

# 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](https://github.com) to access my page ([maieul/ledmac](https://github.com/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>6</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 v0.11, last revised 2012/06/11.

<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 . . . . .	20
11.4 Commands within the line-list file . . . . .	21
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 . . . . .	40
14.4 Line number printing . . . . .	42
14.5 Pstart number printing in side . . . . .	44
14.6 Add insertions to the vertical list . . . . .	46
14.7 Penalties . . . . .	46
14.8 Printing leftover notes . . . . .	47
<b>15 Footnotes</b>	<b>48</b>
15.1 Outer-level footnote commands . . . . .	48
15.2 Normal footnote formatting . . . . .	51
<b>16 Cross referencing</b>	<b>52</b>
<b>17 Side notes</b>	<b>53</b>
<b>18 Familiar footnotes</b>	<b>55</b>
<b>19 Verse</b>	<b>56</b>
<b>20 Naming macros</b>	<b>57</b>
<b>21 Counts and boxes for parallel texts</b>	<b>58</b>
<b>22 Fixing babel</b>	<b>59</b>
<b>23 Parallel columns</b>	<b>61</b>
<b>24 Parallel pages</b>	<b>64</b>

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

<b>25 The End</b>	<b>72</b>
-------------------	-----------

<b>A Examples</b>	<b>73</b>
-------------------	-----------

A.1 Parallel column example . . . . .	82
A.2 Example parallel facing pages . . . . .	84
A.3 Example poetry on parallel facing pages . . . . .	90

<b>References</b>	<b>95</b>
-------------------	-----------

<b>Index</b>	<b>95</b>
--------------	-----------

<b>Change History</b>	<b>104</b>
-----------------------	------------

## List of Figures

1 Output from <code>villon.tex</code> . . . . .	75
2 Left page output from <code>djd17nov.tex</code> . . . . .	76
3 Right page output from <code>djd17nov.tex</code> . . . . .	77
4 First left page output from <code>djdpoems.tex</code> . . . . .	78
5 First right page output from <code>djdpoems.tex</code> . . . . .	79
6 Second left page output from <code>djdpoems.tex</code> . . . . .	80
7 Second right page output from <code>djdpoems.tex</code> . . . . .	81

## 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 `pdflatex` or `xelatex`. 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. When you use `\stanza`, the visible rule may shift when a verse has a hanging indent. To prevent shifting, use `\setstanzaindents` outside the **Leftside** or **Rightside** environment.

## 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.

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.

```
\firstlinenum
\linenumincrement
\firstsublinenum
\sublinenumincrement
```

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

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 `\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}[2012/06/11 v0.11 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; sometimes you want line numbers that start at 1 at each `\pstart`; 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.

The `\ifbypage@R` and `\ifbypstart@R` flag specify the current lineation system:

- line-of-page : `bypstart@R = false` and `bypage@R = true`.
- line-of-pstart : `bypstart@R = true` and `bypage@R = false`.

`ledpar` will use the line-of-section system unless instructed otherwise.

```

122 \newif\ifbypage@R
123 \newif\ifbypstart@R
124 \bypage@Rfalse
125 \bypstart@Rfalse

```

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

```

126 \newcommand*{\lineationR}[1]{%
127 \ifnumbering
128 \led@err@LineationInNumbered
129 \else
130 \def\@tempa{#1}\def\@tempb{page}%
131 \ifx\@tempa\@tempb
132 \global\bypage@Rtrue
133 \global\bypstart@Rfalse
134 \else
135 \def\@tempb{pstart}%

```



```

136     \ifx\@tempa\@tempb
137         \global\bypage@Rfalse
138         \global\bystart@Rtrue
139     \else
140         \def@tempb{section}
141         \ifx\@tempa\@tempb
142             \global\bypage@Rfalse
143             \global\bystart@Rfalse
144         \else
145             \led@warn@BadLineation
146         \fi
147     \fi
148 \fi
149 \fi}}

```

`\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.

```

150 \newcount\line@marginR
151 \renewcommand*{\linenummargin}[1]{%
152     \l@getline@margin{#1}%
153     \ifnum\@l@tempcntb>\m@ne
154         \ifledRcol
155             \global\line@marginR=\@l@tempcntb
156         \else
157             \global\line@margin=\@l@tempcntb
158         \fi
159     \fi}}

```

By default put right text numbers at the right.

```

160 \line@marginR=\@ne
161

```

`\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.

```

162 \newcounter{firstlinenumR}
163 \setcounter{firstlinenumR}{5}
164 \newcounter{linenumincrementR}
165 \setcounter{linenumincrementR}{5}

```

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

```
166 \newcounter{firstsublinenumR}
167 \setcounter{firstsublinenumR}{5}
168 \newcounter{sublinenumincrementR}
169 \setcounter{sublinenumincrementR}{5}
170
```

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

```
\firstsublinenum 171 \providecommand*\firstlinenum{}
\sublinenumincrement 172 \providecommand*\linenumincrement{}
173 \providecommand*\firstsublinenum{}
174 \providecommand*\sublinenumincrement{}
175 \renewcommand*\firstlinenum[1]{%
176   \ifledRcol \setcounter{firstlinenumR}{#1}%
177   \else      \setcounter{firstlinenumR}{#1}%
178   \fi}
179 \renewcommand*\linenumincrement[1]{%
180   \ifledRcol \setcounter{linenumincrementR}{#1}%
181   \else      \setcounter{linenumincrementR}{#1}%
182   \fi}
183 \renewcommand*\firstsublinenum[1]{%
184   \ifledRcol \setcounter{firstsublinenumR}{#1}%
185   \else      \setcounter{firstsublinenumR}{#1}%
186   \fi}
187 \renewcommand*\sublinenumincrement[1]{%
188   \ifledRcol \setcounter{sublinenumincrementR}{#1}%
189   \else      \setcounter{sublinenumincrementR}{#1}%
190   \fi}
191
```

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

```
192 \newcommand*\Rlineflag{R}
193
```

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

```
194 \newcommand*\linenumrepR[1]{\@arabic{#1}}
195 \newcommand*\sublinenumrepR[1]{\@arabic{#1}}
196
```

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

```
197 \newcommand*\leftlinenumR{%
198   \l@dlinenumR
199   \kern\linenumsep}
```

```

200 \newcommand*{\rightlinenumR}{%
201   \kern\linenumsep
202   \l@dlinenumR}
203 \newcommand*{\l@dlinenumR}{%
204   \numlabfont\linenumrepR{\line@numR}\Rlineflag%
205   \ifsublines@
206     \ifnum\subline@num>\z@
207       \unskip\fullstop\sublinenumrepR{\subline@numR}%
208     \fi
209   \fi}
210

```

## 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.

```

211 \newcount\line@numR
212 \newcount\subline@numR
213 \newcount\absline@numR
214

```

`\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:

```

215 \list@create{\line@listR}
216 \list@create{\insertlines@listR}
217 \list@create{\actionlines@listR}
218 \list@create{\actions@listR}
219

```

`\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.

```

220 \list@create{\linesinpar@listL}
221 \list@create{\linesinpar@listR}
222 \list@create{\maxlinesinpar@list}
223

```

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

```

224 \newcount\page@numR
225

```

### 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.

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

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

```
227 \ifledRcol
228 \list@clear{\line@listR}%
229 \list@clear{\insertlines@listR}%
230 \list@clear{\actionlines@listR}%
231 \list@clear{\actions@listR}%
232 \list@clear{\linesinpar@listR}%
233 \list@clear{\linesonpage@listR}%
234 \else
235 \list@clearing@reg
236 \list@clear{\linesinpar@listL}%
237 \list@clear{\linesonpage@listL}%
238 \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).

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

Now get the file and interpret it.

```
240 \get@linelistfile{#1}%
241 \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.

```
242 \ifledRcol
243 \global\page@numR=\m@ne
244 \ifx\actionlines@listR\empty
245 \gdef\next@actionlineR{1000000}%
246 \else
247 \gl@p\actionlines@listR\to\next@actionlineR
248 \gl@p\actions@listR\to\next@actionR
249 \fi
250 \else
251 \global\page@num=\m@ne
252 \ifx\actionlines@list\empty
253 \gdef\next@actionline{1000000}%
254 \else
255 \gl@p\actionlines@list\to\next@actionline
256 \gl@p\actions@list\to\next@action
257 \fi
258 \fi}
259
```

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

## 11.4 Commands within the line-list file

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

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

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

```

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

```

```

295     \ifnum\sub@lockR<\tw@
296         \advance\subline@numR \@ne
297     \fi
298 \else
299     \ifnum\@lockR<\tw@
300         \advance\line@numR \@ne \subline@numR \z@
301     \fi
302 \fi}
303
304 \renewcommand*{\@l}[2]{%
305     \fix@page{#1}%
306     \ifledRcol
307         \@l@regR
308     \else
309         \@l@reg
310     \fi}
311

```

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

```

\fix@page 312 \newcount\last@page@numR
313     \last@page@numR=-10000
314 \renewcommand*{\fix@page}[1]{%
315     \ifledRcol
316         \ifnum #1=\last@page@numR
317         \else
318             \ifbypage@R
319                 \line@numR \z@ \subline@numR \z@
320             \fi
321             \page@numR=#1\relax
322             \last@page@numR=#1\relax
323             \def\next@page@numR{#1}%
324         \fi
325     \else
326         \ifnum #1=\last@page@num
327         \else
328             \ifbypage@
329                 \line@num \z@ \subline@num \z@
330             \fi
331             \page@num=#1\relax
332             \last@page@num=#1\relax
333             \def\next@page@num{#1}%
334         \fi
335     \fi}
336

```

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

```

337 \renewcommand*{\@adv}[1]{%
338     \ifsublines@

```

```

339 \ifledRcol
340 \advance\subline@numR by #1\relax
341 \ifnum\subline@numR<\z@
342 \led@warn@BadAdvancelineSubline
343 \subline@numR \z@
344 \fi
345 \else
346 \advance\subline@num by #1\relax
347 \ifnum\subline@num<\z@
348 \led@warn@BadAdvancelineSubline
349 \subline@num \z@
350 \fi
351 \fi
352 \else
353 \ifledRcol
354 \advance\line@numR by #1\relax
355 \ifnum\line@numR<\z@
356 \led@warn@BadAdvancelineLine
357 \line@numR \z@
358 \fi
359 \else
360 \advance\line@num by #1\relax
361 \ifnum\line@num<\z@
362 \led@warn@BadAdvancelineLine
363 \line@num \z@
364 \fi
365 \fi
366 \fi
367 \set@line@action}
368

```

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

```

369 \renewcommand*{\@set}[1]{%
370 \ifledRcol
371 \ifsublines@
372 \subline@numR=#1\relax
373 \else
374 \line@numR=#1\relax
375 \fi
376 \set@line@action
377 \else
378 \ifsublines@
379 \subline@num=#1\relax
380 \else
381 \line@num=#1\relax
382 \fi
383 \set@line@action
384 \fi}
385

```

`\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.

```

386 \renewcommand*{\l@d@set}[1]{%
387   \ifl@dchset@num
388     \line@numR=#1\relax
389     \advance\line@numR \@ne
390     \def\l@dchset@num{#1}
391   \else
392     \line@num=#1\relax
393     \advance\line@num \@ne
394     \def\l@dchset@num{#1}
395   \fi}
396 \let\l@dchset@num\relax
397

```

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

```

398 \renewcommand*{\page@action}{%
399   \ifl@dchset@num
400     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
401     \xright@appenditem{\next@page@numR}\to\actions@listR
402   \else
403     \xright@appenditem{\the\absline@num}\to\actionlines@list
404     \xright@appenditem{\next@page@num}\to\actions@list
405   \fi}

```

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

```

406 \renewcommand*{\set@line@action}{%
407   \ifl@dchset@num
408     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
409     \ifsublines@
410       \l@dtempcnta=-\subline@numR
411     \else
412       \l@dtempcnta=-\line@numR
413     \fi
414     \advance\l@dtempcnta by -5000\relax
415     \xright@appenditem{\the\l@dtempcnta}\to\actions@listR
416   \else
417     \xright@appenditem{\the\absline@num}\to\actionlines@list
418     \ifsublines@
419       \l@dtempcnta=-\subline@num
420     \else
421       \l@dtempcnta=-\line@num
422     \fi
423     \advance\l@dtempcnta by -5000\relax
424     \xright@appenditem{\the\l@dtempcnta}\to\actions@list
425   \fi}

```



426

`\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.

```

427 \renewcommand*{\sub@action}{%
428   \ifledRcol
429     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
430     \ifsublines@
431       \xright@appenditem{-1001}\to\actions@listR
432     \else
433       \xright@appenditem{-1002}\to\actions@listR
434     \fi
435   \else
436     \xright@appenditem{\the\absline@num}\to\actionlines@list
437     \ifsublines@
438       \xright@appenditem{-1001}\to\actions@list
439     \else
440       \xright@appenditem{-1002}\to\actions@list
441     \fi
442   \fi}
443
```

`\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.

```

444 \newcount\@lockR
445 \newcount\sub@lockR
446
447 \newcommand*{\do@lockonR}{%
448   \xright@appenditem{\the\absline@numR}\to\actionlines@listR
449   \ifsublines@
450     \xright@appenditem{-1005}\to\actions@listR
451     \ifnum\sub@lockR=\z@
452       \sub@lockR \@ne
453     \else
454       \ifnum\sub@lockR=\thr@@
455         \sub@lockR \@ne
456       \fi
457     \fi
458   \else
459     \xright@appenditem{-1003}\to\actions@listR
460     \ifnum\@lockR=\z@
461       \@lockR \@ne
462     \else
463       \ifnum\@lockR=\thr@@
464         \@lockR \@ne
465       \fi
466     \fi
467   \fi}

```

```

468
469 \renewcommand*{\do@lockon}{%
470   \ifx\next\lock@off
471     \global\let\lock@off=\skip@lockoff
472   \else
473     \ifledRcol
474       \do@lockonR
475     \else
476       \do@lockonL
477     \fi
478   \fi}

\lock@off \lock@off adds an entry to the action-code list to turn line number locking off.
\do@lockoff 479
\do@lockoffR 480
\skip@lockoff 481 \newcommand{\do@lockoffR}{%
482   \xright@appenditem{\the\absline@numR}\to\actionlines@listR
483   \ifsublines@
484     \xright@appenditem{-1006}\to\actions@listR
485     \ifnum\sub@lockR=\tw@
486       \sub@lockR \thr@@
487     \else
488       \sub@lockR \z@
489     \fi
490   \else
491     \xright@appenditem{-1004}\to\actions@listR
492     \ifnum\@lockR=\tw@
493       \@lockR \thr@@
494     \else
495       \@lockR \z@
496     \fi
497   \fi}
498
499 \renewcommand*{\do@lockoff}{%
500   \ifledRcol
501     \do@lockoffR
502   \else
503     \do@lockoffL
504   \fi}
505 \global\let\lock@off=\do@lockoff
506

\n@num This macro implements the \skipnumbering command. It uses a new action code,
namely 1007.
507 \providecommand*{\n@num}{%
508 \renewcommand*{\n@num}{%
509   \ifledRcol
510     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
511     \xright@appenditem{-1007}\to\actions@listR
512   \else

```

```

513     \n@num@reg
514     \fi}
515

```

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

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

```

516     \newcount\insert@countR

```

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

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

```

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

```

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

```

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

```

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

```

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

```

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

```
540 #2
541 \else
```

And when not in right text

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

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

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

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

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

## 11.5 Writing to the line-list file

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

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

```
558 \newwrite\linenum@outR
```

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

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

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

```

601 \ifdim\dimen0>0pt \hskip\dimen0 \fi}
602

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

## 12 Marking text for notes

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

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

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

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

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

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

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

```

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

```

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

```

656 \renewcommand{\edtext}[2]{\leavevmode
657   \begingroup

```

```

658 \no@expands
659 \xdef\@tag{#1}%
660 \set@line
661 \ifledRcol \global\insert@countR \z@
662 \else \global\insert@count \z@ \fi
663 \ignorespaces #2\relax
664 \flag@start
665 \endgroup
666 \showlemma{#1}%
667 \ifx\end@lemmas\empty \else
668 \gl@p\end@lemmas\to\x@lemma
669 \x@lemma
670 \global\let\x@lemma=\relax
671 \fi
672 \flag@end}
673

```

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

```

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

```

## 13 Parallel environments

The initial set up for parallel processing is deceptively simple.

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

`chapterinpages` 695 \newenvironment{pairs}{%}



```

696 \l@dpairingtrue
697 \l@dpagingfalse
698 }{%
699 \l@dpairingfalse
700 }

```

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

```

701 \newenvironment{pages}{%
702 \let\oldchapter\chapter
703 \let\chapter\chapterinpages
704 \l@dpairingtrue
705 \l@dpagingtrue
706 \setlength{\Lcolwidth}{\textwidth}%
707 \setlength{\Rcolwidth}{\textwidth}%
708 }{%
709 \l@dpairingfalse
710 \l@dpagingfalse
711 \let\chapter\oldchapter
712 }
713 \newcommand{\chapterinpages}{\thispagestyle{plain}%
714 \global\@topnum\z@
715 \afterindentfalse
716 \secdef\@chapter\@schapter}
717

```

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

```

718 \newif\ifinstanzaL
719 \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.

```

720 \newenvironment{Leftside}{%
721 \ledRcolfalse
722 \let\beginnumbering\beginnumbering\setcounter{pstartL}{1}
723 \let\pstart\pstartL
724 \let\thepstart\thepstartL
725 \let\pend\pendL
726 \let\memorydump\memorydumpL
727 \Leftsidehook
728 \let\oldstanza\stanza
729 \renewcommand{\stanza}{\oldstanza\global\instanzaLtrue}
730 }{
731 \let\stanza\oldstanza
732 \Leftsidehookend}

```

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

`\Leftsidehookend`

`\Rightsidehook` 733 `\newcommand*{\Leftsidehook}{}`

`\Rightsidehookend` 734 `\newcommand*{\Leftsidehookend}{}`

735 `\newcommand*{\Rightsidehook}{}`

736 `\newcommand*{\Rightsidehookend}{}`

737

`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.

```
738 \newenvironment{Rightside}{%
739   \ledRcoltrue
740   \let\beginnumbering\beginnumberingR
741   \let\endnumbering\endnumberingR
742   \let\pausenumbering\pausenumberingR
743   \let\resumenummering\resumenummeringR
744   \let\memorydump\memorydumpR
745   \let\thepstart\thepstartR
746   \let\pstart\pstartR
747   \let\pend\pendR
748   \let\lineation\lineationR
749   \Rightsidehook
750   \let\oldstanza\stanza
751   \renewcommand{\stanza}{\oldstanza\global\instanzaRtrue}
752 }{%
753   \ledRcolfalse
754   \let\stanza\oldstanza
755   \Rightsidehookend
756 }
757
```

## 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.

```
758 \newcount\num@linesR
759 \newbox\one@lineR
760 \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. The old counters are used to have the good reset of the pstart counters at the begining of the \Pages command.

```
761
762 \newcounter{pstartL}
763 \newcounter{pstartLold}
764 \renewcommand{\thepstartL}{\bfseries\@arabic{c@pstartL}. }
765 \newcounter{pstartR}
766 \newcounter{pstartRold}
767 \renewcommand{\thepstartR}{\bfseries\@arabic{c@pstartR}. }
768
769 \newcommand*{\pstartL}{
770 \if@nobreak
771 \let\@oldnobreak\@nobreaktrue
772 \else
773 \let\@oldnobreak\@nobreakfalse
774 \fi
775 \@nobreaktrue
776 \ifnumbering \else
777 \led@err@PstartNotNumbered
778 \beginnumbering
779 \fi
780 \ifnumberedpar@
781 \led@err@PstartInPstart
782 \pend
783 \fi
```

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

```
784 \ifpst@rtedL\else
785 \setcounter{pstartLold}{\value{pstartL}}%
```

```

786 \list@clear{\inserts@list}%
787 \global\let\next@insert=\empty
788 \global\pst@rtedLtrue
789 \fi
790 \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.

```

791 \global\advance\l@dnumpstartsL \@ne
792 \ifnum\l@dnumpstartsL>\l@dc@maxchunks
793 \led@err@TooManyPstarts
794 \global\l@dnumpstartsL=\l@dc@maxchunks
795 \fi
796 \global\setnamebox{\l@dLcolrawbox\the\l@dnumpstartsL}=\vbox\bgroup\ifautopar\else\ifnumb
797 \hsize=\Lcolwidth
798 \numberedpar@true}

799 \newcommand*{\pstartR}{
800 \if@nbreak
801 \let\@oldnbreak\@nbreaktrue
802 \else
803 \let\@oldnbreak\@nbreakfalse
804 \fi
805 \@nbreaktrue
806 \ifnumberingR \else
807 \led@err@PstartNotNumbered
808 \beginnumberingR
809 \fi
810 \ifnumberedpar@
811 \led@err@PstartInPstart
812 \pendR
813 \fi
814 \ifpst@rtedR\else
815 \setcounter{pstartRold}{\value{pstartR}}%
816 \list@clear{\inserts@listR}%
817 \global\let\next@insertR=\empty
818 \global\pst@rtedRtrue
819 \fi
820 \begingroup\normal@pars
821 \global\advance\l@dnumpstartsR \@ne
822 \ifnum\l@dnumpstartsR>\l@dc@maxchunks
823 \led@err@TooManyPstarts
824 \global\l@dnumpstartsR=\l@dc@maxchunks
825 \fi
826 \global\setnamebox{\l@dRcolrawbox\the\l@dnumpstartsR}=\vbox\bgroup\ifautopar\else\ifnumb
827 \hsize=\Rcolwidth
828 \numberedpar@true}

```

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

```

829 \newcommand*{\pendL}{\ifnumbering \else
830   \led@err@PendNotNumbered
831   \fi
832   \ifnumberedpar@ \else
833   \led@err@PendNoPstart
834   \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.

```

835   \l@dzeropenalties
836   \endgraf\global\num@lines=\prevgraf\egroup
837   \global\par@line=0

```

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

```

838   \endgroup
839   \ignorespaces
840   \@oldnobreak
841   \ifnumberpstart
842   \addtocounter{pstartL}{1}
843   \fi}
844

```

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

```

845 \newcommand*{\pendR}{\ifnumberingR \else
846   \led@err@PendNotNumbered
847   \fi
848   \ifnumberedpar@ \else
849   \led@err@PendNoPstart
850   \fi
851   \l@dzeropenalties
852   \endgraf\global\num@linesR=\prevgraf\egroup
853   \global\par@lineR=0
854   \endgroup
855   \ignorespaces
856   \@oldnobreak
857   \ifnumberpstart
858   \addtocounter{pstartR}{1}
859   \fi
860 }
861

```

## 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  
`\l@drightbox` of right text.

```

862 \newbox\l@dleftbox
863 \newbox\l@drightbox
864

\countLline We need to know the number of lines processed.
\countRline 865 \newcount\countLline
866 \countLline \z@
867 \newcount\countRline
868 \countRline \z@
869

\@donereallinesL We need to know the number of ‘real’ lines output (i.e., those that have been input
\@donetotallinesL by the user), and the total lines output (which includes any blank lines output for
\@donereallinesR synchronisation).
\@donetotallinesR 870 \newcount\@donereallinesL
871 \newcount\@donetotallinesL
872 \newcount\@donereallinesR
873 \newcount\@donetotallinesR
874

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

875 \newcommand*{\do@lineL}{%
876 \advance\countLline \@ne
877 \ifvbox\namebox{\l@dLcolrawbox\the\l@dpscL}%
878 {\vbadness=10000
879 \splittopskip=\z@
880 \do@lineLhook
881 \l@emptyd@ta
882 \global\setbox\one@line=\vsplit\namebox{\l@dLcolrawbox\the\l@dpscL}
883 to\baselineskip}%
884 \unvbox\one@line \global\setbox\one@line=\lastbox
885 \getline@numL
886 \setbox\l@dleftbox
887 \hb@xt@ \Lcolwidth{%
888 \affixpstart@numL
889 \affixline@num
890 \l@dld@ta
891 \add@inserts
892 \affixside@note
893 \l@dlsn@te
894 {\ledllfill\hb@xt@ \wd\one@line{\inserthangingsymbolL\new@line\l@dunhbox@line{\one@line
895 \l@drsn@te
896 }}}%
897 \prevline@num=\line@num%
898 \add@penaltiesL
899 \global\advance\@donereallinesL\@ne
900 \global\advance\@donetotallinesL\@ne

```

```

901 \else
902   \setbox\l@leftbox \hb@xt@ \Lcolwidth{\hspace*{\Lcolwidth}}%
903   \global\advance\@donetotallinesL\@ne
904 \fi}
905
906

```

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

```

\do@lineRhook 907 \newcommand*{\do@lineLhook}{}
               908 \newcommand*{\do@lineRhook}{}
               909

```

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

```

910 \newcommand*{\do@lineR}{%
911   \advance\countRline \@ne
912   \ifvbox\namebox{l@dRcolrawbox\the\l@dpscR}%
913   {\vbadness=10000
914     \splittopskip=\z@
915     \do@lineRhook
916     \l@emptyd@ta
917     \global\setbox\one@lineR=\vsplit\namebox{l@dRcolrawbox\the\l@dpscR}
918                               to\baselineskip}%
919   \unvbox\one@lineR \global\setbox\one@lineR=\lastbox
920   \getline@numR
921   \setbox\l@drightbox
922   \hb@xt@ \Rcolwidth{%
923     \affixpstart@numR
924     \affixline@numR
925     \l@dld@ta
926     \add@insertsR
927     \affixside@noteR
928     \l@dlsn@te
929     {\correcthangingR\ledllfill\hb@xt@ \wd\one@lineR{\inserthangingsymbolR\new@lineR\l@dunhbox@line{
930       \l@drsn@te
931     }}}%
932   \prevline@numR=\line@numR%
933   \add@penaltiesR
934   \global\advance\@donereallinesR\@ne
935   \global\advance\@donetotallinesR\@ne
936 \else
937   \setbox\l@drightbox \hb@xt@ \Rcolwidth{\hspace*{\Rcolwidth}}
938   \global\advance\@donetotallinesR\@ne
939 \fi}
940
941

```

### 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.

```

942 \newcommand*{\getline@numR}{%
943 \ifnumberline
944   \global\advance\absline@numR \@ne
945 \fi
946   \do@actionsR
947   \do@ballastR
948 \ifnumberline
949   \ifsublines@
950     \ifnum\sub@lockR<\tw@
951       \global\advance\subline@numR \@ne
952     \fi
953   \else
954     \ifnum\@lockR<\tw@
955       \global\advance\line@numR \@ne
956       \global\subline@numR \z@
957     \fi
958   \fi
959 \fi
960 }
961 \newcommand*{\getline@numL}{%
962 \ifnumberline
963   \global\advance\absline@num \@ne
964 \fi
965   \do@actions
966   \do@ballast
967 \ifnumberline
968   \ifsublines@
969     \ifnum\sub@lock<\tw@
970       \global\advance\subline@num \@ne
971     \fi
972   \else
973     \ifnum\@lock<\tw@
974       \global\advance\line@num \@ne
975       \global\subline@num \z@
976     \fi
977   \fi
978 \fi
979 }
980
981

```

`\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.

```

982 \newcommand*{\do@ballastR}{\global\ballast@count=\z@
983   \begingroup
984     \advance\absline@numR \@ne

```



```

985     \ifnum\next@actionlineR=\absline@numR
986     \ifnum\next@actionR>-1001
987         \global\advance\ballast@count by -\c@ballast
988     \fi
989     \fi
990 \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 `\do@actions@fixedcodeR` line.  
`\do@actions@nextR`

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

```

991 \newcommand*{\do@actions@fixedcodeR}{%
992     \ifcase\@l@dttempcnta%
993     \or%                % 1001
994         \global\sublines@true
995     \or%                % 1002
996         \global\sublines@false
997     \or%                % 1003
998         \global\@lockR=\@ne
999     \or%                % 1004
1000         \ifnum\@lockR=\tw@
1001             \global\@lockR=\thr@@
1002         \else
1003             \global\@lockR=\z@
1004         \fi
1005     \or%                % 1005
1006         \global\sub@lockR=\@ne
1007     \or%                % 1006
1008         \ifnum\sub@lockR=\tw@
1009             \global\sub@lockR=\thr@@
1010         \else
1011             \global\sub@lockR=\z@
1012         \fi
1013     \or%                % 1007
1014         \l@dskipnumbertrue
1015     \else
1016         \led@warn@BadAction
1017     \fi}
1018
1019
1020 \newcommand*{\do@actionsR}{%
1021     \global\let\do@actions@nextR=\relax
1022     \@l@dttempcntb=\absline@numR
1023     \ifnum\@l@dttempcntb<\next@actionlineR\else
1024         \ifnum\next@actionR>-1001\relax
1025             \global\page@numR=\next@actionR
1026             \ifbypage@R
1027                 \global\line@numR \z@ \global\subline@numR \z@

```

```

1028     \fi
1029   \else
1030     \ifnum\next@actionR<-4999\relax    % 9/05 added relax here
1031     \@l@tempcnta=-\next@actionR
1032     \advance\@l@tempcnta by -5001\relax
1033     \ifsublines@
1034       \global\subline@numR=\@l@tempcnta
1035     \else
1036       \global\line@numR=\@l@tempcnta
1037     \fi
1038   \else
1039     \@l@tempcnta=-\next@actionR
1040     \advance\@l@tempcnta by -1000\relax
1041     \do@actions@fixedcodeR
1042   \fi
1043 \fi
1044 \ifx\actionlines@listR\empty
1045   \gdef\next@actionlineR{1000000}%
1046 \else
1047   \glp\actionlines@listR\to\next@actionlineR
1048   \glp\actions@listR\to\next@actionR
1049   \global\let\do@actions@nextR=\do@actionsR
1050 \fi
1051 \fi
1052 \do@actions@nextR}
1053

```

#### 14.4 Line number printing

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

```

\ch@cksub@l@ckR 1054
\ch@ck@l@ckR 1055 \providecommand*\l@dcalcnun}[3]{%
\fx@l@cksR 1056   \ifnum #1 > #2\relax
\affixline@numR 1057     \@l@tempcnta = #1\relax
1058     \advance\@l@tempcnta by -#2\relax
1059     \divide\@l@tempcnta by #3\relax
1060     \multiply\@l@tempcnta by #3\relax
1061     \advance\@l@tempcnta by #2\relax
1062   \else
1063     \@l@tempcnta=#2\relax
1064   \fi}
1065
1066 \newcommand*\ch@cksub@l@ckR}{%
1067   \ifcase\sub@lockR
1068   \or
1069     \ifnum\sublock@disp=\@ne
1070       \@l@tempcntb \z@ \@l@tempcnta \@ne
1071     \fi
1072   \or

```

```

1073 \ifnum\sublock@disp=\tw@
1074 \else
1075 \@l@tempcntb \z@ \@l@tempcnta \@ne
1076 \fi
1077 \or
1078 \ifnum\sublock@disp=\z@
1079 \@l@tempcntb \z@ \@l@tempcnta \@ne
1080 \fi
1081 \fi}
1082
1083 \newcommand*{\ch@ck@l@ckR}{%
1084 \ifcase\@lockR
1085 \or
1086 \ifnum\lock@disp=\@ne
1087 \@l@tempcntb \z@ \@l@tempcnta \@ne
1088 \fi
1089 \or
1090 \ifnum\lock@disp=\tw@
1091 \else
1092 \@l@tempcntb \z@ \@l@tempcnta \@ne
1093 \fi
1094 \or
1095 \ifnum\lock@disp=\z@
1096 \@l@tempcntb \z@ \@l@tempcnta \@ne
1097 \fi
1098 \fi}
1099
1100 \newcommand*{\f@x@l@cksR}{%
1101 \ifcase\@lockR
1102 \or
1103 \global\@lockR \tw@
1104 \or \or
1105 \global\@lockR \z@
1106 \fi
1107 \ifcase\sub@lockR
1108 \or
1109 \global\sub@lockR \tw@
1110 \or \or
1111 \global\sub@lockR \z@
1112 \fi}
1113
1114
1115 \newcommand*{\affixline@numR}{%
1116 \ifnumberline
1117 \ifl@dskipnumber
1118 \global\l@dskipnumberfalse
1119 \else
1120 \ifsublines@
1121 \@l@tempcntb=\subline@numR
1122 \l@dcalcnum{\subline@numR}{\c@firstsublinenumR}{\c@sublinenumincrementR}%

```

```

1123 \ch@cksub@lockR
1124 \else
1125 \l@tempcntb=\line@numR
1126 \ifx\linenumberlist\empty
1127 \l@dcalcnum{\line@numR}{\c@firstlinenumR}{\c@linenumincrementR}%
1128 \else
1129 \l@tempcnta=\line@numR
1130 \edef\rem@inder{\linenumberlist,\number\line@numR,%
1131 \edef\sc@n@list{\def\noexpand\sc@n@list
1132 ###1,\number\l@tempcnta,###2|{\def\noexpand\rem@inder{###2}}}%
1133 \sc@n@list\expandafter\sc@n@list\rem@inder|}%
1134 \ifx\rem@inder\empty\advance\l@tempcnta\@ne\fi
1135 \fi
1136 \ch@ck@l@ckR
1137 \fi
1138 \ifnum\l@tempcnta=\l@tempcntb
1139 \if@twocolumn
1140 \if@firstcolumn
1141 \gdef\l@dld@ta{\llap{\leftlinenumR}}}%
1142 \else
1143 \gdef\l@drd@ta{\rlap{\rightlinenumR}}}%
1144 \fi
1145 \else
1146 \l@tempcntb=\line@marginR
1147 \ifnum\l@tempcntb>\@ne
1148 \advance\l@tempcntb by\page@numR
1149 \fi
1150 \ifodd\l@tempcntb
1151 \gdef\l@drd@ta{\rlap{\rightlinenumR}}}%
1152 \else
1153 \gdef\l@dld@ta{\llap{\leftlinenumR}}}%
1154 \fi
1155 \fi
1156 \fi
1157 \f@x@l@cksR
1158 \fi
1159 \fi}

```

## 14.5 Pstart number printing in side

The printing of the pstart number is like in ledmac, with two differences :

- Some commands have versions suffixed by R or L.
- The `\affixpstart@num` and `\affixpstart@numR` commands are called in the `\Pages` command. Consequently, the `pstartL` and `pstartR` counters must be reset at the beginning of this command.

```

\affixpstart@numL
\affixpstart@numR 1160
\leftpstartnumR
\rightpstartnumR
\leftpstartnumL
\rightpstartnumL
\ifpstartnumR

```

```

1161 \newcommand*{\affixpstart@numL}{%
1162 \ifsidepstartnum
1163 \if@twocolumn
1164     \if@firstcolumn
1165         \gdef\l@dld@ta{\llap{\leftpstartnumL}}}%
1166     \else
1167         \gdef\l@drd@ta{\rlap{\rightpstartnumL}}}%
1168     \fi
1169 \else
1170     \l@dttempcntb=\line@margin
1171     \ifnum\l@dttempcntb>\@ne
1172         \advance\l@dttempcntb \page@num
1173     \fi
1174     \ifodd\l@dttempcntb
1175         \gdef\l@drd@ta{\rlap{\rightpstartnumL}}}%
1176     \else
1177         \gdef\l@dld@ta{\llap{\leftpstartnumL}}}%
1178     \fi
1179 \fi
1180 \fi
1181 }
1182 \newcommand*{\affixpstart@numR}{%
1183 \ifsidepstartnum
1184 \if@twocolumn
1185     \if@firstcolumn
1186         \gdef\l@dld@ta{\llap{\leftpstartnumR}}}%
1187     \else
1188         \gdef\l@drd@ta{\rlap{\rightpstartnumR}}}%
1189     \fi
1190 \else
1191     \l@dttempcntb=\line@marginR
1192     \ifnum\l@dttempcntb>\@ne
1193         \advance\l@dttempcntb \page@numR
1194     \fi
1195     \ifodd\l@dttempcntb
1196         \gdef\l@drd@ta{\rlap{\rightpstartnumR}}}%
1197     \else
1198         \gdef\l@dld@ta{\llap{\leftpstartnumR}}}%
1199     \fi
1200 \fi
1201 \fi
1202 }
1203
1204 \newcommand*{\leftpstartnumL}{
1205 \ifpstartnum
1206 \thepstartL
1207 \kern\linenumsep\global\pstartnumfalse\fi
1208 }
1209 \newcommand*{\rightpstartnumL}{
1210 \ifpstartnum\kern\linenumsep

```

```

1211 \thepstartL
1212 \global\pstartnumfalse\fi
1213 }
1214 \newif\ifpstartnumR
1215 \pstartnumRtrue
1216 \newcommand*{\leftpstartnumR}{
1217 \ifpstartnumR
1218 \thepstartR
1219 \kern\linenumsep\global\pstartnumRfalse\fi
1220 }
1221 \newcommand*{\rightpstartnumR}{
1222 \ifpstartnumR\kern\linenumsep
1223 \thepstartR
1224 \global\pstartnumRfalse\fi
1225 }

```

## 14.6 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.

```

1226 \list@create{\inserts@listR}

```

`\add@insertsR` The right text version.

```

\add@inserts@nextR 1227 \newcommand*{\add@insertsR}{%
1228 \global\let\add@inserts@nextR=\relax
1229 \ifx\inserts@listR\empty \else
1230 \ifx\next@insertR\empty
1231 \ifx\insertlines@listR\empty
1232 \global\noteschanged@true
1233 \gdef\next@insertR{100000}%
1234 \else
1235 \gl@p\insertlines@listR\to\next@insertR
1236 \fi
1237 \fi
1238 \ifnum\next@insertR=\absline@numR
1239 \gl@p\inserts@listR\to\@insertR
1240 \@insertR
1241 \global\let\@insertR=\undefined
1242 \global\let\next@insertR=\empty
1243 \global\let\add@inserts@nextR=\add@insertsR
1244 \fi
1245 \fi
1246 \add@inserts@nextR}
1247

```

## 14.7 Penalties

`\add@penaltiesL` `\add@penaltiesL` is the last macro used by `\do@lineL`. It adds up the club, `\add@penaltiesR` widow, and interline penalties, and puts a single penalty of the appropriate size

back into the paragraph; these penalties get removed by the `\vsplit` operation. `\displaywidowpenalty` and `\brokenpenalty` are not restored, since we have no easy way to find out where we should insert them.

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

```
\newcommand*{\add@penaltiesR}{\@l@dttempcnta=\ballast@count
\ifnum\num@linesR>\@ne
\global\advance\par@lineR \@ne
\ifnum\par@lineR=\@ne
\advance\@l@dttempcnta by \clubpenalty
\fi
\@l@dttempcntb=\par@lineR \advance\@l@dttempcntb \@ne
\ifnum\@l@dttempcntb=\num@linesR
\advance\@l@dttempcnta by \widowpenalty
\fi
\ifnum\par@lineR<\num@linesR
\advance\@l@dttempcnta by \interlinepenalty
\fi
\fi
\ifnum\@l@dttempcnta=\z@
\relax
\else
\ifnum\@l@dttempcnta>-10000
\penalty\@l@dttempcnta
\else
\penalty -10000
\fi
\fi}
```

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

```
1248 \newcommand*{\add@penaltiesL}{\}
1249 \newcommand*{\add@penaltiesR}{\}
1250
```

## 14.8 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.

```
1251 \newcommand*{\flush@notesR}{%
```

```

1252 \@xloop
1253   \ifx\inserts@listR\empty \else
1254     \glp\inserts@listR\to\@insertR
1255     \@insertR
1256     \global\let\@insertR=\undefined
1257   \repeat}
1258

```

## 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.

```

1259 \renewcommand*{\Afootnote}[1]{%
1260   \ifnumberedpar@
1261     \ifledRcol
1262       \xright@appenditem{\noexpand\Afootnote{A}%
1263         {{\l@d@nums}{\@tag}{#1}}}\to\inserts@listR
1264       \global\advance\insert@countR \@ne
1265     \else
1266       \xright@appenditem{\noexpand\Afootnote{A}%
1267         {{\l@d@nums}{\@tag}{#1}}}\to\inserts@list
1268       \global\advance\insert@count \@ne
1269     \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`.

```

1270   \else
1271     \Afootnote{A}{{0|0|0|0|0|0|0|0}{\@tag}{#1}}%
1272   \fi\ignorespaces}

```

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

```

\Cfootnote 1273 \renewcommand*{\Bfootnote}[1]{%
\Dfootnote 1274   \ifnumberedpar@
\Efootnote 1275     \ifledRcol
1276       \xright@appenditem{\noexpand\Bfootnote{B}%
1277         {{\l@d@nums}{\@tag}{#1}}}\to\inserts@listR
1278       \global\advance\insert@countR \@ne
1279     \else
1280       \xright@appenditem{\noexpand\Bfootnote{B}%
1281         {{\l@d@nums}{\@tag}{#1}}}\to\inserts@list
1282       \global\advance\insert@count \@ne

```



```

1283 \fi
1284 \else
1285 \vBfootnote{B}{\{0|0|0|0|0|0|0|0\}-\{#1\}}%
1286 \fi\ignorespaces}
1287 \renewcommand*{\Cfootnote}[1]{%
1288 \ifnumberedpar@
1289 \ifledRcol
1290 \xright@appenditem{\noexpand\vCfootnote{C}%
1291 {\l@d@nums}\@tag{#1}}\to\inserts@listR
1292 \global\advance\insert@countR \@ne
1293 \else
1294 \xright@appenditem{\noexpand\vCfootnote{C}%
1295 {\l@d@nums}\@tag{#1}}\to\inserts@list
1296 \global\advance\insert@count \@ne
1297 \fi
1298 \else
1299 \vCfootnote{C}{\{0|0|0|0|0|0|0|0\}-\{#1\}}%
1300 \fi\ignorespaces}
1301 \renewcommand*{\Dfootnote}[1]{%
1302 \ifnumberedpar@
1303 \ifledRcol
1304 \xright@appenditem{\noexpand\vDfootnote{D}%
1305 {\l@d@nums}\@tag{#1}}\to\inserts@listR
1306 \global\advance\insert@countR \@ne
1307 \else
1308 \xright@appenditem{\noexpand\vDfootnote{D}%
1309 {\l@d@nums}\@tag{#1}}\to\inserts@list
1310 \global\advance\insert@count \@ne
1311 \fi
1312 \else
1313 \vDfootnote{D}{\{0|0|0|0|0|0|0|0\}-\{#1\}}%
1314 \fi\ignorespaces}
1315 \renewcommand*{\Efootnote}[1]{%
1316 \ifnumberedpar@
1317 \ifledRcol
1318 \xright@appenditem{\noexpand\vEfootnote{E}%
1319 {\l@d@nums}\@tag{#1}}\to\inserts@listR
1320 \global\advance\insert@countR \@ne
1321 \else
1322 \xright@appenditem{\noexpand\vEfootnote{E}%
1323 {\l@d@nums}\@tag{#1}}\to\inserts@list
1324 \global\advance\insert@count \@ne
1325 \fi
1326 \else
1327 \vEfootnote{E}{\{0|0|0|0|0|0|0|0\}-\{#1\}}%
1328 \fi\ignorespaces}
1329

```

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

`\mpBfootnote`

`\mpCfootnote`

`\mpDfootnote`

`\mpEfootnote`

```

1330 \renewcommand*{\mpAfootnote}[1]{%
1331   \ifnumberedpar@
1332   \iflabeledRcol
1333     \xright@appenditem{\noexpand\mpvAfootnote{A}%
1334       {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1335     \global\advance\insert@countR \@ne
1336   \else
1337     \xright@appenditem{\noexpand\mpvAfootnote{A}%
1338       {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1339     \global\advance\insert@count \@ne
1340   \fi
1341   \else
1342     \mpvAfootnote{A}{\{0|0|0|0|0|0|0\}}{#1}%
1343   \fi\ignorespaces}

1344 \renewcommand*{\mpBfootnote}[1]{%
1345   \ifnumberedpar@
1346   \iflabeledRcol
1347     \xright@appenditem{\noexpand\mpvBfootnote{B}%
1348       {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1349     \global\advance\insert@countR \@ne
1350   \else
1351     \xright@appenditem{\noexpand\mpvBfootnote{B}%
1352       {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1353     \global\advance\insert@count \@ne
1354   \fi
1355   \else
1356     \mpvBfootnote{B}{\{0|0|0|0|0|0|0\}}{#1}%
1357   \fi\ignorespaces}

1358 \renewcommand*{\mpCfootnote}[1]{%
1359   \ifnumberedpar@
1360   \iflabeledRcol
1361     \xright@appenditem{\noexpand\mpvCfootnote{C}%
1362       {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1363     \global\advance\insert@countR \@ne
1364   \else
1365     \xright@appenditem{\noexpand\mpvCfootnote{C}%
1366       {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1367     \global\advance\insert@count \@ne
1368   \fi
1369   \else
1370     \mpvCfootnote{C}{\{0|0|0|0|0|0|0\}}{#1}%
1371   \fi\ignorespaces}

1372 \renewcommand*{\mpDfootnote}[1]{%
1373   \ifnumberedpar@
1374   \iflabeledRcol
1375     \xright@appenditem{\noexpand\mpvDfootnote{D}%
1376       {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1377     \global\advance\insert@countR \@ne
1378   \else

```

```

1379 \xright@appenditem{\noexpand\mpvDfootnote{D}%
1380                 {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1381 \global\advance\insert@count \@ne
1382 \fi
1383 \else
1384 \mpvDfootnote{D}{\l@d@nums}{\@tag}{#1}%
1385 \fi\ignorespaces}

1386 \renewcommand*{\mpEfootnote}[1]{%
1387 \ifnumberedpar@
1388 \ifledRcol
1389 \xright@appenditem{\noexpand\mpvEfootnote{E}%
1390                 {\l@d@nums}{\@tag}{#1}}\to\inserts@listR
1391 \global\advance\insert@countR \@ne
1392 \else
1393 \xright@appenditem{\noexpand\mpvEfootnote{E}%
1394                 {\l@d@nums}{\@tag}{#1}}\to\inserts@list
1395 \global\advance\insert@count \@ne
1396 \fi
1397 \else
1398 \mpvEfootnote{E}{\l@d@nums}{\@tag}{#1}%
1399 \fi\ignorespaces}

\l@dedendmini

1400 \renewcommand*{\l@dedendmini}{%
1401 \ifl@dpairing
1402 \ifledRcol
1403 \flush@notesR
1404 \else
1405 \flush@notes
1406 \fi
1407 \fi
1408 \ifvoid\mpAfootins\else\mpAfootgroup{A}\fi%
1409 \ifvoid\mpBfootins\else\mpBfootgroup{B}\fi%
1410 \ifvoid\mpCfootins\else\mpCfootgroup{C}\fi%
1411 \ifvoid\mpDfootins\else\mpDfootgroup{D}\fi%
1412 \ifvoid\mpEfootins\else\mpEfootgroup{E}\fi}

```

## 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
1413 \def\printlinesR#1|#2|#3|#4|#5|#6|#7|{\begingroup
1414   \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
1415   \ifl@d@pnum #1\fullstop\fi
1416   \ifl@dplinenum \linenumr@p{#2}\Rlineflag\else \sympmlinenum\fi
1417   \ifl@d@ssub \fullstop \sublinenumr@p{#3}\fi
1418   \ifl@d@dash \endashchar\fi
1419   \ifl@d@pnum #4\fullstop\fi
1420   \ifl@d@elin \linenumr@p{#5}\Rlineflag\fi
1421   \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumr@p{#6}\fi
1422 \endgroup}
1423
1424 \let\ledsavedprintlines\printlines
1425

```

## 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.

```

1426 \list@create{\labelref@listR}
1427

```

`\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.

```

1428 \renewcommand*{\edlabel}[1]{\@bsphack
1429   \ifl@dRcol
1430     \write\linenum@outR{\string\@lab}%
1431     \ifx\labelref@listR\empty
1432       \xdef\label@refs{\zz@@@}%
1433     \else
1434       \gl@p\labelref@listR\to\label@refs
1435     \fi
1436     \ifvmode
1437       \advancelabel@refs
1438     \fi
1439     \protected@write\@auxout{%
1440       {\string\l@dmake@labelsR\space\thepage|\label@refs|{#1}}%
1441   \else
1442     \write\linenum@out{\string\@lab}%
1443     \ifx\labelref@listR\empty
1444       \xdef\label@refs{\zz@@@}%
1445     \else
1446       \gl@p\labelref@listR\to\label@refs

```

```

1447 \fi
1448 \ifvmode
1449 \advancelabel@refs
1450 \fi
1451 \protected@write\@auxout{}%
1452 {\string\l@dmake@labels\space\thepage|\label@refs|{#1}}%
1453 \fi
1454 \@esphack}
1455

```

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

```

1456 \def\l@dmake@labelsR#1|#2|#3|#4{%
1457 \expandafter\ifx\csname the@label#4\endcsname \relax\else
1458 \led@warn@DuplicateLabel{#4}%
1459 \fi
1460 \expandafter\gdef\csname the@label#4\endcsname{#1|#2\Rlineflag|#3}%
1461 \ignorespaces}
1462 \AtBeginDocument{%
1463 \def\l@dmake@labelsR#1|#2|#3|#4{%
1464 }
1465

```

`\@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.

```

1466 \renewcommand*{\@lab}{%
1467 \ifledRcol
1468 \xright@appenditem{\linenumr@p{\line@numR}}{|%
1469 \ifsublines@ \sublinenumr@p{\subline@numR}\else 0\fi}%
1470 \to\labelref@listR
1471 \else
1472 \xright@appenditem{\linenumr@p{\line@num}}{|%
1473 \ifsublines@ \sublinenumr@p{\subline@num}\else 0\fi}%
1474 \to\labelref@list
1475 \fi}
1476

```

## 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 1477 \newcount\sidenote@marginR
1478 \renewcommand*{\sidenotemargin}[1]{%
1479 \l@dgetsidenote@margin{#1}%
1480 \ifnum\@l@dttempcntb>\m@ne

```

```

1481 \ifledRcol
1482 \global\sidenote@marginR=\@l@tempcntb
1483 \else
1484 \global\sidenote@margin=\@l@tempcntb
1485 \fi
1486 \fi}}
1487 \sidenotemargin{right}
1488 \global\sidenote@margin=\@ne
1489

```

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

```

\l@dcsnote 1490 \renewcommand*{\l@dlsnote}[1]{%
1491 \ifnumberedpar@
1492 \ifledRcol
1493 \xright@appenditem{\noexpand\l@dlsnote{#1}}%
1494 \to\inserts@listR
1495 \global\advance\insert@countR \@ne
1496 \else
1497 \xright@appenditem{\noexpand\l@dlsnote{#1}}%
1498 \to\inserts@list
1499 \global\advance\insert@count \@ne
1500 \fi
1501 \fi\ignorespaces}
1502 \renewcommand*{\l@drsnote}[1]{%
1503 \ifnumberedpar@
1504 \ifledRcol
1505 \xright@appenditem{\noexpand\l@drsnote{#1}}%
1506 \to\inserts@listR
1507 \global\advance\insert@countR \@ne
1508 \else
1509 \xright@appenditem{\noexpand\l@drsnote{#1}}%
1510 \to\inserts@list
1511 \global\advance\insert@count \@ne
1512 \fi
1513 \fi\ignorespaces}
1514 \renewcommand*{\l@dcsnote}[1]{%
1515 \ifnumberedpar@
1516 \ifledRcol
1517 \xright@appenditem{\noexpand\l@dcsnote{#1}}%
1518 \to\inserts@listR
1519 \global\advance\insert@countR \@ne
1520 \else
1521 \xright@appenditem{\noexpand\l@dcsnote{#1}}%
1522 \to\inserts@list
1523 \global\advance\insert@count \@ne
1524 \fi
1525 \fi\ignorespaces}
1526

```

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

```

1527 \newcommand*{\affixside@noteR}{%
1528   \gdef\@templ@d{%
1529     \ifx\@templ@d\l@dcstotetext \else
1530       \if@twocolumn
1531         \if@firstcolumn
1532           \setl@dlp@rbox{\l@dcstotetext}%
1533         \else
1534           \setl@drp@rbox{\l@dcstotetext}%
1535         \fi
1536       \else
1537         \@l@tempcntb=\sidenote@marginR
1538         \ifnum\@l@tempcntb>\@ne
1539           \advance\@l@tempcntb by\page@num
1540         \fi
1541         \ifodd\@l@tempcntb
1542           \setl@drp@rbox{\l@dcstotetext}%
1543         \else
1544           \setl@dlp@rbox{\l@dcstotetext}%
1545         \fi
1546       \fi
1547     \fi}
1548
```

## 18 Familiar footnotes

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

```

1549 \renewcommand{\l@dbfnote}[1]{%
1550   \ifnumberedpar@
1551     \ifledRcol
1552       \xright@appenditem{\noexpand\vl@dbfnote{#1}}{\@thefnmark}}%
1553       \to\inserts@listR
1554     \global\advance\insert@countR \@ne
1555   \else
1556     \xright@appenditem{\noexpand\vl@dbfnote{#1}}{\@thefnmark}}%
1557     \to\inserts@list
1558   \global\advance\insert@count \@ne
1559   \fi
1560 \fi\ignorespaces}
1561
```

`\normalbfnoteX`

```

1562 \renewcommand{\normalbfnoteX}[2]{%
1563   \ifnumberedpar@
1564     \ifledRcol
1565       \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}}{\@nameuse{thefootnote#1}}%
1566       \to\inserts@listR

```

```

1567     \global\advance\insert@countR \@ne
1568     \else
1569         \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}{\@nameuse{thefootnote#1}}}%
1570                             \to\inserts@list
1571     \global\advance\insert@count \@ne
1572     \fi
1573     \fi\ignorespaces}
1574

```

## 19 Verse

Like in ledmac, the insertion of hangingsymbol is base on a control of the change of line's numbers. But we need different macros for left and right sides.

```

\prevline@numR
\inserthangingsymbolL 1575 \newcount\prevline@numR
\inserthangingsymbolR 1576 \newcommand{\inserthangingsymbolL}{\ifinstanzaL\ifnum\line@num=\prevline@num\hfill\hang
1577 \newcommand{\inserthangingsymbolR}{\ifinstanzaR\ifnum\line@numR=\prevline@numR\hfill\hang

```

When a verse is hanged, the column separator is shifted. To prevent it, the `\do@lineL` and `\do@lineR` commands call `\correcthangingL` and `\correcthangingR` commands. These commands insert horizontal skip which length is equal to the hang indent.

```

\correcthangingL
\correcthangingR 1578 \newcommand{\correcthangingL}{\ifl@dpaging\else\ifinstanzaL\ifnum\line@num=\prevline@num\l
1579                 \noexpand\csname sza@0@\endcsname\stanzaindentbase\fi\fi\fi}
1580
1581 \newcommand{\correcthangingR}{\ifl@dpaging\else\ifinstanzaR\ifnum\line@numR=\prevline@numR\l
1582                 \noexpand\csname sza@0@\endcsname\stanzaindentbase\fi\fi\fi}

```

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

```

1583     \chardef\next=\catcode'\&
1584     \catcode'\&=\active
1585

```

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

```

1586 \newenvironment{astanza}{%
1587     \startstanzahook
1588     \catcode'\&\active
1589     \global\stanza@count\@ne
1590     \ifnum\usenamecount{sza@0@}=\z@
1591         \let\stanza@hang\relax
1592         \let\endlock\relax
1593     \else

```



```

1594 %% \interlinepenalty\@M % this screws things up, but I don't know why
1595 \rightskip\z@ plus 1fil\relax
1596 \fi
1597 \ifnum\usernamecount{szp@0@}=\z@
1598 \let\sza@penalty\relax
1599 \fi
1600 \def&{%
1601 \endlock\mbox{}}%
1602 \sza@penalty
1603 \global\advance\stanza@count\@ne
1604 \@astanza@line}%
1605 \def\&{%
1606 \endlock\mbox{}}
1607 \pend
1608 \endstanzaextra}%
1609 \pstart
1610 \@astanza@line
1611 }{}
1612

```

`\@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.

```

1613 \newcommand*{\@astanza@line}{%
1614 \parindent=\csname szp@\number\stanza@count @\endcsname\stanzaindentbase
1615 \par
1616 \stanza@hang%\mbox{}}%
1617 \ignorespaces}
1618

```

Lastly reset the modified category codes.

```

1619 \catcode'\&=\next
1620

```

## 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 1621 \providecommand*{\newnamebox}[1]{%
\unvnamebox 1622 \expandafter\newbox\csname #1\endcsname}
\namebox 1623 \providecommand*{\setnamebox}[1]{%
1624 \expandafter\setbox\csname #1\endcsname}
1625 \providecommand*{\unhnamebox}[1]{%
1626 \expandafter\unhbox\csname #1\endcsname}
1627 \providecommand*{\unvnamebox}[1]{%

```

```

1628 \expandafter\unvbox\csname #1\endcsname}
1629 \providecommand*\namebox}[1]{%
1630     \csname #1\endcsname}
1631

```

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

```

\usenamecount 1632 \providecommand*\newnamecount}[1]{%
1633     \expandafter\newcount\csname #1\endcsname}
1634 \providecommand*\usenamecount}[1]{%
1635     \csname #1\endcsname}
1636

```

## 21 Counts and boxes for parallel texts

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

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

```

1637 \newcount\l@dc@maxchunks
1638 \newcommand*\maxchunks}[1]{\l@dc@maxchunks=#1}
1639 \maxchunks{10}
1640

```

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

```

\l@dnumpstartsR 1641 \newcount\l@dnumpstartsR
1642

```

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

```

\l@pscR 1643 \newcount\l@dpscL
1644 \newcount\l@dpscR
1645

```

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

```

1646 \newcommand*\l@dsetuprawboxes){%
1647     \@l@tempcntb=\l@dc@maxchunks
1648     \loop\ifnum\@l@tempcntb>\z@
1649         \newnamebox{l@dLcolrawbox\the\@l@tempcntb}
1650         \newnamebox{l@dRcolrawbox\the\@l@tempcntb}
1651         \advance\@l@tempcntb \m@ne
1652     \repeat}
1653

```

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

```

1654 \newcommand*{\l@dsetupmaxlinecounts}{%
1655   \l@dtempcntb=\l@dc@maxchunks
1656   \loop\ifnum\l@dtempcntb>\z@
1657     \newnamecount{l@dmaxlinesinpar\the\l@dtempcntb}
1658     \advance\l@dtempcntb \m@ne
1659   \repeat}
1660 \newcommand*{\l@dzeromaxlinecounts}{%
1661   \begingroup
1662   \l@dtempcntb=\l@dc@maxchunks
1663   \loop\ifnum\l@dtempcntb>\z@
1664     \global\usenamecount{l@dmaxlinesinpar\the\l@dtempcntb}=\z@
1665     \advance\l@dtempcntb \m@ne
1666   \repeat
1667   \endgroup}
1668
```

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

```

1669 \AtBeginDocument{%
1670   \l@dsetuprawboxes
1671   \l@dsetupmaxlinecounts
1672   \l@dzeromaxlinecounts
1673   \l@dnumstartsl=\z@
1674   \l@dnumstartsr=\z@
1675   \l@dpscl=\z@
1676   \l@dpscr=\z@}
1677
```

## 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 1678 \newif\ifl@dusedbabel
\l@dusedbabeltrue 1679 \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 1680 \newif\ifl@dsamelang
                  1681 \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.
                1682 \newcommand*{\l@dchecklang}{%
                1683 \l@dsamelangfalse
                1684 \edef\@tempa{\theledlanguageL}\edef\@temp{\theledlanguageR}%
                1685 \ifx\@tempa\@tempb
                1686 \l@dsamelangtrue
                1687 \fi}
                1688

\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 \.
                      1689 \newcommand*{\l@dbbl@set@language}[1]{%
                      1690 \edef\language{#1}%
                      1691 \select@language{\language}%
                      1692 \if@filesw
                      1693 \protected@write\@auxout{}\string\select@language{\language}%
                      1694 \addtocontents{toc}{\string\select@language{\language}%
                      1695 \addtocontents{lof}{\string\select@language{\language}%
                      1696 \addtocontents{lot}{\string\select@language{\language}%
                      1697 \fi}
                      1698

                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 1699 \providecommand{\selectlanguage}[1]{}
                  1700 \newcommand*{\l@duselanguage}[1]{}
                  1701 \gdef\theledlanguageL{}
                  1702 \gdef\theledlanguageR{}
                  1703

                Now do the babel fix or polyglossia, if necessary.
                1704 \AtBeginDocument{%
                1705 \ifundefined{xpg@main@language}{%
                1706 \ifundefined{bbl@main@language}{%

```

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

```
1707 \l@dusedbabelfalse
1708 \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.

```
1709 \l@dusedbabeltrue
1710 \let\l@doldselectlanguage\selectlanguage
1711 \let\l@doldbbl@set@language\bbl@set@language
1712 \let\bbl@set@language\l@dbbl@set@language
1713 \renewcommand{\selectlanguage}[1]{%
1714   \l@doldselectlanguage{#1}%
1715   \ifledRcol \gdef\theledlanguageR{#1}%
1716   \else      \gdef\theledlanguageL{#1}%
1717   \fi}
```

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

```
1718 \renewcommand*{\l@duselanguage}[1]{%
1719   \l@doldselectlanguage{#1}}
```

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

```
1720 \gdef\theledlanguageL{\bbl@main@language}%
1721 \gdef\theledlanguageR{\bbl@main@language}%
1722 }%
1723 }
```

If on Polyglossia

```
1724 { \apptocmd{\xpg@set@language}{%
1725   \ifledRcol \gdef\theledlanguageR{#1}%
1726   \else      \gdef\theledlanguageL{#1}%
1727   \fi}%
1728   \let\l@duselanguage\xpg@set@language
1729   \gdef\theledlanguageL{\xpg@main@language}%
1730   \gdef\theledlanguageR{\xpg@main@language}%
1731 % \end{macrocode}
1732 % That's it.
1733 % \begin{macrocode}
1734 }}
```

## 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.

```
1735 \newcommand*\Columns{%
1736   \setcounter{pstartL}{\value{pstartLold}}
1737   \setcounter{pstartR}{\value{pstartRold}}
```

```

1738 \ifnum\l@dnumstartsL=\l@dnumstartsR\else
1739 \led@err@BadLeftRightPstarts{\the\l@dnumstartsL}{\the\l@dnumstartsR}%
1740 \fi

```

Start a group and zero counters, etc.

```

1741 \begingroup
1742 \l@dzeropenalties
1743 \endgraf\global\num@lines=\prevgraf
1744 \global\num@linesR=\prevgraf
1745 \global\par@line=\z@
1746 \global\par@lineR=\z@
1747 \global\l@dpscL=\z@
1748 \global\l@dpscR=\z@

```

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

```

1749 \check@pstarts
1750 \loop\if@pstarts
1751 \global\pstartnumtrue
1752 \global\pstartnumRtrue

```

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

```

1753 \global\advance\l@dpscL \@ne
1754 \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

```

1755 \checkraw@text
1756 \l@dchecklang
1757 { \loop\ifaraw@text

```

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

```

1758 \ifl@dsamelang
1759 \do@lineL
1760 \do@lineR
1761 \else
1762 \l@duselanguage{\theledlanguageL}%
1763 \do@lineL
1764 \l@duselanguage{\theledlanguageR}%
1765 \do@lineR
1766 \fi
1767 \hb@xt@ \hsize{%
1768 \hfill \unhbox\l@dleftbox
1769 \hfill \columnseparator \hfill
1770 \unhbox\l@drightbox
1771 }%
1772 \checkraw@text
1773 \repeat}

```

Having completed a pair of chunks, write the number of lines in each chunk to the respective section files. Increment pstart counters and reset line numbering if it's by pstart.

```

1774      \@writelinesinparL
1775      \@writelinesinparR
1776      \check@pstarts
1777          \ifbypstart@
1778          \write\linenum@out{\string\@set[1]}
1779      \fi
1780      \ifbypstart@R
1781          \write\linenum@outR{\string\@set[1]}
1782      \fi
1783      \addtocounter{pstartL}{1}
1784      \addtocounter{pstartR}{1}
1785      \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.

```

1786      \flush@notes
1787      \flush@notesR
1788      \endgroup
1789      \global\l@dpscL=\z@
1790      \global\l@dpscR=\z@
1791      \global\l@dnumpstartsL=\z@
1792      \global\l@dnumpstartsR=\z@
1793      \ignorespaces
1794      \global\instanzaLfalse
1795      \global\instanzaRfalse}
1796

```

`\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).

```

1797 \newcommand*{\columnseparator}{%
1798   \smash{\rule[-0.2\baselineskip]{\columnrulewidth}{1.05\baselineskip}}}
1799 \newdimen\columnrulewidth
1800 \columnrulewidth=\z@
1801

```

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

```

\@pstartstrue 1802 \newif\if@pstarts
\@pstartsfalse 1803 \newcommand*{\check@pstarts}{%
\check@pstarts 1804   \@pstartsfalse
1805   \ifnum\l@dnumpstartsL>\l@dpscL
1806     \@pstartstrue
1807   \else
1808     \ifnum\l@dnumpstartsR>\l@dpscR
1809       \@pstartstrue

```

```

1810     \fi
1811     \fi
1812 }
1813

```

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

```

\checkraw@text 1814 \newif\ifaraw@text
1815     \araw@textfalse
1816 \newcommand*{\checkraw@text}{%
1817     \araw@textfalse
1818     \ifvbox\namebox{1@dLcolrawbox\the\1@dpscL}
1819         \araw@texttrue
1820     \else
1821         \ifvbox\namebox{1@dRcolrawbox\the\1@dpscR}
1822             \araw@texttrue
1823         \fi
1824     \fi
1825 }
1826

```

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

```

1827 \newcommand*{\@writelinesinparL}{%
1828     \edef\next{%
1829         \write\linenum@out{\string\@pend[\the\@donereallinesL]}}%
1830     \next
1831     \global\@donereallinesL \z@}
1832 \newcommand*{\@writelinesinparR}{%
1833     \edef\next{%
1834         \write\linenum@outR{\string\@pendR[\the\@donereallinesR]}}%
1835     \next
1836     \global\@donereallinesR \z@}
1837

```

## 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 1838 \newcount\numpagelinesL
1839 \newcount\numpagelinesR
1840 \newcount\l@dminpagelines
1841

```



`\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.

```

1842 \newcommand*{\Pages}{%
1843   \setcounter{pstartL}{\value{pstartLold}}
1844   \setcounter{pstartR}{\value{pstartRold}}
1845   \typeout{}
1846   \typeout{***** PAGES *****}
1847   \ifnum\l@dnumpstartsL=\l@dnumpstartsR\else
1848     \led@err@BadLeftRightPstarts{\the\l@dnumpstartsL}{\the\l@dnumpstartsR}%
1849   \fi

```

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

```

1850 \cleartol@evenpage
1851 \begingroup
1852   \l@dzeropenalties
1853   \endgraf\global\num@lines=\prevgraf
1854   \global\num@linesR=\prevgraf
1855   \global\par@line=\z@
1856   \global\par@lineR=\z@
1857   \global\l@dpscL=\z@
1858   \global\l@dpscR=\z@
1859   \writtenlinesLfalse
1860   \writtenlinesRfalse

```

Check if there are chunks to be processed.

```

1861   \check@pstarts
1862   \loop\if@pstarts

```

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

```

1863     \global\advance\l@dpscL \@ne
1864     \global\advance\l@dpscR \@ne

```

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

```

1865     \getlinesfromparlistL
1866     \getlinesfromparlistR
1867     \l@dcalc@maxoftwo{\@cs@linesinparL}{\@cs@linesinparR}%
1868     {\usenamecount{l@dmaxlinesinpar\the\l@dpscL}}%
1869     \check@pstarts
1870   \repeat

```

Zero the counts again, ready for the next bit.

```

1871   \global\l@dpscL=\z@
1872   \global\l@dpscR=\z@

```

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

```

1873   \getlinesfrompagelistL
1874   \getlinesfrompagelistR

```

```

1875 \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
1876 {\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.

```

1877 \check@pstarts
1878 \if@pstarts

```

Increment the chunk counts to get the first pair.

```

1879 \global\advance\l@dpscL \@ne
1880 \global\advance\l@dpscR \@ne

```

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

```

1881 \global\@donereallinesL=\z@
1882 \global\@donetotallinesL=\z@
1883 \global\@donereallinesR=\z@
1884 \global\@donetotallinesR=\z@

```

Start a loop over the boxes (chunks).

```

1885 \checkraw@text
1886 % \begingroup
1887 { \loop\ifaraw@text

```

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

```

1888 \checkpageL
1889 \l@duselanguage{\theledlanguageL}%
1890 %% \begingroup
1891 { \loop\ifl@dsamepage
1892

```

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

```

1893 \do@lineL
1894 \advance\numpagelinesL \@ne
1895 \ifshiftedverses
1896 \ifdim\ht\l@dleftbox>0pt\hb@xt@ \hsize{\ledstrutL\unhbox\l@dleftbox}\fi%
1897 \else
1898 \hb@xt@ \hsize{\ledstrutL\unhbox\l@dleftbox}%
1899 \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.

```

1900
1901 \get@nextboxL
1902 \checkpageL
1903 \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.

```

1904      \ifl@dpagfull
1905      \@writelinesonpageL{\the\numpagelinesL}%
1906      \else
1907      \@writelinesonpageL{1000}%
1908      \fi

```

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

```

1909      \numpagelinesL \z@
1910      \clearl@dleftpage }%

```

Now do the same for the right text.

```

1911      \checkpageR
1912      \l@duselanguage{\theledlanguageR}%
1913 {
1914      \loop\ifl@dsamepage
1915      \do@lineR
1916      \advance\numpagelinesR \@ne
1917      \ifshiftedverses
1918      \ifdim\ht\l@drightbox>0pt\hb@xt@ \hsize{\ledstrutR\unhbox\l@drightbox}\fi%
1919      \else
1920      \hb@xt@ \hsize{\ledstrutR\unhbox\l@drightbox}%
1921      \fi
1922      \get@nextboxR
1923      \checkpageR
1924      \repeat
1925      \ifl@dpagfull
1926      \@writelinesonpageR{\the\numpagelinesR}%
1927      \else
1928      \@writelinesonpageR{1000}%
1929      \fi
1930      \numpagelinesR=\z@

```

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

```

1930      \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.

```

1931      \checkraw@text
1932      \ifaraw@text
1933      \getlinesfrompagelistL
1934      \getlinesfrompagelistR
1935      \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
1936      {\l@dminpagelines}%
1937      \fi
1938      \repeat}

```

We have now output the text from all the chunks.

```

1939      \fi

```

Make sure that there are no inserts hanging around.

```

1940      \flush@notes

```

```

1941 \flush@notesR
1942 \endgroup

```

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

```

1943 \global\l@dpscL=\z@
1944 \global\l@dpscR=\z@
1945 \global\l@dnumstartsL=\z@
1946 \global\l@dnumstartsR=\z@
1947 \global\instanzaLfalse
1948 \global\instanzaRfalse
1949 \ignorespaces}
1950

```

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

```

\ledstrutR 1951 \newcommand*{\ledstrutL}{\strut}
1952 \newcommand*{\ledstrutR}{\strut}
1953

```

`\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.

```

1954 \providecommand{\cleartoevenpage}[1][\@empty]{%
1955 \clearpage
1956 \ifodd\c@page\hbox{#1}\clearpage\fi}
1957 \newcommand*{\cleartol@devenpage}{%
1958 \ifdim\pagetotal<\topskip% on an empty page
1959 \else
1960 \clearpage
1961 \fi
1962 \ifodd\c@page\hbox{}%\clearpage\fi}
1963 \newcommand*{\clearl@dleftpage}{%
1964 \clearpage
1965 \ifodd\c@page\else
1966 \led@err@LeftOnRightPage
1967 \hbox{}%
1968 \cleardoublepage
1969 \fi}
1970 \newcommand*{\clearl@drightpage}{%
1971 \clearpage
1972 \ifodd\c@page
1973 \led@err@RightOnLeftPage
1974 \hbox{}%
1975 \cleartoevenpage
1976 \fi}
1977

```

`\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@linesinparL 1978 \newcommand*\getlinesfromparlistL{%
1979   \ifx\linesinpar@listL\empty
1980     \gdef\@cs@linesinparL{0}%
1981   \else
1982     \gl@p\linesinpar@listL\to\@cs@linesinparL
1983   \fi}
1984 \newcommand*\getlinesfromparlistR{%
1985   \ifx\linesinpar@listR\empty
1986     \gdef\@cs@linesinparR{0}%
1987   \else
1988     \gl@p\linesinpar@listR\to\@cs@linesinparR
1989   \fi}
1990
```

`\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 1991 \newcommand*\getlinesfrompagelistL{%
1992   \ifx\linesonpage@listL\empty
1993     \gdef\@cs@linesonpageL{1000}%
1994   \else
1995     \gl@p\linesonpage@listL\to\@cs@linesonpageL
1996   \fi}
1997 \newcommand*\getlinesfrompagelistR{%
1998   \ifx\linesonpage@listR\empty
1999     \gdef\@cs@linesonpageR{1000}%
2000   \else
2001     \gl@p\linesonpage@listR\to\@cs@linesonpageR
2002   \fi}
2003
```

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

```
2004 \newcommand*\@writelinesonpageL}[1]{%
2005   \edef\next{\write\linenum@out{\string\@lopL{#1}}}%
2006   \next}
2007 \newcommand*\@writelinesonpageR}[1]{%
2008   \edef\next{\write\linenum@outR{\string\@lopR{#1}}}%
2009   \next}
2010
```

`\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>`.

```

2011 \newcommand*{\l@dcalc@maxoftwo}[3]{%
2012   \ifnum #2>#1\relax
2013     #3=#2\relax
2014   \else
2015     #3=#1\relax
2016   \fi}
2017 \newcommand*{\l@dcalc@minoftwo}[3]{%
2018   \ifnum #2<#1\relax
2019     #3=#2\relax
2020   \else
2021     #3=#1\relax
2022   \fi}
2023
\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@dpagfull is set FALSE and
\l@dsamepagefalse \ifl@dsamepage is set TRUE. If the page is spatially full then \ifl@dpagfull
\ifl@dpagfull is set TRUE and \ifl@dsamepage is set FALSE. If it is not spatially full but
\l@dpagfulltrue the maximum number of lines have been output then both \ifl@dpagfull and
\l@dpagfullfalse \ifl@dsamepage are set FALSE.
\checkpageL
\checkpageR
2024 \newif\ifl@dsamepage
2025 \l@dsamepagetrue
2026 \newif\ifl@dpagfull
2027 \newcommand*{\checkpageL}{%
2028   \l@dpagfulltrue
2029   \l@dsamepagetrue
2030   \check@goal
2031   \ifdim\pagetotal<\ledthegoal
2032     \ifnum\numpagelinesL<\l@dmnpagelines
2033       \else
2034         \l@dsamepagefalse
2035         \l@dpagfullfalse
2036       \fi
2037     \else
2038       \l@dsamepagefalse
2039       \l@dpagfulltrue
2040     \fi}
2041 \newcommand*{\checkpageR}{%
2042   \l@dpagfulltrue
2043   \l@dsamepagetrue
2044   \check@goal
2045   \ifdim\pagetotal<\ledthegoal
2046     \ifnum\numpagelinesR<\l@dmnpagelines
2047       \else
2048         \l@dsamepagefalse
2049         \l@dpagfullfalse
2050       \fi
2051     \else
2052       \l@dsamepagefalse

```

```

2053     \l@dpagefulltrue
2054 \fi}
2055

\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.

2056 \newdimen\ledthegoal
2057 \ifshiftedverses
2058     \newcommand*{\goalfraction}{0.95}
2059 \else
2060     \newcommand*{\goalfraction}{0.9}
2061 \fi
2062
2063 \newcommand*{\check@goal}{%
2064     \ledthegoal=\goalfraction\pagegoal}
2065

\ifwrittenlinesL Booleans for whether line data has been written to the section file.
\ifwrittenlinesL 2066 \newif\ifwrittenlinesL
2067 \newif\ifwrittenlinesR
2068

\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.

2069 \newcommand*{\get@nextboxL}{%
2070     \ifvbox\namebox{\l@dLcolrawbox\the\l@dpscL}% box is not empty

    The current box is not empty; do nothing.

2071 \else%
                                box is empty

    The box is empty; check if enough lines (real and blank) have been output.
2072     \ifnum\usenamecount{\l@dmaxlinesinpar\the\l@dpscL}>\@donetotallinesL
2073     \else

    Sufficient lines have been output.

2074     \ifwrittenlinesL
2075     \else

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

2076         \@writelinesinparL
2077         \writtenlinesLtrue
2078     \fi
2079     \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). If needed, restart
    the line numbering. Increment the pstartL counter.

2080         \writtenlinesLfalse
2081         \ifbypstart@

```

```

2082         \ifnum\value{pstartL}<\value{pstartLold}
2083         \else
2084             \setline{0}
2085         \fi
2086     \fi
2087     \addtocounter{pstartL}{1}
2088     \global\pstartnumtrue
2089     \l@dcalc@maxoftwo{\the\usernamecount{1@dmaxlinesinpar\the\l@dpscL}}%
2090         {\the\@donetotallinesL}%
2091         {\usernamecount{1@dmaxlinesinpar\the\l@dpscL}}%
2092     \global\@donetotallinesL \z@
2093     \global\advance\l@dpscL \@ne
2094 \fi
2095 \fi
2096 \fi}
2097 \newcommand*{\get@nextboxR}{%
2098     \ifvbox\namebox{1@dRcolrawbox\the\l@dpscR}% box is not empty
2099     \else% box is empty
2100         \ifnum\usernamecount{1@dmaxlinesinpar\the\l@dpscR}>\@donetotallinesR
2101         \else
2102             \ifwritelinesR
2103             \else
2104                 \@writelinesinparR
2105                 \writelinesRtrue
2106             \fi
2107             \ifnum\l@dnumpstartsR>\l@dpscR
2108                 \writelinesRfalse
2109                 \ifbypstart@R
2110                     \ifnum\value{pstartR}<\value{pstartRold}
2111                     \else
2112                         \write\linenum@outR{\string\@set[0]}
2113                     \fi
2114                 \fi
2115                 \addtocounter{pstartR}{1}
2116                 \global\pstartnumRtrue
2117                 \l@dcalc@maxoftwo{\the\usernamecount{1@dmaxlinesinpar\the\l@dpscR}}%
2118                     {\the\@donetotallinesR}%
2119                     {\usernamecount{1@dmaxlinesinpar\the\l@dpscR}}%
2120                 \global\@donetotallinesR \z@
2121                 \global\advance\l@dpscR \@ne
2122             \fi
2123         \fi
2124     \fi}
2125

```

## 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—risisse Cupido  
 4 dicitur atque unum surripuisse pedem.  
 -----  
 5 “Quis tibi, saeve puer, dedit hoc in carmina iuris?  
 6 Pieridum vates, non tua turba sumus.  
 7 Quid si praecripiat flavae Vēnus arma Minervae,  
 8 ventilet accensas flava Minerva faces?  
 -----  
 9 Quis probet in silvis Cererem regnare iugosis,  
 10 lege pharetratae Virginis arva coli?  
 11 Crinibus insignem quis acuta cuspide Phoebum  
 -----  
 12 instruat, Aoniam Marte movente lyram?  
 -----

---

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

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

<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—risisse Cupido  
 4 dicitur atque unum surripuisse pedem.  
 —————  
 5 “Quis tibi, saeve puer, dedit hoc in carmina iuris?  
 6 Pieridum vates, non tua turba sumus.  
 7 Quid si praecripiat flavae Vēnus arma Minervae,  
 8 ventilet accensas flava Minerva faces?  
 —————  
 9 Quis probet in silvis Cererem regnare iugosis,  
 10 lege pharetratae Virginis arva coli?  
 11 Crinibus insignem quis acuta cuspide Phoebum  
 12 instruat, Aoniam Marte movente lyram?  
 —————

---

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

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



<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.

---

```

2126 (*villon)
2127 %% villon.tex Example parallel columns
2128 \documentclass{article}
2129 \addtolength{\textheight}{-10\baselineskip}
2130 \usepackage{ledmac,ledpar}
2131 %% Use r instead of R to flag right text line numbers
2132 \renewcommand{\Rlineflag}{r}
2133 %% Use the flag in the notes
2134 \let\oldBfootfmt\Bfootfmt
2135 \renewcommand{\Bfootfmt}[3]{%
2136   \let\printlines\printlinesR
2137   \oldBfootfmt{#1}{#2}{#3}}
2138 \begin{document}
2139
2140 I thought that limericks were peculiarly English, but this appears not
2141 to be the case. As with most limericks this one is by Anonymous.
2142
2143 \vspace*{\baselineskip}
2144
2145 \begin{pairs}
2146 %% no indentation
2147 \setstanzaindents{0,0,0,0,0,0,0,0,0}
2148 %% no number flag
2149 \renewcommand{\Rlineflag}{}
2150 %% draw a rule and widen the columns
2151 \setlength{\columnrulewidth}{0.4pt}
2152 \setlength{\Lcolwidth}{0.46\textwidth}
2153 \setlength{\Rcolwidth}{\Lcolwidth}
2154
2155 \begin{Leftside}
2156 %% set left text line numbering sequence
2157 \firstlinenum{2}
2158 \linenumincrement{2}
2159 \linenummargin{left}
2160 \beginnumbering
2161 \stanza
2162 Il y avait un jeune homme de Dijon, &
2163 Qui n'avait que peu de religion. &
2164 Il dit: 'Quant \'{a} moi, &
2165 Je d'\{e\}teste tous les trois, &
2166 Le P\{e\}re, et le Fils, et le Pigeon.' \&
2167 \endnumbering
2168 \end{Leftside}

```

```

2169
2170 \begin{Rightside}
2171 %% different right text line numbering sequence
2172 \firstlinenum{1}
2173 \linenumincrement{2}
2174 \linenummargin{right}
2175 \beginnumbering
2176 \stanza
2177 There was a young man of Dijon, &
2178 Who had only a little religion, &
2179 He said: 'As for me, &
2180 I detest all the three, &
2181 The Father, the Son, and the Pigeon.' \&
2182 \endnumbering
2183 \end{Rightside}
2184
2185 \Columns
2186 \end{pairs}
2187
2188 \vspace*{\baselineskip}
2189
2190 The following is verse \textsc{lxixiii} of Fran\c{c}ois Villon's
2191 \textit{Le Testament} (The Testament), composed in 1461.
2192
2193 %% Allow for hanging indentation for long lines
2194 \setstanzaindents{1,0,0,0,0,0,0,0}
2195 %% Columns wider than the default
2196 \setlength{\Lcolwidth}{0.46\textwidth}
2197 \setlength{\Rcolwidth}{\Lcolwidth}
2198 \vspace*{\baselineskip}
2199
2200 \begin{pairs}
2201 \begin{Leftside}
2202 \firstlinenum{2}
2203 \linenumincrement{2}
2204 \linenummargin{left}
2205 \beginnumbering
2206 \stanza
2207 Dieu mercy et Tacque Thibault, &
2208 Qui tant d'eaue froid m'a fait boire, &
2209 Mis en bas lieu, non pas en hault, &
2210 Mengier d'angoisse maints \edtext{poire}{\lemma{poire d'angoisse}}%
2211 \Afootnote{This has a triple meaning: literally it is the fruit of the
2212 choke pear,
2213 figuratively it means 'bitter fruit', and it also refers to a torture
2214 instrument.}}, &
2215 Enferr'\{e} \ldots Quant j'en ay memoire, &
2216 Je Prie pour luy \edtext{\textit{et reliqua}}{\Afootnote{and so on}}, &
2217 Que Dieu luy doint, et voire, voire! &
2218 Ce que je pense \ldots \textit{et cetera}. \&

```

```

2219 \endnumbering
2220 \end{Leftside}
2221
2222 \begin{Rightside}
2223 \firstlinenum{2}
2224 \linenumincrement{2}
2225 \linenummargin{right}
2226 \beginnumbering
2227 \stanza
2228 Thanks to God --- and to \edtext{Tacque Thibaud}{%
2229   \Bfootnote{A favourite of Jean, Duc de Berry and loathed for his exactions
2230     and debauchery. Villon uses his name as an insulting nickname for
2231     Thibaud d'Auxigny, the Bishop of Orl\{'e}ans.}} &
2232 Who made me drink so much \edtext{cold water}{%
2233   \Bfootnote{Can either refer to the normal prison diet of bread and
2234     water or to a common medieval torture which involved forced drinking
2235     of cold water.}}, &
2236 Put me underground instead of higher up &
2237 And made me eat such bitter fruit, &
2238 In chains \ldots When I think of this, &
2239 I pray for him---\textit{et reliqua;} &
2240 May God grant him (yes, by God) &
2241 What I think \ldots \textit{et cetera}. \&
2242 \endnumbering
2243 \end{Rightside}
2244
2245 \Columns
2246 \end{pairs}
2247
2248 \vspace*{\baselineskip}
2249
2250   The translation and notes are by Anthony Bonner,
2251 \textit{The Complete Works of Fran\c{c}ois Villon}, published by
2252 Bantam Books in 1960.
2253
2254 \end{document}
2255
2256 </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.

---

```

2257 <*djd17nov>
2258 %%% This is djdl7nov.tex, a sample critical text edition
2259 %%% written in LaTeX2e with the ledmac and ledpar packages.
2260 %%% (c) 2003--2004 by Dr. Dirk-Jan Dekker,

```

```

2261 %% Radboud University, Nijmegen (The Netherlands)
2262 %% (PRW) Modified slightly by PRW to fit the ledpar manual
2263
2264 \documentclass[10pt, letterpaper, twoside]{article}
2265 \usepackage[latin,english]{babel}
2266 \usepackage{makeidx}
2267 \usepackage{ledmac,ledpar}
2268 \lineation{section}
2269 \linenummargin{inner}
2270 \sidenotemargin{outer}
2271
2272 \makeindex
2273
2274 \renewcommand{\notenumfont}{\footnotesize}
2275 \newcommand{\notetextfont}{\footnotesize}
2276
2277 %\let\Afootnoterule=\relax
2278 \let\Bfootnoterule=\relax
2279 \let\Cfootnoterule=\relax
2280
2281 \addtolength{\skip\Afootins}{1.5mm}
2282 %\addtolength{\skip\Bfootins}{1.5mm}
2283 %\addtolength{\skip\Cfootins}{1.5mm}
2284
2285 \makeatletter
2286
2287 \renewcommand*{\para@vfootnote}[2]{%
2288   \insert\csname #1footins\endcsname
2289   \bgroup
2290     \notefontsetup
2291     \interlinepenalty=\interfootnotelinepenalty
2292     \floatingpenalty=\@MM
2293     \splittopskip=\ht\strutbox \splitmaxdepth=\dp\strutbox
2294     \leftskip=\z@skip \rightskip=\z@skip
2295     \l@dparsedfootspec #2\ledplinenumtrue% new from here
2296     \ifnum\@nameuse{previous@#1@number}=\l@dparsedstartline\relax
2297       \ledplinenumfalse
2298     \fi
2299     \ifnum\previous@page=\l@dparsedstartpage\relax
2300     \else \ledplinenumtrue \fi
2301     \ifnum\l@dparsedstartline=\l@dparsedendline\relax
2302     \else \ledplinenumtrue \fi
2303     \expandafter\xdef\csname previous@#1@number\endcsname{\l@dparsedstartline}%
2304     \xdef\previous@page{\l@dparsedstartpage}% to here
2305     \setbox0=\vbox{\hsize=\maxdimen
2306       \noindent\csname #1footfmt\endcsname#2}%
2307     \setbox0=\hbox{\unvvh0}%
2308     \dp0=0pt
2309     \ht0=\csname #1footfudgefactor\endcsname\wd0
2310     \box0

```

```

2311     \penalty0
2312     \egroup
2313 }
2314
2315 \newcommand*{\previous@A@number}{-1}
2316 \newcommand*{\previous@B@number}{-1}
2317 \newcommand*{\previous@C@number}{-1}
2318 \newcommand*{\previous@page}{-1}
2319
2320 \newcommand{\abb}[1]{#1%
2321     \let\rbracket\nobrak\relax}
2322 \newcommand{\nobrak}{\textnormal{}}
2323 \newcommand{\morenoexpands}{%
2324     \let\abb=0%
2325 }
2326
2327 \newcommand{\Aparafootfmt}[3]{%
2328     \ledsetnormalparstuff
2329     \scriptsize
2330     \notenumfont\printlines#1\enspace
2331 %   \lemmafont#1/#2\enskip
2332     \notetextfont
2333     #3\penalty-10\hskip 1em plus 4em minus.4em\relax}
2334
2335 \newcommand{\Bparafootfmt}[3]{%
2336     \ledsetnormalparstuff
2337     \scriptsize
2338     \notenumfont\printlines#1/%
2339     \ifledplinenum
2340     \enspace
2341     \else
2342     {\hskip 0em plus 0em minus .3em}%
2343     \fi
2344     \select@lemmafont#1/#2\rbracket\enskip
2345     \notetextfont
2346     #3\penalty-10\hskip 1em plus 4em minus.4em\relax }
2347
2348 \newcommand{\Cparafootfmt}[3]{%
2349     \ledsetnormalparstuff
2350     \scriptsize
2351     \notenumfont\printlines#1\enspace
2352 %   \lemmafont#1/#2\enskip
2353     \notetextfont
2354     #3\penalty-10\hskip 1em plus 4em minus.4em\relax}
2355
2356 \makeatother
2357
2358 \footparagraph{A}
2359 \footparagraph{B}
2360 \footparagraph{C}

```

```

2361
2362 \let\Afootfmt=\Aparafootfmt
2363 \let\Bfootfmt=\Bparafootfmt
2364 \let\Cfootfmt=\Cparafootfmt
2365
2366 \renewcommand*{\Rlineflag}{}
2367
2368 \emergencystretch40pt
2369
2370 \author{Guillelmus de Berchen}
2371 \title{Chronicon Geldriae}
2372 \date{}
2373 \hyphenation{archi-epi-sco-po Huns-dis-brug li-be-ra No-vi-ma-gen-si}
2374 \begin{document}
2375 \begin{pages}
2376 \begin{Leftside}
2377 \beginnumbering\pstart
2378 \selectlanguage{latin}
2379 \section{De ecclesia S. Stephani Novimagensi}
2380
2381 \noindent\setline{1}
2382 Nobilis itaque comes Otto\protect\edindex{Otto II of Guelders}
2383 imperio et dominio Novimagensi sibi, ut praeferetur, impignoratis
2384 et commissis
2385 \edtext{proinde}{\Bfootnote{primum D}} praeesse cupiens, anno
2386 \textsc{liiii} superius descripto, mense
2387 \u\edtext{}{\Afootnote{p.\ 227~R}}nio, una cum iudice, scabinis ceterisque
2388 civibus civitatis Novimagensis, pro ipsius et inhabitantium in ea
2389 necessitate,\edtext{}{\Afootnote{p.\ 97~N}} commodo et utilitate,
2390 ut \edtext{ecclesia eius}{\Bfootnote{ecclesia D: eius eius H}} parochialis
2391 \edtext{\abb{extra civitatem}}{\Bfootnote{\textit{om.}~H}} sita
2392 destrueretur et \edtext{infra}{\Bfootnote{intra D}} muros
2393 \edtext{transfer}\edtext{}{\Afootnote{p.\ 129~D}}retur}%
2394 {\Bfootnote{transferreretur NH}}
2395 ac de novo construeretur,
2396 \edtext{a reverendo patre domino
2397 Conrado\protect\edindex{Conrad of Hochstaden} de
2398 \edtext{Hofsteden}{\Bfootnote{Hoffstede D: Hoffsteden H}}, archiepiscopo
2399 \edtext{Coloniensi}{\Bfootnote{Colononiensi H}}, licentiam}%
2400 {\Cfootnote{William is confusing two charters that are five years
2401 apart. Permission from St.\ Apostles' Church in Cologne had been
2402 obtained as early as 1249. Cf.\
2403 Sloet\protect\index{Sloet van de Beele, L.A.J.W.},
2404 \textit{Oorkondenboek} nr.\ 707 (14 November 1249):
2405 ‘‘\ldots{nos devotionis tue precibus annuentes, ut ipsam ecclesiam
2406 faciens demoliri transferas in locum alium competentem, tibi
2407 auctoritate presentium indulgemus\ldots’’}}, et a venerabilibus
2408 \edtext{dominis}{\Bfootnote{viris H}} decano et capitulo sanctorum
2409 Apostolorum\protect\edindex{St. Apostles' (Cologne)}
2410 \edtext{Coloniensi}{\Bfootnote{Coloniae H}}, ipsius ecclesiae ab

```

```

2411 antiquo veris et pacificis patronis, consensum, citra tamen
2412 praeiudicium, damnum aut gravamen \edtext{iurium}{\Bfootnote{virium D}}
2413 et bonorum eorundem, impetravit.
2414 \pend
2415
2416 \pstart
2417 \edtext{Et exinde \edtext{liberum}{\Bfootnote{librum H}}}
2418 locum eiusdem civitatis
2419 \edtext{qui}{\Bfootnote{quae D}} dicitur
2420 \edtext{Hundisburg}{\Bfootnote{Hundisburch D: Hundisbrug HMN:
2421 Hunsdisbrug R}}\protect\edindex{Hundisburg},
2422 de praelibati Wilhelmi\protect\edindex{William II of Holland} Romanorum
2423 \edtext{regis}{\Bfootnote{imperatoris D}}, ipsius fundi
2424 do\edtext{}{\Afootnote{f.\ 72v~M}}mini, consensu, ad aedificandum
2425 \edtext{\abb{et consecrandum}}{\Bfootnote{\textit{om.}\ H}}
2426 ecclesi\edtext{}{\Afootnote{p.\ 228~R}}am et coemeterium,
2427 \edtext{eisdem}{\Bfootnote{eiusdem D}} decano et capitulo de expresso
2428 eiusdem civitatis assensu libera contradiderunt voluntate, obligantes
2429 se ipsi \edtext{comes}{\Bfootnote{comites D}} et civitas
2430 \edtext{\abb{dictis}}{\Bfootnote{\textit{om.}\ H}} decano et capitulo,
2431 quod in recompensationem illius areae infra castrum et portam, quae
2432 fuit dos ecclesiae, in qua plebanus habitare solebat---quae
2433 \edtext{tunc}{\Bfootnote{nunc H}} per novum fossatum civitatis est
2434 destructa---aliam aream competentem et ecclesiae novae,
2435 \edtext{ut praeferitur, aedificandae}{%
2436 \lemma{\abb{ut\ldots aedificandae}}{\Bfootnote{\textit{om.}\ H}} satis
2437 \edtext{contiguam}{\Bfootnote{contiguum M}}, ipsi plebano darent et
2438 assignarent.}{\Cfootnote{Cf.\ Sloet, \textit{Oorkondenboek} nr.\ 762
2439 (June 1254)}} Et desuper
2440 \edtext{\abb{apud}}{\Bfootnote{\textit{om.}\ H}} dictam ecclesiam
2441 sanctorum Apostolorum \edtext{est}{\Bfootnote{et H}}
2442 \edtext{littera}{\Bfootnote{litteram H}} sigillis ipsorum
2443 Ottonis\edtext{}{\Afootnote{p.\ 130~D}} comitis et civitatis
2444 \edtext{Novimagensis}{\Bfootnote{Novimagii D}}
2445 \edtext{sigillata}{\Bfootnote{sigillis communita H}}.
2446 \pend
2447
2448 \pstart
2449 // One additional line to show synchronization. //
2450 \pend
2451 \endnumbering
2452 \end{Leftside}
2453
2454 \begin{Rightside}
2455 \sidenotemargin{right}\selectlanguage{english}
2456 \beginnumbering
2457 \pstart
2458 \addtocounter{section}{-1}%
2459 \leavevmode\section{St.\ Stephen's Church in Nijmegen}
2460

```



2461 \noindent\setline{1}%  
 2462 After the noble count Otto had taken in pledge the power over  
 2463 Nijmegen,\footnote{In 1247 William II\protect\index{William II of Holland}  
 2464 (1227--1256) count of Holland needed money to fight his way to  
 2465 Aachen\protect\index{Aachen} to be crowned King of the Holy Roman  
 2466 Empire. He gave the town of Nijmegen in pledge to Otto  
 2467 II\protect\index{Otto II of Guelders} (1229--1271) count of Guelders.}  
 2468 like I have written above, he wanted to protect the town. So in June  
 2469 1254\ledsidenote{1254} he and the judge, the sheriffs and other  
 2470 citizens of Nijmegen obtained permission to demolish the parish  
 2471 church that lay outside the town walls,\footnote{Since the early  
 2472 seventh century old St.\ Stephen's church had been located close  
 2473 to the castle, at today's  
 2474 Kelfkensbos\protect\index{Kelfkensbos (Nijmegen)}} square.  
 2475 Traces of the church and the presbytery were found during excavations  
 2476 in 1998--1999.} to move it inside the walls and to rebuild it new.  
 2477 This operation was necessary and useful both for Otto himself and  
 2478 for the inhabitants of the town. The reverend father Conrad of  
 2479 Hochstaden, archbishop of  
 2480 Cologne,\footnote{Conrad of Hochstaden ({\textdag} 1261) was  
 2481 archbishop of Cologne in 1238--1261. Nijmegen belonged to the  
 2482 archdiocese of Cologne until 1559.} gave his permission. So did the  
 2483 reverend dean and canons of the chapter of St.\  
 2484 Apostles'\protect\index{St. Apostles' (Cologne)} in Cologne, who had  
 2485 long\footnote{They probably became the patrons when the chapter was  
 2486 established in the early eleventh century. About the church and the  
 2487 chapter, see Gottfried Stracke\protect\index{Stracke, G.},  
 2488 \textit{K}\{"o}ln: St.\ Aposteln}, Stadspuren -- Denkm\{"a}ler in  
 2489 K\{"o}ln, vol.\ 19, K\{"o}ln: J.\,P.\ Bachem, 1992.} been the true  
 2490 and benevolent patrons of the church---but they did not allow Otto  
 2491 to do anything without their knowledge, nor to infringe their rights,  
 2492 nor to damage their property.  
 2493 \pend  
 2494  
 2495 \pstart  
 2496 And so the count and the town voluntarily gave an open space in town  
 2497 called Hundisburg, which was owned by the aforementioned king William,  
 2498 to the dean and chapter of St.\ Apostles' in order to build and  
 2499 consecrate a church and graveyard. King William approved and the  
 2500 town of Nijmegen explicitly expressed its assent. A new ditch was dug  
 2501 on property of the church near the castle and the  
 2502 harbour,\footnote{Nowadays, the exact location of the medieval  
 2503 ditch---and of two Roman ones---can be seen in the pavement of  
 2504 Kelfkensbos\protect\index{Kelfkensbos (Nijmegen)}} square.} causing  
 2505 the demolition of the presbytery. In compensation, the count and  
 2506 citizens committed themselves to giving the parish priest another  
 2507 suitable space close enough to the new church that was about to be  
 2508 built. A letter about these transactions, with the seals of count  
 2509 Otto and the town of Nijmegen, is kept at St.\ Apostles'  
 2510 church.\footnote{The original letter is lost. A 15th century

```

2511 transcription of it is kept at the Historisches Archiv der
2512 Stadt K\{"o}ln (HASTK).}
2513 \pend
2514
2515 \pstart
2516 // One additional line to show synchronization. //
2517 \pend
2518 \endnumbering
2519 \end{Rightside}
2520 \Pages
2521 \end{pages}
2522
2523 %%%%%%%%%%%%%%
2524 \printindex
2525 \end{document}
2526 %%%%%%%%%%%%%%
2527
2528 \</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.

---

```

2529 (*djdpoems)
2530 %% djdpoems.tex  example parallel verses on facing pages
2531 \documentclass{article}
2532 \usepackage{ledmac,ledpar}
2533 \addtolength{\textheight}{-15\baselineskip}
2534
2535 \maxchunks{24} % default value = 10
2536 \setstanzaindents{6,0,1,0,1}
2537
2538 \newcommand{\longdash}{-----}
2539
2540 \footparagraph{A} % for left pages
2541 \footparagraph{B} % for right pages
2542 \firstlinenum{1}
2543 \linenumincrement{1}
2544
2545 \let\oldBfootfmt\Bfootfmt
2546 \renewcommand{\Bfootfmt}[3]{%
2547   \let\printlines\printlinesR
2548   \oldBfootfmt{#1}{#2}{#3}}

```

```

2549
2550 \begin{document}
2551
2552 \newcommand{\interstanza}{\pstart\centering\longdash\skipnumbering\pend}
2553
2554 \begin{pages}
2555 \begin{Leftside}
2556 \def\endstanzaextra{\interstanza}
2557 \beginnumbering
2558
2559 \stanza
2560 Arma gravi numero violentaque bella parabam &
2561 edere, materi\={a} conveniente modis. &
2562 Par erat inferior versus---risisse Cupido &
2563 dicitur atque unum surripuisse pedem. \&
2564
2565 \stanza
2566 ‘Quis tibi, saeve puer, dedit hoc in carmina iuris? &
2567 Pieridum vates, non tua turba \edtext{sumus}{\Afootnote{note lost}}. &
2568 Quid si praeripiat flavae V\ue{nus arma Minervae, &
2569 ventilet accensas flava Minerva faces? \&
2570
2571 \stanza
2572 Quis probet in silvis Cererem regnare iugosis, &
2573 lege pharetratae Virginis arva coli? &
2574 Crinibus insignem quis \edtext{acuta}{\Afootnote{acut\={a} (abl.\ abs.)}}
2575 cuspidе Phoebum &
2576 instruat, Aoniam Marte movente lyram? \&
2577 \endnumbering
2578 \end{Leftside}
2579
2580 \begin{Rightside}
2581 \def\endstanzaextra{\interstanza}
2582 \beginnumbering
2583 \firstlinenum{1}
2584 \linenumincrement{1}
2585 \setstanzaindents{6,0,1,0,1,0}
2586
2587 \stanza
2588 I was preparing to sing of weapons and violent wars, &
2589 in heavy numbers, with the subject matter suited to the verse measure. &
2590 The even lines were as long as the odd ones, but Cupid laughed, &
2591 they said, and he stole away one foot.\footnote{I.e., the even lines,
2592 which were hexameters (with six feet) became pentameters
2593 (with five feet).} \&
2594
2595 \stanza
2596 ‘O cruel boy, who gave you the right over poetry? &
2597 We poets belong to the Pierides,\footnote{Muses} we are not your folk. &
2598 \edlabel{beginparadox}What if Venus should seize away the arms of

```

```

2599 Minerva with the golden hair, &
2600 if Minerva with the golden hair should fan alight the kindled torch
2601 of love? \&
2602
2603 \stanza
2604 Who would approve of Ceres\footnote{Ceres was the Roman goddess of
2605 the harvest.} reigning on the woodland ridges, &
2606 and of land tilled under the law of the Maid with the
2607 quiver\footnote{By '\textit{Virgo}' ('Virgin') Ovid means Diana, the
2608 Roman goddess of the hunt.}? &
2609 Who would provide Phoebus with his beautiful hair with a sharp-pointed
2610 spear, &
2611 while Mars stirs the \edtext{Aonian}{\Bfootnote{Mount Parnassus,
2612 where the Muses live, is located in Aonia.}}
2613 lyre?\edlabel{endparadox}\footnote{Lines
2614 \xlineref{beginparadox}--\xlineref{endparadox} show some paradoxical
2615 situations that would occur if the gods didn't stay with their own
2616 business.} \&
2617 \endnumbering
2618 \end{Rightside}
2619
2620 \Pages
2621 \end{pages}
2622
2623 \begin{pages}
2624 \begin{Leftside}
2625 \def\endstanzaextra{\interstanza}
2626 \beginnumbering
2627
2628 \begin{astanza}
2629 Arma gravi numero violentaque bella parabam &
2630 edere, materi\={a} conveniente modis. &
2631 Par erat inferior versus---risisse Cupido &
2632 dicitur atque unum surripuisse pedem. \&
2633 \end{astanza}
2634
2635 \begin{astanza}
2636 ‘Quis tibi, saeve puer, dedit hoc in carmina iuris? &
2637 Pieridum vates, non tua turba \edtext{sumus}{\Afootnote{note lost}}. &
2638 Quid si praecripiat flavae V\ufe}nus arma Minervae, &
2639 ventilet accensas flava Minerva faces? \&
2640 \end{astanza}
2641
2642 \begin{astanza}
2643 Quis probet in silvis Cererem regnare iugosis, &
2644 lege pharetratae Virginis arva coli? &
2645 Crinibus insignem quis \edtext{acuta}{\Afootnote{acut\={a} (abl.\ abs.)}}
2646 cuspide Phoebum &
2647 instruat, Aoniam Marte movente lyram? \&
2648 \end{astanza}

```

```

2649
2650 \endnumbering
2651 \end{Leftside}
2652
2653 \begin{Rightside}
2654 \def\endstanzaextra{\interstanza}
2655 \beginnumbering
2656 \firstlinenum{1}
2657 \linenumincrement{1}
2658 \setstanzaindents{6,0,1,0,1,0}
2659
2660 \begin{astanza}
2661 I was preparing to sing of weapons and violent wars, &
2662 in heavy numbers, with the subject matter suited to the verse measure. &
2663 The even lines were as long as the odd ones, but Cupid laughed, &
2664 they said, and he stole away one foot.\footnote{I.e., the even lines,
2665 which were hexameters (with six feet) became pentameters
2666 (with five feet).} \&
2667 \end{astanza}
2668
2669 \begin{astanza}
2670 ‘‘O cruel boy, who gave you the right over poetry? &
2671 We poets belong to the Pierides,\footnote{Muses} we are not your folk. &
2672 \edlabel{beginparadox}What if Venus should seize away the arms of
2673 Minerva with the golden hair, &
2674 if Minerva with the golden hair should fan alight the kindled torch
2675 of love? \&
2676 \end{astanza}
2677
2678 \begin{astanza}
2679 Who would approve of Ceres\footnote{Ceres was the Roman goddess of the
2680 harvest.} reigning on the woodland ridges, &
2681 and of land tilled under the law of the Maid with the
2682 quiver\footnote{By ‘\textit{Virgo}’ (‘Virgin’) Ovid means Diana,
2683 the Roman goddess of the hunt.}? &
2684 Who would provide Phoebus with his beautiful hair with a sharp-pointed
2685 spear, &
2686 while Mars stirs the \edtext{Aonian}{\Bfootnote{Mount Parnassus, where
2687 the Muses live, is located in Aonia.}}
2688 lyre?\edlabel{endparadox}\footnote{Lines
2689 \xlineref{beginparadox}--\xlineref{endparadox} show some paradoxical
2690 situations that would occur if the gods didn’t stay with their
2691 own business.} \&
2692 \end{astanza}
2693
2694 \endnumbering
2695 \end{Rightside}
2696
2697 \Pages
2698 \end{pages}

```

```
2699
2700 \end{document}
2701
2702 </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> . . . 1583, 1584, 1588, 1605, 1619, 2166, 2181, 2218, 2241, 2563, 2569, 2576, 2593, 2601, 2616, 2632, 2639, 2647, 2666, 2675, 2691	<code>\@donetotallinesR</code> . . . . . 870, 935, 938, 1884, 2100, 2118, 2120
<code>\@M</code> . . . . . 1594	<code>\@insertR</code> . . . . . 1239–1241, 1254–1256
<code>\@MM</code> . . . . . 2292	<code>\@l</code> . . . . . 260, 573
<code>\@adv</code> . . . . . 337, 604, 605	<code>\@l@dttempcnta</code> . . . . . . 410, 412, 414, 415, 419, 421, 423, 424, 992, 1031, 1032, 1034, 1036, 1039, 1040, 1057–1061, 1063, 1070, 1075, 1079, 1087, 1092, 1096, 1129, 1132, 1134, 1138
<code>\@afterindentfalse</code> . . . . . 715	<code>\@l@dttempcntb</code> . 153, 155, 157, 1022, 1023, 1070, 1075, 1079, 1087, 1092, 1096, 1121, 1125, 1138, 1146–1148, 1150, 1170–1172, 1174, 1191–1193, 1195, 1480, 1482, 1484, 1537–1539, 1541, 1647–1651, 1655–1658, 1662–1665
<code>\@arabic</code> . . . . . 194, 195, 764, 767	<code>\@l@reg</code> . . . . . 309
<code>\@astanza@line</code> . . . . . 1604, 1610, <u>1613</u>	<code>\@l@regR</code> . . . . . 260
<code>\@auxout</code> . . . . . 1439, 1451, 1693	<code>\@lab</code> . . . . . 529, 1430, 1442, <u>1466</u>
<code>\@chapter</code> . . . . . 716	<code>\@lock</code> . . . . . 973
<code>\@cs@linesinparL</code> . . . . . 1867, <u>1978</u>	<code>\@lockR</code> . . . . . 58, 282, 284, 286, 299, 444, 460, 461, 463, 464, 492, 493, 495, 954, 998, 1000,
<code>\@cs@linesinparR</code> . . . . . 1867, <u>1978</u>	
<code>\@cs@linesonpageL</code> . . 1875, 1935, <u>1991</u>	
<code>\@cs@linesonpageR</code> . . 1875, 1935, <u>1991</u>	
<code>\@donereallinesL</code> . . . . . . . . . . 870, 899, 1829, 1831, 1881	
<code>\@donereallinesR</code> . . . . . . . . . . 870, 934, 1834, 1836, 1883	
<code>\@donetotallinesL</code> . . . . . 870, 900, 903, 1882, 2072, 2090, 2092	

- 1001, 1003, 1084, 1101, 1103, 1105  
 \@lopL ..... 551, 2005  
 \@lopR ..... 551, 2008  
 \@nameuse ..... 1565, 1569, 2296  
 \@nobreakfalse ..... 773, 803  
 \@nobreaktrue ..... 771, 775, 801, 805  
 \@oldnobreak 771, 773, 801, 803, 840, 856  
 \@pend ..... 544, 1829  
 \@pendR ..... 544, 1834  
 \@pstartfalse ..... 1802  
 \@pstarttrue ..... 1802  
 \@ref ..... 516, 577, 581  
 \@ref@reg ..... 542  
 \@schapter ..... 716  
 \@set .. 369, 611, 612, 1778, 1781, 2112  
 \@tag 642, 659, 1263, 1267, 1277, 1281,  
     1291, 1295, 1305, 1309, 1319,  
     1323, 1334, 1338, 1348, 1352,  
     1362, 1366, 1376, 1380, 1390, 1394  
 \@temp ..... 1684  
 \@templ@d ..... 1528, 1529  
 \@writelinesinparL .. 1774, 1827, 2076  
 \@writelinesinparR .. 1775, 1827, 2104  
 \@writelinesonpageL . 1905, 1907, 2004  
 \@writelinesonpageR . 1925, 1927, 2004  
 \@xloop ..... 1252
- \\_ 2387, 2389, 2393, 2401, 2402, 2404,  
     2424–2426, 2430, 2436, 2438,  
     2440, 2443, 2459, 2472, 2483,  
     2488, 2489, 2498, 2509, 2574, 2645
- ### A
- \abb ..... 2320,  
     2324, 2391, 2425, 2430, 2436, 2440  
 \absline@num ..... 403, 417, 436, 963  
 \absline@numR ... 56, 211, 262, 265,  
     268, 400, 408, 429, 448, 482,  
     510, 521, 944, 984, 985, 1022, 1238  
 \actionlines@list .....  
     ..... 252, 255, 403, 417, 436  
 \actionlines@listR .....  
     ..... 215, 230, 244, 247, 400,  
     408, 429, 448, 482, 510, 1044, 1047  
 \actions@list . 256, 404, 424, 438, 440  
 \actions@listR .....  
     . 215, 231, 248, 401, 415, 431,  
     433, 450, 459, 484, 491, 511, 1048  
 \add@inserts ..... 891  
 \add@inserts@nextR ..... 1227  
 \add@insertsR ..... 926, 1227  
 \add@penaltiesL ..... 898, 1248  
 \add@penaltiesR ..... 933, 1248  
 \addtocontents ..... 1694–1696  
 \addtocounter ..... 842,  
     858, 1783, 1784, 2087, 2115, 2458  
 \addtolength ... 2129, 2281–2283, 2533  
 \advancelabel@refs ..... 1437, 1449  
 \advanceline ..... 603, 634  
 \affixline@num ..... 889  
 \affixline@numR ..... 924, 1054  
 \affixpstart@numL ..... 888, 1160  
 \affixpstart@numR ..... 923, 1160  
 \affixside@note ..... 892  
 \affixside@noteR ..... 927, 1527  
 \Afootfmt ..... 2362  
 \Afootins ..... 2281  
 \Afootnote ..... 1259, 2211,  
     2216, 2387, 2389, 2393, 2424,  
     2426, 2443, 2567, 2574, 2637, 2645  
 \Afootnoterule ..... 2277  
 \Aparafootfmt ..... 2327, 2362  
 \apptocmd ..... 1724  
 \araw@textfalse ..... 1814  
 \araw@texttrue ..... 1814  
 astanza (environment) ..... 9, 1586  
 \AtBeginDocument ... 1462, 1669, 1704  
 \author ..... 2370
- ### B
- \ballast@count ..... 982, 987  
 \bbl@main@language ..... 1720, 1721  
 \bbl@set@language ..... 1711, 1712  
 \beginnumbering .. 8, 34, 722, 740,  
     778, 2160, 2175, 2205, 2226,  
     2377, 2456, 2557, 2582, 2626, 2655  
 \beginnumberingR ... 47, 102, 740, 808  
 \Bfootfmt 2134, 2135, 2363, 2545, 2546  
 \Bfootins ..... 2282  
 \Bfootnote ..... 1273, 2229, 2233,  
     2385, 2390–2392, 2394, 2398,  
     2399, 2408, 2410, 2412, 2417,  
     2419, 2420, 2423, 2425, 2427,  
     2429, 2430, 2433, 2436, 2437,  
     2440–2442, 2444, 2445, 2611, 2686  
 \Bfootnoterule ..... 2278  
 \bfseries ..... 764, 767  
 \box ..... 2310  
 \Bparafootfmt ..... 2335, 2363



\bypage@Rfalse ..... 122, 137, 142  
 \bypage@Rtrue ..... 122, 132  
 \bypstart@Rfalse ..... 122, 133, 143  
 \bypstart@Rtrue ..... 122, 138

## C

\c@ballast ..... 987  
 \c@firstlinenumR ..... 162, 1127  
 \c@firstsublinenumR ..... 166, 1122  
 \c@linenumincrementR ..... 162, 1127  
 \c@page ... 573, 1956, 1962, 1965, 1972  
 \c@pstartL ..... 764  
 \c@pstartR ..... 767  
 \c@sublinenumincrementR .. 166, 1122  
 \centering ..... 2552  
 \Cfootfmt ..... 2364  
 \Cfootins ..... 2283  
 \Cfootnote ..... 1273, 2400, 2438  
 \Cfootnoterule ..... 2279  
 \ch@ck@l@ckR ..... 1054  
 \ch@cksub@l@ckR ..... 1054  
 \ch@cksub@lockR ..... 1123  
 \chapter ..... 702, 703, 711  
 \chapterinpages ..... 695, 703, 713  
 \chardef ..... 1583  
 \check@goal ..... 2030, 2044, 2056  
 \check@pstarts .....  
     1749, 1776, 1802, 1861, 1869, 1877  
 \checkpageL ..... 1888, 1902, 2024  
 \checkpageR ..... 1911, 1922, 2024  
 \checkraw@text .....  
     .... 1755, 1772, 1814, 1885, 1931  
 \cleardoublepage ..... 1968  
 \clearl@dleftpage ..... 1910, 1954  
 \clearl@drightpage ..... 1930, 1954  
 \cleartoevenpage ..... 1954  
 \cleartol@devenpage ..... 1850, 1954  
 \closeout ..... 564, 568  
 \columnrulewidth ..... 5, 1797, 2151  
 \Columns ..... 5, 1735, 2185, 2245  
 \columnseparator ..... 5, 1769, 1797  
 \correcthangingL ..... 894, 1578  
 \correcthangingR ..... 929, 1578  
 \countLline ..... 865, 876  
 \countRline ..... 865, 911  
 \Cparafootfmt ..... 2348, 2364  
 \critext ..... 639

## D

\date ..... 2372

\DeclareOption ..... 7  
 \def@tempb ..... 140  
 Dekker, Dirk-Jan ..... 84, 90  
 \Dfootnote ..... 1273  
 \dimen ..... 590, 591, 595–597, 601  
 \divide ..... 1059  
 \do@actions ..... 965  
 \do@actions@fixedcodeR ..... 991  
 \do@actions@nextR ..... 991  
 \do@actionsR ..... 946, 991  
 \do@ballast ..... 966  
 \do@ballastR ..... 947, 982  
 \do@lineL ..... 875, 1759, 1763, 1893  
 \do@lineLhook ..... 880, 907  
 \do@lineR ..... 910, 1760, 1765, 1914  
 \do@lineRhook ..... 907, 915  
 \do@lockoff ..... 479  
 \do@lockoffL ..... 503  
 \do@lockoffR ..... 479  
 \do@lockon ..... 444  
 \do@lockonL ..... 476  
 \do@lockonR ..... 444  
 \documentclass ..... 2128, 2264, 2531  
 \dp ..... 2293, 2308  
 \dummy@ref ..... 525

## E

\edfont@info ..... 678, 681, 687, 690  
 \edindex .. 2382, 2397, 2409, 2421, 2422  
 \edlabel .. 1428, 2598, 2613, 2672, 2688  
 \edtext ..... 656, 2210,  
     2216, 2228, 2232, 2385, 2387,  
     2389–2393, 2396, 2398, 2399,  
     2408, 2410, 2412, 2417, 2419,  
     2420, 2423–2427, 2429, 2430,  
     2433, 2435, 2437, 2440–2445,  
     2567, 2574, 2611, 2637, 2645, 2686  
 \Efootnote ..... 1273  
 \emergencystretch ..... 2368  
 \empty .. 76, 79, 244, 252, 650, 667,  
     676, 685, 787, 817, 1044, 1126,  
     1134, 1229–1231, 1242, 1253,  
     1431, 1443, 1979, 1985, 1992, 1998  
 \end@lemmas ..... 650, 651, 667, 668  
 \endashchar ..... 1418  
 \endgraf ..... 836, 852, 1743, 1853  
 \endline@num ..... 532, 538  
 \endlock ..... 623, 1592, 1601, 1606

- `\endnumbering` . . . . . 8, 37, 69, 106,  
     741, 2167, 2182, 2219, 2242,  
     2451, 2518, 2577, 2617, 2650, 2694  
`\endnumberingR` 50, 69, 91, 101, 114, 741  
`\endpage@num` . . . . . 531, 538  
`\endstanzaextra` . . . . .  
     . . . . . 1608, 2556, 2581, 2625, 2654  
`\endsub` . . . . . 590  
`\endsubline@num` . . . . . 533, 539  
`\enskip` . . . . . 2331, 2344, 2352  
`\enspace` . . . . . 2330, 2340, 2351  
environments:  
    `astanza` . . . . . 9, 1586  
    `Leftside` . . . . . 6, 720  
    `pages` . . . . . 6, 695  
    `pairs` . . . . . 5, 695  
    `Rightside` . . . . . 6, 738  
`\extensionchars` . . 45, 64, 97, 111, 119
- F**
- `\f@x@l@cksR` . . . . . 1054  
`\first@linenum@out@Rfalse` . . 559, 565  
`\first@linenum@out@Rtrue` . . . . . 559  
`\firstlinenum` . . . . . 6, 171, 2157,  
     2172, 2202, 2223, 2542, 2583, 2656  
`\firstsublinenum` . . . . . 6, 171  
`\fix@page` . . . . . 305, 312  
`\flag@end` . . . . . 574, 655, 672  
`\flag@start` . . . . . 574, 647, 664  
`\floatingpenalty` . . . . . 2292  
`\flush@notes` . . . . . 1405, 1786, 1940  
`\flush@notesR` . . 1251, 1403, 1787, 1941  
`\footnote` . . . . .  
     . 2463, 2471, 2480, 2485, 2502,  
     2510, 2591, 2597, 2604, 2607,  
     2613, 2664, 2671, 2679, 2682, 2688  
`\footnotesize` . . . . . 2274, 2275  
`\footparagraph` . 2358–2360, 2540, 2541  
`\fullstop` . 207, 1415, 1417, 1419, 1421
- G**
- `\get@linelistfile` . . . . . 240  
`\get@nextboxL` . . . . . 1901, 2069  
`\get@nextboxR` . . . . . 1921, 2069  
`\getline@numL` . . . . . 885, 961  
`\getline@numR` . . . . . 920, 942  
`\getlinesfrompagelistL` . . . . .  
     . . . . . 1873, 1933, 1991  
`\getlinesfrompagelistR` . . . . .  
     . . . . . 1874, 1934, 1991  
`\getlinesfromparlistL` . . . 1865, 1978  
`\getlinesfromparlistR` . . . 1866, 1978  
`\gl@p` . . . . . 247, 248,  
     255, 256, 651, 668, 680, 689,  
     1047, 1048, 1235, 1239, 1254,  
     1434, 1446, 1982, 1988, 1995, 2001  
`\goalfraction` . . . . . 6, 2056
- H**
- `\hangingsymbol` . . . . . 11, 1576, 1577  
`\hb@xt@` . . . 887, 894, 902, 922, 929,  
     937, 1767, 1896, 1898, 1917, 1919  
`\hsize` . . . . . 797, 827,  
     1767, 1896, 1898, 1917, 1919, 2305  
`\hyphenation` . . . . . 2373
- I**
- `\if@filesw` . . . . . 1692  
`\if@firstcolumn` 1140, 1164, 1185, 1531  
`\if@nobreak` . . . . . 770, 800  
`\if@pstarts` . . . 1750, 1802, 1862, 1878  
`\ifaraw@text` . . 1757, 1814, 1887, 1932  
`\ifautopar` . . . . . 796, 826  
`\ifbypage@` . . . . . 328  
`\ifbypage@R` . . . . . 122, 318, 1026  
`\ifbypstart@` . . . . . 1777, 2081  
`\ifbypstart@R` . . . . . 122, 1780, 2109  
`\ifdim` . . . . . 591, 595, 597,  
     601, 1896, 1917, 1958, 2031, 2045  
`\iffirst@linenum@out@R` . . . . 559, 563  
`\ifinstanzaL` . . . 718, 718, 1576, 1578  
`\ifinstanzaR` . . . 718, 719, 1577, 1581  
`\ifl@d@dash` . . . . . 1418  
`\ifl@d@elin` . . . . . 1420, 1421  
`\ifl@d@esl` . . . . . 1421  
`\ifl@d@pnum` . . . . . 1415, 1419  
`\ifl@d@ssub` . . . . . 1417  
`\ifl@dpagefull` . . . . . 1904, 1924, 2024  
`\ifl@dpaging` . . . . . 9, 1578, 1581  
`\ifl@dpairing` . . . . . 9, 73, 1401  
`\ifl@dsamelang` . . . . . 1680, 1758  
`\ifl@dsamepage` . . . . . 1891, 1913, 2024  
`\ifl@dskipnumber` . . . . . 1117  
`\ifl@dusedbabel` . . . . . 1678  
`\ifledplinenum` . . . . . 1416, 2339  
`\ifledRcol` . . . . . 9, 154, 176, 180,  
     184, 188, 227, 242, 306, 315,  
     339, 353, 370, 387, 399, 407,  
     428, 473, 500, 509, 518, 575,  
     585, 592, 598, 604, 611, 619,

- 624, 628, 633, 644, 661, 675,  
1261, 1275, 1289, 1303, 1317,  
1332, 1346, 1360, 1374, 1388,  
1402, 1429, 1467, 1481, 1492,  
1504, 1516, 1551, 1564, 1715, 1725  
`\ifnoteschanged@` ..... 83  
`\ifnumberedpar@` ... 780, 810, 832,  
848, 1260, 1274, 1288, 1302,  
1316, 1331, 1345, 1359, 1373,  
1387, 1491, 1503, 1515, 1550, 1563  
`\ifnumbering` ..... 35, 127, 776, 829  
`\ifnumberingR` .... 48, 70, 93, 806, 845  
`\ifnumberline` . 943, 948, 962, 967, 1116  
`\ifnumberpstart` ... 796, 826, 841, 857  
`\ifodd` ..... 1150, 1174,  
1195, 1541, 1956, 1962, 1965, 1972  
`\ifpst@rtedL` ..... 30, 784  
`\ifpst@rtedR` ..... 30, 814  
`\ifpstartnum` ..... 1205, 1210  
`\ifpstartnumR` ..... 1160  
`\ifshiftedverses` . 5, 1895, 1916, 2057  
`\ifsidepstartnum` 796, 826, 1162, 1183  
`\ifsublines@` ..... 205,  
294, 338, 371, 378, 409, 418,  
430, 437, 449, 483, 537, 539,  
949, 968, 1033, 1120, 1469, 1473  
`\ifvbox` 877, 912, 1818, 1821, 2070, 2098  
`\ifvmode` ..... 1436, 1448  
`\ifvoid` ..... 1408–1412  
`\ifwrittenlinesL` ..... 2066, 2074  
`\ifwrittenlinesR` ..... 2067, 2102  
`\initnumbering@reg` ..... 43  
`\insert` ..... 2288  
`\insert@count` ..... 515, 581, 645,  
662, 1268, 1282, 1296, 1310,  
1324, 1339, 1353, 1367, 1381,  
1395, 1499, 1511, 1523, 1558, 1571  
`\insert@countR` .... 516, 577, 644,  
661, 1264, 1278, 1292, 1306,  
1320, 1335, 1349, 1363, 1377,  
1391, 1495, 1507, 1519, 1554, 1567  
`\inserthangingsymbolL` .... 894, 1575  
`\inserthangingsymbolR` .... 929, 1575  
`\insertlines@listR` .....  
.... 76, 215, 229, 521, 1231, 1235  
`\inserts@list` .....  
.. 786, 1267, 1281, 1295, 1309,  
1323, 1338, 1352, 1366, 1380,  
1394, 1498, 1510, 1522, 1557, 1570  
`\inserts@listR` .....  
.. 816, 1226, 1229, 1239, 1253,  
1254, 1263, 1277, 1291, 1305,  
1319, 1334, 1348, 1362, 1376,  
1390, 1494, 1506, 1518, 1553, 1566  
`\istanzaLfalse` ..... 1794, 1947  
`\istanzaLtrue` ..... 729  
`\istanzaRfalse` ..... 1795, 1948  
`\istanzaRtrue` ..... 751  
`\interfootnotelinepenalty` ..... 2291  
`\interlinepenalty` ..... 1594, 2291  
`\interstanza` .....  
.... 2552, 2556, 2581, 2625, 2654
- L**
- `\l@d@nums` ..... 678, 681, 687,  
690, 1263, 1267, 1277, 1281,  
1291, 1295, 1305, 1309, 1319,  
1323, 1334, 1338, 1348, 1352,  
1362, 1366, 1376, 1380, 1390, 1394  
`\l@d@set` ..... 386, 619, 620  
`\l@dbbl@set@language` .... 1689, 1712  
`\l@dbfnote` ..... 1549  
`\l@dc@maxchunks` ..... 792, 794,  
822, 824, 1637, 1647, 1655, 1662  
`\l@dcalc@maxoftwo` .....  
..... 1867, 2011, 2089, 2117  
`\l@dcalc@minoftwo` .. 1875, 1935, 2011  
`\l@dcalcnun` ..... 1054  
`\l@dchecklang` ..... 1682, 1756  
`\l@dchset@num` ..... 261, 264, 386  
`\l@dcsnote` ..... 1490  
`\l@dcsnotetext` .....  
.... 1529, 1532, 1534, 1542, 1544  
`\l@dedendmini` ..... 1400  
`\l@emptyd@ta` ..... 881, 916  
`\l@dend@stuff` .... 46, 65, 98, 112, 120  
`\l@dgetline@margin` ..... 152  
`\l@dgetsidenote@margin` ..... 1479  
`\l@dld@ta` ..... 890, 925,  
1141, 1153, 1165, 1177, 1186, 1198  
`\l@dleftbox` .....  
.. 862, 886, 902, 1768, 1896, 1898  
`\l@dlinenumR` ..... 197  
`\l@dlsn@te` ..... 893, 928  
`\l@dlsnote` ..... 1490  
`\l@dmake@labels` ..... 1452  
`\l@dmake@labelsR` ..... 1440, 1456  
`\l@dminpagelines` .....  
.... 1838, 1876, 1936, 2032, 2046

- \l@dnumpstartsL . 39, 791, 792, 794,  
796, 1641, 1673, 1738, 1739,  
1791, 1805, 1847, 1848, 1945, 2079
- \l@dnumpstartsR . 52, 821, 822, 824,  
826, 1641, 1674, 1738, 1739,  
1792, 1808, 1847, 1848, 1946, 2107
- \l@doldbbl@set@language ..... 1711
- \l@doldselectlanguage 1710, 1714, 1719
- \l@dpagetrue ..... 2024
- \l@dpagetrue ..... 2024
- \l@dpagingfalse ..... 11, 697, 710
- \l@dpagingtrue ..... 705
- \l@dpairingfalse ..... 9, 699, 709
- \l@dpairingtrue ..... 696, 704
- \l@dparsedendline ..... 2301
- \l@dparsedstartline . 2296, 2301, 2303
- \l@dparsedstartpage ..... 2299, 2304
- \l@dparsefootspec ..... 2295
- \l@dpscL 877, 882, 1643, 1675, 1747,  
1753, 1789, 1805, 1818, 1857,  
1863, 1868, 1871, 1879, 1943,  
2070, 2072, 2079, 2089, 2091, 2093
- \l@dpscR .... 912, 917, 1644, 1676,  
1748, 1754, 1790, 1808, 1821,  
1858, 1864, 1872, 1880, 1944,  
2098, 2100, 2107, 2117, 2119, 2121
- \l@drd@ta ..... 894, 929,  
1143, 1151, 1167, 1175, 1188, 1196
- \l@drighthbox .....  
.. 862, 921, 937, 1770, 1917, 1919
- \l@drsn@te ..... 895, 930
- \l@drsnote ..... 1490
- \l@dsamelangfalse ..... 1680, 1683
- \l@dsamelangtrue ..... 1680, 1686
- \l@dsamepagefalse ..... 2024
- \l@dsamepagetrue ..... 2024
- \l@dsetupmaxlinecounts .. 1654, 1671
- \l@dsetuprawboxes ..... 1646, 1670
- \l@dskipnumberfalse ..... 1118
- \l@dskipnumbertrue ..... 1014
- \l@dunhbox@line ..... 894, 929
- \l@dusedbabelfalse ..... 1678, 1707
- \l@dusedbabeltrue ..... 1678, 1709
- \l@duselanguage .....  
.... 1699, 1762, 1764, 1889, 1912
- \l@deromaxlinecounts ... 1654, 1672
- \l@deropenalties 835, 851, 1742, 1852
- \l@pscL ..... 1643
- \l@pscR ..... 1643
- \label@refs .....  
1432, 1434, 1440, 1444, 1446, 1452
- \labelref@list ..... 1443, 1446, 1474
- \labelref@listR 1426, 1431, 1434, 1470
- \language ..... 1690, 1691, 1693–1696
- \last@page@num ..... 326, 332
- \last@page@numR ..... 312
- \lastbox ..... 884, 919
- \lastskip ..... 590, 596
- \lcolwidth .... 5, 6, 13, 706, 797,  
887, 902, 2152, 2153, 2196, 2197
- \ldots ..... 2215,  
2218, 2238, 2241, 2405, 2407, 2436
- \led@err@BadLeftRightPstarts ...  
..... 21, 1739, 1848
- \led@err@LeftOnRightPage ... 24, 1966
- \led@err@LineationInNumbered ... 128
- \led@err@NumberingNotStarted ... 87
- \led@err@numberingShouldHaveStarted  
..... 100
- \led@err@NumberingStarted .... 36, 49
- \led@err@PendNoPstart ..... 833, 849
- \led@err@PendNotNumbered ... 830, 846
- \led@err@PstartInPstart ... 781, 811
- \led@err@PstartNotNumbered . 777, 807
- \led@err@RightOnLeftPage ... 24, 1973
- \led@err@TooManyPstarts 18, 793, 823
- \led@mess@NotesChanged ..... 84
- \led@mess@SectionContinued .....  
..... 96, 110, 118
- \led@warn@BadAction ..... 1016
- \led@warn@BadAdvancelineLine 356, 362
- \led@warn@BadAdvancelineSubline .  
..... 342, 348
- \led@warn@BadLineation ..... 145
- \led@warn@BadSetline ..... 609
- \led@warn@BadSetlinenum ..... 617
- \led@warn@DuplicateLabel ..... 1458
- \ledllfill ..... 894, 929
- \ledmac@error ..... 19, 22, 25, 27
- \ledplinenumfalse ..... 2297
- \ledplinenumtrue ... 2295, 2300, 2302
- \ledRcolfalse ..... 12, 721, 753
- \ledRcoltrue ..... 739
- \ledrlfill ..... 894, 929
- \ledsavedprintlines ..... 8, 1413
- \ledsetnormalparstuff 2328, 2336, 2349
- \ledsidenote ..... 2469
- \ledstrutL ..... 1896, 1898, 1951
- \ledstrutR ..... 1917, 1919, 1951

- \ledthegoal . . . . . 2031, 2045, 2056
  - \leftlinenumR . . . . . 197, 1141, 1153
  - \leftpstartnumL . . . . . 1160
  - \leftpstartnumR . . . . . 1160
  - Leftside (environment) . . . . . 6, 720
  - \Leftsidehook . . . . . 727, 733
  - \Leftsidehookend . . . . . 732, 733
  - \lemma . . . . . 2210, 2436
  - \lemmafont . . . . . 2331, 2352
  - \line@list . . . . . 685, 689
  - \line@list@stuff . . . . . 45, 111
  - \line@list@stuffR . . . . . 64, 97, 119, 561
  - \line@listR . . . . . 79, 215, 228, 539, 676, 680
  - \line@margin . . . . . 157, 1170
  - \line@marginR . . . . . 150, 1146, 1191
  - \line@num 329, 360, 361, 363, 381, 392,  
393, 421, 897, 974, 1472, 1576, 1578
  - \line@numR . . . . . 57, 204,  
211, 266, 300, 319, 354, 355,  
357, 374, 388, 389, 412, 532,  
536, 932, 955, 1027, 1036, 1125,  
1127, 1129, 1130, 1468, 1577, 1581
  - \lineation . . . . . 748, 2268
  - \lineationR . . . . . 126, 748
  - \linenum@out . . . . . 580,  
588, 593, 599, 605, 612, 620,  
625, 629, 1442, 1778, 1829, 2005
  - \linenum@outR . . . . . 558, 564, 566,  
568, 569, 573, 576, 586, 592,  
598, 604, 611, 619, 624, 628,  
633, 1430, 1781, 1834, 2008, 2112
  - \linenumberlist . . . . . 1126, 1130
  - \linenumincrement . . . . . 6, 171, 2158,  
2173, 2203, 2224, 2543, 2584, 2657
  - \linenummargin . . . . .  
150, 2159, 2174, 2204, 2225, 2269
  - \linenumr@p . . . . . 1416, 1420, 1468, 1472
  - \linenumrepR . . . . . 194, 204
  - \linenumsep . . . . .  
. 199, 201, 1207, 1210, 1219, 1222
  - \linesinpar@listL . . . . .  
. . . . . 220, 236, 546, 1979, 1982
  - \linesinpar@listR . . . . .  
. . . . . 220, 232, 549, 1985, 1988
  - \linesonpage@listL 237, 553, 1992, 1995
  - \linesonpage@listR 233, 556, 1998, 2001
  - \list@clear . . . . .  
. 228–233, 236, 237, 239, 786, 816
  - \list@clearing@reg . . . . . 235
  - \list@create . . . . .  
. . . . . 215–218, 220–222, 1226, 1426
  - \lock@disp . . . . . 1086, 1090, 1095
  - \lock@off . . . . . 470, 471, 479, 628, 629
  - \lock@on . . . . . 624, 625
  - \longdash . . . . . 2538, 2552
- M
- \maxchunks . . . . . 4, 1637, 2535
  - \maxdimen . . . . . 2305
  - \maxlinesinpar@list . . . . . 220, 239
  - \memorydump . . . . . 8, 726, 744
  - \memorydumpL . . . . . 105, 726
  - \memorydumpR . . . . . 105, 744
  - \message . . . . . 44, 63
  - \morenoexpands . . . . . 2323
  - \mpAfootgroup . . . . . 1408
  - \mpAfootins . . . . . 1408
  - \mpAfootnote . . . . . 1330
  - \mpBfootgroup . . . . . 1409
  - \mpBfootins . . . . . 1409
  - \mpBfootnote . . . . . 1330
  - \mpCfootgroup . . . . . 1410
  - \mpCfootins . . . . . 1410
  - \mpCfootnote . . . . . 1330
  - \mpDfootgroup . . . . . 1411
  - \mpDfootins . . . . . 1411
  - \mpDfootnote . . . . . 1330
  - \mpEfootgroup . . . . . 1412
  - \mpEfootins . . . . . 1412
  - \mpEfootnote . . . . . 1330
  - \mpvAfootnote . . . . . 1333, 1337, 1342
  - \mpvBfootnote . . . . . 1347, 1351, 1356
  - \mpvCfootnote . . . . . 1361, 1365, 1370
  - \mpvDfootnote . . . . . 1375, 1379, 1384
  - \mpvEfootnote . . . . . 1389, 1393, 1398
  - \multiply . . . . . 1060
- N
- \n@num . . . . . 507, 633
  - \n@num@reg . . . . . 513
  - \namebox . . . . . 877, 882, 912,  
917, 1621, 1818, 1821, 2070, 2098
  - \NeedsTeXFormat . . . . . 2
  - \new@line . . . . . 894
  - \new@lineR . . . . . 572, 929
  - \newbox . . . . . 759, 862, 863, 1622
  - \newcounter . . . . . 162,  
164, 166, 168, 762, 763, 765, 766

`\newif` ..... 5, 10, 31, 122, 123,  
           559, 718, 719, 1214, 1678, 1680,  
           1802, 1814, 2024, 2026, 2066, 2067  
`\newnamebox` ..... 1621, 1649, 1650  
`\newnamecount` ..... 1632, 1657  
`\newwrite` ..... 558  
`\next@action` ..... 256  
`\next@actionline` ..... 253, 255  
`\next@actionlineR` .....  
       .. 245, 247, 985, 1023, 1045, 1047  
`\next@actionR` ..... 248, 986,  
           1024, 1025, 1030, 1031, 1039, 1048  
`\next@insert` ..... 787  
`\next@insertR` .....  
       817, 1230, 1233, 1235, 1238, 1242  
`\next@page@num` ..... 333, 404  
`\next@page@numR` 61, 269, 271, 323, 401  
`\no@expands` ..... 641, 658  
`\nobrak` ..... 2321, 2322  
`\noindent` ..... 2306, 2381, 2461  
`\normal@pars` ..... 72, 790, 820  
`\normalbfnoteX` ..... 1562  
`\notefontsetup` ..... 2290  
`\notenumfont` ... 2274, 2330, 2338, 2351  
`\noteschanged@true` .....  
       ..... 77, 80, 677, 686, 1232  
`\notetextfont` .. 2275, 2332, 2345, 2353  
`\num@lines` ..... 836, 1743, 1853  
`\num@linesR` ..... 758, 852, 1744, 1854  
`\numberedpar@true` ..... 798, 828  
`\numberingRfalse` ..... 71  
`\numberingRtrue` ..... 54, 91, 115  
`\numberingtrue` ..... 41, 107  
`\numberpstartfalse` ..... 9  
`\numberpstarttrue` ..... 9  
`\numlabfont` ..... 204  
`\numpagelinesL` .....  
       .... 1838, 1894, 1905, 1909, 2032  
`\numpagelinesR` .....  
       .... 1838, 1915, 1925, 1929, 2046

## O

`\oldBfootfmt` ... 2134, 2137, 2545, 2548  
`\oldchapter` ..... 702, 711  
`\oldstanza` 728, 729, 731, 750, 751, 754  
`\one@line` ..... 882, 884, 894  
`\one@lineR` ..... 758, 917, 919, 929  
`\openout` ..... 566, 569

## P

`\page@action` ..... 270, 398, 526  
`\page@num` ..... 251, 331, 1172, 1539  
`\page@numR` ..... 224,  
           243, 321, 531, 536, 1025, 1148, 1193  
`\pagegoal` ..... 2064  
`\Pages` ..... 6, 1842, 2520, 2620, 2697  
`pages` (environment) ..... 6, 695  
`\pagetotal` ..... 1958, 2031, 2045  
`pairs` (environment) ..... 5, 695  
`\par@line` ..... 837, 1745, 1855  
`\par@lineR` ..... 758, 853, 1746, 1856  
`\para@vfootnote` ..... 2287  
`\pausenumbering` ..... 742  
`\pausenumberingR` ..... 90, 742  
`\pend` .. 7, 725, 747, 782, 1607, 2414,  
       2446, 2450, 2493, 2513, 2517, 2552  
`\pendL` ..... 725, 829  
`\pendR` ..... 747, 812, 845  
`\prevgraf` .....  
       . 836, 852, 1743, 1744, 1853, 1854  
`\previous@A@number` ..... 2315  
`\previous@B@number` ..... 2316  
`\previous@C@number` ..... 2317  
`\previous@page` ..... 2299, 2304, 2318  
`\prevline@num` ..... 897, 1576, 1578  
`\prevline@numR` ..... 932, 1575, 1581  
`\printindex` ..... 2524  
`\printlines` .....  
       1424, 2136, 2330, 2338, 2351, 2547  
`\printlinesR` ..... 8, 1413, 2136, 2547  
`\ProcessOptions` ..... 8  
`\protected@write` ... 1439, 1451, 1693  
`\ProvidesPackage` ..... 3  
`\pst@rtedLfalse` ..... 30, 40  
`\pst@rtedLtrue` ..... 108, 788  
`\pst@rtedRfalse` ..... 32, 53, 74  
`\pst@rtedRtrue` ..... 94, 116, 818  
`\pstart` 7, 19, 23, 723, 746, 1609, 2377,  
       2416, 2448, 2457, 2495, 2515, 2552  
`\pstartL` ..... 723, 761  
`\pstartnumfalse` ..... 1207, 1212  
`\pstartnumRfalse` ..... 1219, 1224  
`\pstartnumRtrue` ..... 1215, 1752, 2116  
`\pstartnumtrue` ..... 1751, 2088  
`\pstartR` ..... 746, 761

## R

`\rbracket` ..... 2321, 2344

- `\Rcolwidth` ..... 5, 6,  
     13, 707, 827, 922, 937, 2153, 2197  
`\read@linelist` ..... 226, 562  
`\rem@inder` ..... 1130, 1132–1134  
`\resumenumbering` ..... 743  
`\resumenumberingR` ..... 90, 743  
`\rightlinenumR` ..... 197, 1143, 1151  
`\rightpstartnumL` ..... 1160  
`\rightpstartnumR` ..... 1160  
`Rightside (environment)` ..... 6, 738  
`\Rightsidehook` ..... 733, 749  
`\Rightsidehookend` ..... 733, 755  
`\rlap` 1143, 1151, 1167, 1175, 1188, 1196  
`\Rlineflag` ..... 8, 192, 204,  
     1416, 1420, 1460, 2132, 2149, 2366  
`\rule` ..... 1798
- S**
- `\sc@n@list` ..... 1131, 1133  
`\secdef` ..... 716  
`\section@num` ..... 42, 44, 45, 109–111  
`\section@numR` .....  
     ... 28, 55, 63, 64, 95–97, 117–119  
`\select@language` ... 1691, 1693–1696  
`\select@lemmafnt` ..... 2344  
`\selectlanguage` .... 1699, 2378, 2455  
`\set@line` ..... 643, 660, 674  
`\set@line@action` .....  
     ..... 263, 367, 376, 383, 406, 528  
`\setl@dlp@rbox` ..... 1532, 1544  
`\setl@drp@rbox` ..... 1534, 1542  
`\setline` ..... 607, 2084, 2381, 2461  
`\setlinenum` ..... 615  
`\setnamebox` ..... 796, 826, 1621  
`\setprintlines` ..... 1414  
`\setstanzaindents` .....  
     ..... 2147, 2194, 2536, 2585, 2658  
`\shiftedversesfalse` ..... 6  
`\shiftedversestrue` ..... 7  
`\showlemma` ..... 649, 666  
`\sidenote@margin` ..... 1484, 1488  
`\sidenote@marginR` ..... 1477, 1537  
`\sidenotemargin` .... 1477, 2270, 2455  
`\skip` ..... 2281–2283  
`\skip@lockoff` ..... 471, 479  
`\skipnumbering` ..... 9, 632, 2552  
`\skipnumbering@reg` ..... 636  
`\smash` ..... 1798  
`\splitmaxdepth` ..... 2293  
`\splittopskip` ..... 879, 914, 2293
- `\stanza` ... 728, 729, 731, 750, 751,  
     754, 2161, 2176, 2206, 2227,  
     2559, 2565, 2571, 2587, 2595, 2603  
`\stanza@count` ..... 1589, 1603, 1614  
`\stanza@hang` ..... 1591, 1616  
`\stanzaindentbase` .. 1579, 1582, 1614  
`\startlock` ..... 623  
`\startstanzahook` ..... 1587  
`\startsub` ..... 590  
`\sub@action` ..... 279, 427, 527  
`\sub@change` ..... 62, 273, 274, 280  
`\sub@lock` ..... 969  
`\sub@lockR` ..... 59, 288, 290, 292,  
     295, 445, 451, 452, 454, 455,  
     485, 486, 488, 950, 1006, 1008,  
     1009, 1011, 1067, 1107, 1109, 1111  
`\sub@off` ..... 598, 599  
`\sub@on` ..... 592, 593  
`\subline@num` ..... 206, 329, 346,  
     347, 349, 379, 419, 970, 975, 1473  
`\subline@numR` ..... 207,  
     211, 296, 300, 319, 340, 341,  
     343, 372, 410, 533, 537, 951,  
     956, 1027, 1034, 1121, 1122, 1469  
`\sublinenumincrement` ..... 6, 171  
`\sublinenumr@p` . 1417, 1421, 1469, 1473  
`\sublinenumrepR` ..... 194, 207  
`\sublines@false` ..... 60, 277, 996  
`\sublines@true` ..... 275, 994  
`\sublock@disp` ..... 1069, 1073, 1078  
`\symplinenum` ..... 1416  
`\sza@penalty` ..... 1598, 1602
- T**
- `\textdagger` ..... 2480  
`\textheight` ..... 2129, 2533  
`\textit` 2191, 2216, 2218, 2239, 2241,  
     2251, 2391, 2404, 2425, 2430,  
     2436, 2438, 2440, 2488, 2607, 2682  
`\textnormal` ..... 2322  
`\textsc` ..... 2190, 2386  
`\textwidth` 14, 16, 706, 707, 2152, 2196  
`\theledlanguageL` 1684, 1699, 1762, 1889  
`\theledlanguageR` 1684, 1699, 1764, 1912  
`\thepage` ..... 573, 1440, 1452  
`\thepstart` ..... 724, 745  
`\thepstartL` 9, 724, 764, 796, 1206, 1211  
`\thepstartR` 9, 745, 767, 826, 1218, 1223  
`\thr@@` .. 454, 463, 486, 493, 1001, 1009  
`\title` ..... 2371

<code>\topskip</code> . . . . .	1958		
<b>U</b>		<b>W</b>	
<code>\unhbox</code> . . . . .	1626, 1768, 1770, 1896, 1898, 1917, 1919	<code>\wd</code> . . . . .	894, 929, 2309
<code>\unhnamebox</code> . . . . .	<u>1621</u>	<code>\writtenlinesLfalse</code> . . . . .	1859, 2080
<code>\unvbox</code> . . . . .	884, 919, 1628	<code>\writtenlinesLtrue</code> . . . . .	2077
<code>\unvnamebox</code> . . . . .	<u>1621</u>	<code>\writtenlinesRfalse</code> . . . . .	1860, 2108
<code>\unvxh</code> . . . . .	2307	<code>\writtenlinesRtrue</code> . . . . .	2105
<code>\usernamecount</code> . . . . .	. 1590, 1597, <u>1632</u> , 1664, 1868, 2072, 2089, 2091, 2100, 2117, 2119	<b>X</b>	
<code>\usepackage</code> . . . . .	2130, 2265–2267, 2532	<code>\x@lemma</code> . . . . .	651–653, 668–670
<b>V</b>		<code>\xlineref</code> . . . . .	2614, 2689
<code>\vAfootnote</code> . . . . .	1262, 1266, 1271	<code>\xpg@main@language</code> . . . . .	1729, 1730
<code>\value</code> . . . . .	785, 815, 1736, 1737, 1843, 1844, 2082, 2110	<code>\xpg@set@language</code> . . . . .	1724, 1728
<code>\vbadness</code> . . . . .	878, 913	<code>\xright@appenditem</code> . . . . .	. . . . . 400, 401, 403, 404, 408, 415, 417, 424, 429, 431, 433, 436, 438, 440, 448, 450, 459, 482, 484, 491, 510, 511, 521, 535, 546, 549, 553, 556, 1262, 1266, 1276, 1280, 1290, 1294, 1304, 1308, 1318, 1322, 1333, 1337, 1347, 1351, 1361, 1365, 1375, 1379, 1389, 1393, 1468, 1472, 1493, 1497, 1505, 1509, 1517, 1521, 1552, 1556, 1565, 1569
<code>\vbfnoteX</code> . . . . .	1565, 1569	<b>Z</b>	
<code>\vBfootnote</code> . . . . .	1276, 1280, 1285	<code>\z@skip</code> . . . . .	2294
<code>\vbox</code> . . . . .	796, 826, 2305	<code>\zz@@@</code> . . . . .	1432, 1444
<code>\vCfootnote</code> . . . . .	1290, 1294, 1299		
<code>\vDfootnote</code> . . . . .	1304, 1308, 1313		
<code>\vEfootnote</code> . . . . .	1318, 1322, 1327		
<code>\vl@dbfnote</code> . . . . .	1552, 1556		
<code>\vl@dcsnote</code> . . . . .	1517, 1521		
<code>\vl@dlsnote</code> . . . . .	1493, 1497		
<code>\vl@drsnote</code> . . . . .	1505, 1509		
<code>\vsplit</code> . . . . .	882, 917		

## Change History

v0.1		<code>\do@lineR</code> to allow line numbering by <code>pstart</code> (like in <code>ledmac</code> 0.15). . . . .	38
General: First public release . . . . .	1	Lineation can be by <code>pstart</code> (like in <code>ledmac</code> 0.15). . . . .	16
v0.10		New management of hangingsymbol insertion, preventing undesirable insertions. . . . .	56
General: <code>\edlabel</code> commands on the right side are now correctly indicated. . . . .	1	Prevent shift of column separator when a verse is hanged . . . . .	56
<code>\edlabel</code> commands which start a paragraph are now put in the right place. . . . .	1	<code>\affixline@numR:</code> . . . . .	Changed
v0.11			
General: Change <code>\do@lineL</code> and			



\affixline@numR to allow to disable line numbering (like in ledmac 0.15). . . . .	42	\ledsavedprintlines: Simpli- fied \printlinesR by using \setprintlines . . . . .	52
\Columns: Line numbering by pstart. . . . .	63	\ledstrutR: Added \ledtrutL and \ledstrutR . . . . .	68
\get@nextboxR: Change \get@nextboxL and \get@nextboxR to allow to disable line numbering (like in ledmac 0.15). . . . .	71	\normalbfnoteX: Removed extraneous spaces from \normalbfnoteX . . . . .	55
Pstart number can be printed in side . . . . .	71	\Pages: Added \ledstrutL to \Pages . . . . .	66
v0.2		Added \ledstrutR to \Pages .	67
General: Added section of babel re- lated code . . . . .	59	\Rightsidehookend: Added \Leftsidehook, \Leftsidehookend, \Rightsidehook and \Rightsidehookend . . . . .	34
Fix babel problems . . . . .	1	\sublinenumrepR: Added \linenumrepR and \sublinenumrepR . . . . .	18
\Columns: Added \l@dchecklang and \l@duselanguage to \Columns . . . . .	62	v0.3a	
\Pages: Added \l@duselanguage to \Pages . . . . .	66	General: Minor \linenummargin fix . . . . .	1
v0.3		\line@marginR: Don't just set \line@marginR in \linenummargin . . . . .	17
General: Reorganize for ledarab . .	1	v0.3b	
\affixline@numR: Changed \affixline@numR to match new ledmac . . . . .	42	General: Improved parallel page balancing . . . . .	1
\do@actions@nextR: Used \do@actions@fixedcode in \do@actionsR . . . . .	41	\Pages: Added \l@dminpagelines calculation for succeeding page pairs . . . . .	67
\do@lineL: Added \do@lineLhook to \do@lineL . . . . .	38	v0.3c	
Simplified \do@lineL by using macros for some common code	38	General: Compatibilty with Poly- glossia . . . . .	1
\do@lineR: Changed \do@lineR similarly to \do@lineL . . . .	39	v0.4	
\do@lineRhook: Added \do@lineLhook and \do@lineRhook . . . . .	39	General: No more ledparpatch. All patches are now in the main file. . . . .	1
Leftside: Added hooks into Left- side environment . . . . .	33	v0.5	
\flag@end: Removed extraneous spaces from \flag@end . . . .	29	General: Corrections about \section and other titles in numbered sections . . . . .	1
\ifledRcol: Moved \ifl@dpairing to ledmac . . . . .	13	v0.6	
\ifpst@rtedR: Moved \ifpst@rtedL to ledmac . . . . .	14	General: Be able to us \chapter in parallel pages. . . . .	1
\l@dlinenumR: Simplified \leftlinenumR and \rightlinenumR by introducing \l@dlinenumR	18	v0.7	
\l@dnumpstartsR: Moved \l@dnumpstartsL to ledmac .	58	General: Option 'shiftedverses' which make there is no blank between two parallel verses with inequal length. . . . .	1

v0.8		v0.9.1	
General: Possibility to have a symbol on each hanging of verses, like in the french typography. Redefine the commande <code>\hangingsymbol</code> to define the character. . . . .	1	General: The numbering of the pstarts restarts on each <code>\beginnumbering</code> . . . . .	1
v0.9		v0.9.2	
General: Possibility to number <code>\pstart</code> . . . . .	9	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. . . . .	1
Possibility to number the pstart with the commands <code>\numberpstarttrue</code> . . . . .	1	v0.9.3	
<code>\iflledRcol</code> : Moved <code>\iflledRcol</code> and <code>\ifnumberingR</code> to ledmac	13	General: <code>\thepstartL</code> and <code>\thepstartR</code> use now <code>\bfseries</code> and not <code>\bf</code> , which is deprecated and makes conflicts with memoir class. . . . .	1