehook}{}
\Rightsidehookend 725 \newcommand*{\Leftsidehookend}{}
726 \newcommand*{\Rightsidehook}{}
727 \newcommand*{\Rightsidehookend}{}
728

```

**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.

```

729 \newenvironment{Rightside}{%
730   \ledRcoltrue
731   \let\beginnumbering\beginnumberingR
732   \let\endnumbering\endnumberingR
733   \let\pausenumbering\pausenumberingR
734   \let\resumenumbering\resumenumberingR
735   \let\memorydump\memorydumpR
736   \let\thepstart\thepstartR
737   \let\pstart\pstartR
738   \let\pend\pendR
739   \let\lineation\lineationR
740   \Rightsidehook
741   \let\oldstanza\stanza
742   \renewcommand{\stanza}{\oldstanza\global\instanzaRtrue}
743 }{%
744   \ledRcolfalse
745   \let\stanza\oldstanza
746   \Rightsidehookend
747 }
748

```

## 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.

```

749 \newcount\num@linesR
750 \newbox\one@lineR
751 \newcount\par@lineR

```

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

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

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

```

752
753 \newcounter{pstartL}
754 \renewcommand{\thepstartL}{\bf\@arabic\c@pstartL}. }
755 \newcounter{pstartR}
756 \renewcommand{\thepstartR}{\bf\@arabic\c@pstartR}. }
757
758 \newcommand*{\pstartL}{
759 \if@nbreak
760 \let\@oldnbreak\@nbreaktrue
761 \else
762 \let\@oldnbreak\@nbreakfalse
763 \fi
764 \@nbreaktrue
765 \ifnumbering \else
766 \led@err@PstartNotNumbered
767 \beginnumbering
768 \fi
769 \ifnumberedpar@
770 \led@err@PstartInPstart
771 \pend
772 \fi

```

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

```

773 \ifpst@rtedL\else
774 \list@clear{\inserts@list}%
775 \global\let\next@insert=\empty
776 \global\pst@rtedLtrue
777 \fi
778 \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.

```

779 \global\advance\l@dnumpstartsL \@ne
780 \ifnum\l@dnumpstartsL>\l@dc@maxchunks
781 \led@err@TooManyPstarts
782 \global\l@dnumpstartsL=\l@dc@maxchunks
783 \fi
784 \global\setnamebox{\l@dLcolrawbox\the\l@dnumpstartsL}=\vbox\bgroup\ifautopar\else\ifnumberpstart\the
785 \hsize=\lcolwidth

```

```

786 \numberedpar@true}
787 \newcommand*{\pstartR}{
788 \if@nobreak
789 \let\@oldnobreak\@nobreaktrue
790 \else
791 \let\@oldnobreak\@nobreakfalse
792 \fi
793 \@nobreaktrue
794 \ifnumberingR \else
795 \led@err@PstartNotNumbered
796 \beginnumberingR
797 \fi
798 \ifnumberedpar@
799 \led@err@PstartInPstart
800 \pendR
801 \fi
802 \ifpst@rtedR\else
803 \list@clear{\inserts@listR}%
804 \global\let\next@insertR=\empty
805 \global\pst@rtedRtrue
806 \fi
807 \begingroup\normal@pars
808 \global\advance\l@dnumpstartsR \@ne
809 \ifnum\l@dnumpstartsR>\l@dc@maxchunks
810 \led@err@TooManyPstarts
811 \global\l@dnumpstartsR=\l@dc@maxchunks
812 \fi
813 \global\setnamebox{\l@dRcolrawbox\the\l@dnumpstartsR}=\vbox\bgroup\ifautopar\else\ifnumb
814 \hspace=\Rcolwidth
815 \numberedpar@true}

```

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

```

816 \newcommand*{\pendL}{\ifnumbering \else
817 \led@err@PendNotNumbered
818 \fi
819 \ifnumberedpar@ \else
820 \led@err@PendNoPstart
821 \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.

```

822 \l@dzeropenalties
823 \endgraf\global\num@lines=\prevgraf\egroup
824 \global\par@line=0

```

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

```

825 \endgroup
826 \ignorespaces
827 \@oldnobreak
828 \ifnumberpstart
829 \addtocounter{pstartL}{1}
830 \fi}
831

```

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

```

832 \newcommand*{\pendR}{\ifnumberingR \else
833   \led@err@PendNotNumbered
834   \fi
835   \ifnumberedpar@ \else
836   \led@err@PendNoPstart
837   \fi
838   \l@dzeropenalties
839   \endgraf\global\num@linesR=\prevgraf\egroup
840   \global\par@lineR=0
841   \endgroup
842   \ignorespaces
843   \@oldnobreak
844   \ifnumberpstart
845   \addtocounter{pstartR}{1}
846   \fi
847 }
848

```

## 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 analogously for a line of right text.

```

849 \newbox\l@dleftbox
850 \newbox\l@drightbox
851

```

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

```

\countRline 852 \newcount\countLline
853   \countLline \z@
854 \newcount\countRline
855   \countRline \z@
856

```

`\@donereallinesL` We need to know the number of ‘real’ lines output (i.e., those that have been input by the user), and the total lines output (which includes any blank lines output for synchronisation).

`\@donereallinesR`

`\@donetotallinesL`

`\@donetotallinesR`

```

857 \newcount\@donereallinesL
858 \newcount\@donetotallinesL
859 \newcount\@donereallinesR
860 \newcount\@donetotallinesR
861

```

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

```

862 \newcommand*{\do@lineL}{%
863 \ifinstanzaL\manageparhangingsymbol\fi
864 \advance\countLline \@ne
865 \ifvbox\namebox{l@dLcolrawbox\the\l@dpscL}%
866 {\vbadness=10000
867 \splittopskip=\z@
868 \do@lineLhook
869 \l@demptyd@ta
870 \global\setbox\one@line=\vsplit\namebox{l@dLcolrawbox\the\l@dpscL}
871 to\baselineskip}%
872 \unvbox\one@line \global\setbox\one@line=\lastbox
873 \getline@numL
874 \setbox\l@dleftbox
875 \hb@xt@ \Lcolwidth{%
876 \affixline@num
877 \l@dld@ta
878 \add@inserts
879 \affixside@note
880 \l@dlsn@te
881 {\ledllfill\hb@xt@ \wd\one@line{\new@line\l@dunhbox@line{\one@line}}\ledrlfill\l@drd@
882 \l@drrn@te
883 }}%
884 \add@penaltiesL
885 \global\advance\@donereallinesL\@ne
886 \global\advance\@donetotallinesL\@ne
887 \else
888 \setbox\l@dleftbox \hb@xt@ \Lcolwidth{\hspace*\Lcolwidth}}%
889 \global\advance\@donetotallinesL\@ne
890 \fi}
891
892

```

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

```

\do@lineRhook 893 \newcommand*{\do@lineLhook}{}
894 \newcommand*{\do@lineRhook}{}
895

```

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

```

896 \newcommand*{\do@lineR}{%
897 \ifinstanzaR\manageparhangingsymbol\fi

```

```

898 \advance\countRline \@ne
899 \ifvbox\namebox{\l@dRcolrawbox\the\l@dpscR}%
900 {\vbadness=10000
901 \splittopskip=\z@
902 \do@lineRhook
903 \l@emptyd@ta
904 \global\setbox\one@lineR=\vsplit\namebox{\l@dRcolrawbox\the\l@dpscR}
905 to\baselineskip}%
906 \unvbox\one@lineR \global\setbox\one@lineR=\lastbox
907 \getline@numR
908 \setbox\l@drighthbox
909 \hb@xt@ \Rcolwidth{%
910 \affixline@numR
911 \l@dld@ta
912 \add@insertsR
913 \affixside@noteR
914 \l@dlsn@te
915 {\ledllfill\hb@xt@ \wd\one@lineR{\new@lineR\l@dunhbox@line{\one@lineR}}\ledrlfill\l@drd@ta%
916 \l@drsn@te
917 }}%
918 \add@penaltiesR
919 \global\advance\@donereallinesR\@ne
920 \global\advance\@donetotallinesR\@ne
921 \else
922 \setbox\l@drighthbox \hb@xt@ \Rcolwidth{\hspace*\Rcolwidth}}
923 \global\advance\@donetotallinesR\@ne
924 \fi}
925
926

```

### 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.

```

927 \newcommand*{\getline@numR}{%
928 \global\advance\absline@numR \@ne
929 \do@actionsR
930 \do@ballastR
931 \ifsublines@
932 \ifnum\sub@lockR<\tw@
933 \global\advance\subline@numR \@ne
934 \fi
935 \else
936 \ifnum\@lockR<\tw@
937 \addtocounter{hbox}{10}%
938 \global\advance\line@numR \@ne
939 \global\subline@numR \z@
940 \fi
941 \fi}

```

```

942 \newcommand*{\getline@numL}{%
943   \global\advance\absline@num \@ne
944   \do@actions
945   \do@ballast
946   \ifsublines@
947     \ifnum\sub@lock<\tw@
948       \global\advance\subline@num \@ne
949     \fi
950   \else
951     \ifnum\@lock<\tw@
952       \global\advance\line@num \@ne
953       \addtocounter{hbox}{10}%
954       \global\subline@num \z@
955     \fi
956   \fi}
957
958

```

`\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.

```

959 \newcommand*{\do@ballastR}{\global\ballast@count=\z@
960   \begingroup
961     \advance\absline@numR \@ne
962     \ifnum\next@actionlineR=\absline@numR
963       \ifnum\next@actionR>-1001
964         \global\advance\ballast@count by -\c@ballast
965       \fi
966     \fi
967   \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.

```

968 \newcommand*{\do@actions@fixedcodeR}{%
969   \ifcase\@l@dttempcnta%
970     \or%                % 1001
971       \global\sublines@true
972     \or%                % 1002
973       \global\sublines@false
974     \or%                % 1003
975       \global\@lockR=\@ne
976     \or%                % 1004
977       \ifnum\@lockR=\tw@
978         \global\@lockR=\thr@@
979       \else
980         \global\@lockR=\z@
981       \fi

```



```

982 \or% % 1005
983 \global\sub@lockR=\@ne
984 \or% % 1006
985 \ifnum\sub@lockR=\tw@
986 \global\sub@lockR=\thr@@
987 \else
988 \global\sub@lockR=\z@
989 \fi
990 \or% % 1007
991 \l@dskipnumbertrue
992 \else
993 \led@warn@BadAction
994 \fi}
995
996
997 \newcommand*{\do@actionsR}{%
998 \global\let\do@actions@nextR=\relax
999 \@l@dttempcntb=\absline@numR
1000 \ifnum\@l@dttempcntb<\next@actionlineR\else
1001 \ifnum\next@actionR>-1001\relax
1002 \global\page@numR=\next@actionR
1003 \ifbypage@R
1004 \global\line@numR \z@ \global\subline@numR \z@
1005 \fi
1006 \else
1007 \ifnum\next@actionR<-4999\relax % 9/05 added relax here
1008 \@l@dttempcnta=-\next@actionR
1009 \advance\@l@dttempcnta by -5001\relax
1010 \ifsublines@
1011 \global\subline@numR=\@l@dttempcnta
1012 \else
1013 \global\line@numR=\@l@dttempcnta
1014 \fi
1015 \else
1016 \@l@dttempcnta=-\next@actionR
1017 \advance\@l@dttempcnta by -1000\relax
1018 \do@actions@fixedcodeR
1019 \fi
1020 \fi
1021 \ifx\actionlines@listR\empty
1022 \gdef\next@actionlineR{1000000}%
1023 \else
1024 \gl@p\actionlines@listR\to\next@actionlineR
1025 \gl@p\actions@listR\to\next@actionR
1026 \global\let\do@actions@nextR=\do@actionsR
1027 \fi
1028 \fi
1029 \do@actions@nextR}
1030

```

## 14.4 Line number printing

```

\l@dcalcnun \affixline@numR is the right text version of the \affixline@num macro.
\ch@cksub@l@ckR 1031
\ch@ck@l@ckR 1032 \providecommand*\l@dcalcnun}[3]{%
\fx@l@cksR 1033 \ifnum #1 > #2\relax
\affixline@numR 1034 \@l@tempcnta = #1\relax
1035 \advance\@l@tempcnta by -#2\relax
1036 \divide\@l@tempcnta by #3\relax
1037 \multiply\@l@tempcnta by #3\relax
1038 \advance\@l@tempcnta by #2\relax
1039 \else
1040 \@l@tempcnta=#2\relax
1041 \fi}
1042
1043 \newcommand*\ch@cksub@l@ckR}{%
1044 \ifcase\sub@lockR
1045 \or
1046 \ifnum\sublock@disp=\@ne
1047 \@l@tempcntb \z@ \@l@tempcnta \@ne
1048 \fi
1049 \or
1050 \ifnum\sublock@disp=\tw@
1051 \else
1052 \@l@tempcntb \z@ \@l@tempcnta \@ne
1053 \fi
1054 \or
1055 \ifnum\sublock@disp=\z@
1056 \@l@tempcntb \z@ \@l@tempcnta \@ne
1057 \fi
1058 \fi}
1059
1060 \newcommand*\ch@ck@l@ckR}{%
1061 \ifcase\@lockR
1062 \or
1063 \ifnum\lock@disp=\@ne
1064 \@l@tempcntb \z@ \@l@tempcnta \@ne
1065 \fi
1066 \or
1067 \ifnum\lock@disp=\tw@
1068 \else
1069 \@l@tempcntb \z@ \@l@tempcnta \@ne
1070 \fi
1071 \or
1072 \ifnum\lock@disp=\z@
1073 \@l@tempcntb \z@ \@l@tempcnta \@ne
1074 \fi
1075 \fi}
1076
1077 \newcommand*\fx@l@cksR}{%

```

```

1078 \ifcase\@lockR
1079 \or
1080 \global\@lockR \tw@
1081 \or \or
1082 \global\@lockR \z@
1083 \fi
1084 \ifcase\sub@lockR
1085 \or
1086 \global\sub@lockR \tw@
1087 \or \or
1088 \global\sub@lockR \z@
1089 \fi}
1090
1091
1092 \newcommand*{\affixline@numR}{%
1093 \ifl@dskipnumber
1094 \global\l@dskipnumberfalse
1095 \else
1096 \ifsublines@
1097 \l@dttempcntb=\subline@numR
1098 \l@dcalcnum{\subline@numR}{\c@firstsublinenumR}{\c@sublinenumincrementR}%
1099 \ch@cksub@lockR
1100 \else
1101 \l@dttempcntb=\line@numR
1102 \ifx\linenumberlist\empty
1103 \l@dcalcnum{\line@numR}{\c@firstlinenumR}{\c@linenumincrementR}%
1104 \else
1105 \l@dttempcnta=\line@numR
1106 \edef\rem@inder{\linenumberlist,\number\line@numR,}%
1107 \edef\sc@n@list{\def\noexpand\sc@n@list
1108 ###1,\number\l@dttempcnta,###2|{\def\noexpand\rem@inder{###2}}}%
1109 \sc@n@list\expandafter\sc@n@list\rem@inder|
1110 \ifx\rem@inder\empty\advance\l@dttempcnta\@ne\fi
1111 \fi
1112 \ch@ck@l@ckR
1113 \fi
1114 \ifnum\l@dttempcnta=\l@dttempcntb
1115 \if@twocolumn
1116 \if@firstcolumn
1117 \gdef\l@dld@ta{\llap{\leftlinenumR}}}%
1118 \else
1119 \gdef\l@drd@ta{\rlap{\rightlinenumR}}}%
1120 \fi
1121 \else
1122 \l@dttempcntb=\line@marginR
1123 \ifnum\l@dttempcntb>\@ne
1124 \advance\l@dttempcntb by\page@numR
1125 \fi
1126 \ifodd\l@dttempcntb
1127 \gdef\l@drd@ta{\rlap{\rightlinenumR}}}%

```

```

1128     \else
1129         \gdef\l@dld@ta{\llap{\leftlinenumR}}}%
1130     \fi
1131 \fi
1132 \fi
1133 \f@x@l@cksR
1134 \fi}
1135

```

### 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.

```

1136 \list@create{\inserts@listR}

```

`\add@insertsR` The right text version.

```

\add@inserts@nextR 1137 \newcommand*{\add@insertsR}{%
1138     \global\let\add@inserts@nextR=\relax
1139     \ifx\inserts@listR\empty \else
1140         \ifx\next@insertR\empty
1141             \ifx\insertlines@listR\empty
1142                 \global\noteschanged@true
1143                 \gdef\next@insertR{100000}%
1144             \else
1145                 \gl@p\insertlines@listR\to\next@insertR
1146             \fi
1147         \fi
1148         \ifnum\next@insertR=\absline@numR
1149             \gl@p\inserts@listR\to\@insertR
1150             \@insertR
1151             \global\let\@insertR=\undefined
1152             \global\let\next@insertR=\empty
1153             \global\let\add@inserts@nextR=\add@insertsR
1154         \fi
1155     \fi
1156     \add@inserts@nextR}
1157

```

### 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@tempcnta` 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@tempcnta=\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.

```
1158 \newcommand*{\add@penaltiesL}{\}
1159 \newcommand*{\add@penaltiesR}{\}
1160
```

## 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.

```
1161 \newcommand*{\flush@notesR}{%
1162 \xloop
1163 \ifx\inserts@listR\empty \else
1164 \glp\inserts@listR\to\@insertR
1165 \@insertR
1166 \global\let\@insertR=\undefined
1167 \repeat}
```

1168

## 15 Footnotes

### 15.1 Outer-level footnote commands

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

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

```
1169 \renewcommand*{\Afootnote}[1]{%
1170   \ifnumberedpar@
1171     \ifledRcol
1172       \xright@appenditem{\noexpand\vAfootnote{A}%
1173         {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1174       \global\advance\insert@countR \@ne
1175     \else
1176       \xright@appenditem{\noexpand\vAfootnote{A}%
1177         {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1178       \global\advance\insert@count \@ne
1179     \fi
```

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

```
1180   \else
1181     \vAfootnote{A}{\l@d@nums}{\@tag}{#1}%
1182   \fi\ignorespaces}
```

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

```
\Cfootnote 1183 \renewcommand*{\Bfootnote}[1]{%
\Dfootnote 1184   \ifnumberedpar@
\Efootnote 1185     \ifledRcol
1186       \xright@appenditem{\noexpand\vBfootnote{B}%
1187         {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1188       \global\advance\insert@countR \@ne
1189     \else
1190       \xright@appenditem{\noexpand\vBfootnote{B}%
1191         {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1192       \global\advance\insert@count \@ne
1193     \fi
1194   \else
1195     \vBfootnote{B}{\l@d@nums}{\@tag}{#1}%
1196   \fi\ignorespaces}

1197 \renewcommand*{\Cfootnote}[1]{%
1198   \ifnumberedpar@
```

```

1199 \ifledRcol
1200 \xright@appenditem{\noexpand\VCfootnote{C}%
1201                    {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1202 \global\advance\insert@countR \@ne
1203 \else
1204 \xright@appenditem{\noexpand\VCfootnote{C}%
1205                    {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1206 \global\advance\insert@count \@ne
1207 \fi
1208 \else
1209 \VCfootnote{C}{\{0|0|0|0|0|0|0|0\}}{#1}}%
1210 \fi\ignorespaces}
1211 \renewcommand*{\Dfootnote}[1]{%
1212 \ifnumberedpar@
1213 \ifledRcol
1214 \xright@appenditem{\noexpand\VDfootnote{D}%
1215                    {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1216 \global\advance\insert@countR \@ne
1217 \else
1218 \xright@appenditem{\noexpand\VDfootnote{D}%
1219                    {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1220 \global\advance\insert@count \@ne
1221 \fi
1222 \else
1223 \VDfootnote{D}{\{0|0|0|0|0|0|0|0\}}{#1}}%
1224 \fi\ignorespaces}
1225 \renewcommand*{\Efootnote}[1]{%
1226 \ifnumberedpar@
1227 \ifledRcol
1228 \xright@appenditem{\noexpand\VEfootnote{E}%
1229                    {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1230 \global\advance\insert@countR \@ne
1231 \else
1232 \xright@appenditem{\noexpand\VEfootnote{E}%
1233                    {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1234 \global\advance\insert@count \@ne
1235 \fi
1236 \else
1237 \VEfootnote{E}{\{0|0|0|0|0|0|0|0\}}{#1}}%
1238 \fi\ignorespaces}
1239

```

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

```

\mpBfootnote 1240 \renewcommand*{\mpAfootnote}[1]{%
\mpCfootnote 1241 \ifnumberedpar@
\mpDfootnote 1242 \ifledRcol
\mpEfootnote 1243 \xright@appenditem{\noexpand\mpvAfootnote{A}%
1244                    {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1245 \global\advance\insert@countR \@ne

```

```

1246 \else
1247   \xright@appenditem{\noexpand\mpvAfootnote{A}%
1248                     {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1249   \global\advance\insert@count \@ne
1250 \fi
1251 \else
1252   \mpvAfootnote{A}{0|0|0|0|0|0|0}{#1}%
1253 \fi\ignorespaces}
1254 \renewcommand*{\mpBfootnote}[1]{%
1255   \ifnumberedpar@
1256   \ifledRcol
1257     \xright@appenditem{\noexpand\mpvBfootnote{B}%
1258                       {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1259     \global\advance\insert@countR \@ne
1260   \else
1261     \xright@appenditem{\noexpand\mpvBfootnote{B}%
1262                       {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1263     \global\advance\insert@count \@ne
1264   \fi
1265   \else
1266     \mpvBfootnote{B}{0|0|0|0|0|0|0}{#1}%
1267   \fi\ignorespaces}
1268 \renewcommand*{\mpCfootnote}[1]{%
1269   \ifnumberedpar@
1270   \ifledRcol
1271     \xright@appenditem{\noexpand\mpvCfootnote{C}%
1272                       {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1273     \global\advance\insert@countR \@ne
1274   \else
1275     \xright@appenditem{\noexpand\mpvCfootnote{C}%
1276                       {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1277     \global\advance\insert@count \@ne
1278   \fi
1279   \else
1280     \mpvCfootnote{C}{0|0|0|0|0|0|0}{#1}%
1281   \fi\ignorespaces}
1282 \renewcommand*{\mpDfootnote}[1]{%
1283   \ifnumberedpar@
1284   \ifledRcol
1285     \xright@appenditem{\noexpand\mpvDfootnote{D}%
1286                       {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1287     \global\advance\insert@countR \@ne
1288   \else
1289     \xright@appenditem{\noexpand\mpvDfootnote{D}%
1290                       {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1291     \global\advance\insert@count \@ne
1292   \fi
1293   \else
1294     \mpvDfootnote{D}{0|0|0|0|0|0|0}{#1}%

```



```

1295 \fi\ignorespaces}
1296 \renewcommand*{\mpEfootnote}[1]{%
1297 \ifnumberedpar@
1298 \ifledRcol
1299 \xright@appenditem{\noexpand\mpvEfootnote{E}%
1300 {{\l@d@nums}{\@tag}{#1}}}\to\inserts@listR
1301 \global\advance\insert@countR \@ne
1302 \else
1303 \xright@appenditem{\noexpand\mpvEfootnote{E}%
1304 {{\l@d@nums}{\@tag}{#1}}}\to\inserts@list
1305 \global\advance\insert@count \@ne
1306 \fi
1307 \else
1308 \mpvEfootnote{E}{{0|0|0|0|0|0|0}{#1}}%
1309 \fi\ignorespaces}

```

## 15.2 Normal footnote formatting

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

```

\printlinesR This is the right text version of \printlines and takes account of \Rlineflag.
\ledsavedprintlines Just in case, \ledsavedprintlines is a copy of the original \printlines.
    Just a reminder of the arguments:
    \printlinesR #1 | #2 | #3 | #4 | #5 | #6 | #7
    \printlinesR start-page | line | subline | end-page | line | subline | font
1310 \def\printlinesR#1|#2|#3|#4|#5|#6|#7|{\begingroup
1311 \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
1312 \ifl@d@pnum #1\fullstop\fi
1313 \ifledplinenum \linenumr@p{#2}\Rlineflag\else \symlinenum\fi
1314 \ifl@d@ssub \fullstop \sublinenumr@p{#3}\fi
1315 \ifl@d@dash \endashchar\fi
1316 \ifl@d@pnum #4\fullstop\fi
1317 \ifl@d@elin \linenumr@p{#5}\Rlineflag\fi
1318 \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumr@p{#6}\fi
1319 \endgroup}
1320
1321 \let\ledsavedprintlines\printlines
1322

```

## 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.

```

```

1323 \list@create{\labelref@listR}
1324

```

**\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.

```

1325 \renewcommand*{\edlabel}[1]{\@bsphack
1326   \ifledRcol
1327     \write\linenum@outR{\string\@lab}%
1328     \ifx\labelref@listR\empty
1329       \xdef\label@refs{\zz@@@}%
1330     \else
1331       \gl@p\labelref@listR\to\label@refs
1332     \fi
1333     \protected@write\@auxout{%
1334       {\string\l@dmake@labelsR\space\thepage|\label@refs|{#1}}%
1335   \else
1336     \write\linenum@out{\string\@lab}%
1337     \ifx\labelref@listR\empty
1338       \xdef\label@refs{\zz@@@}%
1339     \else
1340       \gl@p\labelref@listR\to\label@refs
1341     \fi
1342   \fi
1343   \protected@write\@auxout{%
1344     {\string\l@dmake@labelsR\space\thepage|\label@refs|{#1}}%
1345   \@esphack}
1346

```

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

```

1347 \def\l@dmake@labelsR#1|#2|#3|#4{%
1348   \expandafter\ifx\csname the@label#4\endcsname \relax\else
1349     \led@warn@DuplicateLabel{#4}%
1350   \fi
1351   \expandafter\gdef\csname the@label#4\endcsname{#1|#2\Rlineflag|#3}%
1352   \ignorespaces}
1353 \AtBeginDocument{%
1354   \def\l@dmake@labelsR#1|#2|#3|#4{%
1355   }
1356

```

**\@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.

```

1357 \renewcommand*{\@lab}{%

```

```

1358 \ifledRcol
1359 \xright@appenditem{\linenumr@p{\line@numR}}|{%
1360 \ifsublines@ \sublinenumr@p{\subline@numR}\else 0\fi}%
1361 \to\labelref@listR
1362 \else
1363 \xright@appenditem{\linenumr@p{\line@num}}|{%
1364 \ifsublines@ \sublinenumr@p{\subline@num}\else 0\fi}%
1365 \to\labelref@list
1366 \fi}
1367

```

## 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 1368 \newcount\sidenote@marginR
1369 \renewcommand*{\sidenotemargin}[1]{%
1370 \l@dgetsidenote@margin{#1}%
1371 \ifnum\@l@dttempcntb>\m@ne
1372 \ifledRcol
1373 \global\sidenote@marginR=\@l@dttempcntb
1374 \else
1375 \global\sidenote@margin=\@l@dttempcntb
1376 \fi
1377 \fi}}
1378 \sidenotemargin{right}
1379 \global\sidenote@margin=\@ne
1380

```

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

```

\l@dcsnote 1381 \renewcommand*{\l@dlsnote}[1]{%
1382 \ifnumberedpar@
1383 \ifledRcol
1384 \xright@appenditem{\noexpand\l@dlsnote{#1}}{%
1385 \to\inserts@listR
1386 \global\advance\insert@countR \@ne
1387 \else
1388 \xright@appenditem{\noexpand\l@dlsnote{#1}}{%
1389 \to\inserts@list
1390 \global\advance\insert@count \@ne
1391 \fi
1392 \fi\ignorespaces}
1393 \renewcommand*{\l@drsnote}[1]{%
1394 \ifnumberedpar@
1395 \ifledRcol
1396 \xright@appenditem{\noexpand\l@drsnote{#1}}{%

```

```

1397             \to\inserts@listR
1398     \global\advance\insert@countR \@ne
1399     \else
1400     \xright@appenditem{\noexpand\vl@drsnote{#1}}%
1401             \to\inserts@list
1402     \global\advance\insert@count \@ne
1403     \fi
1404     \fi\ignorespaces}
1405 \renewcommand*{\l@dcsnote}[1]{%
1406     \ifnumberedpar@
1407     \ifledRcol
1408     \xright@appenditem{\noexpand\vl@dcsnote{#1}}%
1409             \to\inserts@listR
1410     \global\advance\insert@countR \@ne
1411     \else
1412     \xright@appenditem{\noexpand\vl@dcsnote{#1}}%
1413             \to\inserts@list
1414     \global\advance\insert@count \@ne
1415     \fi
1416     \fi\ignorespaces}
1417

```

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

```

1418 \newcommand*{\affixside@noteR}{%
1419     \gdef\@templ@d{}%
1420     \ifx\@templ@d\l@dcsnote@text \else
1421     \if@twocolumn
1422     \if@firstcolumn
1423     \setl@dlp@rbox{\l@dcsnote@text}%
1424     \else
1425     \setl@drp@rbox{\l@dcsnote@text}%
1426     \fi
1427     \else
1428     \@l@tempcntb=\sidenote@marginR
1429     \ifnum\@l@tempcntb>\@ne
1430     \advance\@l@tempcntb by\page@num
1431     \fi
1432     \ifodd\@l@tempcntb
1433     \setl@drp@rbox{\l@dcsnote@text}%
1434     \else
1435     \setl@dlp@rbox{\l@dcsnote@text}%
1436     \fi
1437     \fi
1438     \fi}
1439

```

## 18 Familiar footnotes

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

```

1440 \renewcommand{\l@dbfnote}[1]{%
1441   \ifnumberedpar@
1442     \ifledRcol
1443       \xright@appenditem{\noexpand\vl@dbfnote{#{1}}{\@thefnmark}}%
1444         \to\inserts@listR
1445       \global\advance\insert@countR \@ne
1446     \else
1447       \xright@appenditem{\noexpand\l@dbfnote{#{1}}{\@thefnmark}}%
1448         \to\inserts@list
1449       \global\advance\insert@count \@ne
1450     \fi
1451   \fi\ignorespaces}
1452
```

`\normalbfnoteX`

```

1453 \renewcommand{\normalbfnoteX}[2]{%
1454   \ifnumberedpar@
1455     \ifledRcol
1456       \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}{\@nameuse{thefootnote#1}}}%
1457         \to\inserts@listR
1458       \global\advance\insert@countR \@ne
1459     \else
1460       \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}{\@nameuse{thefootnote#1}}}%
1461         \to\inserts@list
1462       \global\advance\insert@count \@ne
1463     \fi
1464   \fi\ignorespaces}
1465
```

## 19 Verse

The `\manageparhangingsymbol` command is made to insert the hanging symbol (like in the french typography).

```

1466
1467 \newcommand{\manageparhangingsymbol}{%
1468   \setcounter{hbox}{0}%
1469   \everyhbox{%
1470     \ifnum \value{hbox}=-2%
1471       \hangingsymbol%
1472     \fi%
1473     \addtocounter{hbox}{-1}}
1474 %\end{macrocode}
1475 % Before we can define the main stanza macros we need to be able to save
1476 % and reset

```

```

1477 % the category code for \&. To save the current value we use
1478 % \verb+\next+ from the \verb+\loop+ macro.
1479 % \begin{macrocode}
1480 \chardef\next=\catcode'\&
1481 \catcode'\&=\active
1482

```

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

```

1483 \newenvironment{astanza}{%
1484   \startstanzahook
1485   \catcode'\&=\active
1486   \global\stanza@count\@ne
1487   \ifnum\usernamecount{sza@00}=\z@
1488     \let\stanza@hang\relax
1489     \let\endlock\relax
1490   \else
1491   %%% \interlinepenalty\@M % this screws things up, but I don't know why
1492     \rightskip\z@ plus 1fil\relax
1493   \fi
1494   \ifnum\usernamecount{szp@00}=\z@
1495     \let\sza@penalty\relax
1496   \fi
1497   \def&{%
1498     \endlock\mbox{}%
1499     \sza@penalty
1500     \global\advance\stanza@count\@ne
1501     \@astanza@line}%
1502   \def\&{%
1503     \endlock\mbox{}
1504     \pend
1505     \endstanzaextra}%
1506   \pstart
1507   \@astanza@line
1508 }{}
1509

```

**\@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.

```

1510 \newcommand*{\@astanza@line}{%
1511   \parindent=\csname sza@\number\stanza@count @\endcsname\stanzaindentbase
1512   \par
1513   \stanza@hang%\mbox{}%
1514   \ignorespaces}
1515

```

Lastly reset the modified category codes.

```

1516 \catcode'\&=\next
1517

```

## 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 the regular box macros, but including the string ‘name’.

```
\unhnamebox 1518 \providecommand*\newnamebox}[1]{%
\unvnamebox 1519 \expandafter\newbox\csname #1\endcsname}
\namebox 1520 \providecommand*\setnamebox}[1]{%
1521 \expandafter\setbox\csname #1\endcsname}
1522 \providecommand*\unhnamebox}[1]{%
1523 \expandafter\unhbox\csname #1\endcsname}
1524 \providecommand*\unvnamebox}[1]{%
1525 \expandafter\unvbox\csname #1\endcsname}
1526 \providecommand*\namebox}[1]{%
1527 \csname #1\endcsname}
1528
```

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

```
\usenamecount 1529 \providecommand*\newnamecount}[1]{%
1530 \expandafter\newcount\csname #1\endcsname}
1531 \providecommand*\usenamecount}[1]{%
1532 \csname #1\endcsname}
1533
```

## 21 Counts and boxes for parallel texts

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

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

```
1534 \newcount\l@dc@maxchunks
1535 \newcommand*\maxchunks}[1]{\l@dc@maxchunks=#1}
1536 \maxchunks{10}
1537
```

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

```
\l@dnumpstartsR 1538 \newcount\l@dnumpstartsR
1539
```

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

```
\l@pscR 1540 \newcount\l@dpscL
1541 \newcount\l@dpscR
1542
```

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

```

1543 \newcommand*{\l@dssetupprawboxes}{%
1544   \@l@dttempcntb=\l@dc@maxchunks
1545   \loop\ifnum\@l@dttempcntb>\z@
1546     \newnamebox{\l@dLcolrawbox\the\@l@dttempcntb}
1547     \newnamebox{\l@dRcolrawbox\the\@l@dttempcntb}
1548     \advance\@l@dttempcntb \m@ne
1549   \repeat}
1550
```

`\l@dssetupmaxlinecounts` 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@dssetupmaxlinecounts` creates `\maxchunks` new counts called `\l@dmaxlinesinpar1`, etc., and `\l@dzeromaxlinecounts` zeroes all of them.

```

1551 \newcommand*{\l@dssetupmaxlinecounts}{%
1552   \@l@dttempcntb=\l@dc@maxchunks
1553   \loop\ifnum\@l@dttempcntb>\z@
1554     \newnamecount{\l@dmaxlinesinpar\the\@l@dttempcntb}
1555     \advance\@l@dttempcntb \m@ne
1556   \repeat}
1557 \newcommand*{\l@dzeromaxlinecounts}{%
1558   \begingroup
1559   \@l@dttempcntb=\l@dc@maxchunks
1560   \loop\ifnum\@l@dttempcntb>\z@
1561     \global\usenamecount{\l@dmaxlinesinpar\the\@l@dttempcntb}=\z@
1562     \advance\@l@dttempcntb \m@ne
1563   \repeat
1564   \endgroup}
1565
```

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

```

1566 \AtBeginDocument{%
1567   \l@dssetupprawboxes
1568   \l@dssetupmaxlinecounts
1569   \l@dzeromaxlinecounts
1570   \l@dnumstartsL=\z@
1571   \l@dnumstartsR=\z@
1572   \l@dpscL=\z@
1573   \l@dpscR=\z@}
1574
```

## 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 1575 \newif\ifl@dusedbabel
\l@dusedbabeltrue 1576 \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 1577 \newif\ifl@dsamelang
1578 \l@dsamelangtrue

\l@dchecklang 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.
1579 \newcommand*{\l@dchecklang}{%
1580 \l@dsamelangfalse
1581 \edef\@tempa{\theledlanguageL}\edef\@temp{\theledlanguageR}%
1582 \ifx\@tempa\@tempb
1583 \l@dsamelangtrue
1584 \fi}
1585

\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 \.
1586 \newcommand*{\l@dbbl@set@language}[1]{%
1587 \edef\language{#1}%
1588 \select@language{\language}%
1589 \if@filesw
1590 \protected@write\auxout{}{\string\select@language{\language}}%
1591 \addtocontents{toc}{\string\select@language{\language}}%
1592 \addtocontents{lof}{\string\select@language{\language}}%
1593 \addtocontents{lot}{\string\select@language{\language}}%
1594 \fi}
1595

```

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 1596 \providecommand{\selectlanguage}[1]{%
1597 \newcommand*{\l@duselanguage}[1]{%
1598 \gdef\theledlanguageL{}
1599 \gdef\theledlanguageR{}
1600
```

Now do the babel fix or polyglossia, if necessary.

```
1601 \AtBeginDocument{%
1602 \ifundefined{xpg@main@language}{%
1603 \ifundefined{bbl@main@language}{%
```

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

```
1604 \l@dusedbabelfalse
1605 \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.

```
1606 \l@dusedbabeltrue
1607 \let\l@doldselectlanguage\selectlanguage
1608 \let\l@doldbbl@set@language\bbl@set@language
1609 \let\bbl@set@language\l@dbbl@set@language
1610 \renewcommand{\selectlanguage}[1]{%
1611 \l@doldselectlanguage{#1}%
1612 \ifledRcol \gdef\theledlanguageR{#1}%
1613 \else \gdef\theledlanguageL{#1}%
1614 \fi}
```

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

```
1615 \renewcommand*{\l@duselanguage}[1]{%
1616 \l@doldselectlanguage{#1}}
```

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

```
1617 \gdef\theledlanguageL{\bbl@main@language}%
1618 \gdef\theledlanguageR{\bbl@main@language}%
1619 }%
1620 }
```

If on Polyglossia

```
1621 { \apptocmd{\xpg@set@language}{%
1622 \ifledRcol \gdef\theledlanguageR{#1}%
1623 \else \gdef\theledlanguageL{#1}%
1624 \fi}%
1625 \let\l@duselanguage\xpg@set@language
1626 \gdef\theledlanguageL{\xpg@main@language}%
1627 \gdef\theledlanguageR{\xpg@main@language}%
1628 % \end{macrocode}
```

```

1629 % That's it.
1630 %     \begin{macrocode}
1631 }}

```

## 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.

```

1632 \newcommand*{\Columns}{%
1633   \ifnum\l@dnumstartsL=\l@dnumstartsR\else
1634     \led@err@BadLeftRightPstarts{\the\l@dnumstartsL}{\the\l@dnumstartsR}%
1635   \fi

```

Start a group and zero counters, etc.

```

1636   \begingroup
1637     \l@dzeropenalties
1638     \endgraf\global\l@num@lines=\prevgraf
1639     \global\l@num@linesR=\prevgraf
1640     \global\l@par@line=\z@
1641     \global\l@par@lineR=\z@
1642     \global\l@dpscL=\z@
1643     \global\l@dpscR=\z@

```

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

```

1644     \check@pstarts
1645     \loop\if@pstarts

```

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

```

1646     \global\advance\l@dpscL \@ne
1647     \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

```

1648     \checkraw@text
1649     \l@dchecklang
1650 {     \loop\ifaraw@text

```

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

```

1651         \ifl@dsamelang
1652         \do@lineL
1653         \do@lineR
1654     \else
1655         \l@duselanguage{\theledlanguageL}%
1656         \do@lineL
1657         \l@duselanguage{\theledlanguageR}%
1658         \do@lineR

```

```

1659      \fi
1660      \hb@xt@ \hsize{%
1661      \hfill \unhbox\l@dleftbox
1662      \hfill \columnseparator \hfill
1663      \unhbox\l@drightbox
1664      }%
1665      \checkraw@text
1666      \repeat}

```

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

```

1667      \@writelinesinparL
1668      \@writelinesinparR
1669      \check@pstarts
1670      \repeat

```

Having output all chunks, make sure all notes have been output, then zero counts ready for the next set of texts. The boolean tests for stanza are switched to false.

```

1671      \flush@notes
1672      \flush@notesR
1673      \endgroup

1674      \global\l@dpscL=\z@
1675      \global\l@dpscR=\z@
1676      \global\l@dnumstartsL=\z@
1677      \global\l@dnumstartsR=\z@
1678      \ignorespaces
1679      \global\instanzaLfalse
1680      \global\instanzaRfalse}
1681

```

`\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).

```

1682 \newcommand*{\columnseparator}{%
1683   \smash{\rule[-0.2\baselineskip]{\columnrulewidth}{1.05\baselineskip}}}
1684 \newdimen\columnrulewidth
1685   \columnrulewidth=\z@
1686

```

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

```

\@pstartstrue 1687 \newif\if@pstarts
\@pstartsfalse 1688 \newcommand*{\check@pstarts}{%
\check@pstarts 1689   \@pstartsfalse
1690   \ifnum\l@dnumstartsL>\l@dpscL
1691     \@pstartstrue
1692   \else
1693     \ifnum\l@dnumstartsR>\l@dpscR
1694       \@pstartstrue

```

```

1695     \fi
1696     \fi
1697 }
1698

```

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

```

\checkraw@text 1699 \newif\ifaraw@text
1700     \araw@textfalse
1701 \newcommand*{\checkraw@text}{%
1702     \araw@textfalse
1703     \ifvbox\namebox{1@dLcolrawbox\the\1@dpscL}
1704         \araw@texttrue
1705     \else
1706         \ifvbox\namebox{1@dRcolrawbox\the\1@dpscR}
1707             \araw@texttrue
1708         \fi
1709     \fi
1710 }
1711

```

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

```

1712 \newcommand*{\@writelinesinparL}{%
1713     \edef\next{%
1714         \write\linenum@out{\string\@pend[\the\@donereallinesL]}}%
1715     \next
1716     \global\@donereallinesL \z@}
1717 \newcommand*{\@writelinesinparR}{%
1718     \edef\next{%
1719         \write\linenum@outR{\string\@pendR[\the\@donereallinesR]}}%
1720     \next
1721     \global\@donereallinesR \z@}
1722

```

## 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 1723 \newcount\numpagelinesL
1724 \newcount\numpagelinesR
1725 \newcount\l@dminpagelines
1726

```

`\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.

```

1727 \newcommand*{\Pages}{%
1728   \typeout{}
1729   \typeout{***** PAGES *****}
1730   \ifnum\l@dnumpstartsL=\l@dnumpstartsR\else
1731     \led@err@BadLeftRightPstarts{\the\l@dnumpstartsL}{\the\l@dnumpstartsR}%
1732   \fi

```

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

```

1733 \cleartol@devenpage
1734 \begingroup
1735   \l@dzeropenalties
1736   \endgraf\global\num@lines=\prevgraf
1737   \global\num@linesR=\prevgraf
1738   \global\par@line=\z@
1739   \global\par@lineR=\z@
1740   \global\l@dpscL=\z@
1741   \global\l@dpscR=\z@
1742   \writtenlinesLfalse
1743   \writtenlinesRfalse

```

Check if there are chunks to be processed.

```

1744   \check@pstarts
1745   \loop\if@pstarts

```

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

```

1746     \global\advance\l@dpscL \@ne
1747     \global\advance\l@dpscR \@ne

```

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

```

1748     \getlinesfromparlistL
1749     \getlinesfromparlistR
1750     \l@dcalc@maxoftwo{\@cs@linesinparL}{\@cs@linesinparR}%
1751     {\usernamecount{\l@dmxlinesinpar\the\l@dpscL}}%
1752     \check@pstarts
1753   \repeat

```

Zero the counts again, ready for the next bit.

```

1754   \global\l@dpscL=\z@
1755   \global\l@dpscR=\z@

```

Get the number of lines on the first pair of pages and store the mininum in `\l@dminpagelines`.

```

1756   \getlinesfrompagelistL
1757   \getlinesfrompagelistR
1758   \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
1759   {\l@dminpagelines}%

```

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.

```
1760 \check@pstarts
1761 \if@pstarts
```

Increment the chunk counts to get the first pair.

```
1762 \global\advance\l@dpscL \@ne
1763 \global\advance\l@dpscR \@ne
```

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

```
1764 \global\@donereallinesL=\z@
1765 \global\@donetotallinesL=\z@
1766 \global\@donereallinesR=\z@
1767 \global\@donetotallinesR=\z@
```

Start a loop over the boxes (chunks).

```
1768 \checkraw@text
1769 % \begingroup
1770 { \loop\ifaraw@text
```

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

```
1771 \checkpageL
1772 \l@duselanguage{\theledlanguageL}%
1773 %%% \begingroup
1774 { \loop\ifl@dsamepage
```

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

```
1775 \do@lineL
1776 \advance\numpagelinesL \@ne
1777 \ifshiftedverses
1778 \addtocounter{hbox}{-1}
1779 \ifdim\ht\l@dleftbox>0pt\hb@xt@ \hsize{\ledstrutL\unhbox\l@dleftbox}\fi%
1780 \else
1781 \hb@xt@ \hsize{\ledstrutL\unhbox\l@dleftbox}%
1782 \fi
```

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.

```
1783 \get@nextboxL
1784 \checkpageL
1785 \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.

```
1786 \ifl@dpagfull
```

```

1787         \@writelinesonpageL{\the\numpagelinesL}%
1788     \else
1789         \@writelinesonpageL{1000}%
1790     \fi

```

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

```

1791         \numpagelinesL \z@
1792         \clearl@dleftpage }%

```

Now do the same for the right text.

```

1793     \checkpageR
1794     \l@duselanguage{\theledlanguageR}%
1795 {
1796     \loop\ifl@dsamepage
1797     \do@lineR
1798     \advance\numpagelinesR \@ne
1799     \ifshiftedverses
1800 \addtocounter{hbox}{-1}
1801     \ifdim\ht\l@drightbox>0pt\hb@xt@ \hsize{\ledstrutR\unhbox\l@drightbox}\fi%
1802     \else
1803     \hb@xt@ \hsize{\ledstrutR\unhbox\l@drightbox}%
1804     \fi
1805     \get@nextboxR
1806     \checkpageR
1807     \repeat
1808     \ifl@dpagfull
1809     \@writelinesonpageR{\the\numpagelinesR}%
1810     \else
1811     \@writelinesonpageR{1000}%
1812     \fi
1813     \numpagelinesR=\z@

```

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

```

1813         \clearl@drightpage}

```

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

```

1814     \checkraw@text
1815     \ifaraw@text
1816     \getlinesfrompagelistL
1817     \getlinesfrompagelistR
1818     \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
1819     {\l@dminpagelines}%
1820     \fi
1821     \repeat}

```

We have now output the text from all the chunks.

```

1822     \fi

```

Make sure that there are no inserts hanging around.

```

1823     \flush@notes

```



```

1824 \flush@notesR
1825 \endgroup

```

Zero counts ready for the next set of left/right text chunks. The boolean tests for stanza are switched to false.

```

1826 \global\l@dpscL=\z@
1827 \global\l@dpscR=\z@
1828 \global\l@dnumpestartsL=\z@
1829 \global\l@dnumpestartsR=\z@
1830 \global\instanzaLfalse
1831 \global\instanzaRfalse
1832 \ignorespaces}
1833

```

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

```

\ledstrutR 1834 \newcommand*{\ledstrutL}{\strut}
1835 \newcommand*{\ledstrutR}{\strut}
1836

```

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

```

1837 \providecommand{\cleartoevenpage}[1][\@empty]{%
1838 \clearpage
1839 \ifodd\c@page\hbox{ }#1\clearpage\fi}
1840 \newcommand*{\cleartol@devenpage}{%
1841 \ifdim\pagetotal<\topskip% on an empty page
1842 \else
1843 \clearpage
1844 \fi
1845 \ifodd\c@page\hbox{ }\clearpage\fi}
1846 \newcommand*{\clearl@dleftpage}{%
1847 \clearpage
1848 \ifodd\c@page\else
1849 \led@err@LeftOnRightPage
1850 \hbox{ }}%
1851 \cleardoublepage
1852 \fi}
1853 \newcommand*{\clearl@drightpage}{%
1854 \clearpage
1855 \ifodd\c@page
1856 \led@err@RightOnLeftPage
1857 \hbox{ }}%
1858 \cleartoevenpage
1859 \fi}
1860

```

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

```
\@cs@linesinparR 1861 \newcommand*\getlinesfromparlistL{%
1862   \ifx\linesinpar@listL\empty
1863     \gdef\@cs@linesinparL{0}%
1864   \else
1865     \gl@p\linesinpar@listL\to\@cs@linesinparL
1866   \fi}
1867 \newcommand*\getlinesfromparlistR{%
1868   \ifx\linesinpar@listR\empty
1869     \gdef\@cs@linesinparR{0}%
1870   \else
1871     \gl@p\linesinpar@listR\to\@cs@linesinparR
1872   \fi}
1873
```

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

```
\@cs@linesonpageR 1874 \newcommand*\getlinesfrompagelistL{%
1875   \ifx\linesonpage@listL\empty
1876     \gdef\@cs@linesonpageL{1000}%
1877   \else
1878     \gl@p\linesonpage@listL\to\@cs@linesonpageL
1879   \fi}
1880 \newcommand*\getlinesfrompagelistR{%
1881   \ifx\linesonpage@listR\empty
1882     \gdef\@cs@linesonpageR{1000}%
1883   \else
1884     \gl@p\linesonpage@listR\to\@cs@linesonpageR
1885   \fi}
1886
```

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

```
1887 \newcommand*\@writelinesonpageL}[1]{%
1888   \edef\next{\write\linenum@out{\string\@lopL{#1}}}%
1889   \next}
1890 \newcommand*\@writelinesonpageR}[1]{%
1891   \edef\next{\write\linenum@outR{\string\@lopR{#1}}}%
1892   \next}
1893
```

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

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

```

1894 \newcommand*{\l@dcalc@maxoftwo}[3]{%
1895   \ifnum #2>#1\relax
1896     #3=#2\relax
1897   \else
1898     #3=#1\relax
1899   \fi}
1900 \newcommand*{\l@dcalc@minoftwo}[3]{%
1901   \ifnum #2<#1\relax
1902     #3=#2\relax
1903   \else
1904     #3=#1\relax
1905   \fi}
1906

```

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

```

\checkpageL 1907 \newif\ifl@dsamepage
\checkpageR 1908   \l@dsamepagetrue
1909 \newif\ifl@dpagefull
1910 \newcommand*{\checkpageL}{%
1911   \l@dpagefulltrue
1912   \l@dsamepagetrue
1913   \check@goal
1914   \ifdim\pagetotal<\ledthegoal
1915     \ifnum\numpagelinesL<\l@dminpagelines
1916       \else
1917         \l@dsamepagefalse
1918         \l@dpagefullfalse
1919       \fi
1920     \else
1921       \l@dsamepagefalse
1922       \l@dpagefulltrue
1923     \fi}
1924 \newcommand*{\checkpageR}{%
1925   \l@dpagefulltrue
1926   \l@dsamepagetrue
1927   \check@goal
1928   \ifdim\pagetotal<\ledthegoal
1929     \ifnum\numpagelinesR<\l@dminpagelines
1930       \else
1931         \l@dsamepagefalse
1932         \l@dpagefullfalse
1933       \fi
1934     \else
1935       \l@dsamepagefalse

```

```

1936     \l@dpagefulltrue
1937     \fi}
1938
\ledthegoal \ledthegoal is the amount of space allowed to taken by text and footnotes on
\goalfraction a page before a forced pagebreak. This can be controlled via \goalfraction.
\check@goal \ledthegoal is calculated via \check@goal.
1939 \newdimen\ledthegoal
1940 \ifshiftedverses
1941     \newcommand*{\goalfraction}{0.95}
1942 \else
1943     \newcommand*{\goalfraction}{0.9}
1944 \fi
1945
1946 \newcommand*{\check@goal}{%
1947     \ledthegoal=\goalfraction\pagegoal}
1948

\ifwrittenlinesL Booleans for whether line data has been written to the section file.
\ifwrittenlinesL 1949 \newif\ifwrittenlinesL
1950 \newif\ifwrittenlinesR
1951

\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.
1952 \newcommand*{\get@nextboxL}{%
1953     \ifvbox\namebox{\l@dLcolrawbox\the\l@dpscl}% box is not empty
        The current box is not empty; do nothing.
1954     \else%
        box is empty
        The box is empty; check if enough lines (real and blank) have been output.
1955     \ifnum\usernamecount{\l@dmaxlinesinpar\the\l@dpscl}>\@donetotallinesL
1956     \else
        Sufficient lines have been output.
1957     \ifwrittenlinesL
1958     \else
        Write out the number of lines done, and set the boolean so this is only done once.
1959     \@writelinesinparL
1960     \writtenlinesLtrue
1961     \fi
1962     \ifnum\l@dnumstartsL>\l@dpscl
        There are still unprocessed boxes. Recalculate the maximum number of lines
        needed, and move onto the next box (by incrementing \l@dpscl).
1963     \writtenlinesLfalse
1964     \l@dcalc@maxoftwo{\the\usernamecount{\l@dmaxlinesinpar\the\l@dpscl}}%
1965     {\the\@donetotallinesL}%
1966     {\usernamecount{\l@dmaxlinesinpar\the\l@dpscl}}%

```

```

1967      \global\@donetotallinesL \z@
1968      \global\advance\l@dpscL \@ne
1969      \fi
1970  \fi
1971 \fi}

1972 \newcommand*{\get@nextboxR}{%
1973   \ifvbox\namebox{\l@dRcolrawbox\the\l@dpscR}% box is not empty
1974   \else% box is empty
1975     \ifnum\usernamecount{\l@dmaxlinesinpar\the\l@dpscR}>\@donetotallinesR
1976     \else
1977       \ifwrittenlinesR
1978       \else
1979         \@writelinesinparR
1980         \writtenlinesRtrue
1981       \fi
1982       \ifnum\l@dnumstartsR>\l@dpscR
1983         \writtenlinesRfalse
1984         \l@dcalc@maxoftwo{\the\usernamecount{\l@dmaxlinesinpar\the\l@dpscR}}%
1985                           {\the\@donetotallinesR}%
1986                           {\usernamecount{\l@dmaxlinesinpar\the\l@dpscR}}%
1987         \global\@donetotallinesR \z@
1988         \global\advance\l@dpscR \@ne
1989       \fi
1990     \fi
1991   \fi}
1992

```

## 25 The End

i/code<sub>i</sub>

## 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,<sup>1</sup> 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,<sup>2</sup> 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,<sup>3</sup> gave his permission. So did the reverend  
 dean and canons of the chapter of St. Apostles' in Cologne, who had long<sup>4</sup> 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,<sup>5</sup> 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.<sup>6</sup>

// One additional line to show synchronization. //

---

<sup>1</sup>In 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.

<sup>2</sup>Since 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.

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

<sup>4</sup>They 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.

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

<sup>6</sup>The 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—rissime 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 cuspidē 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`.

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

<sup>2</sup>Muses

<sup>3</sup>Ceres was the Roman goddess of the harvest.

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

<sup>5</sup>Lines 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—rissime 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`.

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

<sup>7</sup>Muses

<sup>8</sup>Ceres was the Roman goddess of the harvest.

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

<sup>10</sup>Lines 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.

---

```

1993 (*villon)
1994 %% villon.tex Example parallel columns
1995 \documentclass{article}
1996 \addtolength{\textheight}{-10\baselineskip}
1997 \usepackage{ledmac,ledpar}
1998 %% Use r instead of R to flag right text line numbers
1999 \renewcommand{\Rlineflag}{r}
2000 %% Use the flag in the notes
2001 \let\oldBfootfmt\Bfootfmt
2002 \renewcommand{\Bfootfmt}[3]{%
2003   \let\printlines\printlinesR
2004   \oldBfootfmt{#1}{#2}{#3}}
2005 \begin{document}
2006
2007 I thought that limericks were peculiarly English, but this appears not
2008 to be the case. As with most limericks this one is by Anonymous.
2009
2010 \vspace*{\baselineskip}
2011
2012 \begin{pairs}
2013 %% no indentation
2014 \setstanzaindents{0,0,0,0,0,0,0,0,0}
2015 %% no number flag
2016 \renewcommand{\Rlineflag}{}
2017 %% draw a rule and widen the columns
2018 \setlength{\columnrulewidth}{0.4pt}
2019 \setlength{\Lcolwidth}{0.46\textwidth}
2020 \setlength{\Rcolwidth}{\Lcolwidth}
2021
2022 \begin{Leftside}
2023 %% set left text line numbering sequence
2024 \firstlinenum{2}
2025 \linenumincrement{2}
2026 \linenummargin{left}
2027 \beginnumbering
2028 \stanza
2029 Il y avait un jeune homme de Dijon, &
2030 Qui n'avait que peu de religion. &
2031 Il dit: 'Quant \'{a} moi, &
2032 Je d'\{e\}teste tous les trois, &
2033 Le P\{e\}re, et le Fils, et le Pigeon.' \&
2034 \endnumbering
2035 \end{Leftside}

```

```

2036
2037 \begin{Rightside}
2038 %% different right text line numbering sequence
2039 \firstlinenum{1}
2040 \linenumincrement{2}
2041 \linenummargin{right}
2042 \beginnumbering
2043 \stanza
2044 There was a young man of Dijon, &
2045 Who had only a little religion, &
2046 He said: 'As for me, &
2047 I detest all the three, &
2048 The Father, the Son, and the Pigeon.' \&
2049 \endnumbering
2050 \end{Rightside}
2051
2052 \Columns
2053 \end{pairs}
2054
2055 \vspace*{\baselineskip}
2056
2057 The following is verse \textsc{lxixiii} of Fran\c{c}ois Villon's
2058 \textit{Le Testament} (The Testament), composed in 1461.
2059
2060 %% Allow for hanging indentation for long lines
2061 \setstanzaindents{1,0,0,0,0,0,0,0}
2062 %% Columns wider than the default
2063 \setlength{\Lcolwidth}{0.46\textwidth}
2064 \setlength{\Rcolwidth}{\Lcolwidth}
2065 \vspace*{\baselineskip}
2066
2067 \begin{pairs}
2068 \begin{Leftside}
2069 \firstlinenum{2}
2070 \linenumincrement{2}
2071 \linenummargin{left}
2072 \beginnumbering
2073 \stanza
2074 Dieu mercy et Tacque Thibault, &
2075 Qui tant d'eaue froid m'a fait boire, &
2076 Mis en bas lieu, non pas en hault, &
2077 Mengier d'angoisse maints \edtext{poire}{\lemma{poire d'angoisse}}%
2078 \Afootnote{This has a triple meaning: literally it is the fruit of the
2079 choke pear,
2080 figuratively it means 'bitter fruit', and it also refers to a torture
2081 instrument.}}, &
2082 Enferr'\{e} \ldots Quant j'en ay memoire, &
2083 Je Prie pour luy \edtext{\textit{et reliqua}}{\Afootnote{and so on}}, &
2084 Que Dieu luy doint, et voire, voire! &
2085 Ce que je pense \ldots \textit{et cetera}. \&

```

```

2086 \endnumbering
2087 \end{Leftside}
2088
2089 \begin{Rightside}
2090 \firstlinenum{2}
2091 \linenumincrement{2}
2092 \linenummargin{right}
2093 \beginnumbering
2094 \stanza
2095 Thanks to God --- and to \edtext{Tacque Thibaud}{%
2096   \Bfootnote{A favourite of Jean, Duc de Berry and loathed for his exactions
2097     and debauchery. Villon uses his name as an insulting nickname for
2098     Thibaud d'Auxigny, the Bishop of Orl\{'e}ans.}} &
2099   Who made me drink so much \edtext{cold water}{%
2100     \Bfootnote{Can either refer to the normal prison diet of bread and
2101       water or to a common medieval torture which involved forced drinking
2102       of cold water.}}, &
2103   Put me underground instead of higher up &
2104   And made me eat such bitter fruit, &
2105   In chains \ldots When I think of this, &
2106   I pray for him---\textit{et reliqua;} &
2107   May God grant him (yes, by God) &
2108   What I think \ldots \textit{et cetera}. \&
2109 \endnumbering
2110 \end{Rightside}
2111
2112 \Columns
2113 \end{pairs}
2114
2115 \vspace*{\baselineskip}
2116
2117   The translation and notes are by Anthony Bonner,
2118 \textit{The Complete Works of Fran\c{c}ois Villon}, published by
2119 Bantam Books in 1960.
2120
2121 \end{document}
2122
2123 \</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.

---

```

2124 (*djd17nov)
2125 %%% This is djdl7nov.tex, a sample critical text edition
2126 %%% written in LaTeX2e with the ledmac and ledpar packages.
2127 %%% (c) 2003--2004 by Dr. Dirk-Jan Dekker,

```



```

2128 %% Radboud University, Nijmegen (The Netherlands)
2129 %% (PRW) Modified slightly by PRW to fit the ledpar manual
2130
2131 \documentclass[10pt, letterpaper, twoside]{article}
2132 \usepackage[latin,english]{babel}
2133 \usepackage{makeidx}
2134 \usepackage{ledmac,ledpar}
2135 \lineation{section}
2136 \linenummargin{inner}
2137 \sidenotemargin{outer}
2138
2139 \makeindex
2140
2141 \renewcommand{\notenumfont}{\footnotesize}
2142 \newcommand{\notetextfont}{\footnotesize}
2143
2144 %\let\Afootnoterule=\relax
2145 \let\Bfootnoterule=\relax
2146 \let\Cfootnoterule=\relax
2147
2148 \addtolength{\skip\Afootins}{1.5mm}
2149 %\addtolength{\skip\Bfootins}{1.5mm}
2150 %\addtolength{\skip\Cfootins}{1.5mm}
2151
2152 \makeatletter
2153
2154 \renewcommand*{\para@vfootnote}[2]{%
2155   \insert\csname #1footins\endcsname
2156   \bgroup
2157     \notefontsetup
2158     \interlinepenalty=\interfootnotelinepenalty
2159     \floatingpenalty=\@MM
2160     \splittopskip=\ht\strutbox \splitmaxdepth=\dp\strutbox
2161     \leftskip=\z@skip \rightskip=\z@skip
2162     \l@dparsfootspec #2\ledplinenumtrue% new from here
2163     \ifnum\@nameuse{previous@#1@number}=\l@dparsestartline\relax
2164       \ledplinenumfalse
2165     \fi
2166     \ifnum\previous@page=\l@dparsestartpage\relax
2167     \else \ledplinenumtrue \fi
2168     \ifnum\l@dparsestartline=\l@dparseendline\relax
2169     \else \ledplinenumtrue \fi
2170     \expandafter\xdef\csname previous@#1@number\endcsname{\l@dparsestartline}%
2171     \xdef\previous@page{\l@dparsestartpage}% to here
2172     \setbox0=\vbox{\hsize=\maxdimen
2173       \noindent\csname #1footfmt\endcsname#2}%
2174     \setbox0=\hbox{\unvbox0}%
2175     \dp0=0pt
2176     \ht0=\csname #1footfudgefactor\endcsname\wd0
2177     \box0

```

```

2178     \penalty0
2179   \egroup
2180 }
2181
2182 \newcommand*{\previous@A@number}{-1}
2183 \newcommand*{\previous@B@number}{-1}
2184 \newcommand*{\previous@C@number}{-1}
2185 \newcommand*{\previous@page}{-1}
2186
2187 \newcommand{\abb}[1]{#1%
2188     \let\rbracket\nobrak\relax}
2189 \newcommand{\nobrak}{\textnormal{}}
2190 \newcommand{\morenoexpands}{%
2191     \let\abb=0%
2192 }
2193
2194 \newcommand{\Aparafootfmt}[3]{%
2195     \ledsetnormalparstuff
2196     \scriptsize
2197     \notenumfont\printlines#1\enspace
2198 %   \lemmafont#1/#2\enskip
2199     \notetextfont
2200     #3\penalty-10\hskip 1em plus 4em minus.4em\relax}
2201
2202 \newcommand{\Bparafootfmt}[3]{%
2203     \ledsetnormalparstuff
2204     \scriptsize
2205     \notenumfont\printlines#1/%
2206     \ifledplinenum
2207     \enspace
2208     \else
2209     {\hskip 0em plus 0em minus .3em}%
2210     \fi
2211     \select@lemmafont#1/#2\rbracket\enskip
2212     \notetextfont
2213     #3\penalty-10\hskip 1em plus 4em minus.4em\relax }
2214
2215 \newcommand{\Cparafootfmt}[3]{%
2216     \ledsetnormalparstuff
2217     \scriptsize
2218     \notenumfont\printlines#1\enspace
2219 %   \lemmafont#1/#2\enskip
2220     \notetextfont
2221     #3\penalty-10\hskip 1em plus 4em minus.4em\relax}
2222
2223 \makeatother
2224
2225 \footparagraph{A}
2226 \footparagraph{B}
2227 \footparagraph{C}

```

```

2228
2229 \let\Afootfmt=\Aparafootfmt
2230 \let\Bfootfmt=\Bparafootfmt
2231 \let\Cfootfmt=\Cparafootfmt
2232
2233 \renewcommand*{\Rlineflag}{}
2234
2235 \emergencystretch40pt
2236
2237 \author{Guillelmus de Berchen}
2238 \title{Chronicon Geldriae}
2239 \date{}
2240 \hyphenation{archi-epi-sco-po Huns-dis-brug li-be-ra No-vi-ma-gen-si}
2241 \begin{document}
2242 \begin{pages}
2243 \begin{Leftside}
2244 \beginnumbering\pstart
2245 \selectlanguage{latin}
2246 \section{De ecclesia S. Stephani Novimagensi}
2247
2248 \noindent\setline{1}
2249 Nobilis itaque comes Otto\protect\edindex{Otto II of Guelders}
2250 imperio et dominio Novimagensi sibi, ut praeferetur, impignoratis
2251 et commissis
2252 \edtext{proinde}{\Bfootnote{primum D}} praeesse cupiens, anno
2253 \textsc{liiii} superius descripto, mense
2254 \u\edtext{}{\Afootnote{p.\ 227~R}}nio, una cum iudice, scabinis ceterisque
2255 civibus civitatis Novimagensis, pro ipsius et inhabitantium in ea
2256 necessitate,\edtext{}{\Afootnote{p.\ 97~N}} commodo et utilitate,
2257 ut \edtext{ecclesia eius}{\Bfootnote{ecclesia D: eius eius H}} parochialis
2258 \edtext{\abb{extra civitatem}}{\Bfootnote{\textit{om.}~H}} sita
2259 destrueretur et \edtext{infra}{\Bfootnote{intra D}} muros
2260 \edtext{transfer\edtext{}{\Afootnote{p.\ 129~D}}retur}%
2261 {\Bfootnote{transferreretur NH}}
2262 ac de novo construeretur,
2263 \edtext{a reverendo patre domino
2264 Conrado\protect\edindex{Conrad of Hochstaden} de
2265 \edtext{Hofsteden}{\Bfootnote{Hoffstede D: Hoffsteden H}}, archiepiscopo
2266 \edtext{Coloniensi}{\Bfootnote{Colononiensi H}}, licentiam}%
2267 {\Cfootnote{William is confusing two charters that are five years
2268 apart. Permission from St.\ Apostles' Church in Cologne had been
2269 obtained as early as 1249. Cf.\
2270 Sloet\protect\index{Sloet van de Beele, L.A.J.W.},
2271 \textit{Oorkondenboek} nr.\ 707 (14 November 1249):
2272 ‘‘\ldots}nos devotionis tue precibus annuentes, ut ipsam ecclesiam
2273 faciens demoliri transferas in locum alium competentem, tibi
2274 auctoritate presentium indulgemus\ldots’’}}, et a venerabilibus
2275 \edtext{dominis}{\Bfootnote{viris H}} decano et capitulo sanctorum
2276 Apostolorum\protect\edindex{St. Apostles' (Cologne)}
2277 \edtext{Coloniensi}{\Bfootnote{Coloniae H}}, ipsius ecclesiae ab

```

```

2278 antiquo veris et pacificis patronis, consensum, citra tamen
2279 praeiudicium, damnum aut gravamen \edtext{iurium}{\Bfootnote{virium D}}
2280 et bonorum eorundem, impetravit.
2281 \pend
2282
2283 \pstart
2284 \edtext{Et exinde \edtext{liberum}{\Bfootnote{librum H}}}
2285 locum eiusdem civitatis
2286 \edtext{qui}{\Bfootnote{quae D}} dicitur
2287 \edtext{Hundisburg}{\Bfootnote{Hundisburch D: Hundisbrug HMN:
2288 Hunsdisbrug R}}\protect\edindex{Hundisburg},
2289 de praelibati Wilhelmi\protect\edindex{William II of Holland} Romanorum
2290 \edtext{regis}{\Bfootnote{imperatoris D}}, ipsius fundi
2291 do\edtext{}{\Afootnote{f.\ 72v~M}}mini, consensu, ad aedificandum
2292 \edtext{\abb{et consecrandum}}{\Bfootnote{\textit{om.}\ H}}
2293 ecclesi\edtext{}{\Afootnote{p.\ 228~R}}am et coemeterium,
2294 \edtext{eisdem}{\Bfootnote{eiusdem D}} decano et capitulo de expresso
2295 eiusdem civitatis assensu libera contradiderunt voluntate, obligantes
2296 se ipsi \edtext{comes}{\Bfootnote{comites D}} et civitas
2297 \edtext{\abb{dictis}}{\Bfootnote{\textit{om.}\ H}} decano et capitulo,
2298 quod in recompensationem illius areae infra castrum et portam, quae
2299 fuit dos ecclesiae, in qua plebanus habitare solebat---quae
2300 \edtext{tunc}{\Bfootnote{nunc H}} per novum fossatum civitatis est
2301 destructa---aliam aream competentem et ecclesiae novae,
2302 \edtext{ut praeferitur, aedificandae}{%
2303 \lemma{\abb{ut\ldots aedificandae}}{\Bfootnote{\textit{om.}\ H}} satis
2304 \edtext{contiguam}{\Bfootnote{contiguum M}}, ipsi plebano darent et
2305 assignarent.}{\Cfootnote{Cf.\ Sloet, \textit{Oorkondenboek} nr.\ 762
2306 (June 1254)}} Et desuper
2307 \edtext{\abb{apud}}{\Bfootnote{\textit{om.}\ H}} dictam ecclesiam
2308 sanctorum Apostolorum \edtext{est}{\Bfootnote{et H}}
2309 \edtext{littera}{\Bfootnote{litteram H}} sigillis ipsorum
2310 Ottonis\edtext{}{\Afootnote{p.\ 130~D}} comitis et civitatis
2311 \edtext{Novimagensis}{\Bfootnote{Novimagii D}}
2312 \edtext{sigillata}{\Bfootnote{sigillis communita H}}.
2313 \pend
2314
2315 \pstart
2316 // One additional line to show synchronization. //
2317 \pend
2318 \endnumbering
2319 \end{Leftside}
2320
2321 \begin{Rightside}
2322 \sidenotemargin{right}\selectlanguage{english}
2323 \beginnumbering
2324 \pstart
2325 \addtocounter{section}{-1}%
2326 \leavevmode\section{St.\ Stephen's Church in Nijmegen}
2327

```

2328 \noindent\setline{1}%  
 2329 After the noble count Otto had taken in pledge the power over  
 2330 Nijmegen,\footnote{In 1247 William II\protect\index{William II of Holland}  
 2331 (1227--1256) count of Holland needed money to fight his way to  
 2332 Aachen\protect\index{Aachen} to be crowned King of the Holy Roman  
 2333 Empire. He gave the town of Nijmegen in pledge to Otto  
 2334 II\protect\index{Otto II of Guelders} (1229--1271) count of Guelders.}  
 2335 like I have written above, he wanted to protect the town. So in June  
 2336 1254\ledsidenote{1254} he and the judge, the sheriffs and other  
 2337 citizens of Nijmegen obtained permission to demolish the parish  
 2338 church that lay outside the town walls,\footnote{Since the early  
 2339 seventh century old St.\ Stephen's church had been located close  
 2340 to the castle, at today's  
 2341 Kelfkensbos\protect\index{Kelfkensbos (Nijmegen)}} square.  
 2342 Traces of the church and the presbytery were found during excavations  
 2343 in 1998--1999.} to move it inside the walls and to rebuild it new.  
 2344 This operation was necessary and useful both for Otto himself and  
 2345 for the inhabitants of the town. The reverend father Conrad of  
 2346 Hochstaden, archbishop of  
 2347 Cologne,\footnote{Conrad of Hochstaden ({\textdag} 1261) was  
 2348 archbishop of Cologne in 1238--1261. Nijmegen belonged to the  
 2349 archdiocese of Cologne until 1559.} gave his permission. So did the  
 2350 reverend dean and canons of the chapter of St.\  
 2351 Apostles'\protect\index{St. Apostles' (Cologne)}} in Cologne, who had  
 2352 long\footnote{They probably became the patrons when the chapter was  
 2353 established in the early eleventh century. About the church and the  
 2354 chapter, see Gottfried Stracke\protect\index{Stracke, G.},  
 2355 \textit{K}\{"o}ln: \ St.\ Aposteln}, Stadtspuren -- Denkm\{"a}ler in  
 2356 K\{"o}ln, vol.\ 19, K\{"o}ln: J.\,P.\ Bachem, 1992.} been the true  
 2357 and benevolent patrons of the church---but they did not allow Otto  
 2358 to do anything without their knowledge, nor to infringe their rights,  
 2359 nor to damage their property.  
 2360 \pend  
 2361  
 2362 \pstart  
 2363 And so the count and the town voluntarily gave an open space in town  
 2364 called Hundisburg, which was owned by the aforementioned king William,  
 2365 to the dean and chapter of St.\ Apostles' in order to build and  
 2366 consecrate a church and graveyard. King William approved and the  
 2367 town of Nijmegen explicitly expressed its assent. A new ditch was dug  
 2368 on property of the church near the castle and the  
 2369 harbour,\footnote{Nowadays, the exact location of the medieval  
 2370 ditch---and of two Roman ones---can be seen in the pavement of  
 2371 Kelfkensbos\protect\index{Kelfkensbos (Nijmegen)}} square.} causing  
 2372 the demolition of the presbytery. In compensation, the count and  
 2373 citizens committed themselves to giving the parish priest another  
 2374 suitable space close enough to the new church that was about to be  
 2375 built. A letter about these transactions, with the seals of count  
 2376 Otto and the town of Nijmegen, is kept at St.\ Apostles'  
 2377 church.\footnote{The original letter is lost. A 15th century

```

2378 transcription of it is kept at the Historisches Archiv der
2379 Stadt K\{"o}ln (HASTK).}
2380 \pend
2381
2382 \pstart
2383 // One additional line to show synchronization. //
2384 \pend
2385 \endnumbering
2386 \end{Rightside}
2387 \Pages
2388 \end{pages}
2389
2390 %%%%%%%%%%%
2391 \printindex
2392 \end{document}
2393 %%%%%%%%%%%
2394
2395 \</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.

---

```

2396 (*djdpoems)
2397 %% djdpoems.tex  example parallel verses on facing pages
2398 \documentclass{article}
2399 \usepackage{ledmac,ledpar}
2400 \addtolength{\textheight}{-15\baselineskip}
2401
2402 \maxchunks{24} % default value = 10
2403 \setstanzaindents{6,0,1,0,1}
2404
2405 \newcommand{\longdash}{-----}
2406
2407 \footparagraph{A} % for left pages
2408 \footparagraph{B} % for right pages
2409 \firstlinenum{1}
2410 \linenumincrement{1}
2411
2412 \let\oldBfootfmt\Bfootfmt
2413 \renewcommand{\Bfootfmt}[3]{%
2414   \let\printlines\printlinesR
2415   \oldBfootfmt{#1}{#2}{#3}}

```

```

2416
2417 \begin{document}
2418
2419 \newcommand{\interstanza}{\pstart\centering\longdash\skipnumbering\pend}
2420
2421 \begin{pages}
2422 \begin{Leftside}
2423 \def\endstanzaextra{\interstanza}
2424 \beginnumbering
2425
2426 \stanza
2427 Arma gravi numero violentaque bella parabam &
2428 edere, materi\={a} conveniente modis. &
2429 Par erat inferior versus---risisse Cupido &
2430 dicitur atque unum surripuisse pedem. \&
2431
2432 \stanza
2433 ‘Quis tibi, saeve puer, dedit hoc in carmina iuris? &
2434 Pieridum vates, non tua turba \edtext{sumus}{\Afootnote{note lost}}. &
2435 Quid si praeripiat flavae V\{e}nus arma Minervae, &
2436 ventilet accensas flava Minerva faces? \&
2437
2438 \stanza
2439 Quis probet in silvis Cererem regnare iugosis, &
2440 lege pharetratae Virginis arva coli? &
2441 Crinibus insignem quis \edtext{acuta}{\Afootnote{acut\={a} (abl.\ abs.)}}
2442 cuspidе Phoebum &
2443 instruat, Aoniam Marte movente lyram? \&
2444 \endnumbering
2445 \end{Leftside}
2446
2447 \begin{Rightside}
2448 \def\endstanzaextra{\interstanza}
2449 \beginnumbering
2450 \firstlinenum{1}
2451 \linenumincrement{1}
2452 \setstanzaindents{6,0,1,0,1,0}
2453
2454 \stanza
2455 I was preparing to sing of weapons and violent wars, &
2456 in heavy numbers, with the subject matter suited to the verse measure. &
2457 The even lines were as long as the odd ones, but Cupid laughed, &
2458 they said, and he stole away one foot.\footnote{I.e., the even lines,
2459 which were hexameters (with six feet) became pentameters
2460 (with five feet).} \&
2461
2462 \stanza
2463 ‘O cruel boy, who gave you the right over poetry? &
2464 We poets belong to the Pierides,\footnote{Muses} we are not your folk. &
2465 \edlabel{beginparadox}What if Venus should seize away the arms of

```

```

2466 Minerva with the golden hair, &
2467 if Minerva with the golden hair should fan alight the kindled torch
2468 of love? \&
2469
2470 \stanza
2471 Who would approve of Ceres\footnote{Ceres was the Roman goddess of
2472 the harvest.} reigning on the woodland ridges, &
2473 and of land tilled under the law of the Maid with the
2474 quiver\footnote{By '\textit{Virgo}' ('Virgin') Ovid means Diana, the
2475 Roman goddess of the hunt.}? &
2476 Who would provide Phoebus with his beautiful hair with a sharp-pointed
2477 spear, &
2478 while Mars stirs the \edtext{Aonian}{\Bfootnote{Mount Parnassus,
2479 where the Muses live, is located in Aonia.}}
2480 lyre?\edlabel{endparadox}\footnote{Lines
2481 \xlineref{beginparadox}--\xlineref{endparadox} show some paradoxical
2482 situations that would occur if the gods didn't stay with their own
2483 business.} \&
2484 \endnumbering
2485 \end{Rightside}
2486
2487 \Pages
2488 \end{pages}
2489
2490 \begin{pages}
2491 \begin{Leftside}
2492 \def\endstanzaextra{\interstanza}
2493 \beginnumbering
2494
2495 \begin{astanza}
2496 Arma gravi numero violentaque bella parabam &
2497 edere, materi\={a} conveniente modis. &
2498 Par erat inferior versus---risisse Cupido &
2499 dicitur atque unum surripuisse pedem. \&
2500 \end{astanza}
2501
2502 \begin{astanza}
2503 ‘Quis tibi, saeve puer, dedit hoc in carmina iuris? &
2504 Pieridum vates, non tua turba \edtext{sumus}{\Afootnote{note lost}}. &
2505 Quid si praeripiat flavae V\ufe}nus arma Minervae, &
2506 ventilet accensas flava Minerva faces? \&
2507 \end{astanza}
2508
2509 \begin{astanza}
2510 Quis probet in silvis Cererem regnare iugosis, &
2511 lege pharetratae Virginis arva coli? &
2512 Crinibus insignem quis \edtext{acuta}{\Afootnote{acut\={a} (abl.\ abs.)}}
2513 cuspede Phoebum &
2514 instruat, Aoniam Marte movente lyram? \&
2515 \end{astanza}

```



```

2516
2517 \endnumbering
2518 \end{Leftside}
2519
2520 \begin{Rightside}
2521 \def\endstanzaextra{\interstanza}
2522 \beginnumbering
2523 \firstlinenum{1}
2524 \linenumincrement{1}
2525 \setstanzaindents{6,0,1,0,1,0}
2526
2527 \begin{astanza}
2528 I was preparing to sing of weapons and violent wars, &
2529 in heavy numbers, with the subject matter suited to the verse measure. &
2530 The even lines were as long as the odd ones, but Cupid laughed, &
2531 they said, and he stole away one foot.\footnote{I.e., the even lines,
2532 which were hexameters (with six feet) became pentameters
2533 (with five feet).} \&
2534 \end{astanza}
2535
2536 \begin{astanza}
2537 ‘‘O cruel boy, who gave you the right over poetry? &
2538 We poets belong to the Pierides,\footnote{Muses} we are not your folk. &
2539 \edlabel{beginparadox}What if Venus should seize away the arms of
2540 Minerva with the golden hair, &
2541 if Minerva with the golden hair should fan alight the kindled torch
2542 of love? \&
2543 \end{astanza}
2544
2545 \begin{astanza}
2546 Who would approve of Ceres\footnote{Ceres was the Roman goddess of the
2547 harvest.} reigning on the woodland ridges, &
2548 and of land tilled under the law of the Maid with the
2549 quiver\footnote{By ‘\textit{Virgo}’ (‘Virgin’) Ovid means Diana,
2550 the Roman goddess of the hunt.}? &
2551 Who would provide Phoebus with his beautiful hair with a sharp-pointed
2552 spear, &
2553 while Mars stirs the \edtext{Aonian}{\Bfootnote{Mount Parnassus, where
2554 the Muses live, is located in Aonia.}}
2555 lyre?\edlabel{endparadox}\footnote{Lines
2556 \xlineref{beginparadox}--\xlineref{endparadox} show some paradoxical
2557 situations that would occur if the gods didn’t stay with their
2558 own business.} \&
2559 \end{astanza}
2560
2561 \endnumbering
2562 \end{Rightside}
2563
2564 \Pages
2565 \end{pages}

```

```
2566
2567 \end{document}
2568
2569 </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>\&amp;</code> 1477, 1480, 1481, 1485, 1502, 1516, 2033, 2048, 2085, 2108, 2430, 2436, 2443, 2460, 2468, 2483, 2499, 2506, 2514, 2533, 2542, 2558	<code>\@donetotallinesR</code> ..... 857, 920, 923, 1767, 1975, 1985, 1987
<code>\@M</code> ..... 1491	<code>\@insertR</code> ..... 1149–1151, 1164–1166
<code>\@MM</code> ..... 2159	<code>\@l</code> ..... 251, 564
<code>\@adv</code> ..... 328, 595, 596	<code>\@l@dttempcnta</code> ..... . 401, 403, 405, 406, 410, 412, 414, 415, 969, 1008, 1009, 1011, 1013, 1016, 1017, 1034–1038, 1040, 1047, 1052, 1056, 1064, 1069, 1073, 1105, 1108, 1110, 1114
<code>\@afterindentfalse</code> ..... 706	<code>\@l@dttempcntb</code> ..... 144, 146, 148, 999, 1000, 1047, 1052, 1056, 1064, 1069, 1073, 1097, 1101, 1114, 1122–1124, 1126, 1371, 1373, 1375, 1428–1430, 1432, 1544–1548, 1552–1555, 1559–1562
<code>\@arabic</code> ..... 185, 186, 754, 756	<code>\@l@reg</code> ..... 300
<code>\@astanza@line</code> ..... 1501, 1507, 1510	<code>\@l@regR</code> ..... 251
<code>\@auxout</code> ..... 1333, 1343, 1590	<code>\@lab</code> ..... 520, 1327, 1336, 1357
<code>\@chapter</code> ..... 707	<code>\@lock</code> ..... 951
<code>\@cs@linesinparL</code> ..... 1750, 1861	<code>\@lockR</code> ..... 58, 273, 275, 277, 290, 435, 451, 452, 454, 455, 483, 484, 486, 936, 975, 977, 978, 980, 1061, 1078, 1080, 1082
<code>\@cs@linesinparR</code> ..... 1750, 1861	
<code>\@cs@linesonpageL</code> .. 1758, 1818, 1874	
<code>\@cs@linesonpageR</code> .. 1758, 1818, 1874	
<code>\@donereallinesL</code> ..... ..... 857, 885, 1714, 1716, 1764	
<code>\@donereallinesR</code> ..... ..... 857, 919, 1719, 1721, 1766	
<code>\@donetotallinesL</code> ..... 857, 886, 889, 1765, 1955, 1965, 1967	

`\@lopL` ..... 542, 1888  
`\@lopR` ..... 542, 1891  
`\@nameuse` ..... 1456, 1460, 2163  
`\@nobreakfalse` ..... 762, 791  
`\@nobreaktrue` ..... 760, 764, 789, 793  
`\@oldnobreak` 760, 762, 789, 791, 827, 843  
`\@pend` ..... 535, 1714  
`\@pendR` ..... 535, 1719  
`\@pstartsfalse` ..... 1687  
`\@pstartstrue` ..... 1687  
`\@ref` ..... 507, 568, 572  
`\@ref@reg` ..... 533  
`\@schapter` ..... 707  
`\@set` ..... 360, 602, 603  
`\@tag` 633, 650, 1173, 1177, 1187, 1191,  
1201, 1205, 1215, 1219, 1229,  
1233, 1244, 1248, 1258, 1262,  
1272, 1276, 1286, 1290, 1300, 1304  
`\@temp` ..... 1581  
`\@templd` ..... 1419, 1420  
`\@writelinesinparL` .. 1667, 1712, 1959  
`\@writelinesinparR` .. 1668, 1712, 1979  
`\@writelinesonpageL` . 1787, 1789, 1887  
`\@writelinesonpageR` . 1808, 1810, 1887  
`\@xloop` ..... 1162  
`\add@insertsR` ..... 912, 1137  
`\add@penaltiesL` ..... 884, 1158  
`\add@penaltiesR` ..... 918, 1158  
`\addtocontents` ..... 1591–1593  
`\addtocounter` ..... 829, 845,  
937, 953, 1473, 1778, 1799, 2325  
`\addtolength` ... 1996, 2148–2150, 2400  
`\advanceline` ..... 594, 625  
`\affixline@num` ..... 876  
`\affixline@numR` ..... 910, 1031  
`\affixside@note` ..... 879  
`\affixside@noteR` ..... 913, 1418  
`\Afootfmt` ..... 2229  
`\Afootins` ..... 2148  
`\Afootnote` ..... 1169, 2078,  
2083, 2254, 2256, 2260, 2291,  
2293, 2310, 2434, 2441, 2504, 2512  
`\Afootnoterule` ..... 2144  
`\Aparafootfmt` ..... 2194, 2229  
`\apptocmd` ..... 1621  
`\araw@textfalse` ..... 1699  
`\araw@texttrue` ..... 1699  
`astanza` (environment) ..... 9, 1483  
`\AtBeginDocument` ... 1353, 1566, 1601  
`\author` ..... 2237

## B

`\_` 2254, 2256, 2260, 2268, 2269, 2271,  
2291–2293, 2297, 2303, 2305,  
2307, 2310, 2326, 2339, 2350,  
2355, 2356, 2365, 2376, 2441, 2512  
**A**  
`\abb` ..... 2187,  
2191, 2258, 2292, 2297, 2303, 2307  
`\absline@num` ..... 394, 408, 427, 943  
`\absline@numR` ... 56, 202, 253, 256,  
259, 391, 399, 420, 439, 473,  
501, 512, 928, 961, 962, 999, 1148  
`\actionlines@list` .....  
..... 243, 246, 394, 408, 427  
`\actionlines@listR` .....  
..... 206, 221, 235, 238, 391,  
399, 420, 439, 473, 501, 1021, 1024  
`\actions@list` . 247, 395, 415, 429, 431  
`\actions@listR` .....  
. 206, 222, 239, 392, 406, 422,  
424, 441, 450, 475, 482, 502, 1025  
`\add@inserts` ..... 878  
`\add@inserts@nextR` ..... 1137  
`\ballast@count` ..... 959, 964  
`\bbl@main@language` ..... 1617, 1618  
`\bbl@set@language` ..... 1608, 1609  
`\beginnumbering` .. 8, 34, 713, 731,  
767, 2027, 2042, 2072, 2093,  
2244, 2323, 2424, 2449, 2493, 2522  
`\beginnumberingR` ... 47, 102, 731, 796  
`\Bfootfmt` 2001, 2002, 2230, 2412, 2413  
`\Bfootins` ..... 2149  
`\Bfootnote` ..... 1183, 2096, 2100,  
2252, 2257–2259, 2261, 2265,  
2266, 2275, 2277, 2279, 2284,  
2286, 2287, 2290, 2292, 2294,  
2296, 2297, 2300, 2303, 2304,  
2307–2309, 2311, 2312, 2478, 2553  
`\Bfootnoterule` ..... 2145  
`\box` ..... 2177  
`\Bparafootfmt` ..... 2202, 2230  
`\bypage@Rfalse` ..... 122, 134  
`\bypage@Rtrue` ..... 122, 130  
**C**  
`\c@ballast` ..... 964

- `\c@firstlinenumR` ..... [153](#), 1103  
`\c@firstsublinenumR` ..... [157](#), 1098  
`\c@linenumincrementR` ..... [153](#), 1103  
`\c@page` ... 564, 1839, 1845, 1848, 1855  
`\c@pstartL` ..... 754  
`\c@pstartR` ..... 756  
`\c@sublinenumincrementR` .. [157](#), 1098  
`\centering` ..... 2419  
`\Cfootfmt` ..... 2231  
`\Cfootins` ..... 2150  
`\Cfootnote` ..... [1183](#), 2267, 2305  
`\Cfootnoterule` ..... 2146  
`\ch@ck@l@ckR` ..... [1031](#)  
`\ch@cksub@l@ckR` ..... [1031](#)  
`\ch@cksub@lockR` ..... 1099  
`\chapter` ..... 693, 694, 702  
`\chapterinpages` ..... [686](#), 694, 704  
`\chardef` ..... 1480  
`\check@goal` ..... 1913, 1927, [1939](#)  
`\check@pstarts` .....  
    1644, 1669, [1687](#), 1744, 1752, 1760  
`\checkpageL` ..... 1771, 1784, [1907](#)  
`\checkpageR` ..... 1793, 1805, [1907](#)  
`\checkraw@text` .....  
    .... 1648, 1665, [1699](#), 1768, 1814  
`\cleardoublepage` ..... 1851  
`\clearl@dleftpage` ..... 1792, [1837](#)  
`\clearl@drightpage` ..... 1813, [1837](#)  
`\cleartoevenpage` ..... [1837](#)  
`\cleartol@devenpage` ..... 1733, [1837](#)  
`\closeout` ..... 555, 559  
`\columnrulewidth` ..... 5, [1682](#), 2018  
`\Columns` ..... 5, [1632](#), 2052, 2112  
`\columnseparator` ..... 5, 1662, [1682](#)  
`\countLline` ..... [852](#), 864  
`\countRline` ..... [852](#), 898  
`\Cparafootfmt` ..... 2215, 2231  
`\critext` ..... [630](#)
- D**
- `\date` ..... 2239  
`\DeclareOption` ..... 7  
Dekker, Dirk-Jan ..... 80, 86  
`\Dfootnote` ..... [1183](#)  
`\dimen` ..... 581, 582, 586–588, 592  
`\divide` ..... 1036  
`\do@actions` ..... 944  
`\do@actions@fixedcodeR` ..... [968](#)  
`\do@actions@nextR` ..... [968](#)  
`\do@actionsR` ..... 929, [968](#)
- `\do@ballast` ..... 945  
`\do@ballastR` ..... 930, [959](#)  
`\do@lineL` ..... [862](#), 1652, 1656, 1775  
`\do@lineLhook` ..... 868, [893](#)  
`\do@lineR` ..... [896](#), 1653, 1658, 1796  
`\do@lineRhook` ..... [893](#), 902  
`\do@lockoff` ..... 470  
`\do@lockoffL` ..... 494  
`\do@lockoffR` ..... [470](#)  
`\do@lockon` ..... [435](#)  
`\do@lockonL` ..... 467  
`\do@lockonR` ..... [435](#)  
`\documentclass` ..... 1995, 2131, 2398  
`\dp` ..... 2160, 2175  
`\dummy@ref` ..... 516
- E**
- `\edfont@info` ..... 669, 672, 678, 681  
`\edindex` . 2249, 2264, 2276, 2288, 2289  
`\edlabel` . [1325](#), 2465, 2480, 2539, 2555  
`\edtext` ..... [647](#), 2077,  
    2083, 2095, 2099, 2252, 2254,  
    2256–2260, 2263, 2265, 2266,  
    2275, 2277, 2279, 2284, 2286,  
    2287, 2290–2294, 2296, 2297,  
    2300, 2302, 2304, 2307–2312,  
    2434, 2441, 2478, 2504, 2512, 2553  
`\Efootnote` ..... [1183](#)  
`\emergencystretch` ..... 2235  
`\empty` .. 76, 79, 235, 243, 641, 658,  
    667, 676, 775, 804, 1021, 1102,  
    1110, 1139–1141, 1152, 1163,  
    1328, 1337, 1862, 1868, 1875, 1881  
`\end@lemmas` ..... 641, 642, 658, 659  
`\endashchar` ..... 1315  
`\endgraf` ..... 823, 839, 1638, 1736  
`\endline@num` ..... 523, 529  
`\endlock` ..... [614](#), 1489, 1498, 1503  
`\endnumbering` ..... 8, 37, [69](#), 106,  
    732, 2034, 2049, 2086, 2109,  
    2318, 2385, 2444, 2484, 2517, 2561  
`\endnumberingR` 50, [69](#), 91, 101, 114, 732  
`\endpage@num` ..... 522, 529  
`\endstanzaextra` .....  
    .... 1505, 2423, 2448, 2492, 2521  
`\endsub` ..... [581](#)  
`\endsubline@num` ..... 524, 530  
`\enskip` ..... 2198, 2211, 2219  
`\enspace` ..... 2197, 2207, 2218

## environments:

astanza ..... 9, 1483  
 Leftside ..... 6, 711  
 pages ..... 5, 686  
 pairs ..... 5, 686  
 Rightside ..... 6, 729  
 \everyhbox ..... 1469  
 \extensionchars .. 45, 64, 97, 111, 119

**F**

\f@x@l@cksR ..... 1031  
 \first@linenum@out@Rfalse .. 550, 556  
 \first@linenum@out@Rtrue ..... 550  
 \firstlinenum ..... 6, 162, 2024,  
     2039, 2069, 2090, 2409, 2450, 2523  
 \firstsublinenum ..... 6, 162  
 \fix@page ..... 296, 303  
 \flag@end ..... 565, 646, 663  
 \flag@start ..... 565, 638, 655  
 \floatingpenalty ..... 2159  
 \flush@notes ..... 1671, 1823  
 \flush@notesR ..... 1161, 1672, 1824  
 \footnote .....  
     . 2330, 2338, 2347, 2352, 2369,  
     2377, 2458, 2464, 2471, 2474,  
     2480, 2531, 2538, 2546, 2549, 2555  
 \footnotesize ..... 2141, 2142  
 \footparagraph . 2225–2227, 2407, 2408  
 \fullstop . 198, 1312, 1314, 1316, 1318

**G**

\get@linelistfile ..... 231  
 \get@nextboxL ..... 1783, 1952  
 \get@nextboxR ..... 1804, 1952  
 \getline@numL ..... 873, 942  
 \getline@numR ..... 907, 927  
 \getlinesfrompagelistL .....  
     ..... 1756, 1816, 1874  
 \getlinesfrompagelistR .....  
     ..... 1757, 1817, 1874  
 \getlinesfromparlistL ... 1748, 1861  
 \getlinesfromparlistR ... 1749, 1861  
 \gl@p ..... 238, 239,  
     246, 247, 642, 659, 671, 680,  
     1024, 1025, 1145, 1149, 1164,  
     1331, 1340, 1865, 1871, 1878, 1884  
 \goalfraction ..... 6, 1939

**H**

\hangingsymbol ..... 11, 1471

\hb@xt@ ... 875, 881, 888, 909, 915,  
     922, 1660, 1779, 1781, 1800, 1802  
 \hsize ..... 785, 814,  
     1660, 1779, 1781, 1800, 1802, 2172  
 \hyphenation ..... 2240

**I**

\if@filesw ..... 1589  
 \if@firstcolumn ..... 1116, 1422  
 \if@nobreak ..... 759, 788  
 \if@pstarts .... 1645, 1687, 1745, 1761  
 \ifaraw@text ... 1650, 1699, 1770, 1815  
 \ifautopar ..... 784, 813  
 \ifbypage@ ..... 319  
 \ifbypage@R ..... 122, 309, 1003  
 \ifdim ..... 582, 586, 588,  
     592, 1779, 1800, 1841, 1914, 1928  
 \iffirst@linenum@out@R .... 550, 554  
 \ifinstanzaL ..... 709, 709, 863  
 \ifinstanzaR ..... 709, 710, 897  
 \ifl@d@dash ..... 1315  
 \ifl@d@elin ..... 1317, 1318  
 \ifl@d@esl ..... 1318  
 \ifl@d@pnum ..... 1312, 1316  
 \ifl@d@ssub ..... 1314  
 \ifl@dpagefull ..... 1786, 1807, 1907  
 \ifl@dpadding ..... 9  
 \ifl@dpairing ..... 9, 73  
 \ifl@dsamelang ..... 1577, 1651  
 \ifl@dsamepage ..... 1774, 1795, 1907  
 \ifl@dskipnumber ..... 1093  
 \ifl@dusedbabel ..... 1575  
 \ifledplinenum ..... 1313, 2206  
 \ifledRcol ..... 9, 145, 167,  
     171, 175, 179, 218, 233, 297,  
     306, 330, 344, 361, 378, 390,  
     398, 419, 464, 491, 500, 509,  
     566, 576, 583, 589, 595, 602,  
     610, 615, 619, 624, 635, 652,  
     666, 1171, 1185, 1199, 1213,  
     1227, 1242, 1256, 1270, 1284,  
     1298, 1326, 1358, 1372, 1383,  
     1395, 1407, 1442, 1455, 1612, 1622  
 \ifnoteschanged@ ..... 83  
 \ifnumberedpar@ ... 769, 798, 819,  
     835, 1170, 1184, 1198, 1212,  
     1226, 1241, 1255, 1269, 1283,  
     1297, 1382, 1394, 1406, 1441, 1454  
 \ifnumbering ..... 35, 765, 816  
 \ifnumberingR .. 48, 70, 93, 125, 794, 832

- \ifnumberpstart ... 784, 813, 828, 844
  - \ifodd 1126, 1432, 1839, 1845, 1848, 1855
  - \ifpst@rtedL ..... 30, 773
  - \ifpst@rtedR ..... 30, 802
  - \ifshiftedverses . 5, 1777, 1798, 1940
  - \ifsublines@ ..... 196,  
285, 329, 362, 369, 400, 409,  
421, 428, 440, 474, 528, 530,  
931, 946, 1010, 1096, 1360, 1364
  - \ifvbox 865, 899, 1703, 1706, 1953, 1973
  - \ifwrittenlinesL ..... 1949, 1957
  - \ifwrittenlinesR ..... 1950, 1977
  - \initnumbering@reg ..... 43
  - \insert ..... 2155
  - \insert@count ..... 506, 572, 636,  
653, 1178, 1192, 1206, 1220,  
1234, 1249, 1263, 1277, 1291,  
1305, 1390, 1402, 1414, 1449, 1462
  - \insert@countR .... 507, 568, 635,  
652, 1174, 1188, 1202, 1216,  
1230, 1245, 1259, 1273, 1287,  
1301, 1386, 1398, 1410, 1445, 1458
  - \insertlines@listR .....  
.... 76, 206, 220, 512, 1141, 1145
  - \inserts@list .....  
.. 774, 1177, 1191, 1205, 1219,  
1233, 1248, 1262, 1276, 1290,  
1304, 1389, 1401, 1413, 1448, 1461
  - \inserts@listR .....  
.. 803, 1136, 1139, 1149, 1163,  
1164, 1173, 1187, 1201, 1215,  
1229, 1244, 1258, 1272, 1286,  
1300, 1385, 1397, 1409, 1444, 1457
  - \instanzaLfalse ..... 1679, 1830
  - \instanzaLtrue ..... 720
  - \instanzaRfalse ..... 1680, 1831
  - \instanzaRtrue ..... 742
  - \interfootnotelinepenalty ..... 2158
  - \interlinepenalty ..... 1491, 2158
  - \interstanza .....  
.... 2419, 2423, 2448, 2492, 2521
- L**
- \l@d@nums ..... 669, 672, 678,  
681, 1173, 1177, 1187, 1191,  
1201, 1205, 1215, 1219, 1229,  
1233, 1244, 1248, 1258, 1262,  
1272, 1276, 1286, 1290, 1300, 1304
  - \l@d@set ..... 377, 610, 611
  - \l@dbbl@set@language .... 1586, 1609
  - \l@dbfnote ..... 1440
  - \l@dc@maxchunks ..... 780, 782,  
809, 811, 1534, 1544, 1552, 1559
  - \l@dcalc@maxoftwo .....  
..... 1750, 1894, 1964, 1984
  - \l@dcalc@minoftwo .. 1758, 1818, 1894
  - \l@dcalcnun ..... 1031
  - \l@dchecklang ..... 1579, 1649
  - \l@dchset@num ..... 252, 255, 377
  - \l@dcsnote ..... 1381
  - \l@dcsnotetext .....  
.... 1420, 1423, 1425, 1433, 1435
  - \l@demptyd@ta ..... 869, 903
  - \l@dend@stuff .... 46, 65, 98, 112, 120
  - \l@dgetline@margin ..... 143
  - \l@dgetsidenote@margin ..... 1370
  - \l@dld@ta ..... 877, 911, 1117, 1129
  - \l@dleftbox .....  
.. 849, 874, 888, 1661, 1779, 1781
  - \l@dlinenumR ..... 188
  - \l@dlsn@te ..... 880, 914
  - \l@dlsnote ..... 1381
  - \l@dmake@labels ..... 1344
  - \l@dmake@labelsR ..... 1334, 1347
  - \l@dminpagelines .....  
.... 1723, 1759, 1819, 1915, 1929
  - \l@dnumpstartsL . 39, 779, 780, 782,  
784, 1538, 1570, 1633, 1634,  
1676, 1690, 1730, 1731, 1828, 1962
  - \l@dnumpstartsR . 52, 808, 809, 811,  
813, 1538, 1571, 1633, 1634,  
1677, 1693, 1730, 1731, 1829, 1982
  - \l@doldbbl@set@language ..... 1608
  - \l@doldselectlanguage 1607, 1611, 1616
  - \l@dpagefullfalse ..... 1907
  - \l@dpagefulltrue ..... 1907
  - \l@dpagingfalse ..... 11, 688, 701
  - \l@dpagingtrue ..... 696
  - \l@dpairingfalse ..... 9, 690, 700
  - \l@dpairingtrue ..... 687, 695
  - \l@dparsedendline ..... 2168
  - \l@dparsedstartline . 2163, 2168, 2170
  - \l@dparsedstartpage ..... 2166, 2171
  - \l@dparsefootspec ..... 2162
  - \l@dpscL 865, 870, 1540, 1572, 1642,  
1646, 1674, 1690, 1703, 1740,  
1746, 1751, 1754, 1762, 1826,  
1953, 1955, 1962, 1964, 1966, 1968
  - \l@dpscR .... 899, 904, 1541, 1573,  
1643, 1647, 1675, 1693, 1706,

- 1741, 1747, 1755, 1763, 1827,  
1973, 1975, 1982, 1984, 1986, 1988  
`\l@drd@ta` ..... 881, 915, 1119, 1127  
`\l@dtrightbox` .....  
     .. 849, 908, 922, 1663, 1800, 1802  
`\l@drsn@te` ..... 882, 916  
`\l@drsnote` ..... 1381  
`\l@dsamelangfalse` ..... 1577, 1580  
`\l@dsamelangtrue` ..... 1577, 1583  
`\l@dsamepagefalse` ..... 1907  
`\l@dsamepagetrue` ..... 1907  
`\l@dsetupmaxlinecounts` .. 1551, 1568  
`\l@dsetupprawboxes` ..... 1543, 1567  
`\l@dskipnumberfalse` ..... 1094  
`\l@dskipnumbertrue` ..... 991  
`\l@dunhbox@line` ..... 881, 915  
`\l@dusedbabelfalse` ..... 1575, 1604  
`\l@dusedbabeltrue` ..... 1575, 1606  
`\l@duselanguage` .....  
     .... 1596, 1655, 1657, 1772, 1794  
`\l@dzeromaxlinecounts` ... 1551, 1569  
`\l@dzeropenalties` 822, 838, 1637, 1735  
`\l@pscL` ..... 1540  
`\l@pscR` ..... 1540  
`\label@refs` .....  
     1329, 1331, 1334, 1338, 1340, 1344  
`\labelref@list` ..... 1337, 1340, 1365  
`\labelref@listR` 1323, 1328, 1331, 1361  
`\language` .. 1587, 1588, 1590–1593  
`\last@page@num` ..... 317, 323  
`\last@page@numR` ..... 303  
`\lastbox` ..... 872, 906  
`\lastskip` ..... 581, 587  
`\Lcolwidth` .... 5, 6, 13, 697, 785,  
     875, 888, 2019, 2020, 2063, 2064  
`\ldots` ..... 2082,  
     2085, 2105, 2108, 2272, 2274, 2303  
`\led@err@BadLeftRightPstarts` ...  
     ..... 21, 1634, 1731  
`\led@err@LeftOnRightPage` ... 24, 1849  
`\led@err@LineationInNumbered` ... 126  
`\led@err@NumberingNotStarted` ... 87  
`\led@err@NumberingShouldHaveStarted`  
     ..... 100  
`\led@err@NumberingStarted` .... 36, 49  
`\led@err@PendNoPstart` ..... 820, 836  
`\led@err@PendNotNumbered` ... 817, 833  
`\led@err@PstartInPstart` ... 770, 799  
`\led@err@PstartNotNumbered` . 766, 795  
`\led@err@RightOnLeftPage` ... 24, 1856  
`\led@err@TooManyPstarts` 18, 781, 810  
`\led@mess@NotesChanged` ..... 84  
`\led@mess@SectionContinued` .....  
     ..... 96, 110, 118  
`\led@warn@BadAction` ..... 993  
`\led@warn@BadAdvancelineLine` 347, 353  
`\led@warn@BadAdvancelineSubline` .  
     ..... 333, 339  
`\led@warn@BadLineation` ..... 136  
`\led@warn@BadSetline` ..... 600  
`\led@warn@BadSetlinenum` ..... 608  
`\led@warn@DuplicateLabel` ..... 1349  
`\ledllfill` ..... 881, 915  
`\ledmac@error` ..... 19, 22, 25, 27  
`\ledplinenumfalse` ..... 2164  
`\ledplinenumtrue` ... 2162, 2167, 2169  
`\ledRcolfalse` ..... 12, 712, 744  
`\ledRcoltrue` ..... 730  
`\ledrlfill` ..... 881, 915  
`\ledsavedprintlines` ..... 8, 1310  
`\ledsetnormalparstuff` 2195, 2203, 2216  
`\ledsidenote` ..... 2336  
`\ledstrutL` ..... 1779, 1781, 1834  
`\ledstrutR` ..... 1800, 1802, 1834  
`\ledthegoal` ..... 1914, 1928, 1939  
`\leftlinenumR` ..... 188, 1117, 1129  
Leftside (environment) ..... 6, 711  
`\Leftsidehook` ..... 718, 724  
`\Leftsidehookend` ..... 723, 724  
`\lemma` ..... 2077, 2303  
`\lemmafnt` ..... 2198, 2219  
`\line@list` ..... 676, 680  
`\line@list@stuff` ..... 45, 111  
`\line@list@stuffR` ... 64, 97, 119, 552  
`\line@listR` . 79, 206, 219, 530, 667, 671  
`\line@margin` ..... 148  
`\line@marginR` ..... 141, 1122  
`\line@num` ..... 320, 351, 352,  
     354, 372, 383, 384, 412, 952, 1363  
`\line@numR` . 57, 195, 202, 257, 291,  
     310, 345, 346, 348, 365, 379,  
     380, 403, 523, 527, 938, 1004,  
     1013, 1101, 1103, 1105, 1106, 1359  
`\lineation` ..... 739, 2135  
`\lineationR` ..... 124, 739  
`\linenum@out` 571, 579, 584, 590, 596,  
     603, 611, 616, 620, 1336, 1714, 1888  
`\linenum@outR` .....  
     . 549, 555, 557, 559, 560, 564,



567, 577, 583, 589, 595, 602,  
610, 615, 619, 624, 1327, 1719, 1891  
`\linenumberlist` ..... 1102, 1106  
`\linenumincrement` .. 6, 162, 2025,  
2040, 2070, 2091, 2410, 2451, 2524  
`\linenummargin` .....  
141, 2026, 2041, 2071, 2092, 2136  
`\linenumr@p` .... 1313, 1317, 1359, 1363  
`\linenumrepR` ..... 185, 195  
`\linenumsep` ..... 190, 192  
`\linesinpar@listL` .....  
..... 211, 227, 537, 1862, 1865  
`\linesinpar@listR` .....  
..... 211, 223, 540, 1868, 1871  
`\linesonpage@listL` 228, 544, 1875, 1878  
`\linesonpage@listR` 224, 547, 1881, 1884  
`\list@clear` .....  
. 219–224, 227, 228, 230, 774, 803  
`\list@clearing@reg` ..... 226  
`\list@create` .....  
... 206–209, 211–213, 1136, 1323  
`\lock@disp` ..... 1063, 1067, 1072  
`\lock@off` .... 461, 462, 470, 619, 620  
`\lock@on` ..... 615, 616  
`\longdash` ..... 2405, 2419

## M

`\manageparhangingsymbol` 863, 897, 1467  
`\maxchunks` ..... 4, 1534, 2402  
`\maxdimen` ..... 2172  
`\maxlinesinpar@list` ..... 211, 230  
`\memorydump` ..... 8, 717, 735  
`\memorydumpL` ..... 105, 717  
`\memorydumpR` ..... 105, 735  
`\message` ..... 44, 63  
`\morenoexpands` ..... 2190  
`\mpAfootnote` ..... 1240  
`\mpBfootnote` ..... 1240  
`\mpCfootnote` ..... 1240  
`\mpDfootnote` ..... 1240  
`\mpEfootnote` ..... 1240  
`\mpvAfootnote` ..... 1243, 1247, 1252  
`\mpvBfootnote` ..... 1257, 1261, 1266  
`\mpvCfootnote` ..... 1271, 1275, 1280  
`\mpvDfootnote` ..... 1285, 1289, 1294  
`\mpvEfootnote` ..... 1299, 1303, 1308  
`\multiply` ..... 1037

## N

`\n@num` ..... 498, 624

`\n@num@reg` ..... 504  
`\namebox` ..... 865, 870, 899,  
904, 1518, 1703, 1706, 1953, 1973  
`\NeedsTeXFormat` ..... 2  
`\new@line` ..... 881  
`\new@lineR` ..... 563, 915  
`\newbox` ..... 750, 849, 850, 1519  
`\newcounter` 153, 155, 157, 159, 753, 755  
`\newif` ..... 5, 10, 31,  
122, 550, 709, 710, 1575, 1577,  
1687, 1699, 1907, 1909, 1949, 1950  
`\newnamebox` ..... 1518, 1546, 1547  
`\newnamecount` ..... 1529, 1554  
`\newwrite` ..... 549  
`\next@action` ..... 247  
`\next@actionline` ..... 244, 246  
`\next@actionlineR` .....  
.. 236, 238, 962, 1000, 1022, 1024  
`\next@actionR` ..... 239, 963,  
1001, 1002, 1007, 1008, 1016, 1025  
`\next@insert` ..... 775  
`\next@insertR` .....  
804, 1140, 1143, 1145, 1148, 1152  
`\next@page@num` ..... 324, 395  
`\next@page@numR` 61, 260, 262, 314, 392  
`\no@expands` ..... 632, 649  
`\nobrak` ..... 2188, 2189  
`\noindent` ..... 2173, 2248, 2328  
`\normal@pars` ..... 72, 778, 807  
`\normalbfnoteX` ..... 1453  
`\notefontsetup` ..... 2157  
`\notenumfont` ... 2141, 2197, 2205, 2218  
`\noteschanged@true` .....  
..... 77, 80, 668, 677, 1142  
`\notetextfont` .. 2142, 2199, 2212, 2220  
`\num@lines` ..... 823, 1638, 1736  
`\num@linesR` .... 749, 839, 1639, 1737  
`\numberedpar@true` ..... 786, 815  
`\numberingRfalse` ..... 71  
`\numberingRtrue` ..... 54, 91, 115  
`\numberingtrue` ..... 41, 107  
`\numberpstartfalse` ..... 9  
`\numberpstarttrue` ..... 9  
`\numlabfont` ..... 195  
`\numpagelinesL` .....  
.... 1723, 1776, 1787, 1791, 1915  
`\numpagelinesR` .....  
.... 1723, 1797, 1808, 1812, 1929

## O

\oldBfootfmt . . . 2001, 2004, 2412, 2415  
 \oldchapter . . . . . 693, 702  
 \oldstanza 719, 720, 722, 741, 742, 745  
 \one@line . . . . . 870, 872, 881  
 \one@lineR . . . . . 749, 904, 906, 915  
 \openout . . . . . 557, 560

## P

\page@action . . . . . 261, 389, 517  
 \page@num . . . . . 242, 322, 1430  
 \page@numR . . . . .  
     215, 234, 312, 522, 527, 1002, 1124  
 \pagegoal . . . . . 1947  
 \Pages . . . . . 5, 1727, 2387, 2487, 2564  
 pages (environment) . . . . . 5, 686  
 \pagetotal . . . . . 1841, 1914, 1928  
 pairs (environment) . . . . . 5, 686  
 \par@line . . . . . 824, 1640, 1738  
 \par@lineR . . . . . 749, 840, 1641, 1739  
 \para@vfootnote . . . . . 2154  
 \pausenumbering . . . . . 733  
 \pausenumberingR . . . . . 90, 733  
 \pend . . . 7, 716, 738, 771, 1504, 2281,  
     2313, 2317, 2360, 2380, 2384, 2419  
 \pendL . . . . . 716, 816  
 \pendR . . . . . 738, 800, 832  
 \prevgraf . . . . .  
     . 823, 839, 1638, 1639, 1736, 1737  
 \previous@A@number . . . . . 2182  
 \previous@B@number . . . . . 2183  
 \previous@C@number . . . . . 2184  
 \previous@page . . . . . 2166, 2171, 2185  
 \printindex . . . . . 2391  
 \printlines . . . . .  
     1321, 2003, 2197, 2205, 2218, 2414  
 \printlinesR . . . . . 8, 1310, 2003, 2414  
 \ProcessOptions . . . . . 8  
 \protected@write . . . 1333, 1343, 1590  
 \ProvidesPackage . . . . . 3  
 \pst@rtedLfalse . . . . . 30, 40  
 \pst@rtedLtrue . . . . . 108, 776  
 \pst@rtedRfalse . . . . . 32, 53, 74  
 \pst@rtedRtrue . . . . . 94, 116, 805  
 \pstart 7, 19, 23, 714, 737, 1506, 2244,  
     2283, 2315, 2324, 2362, 2382, 2419  
 \pstartL . . . . . 714, 752  
 \pstartR . . . . . 737, 752

## R

\rbracket . . . . . 2188, 2211  
 \Rcolwidth . . . . . 5, 6,  
     13, 698, 814, 909, 922, 2020, 2064  
 \read@linelist . . . . . 217, 553  
 \rem@inder . . . . . 1106, 1108–1110  
 \resumenumbering . . . . . 734  
 \resumenumberingR . . . . . 90, 734  
 \rightlinenumR . . . . . 188, 1119, 1127  
 Rightside (environment) . . . . . 6, 729  
 \Rightsidehook . . . . . 724, 740  
 \Rightsidehookend . . . . . 724, 746  
 \rlap . . . . . 1119, 1127  
 \Rlineflag . . . . . 8, 183, 195,  
     1313, 1317, 1351, 1999, 2016, 2233  
 \rule . . . . . 1683

## S

\sc@n@list . . . . . 1107, 1109  
 \secdef . . . . . 707  
 \section@num . . . . . 42, 44, 45, 109–111  
 \section@numR . . . . .  
     . . . 28, 55, 63, 64, 95–97, 117–119  
 \select@language . . . 1588, 1590–1593  
 \select@lemmafont . . . . . 2211  
 \selectlanguage . . . . 1596, 2245, 2322  
 \set@line . . . . . 634, 651, 665  
 \set@line@action . . . . .  
     . . . . . 254, 358, 367, 374, 397, 519  
 \setl@dlp@rbox . . . . . 1423, 1435  
 \setl@drp@rbox . . . . . 1425, 1433  
 \setline . . . . . 598, 2248, 2328  
 \setlinenum . . . . . 606  
 \setnamebox . . . . . 784, 813, 1518  
 \setprintlines . . . . . 1311  
 \setstanzaindents . . . . .  
     . . . . . 2014, 2061, 2403, 2452, 2525  
 \shiftedversesfalse . . . . . 6  
 \shiftedversestrue . . . . . 7  
 \showlemma . . . . . 640, 657  
 \sidenote@margin . . . . . 1375, 1379  
 \sidenote@marginR . . . . . 1368, 1428  
 \sidenotemargin . . . . 1368, 2137, 2322  
 \skip . . . . . 2148–2150  
 \skip@lockoff . . . . . 462, 470  
 \skipnumbering . . . . . 9, 623, 2419  
 \skipnumbering@reg . . . . . 627  
 \smash . . . . . 1683  
 \splitmaxdepth . . . . . 2160  
 \splittopskip . . . . . 867, 901, 2160

- `\stanza` . . . 719, 720, 722, 741, 742,  
 745, 2028, 2043, 2073, 2094,  
 2426, 2432, 2438, 2454, 2462, 2470  
`\stanza@count` . . . . . 1486, 1500, 1511  
`\stanza@hang` . . . . . 1488, 1513  
`\stanzaindentbase` . . . . . 1511  
`\startlock` . . . . . 614  
`\startstanzahook` . . . . . 1484  
`\startsub` . . . . . 581  
`\sub@action` . . . . . 270, 418, 518  
`\sub@change` . . . . . 62, 264, 265, 271  
`\sub@lock` . . . . . 947  
`\sub@lockR` . . . . . 59, 279, 281, 283,  
 286, 436, 442, 443, 445, 446,  
 476, 477, 479, 932, 983, 985,  
 986, 988, 1044, 1084, 1086, 1088  
`\sub@off` . . . . . 589, 590  
`\sub@on` . . . . . 583, 584  
`\subline@num` . . . . . 197, 320, 337,  
 338, 340, 370, 410, 948, 954, 1364  
`\subline@numR` . . . . . 198,  
 202, 287, 291, 310, 331, 332,  
 334, 363, 401, 524, 528, 933,  
 939, 1004, 1011, 1097, 1098, 1360  
`\sublinenumincrement` . . . . . 6, 162  
`\sublinenumr@p` . 1314, 1318, 1360, 1364  
`\sublinenumrepR` . . . . . 185, 198  
`\sublines@false` . . . . . 60, 268, 973  
`\sublines@true` . . . . . 266, 971  
`\sublock@disp` . . . . . 1046, 1050, 1055  
`\symplinenum` . . . . . 1313  
`\sza@penalty` . . . . . 1495, 1499
- T**
- `\textdagger` . . . . . 2347  
`\textheight` . . . . . 1996, 2400  
`\textit` 2058, 2083, 2085, 2106, 2108,  
 2118, 2258, 2271, 2292, 2297,  
 2303, 2305, 2307, 2355, 2474, 2549  
`\textnormal` . . . . . 2189  
`\textsc` . . . . . 2057, 2253  
`\textwidth` 14, 16, 697, 698, 2019, 2063  
`\theledlanguageL` 1581, 1596, 1655, 1772  
`\theledlanguageR` 1581, 1596, 1657, 1794  
`\thepage` . . . . . 564, 1334, 1344  
`\thepstart` . . . . . 715, 736  
`\thepstartL` . . . . . 9, 715, 754, 784  
`\thepstartR` . . . . . 9, 736, 756, 813  
`\thr@@` . . . . . 445, 454, 477, 484, 978, 986  
`\title` . . . . . 2238
- `\topskip` . . . . . 1841
- U**
- `\unhbox` . . . . . 1523,  
 1661, 1663, 1779, 1781, 1800, 1802  
`\unhnamebox` . . . . . 1518  
`\unvbox` . . . . . 872, 906, 1525  
`\unvnamebox` . . . . . 1518  
`\unvxh` . . . . . 2174  
`\usernamecount` . . . . .  
 . 1487, 1494, 1529, 1561, 1751,  
 1955, 1964, 1966, 1975, 1984, 1986  
`\usepackage` . . . . 1997, 2132–2134, 2399
- V**
- `\vAfootnote` . . . . . 1172, 1176, 1181  
`\value` . . . . . 1470  
`\vbadness` . . . . . 866, 900  
`\vbfnoteX` . . . . . 1456, 1460  
`\vBfootnote` . . . . . 1186, 1190, 1195  
`\vbox` . . . . . 784, 813, 2172  
`\vCfootnote` . . . . . 1200, 1204, 1209  
`\vDfootnote` . . . . . 1214, 1218, 1223  
`\vEfootnote` . . . . . 1228, 1232, 1237  
`\vl@dbfnote` . . . . . 1443, 1447  
`\vl@dcsnote` . . . . . 1408, 1412  
`\vl@dlsnote` . . . . . 1384, 1388  
`\vl@drsnote` . . . . . 1396, 1400  
`\vsplit` . . . . . 870, 904
- W**
- `\wd` . . . . . 881, 915, 2176  
`\writtenlinesLfalse` . . . . 1742, 1963  
`\writtenlinesLtrue` . . . . . 1960  
`\writtenlinesRfalse` . . . . 1743, 1983  
`\writtenlinesRtrue` . . . . . 1980
- X**
- `\x@lemma` . . . . . 642–644, 659–661  
`\xlineref` . . . . . 2481, 2556  
`\xpg@main@language` . . . . . 1626, 1627  
`\xpg@set@language` . . . . . 1621, 1625  
`\xright@appenditem` . . . . .  
 . . . . . 391, 392, 394, 395, 399,  
 406, 408, 415, 420, 422, 424,  
 427, 429, 431, 439, 441, 450,  
 473, 475, 482, 501, 502, 512,  
 526, 537, 540, 544, 547, 1172,  
 1176, 1186, 1190, 1200, 1204,  
 1214, 1218, 1228, 1232, 1243,

1247, 1257, 1261, 1271, 1275, 1285, 1289, 1299, 1303, 1359, 1363, 1384, 1388, 1396, 1400, 1408, 1412, 1443, 1447, 1456, 1460	<b>Z</b>  <code>\z@skip</code> ..... 2161 <code>\zz@@@</code> ..... 1329, 1338
---------------------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------

## Change History

v0.1	General: First public release ..... 1	by introducing <code>\l@dlinenumR</code> 18
v0.2	General: Added section of babel re- lated code ..... 56 Fix babel problems ..... 1 <code>\Columns</code> : Added <code>\l@dchecklang</code> and <code>\l@duselanguage</code> to <code>\Columns</code> ..... 59 <code>\Pages</code> : Added <code>\l@duselanguage</code> to <code>\Pages</code> ..... 63	<code>\l@dnumprintstartR</code> : Moved <code>\l@dnumprintstartL</code> to <code>ledmac</code> . 55 <code>\ledsavedprintlines</code> : Simpli- fied <code>\printlinesR</code> by using <code>\setprintlines</code> ..... 49 <code>\ledstrutR</code> : Added <code>\ledtrutL</code> and <code>\ledstrutR</code> ..... 65 <code>\normalbfnoteX</code> : Removed extraneous spaces from <code>\normalbfnoteX</code> ..... 53 <code>\Pages</code> : Added <code>\ledstrutL</code> to <code>\Pages</code> ..... 63 Added <code>\ledstrutR</code> to <code>\Pages</code> . 64 <code>\Rightsidehookend</code> : Added <code>\Leftsidehook</code> , <code>\Leftsidehookend</code> , <code>\Rightsidehook</code> and <code>\Rightsidehookend</code> ..... 33 <code>\sublinenumrepR</code> : Added <code>\linenumrepR</code> and <code>\sublinenumrepR</code> ..... 18
v0.3	General: Reorganize for <code>ledarab</code> .. 1 <code>\affixline@numR</code> : Changed <code>\affixline@numR</code> to match new <code>ledmac</code> ..... 42 <code>\do@actions@nextR</code> : Used <code>\do@actions@fixedcode</code> in <code>\do@actionsR</code> ..... 40 <code>\do@lineL</code> : Added <code>\do@lineLhook</code> to <code>\do@lineL</code> ..... 38 Simplified <code>\do@lineL</code> by using macros for some common code 38 <code>\do@lineR</code> : Changed <code>\do@lineR</code> similarly to <code>\do@lineL</code> ..... 38 <code>\do@lineRhook</code> : Added <code>\do@lineLhook</code> and <code>\do@lineRhook</code> ..... 38 Leftside: Added hooks into Left- side environment ..... 33 <code>\flag@end</code> : Removed extraneous spaces from <code>\flag@end</code> ..... 29 <code>\ifledRcol</code> : Moved <code>\ifl@dpairing</code> to <code>ledmac</code> ..... 13 <code>\ifpst@rtedR</code> : Moved <code>\ifpst@rtedL</code> to <code>ledmac</code> ..... 14 <code>\l@dlinenumR</code> : Simplified <code>\leftlinenumR</code> and <code>\rightlinenumR</code>	v0.3a General: Minor <code>\linenummargin</code> fix ..... 1 <code>\line@marginR</code> : Don't just set <code>\line@marginR</code> in <code>\linenummargin</code> ..... 17 v0.3b General: Improved parallel page balancing ..... 1 <code>\Pages</code> : Added <code>\l@dminpagelines</code> calculation for succeeding page pairs ..... 64 v0.3c General: Compatibilty with Poly- glossia ..... 1

v0.4	General: No more ledparpatch. All patches are now in the main file. . . . .	1	<code>\hangingsymbol</code> to define the character. . . . .	1
v0.5	General: Corrections about <code>\section</code> and other titles in numbered sections . . . . .	1	v0.9	General: Possibility to number <code>\pstart</code> . . . . .
v0.6	General: Be able to use <code>\chapter</code> in parallel pages. . . . .	1		Possibility to number the <code>pstart</code> with the commands <code>\numberpstarttrue</code> . . . . .
v0.7	General: Option 'shiftedverses' which make there is no blank between two parallel verses with unequal length. . . . .	1		<code>\ifledRcol</code> : Moved <code>\iflledRcol</code> and <code>\ifnumberingR</code> to ledmac 13
v0.8	General: Possibility to have a symbol on each hanging of verses, like in the french typography. Redefine the commande		v0.9.1	General: The numbering of the <code>pstarts</code> restarts on each <code>\beginnumbering</code> . . . . .
			v0.9.2	General: Debug : with <code>\Columns</code> , the hanging indentation now runs on the left columns and the hanging symbol is shown only when <code>\stanza</code> is used. . . . .