

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

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

Abstract

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

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

Contents

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

*This file (**ledpar.dtx**) has version number v0.13.2, last revised 2012/09/08.

[†]herries dot press at earthlink dot net

[‡]maieul at maieul dot net

9 Preliminaries	12
9.1 Messages	13
10 Sectioning commands	13
11 Line counting	16
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
12 Marking text for notes	31
13 Parallel environments	32
14 Paragraph decomposition and reassembly	34
14.1 Boxes, counters, <code>\pstart</code> and <code>\pend</code>	35
14.2 Processing one line	38
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	47
14.8 Printing leftover notes	48
15 Footnotes	48
15.1 Outer-level footnote commands	48
15.2 Normal footnote formatting	52
16 Cross referencing	53
17 Side notes	54
18 Familiar footnotes	56
19 Verse	57
20 Naming macros	59
21 Counts and boxes for parallel texts	59
22 Fixing babel	61
23 Parallel columns	63
24 Parallel pages	66

<i>List of Figures</i>	3
25 The End	74
A Examples	75
A.1 Parallel column example	84
A.2 Example parallel facing pages	86
A.3 Example poetry on parallel facing pages	92
References	97
Index	97
Change History	106

List of Figures

1	Output from <code>villon.tex</code>	77
2	Left page output from <code>djd17nov.tex</code>	78
3	Right page output from <code>djd17nov.tex</code>	79
4	First left page output from <code>djdpoems.tex</code>	80
5	First right page output from <code>djdpoems.tex</code>	81
6	Second left page output from <code>djdpoems.tex</code>	82
7	Second right page output from <code>djdpoems.tex</code>	83

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 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**¹ and the **ledpar** package use macros instead.

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

¹when 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}[2012/09/08 v0.13.2 ledmac extension for parallel texts]
4
```

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

```

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

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

```

\ifl@dpairing \ifl@dpairing is set TRUE if we are processing parallel texts and \ifl@dpaging
\ifl@dpaging is also set TRUE if we are doing parallel pages. \ifledRcol is set TRUE if we
\ifledRcol are doing the right hand text. \ifl@dpairing is defined in ledmac.
9 \l@dpairingfalse
10 \newif\ifl@dpaging
11 \l@dpagingfalse
12 \ledRcolfalse

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

```

9.1 Messages

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

```

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

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

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

```

10 Sectioning commands

`\section@numR` This is the right side equivalent of `\section@num`.

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

```

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

```

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

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

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

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

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

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

`\endnumbering` This is the left text version of the regular `\endnumbering` and must follow the last

text for a left text numbered section. It sets `\ifpst@rtedL` to FALSE. It is fully defined in `ledmac`.

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

```

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

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

```

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

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

```

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

```

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

```

11 Line counting

11.1 Choosing the system of lineation

M 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{firstlinenum}{#1}%
178   \fi}
179 \renewcommand*\linenumincrement[1]{%
180   \ifledRcol \setcounter{linenumincrementR}{#1}%
181   \else      \setcounter{linenumincrement}{#1}%
182   \fi}
183 \renewcommand*\firstsublinenum[1]{%
184   \ifledRcol \setcounter{firstsublinenumR}{#1}%
185   \else      \setcounter{firstsublinenum}{#1}%
186   \fi}
187 \renewcommand*\sublinenumincrement[1]{%
188   \ifledRcol \setcounter{sublinenumincrementR}{#1}%
189   \else      \setcounter{sublinenumincrement}{#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     \glp\actionlines@listR\to\next@actionlineR
248     \glp\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     \glp\actionlines@list\to\next@actionline
256     \glp\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`. If needed, it resets line number. 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 \ifbypstart@global\line@num=0\fi%
547 \xright@appenditem{#1}\to\linesinpar@listL}
548 \providecommand*\@pendR}[1]{%
549 \renewcommand*\@pendR}[1]{%
550 \ifbypstart@Rglobal\line@numR=0\fi
551 \xright@appenditem{#1}\to\linesinpar@listR}
552
```

`\@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.

```
553 \providecommand*\@lopL}[1]{%
554 \renewcommand*\@lopL}[1]{%
555 \xright@appenditem{#1}\to\linesonpage@listL}
556 \providecommand*\@lopR}[1]{%
557 \renewcommand*\@lopR}[1]{%
558 \xright@appenditem{#1}\to\linesonpage@listR}
559
```

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`.

```
560 \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 561 \newif\iffirst@linenum@out@R
562 \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.

```

563 \newcommand*{\line@list@stuffR}[1]{%
564   \read@linelist{#1}%
565   \iffirst@linenum@out@R
566     \immediate\closeout\linenum@outR
567     \global\first@linenum@out@Rfalse
568     \immediate\openout\linenum@outR=#1
569   \else
570     \closeout\linenum@outR
571     \openout\linenum@outR=#1
572   \fi}
573

```

`\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.

```

574 \newcommand*{\new@lineR}{%
575   \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.

```

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

```

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

```

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

```

```

600 \ifledRcol \write\linenum@outR{\string\sub@off}%
601 \else      \write\linenum@out{\string\sub@off}%
602 \fi
603 \ifdim\dimen0>0pt \hskip\dimen0 \fi}
604

```

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

```

605 \renewcommand*{\advanceline}[1]{%
606 \ifledRcol \write\linenum@outR{\string\@adv[#1]}%
607 \else      \write\linenum@out{\string\@adv[#1]}%
608 \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.

```

609 \renewcommand*{\setline}[1]{%
610 \ifnum#1<\z@
611 \led@warn@BadSetline
612 \else
613 \ifledRcol \write\linenum@outR{\string\@set[#1]}%
614 \else      \write\linenum@out{\string\@set[#1]}%
615 \fi
616 \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.

```

617 \renewcommand*{\setlinenum}[1]{%
618 \ifnum#1<\z@
619 \led@warn@BadSetlinenum
620 \else
621 \ifledRcol \write\linenum@outR{\string\l@d@set[#1]}
622 \else      \write\linenum@out{\string\l@d@set[#1]} \fi
623 \fi}
624

```

\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.

```

625 \renewcommand*{\startlock}{%
626 \ifledRcol \write\linenum@outR{\string\lock@on}%
627 \else      \write\linenum@out{\string\lock@on}%
628 \fi}
629 \def\endlock{%
630 \ifledRcol \write\linenum@outR{\string\lock@off}%
631 \else      \write\linenum@out{\string\lock@off}%
632 \fi}
633

```

`\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.

```

634 \renewcommand*{\skipnumbering}{%
635   \ifledRcol \write\linenum@outR{\string\n@num}%
636             \advanceline{-1}%
637   \else
638     \skipnumbering@reg
639   \fi}
640

```

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.

```

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

```

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

```

657 \renewcommand{\edtext}[2]{\leavevmode
658   \begin{group}
659     \renewcommand{\@tag}{\no@expands #1}%
660     \set@line
661     \ifl@edRcol \global\insert@countR \z@
662     \else      \global\insert@count \z@ \fi
663     \ignorespaces #2\relax
664     \flag@start
665   \end{group}
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   \ifl@edRcol
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
`pages`
`chapterinpages`

parallel pages.

```

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

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

`\pstartR`

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

We have to have specific left and right `\pstart` when parallel processing; among other things because of potential changes in the linewidth. The old counters are used to have the good reset of the `pstart` counters at the beginning 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 \ifpstart@rtedL to FALSE. Save the pstartL counter.

```
784 \ifpstart@rtedL\else
785   \setcounter{pstartLold}{\value{pstartL}}%
786   \list@clear{\inserts@list}%
787   \global\let\next@insert=\empty
788   \global\pstart@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@dc@colrawbox\the\l@dnumpstartsL}=\vbox\bgroup\ifautopar\else\ifnumb
797   \hsize=\l@colwidth
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 \ifpstart@rtedR\else
815   \setcounter{pstartRold}{\value{pstartR}}%
816   \list@clear{\inserts@listR}%
817   \global\let\next@insertR=\empty
818   \global\pstart@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@dc@colrawbox\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@demtyd@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   \ifnum\@lock>\@ne\inserthangingsymboltrue\else\inserthangingsymbolfalse\fi
887   \setbox\l@dleftbox
888   \hb@xt@ \Lcolwidth{%
889     \affixpstart@numL
890     \affixline@num
891     \l@dld@ta
892     \add@inserts
893     \affixside@note

```

```

894 \l@dlsn@te
895 {\ledllfill\hb@xt@ \wd\one@line{\inserthangingsymbolL\new@line\l@dunhbox@line{\one@line}}\correct
896 \l@drsn@te
897 }}%
898 \add@penaltiesL
899 \global\advance\@donereallinesL\@ne
900 \global\advance\@donetotallinesL\@ne
901 \else
902 \setbox\l@dleftbox \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@demptyd@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 \ifnum\@lockR>\@ne\inserthangingsymbolRtrue\else\inserthangingsymbolRfalse\fi
922 \setbox\l@drightbox
923 \hb@xt@ \Rcolwidth{%
924 \affixpstart@numR
925 \affixline@numR
926 \l@dld@ta
927 \add@insertsR
928 \affixside@noteR
929 \l@dlsn@te
930 {\correctchangingR\ledllfill\hb@xt@ \wd\one@lineR{\inserthangingsymbolR\new@lineR\l@dunhbox@line{\l@drsn@te
931 \l@drsn@te
932 }}%
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 line.
`\do@actions@fixedcodeR`
`\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@tempcnta%
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@tempcntb=\absline@numR

```

```

1023 \ifnum\@l@tempcntb<\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 \gl@p\actionlines@listR\to\next@actionlineR
1048 \gl@p\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@dcalcnum \affixline@numR is the right text version of the \affixline@num macro.

```

\ch@cksub@l@ckR 1054
\ch@ck@l@ckR 1055 \providecommand*\l@dcalcnum}[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@l@tempcntb=\subline@numR
1122 \l@dcalcnnum{\subline@numR}{\c@firstsublinenumR}{\c@sublinenumincrementR}%
1123 \ch@cksub@lockR
1124 \else
1125 \l@l@tempcntb=\line@numR
1126 \ifx\linenumberlist\empty
1127 \l@dcalcnnum{\line@numR}{\c@firstlinenumR}{\c@linenumincrementR}%
1128 \else
1129 \l@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@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@l@tempcnta\@ne\fi
1135 \fi
1136 \ch@ck@l@ckR
1137 \fi
1138 \ifnum\l@l@tempcnta=\l@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@l@tempcntb=\line@marginR
1147 \ifnum\l@l@tempcntb>\@ne
1148 \advance\l@l@tempcntb by\page@numR
1149 \fi
1150 \ifodd\l@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 1161 \newcommand*{\affixpstart@numL}{%
\rightpstartnumR 1162 \ifsidepstartnum
  \leftpstartnumL 1163 \if@twocolumn
\rightpstartnumL 1164   \if@firstcolumn
  \ifpstartnumR 1165     \gdef\l@dld@ta{\llap{\leftpstartnumL}}}%
1166     \else
1167     \gdef\l@drd@ta{\rlap{\rightpstartnumL}}}%
1168     \fi
1169   \else
1170     \l@dtempcntb=\line@margin
1171     \ifnum\l@dtempcntb>\@ne
1172       \advance\l@dtempcntb \page@num
1173     \fi
1174     \ifodd\l@dtempcntb
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@dtempcntb=\line@marginR
1192   \ifnum\l@dtempcntb>\@ne
1193     \advance\l@dtempcntb \page@numR
1194   \fi
1195   \ifodd\l@dtempcntb
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, widow, and interline penalties, and puts a single penalty of the appropriate size back into the paragraph; these penalties get removed by the `\vsplit` operation. `\displaywidowpenalty` and `\brokenpenalty` are not restored, since we have no easy way to find out where we should insert them.

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

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

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

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

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

15 Footnotes

`\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.

```

1259 \renewcommand*{\Afootnote}[2] [] {%
1260 \newcommand{\content}{#2}%
1261 \ifnumberedpar@
1262 \ifledRcol%
1263 \footnoteoptions@[R]{#1}{true}%
1264 \xright@appenditem{\noexpand\vAfootnote{A}%
1265 {\l@d@nums}{\csexpandonce{tag}}{\csexpandonce{content}}}}\to\insert
1266 \footnoteoptions@[R]{#1}{false}%
1267 \global\advance\insert@countR \@ne%
1268 \else%
1269 \footnoteoptions@{#1}{true}%
1270 \xright@appenditem{\noexpand\vAfootnote{A}%
1271 {\l@d@nums}{\csexpandonce{tag}}{\csexpandonce{content}}}}\to\insert
1272 \global\advance\insert@count \@ne%
1273 \footnoteoptions@{#1}{false}%
1274 \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`.

```

1275 \else
1276   \vAfootnote{A}{\{0|0|0|0|0|0|0\}}{\{#1\}}%
1277   \fi\ignorespaces}

```

<code>\Bfootnote</code>	We need similar commands for the other footnote series.
<code>\Cfootnote</code>	
<code>\Dfootnote</code>	
<code>\Efootnote</code>	


```

1278 \renewcommand*{\Bfootnote}[2] [] {%
1279 \newcommand{\content}{#2}%
1280 \ifnumberedpar@
1281 \ifledRcol%
1282 \footnoteoptions@{R}{#1}{true}%
1283 \xright@appenditem{\noexpand\Bfootnote{B}%
1284 \{\l@d@nums\}\csexpandonce{tag}}{\csexpandonce{content}}}\to\inserts@listR
1285 \footnoteoptions@{R}{#1}{false}%
1286 \global\advance\insert@countR \@ne%
1287 \else%
1288 \footnoteoptions@{#1}{true}%
1289 \xright@appenditem{\noexpand\Bfootnote{B}%
1290 \{\l@d@nums\}\csexpandonce{tag}}{\csexpandonce{content}}}\to\inserts@list
1291 \global\advance\insert@count \@ne%
1292 \footnoteoptions@{#1}{false}%
1293 \fi
1294 \else
1295 \vBfootnote{B}{\{0|0|0|0|0|0|0\}{#1}}%
1296 \fi\ignorespaces}

1297 \renewcommand*{\Cfootnote}[2] [] {%
1298 \newcommand{\content}{#2}%
1299 \ifnumberedpar@
1300 \ifledRcol%
1301 \footnoteoptions@{R}{#1}{true}%
1302 \xright@appenditem{\noexpand\Cfootnote{C}%
1303 \{\l@d@nums\}\csexpandonce{tag}}{\csexpandonce{content}}}\to\inserts@listR
1304 \footnoteoptions@{R}{#1}{false}%
1305 \global\advance\insert@countR \@ne%
1306 \else%
1307 \footnoteoptions@{#1}{true}%
1308 \xright@appenditem{\noexpand\Cfootnote{C}%
1309 \{\l@d@nums\}\csexpandonce{tag}}{\csexpandonce{content}}}\to\inserts@list
1310 \global\advance\insert@count \@ne%
1311 \footnoteoptions@{#1}{false}%
1312 \fi
1313 \else
1314 \vCfootnote{C}{\{0|0|0|0|0|0|0\}{#1}}%
1315 \fi\ignorespaces}

1316 \renewcommand*{\Dfootnote}[2] [] {%
1317 \newcommand{\content}{#2}%
1318 \ifnumberedpar@
1319 \ifledRcol%
1320 \footnoteoptions@{R}{#1}{true}%
1321 \xright@appenditem{\noexpand\Dfootnote{D}%
1322 \{\l@d@nums\}\csexpandonce{tag}}{\csexpandonce{content}}}\to\inserts@listR
1323 \footnoteoptions@{R}{#1}{false}%
1324 \global\advance\insert@countR \@ne%
1325 \else%
1326 \footnoteoptions@{#1}{true}%

```

```

1327 \xright@appenditem{\noexpand\VDfootnote{D}%
1328 {{\l@d@nums}{\csexpandonce{@tag}}{\csexpandonce{content}}}}\to\insert
1329 \global\advance\insert@count \@ne%
1330 \footnoteoptions@{#1}{false}%
1331 \fi
1332 \else
1333 \VDfootnote{D}{{0|0|0|0|0|0|0|0}{#1}}%
1334 \fi\ignorespaces}

1335 \renewcommand*{\Efootnote}[2][]{%
1336 \newcommand{\content}{#2}%
1337 \ifnumberedpar@
1338 \ifledRcol%
1339 \footnoteoptions@{R}{#1}{true}%
1340 \xright@appenditem{\noexpand\VEfootnote{E}%
1341 {{\l@d@nums}{\csexpandonce{@tag}}{\csexpandonce{content}}}}\to\insert
1342 \footnoteoptions@{R}{#1}{false}%
1343 \global\advance\insert@countR \@ne%
1344 \else%
1345 \footnoteoptions@{#1}{true}%
1346 \xright@appenditem{\noexpand\VEfootnote{E}%
1347 {{\l@d@nums}{\csexpandonce{@tag}}{\csexpandonce{content}}}}\to\insert
1348 \global\advance\insert@count \@ne%
1349 \footnoteoptions@{#1}{false}%
1350 \fi
1351 \else
1352 \VEfootnote{E}{{0|0|0|0|0|0|0|0}{#1}}%
1353 \fi\ignorespaces}
1354

```

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

```

\mpBfootnote 1355 \renewcommand*{\mpAfootnote}[2][]{%
\mpCfootnote 1356 \newcommand{\content}{#2}%
\mpDfootnote 1357 \ifnumberedpar@
\mpEfootnote 1358 \ifledRcol
1359 \xright@appenditem{\noexpand\mpvAfootnote{A}%
1360 {{\l@d@nums}{\csexpandonce{@tag}}{\csexpandonce{content}}}}\to\insert
1361 \global\advance\insert@countR \@ne
1362 \else
1363 \xright@appenditem{\noexpand\mpvAfootnote{A}%
1364 {{\l@d@nums}{\csexpandonce{@tag}}{\csexpandonce{content}}}}\to\insert
1365 \global\advance\insert@count \@ne
1366 \fi
1367 \else
1368 \mpvAfootnote{A}{{0|0|0|0|0|0|0|0}{#1}}%
1369 \fi\ignorespaces}

1370 \renewcommand*{\mpBfootnote}[2][]{%
1371 \newcommand{\content}{#2}%
1372 \ifnumberedpar@
1373 \ifledRcol

```

```

1374 \xright@appenditem{\noexpand\mpvBfootnote{B}%
1375     {\l@d@nums}{\csexpandonce{@tag}}{\csexpandonce{content}}}\to\inserts@listR
1376 \global\advance\insert@countR \@ne
1377 \else
1378 \xright@appenditem{\noexpand\mpvBfootnote{B}%
1379     {\l@d@nums}{\csexpandonce{@tag}}{\csexpandonce{content}}}\to\inserts@list
1380 \global\advance\insert@count \@ne
1381 \fi
1382 \else
1383 \mpvBfootnote{B}{\l@0|0|0|0|0|0|0|0}{\l@1}}%
1384 \fi\ignorespaces}
1385 \renewcommand*{\mpCfootnote}[2][{}]{%
1386 \newcommand{\content}{#2}%
1387 \ifnumberedpar@
1388 \ifledRcol
1389 \xright@appenditem{\noexpand\mpvCfootnote{C}%
1390     {\l@d@nums}{\csexpandonce{@tag}}{\csexpandonce{content}}}\to\inserts@listR
1391 \global\advance\insert@countR \@ne
1392 \else
1393 \xright@appenditem{\noexpand\mpvCfootnote{C}%
1394     {\l@d@nums}{\csexpandonce{@tag}}{\csexpandonce{content}}}\to\inserts@list
1395 \global\advance\insert@count \@ne
1396 \fi
1397 \else
1398 \mpvCfootnote{C}{\l@0|0|0|0|0|0|0|0}{\l@1}}%
1399 \fi\ignorespaces}
1400 \renewcommand*{\mpDfootnote}[2][{}]{%
1401 \newcommand{\content}{#2}%
1402 \ifnumberedpar@
1403 \ifledRcol
1404 \xright@appenditem{\noexpand\mpvDfootnote{D}%
1405     {\l@d@nums}{\csexpandonce{@tag}}{\csexpandonce{content}}}\to\inserts@listR
1406 \global\advance\insert@countR \@ne
1407 \else
1408 \xright@appenditem{\noexpand\mpvDfootnote{D}%
1409     {\l@d@nums}{\csexpandonce{@tag}}{\csexpandonce{content}}}\to\inserts@list
1410 \global\advance\insert@count \@ne
1411 \fi
1412 \else
1413 \mpvDfootnote{D}{\l@0|0|0|0|0|0|0|0}{\l@1}}%
1414 \fi\ignorespaces}
1415 \renewcommand*{\mpEfootnote}[2][{}]{%
1416 \newcommand{\content}{#2}%
1417 \ifnumberedpar@
1418 \ifledRcol
1419 \xright@appenditem{\noexpand\mpvEfootnote{E}%
1420     {\l@d@nums}{\csexpandonce{@tag}}{\csexpandonce{content}}}\to\inserts@listR
1421 \global\advance\insert@countR \@ne
1422 \else

```

```

1423 \xright@appenditem{\noexpand\mpvEfootnote{E}%
1424 \{\l@d@nums\}{\csexpandonce{@tag}}{\csexpandonce{content}}}\to\inserts
1425 \global\advance\insert@count \@ne
1426 \fi
1427 \else
1428 \mpvEfootnote{E}{\{0|0|0|0|0|0|0\}{\#1}}%
1429 \fi\ignorespaces}

\l@dedendmini

1430 \renewcommand*{\l@dedendmini}{%
1431 \ifl@dpairing
1432 \ifledRcol
1433 \flush@notesR
1434 \else
1435 \flush@notes
1436 \fi
1437 \fi
1438 \ifvoid\mpAfootins\else\mpAfootgroup{A}\fi%
1439 \ifvoid\mpBfootins\else\mpBfootgroup{B}\fi%
1440 \ifvoid\mpCfootins\else\mpCfootgroup{C}\fi%
1441 \ifvoid\mpDfootins\else\mpDfootgroup{D}\fi%
1442 \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:

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

```

1443 \def\printlinesR#1|#2|#3|#4|#5|#6|#7|{\begingroup
1444 \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
1445 \ifl@d@pnum #1\fullstop\fi
1446 \ifledplinenum \linenumr@p{#2}\Rlineflag\else \symlinenum\fi
1447 \ifl@d@ssub \fullstop \sublinenumr@p{#3}\fi
1448 \ifl@d@dash \endashchar\fi
1449 \ifl@d@pnum #4\fullstop\fi
1450 \ifl@d@elin \linenumr@p{#5}\Rlineflag\fi
1451 \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumr@p{#6}\fi
1452 \endgroup}
1453
1454 \let\ledsavedprintlines\printlines
1455

```

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.

```
1456 \list@create{\labelref@listR}
1457
```

`\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.

```
1458 \renewcommand*{\edlabel}[1]{\@bsphack
1459   \ifledRcol
1460     \write\linenum@outR{\string\@lab}%
1461     \ifx\labelref@listR\empty
1462       \xdef\label@refs{\zz@@@}%
1463     \else
1464       \glp\labelref@listR\to\label@refs
1465     \fi
1466     \ifvmode
1467       \advancelabel@refs
1468     \fi
1469     \protected@write\@auxout{%
1470       {\string\l@dmake@labelsR\space\thepage|\label@refs|{#1}}%
1471   \else
1472     \write\linenum@out{\string\@lab}%
1473     \ifx\labelref@list\empty
1474       \xdef\label@refs{\zz@@@}%
1475     \else
1476       \glp\labelref@list\to\label@refs
1477     \fi
1478     \ifvmode
1479       \advancelabel@refs
1480     \fi
1481     \protected@write\@auxout{%
1482       {\string\l@dmake@labelsR\space\thepage|\label@refs|{#1}}%
1483   \fi
1484   \@esphack}
1485
```

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

```
1486 \def\l@dmake@labelsR#1|#2|#3|#4{%
1487   \expandafter\ifx\csname the@label#4\endcsname \relax\else
1488     \led@warn@DuplicateLabel{#4}%
1489   \fi
1490   \expandafter\gdef\csname the@label#4\endcsname{#1|#2\Rlineflag|#3}%
1491   \ignorespaces}
1492 \AtBeginDocument{%
```

```

1493 \def\l@dmake@labelsR#1|#2|#3|#4{%
1494 }
1495

```

\@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.

```

1496 \renewcommand*{\@lab}{%
1497   \ifledRcol
1498     \xright@appenditem{\linenumr@p{\line@numR}}{|%
1499     \ifsublines@ \sublinenumr@p{\subline@numR}\else 0\fi}%
1500     \to\labelref@listR
1501   \else
1502     \xright@appenditem{\linenumr@p{\line@num}}{|%
1503     \ifsublines@ \sublinenumr@p{\subline@num}\else 0\fi}%
1504     \to\labelref@list
1505   \fi}
1506

```

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 1507 \newcount\sidenote@marginR
1508 \renewcommand*{\sidenotemargin}[1]{%
1509   \l@dgetsidenote@margin{#1}%
1510   \ifnum\@l@dttempcntb>\m@ne
1511     \ifledRcol
1512       \global\sidenote@marginR=\@l@dttempcntb
1513     \else
1514       \global\sidenote@margin=\@l@dttempcntb
1515     \fi
1516   \fi}}
1517 \sidenotemargin{right}
1518 \global\sidenote@margin=\@ne
1519

```

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

```

\l@dcsnote 1520 \renewcommand*{\l@dlsnote}[1]{%
1521   \ifnumberedpar@
1522     \ifledRcol%
1523     \footnoteoptions@[R]{#1}{true}%
1524     \xright@appenditem{\noexpand\l@dlsnote{#1}}{%
1525       \to\inserts@listR

```

```

1526     \footnoteoptions@{R}{#1}{false}%
1527     \global\advance\insert@countR \@ne%
1528   \else%
1529     \footnoteoptions@{#1}{true}%
1530     \xright@appenditem{\noexpand\vl@dlsnote{#1}}%
1531         \to\inserts@list
1532     \global\advance\insert@count \@ne%
1533     \footnoteoptions@{#1}{false}%
1534   \fi
1535 \fi\ignorespaces}
1536 \renewcommand*{\l@drsnote}[1]{%
1537   \ifnumberedpar@
1538     \ifledRcol%
1539       \footnoteoptions@{R}{#1}{true}%
1540       \xright@appenditem{\noexpand\vl@drsnote{#1}}%
1541           \to\inserts@listR
1542       \footnoteoptions@{R}{#1}{false}%
1543       \global\advance\insert@countR \@ne%
1544     \else%
1545       \footnoteoptions@{#1}{true}%
1546       \xright@appenditem{\noexpand\vl@drsnote{#1}}%
1547           \to\inserts@list
1548       \global\advance\insert@count \@ne%
1549       \footnoteoptions@{#1}{false}%
1550     \fi
1551   \fi\ignorespaces}
1552 \renewcommand*{\l@dcsnote}[1]{%
1553   \ifnumberedpar@
1554     \ifledRcol%
1555       \footnoteoptions@{R}{#1}{true}%
1556       \xright@appenditem{\noexpand\vl@dcsnote{#1}}%
1557           \to\inserts@listR
1558       \footnoteoptions@{R}{#1}{false}%
1559       \global\advance\insert@countR \@ne%
1560     \else%
1561       \footnoteoptions@{#1}{true}%
1562       \xright@appenditem{\noexpand\vl@dcsnote{#1}}%
1563           \to\inserts@list
1564       \global\advance\insert@count \@ne%
1565       \footnoteoptions@{#1}{false}%
1566     \fi
1567   \fi\ignorespaces}
1568

```

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

```

1569 \newcommand*{\affixside@noter}{%
1570   \gdef\@templ@d{}%
1571   \ifx\@templ@d\l@dcsnote\text \else
1572     \if@twocolumn
1573       \if@firstcolumn

```

```

1574     \setl@dlp@rbox{\l@dcsnotetext}%
1575     \else
1576     \setl@drp@rbox{\l@dcsnotetext}%
1577     \fi
1578   \else
1579     \@l@dttempcntb=\sidenote@marginR
1580     \ifnum\@l@dttempcntb>\@ne
1581       \advance\@l@dttempcntb by\page@num
1582     \fi
1583     \ifodd\@l@dttempcntb
1584       \setl@drp@rbox{\l@dcsnotetext}%
1585     \else
1586       \setl@dlp@rbox{\l@dcsnotetext}%
1587     \fi
1588   \fi
1589 \fi}
1590

```

18 Familiar footnotes

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

```

1591 \renewcommand{\l@dbfnote}[1]{%
1592   \ifnumberedpar@
1593     \ifledRcol%
1594       \footnoteoptions@[R]{#1}{true}%
1595       \xright@appenditem{\noexpand\vl@dbfnote{#1}}{\@thefnmark}}%
1596       \to\inserts@listR
1597       \footnoteoptions@[R]{#1}{false}%
1598       \global\advance\insert@countR \@ne%
1599     \else%
1600       \footnoteoptions@{#1}{true}%
1601       \xright@appenditem{\noexpand\vl@dbfnote{#1}}{\@thefnmark}}%
1602       \to\inserts@list
1603       \global\advance\insert@count \@ne%
1604       \footnoteoptions@{#1}{false}%
1605     \fi
1606   \fi\ignorespaces}
1607

```

`\normalbfnoteX`

```

1608 \renewcommand{\normalbfnoteX}[2]{%
1609   \ifnumberedpar@
1610     \ifledRcol%
1611       \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}}{\@nameuse{thefootnote#1}}}%
1612       \to\inserts@listR
1613       \footnoteoptions@[R]{#1}{false}%
1614       \global\advance\insert@countR \@ne%

```



```

1615 \else%
1616 \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}{\@nameuse{thefootnote#1}}}%
1617 \to\inserts@list
1618 \global\advance\insert@count \One%
1619 \fi
1620 \fi\ignorespaces}
1621

```

19 Verse

Like in ledmac, the insertion of hangingsymbol is base on \ifinserthangingsymbol, and, for the right side, on \ifinserthangingsymbolR.

```

\inserthangingsymbolL
\inserthangingsymbolR 1622 \newif\ifinserthangingsymbolR
1623 \newcommand{\inserthangingsymbolL}{%
1624 \ifinserthangingsymbol%
1625 \ifistanzaL%
1626 \hfill\hangingsymbol%
1627 \fi%
1628 \fi}
1629 \newcommand{\inserthangingsymbolR}{%
1630 \ifinserthangingsymbolR%
1631 \ifistanzaR%
1632 \hfill\hangingsymbol%
1633 \fi%
1634 \fi}

```

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 1635 \newcommand{\correcthangingL}{%
1636 \ifl@dpadding\else%
1637 \ifistanzaL%
1638 \ifinserthangingsymbol%
1639 \hskip \@ifundefined{sza@0@}{0}{\expandafter%
1640 \noexpand\csname sza@0@\endcsname}\stanzaindentbase%
1641 \fi%
1642 \fi%
1643 \fi}
1644
1645 \newcommand{\correcthangingR}{%
1646 \ifl@dpadding\else%
1647 \ifistanzaR%
1648 \ifinserthangingsymbolR%
1649 \hskip \@ifundefined{sza@0@}{0}{\expandafter%

```

```

1650          \noexpand\csname sza@0@\endcsname}\stanzaindentbase%
1651 \fi%
1652 \fi%
1653 \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.

```

1654 \chardef\next=\catcode'\&
1655 \catcode'\&=\active
1656

```

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

```

1657 \newenvironment{astanza}{%
1658 \startstanzahook
1659 \catcode'\&\active
1660 \global\stanza@count\@ne
1661 \ifnum\usernamecount{sza@0@}=\z@
1662 \let\stanza@hang\relax
1663 \let\endlock\relax
1664 \else
1665 %% \interlinepenalty\@M % this screws things up, but I don't know why
1666 \rightskip\z@ plus 1fil\relax
1667 \fi
1668 \ifnum\usernamecount{szp@0@}=\z@
1669 \let\sza@penalty\relax
1670 \fi
1671 \def&{%
1672 \endlock\mbox{}}%
1673 \sza@penalty
1674 \global\advance\stanza@count\@ne
1675 \@astanza@line}%
1676 \def\&{%
1677 \endlock\mbox{}}
1678 \pend
1679 \endstanzaextra}%
1680 \pstart
1681 \@astanza@line
1682 }{}
1683

```

\@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.

```

1684 \newcommand*{\@astanza@line}{%
1685 \parindent=\csname sza@\number\stanza@count @\endcsname\stanzaindentbase
1686 \par
1687 \stanza@hang%\mbox{}}%
1688 \ignorespaces}
1689

```

Lastly reset the modified category codes.

```
1690 \catcode'\&=\next
1691
```

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 1692 \providecommand*\newnamebox}[1]{%
\unvnamebox 1693 \expandafter\newbox\csname #1\endcsname}
\namebox 1694 \providecommand*\setnamebox}[1]{%
1695 \expandafter\setbox\csname #1\endcsname}
1696 \providecommand*\unhnamebox}[1]{%
1697 \expandafter\unhbox\csname #1\endcsname}
1698 \providecommand*\unvnamebox}[1]{%
1699 \expandafter\unvbox\csname #1\endcsname}
1700 \providecommand*\namebox}[1]{%
1701 \csname #1\endcsname}
1702
```

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

```
\usenamecount 1703 \providecommand*\newnamecount}[1]{%
1704 \expandafter\newcount\csname #1\endcsname}
1705 \providecommand*\usenamecount}[1]{%
1706 \csname #1\endcsname}
1707
```

21 Counts and boxes for parallel texts

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

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

```
1708 \newcount\l@dc@maxchunks
1709 \newcommand*\maxchunks}[1]{\l@dc@maxchunks=#1}
1710 \maxchunks{10}
1711
```

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

`\l@dumpstartsR` 1712 `\newcount\l@dumpstartsR`
1713

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

`\l@pscR` 1714 `\newcount\l@dpscL`
1715 `\newcount\l@dpscR`
1716

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

1717 `\newcommand*{\l@dsetuprawboxes}{%`
1718 `\@l@dttempcntb=\l@dc@maxchunks`
1719 `\loop\ifnum\@l@dttempcntb>\z@`
1720 `\newnamebox{\l@dLcolrawbox\the\@l@dttempcntb}`
1721 `\newnamebox{\l@dRcolrawbox\the\@l@dttempcntb}`
1722 `\advance\@l@dttempcntb \m@ne`
1723 `\repeat}`
1724

`\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.

`\l@dzeromaxlinecounts`

1725 `\newcommand*{\l@dsetupmaxlinecounts}{%`
1726 `\@l@dttempcntb=\l@dc@maxchunks`
1727 `\loop\ifnum\@l@dttempcntb>\z@`
1728 `\newnamecount{\l@dmaxlinesinpar\the\@l@dttempcntb}`
1729 `\advance\@l@dttempcntb \m@ne`
1730 `\repeat}`
1731 `\newcommand*{\l@dzeromaxlinecounts}{%`
1732 `\begingroup`
1733 `\@l@dttempcntb=\l@dc@maxchunks`
1734 `\loop\ifnum\@l@dttempcntb>\z@`
1735 `\global\usenamecount{\l@dmaxlinesinpar\the\@l@dttempcntb}=\z@`
1736 `\advance\@l@dttempcntb \m@ne`
1737 `\repeat`
1738 `\endgroup}`
1739

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

1740 `\AtBeginDocument{%`
1741 `\l@dsetuprawboxes`
1742 `\l@dsetupmaxlinecounts`
1743 `\l@dzeromaxlinecounts`
1744 `\l@dumpstartsL=\z@`
1745 `\l@dumpstartsR=\z@`
1746 `\l@dpscL=\z@`

```

1747 \l@dpscR=\z@}
1748

```

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 1749 \newif\ifl@dusedbabel
\l@dusedbabeltrue 1750 \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 1751 \newif\ifl@dsamelang
1752 \l@dsamelangtrue

\l@dcchecklang I'm going to use \theledlanguageL and \theledlanguageR to hold the names of
the languages used for the left and right texts. This macro sets \ifl@dsamelang
TRUE if they are the same, otherwise it sets it FALSE.
1753 \newcommand*{\l@dcchecklang}{%
1754 \l@dsamelangfalse
1755 \edef\@tempa{\theledlanguageL}\edef\@tempb{\theledlanguageR}%
1756 \ifx\@tempa\@tempb
1757 \l@dsamelangtrue
1758 \fi}
1759

\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 \.
1760 \newcommand*{\l@dbbl@set@language}[1]{%
1761 \edef\language{#1}%
1762 \select@language{\language}%
1763 \if@filesw
1764 \protected@write\auxout{}\string\select@language{\language}%
1765 \addtocontents{toc}{\string\select@language{\language}%
1766 \addtocontents{lof}{\string\select@language{\language}%

```

```

1767     \addtocontents{lot}{\string\select@language{\language}}%
1768     \fi}
1769

```

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 1770 \providecommand{\selectlanguage}[1]{%
1771 \newcommand*{\l@duselanguage}[1]{%
1772 \gdef\theledlanguageL{}
1773 \gdef\theledlanguageR{}
1774

```

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

```

1775 \AtBeginDocument{%
1776   \@ifundefined{xpg@main@language}{%
1777     \@ifundefined{bbl@main@language}{%

```

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

```

1778     \l@dusedbabelfalse
1779     \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.

```

1780     \l@dusedbabeltrue
1781     \let\l@doldselectlanguage\selectlanguage
1782     \let\l@doldbbl@set@language\bbl@set@language
1783     \let\bbl@set@language\l@dbbl@set@language
1784     \renewcommand{\selectlanguage}[1]{%
1785       \l@doldselectlanguage{#1}%
1786       \ifledRcol \gdef\theledlanguageR{#1}%
1787       \else      \gdef\theledlanguageL{#1}%
1788       \fi}

```

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

```

1789     \renewcommand*{\l@duselanguage}[1]{%
1790       \l@doldselectlanguage{#1}}

```

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

```

1791     \gdef\theledlanguageL{\bbl@main@language}%
1792     \gdef\theledlanguageR{\bbl@main@language}%
1793     }%
1794   }

```

If on Polyglossia

```

1795 { \apptocmd{\xpg@set@language}{%
1796     \ifledRcol \gdef\theledlanguageR{#1}%
1797     \else      \gdef\theledlanguageL{#1}%
1798     \fi}%
1799     \let\l@duselanguage\xpg@set@language
1800     \gdef\theledlanguageL{\xpg@main@language}%
1801     \gdef\theledlanguageR{\xpg@main@language}%
1802 % \end{macrocode}
1803 % That's it.
1804 % \begin{macrocode}
1805 }}

```

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.

```

1806 \newcommand*{\Columns}{%
1807     \setcounter{pstartL}{\value{pstartLold}}
1808     \setcounter{pstartR}{\value{pstartRold}}
1809     \ifnum\l@dnumpstartsL=\l@dnumpstartsR\else
1810         \led@err@BadLeftRightPstarts{\the\l@dnumpstartsL}{\the\l@dnumpstartsR}%
1811     \fi

```

Start a group and zero counters, etc.

```

1812 \begingroup
1813     \l@dzeropenalties
1814     \endgraf\global\num@lines=\prevgraf
1815         \global\num@linesR=\prevgraf
1816     \global\par@line=\z@
1817     \global\par@lineR=\z@
1818     \global\l@dpscL=\z@
1819     \global\l@dpscR=\z@

```

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

```

1820     \check@pstarts
1821     \loop\if@pstarts
1822         \global\pstartnumtrue
1823         \global\pstartnumRtrue

```

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

```

1824     \global\advance\l@dpscL \@ne
1825     \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

```

1826      \checkraw@text
1827      \l@dchecklang
1828 {      \loop\ifaraw@text
        Grab the next pair of left and right text lines and output them, swapping languages
        if they differ
1829          \ifl@dsamelang
1830          \do@lineL
1831          \do@lineR
1832          \else
1833          \l@duselanguage{\theledlanguageL}%
1834          \do@lineL
1835          \l@duselanguage{\theledlanguageR}%
1836          \do@lineR
1837          \fi
1838          \hb@xt@ \hsize{%
1839          \hfill \unhbox\l@dleftbox
1840          \hfill \columnseparator \hfill
1841          \unhbox\l@drightbox
1842          }%
1843          \checkraw@text
1844      \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.

```

1845      \@writelinesinparL
1846      \@writelinesinparR
1847      \check@pstarts
1848          \ifbypstart@
1849          \write\linenum@out{\string\@set[1]}
1850      \fi
1851      \ifbypstart@R
1852      \write\linenum@outR{\string\@set[1]}
1853      \fi
1854      \addtocounter{pstartL}{1}
1855      \addtocounter{pstartR}{1}
1856      \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.

```

1857      \flush@notes
1858      \flush@notesR
1859      \endgroup
1860      \global\l@dpscL=\z@
1861      \global\l@dpscR=\z@
1862      \global\l@dnumpstartsL=\z@
1863      \global\l@dnumpstartsR=\z@
1864      \ignorespaces
1865      \global\instanzaLfalse

```



```

1866     \global\instanzaRfalse}
1867

\columnseparator The separator between line pairs in parallel columns is in the form of a vertical
\columnrulewidth 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).

1868 \newcommand*{\columnseparator}{%
1869   \smash{\rule[-0.2\baselineskip]{\columnrulewidth}{1.05\baselineskip}}}%
1870 \newdimen\columnrulewidth
1871   \columnrulewidth=\z@
1872

\if@pstarts \check@pstarts returns \@pstartstrue if there are any unprocessed chunks.
\@pstartstrue 1873 \newif\if@pstarts
\@pstartsfalse 1874 \newcommand*{\check@pstarts}{%
\check@pstarts 1875   \@pstartsfalse
1876   \ifnum\l@dnumpstartsL>\l@dpscL
1877     \@pstartstrue
1878   \else
1879     \ifnum\l@dnumpstartsR>\l@dpscR
1880       \@pstartstrue
1881     \fi
1882   \fi
1883 }
1884

\ifaraw@text \checkraw@text checks whether the current Left or Right box is void or not. If
\araw@texttrue one or other is not void it sets \araw@texttrue, otherwise both are void and it
\araw@textfalse sets \araw@textfalse.
\checkraw@text 1885 \newif\ifaraw@text
1886   \araw@textfalse
1887 \newcommand*{\checkraw@text}{%
1888   \araw@textfalse
1889   \ifvbox\namebox{\l@dLcolrawbox\the\l@dpscL}
1890     \araw@texttrue
1891   \else
1892     \ifvbox\namebox{\l@dRcolrawbox\the\l@dpscR}
1893       \araw@texttrue
1894     \fi
1895   \fi
1896 }
1897

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

1898 \newcommand*{\@writelinesinparL}{%
1899   \edef\next{%
1900     \write\linenum@out{\string\@pend[\the\@donereallinesL]}}}%

```

```

1901 \next
1902 \global\@donereallinesL \z@}
1903 \newcommand*{\@writelinesinparR}{%
1904 \edef\next{%
1905 \write\linenum@outR{\string\@pendR[\the\@donereallinesR]}}}%
1906 \next
1907 \global\@donereallinesR \z@}
1908

```

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` 1909 `\newcount\numpagelinesL`
1910 `\newcount\numpagelinesR`
1911 `\newcount\l@dminpagelines`
1912

`\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.

```

1913 \newcommand*{\Pages}{%
1914 \setcounter{pstartL}{\value{pstartLold}}
1915 \setcounter{pstartR}{\value{pstartRold}}
1916 \typeout{}
1917 \typeout{***** PAGES *****}
1918 \ifnum\l@dnumstartsL=\l@dnumstartsR\else
1919 \led@err@BadLeftRightPstarts{\the\l@dnumstartsL}{\the\l@dnumstartsR}%
1920 \fi

```

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

```

1921 \cleartol@devenpage
1922 \begingroup
1923 \l@dzeropenalties
1924 \endgraf\global\num@lines=\prevgraf
1925 \global\num@linesR=\prevgraf
1926 \global\par@line=\z@
1927 \global\par@lineR=\z@
1928 \global\l@dpscL=\z@
1929 \global\l@dpscR=\z@
1930 \writtenlinesLfalse
1931 \writtenlinesRfalse

```

Check if there are chunks to be processed.

```

1932 \check@pstarts
1933 \loop\if@pstarts

```

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

```
1934      \global\advance\l@dpscL \@ne
1935      \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.

```
1936      \getlinesfromparlistL
1937      \getlinesfromparlistR
1938      \l@dcalc@maxoftwo{\@cs@linesinparL}{\@cs@linesinparR}%
1939      {\usernamecount{\l@dmaxlinesinpar\the\l@dpscL}}%
1940      \check@pstarts
1941      \repeat
```

Zero the counts again, ready for the next bit.

```
1942      \global\l@dpscL=\z@
1943      \global\l@dpscR=\z@
```

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

```
1944      \getlinesfrompagelistL
1945      \getlinesfrompagelistR
1946      \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
1947      {\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.

```
1948      \check@pstarts
1949      \if@pstarts
```

Increment the chunk counts to get the first pair.

```
1950      \global\advance\l@dpscL \@ne
1951      \global\advance\l@dpscR \@ne
```

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

```
1952      \global\@donereallinesL=\z@
1953      \global\@donetotallinesL=\z@
1954      \global\@donereallinesR=\z@
1955      \global\@donetotallinesR=\z@
```

Start a loop over the boxes (chunks).

```
1956      \checkraw@text
1957 %      \begingroup
1958 {      \loop\ifaraw@text
```

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

```
1959      \checkpageL
1960      \l@duselanguage{\theledlanguageL}%
1961 %%%      \begingroup
1962 {      \loop\ifl@dsamepage
1963
```

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

```

1964      \do@lineL
1965      \advance\numpagelinesL \@ne
1966      \ifshiftedverses
1967      \ifdim\ht\l@dleftbox>0pt\hb@xt@ \hsize{\ledstrutL\unhbox\l@dleftbox}\fi%
1968      \else
1969      \hb@xt@ \hsize{\ledstrutL\unhbox\l@dleftbox}%
1970      \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.

```

1971
1972      \get@nextboxL
1973      \checkpageL
1974      \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.

```

1975      \ifl@dpagfull
1976      \@writelinesonpageL{\the\numpagelinesL}%
1977      \else
1978      \@writelinesonpageL{1000}%
1979      \fi

```

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

```

1980      \numpagelinesL \z@
1981      \clearl@dleftpage }%

```

Now do the same for the right text.

```

1982      \checkpageR
1983      \l@duselanguage{\theledlanguageR}%
1984 {
1985      \loop\ifl@dsamepage
1986      \do@lineR
1987      \advance\numpagelinesR \@ne
1988      \ifshiftedverses
1989      \ifdim\ht\l@drighbox>0pt\hb@xt@ \hsize{\ledstrutR\unhbox\l@drighbox}\fi%
1990      \else
1991      \hb@xt@ \hsize{\ledstrutR\unhbox\l@drighbox}%
1992      \fi
1993      \get@nextboxR
1994      \checkpageR
1995      \repeat
1996      \ifl@dpagfull
1997      \@writelinesonpageR{\the\numpagelinesR}%
1998      \else
1999      \@writelinesonpageR{1000}%
2000      \fi
      \numpagelinesR=\z@

```

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

2001 \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.

2002 \checkraw@text

2003 \ifaraw@text

2004 \getlinesfrompagelistL

2005 \getlinesfrompagelistR

2006 \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%

2007 {\l@dminpagelines}%

2008 \fi

2009 \repeat}

We have now output the text from all the chunks.

2010 \fi

Make sure that there are no inserts hanging around.

2011 \flush@notes

2012 \flush@notesR

2013 \endgroup

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

2014 \global\l@dpscL=\z@

2015 \global\l@dpscR=\z@

2016 \global\l@dnumpstartsL=\z@

2017 \global\l@dnumpstartsR=\z@

2018 \global\instanzaLfalse

2019 \global\instanzaRfalse

2020 \ignorespaces}

2021

\ledstrutL Struts inserted into leftand right text lines.

\ledstrutR 2022 \newcommand*{\ledstrutL}{\strut}

2023 \newcommand*{\ledstrutR}{\strut}

2024

\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.

2025 \providecommand{\cleartoevenpage}[1][\@empty]{%

2026 \clearpage

2027 \ifodd\c@page\hbox{ }#1\clearpage\fi}

2028 \newcommand*{\cleartol@devenpage}{%

2029 \ifdim\pagetotal<\topskip% on an empty page

2030 \else

```

2031 \clearpage
2032 \fi
2033 \ifodd\c@page\hbox{}\clearpage\fi}
2034 \newcommand*\clearl@leftpage}{%
2035 \clearpage
2036 \ifodd\c@page\else
2037 \led@err@LeftOnRightPage
2038 \hbox{}%
2039 \cleardoublepage
2040 \fi}
2041 \newcommand*\clearl@rightpage}{%
2042 \clearpage
2043 \ifodd\c@page
2044 \led@err@RightOnLeftPage
2045 \hbox{}%
2046 \cleartoevenpage
2047 \fi}
2048

```

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

```

\cs@linesinparR 2049 \newcommand*\getlinesfromparlistL}{%
2050 \ifx\linesinpar@listL\empty
2051 \gdef\cs@linesinparL{0}%
2052 \else
2053 \gl@p\linesinpar@listL\to\cs@linesinparL
2054 \fi}
2055 \newcommand*\getlinesfromparlistR}{%
2056 \ifx\linesinpar@listR\empty
2057 \gdef\cs@linesinparR{0}%
2058 \else
2059 \gl@p\linesinpar@listR\to\cs@linesinparR
2060 \fi}
2061

```

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

```

\cs@linesonpageR 2062 \newcommand*\getlinesfrompagelistL}{%
2063 \ifx\linesonpage@listL\empty
2064 \gdef\cs@linesonpageL{1000}%
2065 \else
2066 \gl@p\linesonpage@listL\to\cs@linesonpageL
2067 \fi}
2068 \newcommand*\getlinesfrompagelistR}{%
2069 \ifx\linesonpage@listR\empty
2070 \gdef\cs@linesonpageR{1000}%
2071 \else

```

```

2072   \gl@p\linesonpage@listR\to\cs@linesonpageR
2073   \fi}
2074

```

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

```

2075 \newcommand*{\@writelinesonpageL}[1]{%
2076   \edef\next{\write\linenum@out{\string\@lopL{#1}}}%
2077   \next}
2078 \newcommand*{\@writelinesonpageR}[1]{%
2079   \edef\next{\write\linenum@outR{\string\@lopR{#1}}}%
2080   \next}
2081

```

\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>.

```

2082 \newcommand*{\l@dcalc@maxoftwo}[3]{%
2083   \ifnum #2>#1\relax
2084     #3=#2\relax
2085   \else
2086     #3=#1\relax
2087   \fi}
2088 \newcommand*{\l@dcalc@minoftwo}[3]{%
2089   \ifnum #2<#1\relax
2090     #3=#2\relax
2091   \else
2092     #3=#1\relax
2093   \fi}
2094

```

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

```

\checkpageL 2095 \newif\ifl@dsamepage
\checkpageR 2096   \l@dsamepagetrue
2097   \newif\ifl@dpagetrue
2098   \newcommand*{\checkpageL}{%
2099     \l@dpagetruefull
2100     \l@dsamepagetrue
2101     \checkgoal
2102     \ifdim\pagetotal<\ledthegoal
2103       \ifnum\numpagelinesL<\l@dminpagelines
2104         \else

```

```

2105     \l@dsamepagefalse
2106     \l@dpagfullfalse
2107   \fi
2108 \else
2109   \l@dsamepagefalse
2110   \l@dpagfulltrue
2111 \fi}
2112 \newcommand*{\checkpageR}{%
2113   \l@dpagfulltrue
2114   \l@dsamepagetrue
2115   \check@goal
2116   \ifdim\pagetotal<\ledthegoal
2117     \ifnum\numpagelinesR<\l@dminpagelines
2118       \else
2119         \l@dsamepagefalse
2120         \l@dpagfullfalse
2121       \fi
2122     \else
2123       \l@dsamepagefalse
2124       \l@dpagfulltrue
2125     \fi}
2126

```

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

```

2127 \newdimen\ledthegoal
2128 \ifshiftedverses
2129   \newcommand*{\goalfraction}{0.95}
2130 \else
2131   \newcommand*{\goalfraction}{0.9}
2132 \fi
2133
2134 \newcommand*{\check@goal}{%
2135   \ledthegoal=\goalfraction\pagegoal}
2136

```

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

```

\ifwrittenlinesL 2137 \newif\ifwrittenlinesL
2138 \newif\ifwrittenlinesR
2139

```

`\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.

```

2140 \newcommand*{\get@nextboxL}{%
2141   \ifvbox\namebox{\l@dLcolrawbox\the\l@dpscl}% box is not empty
2142   \else%

```

The current box is not empty; do nothing.

```


```

box is empty

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

```
2143 \ifnum\usernamecount{l@dmaxlinesinpar\the\l@dpscL}>\@donetotallinesL
2144 \else
```

Sufficient lines have been output.

```
2145 \ifwritelinesL
2146 \else
```

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

```
2147 \@writelinesinparL
2148 \writelinesLtrue
2149 \fi
2150 \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.

```
2151 \writelinesLfalse
2152 \ifbypstart@
2153 \ifnum\value{pstartL}<\value{pstartLold}
2154 \else
2155 \global\line@num=0
2156 \fi
2157 \fi
2158 \addtocounter{pstartL}{1}
2159 \global\pstartnumtrue
2160 \l@dcalc@maxoftwo{\the\usernamecount{l@dmaxlinesinpar\the\l@dpscL}}%
2161 \the\@donetotallinesL}%
2162 {\usernamecount{l@dmaxlinesinpar\the\l@dpscL}}}%
2163 \global\@donetotallinesL \z@
2164 \global\advance\l@dpscL \@ne
2165 \fi
2166 \fi
2167 \fi}
```

```
2168 \newcommand*{\get@nextboxR}{%
2169 \ifvbox\namebox{l@dRcolrawbox\the\l@dpscR}% box is not empty
2170 \else% box is empty
2171 \ifnum\usernamecount{l@dmaxlinesinpar\the\l@dpscR}>\@donetotallinesR
2172 \else
2173 \ifwritelinesR
2174 \else
2175 \@writelinesinparR
2176 \writelinesRtrue
2177 \fi
2178 \ifnum\l@dnumstartsR>\l@dpscR
2179 \writelinesRfalse
2180 \ifbypstart@R
2181 \ifnum\value{pstartR}<\value{pstartRold}
2182 \else
2183 \global\line@numR=0
```

```

2184         \fi
2185     \fi
2186     \addtocounter{pstartR}{1}
2187     \global\pstartnumRtrue
2188     \l@dcalc@maxoftwo{\the\usernamecount{l@dmxlinesinpar\the\l@dpscR}}%
2189         {\the\@donetotallinesR}%
2190         {\usernamecount{l@dmxlinesinpar\the\l@dpscR}}%
2191     \global\@donetotallinesR \z@
2192     \global\advance\l@dpscR \@ne
2193 \fi
2194 \fi
2195 \fi}
2196

```

25 The End

i/codei

A Examples

This section presents some sample documents.

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

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

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

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

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

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

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

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

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

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

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

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

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

Figure 1: Output from `villon.tex`.

1 De ecclesia S. Stephani Novimagensi

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

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

// One additional line to show synchronization. //

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

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

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

1 St. Stephen's Church in Nijmegen

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

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

// One additional line to show synchronization. //

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

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

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

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

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

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

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

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

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

 12 instruat, Aoniam Marte movente lyram?

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

Figure 4: First left page output from `djdpoe.ms.tex`.

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

²Muses

³Ceres was the Roman goddess of the harvest.

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

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

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

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

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

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

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

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

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

⁷Muses

⁸Ceres was the Roman goddess of the harvest.

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

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

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

A.1 Parallel column example

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

```

2197 (*villon)
2198 %% villon.tex Example parallel columns
2199 \documentclass{article}
2200 \addtolength{\textheight}{-10\baselineskip}
2201 \usepackage{ledmac,ledpar}
2202 %% Use r instead of R to flag right text line numbers
2203 \renewcommand{\Rlineflag}{r}
2204 %% Use the flag in the notes
2205 \let\oldBfootfmt\Bfootfmt
2206 \renewcommand{\Bfootfmt}[3]{%
2207   \let\printlines\printlinesR
2208   \oldBfootfmt{#1}{#2}{#3}}
2209 \begin{document}
2210
2211 I thought that limericks were peculiarly English, but this appears not
2212 to be the case. As with most limericks this one is by Anonymous.
2213
2214 \vspace*{\baselineskip}
2215
2216 \begin{pairs}
2217 %% no indentation
2218 \setstanzaindents{0,0,0,0,0,0,0,0,0}
2219 %% no number flag
2220 \renewcommand{\Rlineflag}{}
2221 %% draw a rule and widen the columns
2222 \setlength{\columnrulewidth}{0.4pt}
2223 \setlength{\Lcolwidth}{0.46\textwidth}
2224 \setlength{\Rcolwidth}{\Lcolwidth}
2225
2226 \begin{Leftside}
2227 %% set left text line numbering sequence
2228 \firstlinenum{2}
2229 \linenumincrement{2}
2230 \linenummargin{left}
2231 \beginnumbering
2232 \stanza
2233 Il y avait un jeune homme de Dijon, &
2234 Qui n'avait que peu de religion. &
2235 Il dit: 'Quant \'{a} moi, &
2236 Je d'\{e\}teste tous les trois, &
2237 Le P\{e\}re, et le Fils, et le Pigeon.' \&
2238 \endnumbering
2239 \end{Leftside}

```

```

2240
2241 \begin{Rightside}
2242 %% different right text line numbering sequence
2243 \firstlinenum{1}
2244 \linenumincrement{2}
2245 \linenummargin{right}
2246 \beginnumbering
2247 \stanza
2248 There was a young man of Dijon, &
2249 Who had only a little religion, &
2250 He said: 'As for me, &
2251 I detest all the three, &
2252 The Father, the Son, and the Pigeon.' \&
2253 \endnumbering
2254 \end{Rightside}
2255
2256 \Columns
2257 \end{pairs}
2258
2259 \vspace*{\baselineskip}
2260
2261 The following is verse \textsc{lxiii} of Fran\c{c}ois Villon's
2262 \textit{Le Testament} (The Testament), composed in 1461.
2263
2264 %% Allow for hanging indentation for long lines
2265 \setstanzaindents{1,0,0,0,0,0,0,0}
2266 %% Columns wider than the default
2267 \setlength{\Lcolwidth}{0.46\textwidth}
2268 \setlength{\Rcolwidth}{\Lcolwidth}
2269 \vspace*{\baselineskip}
2270
2271 \begin{pairs}
2272 \begin{Leftside}
2273 \firstlinenum{2}
2274 \linenumincrement{2}
2275 \linenummargin{left}
2276 \beginnumbering
2277 \stanza
2278 Dieu mercy et Tacque Thibault, &
2279 Qui tant d'eaue froid m'a fait boire, &
2280 Mis en bas lieu, non pas en hault, &
2281 Mengier d'angoisse maints \edtext{poire}{\lemma{poire d'angoisse}}%
2282 \Afootnote{This has a triple meaning: literally it is the fruit of the
2283 choke pear,
2284 figuratively it means 'bitter fruit', and it also refers to a torture
2285 instrument.}}, &
2286 Enferr'\{e} \ldots Quant j'en ay memoire, &
2287 Je Prie pour luy \edtext{\textit{et reliqua}}{\Afootnote{and so on}}, &
2288 Que Dieu luy doint, et voire, voire! &
2289 Ce que je pense \ldots \textit{et cetera}. \&

```

```

2290 \endnumbering
2291 \end{Leftside}
2292
2293 \begin{Rightside}
2294 \firstlinenum{2}
2295 \linenumincrement{2}
2296 \linenummargin{right}
2297 \beginnumbering
2298 \stanza
2299 Thanks to God --- and to \edtext{Tacque Thibaud}{%
2300   \Bfootnote{A favourite of Jean, Duc de Berry and loathed for his exactions
2301     and debauchery. Villon uses his name as an insulting nickname for
2302     Thibaud d'Auxigny, the Bishop of Orl\{'e}ans.}} &
2303 Who made me drink so much \edtext{cold water}{%
2304   \Bfootnote{Can either refer to the normal prison diet of bread and
2305     water or to a common medieval torture which involved forced drinking
2306     of cold water.}}, &
2307 Put me underground instead of higher up &
2308 And made me eat such bitter fruit, &
2309 In chains \ldots When I think of this, &
2310 I pray for him---\textit{et reliqua;} &
2311 May God grant him (yes, by God) &
2312 What I think \ldots \textit{et cetera}. \&
2313 \endnumbering
2314 \end{Rightside}
2315
2316 \Columns
2317 \end{pairs}
2318
2319 \vspace*{\baselineskip}
2320
2321   The translation and notes are by Anthony Bonner,
2322 \textit{The Complete Works of Fran\c{c}ois Villon}, published by
2323 Bantam Books in 1960.
2324
2325 \end{document}
2326
2327 </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.

```

2328 <*djd17nov>
2329 %%% This is djd17nov.tex, a sample critical text edition
2330 %%% written in LaTeX2e with the ledmac and ledpar packages.
2331 %%% (c) 2003--2004 by Dr. Dirk-Jan Dekker,

```

```

2332 %% Radboud University, Nijmegen (The Netherlands)
2333 %% (PRW) Modified slightly by PRW to fit the ledpar manual
2334
2335 \documentclass[10pt, letterpaper, twoside]{article}
2336 \usepackage[latin,english]{babel}
2337 \usepackage{makeidx}
2338 \usepackage{ledmac,ledpar}
2339 \lineation{section}
2340 \linenummargin{inner}
2341 \sidenotemargin{outer}
2342
2343 \makeindex
2344
2345 \renewcommand{\notenumfont}{\footnotesize}
2346 \newcommand{\notetextfont}{\footnotesize}
2347
2348 %\let\Afootnoterule=\relax
2349 \let\Bfootnoterule=\relax
2350 \let\Cfootnoterule=\relax
2351
2352 \addtolength{\skip\Afootins}{1.5mm}
2353 %\addtolength{\skip\Bfootins}{1.5mm}
2354 %\addtolength{\skip\Cfootins}{1.5mm}
2355
2356 \makeatletter
2357
2358 \renewcommand*{\para@vfootnote}[2]{%
2359   \insert\csname #1footins\endcsname
2360   \bgroup
2361     \notefontsetup
2362     \interlinepenalty=\interfootnotelinepenalty
2363     \floatingpenalty=\@MM
2364     \splittopskip=\ht\strutbox \splitmaxdepth=\dp\strutbox
2365     \leftskip=\z@skip \rightskip=\z@skip
2366     \l@dparsfootspec #2\ledplinenumtrue% new from here
2367     \ifnum\@nameuse{previous@#1@number}=\l@dparsestartline\relax
2368       \ledplinenumfalse
2369     \fi
2370     \ifnum\previous@page=\l@dparsestartpage\relax
2371     \else \ledplinenumtrue \fi
2372     \ifnum\l@dparsestartline=\l@dparseendline\relax
2373     \else \ledplinenumtrue \fi
2374     \expandafter\xdef\csname previous@#1@number\endcsname{\l@dparsestartline}%
2375     \xdef\previous@page{\l@dparsestartpage}% to here
2376     \setbox0=\vbox{\hsize=\maxdimen
2377       \noindent\csname #1footfmt\endcsname#2}%
2378     \setbox0=\hbox{\unvbox0}%
2379     \dp0=0pt
2380     \ht0=\csname #1footfudgefactor\endcsname\wd0
2381     \box0

```

```

2382     \penalty0
2383     \egroup
2384 }
2385
2386 \newcommand*{\previous@A@number}{-1}
2387 \newcommand*{\previous@B@number}{-1}
2388 \newcommand*{\previous@C@number}{-1}
2389 \newcommand*{\previous@page}{-1}
2390
2391 \newcommand{\abb}[1]{#1%
2392     \let\rbracket\nobrak\relax}
2393 \newcommand{\nobrak}{\textnormal{}}
2394 \newcommand{\morenoexpands}{%
2395     \let\abb=0%
2396 }
2397
2398 \newcommand{\Aparafootfmt}[3]{%
2399     \ledsetnormalparstuff
2400     \scriptsize
2401     \notenumfont\printlines#1\enspace
2402 %   \lemmafont#1/#2\enskip
2403     \notetextfont
2404     #3\penalty-10\hskip 1em plus 4em minus.4em\relax}
2405
2406 \newcommand{\Bparafootfmt}[3]{%
2407     \ledsetnormalparstuff
2408     \scriptsize
2409     \notenumfont\printlines#1/%
2410     \ifledplinenum
2411     \enspace
2412     \else
2413     {\hskip 0em plus 0em minus .3em}%
2414     \fi
2415     \select@lemmafont#1/#2\rbracket\enskip
2416     \notetextfont
2417     #3\penalty-10\hskip 1em plus 4em minus.4em\relax }
2418
2419 \newcommand{\Cparafootfmt}[3]{%
2420     \ledsetnormalparstuff
2421     \scriptsize
2422     \notenumfont\printlines#1\enspace
2423 %   \lemmafont#1/#2\enskip
2424     \notetextfont
2425     #3\penalty-10\hskip 1em plus 4em minus.4em\relax}
2426
2427 \makeatother
2428
2429 \footparagraph{A}
2430 \footparagraph{B}
2431 \footparagraph{C}

```



```

2432
2433 \let\Afootfmt=\Aparafootfmt
2434 \let\Bfootfmt=\Bparafootfmt
2435 \let\Cfootfmt=\Cparafootfmt
2436
2437 \renewcommand*{\Rlineflag}{}
2438
2439 \emergencystretch40pt
2440
2441 \author{Guillelmus de Berchen}
2442 \title{Chronicon Geldriae}
2443 \date{}
2444 \hyphenation{archi-epi-sco-po Huns-dis-brug li-be-ra No-vi-ma-gen-si}
2445 \begin{document}
2446 \begin{pages}
2447 \begin{Leftside}
2448 \beginnumbering\pstart
2449 \selectlanguage{latin}
2450 \section{De ecclesia S. Stephani Novimagensi}
2451
2452 \noindent\setline{1}
2453 Nobilis itaque comes Otto\protect\edindex{Otto II of Guelders}
2454 imperio et dominio Novimagensi sibi, ut praefertur, impignoratis
2455 et commissis
2456 \edtext{proinde}{\Bfootnote{primum D}} praeesse cupiens, anno
2457 \textsc{liiii} superius descripto, mense
2458 Iu\edtext{}{\Afootnote{p.\ 227~R}}nio, una cum iudice, scabinis ceterisque
2459 civibus civitatis Novimagensis, pro ipsius et inhabitantium in ea
2460 necessitate,\edtext{}{\Afootnote{p.\ 97~N}} commodo et utilitate,
2461 ut \edtext{ecclesia eius}{\Bfootnote{ecclesia D: eius eius H}} parochialis
2462 \edtext{\abb{extra civitatem}}{\Bfootnote{\textit{om.}~H}} sita
2463 destrueretur et \edtext{infra}{\Bfootnote{intra D}} muros
2464 \edtext{transfer}\edtext{}{\Afootnote{p.\ 129~D}}retur}%
2465 {\Bfootnote{transferreretur NH}}
2466 ac de novo construeretur,
2467 \edtext{a reverendo patre domino
2468 Conrado\protect\edindex{Conrad of Hochstaden} de
2469 \edtext{Hofsteden}{\Bfootnote{Hoffstede D: Hoffsteden H}}, archiepiscopo
2470 \edtext{Coloniensi}{\Bfootnote{Colononiensi H}}, licentiam}%
2471 {\Cfootnote{William is confusing two charters that are five years
2472 apart. Permission from St.\ Apostles' Church in Cologne had been
2473 obtained as early as 1249. Cf.\
2474 Sloet\protect\index{Sloet van de Beele, L.A.J.W.},
2475 \textit{Oorkondenboek} nr.\ 707 (14 November 1249):
2476 ‘‘\ldots}nos devotionis tue precibus annuentes, ut ipsam ecclesiam
2477 faciens demoliri transferas in locum alium competentem, tibi
2478 auctoritate presentium indulgemus\ldots’’}}, et a venerabilibus
2479 \edtext{dominis}{\Bfootnote{viris H}} decano et capitulo sanctorum
2480 Apostolorum\protect\edindex{St. Apostles' (Cologne)}
2481 \edtext{Coloniensi}{\Bfootnote{Coloniae H}}, ipsius ecclesiae ab

```

```

2482 antiquo veris et pacificis patronis, consensum, citra tamen
2483 praeiudicium, damnum aut gravamen \edtext{iurium}{\Bfootnote{virium D}}
2484 et bonorum eorundem, impetravit.
2485 \pend
2486
2487 \pstart
2488 \edtext{Et exinde \edtext{liberum}{\Bfootnote{librum H}}}
2489 locum eiusdem civitatis
2490 \edtext{qui}{\Bfootnote{quae D}} dicitur
2491 \edtext{Hundisburg}{\Bfootnote{Hundisburch D: Hundisbrug HMN:
2492 Hunsdisbrug R}}\protect\edindex{Hundisburg},
2493 de praelibati Wilhelmi\protect\edindex{William II of Holland} Romanorum
2494 \edtext{regis}{\Bfootnote{imperatoris D}}, ipsius fundi
2495 do\edtext{}{\Afootnote{f.\ 72v~M}}mini, consensu, ad aedificandum
2496 \edtext{\abb{et consecrandum}}{\Bfootnote{\textit{om.}\ H}}
2497 ecclesi\edtext{}{\Afootnote{p.\ 228~R}}am et coemeterium,
2498 \edtext{eisdem}{\Bfootnote{eiusdem D}} decano et capitulo de expresso
2499 eiusdem civitatis assensu libera contradiderunt voluntate, obligantes
2500 se ipsi \edtext{comes}{\Bfootnote{comites D}} et civitas
2501 \edtext{\abb{dictis}}{\Bfootnote{\textit{om.}\ H}} decano et capitulo,
2502 quod in recompensationem illius areae infra castrum et portam, quae
2503 fuit dos ecclesiae, in qua plebanus habitare solebat---quae
2504 \edtext{tunc}{\Bfootnote{nunc H}} per novum fossatum civitatis est
2505 destructa---aliam aream competentem et ecclesiae novae,
2506 \edtext{ut praefertur, aedificandae}{%
2507 \lemma{\abb{ut\ldots aedificandae}}{\Bfootnote{\textit{om.}\ H}} satis
2508 \edtext{contiguam}{\Bfootnote{contiguum M}}, ipsi plebano darent et
2509 assignarent.}{\Cfootnote{Cf.\ Sloet, \textit{Oorkondenboek} nr.\ 762
2510 (June 1254)}} Et desuper
2511 \edtext{\abb{apud}}{\Bfootnote{\textit{om.}\ H}} dictam ecclesiam
2512 sanctorum Apostolorum \edtext{est}{\Bfootnote{et H}}
2513 \edtext{littera}{\Bfootnote{litteram H}} sigillis ipsorum
2514 Ottonis\edtext{}{\Afootnote{p.\ 130~D}} comitis et civitatis
2515 \edtext{Novimagensis}{\Bfootnote{Novimagii D}}
2516 \edtext{sigillata}{\Bfootnote{sigillis communita H}}.
2517 \pend
2518
2519 \pstart
2520 // One additional line to show synchronization. //
2521 \pend
2522 \endnumbering
2523 \end{Leftside}
2524
2525 \begin{Rightside}
2526 \sidenotemargin{right}\selectlanguage{english}
2527 \beginnumbering
2528 \pstart
2529 \addtocounter{section}{-1}%
2530 \leavevmode\section{St.\ Stephen's Church in Nijmegen}
2531

```

2532 \noindent\setline{1}%
 2533 After the noble count Otto had taken in pledge the power over
 2534 Nijmegen,\footnote{In 1247 William II\protect\index{William II of Holland}
 2535 (1227--1256) count of Holland needed money to fight his way to
 2536 Aachen\protect\index{Aachen} to be crowned King of the Holy Roman
 2537 Empire. He gave the town of Nijmegen in pledge to Otto
 2538 II\protect\index{Otto II of Guelders} (1229--1271) count of Guelders.}
 2539 like I have written above, he wanted to protect the town. So in June
 2540 1254\ledsidenote{1254} he and the judge, the sheriffs and other
 2541 citizens of Nijmegen obtained permission to demolish the parish
 2542 church that lay outside the town walls,\footnote{Since the early
 2543 seventh century old St.\ Stephen's church had been located close
 2544 to the castle, at today's
 2545 Kelfkensbos\protect\index{Kelfkensbos (Nijmegen)}} square.
 2546 Traces of the church and the presbytery were found during excavations
 2547 in 1998--1999.} to move it inside the walls and to rebuild it new.
 2548 This operation was necessary and useful both for Otto himself and
 2549 for the inhabitants of the town. The reverend father Conrad of
 2550 Hochstaden, archbishop of
 2551 Cologne,\footnote{Conrad of Hochstaden ({\textdagger} 1261) was
 2552 archbishop of Cologne in 1238--1261. Nijmegen belonged to the
 2553 archdiocese of Cologne until 1559.} gave his permission. So did the
 2554 reverend dean and canons of the chapter of St.\
 2555 Apostles'\protect\index{St. Apostles' (Cologne)} in Cologne, who had
 2556 long\footnote{They probably became the patrons when the chapter was
 2557 established in the early eleventh century. About the church and the
 2558 chapter, see Gottfried Stracke\protect\index{Stracke, G.},
 2559 \textit{K}\{o\}ln:\ St.\ Aposteln}, Stadtspuren -- Denkm\{a\}ler in
 2560 K\{o\}ln, vol.\ 19, K\{o\}ln: J.\,P.\ Bachem, 1992.} been the true
 2561 and benevolent patrons of the church---but they did not allow Otto
 2562 to do anything without their knowledge, nor to infringe their rights,
 2563 nor to damage their property.
 2564 \pend
 2565
 2566 \pstart
 2567 And so the count and the town voluntarily gave an open space in town
 2568 called Hundisburg, which was owned by the aforementioned king William,
 2569 to the dean and chapter of St.\ Apostles' in order to build and
 2570 consecrate a church and graveyard. King William approved and the
 2571 town of Nijmegen explicitly expressed its assent. A new ditch was dug
 2572 on property of the church near the castle and the
 2573 harbour,\footnote{Nowadays, the exact location of the medieval
 2574 ditch---and of two Roman ones---can be seen in the pavement of
 2575 Kelfkensbos\protect\index{Kelfkensbos (Nijmegen)}} square.} causing
 2576 the demolition of the presbytery. In compensation, the count and
 2577 citizens committed themselves to giving the parish priest another
 2578 suitable space close enough to the new church that was about to be
 2579 built. A letter about these transactions, with the seals of count
 2580 Otto and the town of Nijmegen, is kept at St.\ Apostles'
 2581 church.\footnote{The original letter is lost. A 15th century

```

2582 transcription of it is kept at the Historisches Archiv der
2583 Stadt K\{"o}ln (HASTK).}
2584 \pend
2585
2586 \pstart
2587 // One additional line to show synchronization. //
2588 \pend
2589 \endnumbering
2590 \end{Rightside}
2591 \Pages
2592 \end{pages}
2593
2594 %%%%%%%%%%%%%%%
2595 \printindex
2596 \end{document}
2597 %%%%%%%%%%%%%%%
2598
2599 </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.

```

2600 (*djdpoems)
2601 %% djdpoems.tex  example parallel verses on facing pages
2602 \documentclass{article}
2603 \usepackage{ledmac,ledpar}
2604 \addtolength{\textheight}{-15\baselineskip}
2605
2606 \maxchunks{24} % default value = 10
2607 \setstanzaindents{6,0,1,0,1}
2608
2609 \newcommand{\longdash}{-----}
2610
2611 \footparagraph{A} % for left pages
2612 \footparagraph{B} % for right pages
2613 \firstlinenum{1}
2614 \linenumincrement{1}
2615
2616 \let\oldBfootfmt\Bfootfmt
2617 \renewcommand{\Bfootfmt}[3]{%
2618   \let\printlines\printlinesR
2619   \oldBfootfmt{#1}{#2}{#3}}

```

```

2620
2621 \begin{document}
2622
2623 \newcommand{\interstanza}{\pstart\centering\longdash\skipnumbering\pend}
2624
2625 \begin{pages}
2626 \begin{Leftside}
2627 \def\endstanzaextra{\interstanza}
2628 \beginnumbering
2629
2630 \stanza
2631 Arma gravi numero violentaque bella parabam &
2632 edere, materi\={a} conveniente modis. &
2633 Par erat inferior versus---risisse Cupido &
2634 dicitur atque unum surripuisse pedem. \&
2635
2636 \stanza
2637 ‘‘Quis tibi, saeve puer, dedit hoc in carmina iuris? &
2638 Pieridum vates, non tua turba \edtext{sumus}{\Afootnote{note lost}}. &
2639 Quid si praeripiat flavae V\ue}nus arma Minervae, &
2640 ventilet accensas flava Minerva faces? \&
2641
2642 \stanza
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 cuspidе Phoebum &
2647 instruat, Aoniam Marte movente lyram? \&
2648 \endnumbering
2649 \end{Leftside}
2650
2651 \begin{Rightside}
2652 \def\endstanzaextra{\interstanza}
2653 \beginnumbering
2654 \firstlinenum{1}
2655 \linenumincrement{1}
2656 \setstanzaindents{6,0,1,0,1,0}
2657
2658 \stanza
2659 I was preparing to sing of weapons and violent wars, &
2660 in heavy numbers, with the subject matter suited to the verse measure. &
2661 The even lines were as long as the odd ones, but Cupid laughed, &
2662 they said, and he stole away one foot.\footnote{I.e., the even lines,
2663 which were hexameters (with six feet) became pentameters
2664 (with five feet).} \&
2665
2666 \stanza
2667 ‘‘O cruel boy, who gave you the right over poetry? &
2668 We poets belong to the Pierides,\footnote{Muses} we are not your folk. &
2669 \edlabel{beginparadox}What if Venus should seize away the arms of

```

```

2670 Minerva with the golden hair, &
2671 if Minerva with the golden hair should fan alight the kindled torch
2672 of love? \&
2673
2674 \stanza
2675 Who would approve of Ceres\footnote{Ceres was the Roman goddess of
2676 the harvest.} reigning on the woodland ridges, &
2677 and of land tilled under the law of the Maid with the
2678 quiver\footnote{By '\textit{Virgo}' ('Virgin') Ovid means Diana, the
2679 Roman goddess of the hunt.}? &
2680 Who would provide Phoebus with his beautiful hair with a sharp-pointed
2681 spear, &
2682 while Mars stirs the \edtext{Aonian}{\Bfootnote{Mount Parnassus,
2683 where the Muses live, is located in Aonia.}}
2684 lyre?\edlabel{endparadox}\footnote{Lines
2685 \xlineref{beginparadox}--\xlineref{endparadox} show some paradoxical
2686 situations that would occur if the gods didn't stay with their own
2687 business.} \&
2688 \endnumbering
2689 \end{Rightside}
2690
2691 \Pages
2692 \end{pages}
2693
2694 \begin{pages}
2695 \begin{Leftside}
2696 \def\endstanzaextra{\interstanza}
2697 \beginnumbering
2698
2699 \begin{astanza}
2700 Arma gravi numero violentaque bella parabam &
2701 edere, materi\={a} conveniente modis. &
2702 Par erat inferior versus---risisse Cupido &
2703 dicitur atque unum surripuisse pedem. \&
2704 \end{astanza}
2705
2706 \begin{astanza}
2707 'Quis tibi, saeve puer, dedit hoc in carmina iuris? &
2708 Pieridum vates, non tua turba \edtext{sumus}{\Afootnote{note lost}}. &
2709 Quid si praeripiat flavae V\ufe}nus arma Minervae, &
2710 ventilet accensas flava Minerva faces? \&
2711 \end{astanza}
2712
2713 \begin{astanza}
2714 Quis probet in silvis Cererem regnare iugosis, &
2715 lege pharetratae Virginis arva coli? &
2716 Crinibus insignem quis \edtext{acuta}{\Afootnote{acut\={a} (abl.\ abs.)}}
2717 cuspide Phoebum &
2718 instruat, Aoniam Marte movente lyram? \&
2719 \end{astanza}

```

```

2720
2721 \endnumbering
2722 \end{Leftside}
2723
2724 \begin{Rightside}
2725 \def\endstanzaextra{\interstanza}
2726 \beginnumbering
2727 \firstlinenum{1}
2728 \linenumincrement{1}
2729 \setstanzaindents{6,0,1,0,1,0}
2730
2731 \begin{astanza}
2732 I was preparing to sing of weapons and violent wars, &
2733 in heavy numbers, with the subject matter suited to the verse measure. &
2734 The even lines were as long as the odd ones, but Cupid laughed, &
2735 they said, and he stole away one foot.\footnote{I.e., the even lines,
2736 which were hexameters (with six feet) became pentameters
2737 (with five feet).} \&
2738 \end{astanza}
2739
2740 \begin{astanza}
2741 ‘‘O cruel boy, who gave you the right over poetry? &
2742 We poets belong to the Pierides,\footnote{Muses} we are not your folk. &
2743 \edlabel{beginparadox}What if Venus should seize away the arms of
2744 Minerva with the golden hair, &
2745 if Minerva with the golden hair should fan alight the kindled torch
2746 of love? \&
2747 \end{astanza}
2748
2749 \begin{astanza}
2750 Who would approve of Ceres\footnote{Ceres was the Roman goddess of the
2751 harvest.} reigning on the woodland ridges, &
2752 and of land tilled under the law of the Maid with the
2753 quiver\footnote{By ‘\textit{Virgo}’ (‘Virgin’) Ovid means Diana,
2754 the Roman goddess of the hunt.}? &
2755 Who would provide Phoebus with his beautiful hair with a sharp-pointed
2756 spear, &
2757 while Mars stirs the \edtext{Aonian}{\Bfootnote{Mount Parnassus, where
2758 the Muses live, is located in Aonia.}}
2759 lyre?\edlabel{endparadox}\footnote{Lines
2760 \xlineref{beginparadox}--\xlineref{endparadox} show some paradoxical
2761 situations that would occur if the gods didn’t stay with their
2762 own business.} \&
2763 \end{astanza}
2764
2765 \endnumbering
2766 \end{Rightside}
2767
2768 \Pages
2769 \end{pages}

```

```
2770
2771 \end{document}
2772
2773 \end{djdpoe
```

References

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

Index

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

Symbols	
<code>\&</code> . . . 1654, 1655, 1659, 1676, 1690, 2237, 2252, 2289, 2312, 2634, 2640, 2647, 2664, 2672, 2687, 2703, 2710, 2718, 2737, 2746, 2762	<code>\@donetotallinesR</code> 870, 935, 938, 1955, 2171, 2189, 2191
<code>\@M</code> 1665	<code>\@insertR</code> 1239–1241, 1254–1256
<code>\@MM</code> 2363	<code>\@l</code> 260, 575
<code>\@adv</code> 337, 606, 607	<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, 1510, 1512, 1514, 1579–1581, 1583, 1718–1722, 1726–1729, 1733–1736
<code>\@arabic</code> 194, 195, 764, 767	<code>\@l@reg</code> 309
<code>\@astanza@line</code> 1675, 1681, <u>1684</u>	<code>\@l@regR</code> 260
<code>\@auxout</code> 1469, 1481, 1764	<code>\@lab</code> 529, 1460, 1472, <u>1496</u>
<code>\@chapter</code> 716	<code>\@lock</code> 886, 973
<code>\@cs@linesinparL</code> 1938, <u>2049</u>	<code>\@lockR</code> 58, 282, 284, 286, 299, 444, 460, 461, 463, 464, 492, 493, 495, 921, 954, 998, 1000,
<code>\@cs@linesinparR</code> 1938, <u>2049</u>	
<code>\@cs@linesonpageL</code> . . 1946, 2006, <u>2062</u>	
<code>\@cs@linesonpageR</code> . . 1946, 2006, <u>2062</u>	
<code>\@donereallinesL</code> 870, 899, 1900, 1902, 1952	
<code>\@donereallinesR</code> 870, 934, 1905, 1907, 1954	
<code>\@donetotallinesL</code> 870, 900, 903, 1953, 2143, 2161, 2163	

- 1001, 1003, 1084, 1101, 1103, 1105
 \@lopL 553, 2076
 \@lopR 553, 2079
 \@nameuse 1611, 1616, 2367
 \@nobreakfalse 773, 803
 \@nobreaktrue 771, 775, 801, 805
 \@oldnobreak 771, 773, 801, 803, 840, 856
 \@pend 544, 1900
 \@pendR 544, 1905
 \@pstartsfalse 1873
 \@pstartstrue 1873
 \@ref 516, 579, 583
 \@ref@reg 542
 \@schapter 716
 \@set 369, 613, 614, 1849, 1852
 \@tag 643, 659
 \@temp 1755
 \@templ@d 1570, 1571
 \@writelinesinparL .. 1845, 1898, 2147
 \@writelinesinparR .. 1846, 1898, 2175
 \@writelinesonpageL . 1976, 1978, 2075
 \@writelinesonpageR . 1996, 1998, 2075
 \@xloop 1252
- _ 2458, 2460, 2464, 2472, 2473, 2475,
 2495–2497, 2501, 2507, 2509,
 2511, 2514, 2530, 2543, 2554,
 2559, 2560, 2569, 2580, 2645, 2716
- ### A
- \abb 2391,
 2395, 2462, 2496, 2501, 2507, 2511
 \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 892
 \add@inserts@nextR 1227
 \add@insertsR 927, 1227
 \add@penaltiesL 898, 1248
 \add@penaltiesR 933, 1248
 \addtocontents 1765–1767
 \addtocounter 842,
 858, 1854, 1855, 2158, 2186, 2529
 \addtolength ... 2200, 2352–2354, 2604
 \advancelabel@refs 1467, 1479
 \advanceline 605, 636
 \affixline@num 890
 \affixline@numR 925, 1054
 \affixpstart@numL 889, 1160
 \affixpstart@numR 924, 1160
 \affixside@note 893
 \affixside@noteR 928, 1569
 \Afootfmt 2433
 \Afootins 2352
 \Afootnote 1259, 2282,
 2287, 2458, 2460, 2464, 2495,
 2497, 2514, 2638, 2645, 2708, 2716
 \Afootnoterule 2348
 \Aparafootfmt 2398, 2433
 \apptocmd 1795
 \araw@textfalse 1885
 \araw@texttrue 1885
 astanza (environment) 7, 1657
 \AtBeginDocument ... 1492, 1740, 1775
 \author 2441
- ### B
- \ballast@count 982, 987
 \bbl@main@language 1791, 1792
 \bbl@set@language 1782, 1783
 \beginnumbering .. 6, 34, 722, 740,
 778, 2231, 2246, 2276, 2297,
 2448, 2527, 2628, 2653, 2697, 2726
 \beginnumberingR ... 47, 102, 740, 808
 \Bfootfmt 2205, 2206, 2434, 2616, 2617
 \Bfootins 2353
 \Bfootnote 1278, 2300, 2304,
 2456, 2461–2463, 2465, 2469,
 2470, 2479, 2481, 2483, 2488,
 2490, 2491, 2494, 2496, 2498,
 2500, 2501, 2504, 2507, 2508,
 2511–2513, 2515, 2516, 2682, 2757
 \Bfootnoterule 2349
 \bfseries 764, 767
 \box 2381
 \Bparafootfmt 2406, 2434
 \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 ... 575, 2027, 2033, 2036, 2043
 \c@pstartL 764
 \c@pstartR 767
 \c@sublinenumincrementR .. 166, 1122
 \centering 2623
 \Cfootfmt 2435
 \Cfootins 2354
 \Cfootnote 1278, 2471, 2509
 \Cfootnoterule 2350
 \ch@ck@l@ckR 1054
 \ch@cksub@l@ckR 1054
 \ch@cksub@lockR 1123
 \chapter 702, 703, 711
 \chapterinpages 695, 703, 713
 \chardef 1654
 \check@goal 2101, 2115, 2127
 \check@pstarts
 1820, 1847, 1873, 1932, 1940, 1948
 \checkpageL 1959, 1973, 2095
 \checkpageR 1982, 1993, 2095
 \checkraw@text
 1826, 1843, 1885, 1956, 2002
 \cleardoublepage 2039
 \clearl@dleftpage 1981, 2025
 \clearl@drightpage 2001, 2025
 \cleartoevenpage 2025
 \cleartol@devenpage 1921, 2025
 \closeout 566, 570
 \columnrulewidth 4, 1868, 2222
 \Columns 3, 1806, 2256, 2316
 \columnseparator 4, 1840, 1868
 \content .. 1260, 1279, 1298, 1317,
 1336, 1356, 1371, 1386, 1401, 1416
 \correcthangingL 895, 1635
 \correcthangingR 930, 1635
 \countLline 865, 876
 \countRline 865, 911
 \Cparafootfmt 2419, 2435
 \critext 641
 \csexpandonce 1265, 1271, 1284, 1290,
 1303, 1309, 1322, 1328, 1341,
 1347, 1360, 1364, 1375, 1379,
 1390, 1394, 1405, 1409, 1420, 1424

D

\date 2443
 \DeclareOption 7
 \def@tempb 140
 Dekker, Dirk-Jan 84, 90
 \Dfootnote 1278
 \dimen 592, 593, 597–599, 603
 \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, 1830, 1834, 1964
 \do@lineLhook 880, 907
 \do@lineR 910, 1831, 1836, 1985
 \do@lineRhook 907, 915
 \do@lockoff 479
 \do@lockoffL 503
 \do@lockoffR 479
 \do@lockon 444
 \do@lockonL 476
 \do@lockonR 444
 \documentclass 2199, 2335, 2602
 \dp 2364, 2379
 \dummy@ref 525

E

\edfont@info 678, 681, 687, 690
 \edindex .. 2453, 2468, 2480, 2492, 2493
 \edlabel .. 1458, 2669, 2684, 2743, 2759
 \edtext 657, 2281,
 2287, 2299, 2303, 2456, 2458,
 2460–2464, 2467, 2469, 2470,
 2479, 2481, 2483, 2488, 2490,
 2491, 2494–2498, 2500, 2501,
 2504, 2506, 2508, 2511–2516,
 2638, 2645, 2682, 2708, 2716, 2757
 \Efootnote 1278
 \emergencystretch 2439
 \empty .. 76, 79, 244, 252, 651, 667,
 676, 685, 787, 817, 1044, 1126,
 1134, 1229–1231, 1242, 1253,
 1461, 1473, 2050, 2056, 2063, 2069
 \end@lemmas 651, 652, 667, 668
 \endashchar 1448
 \endgraf 836, 852, 1814, 1924
 \endline@num 532, 538
 \endlock 625, 1663, 1672, 1677

- `\endnumbering` 6, 37, 69, 106,
 741, 2238, 2253, 2290, 2313,
 2522, 2589, 2648, 2688, 2721, 2765
`\endnumberingR` 50, 69, 91, 101, 114, 741
`\endpage@num` 531, 538
`\endstanzaextra`
 1679, 2627, 2652, 2696, 2725
`\endsub` 592
`\endsubline@num` 533, 539
`\enskip` 2402, 2415, 2423
`\enspace` 2401, 2411, 2422
 environments:
 `astanza` 7, 1657
 `Leftside` 5, 720
 `pages` 4, 695
 `pairs` 3, 695
 `Rightside` 5, 738
`\extensionchars` . . 45, 64, 97, 111, 119
- F**
- `\f@x@l@cksR` 1054
`\first@linenum@out@Rfalse` . . 561, 567
`\first@linenum@out@Rtrue` 561
`\firstlinenum` 5, 171, 2228,
 2243, 2273, 2294, 2613, 2654, 2727
`\firstsublinenum` 5, 171
`\fix@page` 305, 312
`\flag@end` 576, 656, 672
`\flag@start` 576, 648, 664
`\floatingpenalty` 2363
`\flush@notes` 1435, 1857, 2011
`\flush@notesR` . . 1251, 1433, 1858, 2012
`\footnote`
 . 2534, 2542, 2551, 2556, 2573,
 2581, 2662, 2668, 2675, 2678,
 2684, 2735, 2742, 2750, 2753, 2759
`\footnoteoptions@` 1263,
 1266, 1269, 1273, 1282, 1285,
 1288, 1292, 1301, 1304, 1307,
 1311, 1320, 1323, 1326, 1330,
 1339, 1342, 1345, 1349, 1523,
 1526, 1529, 1533, 1539, 1542,
 1545, 1549, 1555, 1558, 1561,
 1565, 1594, 1597, 1600, 1604, 1613
`\footnotesize` 2345, 2346
`\footparagraph` . 2429–2431, 2611, 2612
`\fullstop` . . 207, 1445, 1447, 1449, 1451
- G**
- `\get@linelistfile` 240
- `\get@nextboxL` 1972, 2140
`\get@nextboxR` 1992, 2140
`\getline@numL` 885, 961
`\getline@numR` 920, 942
`\getlinesfrompagelistL`
 1944, 2004, 2062
`\getlinesfrompagelistR`
 1945, 2005, 2062
`\getlinesfromparlistL` . . . 1936, 2049
`\getlinesfromparlistR` . . . 1937, 2049
`\gl@p` 247, 248,
 255, 256, 652, 668, 680, 689,
 1047, 1048, 1235, 1239, 1254,
 1464, 1476, 2053, 2059, 2066, 2072
`\goalfraction` 4, 2127
- H**
- `\hangingsymbol` 9, 1626, 1632
`\hb@xt@` . . . 888, 895, 902, 923, 930,
 937, 1838, 1967, 1969, 1988, 1990
`\hsize` 797, 827,
 1838, 1967, 1969, 1988, 1990, 2376
`\hyphenation` 2444
- I**
- `\if@filesw` 1763
`\if@firstcolumn` 1140, 1164, 1185, 1573
`\if@nobreak` 770, 800
`\if@pstarts` . . . 1821, 1873, 1933, 1949
`\ifaraw@text` . . . 1828, 1885, 1958, 2003
`\ifautopar` 796, 826
`\ifbypage@` 328
`\ifbypage@R` 122, 318, 1026
`\ifbypstart@` 546, 1848, 2152
`\ifbypstart@R` . . . 122, 550, 1851, 2180
`\ifdim` 593, 597, 599,
 603, 1967, 1988, 2029, 2102, 2116
`\iffirst@linenum@out@R` 561, 565
`\ifinserthangingsymbol` . . . 1624, 1638
`\ifinserthangingsymbolR`
 1622, 1630, 1648
`\ifinstanzaL` 718, 718, 1625, 1637
`\ifinstanzaR` 718, 719, 1631, 1647
`\ifl@d@dash` 1448
`\ifl@d@elin` 1450, 1451
`\ifl@d@esl` 1451
`\ifl@d@pnum` 1445, 1449
`\ifl@d@ssub` 1447
`\ifl@dpagefull` 1975, 1995, 2095
`\ifl@dpageing` 9, 1636, 1646

- \ifl@dpairing 9, 73, 1431
 - \ifl@dsamelang 1751, 1829
 - \ifl@dsamepage 1962, 1984, 2095
 - \ifl@dskipnumber 1117
 - \ifl@dusedbabel 1749
 - \ifledplinenum 1446, 2410
 - \ifledRcol 9, 154, 176, 180, 184, 188, 227, 242, 306, 315, 339, 353, 370, 387, 399, 407, 428, 473, 500, 509, 518, 577, 587, 594, 600, 606, 613, 621, 626, 630, 635, 645, 661, 675, 1262, 1281, 1300, 1319, 1338, 1358, 1373, 1388, 1403, 1418, 1432, 1459, 1497, 1511, 1522, 1538, 1554, 1593, 1610, 1786, 1796
 - \ifnoteschanged@ 83
 - \ifnumberedpar@ ... 780, 810, 832, 848, 1261, 1280, 1299, 1318, 1337, 1357, 1372, 1387, 1402, 1417, 1521, 1537, 1553, 1592, 1609
 - \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, 1583, 2027, 2033, 2036, 2043
 - \ifpst@rtedL 30, 784
 - \ifpst@rtedR 30, 814
 - \ifpstartnum 1205, 1210
 - \ifpstartnumR 1160
 - \ifshiftedverses . 5, 1966, 1987, 2128
 - \ifsidepstartnum 796, 826, 1162, 1183
 - \ifsublines@ 205, 294, 338, 371, 378, 409, 418, 430, 437, 449, 483, 537, 539, 949, 968, 1033, 1120, 1499, 1503
 - \ifvbox 877, 912, 1889, 1892, 2141, 2169
 - \ifvmode 1466, 1478
 - \ifvoid 1438–1442
 - \ifwrittenlinesL 2137, 2145
 - \ifwrittenlinesR 2138, 2173
 - \initnumbering@reg 43
 - \insert 2359
 - \insert@count 515, 583, 646, 662, 1272, 1291, 1310, 1329, 1348, 1365, 1380, 1395, 1410, 1425, 1532, 1548, 1564, 1603, 1618
 - \insert@countR 516, 579, 645, 661, 1267, 1286, 1305, 1324, 1343, 1361, 1376, 1391, 1406, 1421, 1527, 1543, 1559, 1598, 1614
 - \inserthangingsymbolfalse 886
 - \inserthangingsymbolL 895, 1622
 - \inserthangingsymbolR 930, 1622
 - \inserthangingsymbolRfalse 921
 - \inserthangingsymbolRtrue 921
 - \inserthangingsymboltrue 886
 - \insertlines@listR 76, 215, 229, 521, 1231, 1235
 - \inserts@list 786, 1271, 1290, 1309, 1328, 1347, 1364, 1379, 1394, 1409, 1424, 1531, 1547, 1563, 1602, 1617
 - \inserts@listR 816, 1226, 1229, 1239, 1253, 1254, 1265, 1284, 1303, 1322, 1341, 1360, 1375, 1390, 1405, 1420, 1525, 1541, 1557, 1596, 1612
 - \istanzaLfalse 1865, 2018
 - \istanzaLtrue 729
 - \istanzaRfalse 1866, 2019
 - \istanzaRtrue 751
 - \interfootnotelinepenalty 2362
 - \interlinepenalty 1665, 2362
 - \interstanza 2623, 2627, 2652, 2696, 2725
- L**
- \l@d@nums 678, 681, 687, 690, 1265, 1271, 1284, 1290, 1303, 1309, 1322, 1328, 1341, 1347, 1360, 1364, 1375, 1379, 1390, 1394, 1405, 1409, 1420, 1424
 - \l@d@set 386, 621, 622
 - \l@dbbl@set@language 1760, 1783
 - \l@dbfnote 1591
 - \l@dc@maxchunks 792, 794, 822, 824, 1708, 1718, 1726, 1733
 - \l@dcalc@maxoftwo 1938, 2082, 2160, 2188
 - \l@dcalc@minoftwo .. 1946, 2006, 2082
 - \l@dcalcnun 1054
 - \l@dchecklang 1753, 1827
 - \l@dchset@num 261, 264, 386
 - \l@dcsnote 1520
 - \l@dcsnotetext 1571, 1574, 1576, 1584, 1586
 - \l@dedendmini 1430
 - \l@demptyd@ta 881, 916

- \l@dend@stuff 46, 65, 98, 112, 120
- \l@dgetline@margin 152
- \l@dgetsidenote@margin 1509
- \l@dld@ta 891, 926,
1141, 1153, 1165, 1177, 1186, 1198
- \l@dleftbox
.. 862, 887, 902, 1839, 1967, 1969
- \l@dlinenumR 197
- \l@dlsn@te 894, 929
- \l@dlsnote 1520
- \l@dmake@labels 1482
- \l@dmake@labelsR 1470, 1486
- \l@dminpagelines
.... 1909, 1947, 2007, 2103, 2117
- \l@dnumpstartsL . 39, 791, 792, 794,
796, 1712, 1744, 1809, 1810,
1862, 1876, 1918, 1919, 2016, 2150
- \l@dnumpstartsR . 52, 821, 822, 824,
826, 1712, 1745, 1809, 1810,
1863, 1879, 1918, 1919, 2017, 2178
- \l@doldbbl@set@language 1782
- \l@doldselectlanguage 1781, 1785, 1790
- \l@dpagfullfalse 2095
- \l@dpagfulltrue 2095
- \l@dpagingfalse 11, 697, 710
- \l@dpagingtrue 705
- \l@dpairingfalse 9, 699, 709
- \l@dpairingtrue 696, 704
- \l@dparsedendline 2372
- \l@dparsedstartline . 2367, 2372, 2374
- \l@dparsedstartpage 2370, 2375
- \l@dparsefootspec 2366
- \l@dpscl 877, 882, 1714, 1746, 1818,
1824, 1860, 1876, 1889, 1928,
1934, 1939, 1942, 1950, 2014,
2141, 2143, 2150, 2160, 2162, 2164
- \l@dpscr 912, 917, 1715, 1747,
1819, 1825, 1861, 1879, 1892,
1929, 1935, 1943, 1951, 2015,
2169, 2171, 2178, 2188, 2190, 2192
- \l@drd@ta 895, 930,
1143, 1151, 1167, 1175, 1188, 1196
- \l@drightbox
.. 862, 922, 937, 1841, 1988, 1990
- \l@drsn@te 896, 931
- \l@drsnote 1520
- \l@dsamelangfalse 1751, 1754
- \l@dsamelangtrue 1751, 1757
- \l@dsamepagefalse 2095
- \l@dsamepagetrue 2095
- \l@dsetupmaxlinecounts . . . 1725, 1742
- \l@dsetuprawboxes 1717, 1741
- \l@dskipnumberfalse 1118
- \l@dskipnumbertrue 1014
- \l@dunhbox@line 895, 930
- \l@dusedbabelfalse 1749, 1778
- \l@dusedbabeltrue 1749, 1780
- \l@duselanguage
.... 1770, 1833, 1835, 1960, 1983
- \l@dzeromaxlinecounts . . . 1725, 1743
- \l@dzeropenalties 835, 851, 1813, 1923
- \l@pscL 1714
- \l@pscr 1714
- \label@refs
1462, 1464, 1470, 1474, 1476, 1482
- \labelref@list 1473, 1476, 1504
- \labelref@listR 1456, 1461, 1464, 1500
- \language name . . 1761, 1762, 1764–1767
- \last@page@num 326, 332
- \last@page@numR 312
- \lastbox 884, 919
- \lastskip 592, 598
- \Lcolwidth 3, 4, 13, 706, 797,
888, 902, 2223, 2224, 2267, 2268
- \ldots 2286,
2289, 2309, 2312, 2476, 2478, 2507
- \led@err@BadLeftRightPstarts . . .
..... 21, 1810, 1919
- \led@err@LeftOnRightPage . . . 24, 2037
- \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, 2044
- \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 611
- \led@warn@BadSetlinenum 619

- \led@warn@DuplicateLabel 1488
 - \ledllfill 895, 930
 - \ledmac@error 19, 22, 25, 27
 - \ledplinenumfalse 2368
 - \ledplinenumtrue ... 2366, 2371, 2373
 - \ledRcolfalse 12, 721, 753
 - \ledRcoltrue 739
 - \ledrlfill 895, 930
 - \ledsavedprintlines 7, 1443
 - \ledsetnormalparstuff 2399, 2407, 2420
 - \ledsidenote 2540
 - \ledstrutL 1967, 1969, 2022
 - \ledstrutR 1988, 1990, 2022
 - \ledthegoal 2102, 2116, 2127
 - \leftlinenumR 197, 1141, 1153
 - \leftpstartnumL 1160
 - \leftpstartnumR 1160
 - Leftside (environment) 5, 720
 - \Leftsidehook 727, 733
 - \Leftsidehookend 732, 733
 - \lemma 2281, 2507
 - \lemmafnt 2402, 2423
 - \line@list 685, 689
 - \line@list@stuff 45, 111
 - \line@list@stuffR ... 64, 97, 119, 563
 - \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, 546, 974, 1502, 2155
 - \line@numR 57,
204, 211, 266, 300, 319, 354,
355, 357, 374, 388, 389, 412,
532, 536, 550, 955, 1027, 1036,
1125, 1127, 1129, 1130, 1498, 2183
 - \lineation 748, 2339
 - \lineationR 126, 748
 - \linenum@out 582,
590, 595, 601, 607, 614, 622,
627, 631, 1472, 1849, 1900, 2076
 - \linenum@outR 560, 566,
568, 570, 571, 575, 578, 588,
594, 600, 606, 613, 621, 626,
630, 635, 1460, 1852, 1905, 2079
 - \linenumberlist 1126, 1130
 - \linenumincrement .. 5, 171, 2229,
2244, 2274, 2295, 2614, 2655, 2728
 - \linenummargin
 - 150, 2230, 2245, 2275, 2296, 2340
 - \linenumr@p 1446, 1450, 1498, 1502
 - \linenumrepR 194, 204
 - \linenumsep
. 199, 201, 1207, 1210, 1219, 1222
 - \linesinpar@listL
..... 220, 236, 547, 2050, 2053
 - \linesinpar@listR
..... 220, 232, 551, 2056, 2059
 - \linesonpage@listL 237, 555, 2063, 2066
 - \linesonpage@listR 233, 558, 2069, 2072
 - \list@clear
. 228–233, 236, 237, 239, 786, 816
 - \list@clearing@reg 235
 - \list@create
... 215–218, 220–222, 1226, 1456
 - \lock@disp 1086, 1090, 1095
 - \lock@off 470, 471, 479, 630, 631
 - \lock@on 626, 627
 - \longdash 2609, 2623
- M**
- \maxchunks 3, 1708, 2606
 - \maxdimen 2376
 - \maxlinesinpar@list 220, 239
 - \memorydump 6, 726, 744
 - \memorydumpL 105, 726
 - \memorydumpR 105, 744
 - \message 44, 63
 - \morenoexpands 2394
 - \mpAfootgroup 1438
 - \mpAfootins 1438
 - \mpAfootnote 1355
 - \mpBfootgroup 1439
 - \mpBfootins 1439
 - \mpBfootnote 1355
 - \mpCfootgroup 1440
 - \mpCfootins 1440
 - \mpCfootnote 1355
 - \mpDfootgroup 1441
 - \mpDfootins 1441
 - \mpDfootnote 1355
 - \mpEfootgroup 1442
 - \mpEfootins 1442
 - \mpEfootnote 1355
 - \mpvAfootnote 1359, 1363, 1368
 - \mpvBfootnote 1374, 1378, 1383
 - \mpvCfootnote 1389, 1393, 1398
 - \mpvDfootnote 1404, 1408, 1413
 - \mpvEfootnote 1419, 1423, 1428
 - \multiply 1060

N

`\n@num` 507, 635
`\n@num@reg` 513
`\namebox` 877, 882, 912,
 917, 1692, 1889, 1892, 2141, 2169
`\NeedsTeXFormat` 2
`\new@line` 895
`\new@lineR` 574, 930
`\newbox` 759, 862, 863, 1693
`\newcounter` 162,
 164, 166, 168, 762, 763, 765, 766
`\newif` . 5, 10, 31, 122, 123, 561, 718,
 719, 1214, 1622, 1749, 1751,
 1873, 1885, 2095, 2097, 2137, 2138
`\newnamebox` 1692, 1720, 1721
`\newnamecount` 1703, 1728
`\newwrite` 560
`\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` 643, 659
`\nobrak` 2392, 2393
`\noindent` 2377, 2452, 2532
`\normal@pars` 72, 790, 820
`\normalbfnoteX` 1608
`\notefontsetup` 2361
`\notenumfont` . . . 2345, 2401, 2409, 2422
`\noteschanged@true`
 77, 80, 677, 686, 1232
`\notetextfont` . . 2346, 2403, 2416, 2424
`\num@lines` 836, 1814, 1924
`\num@linesR` 758, 852, 1815, 1925
`\numberedpar@true` 798, 828
`\numberingRfalse` 71
`\numberingRtrue` 54, 91, 115
`\numberingtrue` 41, 107
`\numberpstartfalse` 7
`\numberpstarttrue` 7
`\numlabfont` 204
`\numpagelinesL`
 1909, 1965, 1976, 1980, 2103

`\numpagelinesR`
 1909, 1986, 1996, 2000, 2117

O

`\oldBfootfmt` ... 2205, 2208, 2616, 2619
`\oldchapter` 702, 711
`\oldstanza` 728, 729, 731, 750, 751, 754
`\one@line` 882, 884, 895
`\one@lineR` 758, 917, 919, 930
`\openout` 568, 571

P

`\page@action` 270, 398, 526
`\page@num` 251, 331, 1172, 1581
`\page@numR` 224,
 243, 321, 531, 536, 1025, 1148, 1193
`\pagegoal` 2135
`\Pages` 4, 1913, 2591, 2691, 2768
`pages` (environment) 4, 695
`\pagetotal` 2029, 2102, 2116
`pairs` (environment) 3, 695
`\par@line` 837, 1816, 1926
`\par@lineR` 758, 853, 1817, 1927
`\para@vfootnote` 2358
`\pausenumbering` 742
`\pausenumberingR` 90, 742
`\pend` . . 5, 725, 747, 782, 1678, 2485,
 2517, 2521, 2564, 2584, 2588, 2623
`\pendL` 725, 829
`\pendR` 747, 812, 845
`\prevgraf`
 . 836, 852, 1814, 1815, 1924, 1925
`\previous@A@number` 2386
`\previous@B@number` 2387
`\previous@C@number` 2388
`\previous@page` 2370, 2375, 2389
`\printindex` 2595
`\printlines`
 1454, 2207, 2401, 2409, 2422, 2618
`\printlinesR` 7, 1443, 2207, 2618
`\ProcessOptions` 8
`\protected@write` ... 1469, 1481, 1764
`\ProvidesPackage` 3
`\pst@rtedLfalse` 30, 40
`\pst@rtedLtrue` 108, 788
`\pst@rtedRfalse` 32, 53, 74
`\pst@rtedRtrue` 94, 116, 818
`\pstart` 5, 19, 23, 723, 746, 1680, 2448,
 2487, 2519, 2528, 2566, 2586, 2623
`\pstartL` 723, 761

- `\pstartnumfalse` 1207, 1212
`\pstartnumRfalse` 1219, 1224
`\pstartnumRtrue` 1215, 1823, 2187
`\pstartnumtrue` 1822, 2159
`\pstartR` 746, 761
- R**
- `\rbracket` 2392, 2415
`\Rcolwidth` 3, 4,
 13, 707, 827, 923, 937, 2224, 2268
`\read@linelist` 226, 564
`\rem@inder` 1130, 1132–1134
`\resumenumbering` 743
`\resumenumberingR` 90, 743
`\rightlinenumR` 197, 1143, 1151
`\rightpstartnumL` 1160
`\rightpstartnumR` 1160
Rightside (environment) 5, 738
`\Rightsidehook` 733, 749
`\Rightsidehookend` 733, 755
`\rlap` 1143, 1151, 1167, 1175, 1188, 1196
`\Rlineflag` 7, 192, 204,
 1446, 1450, 1490, 2203, 2220, 2437
`\rule` 1869
- 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` ... 1762, 1764–1767
`\select@lemmfont` 2415
`\selectlanguage` 1770, 2449, 2526
`\set@line` 644, 660, 674
`\set@line@action`
 ... 263, 367, 376, 383, 406, 528
`\setl@dlp@rbox` 1574, 1586
`\setl@drp@rbox` 1576, 1584
`\setline` 609, 2452, 2532
`\setlinenum` 617
`\setnamebox` 796, 826, 1692
`\setprintlines` 1444
`\setstanzaindents`
 ... 2218, 2265, 2607, 2656, 2729
`\shiftedversesfalse` 6
`\shiftedversestrue` 7
`\showlemma` 650, 666
`\sidenote@margin` 1514, 1518
`\sidenote@marginR` 1507, 1579
`\sidenotemargin` 1507, 2341, 2526
`\skip` 2352–2354
`\skip@lockoff` 471, 479
`\skipnumbering` 8, 634, 2623
`\skipnumbering@reg` 638
`\smash` 1869
`\splitmaxdepth` 2364
`\splittopskip` 879, 914, 2364
`\stanza` ... 728, 729, 731, 750, 751,
 754, 2232, 2247, 2277, 2298,
 2630, 2636, 2642, 2658, 2666, 2674
`\stanza@count` 1660, 1674, 1685
`\stanza@chang` 1662, 1687
`\stanzaindentbase` .. 1640, 1650, 1685
`\startlock` 625
`\startstanzahook` 1658
`\startsub` 592
`\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` 600, 601
`\sub@on` 594, 595
`\subline@num` 206, 329, 346,
 347, 349, 379, 419, 970, 975, 1503
`\subline@numR` 207,
 211, 296, 300, 319, 340, 341,
 343, 372, 410, 533, 537, 951,
 956, 1027, 1034, 1121, 1122, 1499
`\sublinenumincrement` 5, 171
`\sublinenumr@p` . 1447, 1451, 1499, 1503
`\sublinenumrepR` 194, 207
`\sublines@false` 60, 277, 996
`\sublines@true` 275, 994
`\sublock@disp` 1069, 1073, 1078
`\symplinenum` 1446
`\sza@penalty` 1669, 1673
- T**
- `\textdagger` 2551
`\textheight` 2200, 2604
`\textit` 2262, 2287, 2289, 2310, 2312,
 2322, 2462, 2475, 2496, 2501,
 2507, 2509, 2511, 2559, 2678, 2753
`\textnormal` 2393
`\textsc` 2261, 2457
`\textwidth` 14, 16, 706, 707, 2223, 2267

Lineation can be by pstart (like in ledmac 0.15).	14	<code>\do@lineL</code> : Added <code>\do@lineLhook</code> to <code>\do@lineL</code>	36
New management of hangingsymbol insertion, preventing undesirable insertions.	55	Simplified <code>\do@lineL</code> by using macros for some common code	36
Prevent shift of column separator when a verse is hanged	55	<code>\do@lineR</code> : Changed <code>\do@lineR</code> similarly to <code>\do@lineL</code>	37
<code>\affixline@numR</code> : Changed <code>\affixline@numR</code> to allow to disable line numbering (like in ledmac 0.15).	40	<code>\do@lineRhook</code> : Added <code>\do@lineLhook</code> and <code>\do@lineRhook</code>	37
<code>\Columns</code> : Line numbering by pstart.	62	<code>Leftside</code> : Added hooks into Leftside environment	31
<code>\get@nextboxR</code> : Change <code>\get@nextboxL</code> and <code>\get@nextboxR</code> to allow to disable line numbering (like in ledmac 0.15).	70	<code>\flag@end</code> : Removed extraneous spaces from <code>\flag@end</code>	27
Pstart number can be printed in side	71	<code>\ifledRcol</code> : Moved <code>\ifl@dpairing</code> to ledmac	11
v0.12		<code>\ifpst@rtedR</code> : Moved <code>\ifpst@rtedL</code> to ledmac	12
General: New new management of hangingsymbol insertion, preventing undesirable insertions.	55	<code>\l@dlinenumR</code> : Simplified <code>\leftlinenumR</code> and <code>\rightlinenumR</code> by introducing <code>\l@dlinenumR</code>	16
v0.13.1		<code>\l@dnumpstartsR</code> : Moved <code>\l@dnumpstartsL</code> to ledmac	58
General: Compatibility with optional argument in <code>\Xfootnotes</code> introduced by ledmac 0.18.1	46	<code>\ledsavesprintlines</code> : Simplified <code>\printlinesR</code> by using <code>\setprintlines</code>	50
Debug in lineation by pstart	14	<code>\ledstrutR</code> : Added <code>\ledstrutL</code> and <code>\ledstrutR</code>	67
v0.13.2		<code>\normalbfnoteX</code> : Removed extraneous spaces from <code>\normalbfnoteX</code>	54
<code>\normalbfnoteX</code> : Correct bug with familiar footnotes introduced by 0.13.1	54	<code>\Pages</code> : Added <code>\ledstrutL</code> to <code>\Pages</code>	66
v0.2		Added <code>\ledstrutR</code> to <code>\Pages</code>	66
General: Added section of babel related code	59	<code>\Rightsidehookend</code> : Added <code>\Leftsidehook</code> , <code>\Leftsidehookend</code> , <code>\Rightsidehook</code> and <code>\Rightsidehookend</code>	32
Fix babel problems	1	<code>\sublinenumrepR</code> : Added <code>\linenumrepR</code> and <code>\sublinenumrepR</code>	16
<code>\Columns</code> : Added <code>\l@dchecklang</code> and <code>\l@duselanguage</code> to <code>\Columns</code>	61	v0.3a	
<code>\Pages</code> : Added <code>\l@duselanguage</code> to <code>\Pages</code>	65	General: Minor <code>\linenummargin</code> fix	1
v0.3		<code>\line@marginR</code> : Don't just set <code>\line@marginR</code> in <code>\linenummargin</code>	15
General: Reorganize for ledarab	1	v0.3b	
<code>\affixline@numR</code> : Changed <code>\affixline@numR</code> to match new ledmac	40	General: Improved parallel page balancing	1
<code>\do@actions@nextR</code> : Used <code>\do@actions@fixedcode</code> in <code>\do@actionsR</code>	39		

	<code>\Pages</code> : Added <code>\l@dminpagelines</code> calculation for succeeding page pairs 67	<code>\hangingsymbol</code> to define the character. 1
v0.3c	General: Compatibilty with Polyglossia 1	General: Possibility to number <code>\pstart</code> 7
v0.4	General: No more <code>ledparpatch</code> . All patches are now in the main file. 1	Possibility to number the <code>pstart</code> with the commands <code>\numberpstarttrue</code> 1
v0.5	General: Corrections about <code>\section</code> and other titles in numbered sections 1	<code>\ifledRcol</code> : Moved <code>\iflledRcol</code> and <code>\ifnumberingR</code> to <code>ledmac</code> 11
v0.6	General: Be able to use <code>\chapter</code> in parallel pages. 1	v0.9.1 General: The numbering of the <code>pstarts</code> restarts on each <code>\beginnumbering</code> 1
v0.7	General: Option ‘ <code>shiftedverses</code> ’ which make there is no blank between two parallel verses with inequal length. 1	v0.9.2 General: Debug : with <code>\Columns</code> , the hanging indentation now runs on the left columns and the hanging symbol is shown only when <code>\stanza</code> is used. 1
v0.8	General: Possibility to have a symbol on each hanging of verses, like in the french typography. Redefine the commande	v0.9.3 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 <code>memoir</code> class. 1