

Parallel typesetting for critical editions: the `reledpar` package*

Maïeul Rouquette[†] based on the original `ledpar` by Peter Wilson
Herries Press[‡]

Abstract

The `reledmac` package has been used for some time for typesetting critical editions. The `reledpar` package is an extension to `reledmac` which enables texts and their critical apparatus to be typeset in parallel, either in two columns or on pairs of facing pages.

`reledpar` provides many tools and options. Normally, they are all documented in this file. Also provided is a help folder, “examples”. The folder contains additional examples (although not for all cases). Examples starting by “3-” are for basic uses, those starting by “4-” are for advanced uses.

To report bugs, please go to `ledmac`’s GitHub page and click “New Issue”: <https://github.com/maieul/ledmac/issues/>. You must open an account with `github.com` to access my page (`maieul/ledmac`). GitHub accounts are free for open-source users. You can report bug in English or in French (better).

You can subscribe to the `reledmac` email list in:
<http://geekographie.maieul.net/146>

Contents

1 Introduction	5
1.1 Aim of this package	5
1.2 Historical overview	5
2 General	5
3 Parallel columns	6
3.1 Basic use	6
3.2 Setting	7
3.2.1 Column’s width	7
3.2.2 Column’s separator	7
3.2.3 Column’s positions	7
3.2.4 Mixing two columns and one column texts	8

*This file (`reledpar.dtx`) has version number v2.2.0, last revised 2015/08/08.

[†]`maieul` at `maieul` dot `net`

[‡]`herries` dot `press` at `earthlink` dot `net`

4 Facing pages	8
4.1 Basic usage	8
4.2 Setting	9
4.2.1 Text width	9
4.2.2 Page number	9
4.2.3 Page breaking	9
4.2.4 Right page before \Pages	9
4.3 Critical and familiar footnotes	10
4.3.1 Notes height setting	10
4.3.2 Notes for one side only	10
4.3.3 Familiar notes called in the right side, but to be printed in the left side	10
5 Left and right texts	11
5.1 Environments	11
5.2 Numbering text lines and paragraphs	11
5.3 Line numbering scheme	12
5.4 Lineation system	12
5.5 Chunks	13
5.6 \AtEveryPstart and \AtEveryPstartCall	13
5.7 Language setting	14
5.8 Shifting	14
6 Verse	14
7 Side notes	15
8 Parallel ledgroups	15
8.1 General	15
8.2 Parallel ledgroups and setspace package	16
9 Sectioning commands	16
10 Notes about page number	17
I Implementation overview	18
II Preliminaries	18
II.1 Package's meta-data	18
II.2 Package's requirement	18
II.3 Package's options	18
II.4 Determining side and category of parallel processing	19
II.5 Text's width	20
II.6 Messages	20
III Sectioning commands	21

IV Line counting	25
IV.1 Setting lineation reset	25
IV.2 Setting line number margin	26
IV.3 Setting lineation start and step	26
IV.4 Setting line flag	27
IV.5 Setting line number style	27
IV.6 Print marginal line number	27
IV.7 Line-number counters and lists	28
IV.7.1 Correspond to those in <code>reledmac</code> for regular or left text	28
IV.7.2 Specific to <code>reledpar</code>	29
IV.8 Reading the line-list file	29
IV.9 Commands within the line-list file	29
IV.10 Writing to the line-list file	35
V Marking text for notes	36
V.1 Specific hooks and commands for notes	36
V.1.1 Notes to be printed on one side only	37
V.1.2 Familiar footnotes without marks	37
V.1.3 Create hooks	38
V.1.4 Init standards series (A,B,C,D,E,Z)	39
VI Pstart numbers dumping and restoration	39
VII Parallel environments	40
VIII Paragraph decomposition and reassembly	42
VIII.1 Boxes, counters, <code>\pstart</code> and <code>\pend</code>	43
VIII.2 Processing one line	48
VIII.3 Line and page number computation	52
VIII.4 Line number printing	54
VIII.5 Pstart number printing in side	57
VIII.6 Add insertions to the vertical list	58
VIII.7 Penalties	59
VIII.8 Printing leftover notes	60
IX Footnotes	60
IX.1 Line number printing	60
IX.2 Footnotes output specific to <code>\Pages</code>	61
X Cross referencing	66
XI Side notes	66
XII Familiar footnotes	68
XIII Verse	68

XIV Naming macros	70
XV Fixing babel and polyglossia	71
XVI Counts and boxes for parallel texts	73
XVII Checking text to be processed	74
XVIII Parallel columns	76
XIX Parallel pages	84
XIX.1 Specific counters	84
XIX.2 Main macro	84
XIX.3 Ensure all notes be printed at the end of parallel pages	90
XIX.4 Struts	90
XIX.5 Page clearing	91
XIX.6 Lines managing	92
XIX.7 Page break managing	93
XIX.8 Getting boxes content	96
XX Page numbering	99
XXI Sections' titles' commands	100
XXII Page break/no page break, depending on the specific line	101
XXIII Parallel ledgroup	102
XXIV Compatibility with <code>eledmac</code>	106
XXV The End	106
Appendix A Some things to do when changing version	107
Appendix A.1 Migration to <code>eledpar</code> 1.4.3	107
Appendix A.2 Migration from <code>eledpar</code> to <code>reledpar</code>	107
Appendix A.2.1 Deprecated options	107
Appendix A.2.2 <code>\renewcommandreplaced</code> with <code>command</code>	107
Appendix A.2.3 Commands the names of which have changed	108
Appendix A.3 Migration to <code>reledpar</code> 2.2.0	108
References	108
Index	108
Change History	125

1 Introduction

1.1 Aim of this package

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 `reledpar` package is an extension to `reledmac` 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. In this case, please reports them to the author via github's issues: <https://github.com/maieul/ledmac/issues/>.

This manual contains a general description of how to use `reledpar` starting in section 2; the complete source code for the package, with extensive documentation (in sections I through XXV); and an Index to the source code. As `reledpar` is an adjunct to `reledmac` we assume that you have read the `reledmac` manual. Also `reledpar` requires `reledmac` to be used, in the version distributed with version.

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. The documentation's sections are numbered in roman numeral.

On a first reading, We suggest that you should skip anything after the general documentation in first sections until I, unless you are particularly interested in the innards of `reledpar`.

1.2 Historical overview

Many of the code of this package is based on the `eledpar` package, which was based on the `ledpar`, created as an extension of the `ledmac` package.

Names of the package related to parallel typesetting have moved in parallel of names of the package related to critical edition.

Please read `reledmac`'s handbook in order to understand this evolution.

2 General

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 will want to print the text that you are editing. Unnumbered text is not printed with line numbers, and you can't use `reledmac`'s note commands with it: this is appropriate for introductions and other material added by the editor around the edited text.

The `reledpar` package lets you typeset two *numbered* texts in parallel¹. 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

¹You can use, anyway, `\numberlinefalse` to disable printing of line numbers.

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.

`reledmac` 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.

`reledpar` 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{5120}
```

meaning that there can be up to 5120 chunks in the left text and up to 5120 chunks in the right text, requiring a total of 10240 boxes. If you need more chunks then you can increase `\maxchunks`. The `\maxchunks` must be called in the preamble.

If you `\maxchunks` is too little you can get a `reledpar` 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 `\stanza`, 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, `reledmac` is a \TeX resource hog, and `reledpar` only makes things worse in this respect.

3 Parallel columns

3.1 Basic use

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

`\AtBeginPairs` Keep in mind that the `\Columns` **must be** outside of the `pairs` environment. You can use the macro `\AtBeginPairs` to insert a code at the beginning of each `pairs` environments. That could be useful to add the `\sloppy` macro to prevent overfull hboxes in two columns.

```
\AtBeginPairs{\sloppy}
```

There is no required pagebreak before or after the columns.

3.2 Setting

3.2.1 Column's width

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

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

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

3.2.2 Column's separator

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

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

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

3.2.3 Column's positions

`\columnspanposition` By default, columns are positioned to the right of the page. However, you can use `\columnspanposition{L}` to align them to the left, or `\columnspanposition{C}` to center them.

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

`\beforecolumnseparator`
`\aftercolumnseparator`

By default, the spaces around column separator are the same as the space:

- On the left of columns, if columns are aligned right.
- On the right of columns, if columns are aligned left.
- On both the left and right columns, if columns are centered.

You can redefine `\beforecolumnseparator` and `\aftercolumnseparator` length to define spaces before or after the column separator, instead of letting `reledpar` calculate them automatically.

```
\setlength{\beforecolumnseparator}{length}
\setlength{\aftercolumnseparator}{length}
```

If you want to revert to the previous behavior, just set with a negative value.

3.2.4 Mixing two columns and one column texts

`\widthliketwocolumns`

If you want to mix two-column with single-column text, you can align horizontally single-column text to two-column text with `\widthliketwocolumnstrue`. To reset this feature, use `\widthliketwocolumnsfalse`. You can also call `\widthliketwocolumns` as a global option when loading `reledmac` or `reledpar`.

`\Xnoteswidthliketwocolumns`
`\notesXwidthliketwocolumns`

In most cases, you should use `\widthliketwocolumns` in combination with `\Xnoteswidthliketwocolumns` and `\notesXwidthliketwocolumns` to align the critical/familiar footnotes with the two columns. See `reledmac`'s handbook for more details.

4 Facing pages

4.1 Basic usage

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


`\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. Note that the `\Pages` **must be** outside of the `pages` environment.

4.2 Setting

4.2.1 Text width

`\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.

`\Rcolwidth`

4.2.2 Page number

By default, `\Pages` use the standard \TeX page number scheme. This means that pages are numbered continuously following printed-book conventions: from left-hand to right-hand side, left-hand pages having even numbers, right-hand pages having odd numbers.

However, you can use the package option `sameparallelpagenumbers` to have the same page number for both left and right side. In this case, this setting will apply only for pages typeset by `\Pages`, not for “normal” pages.

Please also read advising in 10 p. 17.

4.2.3 Page breaking

`\setgoalfraction` When doing parallel pages `reledpar` has to guess where \TeX is going to put page-breaks 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 standard value is 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. You can change it using `\setgoalfraction{<newvalue>}`.

4.2.4 Right page before `\Pages`

When `\Pages` are called, it starts at a new left page, in order to have parallel pages. Consequently, if it is called on a left page, it clears the current page and then lets a right void page.

`reledpar` provides two options to customize this (eventual) right page.

`prevpgstyle=<style>` in order to set the style of this page. A common value of `<style>` is empty. Use `prevpgstyle=empty` will suppress header and footer in this page. Please also read advising in 10 p. 17.

`prevpgnotnumbered` will make this page won’t be counted in the page counter.

4.3 Critical and familiar footnotes

Of course, in “Facing pages”, the `reledmac`’s both critical and familiar footnotes can be used. However, some specific points must be taken into consideration.

4.3.1 Notes height setting

Since `eledpar` v1.13.0, long notes in facing pages can flow from left to right pages, and *vice-versa*.

However, the `reledmac` default setting for the maximum allotted size to notes is greater than `\textheight`. That makes impossible for long notes to flow across pages.² We have not changed this default setting, because we do not want to break compatibility with older version of `reledmac` and we want to be as close as possible to default \LaTeX ’s feature.

So, you MUST change the default setting via `\Xmaxnotes` (for critical notes) and `\maxhnotesX` (for familiar notes). Both commands are explained in `reledmac` handbook (?? p. ??). As an advisable setting:

```
\maxhXnotes{0.6\textheight}
\maxhnotesX{0.6\textheight}
```

4.3.2 Notes for one side only

`\Xonlyside` You may want to typeset notes on one side only (either left or right). Use `\Xonly-`
`\onlysideX` `side[⟨s⟩]{⟨p⟩}` to set critical notes, and `\onlysideX[⟨s⟩]{⟨p⟩}` to set familiar
 notes. `{⟨p⟩}` must be set to L for notes to be confined only on the left side and to R for
 notes to be confined only on the right side.

4.3.3 Familiar notes called in the right side, but to be printed in the left side

`\footnoteXnomk` As often happens, the left side has less room for text. We may want to call familiar notes
`\footnoteXmk` in the right side while using at the same time the available space in the left side to print
 them.

To achieve this, we call `\footnoteXnomk{⟨notecontent⟩}` in the left side. X is to be replaced by the series letter. We do this call in the left side after the word which matches up to the one in the right side after which we want to insert the actual footnote mark.

In the right side, we call `\footnoteXmk` at the place we want to have the footnote mark. X is to be replaced by the series letter. For example:

```
\begin{Leftside}
\beginnumbering
\pstart
A little cat\footnoteAnomk{A note.}. And so one ...
\pend
```

²The same applies to \LaTeX normal notes. Read <http://tex.stackexchange.com/a/228283/7712> for technical informations.

```

\endnumbering
\end{Leftside}
\begin{Rightside}
\beginnumbering
\pstart
Un petit chat\footnotemk. And so one ...
\pend
\endnumbering
\end{Rightside}

```

5 Left and right texts

5.1 Environments

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
Rightside in the Rightside environment. The number of Leftside and Rightside environments must be the same.

5.2 Numbering text lines and paragraphs

\beginnumbering Each section of numbered text must be preceded by \beginnumbering and fol-
\endnumbering lowed 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 T_EX’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{pages}
\begin{Leftside}
\beginnumbering

```

```

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

```

```

\numberpstarttrue
\numberpstartfalse
  \thepstartL
  \thepstartR
\skipnumbering

```

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

The command `\skipnumbering` when inserted in a line of parallel text causes the numbering of that particular line to be skipped. This can be 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.

5.3 Line numbering scheme

Within these environments you can designate the line numbering scheme(s) to be used.

5.4 Lineation system

```

\firstlinenum
\linenumincrement
\firstsublinenum
\sublinenumincrement
  \firstlinenum*
  \linenumincrement*
  \firstsublinenum*
  \sublinenumincrement*

```

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. The starred versions change both left and right numbering schemes.

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.

```

\lineationR
\lineation*

```

`\lineationR` macro is the equivalent of `reledmac \lineation` macro for the right side.

`\lineation*` macro is the equivalent of `reledmac \lineation` macro for both sides.

`\setRlineflag` A “R” is appended to the line numbers of the right texts. This may be useful for parallel columns but for parallel pages it might be more appropriate to redefine it using `\setRlineflag{<flag>}`. Use `\setRlineflag{}` to empty it.

5.5 Chunks

`\pstart` `\pend` 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.

`\autopar` The `\autopar` macro can be used, instead of manually inserting `\pstart... \pends`. Please read `reledmac`’s handbook (4.2.2 p. 15).

5.6 `\AtEveryPstart` and `\AtEveryPstartCall`

In general, remember that the moment where a `\pstart` is called is different from the moment when the `\pstart... \pend` content is printed, which is when `\Pages` or `\Columns` is processed.

Consequently:

- The argument of `\AtEveryPstart` (see 4.2.4 p. 16) is called before every chunk is printed, except if you used an optional argument for the `\pstart`.
- The argument of `\AtEveryPstartCall` is called before every `\pstart`.

5.7 Language setting

If you are using the `babel` package or the `polyglossia` 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* language setting 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.

5.8 Shifting

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.

However, sometime if the left `pstarts` are much greater than right `pstarts`, or *vice-versa*, you can decide to shift the `pstarts` on the left and right side. That means the start of `pstarts` are not aligned horizontally on the page, the shift is offset at the end of each double pages. To enable this function, load `reledpar` with the option `shiftedpstarts`.

6 Verse

If you are typesetting verse with `reledmac` 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` `reledpar` provides an `astanza` environment which you can use instead of `\stanza`. A `astanza` environment is a chunk. Consequently left and right *verse* are matched, and not, as with standard `\stanza`, left and right *verse lines*.

Within the `astanza` environment each verse line is treated as an individual paragraph, so there must be no blank lines in the environment otherwise there will be some extraneous vertical spacing. To use `astanza`, imply replace `\astanza` by `\begin{astanza}` and add `\end{astanza}` after the ending `\&`.

If you get an error message along the lines of 'Missing number, treated as zero `\sza@0@`' it is because you have forgotten to use `\setstanzaindent` to set the stanza indents.

As `astanza` is a specify type `\pstart...\pend` structure, you can:

- Add optional argument (in brackets) after `\begin{astanza}`, as the optional argument of `\pstart`.
- Use optional argument after the last `\&` as optional argument of `\pend`.

`\sethangingsymbol` Like in `reledmac`, you could use the `\sethangingsymbol` command to insert a character in each hanging line. If you use it, you must run \TeX two time. Example for the French typography

```
\sethangingsymbol{[,]}
```

You can also use it to force hanging verse to be flush right:

```
\sethangingsymbol{\protect\hfill}
```

When you use `\lednopb` make sure to use it on both sides in the corresponding verses to keep the pages in sync.

`\thestanzaL` When using `\stanzanumtrue` (8.10 p. 41) in parallel typesetting, `stanza` counter is replaced by `stanzaL` counter in left side and by `stanzaR` counter in right side. Consequently, you can redefine `\thestanzaL` and `\thestanzaR` to change their aspect.

7 Side notes

As in `reledmac`, you must use one of the following commands to add side notes: `\ledsidenote`, `\ledleftnote`, `\ledrightnote`, `\ledouterote`, `\ledinnerrote`.

The `\sidenotemargin` defines the margin of the sidenote for either left or right side, depending on the current environment. You can use `\sidenotemargin*` to define it for both sides.

8 Parallel ledgroups

8.1 General

You can also make parallel ledgroups (see the documentation of `reledmac` about ledgroups, 9 p. 42). To do it you have:

- To load `reledpar` package with the `parledgroup` option, or to add `\parledgrouptrue`.
- To push each ledgroup between `\pstart...``\pend` command.

See the following example:

```
\begin{pages}
\begin{Leftside}
\beginnumbering
```

```

\pstart
  \begin{ledgroup}
    ledgroup content
  \end{ledgroup}
\pend
\pstart
  \begin{ledgroup}
    ledgroup content
  \end{ledgroup}
\pend
\endnumbering
\end{Leftside}
\begin{Rightside}
\beginnumbering
\pstart
  \begin{ledgroup}
    ledgroup content
  \end{ledgroup}
\pend
\pstart
  \begin{ledgroup}
    ledgroup content
  \end{ledgroup}
\pend
\endnumbering
\end{Rightside}
\end{pages}
\Pages

```

8.2 Parallel ledgroups and **set space** package

If you use the **set space** package and want your notes in parallel ledgroups to be single-spaced (not half-spaced or double-spaced), just add to your preamble:

```
\setparledgroupnotespacing{\singlespacing}
```

In effect, to have correct spacing, do not change the font size of your notes.

9 Sectioning commands

The standard sectioning commands of **reledmac** are available, and provide parallel sectioning, for both two-column and two-page layout.

\eledsectnotoc By default, the section commands of the right side are not added to the table of contents. But you can change it, using **\eledsectnotoc{⟨arg⟩}**, where ⟨arg⟩ could be L (for left side) or R (for right side).

\eledsectmark By default, the headers are tokens from the left side. You can change them, using **\eledsectmark{⟨arg⟩}**, where ⟨arg⟩ could be L (for left side) or R (for right side).

10 Notes about page number

If you use `sameparallepagenumber` option (4.2.2 p. 9) or `prevpgnotnumbered` option (4.2.4 p. 9), please read the following paragraph if you want to manipulate page numbers manually.

In order to implement these two options, `reledpar` uses its own page counter, called `par@page`. Consequently, if you use at least one of these options:

1. If you modify `\thepage` command, use the value of `par@page` counter inside and not the value of `page` counter.
2. If you want to modify a page number, modify the value of `page` counter AND the value `par@page` counter.

Notes that `reledpar` automatically do it when you use `\frontmatter` and `\mainmatter` commands.

I 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.

`reledmac` 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 `reledmac` code is concerned with handling this box and its contents.

`reledpar`’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.

II Preliminaries

II.1 Package’s meta-data

Announce the name and version of the package, which is targeted for $\text{\LaTeX}2\text{e}$. The package also requires the `reledmac` package, however we do not load it automatically, because we prefer users to know it.

```

1 %<*code>
2 \NeedsTeXFormat{LaTeX2e}
3 \ProvidesPackage{reledpar}[2015/08/08 v2.2.0 reledmac
  extension for parallel texts]%
4
5 %
```

II.2 Package’s requirement

Few commands use `\xspace` command.

```

6 \RequirePackage{xspace}%
7 %
```

II.3 Package’s options

We use `xkeyval` in order to manage options with arguments.

```

8 \RequirePackage{xkeyval}
9 %
```

With the option ‘shiftedpstarts’ a long pstart on the left side (or on the right side) does not make a blank on the corresponding pstart, but the blank is put on the bottom of the page. Consequently, the pstarts on the parallel pages are shifted, but the shift stops at every end of pages.

```
\ifshiftedpstarts 10 \newif\ifshiftedpstarts
11 \DeclareOptionX{shiftedpstarts}{\shiftedpstartstrue}
12 %
```

The `parledgroup` can be called either on `reledmac` or `reledpar`.

```
13 \DeclareOptionX{parledgroup}{\parledgrouptrue}
14 %
```

`\widthliketwocolumns` The `\widthliketwocolumns` option can be called either on `reledmac` or `reledpar`.

```
15 \DeclareOptionX{widthliketwocolumns}{\widthliketwocolumnstrue}
16 %
```

`\sameparallelpagenunder` Options related to page numbering

```
\ifprevpgnotnumbered 17 \newif\ifsameparallelpagenunder
18 \newif\ifprevpgnotnumbered
19 \DeclareOptionX{sameparallelpagenunder}{\sameparallelpagenundertrue}
20 \DeclareOptionX{prevpgnotnumbered}{\prevpgnotnumberedtrue}
21 %
```

`\prevpgstyle` We store on `\prevpgstyle` the argument of the option `prevpgstyle`.

```
22 \DeclareOptionX{prevpgstyle}{\gdef\prevpgstyle{#1}}%
23 %
```

```
24 \ProcessOptionsX%
25 %
```

II.4 Determining side and category of parallel processing

As noted above, much of the code is a duplication of the original `reledmac` 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 we 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` is also set TRUE if we are doing parallel pages. `\ifledRcol` is set TRUE if we are doing the right hand text. They are defined in `reledmac`.

II.5 Text's width

`\Lcolwidth` The widths of the left and right parallel columns (or pages).

```
\Rcolwidth
26 \newdimen\Lcolwidth
27 \Lcolwidth=0.45\textwidth
28 \newdimen\Rcolwidth
29 \Rcolwidth=0.45\textwidth
30 %
```

II.6 Messages

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

```
\eledpar@error 31 \newcommand{\reledpar@error}[2]{\PackageError{reledpar
} {\#1} {\#2}}
32 %
```

```
\led@err@TooManyPstarts 33 \newcommand*{\led@err@TooManyPstarts}{%
34 \reledpar@error{Too many \string\pstart\space without
printing.
35 Some text will be lost}{\@ehc}}
36 %
```

```
\led@err@BadLeftRightPstarts 37 \newcommand*{\led@err@BadLeftRightPstarts}[2]{%
38 \reledpar@error{The numbers of left (#1) and right (#2)
39 \string\pstart s do not match}{\@ehc}}
40 %
```

```
\led@err@LeftOnRightPage 41 \newcommand*{\led@err@LeftOnRightPage}{%
\led@err@RightOnLeftPage 42 \reledpar@error{The left page has ended on a right page}{\
@ehc}}
43 \newcommand*{\led@err@RightOnLeftPage}{%
44 \reledpar@error{The right page has ended on a left page}{\
@ehc}}
45 %
```

```
rr@Leftside@PreviousNotPrinted 46 \newcommand*{\led@err@Leftside@PreviousNotPrinted}{%
r@Rightside@PreviousNotPrinted 47 \reledpar@error{You call a new Leftside environment while
the previous one has not been typeset by \string\Pages\space
or \string\Columns}{\@ehc}}
48 \newcommand*{\led@err@Rightside@PreviousNotPrinted}{%
49 \reledpar@error{You call a new Rightside environment while
the previous one has not been typeset by \string\Pages\space
or \string\Columns}{\@ehc}}
50 %
```

```

\led@err@Pages@InsideEnv 51 \newcommand*{\led@err@Pages@InsideEnv}{%
\reledpar@error{\string\Pages\space must be called *outside
52 * of the `pages` environment}{\@ehc}}
\newcommand*{\led@err@Columns@InsideEnv}{%
53 \reledpar@error{\string\Columns\space must be called *
54 outside* of the `pairs` environment}{\@ehc}}
55 %

```

```

\led@error@fail@patch@thepage 56 \newcommand{\led@error@fail@patch@thepage}{%
57 \reledpar@error{Fail to patch \string\@thepage\space
command.}{\@ehc}%
58 }%
59 %

```

```

\led@error@fail@patch@pagenumbering 60 \newcommand{\led@error@fail@patch@pagenumbering}{%
61 \reledpar@error{Fail to patch \string\pagenumbering\space
command.}{\@ehc}%
62 }%
63 %

```

```

\led@error@fail@patch@@mempnum 64 \newcommand{\led@error@fail@patch@@mempnum}{%
65 \reledpar@error{Fail to patch \string\@mempnum\space
command.}{\@ehc}%
66 }%
67 %

```

```

\led@error@fail@patch@@outputpage 68 \newcommand{\led@error@fail@patch@@outputpage}{%
69 \reledpar@error{Fail to patch \string\@outputpage\space
command.}{\@ehc}%
70 }%
71 %

```

III 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.

```

72 \newcount\section@numR
73 \section@numR=\z@
74 %

```

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

```

75 \pst@rtedLfalse
76 \newif\ifpst@rtedR
77
78 %

```

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

```

79 \newcommand*{\beginnumberingR}{%
80   \ifnumberingR
81     \led@err@NumberingStarted
82     \endnumberingR
83   \fi
84   \global\l@dnumpstartsR \z@
85   \global\pst@rtedRfalse
86   \global\numberingRtrue
87   \global\advance\section@numR \@ne
88   \global\absline@numR \z@
89   \gdef\normal@page@breakR{}
90   \gdef\l@prev@pbR{}
91   \gdef\l@prev@nopbR{}
92   \global\line@numR \z@
93   \global\@lockR \z@
94   \global\sub@lockR \z@
95   \global\sublines@false
96   \global\let\next@page@numR\relax
97   \global\let\sub@change\relax
98   \message{Section \the\section@numR R }%
99   \line@list@stuffR{\jobname.\extensionchars\the\section@numR
100  R}%
101   \l@dend@stuff
102   \setcounter{pstartR}{1}
103   \begingroup
104   \initnumbering@sectcountR
105   \gdef\eled@sectionsR@@{}%
106   \if@noeled@sec\else%
107     \makeatletter\inputIfFileExists{\jobname.eledsec\the\
section@numR R}{\makeatother}%
108     \immediate\openout\eled@sectioningR@out=\jobname.eledsec\
the\section@numR R\relax%
109   \fi%
110 }
111 %

```

`\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 `reledmac`.

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

```

111 \def\endnumberingR{%
112   \ifnumberingR
113     \global\numberingRfalse
114     \normal@pars
115     \ifnum\l@dnumstartR=0%
116       \led@err@NumberingWithoutPstart%
117     \fi%
118     \ifl@dpairing
119       \global\pst@rtedRfalse
120     \else
121       \ifx\insertlines@listR\empty\else
122         \global\noteschanged@true
123       \fi
124       \ifx\line@listR\empty\else
125         \global\noteschanged@true
126       \fi
127     \fi
128     \ifnoteschanged@
129       \led@mess@NotesChanged
130     \fi
131   \else
132     \led@err@NumberingNotStarted
133   \fi
134 \endgroup
135 \if@noeled@sec\else%
136   \immediate\closeout\eled@sectioningR@out%
137 \fi%
138 }
139
140 %

```

`\numbering@sectcountR` We do not want the numbering of the right-side section commands to be continuous with the numbering of the left side, we switch the \LaTeX counter in `\numberingR`.

```

141 \newcounter{chapterR}
142 \newcounter{sectionR}
143 \newcounter{subsectionR}
144 \newcounter{subsubsectionR}
145 \newcommand{\initnumbering@sectcountR}{
146   \let\c@chapter\c@chapterR
147   \let\c@section\c@sectionR
148   \let\c@subsection\c@subsectionR
149   \let\c@subsubsection\c@subsubsectionR
150 }
151 %

```

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

```

152 \newcommand*{\pausenumberingR}{%
153   \endnumberingR\global\numberingRtrue}
154 \newcommand*{\resumenumberingR}{%
155   \ifnumberingR
156     \global\pst@rtedRtrue
157     \global\advance\section@numR \@ne
158     \led@mess@SectionContinued{\the\section@numR R}%
159     \line@list@stuffR{\jobname.\extensionchars\the\
section@numR R}%
160     \l@dend@stuff
161     \begingroup%
162     \initnumbering@sectcountR%
163   \else
164     \led@err@numberingShouldHaveStarted
165     \endnumberingR
166     \beginnumberingR
167   \fi}
168
169 %

```

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

```

170 \newcommand*{\memorydumpL}{%
171   \endnumbering
172   \numberingtrue
173   \global\pst@rtedLtrue
174   \global\advance\section@num \@ne
175   \led@mess@SectionContinued{\the\section@num}%
176   \line@list@stuff{\jobname.\extensionchars\the\section@num}%
177   \l@dend@stuff}
178
179 \newcommand*{\memorydumpR}{%
180   \endnumberingR
181   \numberingRtrue
182   \global\pst@rtedRtrue
183   \global\advance\section@numR \@ne
184   \led@mess@SectionContinued{\the\section@numR R}%
185   \line@list@stuffR{\jobname.\extensionchars\the\section@numR
R}%
186   \l@dend@stuff}
187
188 %

```


IV Line counting

IV.1 Setting lineation reset

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. `reledpar` lets you choose different schemes for the left and right texts.

`\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`.

```

189 \newcommand*{\lineationR}[1]{\{%
190   \ifnumbering
191     \led@err@LineationInNumbered
192   \else
193     \def\@tempa{#1}\def\@tempb{page}%
194     \ifx\@tempa\@tempb
195       \global\bypage@Rtrue
196       \global\bypstart@Rfalse
197       \unless\ifnocritical@%
198         \Xpstart[][false]%
199       \fi%
200     \else
201       \def\@tempb{pstart}%
202       \ifx\@tempa\@tempb
203         \global\bypage@Rfalse
204         \global\bypstart@Rtrue
205         \unless\ifnocritical@%
206           \Xpstart%
207         \fi%
208       \else
209         \def\@tempb{section}
210         \ifx\@tempa\@tempb
211           \global\bypage@Rfalse%
212           \global\bypstart@Rfalse%
213           \unless\ifnocritical@%
214             \Xpstart[][false]%
215           \fi%
216         \else
217           \led@warn@BadLineation
218         \fi%
219       \fi
220     \fi
221   \fi}}
222 %

```

`\lineation*` `\lineation*` change the lineation system for both sides.

```

223 \WithSuffix\newcommand\lineation*[1]{\%

```

```

224 \lineation{#1}%
225 \lineationR{#1}%
226 }%
227 %

```

IV.2 Setting line number margin

`\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 would like; if it is 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.

It is defined only once time, in `reledmac`.

```

228 \newcount\line@marginR
229 %

```

By default put right text numbers at the right.

```

230 \line@marginR=\@ne
231
232 %

```

IV.3 Setting lineation start and step

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

```

233 \newcounter{firstlinenumR}
234 \setcounter{firstlinenumR}{5}
235 \newcounter{linenumincrementR}
236 \setcounter{linenumincrementR}{5}
237 %

```

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

```

238 \newcounter{firstsublinenumR}
239 \setcounter{firstsublinenumR}{5}
240 \newcounter{sublinenumincrementR}
241 \setcounter{sublinenumincrementR}{5}
242
243 %

```

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

These are the user’s macros for changing (sub) line numbers. They are defined in `reledmac`. The starred versions are specific to `eledpar`.

```

244 \WithSuffix\newcommand\firstlinenum*[1]{%
245   \setcounter{firstlinenumR}{#1}%
246   \setcounter{firstlinenum}{#1}%
247 }
248 \WithSuffix\newcommand\linenumincrement*[1]{%
249   \setcounter{linenumincrementR}{#1}%
250   \setcounter{linenumincrement}{#1}%
251 }
252 \WithSuffix\newcommand\firstsublinenum*[1]{%
253   \setcounter{subfirstlinenumR}{#1}%
254   \setcounter{subfirstlinenum}{#1}%
255 }
256 \WithSuffix\newcommand\sublinenumincrement*[1]{%
257   \setcounter{sublinenumincrementR}{#1}%
258   \setcounter{sublinenumincrement}{#1}%
259 }
260 %

```

IV.4 Setting line flag

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

```

261 \newcommand{\setRlineflag}[1]{%
262   \gdef\@Rlineflag{#1}%
263 }
264 \setRlineflag{R}
265 %

```

IV.5 Setting line number style

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

```

266 \newcommand*{\linenumrepR}[1]{\@arabic{#1}}
267 \newcommand*{\sublinenumrepR}[1]{\@arabic{#1}}
268
269 %

```

IV.6 Print marginal line number

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

```

270 \newcommand*{\leftlinenumR}{%
271   \l@dlinenumR

```

```

272 \kern\linenumsep}
273 \newcommand*{\rightlinenumR}{%
274 \kern\linenumsep
275 \l@dlinenumR}
276 \newcommand*{\l@dlinenumR}{%
277 \numlabfont\linenumrepR{\line@numR}\@Rlineflag%
278 \ifsublines@
279 \ifnum\subline@num>\z@
280 \unskip\fullstop\sublinenumrepR{\subline@numR}%
281 \fi
282 \fi}
283
284 %

```

IV.7 Line-number counters and lists

IV.7.1 Correspond to those in **reledmac** for regular or left text

We need another set of counters and lists for the right text, corresponding to those in **reledpar** for regular or left text.

\line@numR The count **\line@numR** stores the line number that is 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 have actually printed, no matter what numbers we attached to them.

```

285 \newcount\line@numR
286 \newcount\subline@numR
287 \newcount\absline@numR
288
289 %

```

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

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

```

290 \list@create{\line@listR}
291 \list@create{\insertlines@listR}
292 \list@create{\actionlines@listR}
293 \list@create{\actions@listR}
294
295 %

```

\page@numR The right text page number.

```

296 \newcount\page@numR
297
298 %

```

IV.7.2 Specific to **reledpar**

`\linesinpar@listL` `\linesinpar@listR` `\maxlinesinpar@list` 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.

```

299 \list@create{\linesinpar@listL}
300 \list@create{\linesinpar@listR}
301 \list@create{\maxlinesinpar@list}
302
303 %

```

IV.8 Reading the line-list file

`\list@clearing@regR` `\Clear the right lines for \read@linelist`

```

304 \newcommand{\list@clearing@regR}{%
305     \list@clear{\line@listR}%
306     \list@clear{\insertlines@listR}%
307     \list@clear{\actionlines@listR}%
308     \list@clear{\actions@listR}%
309     \list@clear{\linesinpar@listR}%
310     \list@clear{\linesonpage@listR}
311 }
312 %

```

`\read@linelist` `\read@linelist {<file>}` is the control sequence that is called by `\beginnum-`
bering (via `\line@list@stuff`) to open and process a line-list file; its argument
is the name of the file. . It is defined only once time in `reledmac`.

IV.9 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.

`\@nl@regR` `\@nl` `\@nl@regR` is called by `\@nl` if we are on a right side. It does everything related to the start of a new line of numbered text on a right side.

```

313 \newcommand{\@nl@regR}{%
314     \ifx\l@dchset@num\relax \else
315         \advance\absline@numR \@ne
316         \set@line@action
317         \let\l@dchset@num\relax
318         \advance\absline@numR \m@ne
319         \advance\line@numR \m@ne% % do we need this?
320     \fi
321     \advance\absline@numR \@ne

```

```

322 \ifx\next@page@numR\relax \else
323   \page@action
324   \let\next@page@numR\relax
325 \fi
326 \ifx\sub@change\relax \else
327   \ifnum\sub@change>\z@
328     \sublines@true
329   \else
330     \sublines@false
331   \fi
332   \sub@action
333   \let\sub@change\relax
334 \fi
335 \ifcase\@lockR
336 \or
337   \@lockR \tw@
338 \or\or
339   \@lockR \z@
340 \fi
341 \ifcase\sub@lockR
342 \or
343   \sub@lockR \tw@
344 \or\or
345   \sub@lockR \z@
346 \fi
347 \ifsublines@
348   \ifnum\sub@lockR<\tw@
349     \advance\subline@numR \@ne
350   \fi
351 \else
352   \ifnum\@lockR<\tw@
353     \advance\line@numR \@ne \subline@numR \z@
354   \fi
355 \fi}
356
357
358 %

```

`\last@page@numR` `\last@page@numR` store the page number of the last right page. It is modified by `\fix@page` `\fix@page`, defined by `reledmac`.

```

359 \newcount\last@page@numR
360 \last@page@numR=-10000
361
362 %

```

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

<code>\@set</code>	The <code>\@set{⟨num⟩}</code> macro sets the current visible line number to the value specified as its argument. This is used to implement <code>\setline</code> . It is defined in <code>reledmac</code> .
<code>\l@d@set</code>	The <code>\l@d@set{⟨num⟩}</code> macro sets the line number for the next <code>\pstart...</code> to the value specified as its argument. This is used to implement <code>\setlinenum</code> . It is defined in <code>reledmac</code> .
<code>\page@action</code>	<code>\page@action</code> adds an entry to the action-code list to change the page number. It is defined in <code>reledmac</code> .
<code>\set@line@action</code>	<code>\set@line@action</code> adds an entry to the action-code list to change the visible line number. It is defined in <code>reledmac</code> .
<code>\sub@action</code>	<code>\sub@action</code> adds an entry to the action-code list to turn sub-lineation on or off, according to the current value of the <code>\ifsublines@</code> flag. It is defined in <code>reledmac</code> .
<code>\do@lockon</code> <code>\do@lockonR</code>	<code>\lock@on</code> adds an entry to the action-code list to turn line number locking on. The current setting of the sub-lineation flag tells us whether this applies to line numbers or sub-line numbers. It is defined in <code>reledmac</code> , however the code specific to right side is defined here, in <code>\do@lockonR</code> .

```

363 \newcount \@lockR
364 \newcount \sub@lockR
365
366 \newcommand* {\do@lockonR} {%
367   \xright@appenditem{\the\absline@numR}\to\actionlines@listR
368   \ifsublines@
369     \xright@appenditem{-1005}\to\actions@listR
370     \ifnum\sub@lockR=\z@
371       \sub@lockR \@ne
372     \else
373       \ifnum\sub@lockR=\thr@@
374         \sub@lockR \@ne
375       \fi
376     \fi
377   \else
378     \xright@appenditem{-1003}\to\actions@listR
379     \ifnum\@lockR=\z@
380       \@lockR \@ne
381     \else
382       \ifnum\@lockR=\thr@@
383         \@lockR \@ne
384       \fi
385     \fi
386   \fi}
387
388 %

```

<code>\lock@off</code>	<code>\lock@off</code> adds an entry to the action-code list to turn line number locking off. It is defined in <code>reledmac</code> , however the code specific to right side is defined here, in
<code>\do@lockoff</code>	
<code>\do@lockoffR</code>	<code>\do@lockoffR</code> .
<code>\skip@lockoff</code>	

```

389
390
391 \newcommand{\do@lockoffR}{%
392 \xright@appenditem{\the\absline@numR}\to\actionlines@listR
393 \ifsublines@
394 \xright@appenditem{-1006}\to\actions@listR
395 \ifnum\sub@lockR=\tw@
396 \sub@lockR \thr@@
397 \else
398 \sub@lockR \z@
399 \fi
400 \else
401 \xright@appenditem{-1004}\to\actions@listR
402 \ifnum\@lockR=\tw@
403 \@lockR \thr@@
404 \else
405 \@lockR \z@
406 \fi
407 \fi}
408
409
410 %

```

`\n@num`

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

`\@ref@regR`
`\insert@countR`

- #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`.

```

411 \newcount\insert@countR
412 %

```

- #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`.

`\@ref` itself is defined in `reledmac`. It calls `\ref@reg` or `\ref@regR`, depending whether we are in left or right side. Here, we define only `\ref@regR`, `\ref@reg` is already defined in `reledmac`.

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

```

413 \newcommand*{\@ref@regR}[2]{%
414 \global\advance\@edtext@level by 1%
415 \global\insert@countR=#1\relax
416 \loop\ifnum\insert@countR>\z@

```



```

417 \xright@appenditem{\the\absline@numR}\to\
insertlines@listR
418 \global\advance\insert@countR \m@ne
419 \repeat
420 %

```

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.

```

421 \begingroup
422 \let\@ref=\dummy@ref
423 \let\@lopR\@gobble
424 \let\page@action=\relax
425 \let\sub@action=\relax
426 \let\set@line@action=\relax
427 \let\@lab=\relax
428 \let\@lemma=\relax
429 \let\@sw\@gobblethree%
430 #2
431 \global\endpage@num=\page@numR
432 \global\endline@num=\line@numR
433 \global\endsubline@num=\subline@numR
434 \endgroup
435 %

```

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

```

436 \xright@appenditem%
437 {\the\page@numR|\the\line@numR}%
438 \ifsublines@ \the\subline@numR \else 0\fi}%
439 \the\endpage@num|\the\endline@num}%
440 \ifsublines@ \the\endsubline@num \else 0\fi}\to\
line@listR
441 %

```

Create a list which will store all the second argument of each \@sw in this lemma, at this level.

```

442 \expandafter\list@create\expandafter{\csname
sw@list@edtext@tmp@\the\@edtext@level\endcsname}%
443 %

```

Declare and init boolean for lemma in this level.

```

444 \providebool{lemmacommand@\the\@edtext@level}%
445 \boolfalse{lemmacommand@\the\@edtext@level}%
446 %

```

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

```

447 #2
448 % Now, we store the list of \protect\cs{@sw} of this current
\protect\cs{edtext} as an element of
449 % the global list of list of \protect\cs{@sw} for a \protect\
cs{edtext} depth.
450 % \begin{macrocode}
451 \ifnum\@edtext@level>0%
452 \def\create@this@edtext@level{\expandafter\list@create\
expandafter{\csname sw@list@edtextR@the\@edtext@level\
endcsname}}%
453 \ifcsundef{sw@list@edtextR@the\@edtext@level}{\
create@this@edtext@level}{}%
454 \letcs{\@tmp}{sw@list@edtextR@the\@edtext@level}%
455 \letcs{\@tmpp}{sw@list@edtext@tmp@the\@edtext@level}%
456 \xright@appenditem{\expandonce\@tmpp}\to\@tmp%
457 \global\cslet{sw@list@edtextR@the\@edtext@level}{\@tmp
}%
458 \fi%
459 %

```

Decrease edtext level counter.

```

460 \global\advance\@edtext@level by -1%
461 }
462 %

```

\@pend **\@pend**{*num*} adds its argument to the `\linesinpar@listL` list, and analogously for `\@pendR`. If needed, it resets line number. Both are defined in `reledmac`, but they are empty. They are really defined only in `reledpar`.

```

463 \renewcommand*{\@pend}[1]{%
464 \ifbypstart@\global\line@num=0\fi%
465 \xright@appenditem{#1}\to\linesinpar@listL}
466 \renewcommand*{\@pendR}[1]{%
467 \ifbypstartR@\global\line@numR=0\fi
468 \xright@appenditem{#1}\to\linesinpar@listR}
469
470 %

```

\@lopL **\@lopL**{*num*} adds its argument to the `\linesonpage@listL` list, and analogously for `\@lopR`. Both are defined in `reledmac`, but they are empty. They are really defined only in `reledpar`.

```

471 \renewcommand*{\@lopL}[1]{%
472 \xright@appenditem{#1}\to\linesonpage@listL}
473 \renewcommand*{\@lopR}[1]{%
474 \xright@appenditem{#1}\to\linesonpage@listR}
475
476 %

```

IV.10 Writing to the line-list file

We have now defined all the counters, lists, and commands involved in reading the line-list file at the start of a section. Now we will cover the commands that `reledmac` 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`.

```
477 \newwrite\linenum@outR
478 %
```

`\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.

```
\st@linenum@out@Rtrue
\st@linenum@out@Rfalse
479 \newif\iffirst@linenum@out@R
480 \first@linenum@out@Rtrue
481 %
```

`\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.

```
482 \newcommand*{\line@list@stuffR}[1]{%
483   \read@linelist{#1}%
484   \iffirst@linenum@out@R
485     \immediate\closeout\linenum@outR
486     \global\first@linenum@out@Rfalse
487     \immediate\openout\linenum@outR=#1
488     \immediate\write\linenum@outR{\string\line@list@version
489   {\this@line@list@version}}}%
489 \else
490   \if@minipage%
491     \leavevmode%
492   \fi%
493   \closeout\linenum@outR%
494   \openout\linenum@outR=#1%
495 \fi}
496 %
497 %
```

`\new@lineL` The `\new@lineL` macro sends the `\@nl` command to the left text line-list file, to mark the start of a new text line.

```
498 \newcommand*{\new@lineL}{%
499   \write\linenum@out{\string\@nl[\the\c@page][\thepage]}}
500 %
```

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

```

501 \newcommand*{\new@lineR}{%
502   \write\linenum@outR{\string\@nl[\the\c@page][\thepage]}}
503 %

```

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

`\startsub` `\endsub` `\startsub` and `\endsub` turn sub-lineation on and off, by writing appropriate instructions to the line-list file. There are both defined in `reledmac`.

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

`\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. It is defined in `reledmac`.

`\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. It is defined in `reledmac`.

`\startlock` `\endlock` 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. They are defined in `reledmac`.

`\skipnumbering`

V Marking text for notes

The `\edtext` 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`
`\edtext`
`\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`. It is defined in `reledmac`.

V.1 Specific hooks and commands for notes

The `reledmac` `\newseries@` initializes commands which are linked to notes series. However, to keep `reledmac` as light as possible, it does not define commands which are specific to `reledpar`. This is what does `\newseries@par`. The specific hooks are also defined here.

```
\newseries@par\newcommand{\newseries@par}[1]{%
%
```

V.1.1 Notes to be printed on one side only

`reledpar` allows notes to be printed on one side only. We need to declare these options. We also need boolean flags, and to set them to true when a note series is not printed on one side. We check the `nofamiliar` and `nocritical` `Eledmac` options.

```
506 \unless\ifnofamiliar@%
507 \csgdef{onlysideX@#1}{}%
508 \global\newbool{keepforsideX@#1}%
509 \fi%
510 \unless\ifnocritical@%
511 \global\newbool{keepforXside@#1}%
512 \csgdef{Xonlyside@#1}{}%
513 \fi%
514 %
```

V.1.2 Familiar footnotes without marks

The `\footnoteXnomk` commands are for notes which are printed on the left side, while they are called in the right side. Basically, they set first toggle `\nomark@` to true, then call the `\footnoteX`, and finally add the footnote counter in the footnote counter list.

First, check the `nofamiliar` option of `reledmac`.

```
515 \unless\ifnofamiliar@%
516 % So declare the list.
517 % \begin{macrocode}
518 \expandafter\list@create\csname footnote#1@mk\endcsname
519 %
520 %
```

Then, declare the `\footnoteXnomk` command.

```
520 \expandafter\newcommand\csname footnote#1nomk\endcsname
521 [1]{%
522 %
```

First step: just call the normal `\footnoteX`, saying that we do not want to print the mark.

```
522 \toggletrue{nomk@}%
523 \csuse{footnote#1}{##1}%
524 \togglefalse{nomk@}%
525 %
```

Second, and last, step: store the footnote counter in the footnote counters list. We use some `\let`, because `\xright@appenditem` is difficult to use with `\expandafter`.

```

526 \letcs{\@tmp}{footnote#1@mk}%
527 \numdef\@tmpa{\csuse{c@footnote#1}}%
528 \global\xright@appenditem{\@tmpa}\to\@tmp%
529 \global\cslet{footnote#1@mk}{\@tmp}%
530 }%
531 %

```

Then, declare the command which inserts the footnotemark in the right side.

```

532 \expandafter\newcommand\csname footnote#1mk\endcsname{%
533 %

```

Get the first element of the footnote mark list. As `\gl@p` is difficult to use with dynamic name macro, we use `\let` commands.

```

534 \letcs{\@tmp}{footnote#1@mk}%
535 \gl@p\@tmp\to\@tmpa%
536 \global\cslet{footnote#1@mk}{\@tmp}%
537 %

```

Set the footnotecounter with it. For the sake of security, we make a backup of the previous value.

```

538 \letcs{\old@footnote}{c@footnote#1}%
539 \setcounter{footnote#1}{\@tmpa}%
540 %

```

Define the footnote mark and print it

```

541 \protected@csxdef{@thefnmark#1}{\csuse{thefootnote
#1}}%
542 \csuse{@footnotemark#1}%
543 %

```

Restore previous footnote counter and finally add space.

```

544 \setcounter{footnote#1}{\old@footnote}%
545 \xspace%
546 }%
547 %

```

End of tools for familiar notes without marks

```

548 \fi
549 %

```

End of `\newseries@par`.

```

550 }%
551 %

```

V.1.3 Create hooks

Read the `reledmac` code handbook about `\newhookcommand@series`. Here, we create hooks which are specific to `reledpar`.

```

552 \unless\ifnocritical@%
553   \newhookcommand@series{Xonlyside}%
554 \fi%
555 \unless\ifnofamiliar@%
556   \newhookcommand@series{onlysideX}%
557 \fi
558
559
560 %

```

V.1.4 Init standards series (A,B,C,D,E,Z)

`\init@series@par` `\newseries@par` is called by `\newseries`. However, this last command is called before `reledpar` is loaded. Thus, we need to initiate a specific series hook for `reledpar`.

```

561 \newcommand{\init@series@par}{%
562   \def\do##1{\newseries@par{##1}}%
563   \dolistloop{\@series}%
564 }%
565 \init@series@par%
566 %

```

VI Pstart numbers dumping and restoration

While in `reledmac` the footnotes are inserted in the same time as the `\pstart... \pend` are read, in `reledpar` they are inserted when the `\Columns` or `\Pages` commands are called. Consequently, if we do nothing, the value of the `PstartL` and `PstartR` counters are not the same in the main text and in the notes. To solve this problem, we dump the values in two list (one by side) when processing `\pstart` and restore these at each `\pstart` when calling `\Columns` or `\Pages`. We also dump and restore the value of the boolean `\ifnumberpstart`.

So, first step, creating the lists. Here, “pc” means “public counters”.

```

\list@pstartL@pc 567 \list@create{\list@pstartL@pc}%
\list@pstartR@pc 568 \list@create{\list@pstartR@pc}%
569 %

```

Two commands to dump current pstarts. We prefer two commands to one with argument indicating the side, because the commands are short, and so we save one test (or a `\csname` construction).

```

\dump@pstartL@pc 570 \def\dump@pstartL@pc{%
\dump@pstartR@pc 571   \xright@appenditem{\the\c@pstartL}\to\list@pstartL@pc%
572   \global\cslet{numberpstart@L\the\l@dumpstartsL}{\ifnumberpstart}%

```

```

573 }%
574
575 \def\dump@pstartR@pc{%
576   \xright@appenditem{\the\c@pstartR}\to\list@pstartR@pc%
577   \global\cslet{numberpstart@R\the\l@dnumpstartsR}{\
578   ifnumberpstart}%
579 }%
580 %

```

`\restore@pstartL@pc` And so, the commands to restore them.
`\restore@pstartR@pc`

```

581 \def\restore@pstartL@pc{%
582   \ifx\list@pstartL@pc\empty\else%
583     \gl@p\list@pstartL@pc\to\@temp%
584     \global\c@pstartL=\@temp%
585   \fi%
586 }%
587 \def\restore@pstartR@pc{%
588   \ifx\list@pstartR@pc\empty\else%
589     \gl@p\list@pstartR@pc\to\@temp%
590     \global\c@pstartR=\@temp%
591   \fi%
592 }%
593 %

```

VII Parallel environments

The initial set up for parallel processing is deceptively simple.
 pairs pages

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

```

594 \newenvironment{pairs}{%}
595   \l@dpairingtrue
596   \l@dpagingfalse
597   \initnumbering@quote
598   \at@begin@pairs%
599 }{%
600   \l@dpairingfalse
601 }
602
603 %

```

`\AtBeginPairs` The `\AtBeginPairs` macro just define a `\at@begin@pairs` macro, called at the beginning of each `pairs` environments.


```

604 \newcommand{\AtBeginPairs}[1]{\xdef\at@begin@pairs{#1}}%
605 \def\at@begin@pairs{}%
606
607 %

```

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.

```

608 \newenvironment{pages}{%
609   \l@dpairingtrue
610   \l@dpagingtrue
611   \initnumbering@quote
612   \setlength{\Lcolwidth}{\textwidth}%
613   \setlength{\Rcolwidth}{\textwidth}%
614 }{%
615   \l@dpairingfalse
616   \l@dpagingfalse
617 }
618
619 %

```

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

```

620 \newif\ifinstanzaL
621 \newif\ifinstanzaR
622 %

```

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.

```

623 \newenvironment{Leftside}{%
624   \expandafter\ifvoid\csname l@dLcolrawbox1\endcsname\else%
625   \led@err@Leftside@PreviousNotPrinted%
626   \fi%
627   \ledRcolfalse
628   \setcounter{pstartL}{1}
629   \let\pstart\pstartL
630   \let\thepstart\thepstartL
631   \let\pend\pendL
632   \let\memorydump\memorydumpL
633   \Leftsidehook
634   \let\old@startstanza\@startstanza
635   \def\@startstanza[##1]{\global\instanzaLtrue\old@startstanza[##1]}
636 }{
637   \Leftsidehookend}
638 %

```

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

`\Leftsidehookend`

`\Rightsidehook`

`\Rightsidehookend`

```

639 \newcommand*{\Leftsidehook}{}
640 \newcommand*{\Leftsidehookend}{}
641 \newcommand*{\Rightsidehook}{}
642 \newcommand*{\Rightsidehookend}{}
643
644 %

```

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.

```

645 \newenvironment{Rightside}{%
646   \expandafter\ifvoid\cscname_l@dRcolrawbox1\endcsname\else%
647   \led@err@Rightside@PreviousNotPrinted%
648   \fi%
649   \ledRcoltrue
650   \let\beginnumbering\beginnumberingR
651   \let\endnumbering\endnumberingR
652   \let\pausenumbering\pausenumberingR
653   \let\resumenumbering\resumenumberingR
654   \let\memorydump\memorydumpR
655   \let\thepstart\thepstartR
656   \let\pstart\pstartR
657   \let\pend\pendR
658   \let\ledpb\ledpbR
659   \let\lednopb\lednopbR
660   \let\lineation\lineationR
661   \Rightsidehook
662   \let\old@startstanza\@startstanza
663   \def\@startstanza[##1]{\global\instanzaRtrue\
old@startstanza[##1]}
664 }{%
665   \ledRcolfalse
666   \Rightsidehookend
667 }
668
669 %

```

VIII 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.

VIII.1 Boxes, counters, `\pstart` and `\pend`

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

When we first form the paragraph, it goes into a box register, `\l@dLcolrawbox` or `\l@dRcolrawbox` for right text, instead of onto the current vertical list. The `\ifnumberedpar@` flag will be `true` while a paragraph is being processed in that way. `\num@lines(R)` will store the number of lines in the paragraph when it is 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.

```
670 \newcount\num@linesR
671 \newbox\one@lineR
672 \newcount\par@lineR
673 %
```

`\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.
`\pstartR` `\pstart` needs to appear at the start of every paragraph that is to be numbered.

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

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

```
674
675 \newcounter{pstartL}
676 \renewcommand{\thepstartL}{\bfseries\@arabic{c@pstartL}. }
677 \newcounter{pstartR}
678 \renewcommand{\thepstartR}{\bfseries\@arabic{c@pstartR}. }
679
680 \newcommandx*{\pstartL}[1][1]{%
681   \if@nobreak%
682     \let\@oldnobreak\@nobreaktrue%
683   \else%
684     \let\@oldnobreak\@nobreakfalse%
685   \fi%
686   \@nobreaktrue%
687   \ifluatex%
688     \xdef\l@luatextextdir@L{\the\luatextextdir}%
689     \xdef\l@luatexpardir@L{\the\luatexpardir}%
690     \xdef\l@luatexbodydir@L{\the\luatexbodydir}%
691   \fi%
692   \ifnumbering \else%
693     \led@err@PstartNotNumbered%
694     \beginnumbering%
695   \fi%
696   \ifnumberedpar@%
```

```

697 \led@err@PstartInPstart%
698 \pend%
699 \fi%
700 %

```

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

```

701 \ifpstartedL\else%
702 \list@clear{\inserts@list}%
703 \global\let\next@insert=\empty%
704 \global\pstartedLtrue%
705 \fi%
706 \begingroup\normal@pars%
707 %

```

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

```

708 \global\advance\l@dnumpstartsL \@ne%
709 \ifnum\l@dnumpstartsL>\l@dc@maxchunks%
710 \led@err@TooManyPstarts%
711 \global\l@dnumpstartsL=\l@dc@maxchunks%
712 \fi%
713 \global\setnamebox{\l@dLcolrawbox\the\l@dnumpstartsL}=\vbox\
bgroup%
714 %

```

We set all the usual interline penalties to zero; this ensures that there will 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.

```

715 \l@dzeropenalties%
716 \ifautopar\else%
717 \ifnumberpstart%
718 \ifsidepstartnum%
719 \else%
720 \thepstartL%
721 \fi%
722 \fi%
723 \fi%
724 \hsize=\Lcolwidth%
725 \numberedpar@true%
726 \iflabelpstart\protected@edef\@currentlabel%
727 {\p@pstartL\thepstartL}\fi%
728 %

```

Dump the optional arguments

```

729 \ifstrempy{#1}%
730 {\csgdef{before@pstartL@\the\l@dnumpstartsL}{\
at@every@pstart}}%

```

```

731 {\csgdef{before@pstartL@the\l@dnumpstartsL}{\noindent
#1}}%
732 \at@every@pstart@call%
733 %

```

Gobble following space (automatically done if there is no optional argument)

```

734 \ignorespaces%
735 %

```

```

736 }
737 %

```

The same for right side.

```

738 \newcommandx*{\pstartR}[1][1]{%
739 \if@nobreak%
740 \let\@oldnobreak\@nobreaktrue%
741 \else%
742 \let\@oldnobreak\@nobreakfalse%
743 \fi%
744 \@nobreaktrue%
745 \ifluatex%
746 \xdef\l@luatextextdir@R{\the\luatextextdir}%
747 \xdef\l@luatexpardir@R{\the\luatexpardir}%
748 \xdef\l@luatexbodydir@R{\the\luatexbodydir}%
749 \fi%
750 \ifnumberingR \else%
751 \led@err@PstartNotNumbered%
752 \beginnumberingR%
753 \fi%
754 \ifnumberedpar@%
755 \led@err@PstartInPstart%
756 \pendR%
757 \fi%
758 \ifpstart@rtedR\else%
759 \list@clear{\inserts@listR}%
760 \global\let\next@insertR=\empty%
761 \global\pstart@rtedRtrue%
762 \fi%
763 \begingroup\normal@pars%
764 \global\advance\l@dnumpstartsR \@ne%
765 \ifnum\l@dnumpstartsR>\l@dc@maxchunks%
766 \led@err@TooManyPstarts%
767 \global\l@dnumpstartsR=\l@dc@maxchunks%
768 \fi%
769 \global\setnamebox{l@dRcolrawbox\the\l@dnumpstartsR}=\vbox\
bgroup%
770 \l@dzeropenalties%
771 \ifautopar\else%
772 \ifnumberpstart%
773 \ifsidepstartnum\else%

```

```

774     \thepstartR%
775     \fi%
776     \fi%
777     \fi%
778     \hsize=\Rcolwidth%
779     \numberedpar@true%
780     \iflabelpstart\protected@edef\@currentlabel%
781       {\p@pstartR\thepstartR}\fi%
782     \ifstrepty{#1}%
783       {\csgdef{before@pstartR@\the\l@dnumpstartsR}{\
at@every@pstart}}%
784       {\csgdef{before@pstartR@\the\l@dnumpstartsR}{\noindent
#1}}}%
785     \at@every@pstart@call%
786     \ignorespaces%
787   }
788 %

```

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

```

789 \newcommand*{\pendL}[1][1]{%
790   \ifnumbering \else%
791     \led@err@PendNotNumbered%
792   \fi%
793   \ifnumberedpar@ \else%
794     \led@err@PendNoPstart%
795   \fi%
796 %

```

We immediately call `\endgraf` to end the paragraph; this ensures that there will be no large interline penalties to prevent us from slicing the paragraph into pieces.

```

797   \endgraf\global\num@lines=\prevgraf\egroup%
798   \global\par@line=0%
799 %

```

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

```

800   \endgroup%
801   \ignorespaces%
802   \@oldnobreak%
803   \dump@pstartL@pc%
804   \ifnumberpstart%
805     \addtocounter{pstartL}{1}%
806   \fi
807   \parledgroup@beforenotes@save{L}%
808 %

```

Dump content of the optional argument.

```

809   \ifstrepty{#1}%

```

```

810 {\csgdef{after@pendL@the\l@dumpstartsL}{\at@every@pend
}}}%
811 {\csgdef{after@pendL@the\l@dumpstartsL}{\noindent#1}}}%
812 }
813 %

```

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

```

814 \newcommand*{\pendR}[1][1]{%
815   \ifnumberingR \else%
816     \led@err@PendNotNumbered%
817   \fi%
818   \ifnumberedpar@ \else%
819     \led@err@PendNoPstart%
820   \fi%
821   \endgraf\global\num@linesR=\prevgraf\egroup%
822   \global\par@lineR=0%
823   \endgroup%
824   \ignorespaces%
825   \@oldnobreak%
826   \dump@pstartR@pc%
827   \ifnumberpstart%
828     \addtocounter{pstartR}{1}%
829   \fi%
830   \parledgroup@beforenotes@save{R}%
831   \ifstrempy{#1}%
832     {\csgdef{after@pendR@the\l@dumpstartsR}{\at@every@pend
}}}%
833     {\csgdef{after@pendR@the\l@dumpstartsR}{\noindent#1}}}%
834   }
835 %
836 %

```

`\AtEveryPstartCall` The `\AtEveryPstartCall` argument is called when the `\pstartL` or `\pstartR` is called. That is different of `\AtEveryPstart` the argument of which is called when the `\pstarts` are printed.

```

837 \newcommand{\AtEveryPstartCall}[1]{\gdef\at@every@pstart@call
{#1}}}%
838 \gdef\at@every@pstart@call{}%
839 %

```

`\print@last@after@pendL` Two booleans set to true, when the time is to print the last optional argument of a `\pend`.

```

840 \newif\ifprint@last@after@pendL%
841 \newif\ifprint@last@after@pendR%
842 %

```

VIII.2 Processing one line

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

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

```
843 \newbox\l@dleftbox
844 \newbox\l@drightbox
845
846 %
```

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

```
847 \newcount\countLline
848 \countLline \z@
849 \newcount\countRline
850 \countRline \z@
851
852 %
```

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

```
853 \newcount\@donereallinesL
854 \newcount\@donetotallinesL
855 \newcount\@donereallinesR
856 \newcount\@donetotallinesR
857
858 %
```

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

```
859 \newcommand*{\do@lineL}{%
860 \letcs{\ifnumberpstart}{numberpstart@L\the\l@dpscl}%
861 \advance\countLline \@ne%
862 \ifvbox\namebox{l@dLcolrawbox\the\l@dpscl}%
863 {\vbadness=10000%
864 \splittopskip=\z@%
865 \do@lineLhook%
866 \l@emptyd@ta%
867 \global\setbox\one@line=\vsplit\namebox{l@dLcolrawbox\the\l@dpscl}%
868 to\baselineskip}%
869 \IfStrEq{\splitfirstmarks\parledgroup@}{begin}{\parledgroup@notes@startL}{}%
870 \unvbox\one@line \global\setbox\one@line=\lastbox%
```



```

871 \getline@numL%
872 \ifnum \@lock>\@ne%
873   \inserthangingsymboltrue%
874 \else%
875   \inserthangingsymbolfalse%
876 \fi%
877 \setbox\l@dleftbox%
878 \hb@xt@ \Lcolwidth{%
879   \ifl@dhidenumber%
880     \global\l@dhidenumberfalse%
881     \f@x@l@cks%
882   \else%
883     \affixline@num%
884   \fi%
885   \xifinlist{\the\l@dpscL}{\eled@sections@@}%
886   {\add@inserts\affixside@note}%
887   {\print@lineL}}%
888 \add@penaltiesL%
889 \global\advance\@donereallinesL\@ne%
890 \global\advance\@donetotallinesL\@ne%
891 \else%
892   \setbox\l@dleftbox \hb@xt@ \Lcolwidth{\hspace*{\Lcolwidth}}
893   %
894   \global\advance\@donetotallinesL\@ne%
895 \fi}
896
897 %

```

`\print@lineL` `\print@lineL` is for lines without a sectioning command. See `reledmac` definition of `\print@line` for handbook.

```

898 \def\print@lineL{%
899   \affixpstart@numL%
900   \l@dld@ta %space kept for backward compatibility
901   \add@inserts\affixside@note%
902   \l@dlsn@te %space kept for backward compatibility
903   {\ledllfill\hb@xt@ \Lcolwidth{%
904     \do@insidelineLhook%
905     \ifluatex%
906       \luatextextdir\l@luatextextdir@L%
907     \fi%
908     \new@lineL%
909     \inserthangingsymbolL%
910     \l@dunhbox@line{\one@line}}\ledrlfill\l@drd@ta%
911     \l@drsn@te}}
912
913 %

```

`\print@eledsectionL` `\print@eledsectionL` is for line with macro code.

```

914 \def\print@eledsectionL{%%
915   \addtocounter{pstartL}{-1}%
916   \ifdefstring{\@eledsectnotoc}{L}{\ledsectnotoc}{}
917   \ifdefstring{\@eledsectmark}{L}{}{\ledsectnomark}
918   \numdef{\temp@}{\l@dpscL-1}%
919   \xifinlist{\temp@}{\eled@sections@}{\@nbreaktrue}{\@nbreakfalse}%
920   \@eled@sectioningtrue%
921   \bgroup%
922   \ifluatex%
923     \luatextextdir\l@luatextextdir@L%
924     \luatexpardir\l@luatexpardir@L%
925     \luatexbodydir\l@luatexbodydir@L%
926     \ifdefstring{\l@luatextextdir@L}{TRT}{\@RTLtrue}{}%
927     \fi%
928     \csuse{eled@sectioning@the\l@dpscL}%
929   \egroup%
930   \@eled@sectioningfalse%
931   \global\csundef{eled@sectioning@the\l@dpscL}%
932   \if@RTL%
933     \hspace{-3\paperwidth}%
934     {\hbox{\l@dunhbox@line{\one@line}} \new@line}%
935   \else%
936     \hspace{3\paperwidth}%
937     {\new@line \hbox{\l@dunhbox@line{\one@line}}}%
938   \fi%
939   \vskip\eledsection@correcting@skip%
940 }
941
942 %

```

`\dolineLhook` `\dolineRhook` These high-level commands just redefine the low-level commands. They have to be used be user, without `\makeatletter`.

```

\doinlinedlineLhook \doinlinedlineRhook
943 \newcommand*{\dolineLhook}[1]{\gdef\do@lineLhook{#1}}%
944 \newcommand*{\dolineRhook}[1]{\gdef\do@lineRhook{#1}}%
945 \newcommand*{\doinlinedlineLhook}[1]{\gdef\do@insidelineLhook{#1}}%
946 \newcommand*{\doinlinedlineRhook}[1]{\gdef\do@insidelineRhook{#1}}%
947
948 %

```

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

```

949 \newcommand*{\do@lineLhook}{}
950 \newcommand*{\do@lineRhook}{}
951 \newcommand*{\do@insidelineLhook}{}
952 \newcommand*{\do@insidelineRhook}{}

```

953
954 %

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

```

955 \newcommand*{\do@lineR}{%
956   \letcs{\ifnumberpstart}{numberpstart@R\the\l@dpscR}%
957   \ledRcol@true%
958   \advance\countRline \@ne%
959   \ifvbox\namebox{l@dRcolrawbox\the\l@dpscR}%
960   {\vbadness=10000%
961    \splittopskip=\z@%
962    \do@lineRhook%
963    \l@emptyd@ta%
964    \global\setbox\one@lineR=\vsplit\namebox{l@dRcolrawbox\the
\l@dpscR}%
965                                     to\baselineskip}%
966   \IfStrEq{\splitfirstmarks\parledgroup@}{begin}{\
parledgroup@notes@startR}{}%
967   \unvbox\one@lineR \global\setbox\one@lineR=\lastbox%
968   \getline@numR%
969   \ifnum\@lockR>\@ne%
970     \inserthangingsymbolRtrue%
971   \else%
972     \inserthangingsymbolRfalse%
973   \fi%
974   \setbox\l@drightbox%
975   \hb@xt@ \Rcolwidth{%
976     \ifl@dhidenumber%
977       \global\l@dhidenumberfalse%
978       \f@x@l@cksR%
979     \else%
980       \affixline@numR%
981     \fi%
982     \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}%
983     {\add@insertsR\affixside@noteR}%
984     {\print@lineR}%
985   }%
986   \add@penaltiesR%
987   \global\advance\@donereallinesR\@ne%
988   \global\advance\@donetotallinesR\@ne%
989 \else%
990   \setbox\l@drightbox \hb@xt@ \Rcolwidth{\hspace*{\Rcolwidth
}}%
991   \global\advance\@donetotallinesR\@ne%
992 \fi%
993 \ledRcol@false%
994 }
995

```

996
997 %

\print@lineR
\print@eledsectionR

VIII.3 Line and page number computation

\getline@numR The \getline@numR macro determines the page and line numbers for the right text line we are about to send to the vertical list. The \getline@numL is the same for left text.

```

998 \newcommand*{\getline@numR}{%
999   \global\advance\absline@numR \@ne
1000   \do@actionsR
1001   \do@ballastR
1002   \ifledgroupnotesR@ \else
1003     \ifnumberline
1004       \ifsublines@
1005         \ifnum\sub@lockR<\tw@
1006           \global\advance\subline@numR \@ne
1007         \fi
1008       \else
1009         \ifnum\@lockR<\tw@
1010           \global\advance\line@numR \@ne
1011           \global\subline@numR \z@
1012         \fi
1013       \fi
1014     \fi
1015   \fi
1016 }
1017 \newcommand*{\getline@numL}{%
1018   \global\advance\absline@num \@ne
1019   \do@actions
1020   \do@ballast
1021   \ifledgroupnotesL@ \else
1022     \ifnumberline
1023       \ifsublines@
1024         \ifnum\sub@lock<\tw@
1025           \global\advance\subline@num \@ne
1026         \fi
1027       \else
1028         \ifnum\@lock<\tw@
1029           \global\advance\line@num \@ne
1030           \global\subline@num \z@
1031         \fi
1032       \fi
1033     \fi
1034   \fi
1035 }
1036
```

```
1037
1038 %
```

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

```
1039 \newcommand*{\do@ballastR}{\global\ballast@count=\z@
1040 \begingroup
1041   \advance\absline@numR \@ne
1042   \ifnum\next@actionlineR=\absline@numR
1043     \ifnum\next@actionR>-1001
1044       \global\advance\ballast@count by -\c@ballast
1045     \fi
1046   \fi
1047 \endgroup}
1048 %
```

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

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

```
1049
1050 \newif\ifl@dskipversenumberR
1051 \newcommand*{\do@actions@fixedcodeR}{%
1052   \ifcase\l@dttempcnta%
1053     \or% % 1001
1054       \global\sublines@true
1055     \or% % 1002
1056       \global\sublines@false
1057     \or% % 1003
1058       \global\@lockR=\@ne
1059     \or% % 1004%
1060       \ifnum\@lockR=\tw@
1061         \global\@lockR=\thr@@
1062       \else
1063         \global\@lockR=\z@
1064       \fi
1065     \or% % 1005
1066       \global\sub@lockR=\@ne
1067     \or% % 1006
1068       \ifnum\sub@lockR=\tw@
1069         \global\sub@lockR=\thr@@
1070       \else
1071         \global\sub@lockR=\z@
1072       \fi
1073     \or% % 1007
1074       \l@dskipnumbertrue
1075     \or% % 1008
1076       \l@dskipversenumberRtrue%
```

```

1077 \or% % 1009
1078 \l@dhidnumbertrue%
1079 \else%
1080 \led@warn@BadAction
1081 \fi%
1082 }
1083
1084
1085 \newcommand*{\do@actionsR}{%
1086 \global\let\do@actions@nextR=\relax
1087 \@l@dttempcntb=\absline@numR
1088 \ifnum\@l@dttempcntb<\next@actionlineR\else
1089 \ifnum\next@actionR>-1001\relax
1090 \global\page@numR=\next@actionR
1091 \ifbypage@R
1092 \global\line@numR \z@ \global\subline@numR \z@
1093 \fi
1094 \else
1095 \ifnum\next@actionR<-4999\relax % 9/05 added relax
here
1096 \@l@dttempcnta=-\next@actionR
1097 \advance\@l@dttempcnta by -5001\relax
1098 \ifsublines@
1099 \global\subline@numR=\@l@dttempcnta
1100 \else
1101 \global\line@numR=\@l@dttempcnta
1102 \fi
1103 \else
1104 \@l@dttempcnta=-\next@actionR
1105 \advance\@l@dttempcnta by -1000\relax
1106 \do@actions@fixedcodeR
1107 \fi
1108 \fi
1109 \ifx\actionlines@listR\empty
1110 \gdef\next@actionlineR{1000000}%
1111 \else
1112 \gl@p\actionlines@listR\to\next@actionlineR
1113 \gl@p\actions@listR\to\next@actionR
1114 \global\let\do@actions@nextR=\do@actionsR
1115 \fi
1116 \fi
1117 \do@actions@nextR}
1118
1119 %

```

VIII.4 Line number printing

`\l@dcalnum` `\affixline@numR` is the right text version of the `\affixline@num` macro.
`\checksub@l@ckR`
`\check@l@ckR`
`\f@x@l@cksR`
`\affixline@numR`

```

1121 \newcommand*{\l@dcacnum}[3]{%
1122   \ifnum #1 > #2\relax
1123     \@l@dttempcnta = #1\relax
1124     \advance\@l@dttempcnta by -#2\relax
1125     \divide\@l@dttempcnta by #3\relax
1126     \multiply\@l@dttempcnta by #3\relax
1127     \advance\@l@dttempcnta by #2\relax
1128   \else
1129     \@l@dttempcnta=#2\relax
1130   \fi}
1131
1132 \newcommand*{\ch@cksub@lockR}{%
1133   \ifcase\sub@lockR
1134   \or
1135     \ifnum\sublock@disp=\@ne
1136       \@l@dttempcntb \z@ \l@dttempcnta \@ne
1137     \fi
1138   \or
1139     \ifnum\sublock@disp=\tw@
1140     \else
1141       \@l@dttempcntb \z@ \l@dttempcnta \@ne
1142     \fi
1143   \or
1144     \ifnum\sublock@disp=\z@
1145       \@l@dttempcntb \z@ \l@dttempcnta \@ne
1146     \fi
1147   \fi}
1148
1149 \newcommand*{\ch@ck@lockR}{%
1150   \ifcase\@lockR
1151   \or
1152     \ifnum\lock@disp=\@ne
1153       \@l@dttempcntb \z@ \l@dttempcnta \@ne
1154     \fi
1155   \or
1156     \ifnum\lock@disp=\tw@
1157     \else
1158       \@l@dttempcntb \z@ \l@dttempcnta \@ne
1159     \fi
1160   \or
1161     \ifnum\lock@disp=\z@
1162       \@l@dttempcntb \z@ \l@dttempcnta \@ne
1163     \fi
1164   \fi}
1165
1166 \newcommand*{\f@x@locksR}{%
1167   \ifcase\@lockR
1168   \or
1169     \global\@lockR \tw@
1170   \or \or

```

```

1171 \global\@lockR \z@
1172 \fi
1173 \ifcase\sub@lockR
1174 \or
1175 \global\sub@lockR \tw@
1176 \or \or
1177 \global\sub@lockR \z@
1178 \fi}
1179
1180
1181 \newcommand*{\affixline@numR}{%
1182 \ifledgroupnotesR@\else\ifnumberline
1183 \ifl@dskipnumber
1184 \global\l@dskipnumberfalse
1185 \else
1186 \ifsublines@
1187 \@l@dttempcntb=\subline@numR
1188 \l@dcalcnun{\subline@numR}{\c@firstsublinenumR}{\
c@sublinenumincrementR}%
1189 \ch@cksub@lockR
1190 \else
1191 \@l@dttempcntb=\line@numR
1192 \ifx\linenumberlist\empty
1193 \l@dcalcnun{\line@numR}{\c@firstlinenumR}{\
c@linenumincrementR}%
1194 \else
1195 \@l@dttempcnta=\line@numR
1196 \edef\rem@inder{,\linenumberlist,\number\line@numR,}%
1197 \edef\sc@n@list{\def\noexpand\sc@n@list
1198 ###1,\number\@l@dttempcnta,###2|{\def\noexpand\
rem@inder{###2}}}%
1199 \sc@n@list\expandafter\sc@n@list\rem@inder|}%
1200 \ifx\rem@inder\empty\advance\@l@dttempcnta\@ne\fi
1201 \fi
1202 \ch@ck@l@ckR
1203 \fi
1204 \ifnum\@l@dttempcnta=\@l@dttempcntb
1205 \ifl@dskipversenumberR\else
1206 \if@twocolumn
1207 \if@firstcolumn
1208 \gdef\l@dld@ta{\llap{\leftlinenumR}}}%
1209 \else
1210 \gdef\l@drd@ta{\rlap{\rightlinenumR}}}%
1211 \fi
1212 \else
1213 \@l@dttempcntb=\line@marginR
1214 \ifnum\@l@dttempcntb>\@ne
1215 \advance\@l@dttempcntb by\page@numR
1216 \fi
1217 \ifodd\@l@dttempcntb

```



```

1218 \gdef\l@drd@ta{\rlap{\rightlinenumR}}}%
1219 \else
1220 \gdef\l@dld@ta{\llap{\leftlinenumR}}}%
1221 \fi
1222 \fi
1223 \fi
1224 \fi
1225 \f@x@l@cksR
1226 \fi
1227 \fi
1228 \fi}
1229 %

```

VIII.5 Pstart number printing in side

The printing of the pstart number is like in `reledmac`, 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@numL230
\affixpstart@numR231 \newcommand*{\affixpstart@numL}{%
\leftpstartnumR232 \ifsidepstartnum
\rightpstartnumR233 \if@twocolumn
\leftpstartnumL234 \if@firstcolumn
\rightpstartnumL235 \gdef\l@dld@ta{\llap{\leftpstartnumL}}}%
\ifpstartnumR236 \else
1237 \gdef\l@drd@ta{\rlap{\rightpstartnumL}}}%
1238 \fi
1239 \else
1240 \@l@tempcntb=\line@margin
1241 \ifnum\@l@tempcntb>\@ne
1242 \advance\@l@tempcntb \page@num
1243 \fi
1244 \ifodd\@l@tempcntb
1245 \gdef\l@drd@ta{\rlap{\rightpstartnumL}}}%
1246 \else
1247 \gdef\l@dld@ta{\llap{\leftpstartnumL}}}%
1248 \fi
1249 \fi
1250 \fi
1251 }
1252 \newcommand*{\affixpstart@numR}{%
1253 \ifsidepstartnum
1254 \if@twocolumn
1255 \if@firstcolumn

```

```

1256 \gdef\l@dld@ta{\llap{\leftstartnumR}}}%
1257 \else
1258 \gdef\l@drd@ta{\rlap{\rightstartnumR}}}%
1259 \fi
1260 \else
1261 \l@dttempcntb=\line@marginR
1262 \ifnum\l@dttempcntb>\@ne
1263 \advance\l@dttempcntb \page@numR
1264 \fi
1265 \ifodd\l@dttempcntb
1266 \gdef\l@drd@ta{\rlap{\rightstartnumR}}}%
1267 \else
1268 \gdef\l@dld@ta{\llap{\leftstartnumR}}}%
1269 \fi
1270 \fi
1271 \fi
1272 }
1273
1274 \newcommand*{\leftstartnumL}{
1275 \ifpstartnum
1276 \thepstartL
1277 \kern\linenumsep\global\pstartnumfalse\fi
1278 }
1279 \newcommand*{\rightstartnumL}{
1280 \ifpstartnum\kern\linenumsep
1281 \thepstartL
1282 \global\pstartnumfalse\fi
1283 }
1284 \newif\ifpstartnumR
1285 \pstartnumRtrue
1286 \newcommand*{\leftstartnumR}{
1287 \ifpstartnumR
1288 \thepstartR
1289 \kern\linenumsep\global\pstartnumRfalse\fi
1290 }
1291 \newcommand*{\rightstartnumR}{
1292 \ifpstartnumR\kern\linenumsep
1293 \thepstartR
1294 \global\pstartnumRfalse\fi
1295 }
1296 %

```

VIII.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.

```

1297 \list@create{\inserts@listR}
1298 %

```

`\add@insertsR` The right text version.
`\add@inserts@nextR`

```

1299 \newcommand*{\add@insertsR}{%
1300   \global\let\add@inserts@nextR=\relax
1301   \ifx\inserts@listR\empty \else
1302     \ifx\next@insertR\empty
1303       \ifx\insertlines@listR\empty
1304         \global\noteschanged@true
1305         \gdef\next@insertR{100000}%
1306       \else
1307         \gl@p\insertlines@listR\to\next@insertR
1308       \fi
1309     \fi
1310     \ifnum\next@insertR=\absline@numR
1311       \gl@p\inserts@listR\to\@insertR
1312       \@insertR
1313       \global\let\@insertR=\undefined
1314       \global\let\next@insertR=\empty
1315       \global\let\add@inserts@nextR=\add@insertsR
1316     \fi
1317   \fi
1318   \add@inserts@nextR}
1319
1320 %

```

VIII.7 Penalties

`\add@penaltiesL` `\add@penaltiesR` `\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 are 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 does not 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@dttempcnta by \interlinepenalty
\fi
\fi
\ifnum\l@dttempcnta=\z@
  \relax
\else
  \ifnum\l@dttempcnta>-10000
    \penalty\l@dttempcnta
  \else
    \penalty -10000
  \fi
\fi}

```

This is for a single chunk. However, as we are probably dealing with several chunks at a time, the above is not really relevant. Peter Wilson thinks 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, Peter Wilson ends up with the following.

```

1321 \newcommand*{\add@penaltiesL}{}
1322 \newcommand*{\add@penaltiesR}{}
1323
1324 %

```

VIII.8 Printing leftover notes

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

```

1325 \newcommand*{\flush@notesR}{%
1326   \@xloop
1327   \ifx\inserts@listR\empty \else
1328     \gl@p\inserts@listR\to\@insertR
1329     \@insertR
1330     \global\let\@insertR=\undefined
1331   \repeat}
1332
1333 %

```

IX Footnotes

IX.1 Line number printing

The `\printlinesR` 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 V.9 p. 81 of `Eledmac`.

handbook: 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`. Just a reminder of the arguments:

```
\printlinesR #1 | #2 | #3 | #4 | #5 | #6 | #7
\printlinesR start-page | line | subline | end-page | line | sub-
line | font
```

```
1334 \def\printlinesR#1|#2|#3|#4|#5|#6|#7|{\begingroup
1335 \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
1336 \ifl@d@pnum #1\fullstop\fi
1337 \ifledplinenum \linenumr@p{#2}\@Rlineflag\else \symplinenum
\fi
1338 \ifl@d@ssub \fullstop \sublinenumr@p{#3}\fi
1339 \ifl@d@dash \endashchar\fi
1340 \ifl@d@pnum #4\fullstop\fi
1341 \ifl@d@elin \linenumr@p{#5}\@Rlineflag\fi
1342 \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumr@p{#6}\fi
1343 \endgroup}
1344
1345
1346 %
```

IX.2 Footnotes output specific to \Pages

`\print@Xnotes@forpages` The `\Xonlyside` and `\onlysideX` hooks for `\Pages` allow notes to be printed either in left or right pages only. The implementation of such features is delegated to `\print@Xnotes@forpages`, which replaces `\print@Xnotes` inside `\Pages`. Here is how we proceed³:

- If notes are to be printed in both sides, we just proceed the usual way: print the foot starts for the series, then the foot group.
- If notes are to be printed in the left side, we do these prints only for even pages ; if notes are to be printed in the right side, we do these prints only for odd pages.
- However, that is not enough. Because the problem does not only consists in printing notes in any particular page. It is also not to put aside room for notes in the pages where we do not want to print them. To take an example: if some note in the left side is too long by 160pt to be printed in full in the left page, we do not want to put aside 160pt a space for it in the following right page.
- To solve this problem, we change the magnification factor associated with notes before going to the next page. If we start a page where no notes are supposed to be printed, the magnification counter is set to 0. We also set the note skip to 0pt. Before starting a new page where these notes are supposed to be printed, we reset

³See <http://tex.stackexchange.com/a/230332/7712>.

these counter and skip to their default values. (About these counter and skip, read *The TeXbook* p. 122-125).

- There still remains a last problem. This problem is quite complex to understand, so an example will speak for itself. Suppose we allow 10 lines of notes by page. Suppose a long note, be it 25 lines, which needs three pages to be printed. Suppose it must be printed only on left pages, namely odd pages.

On p. 2, the first 10 lines of the notes are printed. On p. 3, the box associated to the notes contains 10 lines. However, as we are in a right page, we do not void this box. So \TeX will keep its content for the pages to come. However, on p. 4 it will also add one line in the footnote box, because in any case, \TeX adds some content in the box when preparing the output routines, even if there is some content left in this box from the previous pages. So the lines in the note box at p. 4 will be $10 + 1 = 11$. There is one line which should not be there. Furthermore, as the box size is for 10 lines and not for 11 lines, this last line will be glued to the previous one.

To fix this double issue:

- For the pages where notes must be NOT printed, we allow to every note box one line less than it ought to be. In our example, that means that we allow \TeX to add only $10 - 1 = 9$ line in the note box on p. 3. Before shifting to the pages where notes must be printed, we allow to every notes the expected number of lines. In our example, that means that we allow \TeX to add 10 lines in the note box on p. 4. As on p. 3 only 9 lines were allowed, that means note box of p. 4 will contain $9 + 1 = 10$ lines. So the “one line too many” problem is solved.
- Still remains the “glue” problem. We solve it by recreating a clean note box. We split the one which is created by \TeX to get the next line printed. Then, we create the new box, by bringing together the first part and the last part of the split box, adding some skip between them. That is achieved by `\correct@Xfootins@box` (or `\correct@footinsX@box` for familiar notes).

The code to print critical notes, when processing `\Pages`.

```
1347 \newcommand\print@Xnotes@forpages[1]{%
1348 %
```

First case: notes are for both sides. Just print the note start and the note group

```
1349 \ifcseempty{Xonlyside@#1}{%
1350 \csuse{#1footstart}{#1}%
1351 \csuse{#1footgroup}{#1}%
1352 }%
1353 %
```

Second case: notes are for one side only. First test if we are in a page where they must be printed.

```

1354 {%
1355   \ifbool{expr{%
1356     ((test {\ifcsstring{Xonlyside@#1}{L}} and not test{\
ifnumodd{\c@page}}))%
1357     or%
1358     (test {\ifcsstring{Xonlyside@#1}{R}} and test{\
ifnumodd{\c@page}}))}%
1359   }%
1360 %

```

If we are in a page where notes must be printed, print the notes, after having made the corrections which are needed for boxes.

```

1361 {%
1362   \correct@Xfootins@box{#1}%
1363   \csuse{#1footstart}{#1}%
1364   \csuse{#1footgroup}{#1}%
1365 %

```

Then, say not to keep room for notes in the next page.

```

1366   \global\count\csuse{#1footins}=0%
1367   \global\skip\csuse{#1footins}=0pt%
1368 %

```

And also, allow one line less for notes in the next page.

```

1369   \csuse{Xnotefontsize@#1}%
1370   \global\advance\dimen\csuse{#1footins} by -\
baselineskip%
1371 %

```

Now we have printed the notes. So we put aside this fact.

```

1372   \global\boolfalse{keepforXside@#1}%
1373 }%
1374 %

```

In case we are on a page where notes must NOT be printed. First, memorize that we have not printed the notes, despite having some to print.

```

1375 {%
1376   \global\booltrue{keepforXside@#1}%
1377 %

```

Then restore expected rooms for notes on the next page.

```

1378   \global\count\csuse{#1footins}=\csuse{default@#1
footins}%
1379   \global\skip\csuse{#1footins}=\csuse{Xbeforenotes@
#1}%
1380 %

```

Last but not least, restore the normal line number allowed to notes for the following page.

```

1381         \bgroup%
1382             \csuse{Xnotefontsize@#1}%
1383             \global\advance\dimen\csuse{#1footins} by \
baselineskip%
1384         \egroup%
1385 %

```

```

1386 % End of \protect\cs{print@Xnotes@forpages}.
1387     }%
1388 }%
1389 }%
1390 %

```

Now, `\correct@Xfootins@box`, to fix problem of last line being glued to the previous one.

```

1391 \newcommand{\correct@Xfootins@box}[1]{%
1392 %

```

We need to make correction only in case we have not printed any note in the previous page, although there was to be “normally” printed.

```

1393     \ifbool{keepforXside@#1}{%
1394 %

```

Some setting needed to do the right splitting.

```

1395         \csuse{Xnotefontsize@#1}%
1396         \splittopskip=0pt%
1397 %

```

And now, split the last line, and push in the right place.

```

1398         \global\setbox\csuse{#1footins}=\vbox{%
1399             \vsplit\csuse{#1footins} to \dimexpr\ht\csuse{#1
footins}-1pt\relax%
1400             \vskip \dimexpr-0.5\baselineskip-0.5\lineskip-0.5pt\
relax%
1401             \unvbox\csuse{#1footins}%
1402         }%
1403 %

```

End of the macro.

```

1404     }{}%
1405 }%
1406 %

```

And now, the same for familiar footnotes.

```

1407 \newcommand\print@notesX@forpages[1]{%
1408     \ifcempty{onlysideX@#1}{%
1409         \csuse{footstart#1}{#1}%
1410         \csuse{footgroup#1}{#1}%
1411     }%

```



```

1412 {%
1413   \ifboolexpr{%
1414     ((test {\ifcsstring{onlysideX@#1}{L}} and not test{\
ifnumodd{\c@page}}))%
1415     or%
1416     (test {\ifcsstring{onlysideX@#1}{R}} and test{\
ifnumodd{\c@page}}))%
1417   }%
1418   {%
1419     \correct@footinsX@box{#1}%
1420     \csuse{footstart#1}{#1}%
1421     \csuse{footgroup#1}{#1}%
1422     \global\count\csuse{footins#1}=0%
1423     \global\skip\csuse{footins#1}=0pt%
1424     \csuse{notefontsizeX@#1}%
1425     \global\advance\dimen\csuse{footins#1} by -\
baselineskip%
1426     \global\boolfalse{keepforsideX@#1}%
1427   }%
1428   {%
1429     \global\booltrue{keepforsideX@#1}%
1430     \global\count\csuse{footins#1}=\csuse{
default@footins#1}%
1431     \global\skip\csuse{footins#1}=\csuse{beforenotesX@
#1}%
1432     \bgroup%
1433       \csuse{notefontsizeX@#1}%
1434       \global\advance\dimen\csuse{footins#1} by \
baselineskip%
1435     \egroup%
1436   }%
1437 }%
1438 }%
1439 \newcommand{\correct@footinsX@box}[1]{%
1440   \ifbool{keepforsideX@#1}{%
1441     \csuse{notefontsizeX@#1}%
1442     \splittopskip=0pt%
1443     \global\setbox\csuse{footins#1}=\vbox{%
1444       \vsplit\csuse{footins#1} to \dimexpr\ht\csuse{footins
#1}-1pt\relax%
1445       \vskip \dimexpr-0.5\baselineskip-0.5\lineskip-0.5pt\
relax%
1446       \unvbox\csuse{footins#1}%
1447     }%
1448   }{}%
1449 }%
1450 %

```

X 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.

```
1451 \list@create{\labelref@listR}
1452
1453 %
```

`\edlabel` This command is defined only one time in `reledmac`, including features for `reledpar`.

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

```
1454 \def\l@dmake@labelsR#1|#2|#3|#4|#5{%
1455   \expandafter\ifx\csname the@label#5\endcsname \relax\else
1456     \led@warn@DuplicateLabel{#4}%
1457   \fi
1458   \expandafter\gdef\csname the@label#5\endcsname{#1|#2\
1459     @Rlineflag|#3|#4}%
1459   \ignorespaces}
1460 \AtBeginDocument{%
1461   \def\l@dmake@labelsR#1|#2|#3|#4|#5{%
1462   }
1463
1464 %
```

`\@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`, `\@nl`, and the `\sub@on` and `\sub@off` commands appearing in the `\linenum@out` file.

It is defined on `reledmac`.

XI Side notes

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

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

`\sidenotemargin*`

```
1465 \WithSuffix\newcommand\sidenotemargin*[1]{%
1466   \l@dgetsidenote@margin{#1}
1467   \global\sidenote@marginR=\@l@tempcntb
1468   \global\sidenote@margin=\@l@tempcntb
1469 }
1470 \newcount\sidenote@marginR
1471 \global\sidenote@margin=\@ne
1472
1473 %
```

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

```

1474 \newcommand*{\affixside@noteR}{%
1475   \def\sidenotecontent@{}%
1476   \numgdef{\itemcount@}{0}%
1477   \def\do##1{%
1478     \ifnumequal{\itemcount@}{0}%
1479       {%
1480         \appto\sidenotecontent@{##1}% Not print not
separator before the 1st note
1481         {\appto\sidenotecontent@{\sidenotesep ##1}%
1482          }%
1483         \numgdef{\itemcount@}{\itemcount@+1}%
1484       }%
1485     \dolistloop{\l@dcstotetext}%
1486     \ifnumgreater{\itemcount@}{1}{\led@err@ManySidenotes}{}%
1487     \gdef\@templ@d{%
1488       \gdef\@templ@n{\l@dcstotetext\l@dcstotetext@l\
l@dcstotetext@r}%
1489     \ifx\@templ@d\@templ@n \else%
1490       \if@twocolumn%
1491         \if@firstcolumn%
1492           \setl@dlp@rbox{##1}{\sidenotecontent@}%
1493         \else%
1494           \setl@drp@rbox{\sidenotecontent@}%
1495         \fi%
1496       \else%
1497         \@l@tempcntb=\sidenote@marginR%
1498         \ifnum\@l@tempcntb>\@ne%
1499           \advance\@l@tempcntb by\page@numR%
1500         \fi%
1501         \ifodd\@l@tempcntb%
1502           \setl@drp@rbox{\sidenotecontent@}%
1503           \gdef\sidenotecontent@{}%
1504           \numdef{\itemcount@}{0}%
1505           \dolistloop{\l@dcstotetext@l}%
1506           \ifnumgreater{\itemcount@}{1}{\led@err@ManyLeftnotes
}}%
1507           \setl@dlp@rbox{\sidenotecontent@}%
1508         \else%
1509           \setl@dlp@rbox{\sidenotecontent@}%
1510           \gdef\sidenotecontent@{}%
1511           \numdef{\itemcount@}{0}%
1512           \dolistloop{\l@dcstotetext@r}%
1513           \ifnumgreater{\itemcount@}{1}{\led@err@ManyRightnotes
}}%
1514           \setl@drp@rbox{\sidenotecontent@}%
1515         \fi%
1516       \fi%
1517     \fi%

```

```

1518 }
1519
1520 %

```

XII Familiar footnotes

`\l@dbfnote` `\l@dbfnote` adds the footnote to the insert list, and `\vl@dbfnote` calls the original `\@footnotetext`. There are both defined in `reledmac`.

`\normalbfnoteX`

XIII Verse

Like in `reledmac`, the insertion of `hangingsymbol` is base on `\ifinserthangingsymbol`, and, for the right side, on `\ifinserthangingsymbolR`. Both commands also include the hanging space, to be sure the `\one@line` of hanging lines has the same width that the `\one@line` of normal lines and to prevent the column separator from shifting.

```

\inserthangingsymbolL621 \newif\ifinserthangingsymbolR
\inserthangingsymbolR622 \newcommand{\inserthangingsymbolL}{%
1523   \ifinserthangingsymbol%
1524   \ifinstanzaL%
1525       \hskip \@ifundefined{sza@0@}{0}{\expandafter%
1526       \noexpand\csname  sza@0@\endcsname}\stanzaindentbase
%
1527   \@hangingsymbol%
1528   \fi%
1529   \fi%
1530 }%
1531 \newcommand{\inserthangingsymbolR}{%
1532   \ifinserthangingsymbolR%
1533   \ifinstanzaR%
1534       \hskip \@ifundefined{sza@0@}{0}{\expandafter%
1535       \noexpand\csname  sza@0@\endcsname}\
stanzaindentbase%
1536   \@hangingsymbol%
1537   \fi%
1538   \fi%
1539 }%
1540 %

```

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.

```

1541 \chardef\next=\catcode`\&
1542 \catcode`\&=\active

```

```

1543
1544 %
    astanza This is roughly an environmental form of \stanza, which treats its
stanza-like contents as a single chunk.
1545 \newenvironment{astanza}[1][]{%
1546 \catcode`\&\active
1547 \global\stanza@count\@ne\stanza@modulo\@ne
1548 \ifnum\usernamecount{sza@0@}=\z@
1549 \let\stanza@hang\relax
1550 \let\endlock\relax
1551 \else
1552 \rightskip\z@ plus 1fil\relax
1553 \fi
1554 \ifnum\usernamecount{szp@0@}=\z@
1555 \let\sza@penalty\relax
1556 \fi
1557 \def&{%
1558 \endlock\mbox{}%
1559 \sza@penalty
1560 \global\advance\stanza@count\@ne
1561 \@astanza@line}%
1562 \def\&\{\@stopastanza}%
1563 \pstart[#1]%
1564 \@astanza@line
1565 }{}
1566
1567 %

```

\@stopastanza This command is called by `\&` in `astanza` environment. It allows optional arguments.

```

1568 \newcommandx{\@stopastanza}[1][1,usedefault]{%
1569 \endlock\mbox{}%
1570 \pend[#1]%
1571 }%
1572 %

```

\@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.

```

1573 \newcommand*\@astanza@line{%
1574 \ifnum\value{stanzaindentrepetition}=0
1575 \parindent=\csname sza@\number\stanza@count
1576 @\endcsname\stanzaindentbase
1577 \else
1578 \parindent=\csname sza@\number\stanza@modulo
1579 @\endcsname\stanzaindentbase
1580 \managestanza@modulo
1581 \fi
1582 \par

```

```

1583 \stanza@hang%\mbox{}%
1584 \ignorespaces}
1585
1586 %

```

Lastly reset the modified category codes.

```

1587 \catcode`\&=\next
1588
1589 %

```

`\thestanzaL` And now, the left and right stanza counter.

```

\thestanzaR
1590 \newcounter{stanzaL}
1591 \newcounter{stanzaR}
1592 \renewcommand{\thestanzaL}{%
1593   \textbf{\arabic{stanzaL}}%
1594 }
1595 \renewcommand{\thestanzaR}{%
1596   \textbf{\arabic{stanzaR}}%
1597 }
1598 %
1599 %

```

XIV Naming macros

The \TeX 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’.

```

\setnamebox
\unhnamebox
\unvnamebox
\namebox
1600 \providecommand*\newnamebox}[1]{%
1601   \expandafter\newbox\csname #1\endcsname}
1602 \providecommand*\setnamebox}[1]{%
1603   \expandafter\setbox\csname #1\endcsname}
1604 \providecommand*\unhnamebox}[1]{%
1605   \expandafter\unhbox\csname #1\endcsname}
1606 \providecommand*\unvnamebox}[1]{%
1607   \expandafter\unvbox\csname #1\endcsname}
1608 \providecommand*\namebox}[1]{%
1609   \csname #1\endcsname}
1610
1611 %

```

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

`\usenamecount`

```

1612 \providecommand*{\newnamecount}[1]{%
1613   \expandafter\newcount\csname #1\endcsname}
1614 \providecommand*{\usenamecount}[1]{%
1615   \csname #1\endcsname}
1616
1617 %

```

XV Fixing babel and polyglossia

With parallel texts there is the possibility that the two sides might use different languages via `babel`. On the other hand, nor `babel` nor `polyglossia` 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`
`\l@dusedbabeltrue`

```

1618 \newif\ifl@dusedbabel
1619 %

```

`\l@dchecklang`

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

```

1620 \patchcmd{\bbl@set@language}%
1621   {\select@language{\language name}}%
1622   {\edef\language name{#1}\select@language{\language name}}%
1623   {}%
1624   {}%
1625
1626 %

```

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

`\select language` `\select language` is a `babel` command. `\theled languageL` and `\theled languageR` are the names of the languages of the left and right texts. `\l@duselanguage` is similar to `\select language`.
`\l@duselanguage`
`\theled languageL`
`\theled languageR`

```

1627 \newcommand*{\l@duselanguage}[1]{}
1628 \gdef\theledlanguageL{}
1629 \gdef\theledlanguageR{}
1630
1631 %

```

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

```

1632 \AtBeginDocument{%
1633   \ifundefined{xpg@main@language}{%
1634     \ifundefined{bbl@main@language}{%
1635 %

```

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

```

1636   \l@dusedbabelfalse
1637 }{%
1638 %

```

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.

```

1639   \l@dusedbabeltrue
1640   \let\l@doldselectlanguage\selectlanguage
1641   \let\l@doldbbl@set@language\bbl@set@language
1642   \renewcommand{\selectlanguage}[1]{%
1643     \l@doldselectlanguage{#1}%
1644     \ifledRcol \gdef\theledlanguageR{#1}%
1645     \else      \gdef\theledlanguageL{#1}%
1646     \fi}
1647 %

```

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

```

1648   \renewcommand*{\l@duselanguage}[1]{%
1649     \l@doldselectlanguage{#1}}
1650 %

```

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

```

1651   \gdef\theledlanguageL{\bbl@main@language}%
1652   \gdef\theledlanguageR{\bbl@main@language}%
1653 }%
1654 }
1655 %

```

If use polyglossia

```

1656 { \let\old@otherlanguage\otherlanguage%
1657   \renewcommand{\otherlanguage}[2][1]{%
1658     \selectlanguage[#1]{#2}%
1659     \ifledRcol \gdef\theledlanguageR{#2}%
1660     \else      \gdef\theledlanguageL{#2}%

```



```

1661 \fi}%
1662 \let\l@duselanguage\select@language%
1663 \gdef\theledlanguageL{\xpg@main@language}%
1664 \gdef\theledlanguageR{\xpg@main@language}%
1665 %

```

That is it.

```

1666 }}
1667 %

```

XVI Counts and boxes for parallel texts

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

`\maxchunks` The maximum number of chunk pairs before printing has to be called for. The default is 5120 chunk pairs.

```

1668 \newcount\l@dc@maxchunks
1669 \newcommand{\maxchunks}[1]{\l@dc@maxchunks=#1}
1670 \maxchunks{5120}
1671
1672 %

```

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

`\l@dnumpstartsR`

```

1673 \newcount\l@dnumpstartsR
1674
1675 %

```

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

`\l@psscR`

```

1676 \newcount\l@dpsscL
1677 \newcount\l@dpsscR
1678
1679 %

```

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

```

1680 \newcommand*\l@dsetuprawboxes{%
1681 \l@dtempcntb=\l@dc@maxchunks
1682 \loop\ifnum\l@dtempcntb>\z@
1683 \newnamebox\l@dLcolrawbox\the\l@dtempcntb}
1684 \newnamebox\l@dRcolrawbox\the\l@dtempcntb}

```

```

1685 \advance\@l@tempcntb \m@ne
1686 \repeat}
1687
1688 %

```

`\l@dssetupmaxlinecounts` To be able to synchronise left and right texts we need to know the maximum number of text lines there are in each pair of chunks. `\l@dssetupmaxlinecounts` creates `\maxchunks` new counts called `\l@dsmaxlinesinpar1`, etc., and `\l@dszeromaxlinecounts` zeroes all of them.

```

1689 \newcommand*{\l@dssetupmaxlinecounts}{%
1690 \@l@tempcntb=\l@dc@maxchunks
1691 \loop\ifnum\@l@tempcntb>\z@
1692 \newnamecount{l@dsmaxlinesinpar\the\@l@tempcntb}
1693 \advance\@l@tempcntb \m@ne
1694 \repeat}
1695 \newcommand*{\l@dszeromaxlinecounts}{%
1696 \begin{group}
1697 \@l@tempcntb=\l@dc@maxchunks
1698 \loop\ifnum\@l@tempcntb>\z@
1699 \global\usenamecount{l@dsmaxlinesinpar\the\@l@tempcntb}=\z@
1700 \advance\@l@tempcntb \m@ne
1701 \repeat
1702 \end{group}}
1703
1704 %

```

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

```

1705 \AtBeginDocument{%
1706 \l@dssetuprawboxes
1707 \l@dssetupmaxlinecounts
1708 \l@dszeromaxlinecounts
1709 \l@dnumppstartsL=\z@
1710 \l@dnumppstartsR=\z@
1711 \l@dpscl=\z@
1712 \l@dpscr=\z@}
1713
1714 %

```

XVII Checking text to be processed

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

`\@pstartstrue`

`\@pstartsfalse` ¹⁷¹⁵ `\newif\if@pstarts`

`\check@pstarts` ¹⁷¹⁶ `\newcommand*{\check@pstarts}{%`

```

1717 \@pstartsfalse
1718 \ifnum\l@dnumpstartsL>\l@dpscL
1719   \@startstrue
1720 \else
1721   \ifnum\l@dnumpstartsR>\l@dpscR
1722     \@startstrue
1723   \fi
1724 \fi
1725 }
1726
1727 %

```

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

```

\checkraw@text
1728 \newif\ifaraw@text
1729 \newcommand*{\checkraw@text}{%
1730   \araw@textfalse
1731   \ifvbox\namebox{ldLcolrawbox\the\l@dpscL}
1732     \araw@texttrue
1733   \else
1734     \ifvbox\namebox{ldRcolrawbox\the\l@dpscR}
1735       \araw@texttrue
1736     \fi
1737   \fi
1738 }
1739
1740 %

```

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

```

1741 \newcommand*{\@writelinesinparL}{%
1742   \edef\next{%
1743     \write\linenum@out{\string\@pend[\the\@donereallinesL]}}%
1744   \next
1745   \global\@donereallinesL \z@}
1746 \newcommand*{\@writelinesinparR}{%
1747   \edef\next{%
1748     \write\linenum@outR{\string\@pendR[\the\@donereallinesR
1749   ]}}%
1750   \next
1751   \global\@donereallinesR \z@}
1752 %

```

XVIII Parallel columns

`\@eledsectionL` The parbox `\@eledsectionL` and `\@eledsectionR` will keep the sections' title.

```
\@eledsectionR
1753 \newsavebox{\@eledsectionL}%
1754 \newsavebox{\@eledsectionR}%
1755 %
```

`\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.

```
1756 \newcommand*{\Columns}{%
1757   \ifl@dpairing%
1758   \led@err@Columns@InsideEnv%
1759   \fi%
1760   \l@dprintingcolumnstrue%
1761   \eledsection@correcting@skip=-\baselineskip% Correction for
sections' titles
1762   \ifnum\l@dnumpstartsL=\l@dnumpstartsR\else
1763     \led@err@BadLeftRightPstarts{\the\l@dnumpstartsL}{\the\l@dnumpstartsR}%
1764   \fi
1765 %
```

Start a group and zero counters, etc.

```
1766 \begingroup
1767   \l@dzeropenalties
1768   \endgraf\global\num@lines=\prevgraf
1769   \global\num@linesR=\prevgraf
1770   \global\par@line=\z@
1771   \global\par@lineR=\z@
1772   \global\l@dpscL=\z@
1773   \global\l@dpscR=\z@
1774 %
```

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

```
1775   \check@pstarts
1776   \loop\if@pstarts
1777     \global\pstartnumtrue
1778     \global\pstartnumRtrue
1779 %
```

Increment `\l@dpscL` and `\l@dpscR` which here count the numbers of left and right chunks. Also restore the value of the public pstart counters.

```
1780   \global\advance\l@dpscL \@ne
1781   \global\advance\l@dpscR \@ne
1782   \restore@pstartL@pc%
1783   \restore@pstartR@pc%
1784 %
```

We print the optional argument of `\pstart` or the argument of `\AtEveryPstart`.

```
1785 \Columns@print@before@pstart%
1786 %
```

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

```
1787 \checkraw@text
1788 { \loop\ifaraw@text
1789 %
```

Grab the next pair of left and right text lines and output them, swapping languages if they differ, adding section title if needed.

```
1790 \l@duselanguage{\theledlanguageL}%
1791 \do@lineL
1792 \xifinlist{\the\l@dpscL}{\eled@sections@@}
1793 {%
1794 \ifdefstring{\@eledsectmark}{L}%
1795 {\csuse{eled@sectmark@the\l@dpscL}%
1796 }{}%
1797 \global\csundef{eled@sectmark@the\l@dpscL}%
1798 \savebox{\@eledsectionL}{\parbox[t][t]{\
Lcolwidth}{\vbox}{\print@eledsectionL}}}%\vbox{}-> prevent
alignment troubles with RTL language
1799 }%
1800 {}%
1801 \l@duselanguage{\theledlanguageR}%
1802 \do@lineR
1803 \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}
1804 {%
1805 \ifdefstring{\@eledsectmark}{R}%
1806 {\csuse{eled@sectmark@the\l@dpscR}%
1807 }{}%
1808 \global\csundef{eled@sectmark@the\l@dpscR}%
1809 %
\savebox{\@eledsectionR}{\parbox[t][t]{\
Rcolwidth}{\vbox}{\print@eledsectionR}}}%\vbox{}-> prevent
alignment troubles with RTL language
1810 }%
1811 \hb@xt@ \hsize{%
1812 \ifdefstring{\columns@position}{L}{}{\hfill}%
1813 \unhbox\l@dleftbox%
1814 \ifhbox{\@eledsectionL}%
1815 \usebox{\@eledsectionL}%
1816 \fi%
1817 \print@columnseparator%
1818 \unhbox\l@drightbox%
1819 \ifhbox{\@eledsectionR}%
1820 \usebox{\@eledsectionR}%
1821 \fi%
```

```

1822         \ifdefstring{\columns@position}{R}{\hfill}%
1823     }%
1824     \checkraw@text
1825     \checkverseL
1826     \checkverseR
1827     \checkpb@columns
1828     \repeat}
1829 %

```

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 is by `pstart`.

```

1830     \@writelinesinparL
1831     \@writelinesinparR
1832     \check@pstarts
1833     \ifbypstart@%
1834         \write\linenum@out{\string\@set[1]}
1835         \resetprevline@
1836     \fi
1837     \ifbypstart@R
1838         \write\linenum@outR{\string\@set[1]}
1839         \resetprevline@
1840     \fi
1841     \Columns@print@after@pend%
1842     \repeat
1843 %

```

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.

```

1844     \flush@notes
1845     \flush@notesR
1846     \endgroup
1847 %

1848     \global\l@dpscL=\z@
1849     \global\l@dpscR=\z@
1850     \global\l@dnumpstartsL=\z@
1851     \global\l@dnumpstartsR=\z@
1852     \l@dprintingcolumnsfalse%
1853     \ignorespaces
1854     \global\instanzaLfalse
1855     \global\instanzaRfalse}
1856
1857 %

```

`\print@columnseparator` `\print@columnseparator` prints the column separator, with surrounding spaces (as the user has set them). We use the \TeX `\ifdim` instead of `etoolbox` to avoid having `\hfill` in a `{}`, which deletes some space (but not much).

```

1858 \def\print@columnseparator{%
1859   \ifdim\beforecolumnseparator<0pt%
1860     \hfill%
1861   \else%
1862     \hspace{\beforecolumnseparator}%
1863   \fi%
1864   \columnseparator%
1865   \ifdim\aftercolumnseparator<0pt%
1866     \hfill%
1867   \else%
1868     \hspace{\beforecolumnseparator}%
1869   \fi%
1870 }%
1871 %

```

\checkpb@columns \checkpb@columns prevent or make pagebreaking in columns, depending of the use of \ledpb or \lednopb.

```

1872
1873 \newcommand{\checkpb@columns}{%
1874   \newif\if@pb
1875   \newif\if@nopb
1876   \IfStrEq{\led@pb@setting}{before}{
1877     \numdef{\next@absline}{\the\absline@num+1}%
1878     \numdef{\next@abslineR}{\the\absline@numR+1}%
1879     \xifinlistcs{\next@absline}{l@prev@pb}{\@pbtrue}{}%
1880     \xifinlistcs{\next@abslineR}{l@prev@pbR}{\@pbtrue}{}
1881     \xifinlistcs{\next@absline}{l@prev@nopb}{\@nopbtrue}{}%
1882     \xifinlistcs{\next@abslineR}{l@prev@nopbR}{\@nopbtrue}{}
1883   }{}
1884   \IfStrEq{\led@pb@setting}{after}{
1885     \xifinlistcs{\the\absline@num}{l@prev@pb}{\@pbtrue}{}%
1886     \xifinlistcs{\the\absline@numR}{l@prev@pbR}{\@pbtrue}{}
1887     \xifinlistcs{\the\absline@num}{l@prev@nopb}{\@nopbtrue}{}%
1888     \xifinlistcs{\the\absline@numR}{l@prev@nopbR}{\@nopbtrue}{}
1889   }{}
1890   \if@nopb\nopagebreak[4]\enlargethispage{\baselineskip}\fi
1891   \if@pb\pagebreak[4]\fi
1892 }
1893 %

```

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

\columnrulewidth

```

1894 \newcommand*{\columnseparator}{%
1895   \smash{\rule[-0.2\baselineskip]{\columnrulewidth}{1.05\baselineskip}}}
1896 \newdimen\columnrulewidth

```

```

1897 \columnrulewidth=\z@
1898
1899 %

```

`\columnspan` The position of the `\Columns` in a page. Default value is R. Stored in `\columns@position`.
`\columns@position`

```

1900 \newcommand*{\columnspan}[1]{%
1901 \xdef\columns@position{#1}%
1902 }%
1903 \xdef\columns@position{R}%
1904 %

```

`\beforecolumnseparator` `\aftercolumnseparator` lengths are defined to -1pt. If user changes them to a positive length, the lengths are used to define blank spaces before / after the column separator, instead of `\hfill`.

```

1905 \newlength{\beforecolumnseparator}%
1906 \setlength{\beforecolumnseparator}{-2pt}%
1907
1908 \newlength{\aftercolumnseparator}%
1909 \setlength{\aftercolumnseparator}{-2pt}%
1910
1911 %

```

`setwidthliketwocolumns@L` The `\setwidth...` macros are called in `\beginnumbering` in a **non-parallel** typesetting context, to fix the width of the lines to be vertically aligned with parallel columns. They are also called at the beginning of a note's group, if some options are enabled. The `\setposition...` macros are called in `\beginnumbering` in a **non-parallel** typesetting context to fix the position of the lines. The `\setnoteposition...` macros are called in `\xxxfootstart` in a **non-parallel** typesetting context to fix the position of notes block.

```

1912 \newcommand{\setwidthliketwocolumns@L}{%
1913 % Temporary dimension, initially equal to the standard hsize,
% i.e. text width
1914 % \begin{macrocode}
1915 \newdimen\temp%
1916 \temp=\hsize%
1917 %

```

Hsize : Left + Right width

```

1918 \hsize=\Lcolwidth%
1919 \advance\hsize\Rcolwidth%
1920 %

```

Now, calculating the remaining space

```

1921 \advance\temp-\hsize%
1922 %

```

And multiply the hsize by 2/3 of this space


```

1923 \multiply\temp by 2%
1924 \divide\temp by 3%
1925 \advance\hsize\temp%
1926 }%
1927
1928 \newcommand{\setpositionliketwocolumns@L}{%
1929 \renewcommand{\ledrlfill}{\hfill}%
1930 }%
1931
1932 \newcommand{\setnotespositionliketwocolumns@L}{%
1933 }%
1934
1935
1936 %

1937 \newcommand{\setwidthliketwocolumns@C}{%
1938 % Temporary dimension, initially equal to the standard hsize,
1939 % i.e. text width
1939 %

1940 \newdimen\temp%
1941 \temp=\hsize%
1942 % Hsize : Left + Right width
1943 %

1944 \hsize=\Lcolwidth%
1945 \advance\hsize\Rcolwidth%
1946 % Now, calculating the remaining space
1947 %

1948 \advance\temp-\hsize%
1949 %

And multiply the hsize by 1/2 of this space

1950 \divide\temp by 2%
1951 \advance\hsize\temp%
1952 }%
1953
1954 \newcommand{\setpositionliketwocolumns@C}{%
1955 \doinsidelinehook{\hfill}%
1956 \renewcommand{\ledrlfill}{\hfill}%
1957 }%
1958
1959 \newcommand{\setnotespositionliketwocolumns@C}{%
1960 \newdimen\temp%
1961 \newdimen\tempa%
1962 \temp=\hsize%
1963 \tempa=\Lcolwidth%
1964 \advance\tempa\Rcolwidth%
1965 \advance\temp-\tempa%

```

```

1966 \divide\temp by 2%
1967 \leftskip=\temp%
1968 \rightskip=-\temp%
1969 }%
1970
1971 \newcommand{\setwidthliketwocolumns@R}{%
1972 %

```

Temporary dimension, initially equal to the standard hsize, i.e. text width

```

1973 \newdimen\temp%
1974 \temp=\hsize%
1975 %

```

Hsize : Left + Right width

```

1976 \hsize=\Lcolwidth%
1977 \advance\hsize\Rcolwidth%
1978 %

```

Now, calculating the remaining space

```

1979 \advance\temp-\hsize%
1980 %

```

And multiply the hsize by 2/3 of this space

```

1981 \multiply\temp by 2%
1982 \divide\temp by 3%
1983 \advance\hsize\temp%
1984 }%
1985
1986 \newcommand{\setpositionliketwocolumns@R}{%
1987 \doinsidelinehook{\hfill}%
1988 }%
1989
1990 \newcommand{\setnotespositionliketwocolumns@R}{%
1991 \newdimen\temp%
1992 \newdimen\tempa%
1993 \temp=\hsize%
1994 \tempa=\Lcolwidth%
1995 \advance\tempa\Rcolwidth%
1996 \advance\temp-\tempa%
1997 \divide\temp by 2%
1998 \leftskip=\temp%
1999 \rightskip=-\temp%
2000 }%
2001
2002 %

```

`\Columns@print@before@pstart`
`\Columns@print@after@pend`

The `\Columns@print@before@pstart` and `\Columns@print@after@pend` print the content of the optional argument of `\pstart` / `\pend`. If this content is not empty, it also print the separator.

```

2003 \newcommand{\Columns@print@before@pstart}{%
2004   \ifboolexpr{%
2005     test{\ifcsstring{before@pstartL@the\l@dpscL}{\
2006       at@every@pstart}}%
2007     and test {\ifcsstring{before@pstartR@the\l@dpscR}{\
2008       at@every@pstart}}%
2009     and test {\ifdefempty{\at@every@pstart}}}%
2010     {}%
2011     {%
2012       \hb@xt@ \hsize{%
2013         \ifdefstring{\columns@position}{L}{\hfill }%
2014         \par\parbox[t][t]{\Lcolwidth}{%
2015           \csuse{before@pstartL@the\l@dpscL}%
2016         }%
2017         \print@columnseparator%
2018         \parbox[t][t]{\Rcolwidth}{%
2019           \csuse{before@pstartR@the\l@dpscR}%
2020         }%
2021         \ifdefstring{\columns@position}{R}{\hfill}%
2022       }%
2023     }%
2024   }%
2025 \newcommand{\Columns@print@after@pend}{%
2026   \ifboolexpr{%
2027     test{\ifcsstring{after@pendL@the\l@dpscL}{\at@every@pend
2028     }}%
2029     and test {\ifcsstring{after@pendR@the\l@dpscR}{\
2030       at@every@pend}}%
2031     and test {\ifdefempty{\at@every@pend}}}%
2032     {}%
2033     {%
2034       \hb@xt@ \hsize{%
2035         \ifdefstring{\columns@position}{L}{\hfill }%
2036         \parbox[t][t]{\Lcolwidth}{%
2037           \csuse{after@pendL@the\l@dpscL}%
2038         }%
2039         \print@columnseparator%
2040         \parbox[t][t]{\Rcolwidth}{%
2041           \csuse{after@pendR@the\l@dpscR}%
2042         }%
2043         \ifdefstring{\columns@position}{R}{\hfill}%
2044       }%
2045     }%
2046   }%
2047   \global\csundef{after@pendL@the\l@dpscL}%
2048   \global\csundef{after@pendR@the\l@dpscR}%

```

XIX Parallel pages

This is considerably more complicated than parallel columns.

XIX.1 Specific counters

<code>\numpagelinesL</code>	Counts for the number of lines on a left or right page, and the smaller of the number of
<code>\numpagelinesR</code>	lines on a pair of facing pages.
<code>\l@dminpagelines</code>	
2048	<code>\newcount\numpagelinesL</code>
2049	<code>\newcount\numpagelinesR</code>
2050	<code>\newcount\l@dminpagelines</code>
2051	
2052	<code>%</code>

XIX.2 Main macro

`\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.

```

2053 \newcommand*{\Pages}{%
2054   \ifl@dpairing%
2055   \led@err@Pages@InsideEnv%
2056   \fi%
2057   \eledsection@correcting@skip=-2\baselineskip% line
correcting for section titles.
2058   \parledgroup@notespacing@set@correction%
2059   \typeout{}%
2060   \typeout{***** PAGES
*****}%
2061   \ifnum\l@dnumpstartsL=\l@dnumpstartsR\else%
2062     \led@err@BadLeftRightPstarts{\the\l@dnumpstartsL}{\the\l@dnumpstartsR}%
2063   \fi%
2064   %

```

As `\Pages` must be called outside of the pages environment, we have to redefine the `\Lcolwidth` and `\Rcolwidth` lengths, to prevent false overfull hboxes.

```

2065   \setlength{\Lcolwidth}{\textwidth}%
2066   \setlength{\Rcolwidth}{\textwidth}%
2067   %

```

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

```

2068   \cleartol@evenpage%
2069   \l@dprintingpagestrue%
2070   \begingroup%
2071     \l@dzeropenalties%
2072     \endgraf\global\num@lines=\prevgraf%

```

```

2073 \global\num@linesR=\prevgraf%
2074 \global\par@line=\z@%
2075 \global\par@lineR=\z@%
2076 \global\l@dpscL=\z@%
2077 \global\l@dpscR=\z@%
2078 \writtenlinesLfalse%
2079 \writtenlinesRfalse%
2080 %

```

The footnotes are printed in a different way from expected in `reledmac`, as we may want to print the notes on one side only.

```

2081 \let\print@Xnotes\print@Xnotes@forpages%
2082 \let\print@notesX\print@notesX@forpages%
2083 %

```

Check if there are chunks to be processed.

```

2084 \check@pstarts%
2085 \loop\if@pstarts%
2086 %

```

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

```

2087 \global\advance\l@dpscL \@ne%
2088 \global\advance\l@dpscR \@ne%
2089 %

```

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

```

2090 \getlinesfromparlistL%
2091 \getlinesfromparlistR%
2092 \l@dcalc@maxoftwo{\@cs@linesinparL}{\@cs@linesinparR}%
2093 {\usernamecount{l@dmaxlinesinpar\the\l@dpscL}}%
2094 \check@pstarts%
2095 \repeat%
2096 %

```

Zero the counts again, ready for the next bit.

```

2097 \global\l@dpscL=\z@%
2098 \global\l@dpscR=\z@%
2099 %

```

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

```

2100 \getlinesfrompagelistL%
2101 \getlinesfrompagelistR%
2102 \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
2103 {\l@dminpagelines}%
2104 %

```

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.

```

2105 \check@pstarts%
2106 \if@pstarts%
2107 %

```

Increment the chunk counts to get the first pair. Restore also the value of public pstart counters.

```

2108 \global\advance\l@dpscL \@ne%
2109 \global\advance\l@dpscR \@ne%
2110 \restore@pstartL@pc%
2111 \restore@pstartR@pc%
2112 %

```

We have not processed any lines from these chunks yet, so zero the respective line counts.

```

2113 \global\@donereallinesL=\z@%
2114 \global\@donetotallinesL=\z@%
2115 \global\@donereallinesR=\z@%
2116 \global\@donetotallinesR=\z@%
2117 %

```

Start a loop over the boxes (chunks).

```

2118 \checkraw@text%
2119 %
2120 % \begingroup
2121 { \loop\ifaraw@text%
2122 %

```

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

```

2123 \checkpageL%
2124 \l@duselanguage{\theledlanguageL}%
2125 { \loop\ifl@dsamepage%
2126 %

```

Process the next (left) text line, adding it to the page. Eventually, adds the optional argument of pstart.

```

2127 \ifdefstring{\@eledsectnotoc}{L}{\ledsectnotoc
2128 }{}%
2129 \csuse{before@pstartL@\the\l@dpscL}%
2130 \global\csundef{before@pstartL@\the\l@dpscL}%
2131 \do@lineL%
2132 \xifinlist{\the\l@dpscL}{\eled@sections@@}
2133 {\print@eledsectionL}%
2134 {}%
2135 \advance\numpagelinesL \@ne%

```

When using shiftedpstarts option, a `\l@dlftbox` with a null height is not printed. That means we do not insert blank lines at the end of a left chunk lower than the corresponding right chunk. However, a `\l@dlftbox` with a null height will advance the

\pagetotal in any case. Because if we do not do this, the \checkpageL could let \ifl@pagefull to false, and consequently a \@lopL equal to 1000 could be written in the numbered file, even if all the lines actually needed for the current page have been printed. l@dleftbox

```

2136         \ifshiftedpstarts%
2137             \ifdim\ht\l@dleftbox>0pt%
2138                 \parledgroup@correction@notespacing{L
2139             }%
2140             \hb@xt@ \hsize{\ledstrutL\unhbox\
2141             l@dleftbox}%
2142         \else%
2143             \dimen0=\pagetotal%
2144             \advance\dimen0 by \baselineskip%
2145             \global\pagetotal=\dimen0%
2146         \fi%
2147     \else%
2148         \parledgroup@correction@notespacing{L}%
2149         \hb@xt@ \hsize{\ledstrutL\unhbox\
2150         l@dleftbox}%
2151     \fi%
2152 %

```

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. Check if we have to print the optional argument of the last pend. Check if the page is full. Check if the verse is split in two subsequent pages. Check there is any forced page breaks. Reset the verse skipnumber boolean

```

2150         \get@nextboxL%
2151     \global\l@dskipversenumberfalse%
2152     \ifprint@last@after@pendL%
2153         \csuse{after@pendL@\the\l@dpscL}%
2154     \global\csundef{after@pendL@\the\l@dpscL
2155     }%
2156     \fi%
2157     \checkpageL%
2158     \checkverseL%
2159     \checkpbl%
2160     \repeat%
2161 %

```

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.

```

2161         \ifl@pagefull%
2162             \@writelinesonpageL{\the\numpagelinesL}%
2163         \else%
2164             \@writelinesonpageL{1000}%
2165         \fi%
2166 %

```

Reset to zero the left-page line count, clear the page to get onto the facing (odd, right) page, and reinitialize the accumulated dimension of interline correction for notes in parallel ledgroup.

```

2167 \numpagelinesL \z@%
2168 \parledgroup@correction@notespacing@init%
2169 \clearl@dleftpage }%
2170 %

```

Now do the same for the right text.

```

2171 \checkpageR%
2172 \l@duselanguage{\theledlanguageR}%
2173 {
2174   \loop\ifl@dsamepage%
2175     \initnumbering@sectcountR%
2176     \ifdefstring{\@eledsectnotoc}{R}{\ledsectnotoc
2177   }{}%
2178   \csuse{before@pstartR@\the\l@dpscR}%
2179   \global\csundef{before@pstartR@\the\l@dpscR}%
2180   \do@lineR%
2181   \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}%
2182   {\print@eledsectionR}%
2183   {}%
2184   \advance\numpagelinesR \@ne%
2185   \ifshiftedpstarts%
2186     \ifdim\ht\l@drightrightbox>0pt%
2187     \parledgroup@correction@notespacing{R}%
2188     \hb@xt@ \hsize{\ledstrutR\unhbox\
2189   l@drightrightbox}%
2190   \else%
2191     \dimen0=\pagetotal%
2192     \advance\dimen0 by \baselineskip%
2193     \global\pagetotal=\dimen0%
2194     \fi%
2195   \else%
2196     \parledgroup@correction@notespacing{R}%
2197     \hb@xt@ \hsize{\ledstrutR\unhbox\
2198   l@drightrightbox}%
2199   \fi%
2200   \get@nextboxR%
2201   \global\l@dskipversenumberRfalse%
2202   \ifprint@last@after@pendR%
2203     \csuse{after@pendR@\the\l@dpscR}%
2204     \global\csundef{after@pendR@\the\l@dpscR}
2205   }%
2206   \fi%
2207   \checkpageR%
2208   \checkverseR%
2209   \checkpbR%
2210   \repeat%

```



```

2206         \ifl@dpagfull%
2207             \@writelinesonpageR{\the\numpagelinesR}%
2208         \else%
2209             \@writelinesonpageR{1000}%
2210         \fi%
2211         \numpagelinesR=\z@%
2212         \parledgroup@correction@notespacing@init%
2213     %

```

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

```

2214         \clearl@drighthousepage}%
2215     %

```

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.

```

2216         \checkraw@text%
2217         \ifaraw@text%
2218             \getlinesfrompagelistL%
2219             \getlinesfrompagelistR%
2220             \l@dcalc@minoftwo{\@cs@linesonpageL}{\
2221 @cs@linesonpageR}%
2222                                     {\l@dminpagelines}%
2223         \fi%
2224     \repeat}%
2225 %

```

We have now output the text from all the chunks.

```

2225     \fi%
2226 %

```

Make sure that there are no inserts hanging around.

```

2227     \flush@notes%
2228     \flush@notesR%
2229     \endgroup%
2230 %

```

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

```

2231     \global\l@dpscL=\z@%
2232     \global\l@dpscR=\z@%
2233     \global\l@dnumstartsL=\z@%
2234     \global\l@dnumstartsR=\z@%
2235     \global\instanzaLfalse%
2236     \global\instanzaRfalse%
2237     \l@dprintingpagesfalse%
2238     \finish@Pages@notes%Needed to prevent final notes overlap
2239     line number
2240     \ignorespaces}

```

```
2241
2242 %
```

XIX.3 Ensure all notes be printed at the end of parallel pages

\finish@Pages@notes This macro ensures that all long notes are printed at the end of \Pages typesetting, and that there is no more long notes left for the next pages.

```
2243 \newcommand{\finish@Pages@notes}{%
2244   \def\do##1{%
2245     %
```

First, declare footnote box if there was no previous declared. E.g. if familiar or critical notes were disabled by `reledmac`'s options.

```
2246   \ifnocritical@%
2247     \global\newnamebox{##1footins}
2248   \fi
2249   \ifnofamiliar@%
2250     \global\newnamebox{footins##1}
2251   \fi
2252 %
```

And now, add a `\newpage` if there is no more footnote to print.

```
2253   \ifvoid\csuse{##1footins}%
2254     \ifvoid\csuse{footins##1}\else%
2255       \newpage\null%
2256       \listbreak%
2257     \fi%
2258   \else%
2259     \newpage\null%
2260     \listbreak%
2261   \fi%
2262 }%
2263 \dolistloop{\@series}%
2264 }%
2265 %
```

XIX.4 Struts

\ledstrutL Struts inserted into leftand right text lines.

```
\ledstrutR
2266 \newcommand*{\ledstrutL}{\strut}
2267 \newcommand*{\ledstrutR}{\strut}
2268
2269 %
```

XIX.5 Page clearing

`\cleartoevenpage` and `\cleartol@devenpage` `\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.

```

2270 \providecommand{\cleartoevenpage}[1][\@empty]{%
2271   \clearpage
2272   \ifodd\c@page\hbox{ }#1\clearpage\fi}
2273
2274 \newcommand*{\cleartol@devenpage}{%
2275   \ifdim\pagetotal<\topskip% on an empty page
2276   \else
2277     \clearpage
2278   \fi
2279   \ifodd\c@page%
2280     \ifprevpgnotnumbered%
2281       \addtocounter{par@page}{-1}%
2282       \ifdef{\prevpgstyle}{\thispagestyle{\prevpgstyle}}{%
2283         \fi%
2284       \hbox{ }\clearpage%
2285     \fi%
2286   }%
2287   %

```

`\clearl@dleftpage` and `\clearl@drighthouse` get us onto an odd and even page, respectively, checking that we end up on the subsequent page. Both commands use `\newpage` and not `\clearpage`. Because `\clearpage` prints all footnotes before the next page, even if it has to add new empty pages, while `\newpage` does not. And as we want notes started in the left page continue in the right page and *vice-versa*, we must use `\newpage` and not `\clearpage`.

```

2288 \newcommand*{\clearl@dleftpage}{%
2289   \ifdim\pagetotal=0pt\hbox{ }\fi%
2290   \newpage%
2291   \ifodd\c@page\else
2292     \led@err@LeftOnRightPage
2293     \hbox{ }%
2294     \cleardoublepage
2295   \fi}
2296
2297 \newcommand*{\clearl@drighthouse}{%
2298   \ifdim\pagetotal=0pt\hbox{ }\fi%
2299   \newpage%
2300   \ifodd\c@page
2301     \led@err@RightOnLeftPage
2302     \hbox{ }%
2303     \cleartoevenpage
2304   \fi}
2305

```

2306 %

XIX.6 Lines managing

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

```
2307 \newcommand*{\getlinesfromparlistL}{%
2308   \ifx\linesinpar@listL\empty
2309     \gdef\@cs@linesinparL{0}%
2310   \else
2311     \gl@p\linesinpar@listL\to\@cs@linesinparL
2312   \fi}
2313 \newcommand*{\getlinesfromparlistR}{%
2314   \ifx\linesinpar@listR\empty
2315     \gdef\@cs@linesinparR{0}%
2316   \else
2317     \gl@p\linesinpar@listR\to\@cs@linesinparR
2318   \fi}
2319
2320 %
```

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

```
2321 \newcommand*{\getlinesfrompagelistL}{%
2322   \ifx\linesonpage@listL\empty
2323     \gdef\@cs@linesonpageL{1000}%
2324   \else
2325     \gl@p\linesonpage@listL\to\@cs@linesonpageL
2326   \fi}
2327 \newcommand*{\getlinesfrompagelistR}{%
2328   \ifx\linesonpage@listR\empty
2329     \gdef\@cs@linesonpageR{1000}%
2330   \else
2331     \gl@p\linesonpage@listR\to\@cs@linesonpageR
2332   \fi}
2333
2334 %
```

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

```
2335 \newcommand*{\@writelinesonpageL}[1]{%
2336   \edef\next{\write\linenum@out{\string\@lopL{#1}}}%
2337   \next}
2338 \newcommand*{\@writelinesonpageR}[1]{%
```

```

2339 \edef\next{\write\linenum@outR{\string\@lopR{#1}}}%
2340 \next}
2341
2342 %

```

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

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

```

2343 \newcommand*{\l@dcalc@maxoftwo}[3]{%
2344   \ifnum #2>#1\relax
2345     #3=#2\relax
2346   \else
2347     #3=#1\relax
2348   \fi}
2349 \newcommand*{\l@dcalc@minoftwo}[3]{%
2350   \ifnum #2<#1\relax
2351     #3=#2\relax
2352   \else
2353     #3=#1\relax
2354   \fi}
2355
2356 %

```

XIX.7 Page break managing

`\ifl@dsamepage` `\l@dsamepagetrue` `\l@dsamepagefalse` `\ifl@dpagfull` `\l@dpagfulltrue` `\l@dpagfullfalse` `\checkpageL` `\checkpageR`

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

```

2357 \newif\ifl@dsamepage
2358 \l@dsamepagetrue
2359 \newif\ifl@dpagfull
2360
2361 \newcommand*{\checkpageL}{%
2362   \l@dpagfulltrue
2363   \l@dsamepagetrue
2364   \check@goal
2365   \ifdim\pagetotal<\ledthegoal
2366     \ifnum\numpagelinesL<\l@dminpagelines
2367       \else
2368         \l@dsamepagefalse
2369         \l@dpagfullfalse
2370       \fi
2371     \else

```

```

2372 \l@dsamepagefalse
2373 \l@dpagfulltrue
2374 \fi%
2375 \ifprint@last@after@pendL%
2376 \l@dpagfullfalse%
2377 \l@dsamepagefalse%
2378 \print@last@after@pendLfalse%
2379 \fi%
2380 }%
2381
2382 \newcommand*{\checkpageR}{%
2383 \l@dpagfulltrue
2384 \l@dsamepagetrue
2385 \check@goal
2386 \ifdim\pagetotal<\ledthegoal
2387 \ifnum\numpagelinesR<\l@dmnpagelines
2388 \else
2389 \l@dsamepagefalse
2390 \l@dpagfullfalse
2391 \fi
2392 \else
2393 \l@dsamepagefalse
2394 \l@dpagfulltrue
2395 \fi%
2396 \ifprint@last@after@pendR%
2397 \l@dpagfullfalse%
2398 \l@dsamepagefalse%
2399 \print@last@after@pendRfalse%
2400 \fi%
2401 }%
2402
2403 %

```

\checkpbL \checkpbL and \checkpbR are called after each line is printed, and after the page is checked. These commands correct page breaks depending on \ledpb and \lednopb.

```

2404 \newcommand{\checkpbL}{
2405   \IfStrEq{\led@pb@setting}{after}{
2406     \xifinlistcs{\the\absline@num}{l@prev@pb}{\
2407     l@dpagfulltrue\l@dsamepagefalse}{
2408     \xifinlistcs{\the\absline@num}{l@prev@nopb}{\
2409     l@dpagfullfalse\l@dsamepagetrue}{
2410     }{
2411     \IfStrEq{\led@pb@setting}{before}{
2412       \numdef{\next@absline}{\the\absline@num+1}
2413       \xifinlistcs{\next@absline}{l@prev@pb}{\l@dpagfulltrue\
2414       l@dsamepagefalse}{
2415       \xifinlistcs{\next@absline}{l@prev@nopb}{\
2416       l@dpagfullfalse\l@dsamepagetrue}{

```

```

2413 }{}
2414 }
2415
2416 \newcommand{\checkpbR}{
2417   \IfStrEq{\led@pb@setting}{after}{
2418     \xifinlistcs{\the\absline@numR}{l@prev@pbR}{\
2419     l@dpagfulltrue\l@dsamepagefalse}{}
2420     \xifinlistcs{\the\absline@numR}{l@prev@nopbR}{\
2421     l@dpagfullfalse\l@dsamepagetrue}{}
2422   }{}
2423   \IfStrEq{\led@pb@setting}{before}{
2424     \numdef{\next@abslineR}{\the\absline@numR+1}
2425     \xifinlistcs{\next@abslineR}{l@prev@pbR}{\
2426     l@dpagfulltrue\l@dsamepagefalse}{}
2427     \xifinlistcs{\next@abslineR}{l@prev@nopbR}{\
2428     l@dpagfullfalse\l@dsamepagetrue}{}
2429   }{}
2430 }
2431 %

```

\checkverseL \checkverseL and \checkverseR are called after each line is printed. They
\checkverseR prevent page break inside line of verse.

```

2428 \newcommand{\checkverseL}{
2429   \ifinstanzaL
2430     \iflednopbinverse
2431       \ifinserthangingsymbol
2432         \numgdef{\prev@abslineverse}{\the\absline@num-1}
2433         \IfStrEq{\led@pb@setting}{after}{\lednopbnum{\
2434         prev@abslineverse}}{}
2435         \IfStrEq{\led@pb@setting}{before}{\ifnum\numpagelinesL
2436         <3\ledpbnum{\prev@abslineverse}\fi}{}
2437       \fi
2438     \fi
2439   \fi
2440 }
2441 \newcommand{\checkverseR}{
2442   \ifinstanzaR
2443     \iflednopbinverse
2444       \ifinserthangingsymbolR
2445         \numgdef{\prev@abslineverse}{\the\absline@numR-1}
2446         \IfStrEq{\led@pb@setting}{after}{\lednopbnumR{\
2447         prev@abslineverse}}{}
2448         \IfStrEq{\led@pb@setting}{before}{\ifnum\numpagelinesR
2449         <3\ledpbnumR{\prev@abslineverse}\fi}{}
2450       \fi
2451     \fi
2452   \fi
2453 }
2454 %

```

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

```

2451 \newdimen\ledthegoal
2452 \ifshiftedpstarts
2453     \newcommand*{\@goalfraction}{0.95}
2454 \else
2455     \newcommand*{\@goalfraction}{0.9}
2456 \fi
2457
2458 \newcommand*{\check@goal}{%
2459     \ledthegoal=\@goalfraction\pagegoal}
2460 \newcommand{\setgoalfraction}[1]{%
2461     \xdef\@goalfraction{#1}%
2462 }
2463 %

```

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

```

\ifwritelinesL
2464 \newif\ifwritelinesL
2465 \newif\ifwritelinesR
2466
2467 %

```

XIX.8 Getting boxes content

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

```

2468 \newcommand*{\get@nextboxL}{%
2469     \ifvbox\namebox{1@dLcolrawbox\the\l@dpscL}% box is not
empty
2470 %

```

The current box is not empty; do nothing.

```

2471 \else%
2472 %

```

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

```

2473 \ifnum\usernamecount{1@dmaxlinesinpar\the\l@dpscL}>\
@donetotallinesL
2474     \parledgroup@notes@endL
2475 \else
2476 %

```

Sufficient lines have been output.

```

2477 \ifnum\usernamecount{1@dmaxlinesinpar\the\l@dpscL}=\
@donetotallinesL

```



```

2478     \parledgroup@notes@endL
2479     \fi
2480     \ifwrittenlinesL\else
2481 %

```

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

```

2482     \@writelinesinparL
2483     \writtenlinesLtrue
2484     \fi
2485     \ifnum\l@dnumpsstartsl>\l@dpscl
2486 %

```

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.

```

2487     \writtenlinesLfalse
2488     \ifbypstart@
2489         \global\line@num=0%
2490         \resetprevline@%
2491     \fi
2492 % Add the content of the optional argument of the previous \
protect\cs{pend}.
2493 %     \begin{macrocode}
2494         \csuse{after@pendL@the\l@dpscl}%
2495         \global\csundef{after@pendL@the\l@dpscl}%
2496 %

```

Check the number of lines

```

2497     \l@dcalc@maxoftwo{\the\usernamecount{l@dmaxlinesinpar\
the\l@dpscl}}%
2498                                     {\the\@donetotallinesL}%
2499                                     {\usernamecount{l@dmaxlinesinpar\the\
l@dpscl}}%
2500     \global\@donetotallinesL \z@
2501 %

```

Go to the next pstart

```

2502     \global\advance\l@dpscl \@ne
2503     \global\pstartnumtrue%
2504     \restore@pstartL@pc%
2505 %

```

Add notes of parallel ledgroup.

```

2506     \parledgroup@notes@endL
2507     \parledgroup@correction@notespacing@final{L}
2508     \else
2509 %

```

```

2510     \fi
2511   \fi
2512 \fi}
2513 %

2514 \newcommand*{\get@nextboxR}{%
2515   \ifvbox\namebox{l@dRcolrawbox\the\l@dpscR}% box is not
empty
2516   \else% box is empty
2517     \ifnum\usernamecount{l@dmaxlinesinpar\the\l@dpscR}>\
@donetotallinesR
2518       \parledgroup@notes@endR
2519     \else
2520       \ifnum\usernamecount{l@dmaxlinesinpar\the\l@dpscR}=\
@donetotallinesR
2521         \parledgroup@notes@endR
2522       \fi
2523       \ifwrittenlinesR\else
2524         \@writelinesinparR
2525         \writtenlinesRtrue
2526       \fi
2527       \ifnum\l@dnumpstartsR>\l@dpscR
2528         \writtenlinesRfalse
2529         \ifbypstart@R
2530           \global\line@numR=0%
2531           \resetprevline@%
2532         \fi
2533         \csuse{after@pendR@\the\l@dpscR}%
2534         \global\csundef{after@pendR@\the\l@dpscR}%
2535         \l@dcalc@maxoftwo{\the\usernamecount{l@dmaxlinesinpar\
the\l@dpscR}}%
2536                                     {\the\@donetotallinesR}%
2537                                     {\usernamecount{l@dmaxlinesinpar\the\
l@dpscR}}}%
2538         \global\@donetotallinesR \z@
2539         \global\advance\l@dpscR \@ne
2540         \global\pstartnumRtrue%
2541         \restore@pstartR@pc%
2542         \parledgroup@notes@endR
2543         \parledgroup@correction@notes@spacing@final{R}
2544       \else
2545         \print@last@after@pendRtrue%
2546       \fi
2547     \fi
2548 \fi}
2549 %
2550 %

```

XX Page numbering

The `sameparallelpagnumber` option allows the same page number on both left and right side. The `prevpgnotnumbered` option allows an empty (not numbered) right-side page before `\Pages`.

We cannot implement these two options by changing the value of the `page` counter, since its value is used by many \LaTeX features to determine whether a page is left (even-numbered) or right (odd-numbered). Consequently, we have to do it by patching `\thepage`, in order to use the value of the `par@page` counter instead of value of `page` counter.

This counter will be increased in a patched version of the \LaTeX 's `\@outputpage` macro, as is the `page` counter in this macro. However, this increase will take account of the options.

`\par@patch@thepage` `\par@patch@thepage` patches `\thepage` in order to use the value of `par@page` counter and not the value of `page`. It must be called after any redefinition of `\thepage`. That why we insert it at the end of the \LaTeX macro `\pagenumbering`, which is called by some `\xxxmatter` commands. In the case of `memoir` class using, we insert it at the end of `\@mempnum`. When using `\pagenumbering`, we also need to restart `par@page` counter. Consequently, we have wrapped `\par@patch@thepage` and counter restart in `\par@patch@pagenumbering`. We also call `\par@patch@thepage` it at the beginning of the document.

```

2551 \newcommand{\par@patch@thepage}{%
2552   \ifboolexpr{%
2553     bool{sameparallelpagnumber}%
2554     or bool{prevpgnotnumbered}%
2555   }%
2556   {%
2557     \patchcmd{\thepage}%
2558       {page}{par@page}%
2559       {}%
2560       {\led@error@fail@patch@thepage}%
2561   }%
2562 }%
2563
2564
2565 \newcommand{\par@patch@pagenumbering}{%
2566   \ifboolexpr{%
2567     bool{sameparallelpagnumber}%
2568     or bool{prevpgnotnumbered}%
2569   }%
2570   {%
2571     \setcounter{par@page}{1}%
2572   }%
2573   {}%
2574   \par@patch@thepage%
2575 }%
2576

```

```

2577 \ifl@dmemoir%
2578   \apptocmd{\@mempnum}%
2579     {\par@patch@pagenumbering}%
2580     {}%%
2581     {\led@error@fail@patch@@mempnum}%
2582
2583 \else%
2584   \apptocmd{\pagenumbering}%
2585     {\par@patch@pagenumbering}%
2586     {}%
2587     {\led@error@fail@patch@pagenumbering}%
2588 \fi%
2589
2590 \AtBeginDocument{\par@patch@thepage}%
2591 %

```

\@outputpage As its name says, \@outputpage is a L^AT_EX's macro called in the output routine. It is this macro which increases the page counter.. We patch it in order to increase, conditionally, the par@page counter.

```

2592 \AtBeginDocument{%
2593   \apptocmd{\@outputpage}{%
2594     \ifsameparallelpagenumber%
2595       \ifl@dprintingpages%
2596         \ifodd\c@page\else%
2597           \stepcounter{par@page}%
2598         \fi%
2599       \else%
2600         \stepcounter{par@page}%
2601       \fi%
2602     \else%
2603       \stepcounter{par@page}%
2604     \fi%
2605   }%
2606   {}%
2607   {\led@error@fail@patch@@outputpage}%
2608 }
2609 %

```

\thepar@page And now, initialize par@page counter.

```

2610 \newcounter{par@page}%
2611 \setcounter{par@page}{1}%
2612 %

```

XXI Sections' titles' commands

As switching from left to right pages does not clear the page since v1.13.0, but only creates new pages, no \vbox{} is inserted, and consequently parallel chapters are

mis-aligned.

So we patch the `\chapter` command in order to prevent this problem.

```
\chapter 2613 \pretocmd{\chapter}{%
2614 \ifl@dprintingpages%
2615 \vbox{}%
2616 \fi%
2617 }%
2618 {}%
2619 {}%
2620 %
```

`\eledsectnotoc` `\eledsectnotoc` just saves its content `\@eledsectnotoc`, which will be tested where sectioning commands will be printed.

```
2621 \newcommand{\eledsectnotoc}[1]{\xdef\@eledsectnotoc{#1}}
2622 \eledsectnotoc{R}
2623 %
```

`\eledsectmark` `\eledsectmark` just saves its content `\@eledsectmark`, which will be tested where sectioning commands will be printed.

```
2624 \newcommand{\eledsectmark}[1]{\xdef\@eledsectmark{#1}}
2625 \eledsectmark{L}
2626 %
```

`\eledsection@correcting@skip` Because the vertical correction needed after inserting a title in parallel depends whether we are in parallel columns or parallel pages, we stock its length in `\eledsection@correcting@skip`.

```
2627 \newskip\eledsection@correcting@skip
2628 %
```

`\eled@sectioningR@out` We save the sectioning commands of the right side in the `\eled@sectioningR@out` file.

```
2629 \newwrite\eled@sectioningR@out
2630 %
```

XXII Page break/no page break, depending on the specific line

We need to adapt the macro of the homonym section of `eledmac` to `eledpar`.

`\prev@pbR` The `\l@prev@pbR` macro is a `etoolbox`'s list, which contains the lines in which page breaks occur (before or after). The `\l@prev@nopbR` macro is a `etoolbox` list, which contains the lines in which NO page breaks occur (before or after).

```

2631 \def\l@prev@pbR{}
2632 \def\l@prev@nopbR{}
2633 %

```

\ledpbR The **\ledpbR** macro writes the call to **\led@pbR** in line-list file. The **\ledpb-**
\ledpbnumR numR macro writes the call to **\led@pbnumR** in line-list file. The **\lednopbR**
\lednopbnum macro writes the call to **\led@nopbR** in line-list file. The **\lednopbnumR** macro
\lednopbnumR writes the call to **\led@nopbnumR** in line-list file.

```

2634 \newcommand{\ledpbR}{\write\linenum@outR{\string\led@pbR}}
2635 \newcommand{\ledpbnumR}[1]{\write\linenum@outR{\string\
led@pbnumR{#1}}}
2636 \newcommand{\lednopbR}{\write\linenum@outR{\string\led@nopbR
}}
2637 \newcommand{\lednopbnumR}[1]{\write\linenum@outR{\string\
led@nopbnumR{#1}}}
2638 %

```

\led@pbR The **\led@pbR** add the absolute line number in the **\prev@pbR** list. The **\led@pbnumR**
\led@pbnumR add the argument in the **\prev@pbR** list. The **\led@nopbR** add the absolute line
\led@nopbR number in the **\prev@nopbR** list. The **\led@nopbnumR** add the argument in the
\led@nopbnumR **\prev@nopbR** list.

```

2639 \newcommand{\led@pbR}{\listxadd{\l@prev@pbR}{\the\
absline@numR}}
2640 \newcommand{\led@pbnumR}[1]{\listxadd{\l@prev@pbR}{#1}}
2641 \newcommand{\led@nopbR}{\listxadd{\l@prev@nopbR}{\the\
absline@numR}}
2642 \newcommand{\led@nopbnumR}[1]{\listxadd{\l@prev@nopbR}{#1}}
2643 %

```

XXIII Parallel ledgroup

\parledgroup@ The marks **\parledgroup@** contains information about the beginnings and endings
\parledgroupseries@ of notes in a parallel ledgroup. **\parledgroup@series** contains the footnote se-
\parledgroup@type@ ries. **\parledgroup@type** contains the type of the footnote: critical (Xfootnote)
or familiar (footnoteX).

```

2644 \newmarks\parledgroup@
2645 \newmarks\parledgroup@series
2646 \newmarks\parledgroup@type
2647 %

```

\parledgroup@notes@startL **\parledgroup@notes@startL** and **\parledgroup@notes@startR** are
\parledgroup@notes@startR used to mark the beginning of a note series in a parallel ledgroup.

```

2648 \newcommand{\parledgroup@notes@startL}{%
2649 \ifnum\usenamecount{1@dmaxlinesinpar\the\l@dpscL}>0%

```

```

2650 \IfStrEq{\splitfirstmarks\parledgroup@type}{footnoteX}{\
csuse{bhooknoteX@\splitfirstmarks\parledgroup@series}}{}%
2651 \IfStrEq{\splitfirstmarks\parledgroup@type}{Xfootnote}{\
csuse{bhookXnote@\splitfirstmarks\parledgroup@series}}{}%
2652 \fi%
2653 \global\ledgroupnotesL@true%
2654 \insert@noterule@ledgroup{L}%
2655 }
2656 \newcommand{\parledgroup@notes@startR}{%
2657 \ifnum\usernamecount{1@dmxlinesinpar\the\1@dpscR}>0%
2658 \IfStrEq{\splitfirstmarks\parledgroup@type}{footnoteX}{\
csuse{bhooknoteX@\splitfirstmarks\parledgroup@series}}{}%
2659 \IfStrEq{\splitfirstmarks\parledgroup@type}{Xfootnote}{\
csuse{bhookXnote@\splitfirstmarks\parledgroup@series}}{}%
2660 \fi%
2661 \global\ledgroupnotesR@true%
2662 \insert@noterule@ledgroup{R}%
2663 }
2664 %

```

`\parledgroup@notes@startL` and `\parledgroup@notes@endL` are used to mark the end of a note series in a parallel ledgroup.

```

2665 \newcommand{\parledgroup@notes@endL}{%
2666 \global\ledgroupnotesL@false%
2667 }
2668 \newcommand{\parledgroup@notes@endR}{%
2669 \global\ledgroupnotesR@false%
2670 }
2671 %

```

`\insert@noterule@ledgroup` A `\vskip` is not used when the boxes are constructed. So we insert it before ledgroup note series when parallel lines are constructed. This is the goal of `\insert@noterule@ledgroup`

```

2672 \newcommand{\insert@noterule@ledgroup}[1]{
2673 \IfStrEq{\splitbotmarks\parledgroup@}{begin}{%
2674 \IfStrEq{\splitbotmarks\parledgroup@type}{Xfootnote}{
2675 \csuse{ifledgroupnotes#1@}
2676 \vskip\skip\csuse{mp\splitbotmarks\
parledgroup@series footins}
2677 \csuse{\splitbotmarks\parledgroup@series
footnoterule}
2678 \fi
2679 }
2680 {}
2681 \IfStrEq{\splitbotmarks\parledgroup@type}{footnoteX}{
2682 \csuse{ifledgroupnotes#1@}
2683 \vskip\skip\csuse{mpfootins\splitbotmarks\
parledgroup@series}

```

```

2684         \csuse{footnoterule\splitbotmarks\
parledgroup@series}
2685         \fi
2686         }{}
2687     }
2688     {}
2689 }
2690 %

```

`\@parledgroupnotespacing` `\@parledgroupnotespacing` can be redefined by the user to change the inter-line spacing of ledgroup notes.

```

2691 \newcommand{\setparledgroupnotespacing}[1]{\gdef\
@parledgroupnotespacing{#1}}
2692 \newcommand{\@parledgroupnotespacing}{}
2693 %

```

`\parledgroup@notespacing@correction` `\parledgroup@notespacing@correction` is the difference between a normal line skip and a line skip in a note. It is set by `\parledgroup@notespacing@set@correction` called at the beginning of `\Pages`.

```

2694 \dimdef{\parledgroup@notespacing@correction}{0pt}
2695 \newcommand{\parledgroup@notespacing@set@correction}{%
2696     {\@getfirstseries\csuse{Xnotefontsize@\@firstseries}%We
suppose all the series has the same footnote size setup
2697     \@parledgroupnotespacing\dimgdef{\temp@spacing}{\
baselineskip}}}%
2698     \dimgdef{\parledgroup@notespacing@correction}{\baselineskip
-\temp@spacing}%
2699 }
2700 %

```

`\parledgroup@correction@notespacing@init` `\parledgroup@correction@notespacing@init` sets the value of accumulated corrections of note spacing to 0 pt. It is called at the beginning of each pages AND at the end of each ledgroup.

```

2701 \newcommand{\parledgroup@correction@notespacing@init}{
2702     \dimdef{\parledgroup@notespacing@correction@accumulated}{0
pt}
2703     \dimdef{\parledgroup@notespacing@correction@modulo}{0pt}
2704 }
2705 \parledgroup@correction@notespacing@init
2706 %

```

`\parledgroup@correction@notespacing@final` `\parledgroup@correction@notespacing@final` adds the total space deleted because of correction for notes, in a parallel ledgroup. It also adds the space needed by the other side spaces between note rules and notes. It is called after the print of each pstart/pend.


```

2707 \newcommand{\parledgroup@correction@notespacing@final}[1]{
2708     \ifparledgroup
2709     \vspace{\parledgroup@notespacing@correction@accumulated}
2710     \parledgroup@correction@notespacing@init%
2711     \ifstrequal{#1}{L}{
2712         \numdef{\@checking}{\the\l@dpscL-1}
2713     }{
2714         \numdef{\@checking}{\the\l@dpscR-1}
2715     }
2716     \dimdef{\@beforenotes@current@diff}{\csuse{
2717         \@parledgroup@beforenotes@\@checking L}-\csuse{
2718         \@parledgroup@beforenotes@\@checking R}}}%
2719     \ifstrequal{#1}{L}{%
2720         {% Left
2721         \ifdimgreater{\@beforenotes@current@diff}{0pt}{%
2722         \vspace{-\@beforenotes@current@diff}}%
2723         {% Right
2724         \ifdimgreater{\@beforenotes@current@diff}{0pt}{\vspace
2725         {\@beforenotes@current@diff}}}%
2726         \fi
2727     }
2728 }
2729 %

```

`\parledgroup@correction@notespacing` is used before each printed line. If it is a line of notes in parallel group, the space `\parledgroup@notespacing@correction` is decreased, to make interline space correct. The decreased space is added to `\parledgroup@notespacing@correction@accumulated` and `\parledgroup@notespacing@correction@modulo` is equal or greater than `\baselineskip`:

- It is decreased by `\baselineskip`.
- The total of line number in the current page is decreased by one.

For example, suppose an normal interline of 24 pt and interline for note of 12 pt. That means that the two lines of notes take the place of one normal line. For every two lines of notes, the line total for the current place is decreased by one.

```

2727 \newcommand{\parledgroup@correction@notespacing}[1]{%
2728     \csuse{ifledgroupnotes#1@}%
2729     \vspace{-\parledgroup@notespacing@correction}%
2730     \dimdef{\parledgroup@notespacing@correction@accumulated}
2731     {\parledgroup@notespacing@correction@accumulated+
\parledgroup@notespacing@correction}%
\dimdef{\parledgroup@notespacing@correction@modulo}{\parledgroup@notespacing@correction@modulo+
\parledgroup@notespacing@correction}%

```

```

2732 \ifdimless{\parledgroup@notespacing@correction@modulo
    }\baselineskip}{\advance\numpagelinesL -\@ne%
2733 \dimdef{\parledgroup@notespacing@correction@modulo}{\
    \parledgroup@notespacing@correction@modulo-\baselineskip}%
2734 }% mean greater than equal
2735 \fi%
2736 }
2737 %

```

`\parledgroup@beforenotesL` and `\parledgroup@beforenotesR` store the total of space before notes in the current parallel ledgroup.

```

2738 \dimdef\parledgroup@beforenotesL{0pt}
2739 \dimdef\parledgroup@beforenotesR{0pt}
2740 %

```

`\parledgroup@beforenotes@save` The macro `\parledgroup@beforenotes@save` dumps the space before notes of the current parallel ledgroup in a macro named with the current pstart number.

```

2741 \newcommand{\parledgroup@beforenotes@save}[1]{
2742 \ifparledgroup
2743 \csdingdef{@parledgroup@beforenotes@\the\csuse{
    l@dnumpstarts#1}#1}{\csuse{\parledgroup@beforenotes#1}}
2744 \csdingdef{\parledgroup@beforenotes#1}{0pt}
2745 \fi
2746 }
2747 %

```

XXIV Compatibility with `eledmac`

Here, we define some command for the `eledmac-compat` option.

```

2748 \ifeledmaccompat@%
2749
2750
2751 \unless\ifnocritical@
2752 \let\onlyXside\Xonlyside
2753 \fi
2754 \fi
2755 %

```

XXV The End

</code>

Appendix A Some things to do when changing version

Appendix A.1 Migration to **eledpar** 1.4.3

Version 1.4.3 corrects a bug added in version 0.12, which made hanging verse always flush right, despite the value of the first element in the `\setstanzaindents` command.

However, if you want to return to automatic flushright margins for verses with hanging indents, you have to redefine the `\hangingsymbol` command.

```
\renewcommand{\hangingsymbol}{\protect\hfill}
```

See the following two examples:

With standard `\hangingsymbol`:

A very long verse should sometimes be hanging. The position of the hanging verse is fixed.

With the modification of the `hangingsymbol`:

A very long verse should sometimes be hanging. And we can see that a hanging verse is flush right.

Appendix A.2 Migration from **eledpar** to **reledpar**

As for migration from *eledmac* to *reledmac*:

- One option has been removed because it is deprecated.
- Some of the customizations previously made by `\renewcommand` have been replaced with commands.
- Some command names have been changed in order to have a more logical and uniform pattern.

Appendix A.2.1 Deprecated options

The `shiftedverses` option has been removed. Use the general `shiftedpstart` option instead.

Appendix A.2.2 `\renewcommand` replaced with command

Many uses of `\renewcommand` have been replaced with uses of specific commands. Please read the handbook about these particular commands.

<i>Deprecated <code>\renewcommand</code></i>	<i>Replaced with</i>
<code>\goalfraction</code>	<code>\setgoalfraction</code>
<code>\parledgroupnotespacing</code>	<code>\setparledgroupnotespacing</code>
<code>\Rlineflag</code>	<code>\setRlineflag</code>

Appendix A.2.3 Commands the names of which have changed

In order to ease the migration from `eledpar` to `reledpar`, you may load `reledmac` with `eledmac-compat` option. However, it is advised to change the command names.

<i>Old command</i>	<i>New command</i>
<code>\onlyXside</code>	<code>\Xonlyside</code>

Appendix A.3 Migration to `reledpar` 2.2.0

The `astanza` can take now an option argument. Consequently, if the first line of verse in a `astanza` environment starts with brackets `[]`, you must precede them with a `\relax`. If you do not do it, the content of the brackets will be considered as an optional argument of the `astanza` environment.

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 and Maïeul Rouquette. *eLedmac A presumptuous attempt to port EDMAC, TABMAC and EDSTANZA to LaTeX*. December 2004. (Available from CTAN in `macros/latex/contrib/eledmac`)

Index

	Symbols	
<code>\@adv</code>	1
<code>\@astanza@line</code>	1
<code>\@cs@linesinparL</code>	1
<code>\@cs@linesinparR</code>	1
<code>\@cs@linesonpageL</code>	1
<code>\@cs@linesonpageR</code>	1
<code>\@donereallinesL</code>	1
<code>\@donereallinesR</code>	1
<code>\@donetotallinesL</code>	1
<code>\@donetotallinesR</code>	1
<code>\@eledsectionL</code>	1
<code>\@eledsectionR</code>	1
<code>\@lab</code>	1

\@lopL	1
\@lopR	1
\@nl	1
\@nl@regR	1
\@outputpage	1
\@parledgroupnotespacing	1
\@pend	1
\@pendR	1
\@pstartsfalse	1
\@pstartstrue	1
\@ref	1
\@ref@regR	1
\@set	1
\@stopastanza	1
\@writelinesinparL	1
\@writelinesinparR	1
\@writelinesonpageL	1
\@writelinesonpageR	1
CLASSmemoir	99
COMMAND\@Rlineflag	61, 66
COMMAND\@adv	30, 130
COMMAND\@cs@linesinparL	92
COMMAND\@cs@linesonpageL	92
COMMAND\@eledsectionL	76
COMMAND\@eledsectionR	76
COMMAND\@eledsectmark	101
COMMAND\@eledsectnotoc	101
COMMAND\@footnotetext	68
COMMAND\@goalfraction	9, 96
COMMAND\@l@dtmpecnta	59
COMMAND\@lab	29, 66, 130
COMMAND\@lopL	34, 87, 92
COMMAND\@lopR	34, 92
COMMAND\@mempnum	99
COMMAND\@namedef	70
COMMAND\@namuse	70
COMMAND\@nl	29, 35, 66, 130
COMMAND\@nl@regR	29
COMMAND\@outputpage	99, 100
COMMAND\@page	66
COMMAND\@parledgroupnotespacing	104
COMMAND\@pend	34
COMMAND\@pendR	34
COMMAND\@pstartstrue	74
COMMAND\@ref	32, 33, 36, 130
COMMAND\@ref@regR	32
COMMAND\@set	31, 130
COMMAND\@sw	33
COMMAND\AtBeginPairs	7, 40, 128
COMMAND\AtEveryPend	128–130

COMMAND\AtEveryPstart	2, 13, 47, 77, 128–130
COMMAND\AtEveryPstartCall	2, 13, 47, 129
COMMAND\Clear the right lines for \read@linelist	29
COMMAND\Columns	6, 7, 13, 39, 76, 80, 125, 126, 128, 130
COMMAND\Columns@print@after@pend	82
COMMAND\Columns@print@before@pstart	82
COMMAND\Lcolwidth	7, 9, 84
COMMAND\Leftsidehook	125
COMMAND\Leftsidehookend	125
COMMAND\Pages	3, 6, 8, 9, 13, 39, 57, 61, 62, 84, 90, 99, 104, 125, 128–131
COMMAND\Rcolwidth	7, 9, 84
COMMAND\Rightsidehook	125
COMMAND\Rightsidehookend	125
COMMAND\Rlineflag	107
COMMAND\Xmaxnotes	10
COMMAND\Xnoteswidthliketwocolumns	8, 128
COMMAND\Xonlyside	10, 61, 108
COMMAND\&	14, 15
COMMAND\absline@numR	28
COMMAND\add@penalties	59
COMMAND\add@penaltiesL	59
COMMAND\advanceline	30, 36, 130
COMMAND\affixline@num	54
COMMAND\affixline@numR	54, 125, 126
COMMAND\affixpstart@num	57
COMMAND\affixpstart@numR	57
COMMAND\affixside@note	67
COMMAND\aftercolumnseparator	8, 80, 127
COMMAND\araw@textfalse	75
COMMAND\araw@texttrue	75
COMMAND\astanza	14
COMMAND\at@begin@pairs	40
COMMAND\autopar	13
COMMAND\ballast@count	59
COMMAND\baselineskip	79, 105
COMMAND\bbl@set@language	71, 72, 130
COMMAND\beforecolumnseparator	8, 80, 127
COMMAND\begin	14, 15
COMMAND\beginnumbering	11–13, 22, 29, 80, 126, 127, 130
COMMAND\beginnumberingR	35
COMMAND\bf	126
COMMAND\bfseries	126
COMMAND\brokenpenalty	59
COMMAND\chapter	101, 125
COMMAND\check@goal	96
COMMAND\check@pstarts	74
COMMAND\checkpageL	87, 93
COMMAND\checkpb@columns	79
COMMAND\checkpbL	94
COMMAND\checkpbR	94

COMMAND\checkdraw@text	75
COMMAND\checkverseL	95
COMMAND\checkverseR	95
COMMAND\clear(double)page	91
COMMAND\clearl@dleftpage	91
COMMAND\clearl@drightpage	91
COMMAND\clearpage	91, 129
COMMAND\cleartoevenpage	91
COMMAND\cleartol@devenpage	91
COMMAND\columnrulewidth	7, 79
COMMAND\columns@position	80
COMMAND\columnseparator	7
COMMAND\columnspan	7, 127
COMMAND\correct@Xfootins@box	62, 64, 129
COMMAND\correct@footinsX@box	62, 129
COMMAND\critext	129
COMMAND\csname	39
COMMAND\displaywidowpenalty	59
COMMAND\do@actions	53
COMMAND\do@actions@fixedcode	125
COMMAND\do@actions@nextR	53
COMMAND\do@actionsR	53, 125
COMMAND\do@ballast	59
COMMAND\do@ballastR	53
COMMAND\do@insidelineLhook	127
COMMAND\do@insidelineRhook	127
COMMAND\do@line	48
COMMAND\do@line(L/R)	50
COMMAND\do@lineL	48, 59, 125, 126
COMMAND\do@lineLhook	125
COMMAND\do@lineR	51, 125–127
COMMAND\do@lineRhook	125
COMMAND\do@lockoff	131
COMMAND\do@lockoffR	31
COMMAND\do@lockon	130
COMMAND\do@lockonR	31
COMMAND\doinsidelineLhook	128
COMMAND\doinsidelineRhook	128
COMMAND\dolineLhook	128
COMMAND\dolineRhook	128
COMMAND\edindex	129
COMMAND\edlabel	126, 129
COMMAND\edtext	32, 36, 129
COMMAND\eled@sectioningR@out	101
COMMAND\eledchapter	129
COMMAND\eledsection	128, 129, 131
COMMAND\eledsection@correcting@skip	101
COMMAND\eledsectmark	16, 101
COMMAND\eledsectnotoc	16, 101
COMMAND\eledxxx	128

COMMAND\end	14
COMMAND\endgraf	46
COMMAND\endlock	36, 130
COMMAND\endnumbering	11, 13, 22, 23, 130
COMMAND\endsub	36, 130
COMMAND\endumbering	11
COMMAND\expandafter	37
COMMAND\extensionchars	21
COMMAND\firstlinenum	12, 127, 131
COMMAND\firstsublinenum	12, 127, 131
COMMAND\fix@page	30, 130
COMMAND\flag@end	36, 125, 128
COMMAND\flag@start	36, 128
COMMAND\flush@notesR	60
COMMAND\footnoteX	37
COMMAND\footnoteXmk	10
COMMAND\footnoteXnomk	10, 37
COMMAND\frontmatter	17
COMMAND\get@nextboxL	126
COMMAND\get@nextboxR	126
COMMAND\getline@numL	52
COMMAND\getline@numR	52
COMMAND\getlinesfrompagelistL	92
COMMAND\getlinesfrompagelistR	92
COMMAND\getlinesfromparlistL	92
COMMAND\getlinesfromparlistR	92
COMMAND\gl@p	38
COMMAND\goalfraction	107
COMMAND\hangingsymbol	107, 126
COMMAND\hfill	78, 80
COMMAND\ifbypage@	130
COMMAND\ifbypstart@R	130
COMMAND\ifdim	78
COMMAND\ifinserthangingsymbol	68
COMMAND\ifinserthangingsymbolR	68
COMMAND\ifl@pagefull	93
COMMAND\ifl@dpaging	19, 128
COMMAND\ifl@dpairing	19, 125
COMMAND\ifl@dsamelang	128
COMMAND\ifl@dsamepage	93
COMMAND\ifl@pagefull	87
COMMAND\ifledRcol	19
COMMAND\iflledRcol	126
COMMAND\ifnumberedpar@	43
COMMAND\ifnumberingR	126
COMMAND\ifnumberpstart	39
COMMAND\ifpst@rtedL	22, 44, 125
COMMAND\ifpst@rtedR	22
COMMAND\ifsublines@	31
COMMAND\insert@countR	32

COMMAND\insert@noterule@ledgroup	103
COMMAND\insertlines@list	32
COMMAND\insertlines@listR	32
COMMAND\inserts@list	43
COMMAND\inserts@listR	58
COMMAND\l@d@nums	36, 60
COMMAND\l@d@set	31, 36, 130
COMMAND\l@dLcolrawbox	43
COMMAND\l@dLcolrawbox1	73
COMMAND\l@dLcolrawbox2	73
COMMAND\l@dRcolrawbox	43
COMMAND\l@dbfnote	68, 130
COMMAND\l@dcalc@maxoftwo	93
COMMAND\l@dcalc@minoftwo	93
COMMAND\l@dchecklang	125, 128
COMMAND\l@dcsnote	127
COMMAND\l@dleftbox	48, 86, 129
COMMAND\l@dlinenumR	27, 125
COMMAND\l@dlsnote	127
COMMAND\l@dmake@labels	66
COMMAND\l@dmaxlinesinpar	85
COMMAND\l@dmaxlinesinpar1	74
COMMAND\l@dminpagelines	85, 125
COMMAND\l@dnumpstartsL	73, 125
COMMAND\l@dprintingcolumnstrue	129
COMMAND\l@dprintingpagestrue	129
COMMAND\l@dpscL	76, 85, 97
COMMAND\l@dpscR	76, 85
COMMAND\l@drsnote	127
COMMAND\l@dsetupmaxlinecounts	74
COMMAND\l@duselanguage	71, 72, 125
COMMAND\l@dzeromaxlinecounts	74
COMMAND\l@prev@nopbR	101
COMMAND\l@prev@pbR	101
COMMAND\labelpstarttrue	126
COMMAND\labelref@list	66
COMMAND\labelref@listR	66
COMMAND\lang	71
COMMAND\last@page@numR	30
COMMAND\led	126
COMMAND\led@nopbR	102
COMMAND\led@nopbnumR	102
COMMAND\led@pbR	102
COMMAND\led@pbnumR	102
COMMAND\ledinnerrote	15
COMMAND\ledleftnote	15
COMMAND\lednopb	15, 79, 94
COMMAND\lednopbR	102
COMMAND\lednopbnumR	102
COMMAND\ledouterote	15

COMMAND\ledpb	79, 94
COMMAND\ledpbR	102
COMMAND\ledpbnumR	102
COMMAND\ledrightnote	15
COMMAND\ledsidenote	15
COMMAND\ledstrutL	125
COMMAND\ledstrutR	125
COMMAND\ledthegoal	96
COMMAND\ledtrutL	125
COMMAND\leftlinenumR	27, 125
COMMAND\let	37, 38
COMMAND\line@list@R	33
COMMAND\line@list@stuff	29, 35
COMMAND\line@margin	26
COMMAND\line@marginR	26, 125
COMMAND\line@numR	28
COMMAND\lineation	12, 13, 130
COMMAND\lineation*	13, 25, 128
COMMAND\lineationR	12, 25, 130
COMMAND\linenum@out	66
COMMAND\linenum@outR	35
COMMAND\linenumincrement	12, 127, 131
COMMAND\linenummargin	12, 26, 125, 130
COMMAND\linenumrepR	27, 125
COMMAND\linesinpar@listL	34, 92
COMMAND\linesonpage@listL	34, 92
COMMAND\lock@off	31
COMMAND\lock@on	31
COMMAND\mainmatter	17
COMMAND\makeatletter	50
COMMAND\maxchunks	6, 14, 73, 74
COMMAND\maxhnotesX	10
COMMAND\memorydump	11, 24
COMMAND\n@num	130
COMMAND\new@lineL	35
COMMAND\new@lineR	35
COMMAND\newhookcommand@series	38
COMMAND\newif	129
COMMAND\newpage	90, 91, 129
COMMAND\newseries	39
COMMAND\newseries@	36
COMMAND\newseries@par	36, 38, 39
COMMAND\noeledxxx	128
COMMAND\nomark@	37
COMMAND\normalbfnoteX	125, 131
COMMAND\notesXwidthliketwocolumns	8, 128
COMMAND\num@lines	59
COMMAND\num@lines(R)	43
COMMAND\numberingR	23
COMMAND\numberlinefalse	5

COMMAND\numberonlyfirstinline	126
COMMAND\numberpstartfalse	12
COMMAND\numberpstarttrue	12, 126, 131
COMMAND\one@line	43, 68
COMMAND\one@lineR	43
COMMAND\onlyXside	108
COMMAND\onlysideX	10, 61, 130
COMMAND\otherlanguage	131
COMMAND\page@action	31, 131
COMMAND\pagenumbering	99, 131
COMMAND\pagetotal	87, 129
COMMAND\par@line	59
COMMAND\par@line(R)	43
COMMAND\par@patch@pagenumbering	99
COMMAND\par@patch@thepage	99
COMMAND\parledgroup@	102
COMMAND\parledgroup@beforenotes@save	106
COMMAND\parledgroup@beforenotesL	106
COMMAND\parledgroup@beforenotesR	106
COMMAND\parledgroup@correction@notespacing	105
COMMAND\parledgroup@correction@notespacing@final	104
COMMAND\parledgroup@correction@notespacing@init	104
COMMAND\parledgroup@notes@endL	103
COMMAND\parledgroup@notes@endR	103
COMMAND\parledgroup@notes@startL	102
COMMAND\parledgroup@notes@startR	102
COMMAND\parledgroup@notespacing@correction	104, 105
COMMAND\parledgroup@notespacing@correction@accumulated	105
COMMAND\parledgroup@notespacing@correction@modulo	105
COMMAND\parledgroup@notespacing@set@correction	104
COMMAND\parledgroup@series	102
COMMAND\parledgroup@type	102
COMMAND\parledgroupnotespacing	107
COMMAND\parledgrouptrue	15
COMMAND\patchcmd	130
COMMAND\pausenumbering	23, 24
COMMAND\pend	3, 6, 13–15, 39, 43, 46, 47, 73, 82, 127–130
COMMAND\pendL	127, 128
COMMAND\pendR	128
COMMAND\pends	13
COMMAND\prev@nopbR	102
COMMAND\prev@pbR	102
COMMAND\prevpgstyle	19
COMMAND\print@Xnotes	61
COMMAND\print@Xnotes@forpages	61, 129
COMMAND\print@columnseparator	78, 128
COMMAND\print@eledsectionL	49
COMMAND\print@line	49
COMMAND\print@lineL	49
COMMAND\print@notesX@forpages	129

COMMAND\printlines	61
COMMAND\printlinesR	60, 125
COMMAND\pstart	3, 6, 12–15, 25, 36, 39, 43, 44, 46, 47, 73, 77, 82, 126, 127, 130, 131
COMMAND\pstartL	47, 127
COMMAND\pstartR	47, 126, 127
COMMAND\pstartinfootnote	130
COMMAND\rw@text	73
COMMAND\read@linelist	29, 110, 131
COMMAND\ref@reg	32
COMMAND\ref@regR	32, 130
COMMAND\relax	108
COMMAND\reledmac	131
COMMAND\renewcommand	107
COMMAND\resumenumbering	23, 24, 128
COMMAND\resumenumberingR	128
COMMAND\rightlinenumR	27, 125
COMMAND\section	125
COMMAND\section@num	21
COMMAND\selectlanguage	14, 71, 72
COMMAND\set@line	36, 131
COMMAND\set@line@action	31, 131
COMMAND\setRlineflag	13, 107
COMMAND\setgoalfraction	9, 107
COMMAND\sethangingsymbol	15
COMMAND\setline	31, 36, 131
COMMAND\setlinenum	31, 36, 131
COMMAND\setnoteposition...	80
COMMAND\setparledgroupnotespacing	107, 131
COMMAND\setposition...	80
COMMAND\setprintlines	125
COMMAND\setstanzaindents	8, 14, 107
COMMAND\setwidth...	80
COMMAND\sidenotemargin	15, 128
COMMAND\sidenotemargin*	15, 128
COMMAND\skipnumbering	12, 130
COMMAND\sloppy	7
COMMAND\stanza	6, 8, 14, 41, 69, 126
COMMAND\stanzanumtrue	15
COMMAND\startlock	36, 130
COMMAND\startsub	36, 130
COMMAND\sub@action	31, 131
COMMAND\sub@off	66
COMMAND\sub@on	66
COMMAND\subline@numR	28
COMMAND\sublinenumincrement	12, 127, 131
COMMAND\sublinenumrepR	27, 125
COMMAND\sza@0@	14
COMMAND\textheight	10
COMMAND\textwidth	41
COMMAND\thefootnoteX	127

COMMAND\theledlanguageL	71, 72
COMMAND\theledlanguageR	71, 72
COMMAND\thepage	17, 99
COMMAND\thepstartL	12, 126
COMMAND\thepstartR	12, 126
COMMAND\thestanzaL	15
COMMAND\thestanzaR	15
COMMAND\vbox	44
COMMAND\vl@dbfnote	68
COMMAND\vskip	103
COMMAND\vsplit	59
COMMAND\widthliketwocolumns	8, 19
COMMAND\widthliketwocolumnsfalse	8
COMMAND\widthliketwocolumnstrue	8
COMMAND\xright@appenditem	37
COMMAND\xspace	18
COMMAND\xxxfootstart	80
COMMAND\xxxmatter	99
ENVIRONMENTLeftside	41
ENVIRONMENTRightside	42
ENVIRONMENTastanza	14, 69, 108, 131
ENVIRONMENTpages	40
ENVIRONMENTpairs	40
PACKAGEEDMAC	108
PACKAGEEDSTANZA	108
PACKAGEEledmac	37, 60, 130
PACKAGEEledpar	130
PACKAGETABMAC	108
PACKAGEbabel	14, 71, 72
PACKAGEedmac	108
PACKAGEeledmac	4, 73, 106–108, 127, 128, 130
PACKAGEeledpar	4, 5, 10, 27, 107, 108, 127–129
PACKAGEetoolbox	78, 101
PACKAGEledmac	5
PACKAGEledpar	1, 5
PACKAGEMemoir	108
PACKAGEMusixtex	127
PACKAGEpolyglossia	14, 71, 72
PACKAGEreledmac	1, 3, 5, 6, 8, 10, 12–16, 18, 19, 22, 26–32, 34–39, 49, 57, 66, 68, 85, 90, 107, 108, 130, 131
PACKAGEreledpar	1, 3–6, 8, 9, 14, 15, 17–19, 25, 28, 29, 34, 36–39, 66, 107, 108, 130
PACKAGESetSpace	2, 16
PACKAGEXkeyval	18

A

\absline@numR	1
\actionlines@listR	1
\actions@listR	1
\add@inserts@nextR	1
\add@insertsR	1

<code>\add@penaltiesL</code>	1
<code>\add@penaltiesR</code>	1
<code>\advanceline</code>	1
<code>\affixline@numR</code>	1
<code>\affixpstart@numL</code>	1
<code>\affixpstart@numR</code>	1
<code>\affixside@noteR</code>	1
<code>\aftercolumnseparator</code>	1, 8
<code>\araw@textfalse</code>	1
<code>\araw@texttrue</code>	1
<code>astanza (environment)</code>	14
<code>\AtBeginPairs</code>	1, 7
<code>\AtEveryPstartCall</code>	1
<code>\autopar</code>	13

B

<code>\bbl@set@language</code>	1
<code>\beforecolumnseparator</code>	1, 8
<code>\beginnumbering</code>	11
<code>\beginnumberingR</code>	1

C

<code>\c@firstlinenumR</code>	1
<code>\c@firstsublinenumR</code>	1
<code>\c@linenumincrementR</code>	1
<code>\c@sublinenumincrementR</code>	1
<code>\check@l@ckR</code>	1
<code>\checksub@l@ckR</code>	1
<code>\chapter</code>	1
<code>\chapterinpages</code>	1
<code>\check@goal</code>	1
<code>\check@pstarts</code>	1
<code>\checkpageL</code>	1
<code>\checkpageR</code>	1
<code>\checkpb@columns</code>	1
<code>\checkpbL</code>	1
<code>\checkpbR</code>	1
<code>\checkraw@text</code>	1
<code>\checkverseL</code>	1
<code>\checkverseR</code>	1
<code>\clearl@dleftpage</code>	1
<code>\clearl@drighthpage</code>	1
<code>\cleartoevenpage</code>	1
<code>\cleartol@devenpage</code>	1
<code>\columnrulewidth</code>	1, 7
<code>\Columns</code>	1, 6
<code>\columns@position</code>	1
<code>\Columns@print@after@pend</code>	1
<code>\Columns@print@before@pstart</code>	1
<code>\columnseparator</code>	1, 7

\columnspan	1, 7
\correct@footinsX@box	1
\correct@Xfootins@box	1
\countLline	1
\countRline	1
\critext	1

D

\do@actions@fixedcodeR	1
\do@actions@nextR	1
\do@actionsR	1
\do@ballastR	1
\do@insidelineLhook	1
\do@insidelineRhook	1
\do@lineL	1
\do@lineLhook	1
\do@lineR	1
\do@lineRhook	1
\do@lockoff	1
\do@lockoffR	1
\do@lockon	1
\do@lockonR	1
\doinsidelineLhook	1
\doinsidelineRhook	1
\dolineLhook	1
\dolineRhook	1
\dump@pstartL@pc	1
\dump@pstartR@pc	1

E

\edlabel	1
\edtext	1
\eled@sectioningR@out	1
\eledpar@error	1
\eledsection@correcting@skip	1
\eledsectmark	1, 16
\eledsectnotoc	1, 16
\endlock	1
\endnumbering	1, 11
\endnumberingR	1
\endsub	1
environments:	
astanza	14
Leftside	11
pages	8
pairs	6
Rightside	11

F

\f@x@l@cksR	1
-------------	---

<code>\finish@Pages@notes</code>	1
<code>\first@linenum@out@Rfalse</code>	1
<code>\first@linenum@out@Rtrue</code>	1
<code>\firstlinenum</code>	1, 12
<code>\firstlinenum*</code>	1, 12
<code>\firstsublinenum</code>	1, 12
<code>\firstsublinenum*</code>	1, 12
<code>\fix@page</code>	1
<code>\flag@end</code>	1
<code>\flag@start</code>	1
<code>\flush@notesR</code>	1
<code>\footnoteXmk</code>	10
<code>\footnoteXnomk</code>	10

G

<code>\get@nextboxL</code>	1
<code>\get@nextboxR</code>	1
<code>\getline@numR</code>	1
<code>\getlinesfrompagelistL</code>	1
<code>\getlinesfrompagelistR</code>	1
<code>\getlinesfromparlistL</code>	1
<code>\getlinesfromparlistR</code>	1
<code>\goalfraction</code>	1

I

<code>\if@pstarts</code>	1
<code>\ifaraw@text</code>	1
<code>\iffirst@linenum@out@R</code>	1
<code>\ifinstanzaL</code>	1
<code>\ifinstanzaR</code>	1
<code>\ifl@dpagfull</code>	1
<code>\ifl@dpaging</code>	1
<code>\ifl@dpairing</code>	1
<code>\ifl@dsamepage</code>	1
<code>\ifl@dusedbabel</code>	1
<code>\ifledRcol</code>	1
<code>\ifprevpgnotnumbered</code>	1
<code>\ifprint@last@after@pendL</code>	1
<code>\ifprint@last@after@pendR</code>	1
<code>\ifpstartedL</code>	1
<code>\ifpstartedR</code>	1
<code>\ifpstartnumR</code>	1
<code>\ifsameparallelpagenunder</code>	1
<code>\ifshiftedpstarts</code>	1
<code>\ifwidthliketwocolumns</code>	1
<code>\ifwrittenlinesL</code>	1
<code>\init@series@par</code>	1
<code>\initnumbering@sectcountR</code>	1
<code>\insert@countR</code>	1
<code>\insert@noterule@ledgroup</code>	1

\inserthangingsymbolL	1
\inserthangingsymbolR	1
\insertlines@listR	1
\inserts@listR	1

L

\l@dd@set	1
\l@dbfnote	1
\l@dc@maxchunks	1
\l@dcalc@maxoftwo	1
\l@dcalc@minoftwo	1
\l@dcalcnun	1
\l@dchecklang	1
\l@dleftbox	1
\l@dlinenumR	1
\l@dmake@labelsR	1
\l@dminpagelines	1
\l@dnumpstartsL	1
\l@dnumpstartsR	1
\l@dpagefullfalse	1
\l@dpagefulltrue	1
\l@drightbox	1
\l@dsamepagefalse	1
\l@dsamepagetrue	1
\l@dsetupmaxlinecounts	1
\l@dsetuprawboxes	1
\l@dskipversenumberR	1
\l@dusedbabelfalse	1
\l@dusedbabeltrue	1
\l@duselanguage	1
\l@dzeromaxlinecounts	1
\l@pscL	1
\l@pscR	1
\labelref@listR	1
\last@page@numR	1
\Lcolwidth	1, 7, 9
\led@err@BadLeftRightPstarts	1
\led@err@Columns@InsideEnv	1
\led@err@LeftOnRightPage	1
\led@err@Leftside@PreviousNotPrinted	1
\led@err@Pages@InsideEnv	1
\led@err@RightOnLeftPage	1
\led@err@Rightside@PreviousNotPrinted	1
\led@err@TooManyPstarts	1
\led@error@fail@patch@@mempnum	1
\led@error@fail@patch@@outputpage	1
\led@error@fail@patch@pagenumbering	1
\led@error@fail@patch@thepage	1
\led@nopbnumR	1
\led@nopbR	1

<code>\led@pbnR</code>	1
<code>\led@pbR</code>	1
<code>\lednopbnum</code>	1
<code>\lednopbnumR</code>	1
<code>\ledpbnR</code>	1
<code>\ledpbR</code>	1
<code>\ledstrutL</code>	1
<code>\ledstrutR</code>	1
<code>\ledthegoal</code>	1
<code>\leftlinenumR</code>	1
<code>\leftpstartnumL</code>	1
<code>\leftpstartnumR</code>	1
<code>Leftside (environment)</code>	11
<code>\Leftsidehook</code>	1
<code>\Leftsidehookend</code>	1
<code>\line@list@stuffR</code>	1
<code>\line@listR</code>	1
<code>\line@marginR</code>	1
<code>\line@numR</code>	1
<code>\lineation*</code>	1, 12
<code>\lineationR</code>	1, 12
<code>\linenum@outR</code>	1
<code>\linenumincrement</code>	1, 12
<code>\linenumincrement*</code>	1, 12
<code>\linenummargin</code>	1
<code>\linenumrepR</code>	1
<code>\linesinpar@listL</code>	1
<code>\linesinpar@listR</code>	1
<code>\list@clearing@regR</code>	1
<code>\list@pstartL@pc</code>	1
<code>\list@pstartR@pc</code>	1
<code>\lock@off</code>	1

M

<code>\maxchunks</code>	1, 6
<code>\maxlinesinpar@list</code>	1
<code>\memorydump</code>	11
<code>\memorydumpL</code>	1
<code>\memorydumpR</code>	1

N

<code>\n@num</code>	1
<code>\namebox</code>	1
<code>\new@lineL</code>	1
<code>\new@lineR</code>	1
<code>\newnamebox</code>	1
<code>\newnamecount</code>	1
<code>\newseries@par</code>	1
<code>\normalbfnoteX</code>	1
<code>\notesXwidthliketwocolumns</code>	8

\num@linesR	1
\numberpstartfalse	12
\numberpstarttrue	12
\numpagelinesL	1
\numpagelinesR	1

O

\one@lineR	1
\onlysideX	10

P

\page@action	1
\page@numR	1
\Pages	1, 8
pages (environment)	8
pairs (environment)	6
\par@lineR	1
\par@patch@pagenumbering	1
\par@patch@thepage	1
\parledgroup@	1
\parledgroup@beforenotes@save	1
\parledgroup@beforenotesL	1
\parledgroup@beforenotesR	1
\parledgroup@correction@notespacing	1
\parledgroup@correction@notespacing@final	1
\parledgroup@correction@notespacing@init	1
\parledgroup@notes@startL	1
\parledgroup@notes@startR	1
\parledgroup@notespacing@correction	1
\parledgroup@notespacing@set@correction	1
\parledgroupseries@	1
\parledgrouptype@	1
\pausenumberingR	1
\pend	13
\pendL	1
\pendR	1
\prev@nopbR	1
\prev@pbR	1
\prevpgstyle	1
\print@columnseparator	1
\print@eledsectionL	1
\print@eledsectionR	1
\print@lineL	1
\print@lineR	1
\print@notesX@forpages	1
\print@Xnotes@forpages	1
\printlinesR	1
\pstart	13
\pstartL	1
\pstartR	1

R

\Rcolwidth	1, 7, 9
\read@linelist	1
\restore@pstartL@pc	1
\restore@pstartR@pc	1
\resumenumberingR	1
\rightlinenumR	1
\rightpstartnumL	1
\rightpstartnumR	1
Rightside (environment)	11
\Rightsidehook	1
\Rightsidehookend	1
\Rlineflag	1

S

\section@numR	1
\selectlanguage	1
\set@line	1
\set@line@action	1
\setgoalfraction	1, 9
\sethangingsymbol	15
\setline	1
\setlinenum	1
\setnamebox	1
\setnotepositionliketwocolumns@C	1
\setnotepositionliketwocolumns@L	1
\setnotepositionliketwocolumns@R	1
\setpositionliketwocolumns@C	1
\setpositionliketwocolumns@L	1
\setpositionliketwocolumns@R	1
\setRlineflag	13
\setwidthliketwocolumns@C	1
\setwidthliketwocolumns@L	1
\setwidthliketwocolumns@R	1
\sidenote@marginR	1
\sidenotemargin*	1
\skip@lockoff	1
\skipnumbering	1, 12
\startlock	1
\startsub	1
\sub@action	1
\subline@numR	1
\sublinenumincrement	1, 12
\sublinenumincrement*	1, 12
\sublinenumrepR	1

T

\theledlanguageL	1
\theledlanguageR	1
\thepar@page	1

<code>\thepstartL</code>	12
<code>\thepstartR</code>	12
<code>\thestanzaL</code>	<u>1</u> , 15
<code>\thestanzaR</code>	<u>1</u> , 15

U

<code>\unhnamebox</code>	<u>1</u>
<code>\unvnamebox</code>	<u>1</u>
<code>\usernamecount</code>	<u>1</u>

W

<code>\widthliketwocolumns</code>	8
-----------------------------------	---

X

<code>\Xnoteswidthliketwocolumns</code>	8
<code>\Xonlyside</code>	10

Change History

v0.1.0.	
General: First public release	1
v0.2.0.	
General: Added section of babel related code	71
Fix babel problems	1
\Columns: Added \l@dchecklang and \l@duselanguage to \Columns	77
\Pages: Added \l@duselanguage to \Pages	86
v0.3.0.	
General: Added \do@lineLhook and \do@lineRhook	50
Added hooks into Leftside environment	41
Reorganize for ledarab	1
\affixline@numR: Changed \affixline@numR to match new eledmac	54
\do@actions@nextR: Used \do@actions@fixedcode in \do@actionsR	53
\do@lineL: Added \do@lineLhook to \do@lineL	48
Simplified \do@lineL by using macros for some common code	48
\do@lineR: Changed \do@lineR similarly to \do@lineL	51
\flag@end: Removed extraneous spaces from \flag@end	36
\ifledRcol: Moved \ifl@dpairing to eledmac	19
\ifpstartedR: Moved \ifpstartedL to eledmac	22
\l@dlinenumR: Simplified \leftlinenumR and \rightlinenumR by intro-	
ducing \l@dlinenumR	27
\l@dnumpstartsR: Moved \l@dnumpstartsL to eledmac	73
\ledstrutR: Added \ledtrutL and \ledstrutR	90
\normalbfnotex: Removed extraneous spaces from \normalbfnotex	68
\Pages: Added \ledstrutL to \Pages	86
Added \ledstrutR to \Pages	88
\printlinesR: Simplified \printlinesR by using \setprintlines	61
\Rightsidehookend: Added \Leftsidehook, \Leftsidehookend, \Right-	
sidehook and \Rightsidehookend	42
\sublinenumrepR: Added \linenumrepR and \sublinenumrepR	27
v0.3.a.	
General: Minor \linenummargin fix	1
\line@marginR: Do not just set \line@marginR in \linenummargin	26
v0.3.b.	
General: Improved parallel page balancing	1
\Pages: Added \l@dminpagelines calculation for succeeding page pairs	89
v0.3.c.	
General: Compatibilty with Polyglossia	1
v0.4.0.	
General: No more ledparpatch. All patches are now in the main file.	1
v0.5.0.	
General: Corrections about \section and other titles in numbered sections	1
v0.6.0.	
General: Be able to use \chapter in parallel pages.	1
v0.7.0.	
General: Option 'shiftedverses' which make there is no blank between two parallel verses	
with inequal length.	1

v0.8.0.	
General: Possibility to have a symbol on each hanging of verses, like in the french typography. Redefine the commande <code>\hangingsymbol</code> to define the character.	1
v0.9.0.	
General: Possibility to number <code>\pstart</code>	12
Possibility to number the pstart with the commands <code>\numberpstarttrue</code>	1
<code>\ifledRcol</code> : Moved <code>\iflledRcol</code> and <code>\ifnumberingR</code> to <code>eledmac</code>	19
v0.9.1.	
General: The numbering of the pstarts restarts on each <code>\beginnumbering</code>	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
v0.9.3.	
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
v0.10.0.	
General: <code>\edlabel</code> commands on the right side are now correctly indicated.	1
<code>\edlabel</code> commands which start a paragraph are now put in the right place.	1
v0.11.0.	
General: Change <code>\do@lineL</code> and <code>\do@lineR</code> to allow line numbering by pstart (like in <code>eledmac</code> 0.15).	48
Lineation can be by pstart (like in <code>eledmac</code> 0.15).	25
New management of <code>hangingsymbol</code> insertion, preventing undesirable insertions. . .	68
<code>\affixline@numR</code> : Changed <code>\affixline@numR</code> to allow to disable line numbering (like in <code>eledmac</code> 0.15).	54
<code>\Columns</code> : Line numbering by pstart.	78
<code>\get@nextboxR</code> : Change <code>\get@nextboxL</code> and <code>\get@nextboxR</code> to allow to disable line numbering (like in <code>eledmac</code> 0.15).	96
Pstart number can be printed in side	97
<code>\inserthangingsymbolR</code> : Prevent the column separator for hanging verse from shifting	68
v0.12.0.	
General: New management of <code>hangingsymbol</code> insertion, preventing undesirable insertions.	68
v1.0.0.	
General: Compatibility with <code>eledmac</code> . Change name to <code>eledpar</code>	1
Debug in lineation by pstart	25
v1.0.1.	
General: Correction on <code>\numberonlyfirstinline</code> with lineation by pstart or by page.	1
v1.1.0.	
General: <code>Shiftedverses</code> becomes <code>shiftedpstarts</code>	1
<code>\pstartR</code> : Add <code>\labelpstarttrue</code> (from <code>eledmac</code>).	43
v1.1.1.	
<code>\pstartR</code> : Correct <code>\pstartR</code> bug introduced by 1.1.	43
v1.1.2.	
<code>\affixside@noteR</code> : Remove spurious space between line number and line content	67
v1.2.0.	
General: Support for <code>\led<section></code> commands in parallel texts.	1

v1.2.1.	
<code>\initnumbering@sectcountR</code> : For the right section, the counter is defined only once.	23
v1.3.0.	
<code>\edtext</code> : Manage RTL language.	36
v1.3.1.	
<code>\l@dbfnote</code> : Compatibility of standard footnotes with <code>eledmac</code> when theses footnotes contain any commands.	68
v1.3.2.	
General: Debug with some classes.	1
v1.3.3.	
General: Debugging the left notes of the right column.	67
<code>\l@dbfnote</code> : Spurious space with footnote in right column.	68
v1.3.4.	
General: Allow use of commands in sidenotes, as introduced by <code>eledmac</code> 1.0.	67
v1.3.5.	
<code>\normalbfnoteX</code> : Allows one to redefine <code>\thefootnoteX</code> with <code>\alph</code> when some packages are loaded.	68
v1.4.0.	
General: Added <code>\do@insidelineLhook</code> and <code>\do@insidelineRhook</code>	50
v1.4.1.	
General: Enable the use of <code>stanzaindentsrepetition</code> within <code>astanza</code> environment.	69
<code>\normalbfnoteX</code> : Fix bug with normal familiar footnotes when mixing RTL and LTR text.	68
v1.4.3.	
General: Corrects a false hanging verse when a verse is exactly the length of a line.	1
<code>\inserthangingsymbolR</code> : Hanging verse is no longer automatically flush right.	68
<code>\pendL</code> : Spurious spaces in <code>\pendL</code>	46
<code>\pendR</code> : Spurious spaces in <code>\pstartR</code>	47
<code>\pstartR</code> : Spurious spaces in <code>\pstartL</code> and <code>\pstartR</code>	43
v1.5.0.	
General: Add, as in <code>eledmac</code> , features to manage page breaks.	1
<code>\sublinenumincrement*</code> : Add starred version of <code>\firstlinenum</code> , <code>\linenumincrement</code> , <code>\firstsublinenum</code> , <code>\sublinenumincrement</code> to change both Left and Rightside.	27
v1.6.0.	
General: Add tool and documentation for parallel <code>ledgroups</code>	15
v1.7.0.	
General: Add, as in <code>eledmac</code> , features to make crossrefs with <code>pstart</code> numbers.	1
v1.8.0.	
General: <code>\beginnumbering</code> is defined only on <code>eledmac</code> , not on <code>eledpar</code>	22
<code>\l@dlsnote</code> , <code>\l@drsnote</code> and <code>\l@dcsnote</code> defined only one time, in <code>eledmac</code>	67
Add <code>\beforecolumnseparator</code> and <code>\aftercolumnseparator</code>	8
Add <code>\columnspan</code>	7
Add, as in <code>eledmac</code> , new system of sectioning commands.	1
Add, as in <code>eledmac</code> , option to insert something after <code>\pends</code> / verses.	1
Add, as in <code>eledmac</code> , option to insert something between <code>\pstarts</code> / verse.	1
Change <code>\do@lineR</code> and <code>\do@lineL</code> to allow new sectioning commands.	48
Compatibility with <code>musixtex</code>	1

Debug eledmac sectioning command after using <code>\resumenumbering</code>	1
New sectioning commands, as in eledmac.	16
Suppress <code>\ifl@dsamelang</code> which did not work and was not logical, because both columns could have the same language but not the main language of the document.	71
<code>\Columns</code> : Modify <code>\Columns</code> to enable to add section's title.	76
Suppress <code>\l@dchecklang</code> from <code>\Columns</code>	77
<code>\l@dchecklang</code> : Suppress <code>\l@dchecklang</code> which did not work and was not logical, because both columns could have the same language but not the main language of the document.	71
<code>\Pages</code> : Modify <code>\Pages</code> to enable to add section's title.	84
<code>\pendL</code> : As in eledmac, <code>\pendL</code> can have an optional argument.	46
<code>\pendR</code> : As in eledmac, <code>\pendR</code> can have an optional argument.	47
<code>\print@columnseparator</code> : Move some code of <code>\Columns</code> to <code>\print@columnseparator</code>	78
<code>\pstartR</code> : As in eledmac, <code>\pendL</code> and <code>\pendR</code> can have an optional argument.	43
<code>\sidenotemargin*</code> : <code>\sidenotemargin</code> is now directly defined in eledmac to be able to manage eledpar.	66
Add <code>\sidenotemargin*</code>	66
<code>\theledlanguageR</code> : Correct left/right language setting with polyglossia.	72
v1.8.1.	
<code>\do@lineL</code> : Fix a bug with critical notes at the beginning of a page, (maybe added by v1.8.0) (?).	48
<code>\do@lineR</code> : Fix a bug with critical notes at the beginning of a page, added by v1.8.0 (?).	51
v1.8.2.	
General: Debug <code>\eledxxx</code> with some paper sizes	1
Debug left and side note (bugs added by 1.8.0)	1
<code>\eledpar@error</code> : Errors specific to eledpar send to eledpar handbook	20
<code>\flag@end</code> : <code>\flag@start</code> and <code>\flag@end</code> are now defined only one time for eledmac and eledpar	36
<code>\lineation*</code> : Add <code>\lineation*</code>	25
v1.8.3.	
General: Add <code>\noeledxxx</code> , as in eledmac	1
<code>\doinsidelineRhook</code> : Added <code>\dolineLhook</code> , <code>\dolineRhook</code> , <code>\doinsidelineLhook</code> and <code>\doinsidelineRhook</code>	50
<code>\Pages</code> : Debug blank pages when using optional argument in the last <code>\pend</code>	84
<code>\resumenumberingR</code> : Debug <code>\resumenumberingR</code>	23
v1.9.0.	
General: Add <code>\AtBeginPairs</code> macro.	7
Compatibility with <code>\Xnoteswidthliketwocolumns</code> and <code>\notesXwidthliketwocolumns</code>	1
<code>\ifwidthliketwocolumns</code> : Added <code>widthliketwocolumns</code> option	19
<code>\theledlanguageR</code> : Debug left/right language switching with polyglossia. Do not write in .aux file when setting left/right lines.	72
v1.9.1.	
<code>\ifledRcol</code> : Moved <code>\ifl@dpaging</code> to eledmac	19
v1.10.0.	
General: Compatibility with <code>\AtEveryPstart</code> and <code>\AtEveryPend</code>	1
Restore critical notes in <code>\eledsection</code> in parallel columns (this bug was added in 1.8.2).	1

\Pages: Debug wrong pages splitting when no optional argument is used in last \pend (bug was added in v.1.8.3).	84
Debug wrong parallel pages synchronization when an \edtext falls across two pages.	84
v1.10.1.	
\line@list@stuffR: Revert modification of 1.4.2, which makes bugs with number- ing. Leave vertical mode to solve spurious space before minipage.	35
v1.11.0.	
General: Compatibility of standard footnotes with some biblatex styles.	1
\edtext: \critext and \edtext are now defined only in eledmac.	36
v1.12.0.	
General: Compatibility with Lua \TeX RTL languages.	1
\Columns: Add \l@dprintingcolumnstrue	76
\edlabel: \edlabel and \edindex works now with hyperref when using eled- par.	66
\edlabel is now defined only one time for both eledmac and eledpar	66
\Pages: Add \l@dprintingpagestrue	84
\print@eledsectionL: Compatibility with Lua \TeX RTL languages.	49
\print@eledsectionR: Compatibility with Lua \TeX RTL languages.	52
\print@lineL: Compatibility with Lua \TeX RTL languages.	49
v1.12.1.	
\print@eledsectionL: Fixes bug with Lua \TeX RTL \eledsection.	49
v1.13.0.	
General: Enable the use of optional argument of & in astanza environment.	69
Fix bug in shiftedpstarts when size difference between pstarts is very important.	1
With parallel pages, long notes can now flow from the Left to the right side and from the Right to the left side.	1
\clearl@drihtpage: Use \newpage instead of \clearpage.	91
\ifledRcol: Remove false boolean settings which are not needed.	19
\Pages: Prevent false overfull hboxes when using \Pages outside of pages environ- ment.	84
When using shiftedpstarts option, a \l@dleftbox with a null height will advance the \pagetotal in any case.	84
v1.13.1.	
\correct@footinsX@box: Call \correct@footinsX@box and \correct@Xfootins@box directly in \print@notesX@forpages and \print@Xnotes@forpages.	61
Correct \correct@footinsX@box and \correct@Xfootins@box	61
\Pages: Prevent false empty page after \Pages (bug added in 1.13.0)	84
v1.14.0.	
General: Fix bug with line number position when using \eledsection and similar commands for RTL texts with Lua \TeX	1
The \newifs are not followed by boolean values set to false, because it is the \TeX default setting.	1
v1.15.0.	
General: Add \AtEveryPstartCall.	1
Add sameparallelpagenumber option.	9
Fix vertical spurious space before right \eledchapter (bug added in v1.13.0).	1
Prevent vertical space when using \AtEveryPstart or \AtEveryPend with a command which prints nothing	1

\do@actions@nextR: Add action 1008 and 1009	53
\inserthangingsymbolR: Prevent more efficiently the column separator from shifting when a verse is hanging	68
\lineationR: As \lineation, \lineationR automatically set the \pstart-infootnote.	25
\n@num: \n@num defined only one time for both Eledmac and Eledpar.	32
\skipnumbering: \skipnumbering defined only one time for both Eledmac and Eledpar	36
v1.16.0.	
General: Error message when calling \Pages inside 'pages' environment and \Columns inside 'pairs' environment.	1
Error message when starting a Leftside/a Rightside while the previous one has not been yet typeset.	1
Error message when using \beginnumbering...\endnumbering without \pstart.	1
Fix bug with nofamiliar / nocritical option of eledmac.	1
New package option sameparallelpagenummer to have the same page number for both left and right side.	1
\newseries@par: Fix bug with \onlysideX.	37
v1.16.1.	
General: Write information about line-list file version in the correct file.	1
v1.16.2.	
General: Fix bug when adding empty lines before a \pend in combination with some specific penalties setting.	1
v1.17.0.	
General: Add compatibility of optional argument of \pstart/\pend and \AtEveryPstart/\AtEveryPend with two columns mode.	1
v2.0.0.	
\@adv: \@adv defined only in reledmac.	30
\@lab: \@lab defined only in eledmac.	66
\@ref@regR: \@ref defined only in reledmac, code specific to right side moved in \ref@regR.	32
\@set: \@set defined only in reledmac.	31
General: \@nl is now defined only in reledmac.	29
\ifbypage@ and \ifbypstart@R defined in eledmac.	25
Fix some bugs with 'sameparallelpagenummer' option.	1
Many code refactored and moved to reledmac.	1
Package's name becomes reledpar.	1
Totally new implementation of 'sameparallelpagenummer' option.	1
\advanceline: \advanceline defined only in reledmac.	36
\bbl@set@language: Patch \bbl@set@language instead of redefining it ..	71
\do@lockonR: \do@lockon defined only in reledmac.	31
\endlock: \startlock and \endlock defined only in reledmac.	36
\endsub: \startsub and \endsub defined only in reledmac.	36
\fix@page: \fix@page is defined only once in reledmac	30
chapterinpages: Deleting the old system of managing parallel chapter, keep only the new one with \patchcmd.	41
\l@d@set: \l@d@set defined only in reledmac.	31
\l@dbfnote: \l@dbfnote defined only in reledmac.	68
\line@marginR: \linenummargin now defined only once time in reledmac.	26

<code>\normalbfnoteX</code> : <code>\normalbfnoteX</code> defined only in <code>reledmac</code>	68
<code>\page@action</code> : <code>\page@action</code> defined only in <code>reledmac</code>	31
<code>\read@linelist</code> : <code>\read@linelist</code> is defined only once time in <code>\reledmac</code>	29
<code>\set@line</code> : <code>\set@line</code> defined only in <code>reledmac</code>	36
<code>\set@line@action</code> : <code>\set@line@action</code> defined only in <code>reledmac</code>	31
<code>\setline</code> : <code>\setline</code> defined only in <code>reledmac</code>	36
<code>\setlinenum</code> : <code>\setlinenum</code> defined only in <code>reledmac</code>	36
<code>\skip@lockoff</code> : <code>\do@lockoff</code> defined only in <code>reledmac</code>	31
<code>\sub@action</code> : <code>\sub@action</code> defined only in <code>reledmac</code>	31
<code>\sublinenumincrement*</code> : <code>\firstlinenum</code> , <code>\linenumincrement</code> , <code>\first-</code> <code>sublinenum</code> , <code>\sublinenumincrement</code> are now defined only in <code>reled-</code> <code>mac</code>	27
<code>\theledlanguageR</code> : Patch <code>\otherlanguage</code> instead of redefining it.	72
v2.1.0.	
General: Fix bug when using <code>\eledsection</code> and related on right pages when page width is short.	1
Fix bug when using <code>\pagenumbering</code> with <code>memoir</code> (bug added in v2.0.0).	1
Fix bug with <code>\setparledgroupnotespacing</code> with the <code>shiftedpstarts</code> option.	1
Fix incompatibility between optional argument of <code>\pstart</code> and <code>\numberp-</code> <code>starttrue</code>	1
Options to custom empty right page before <code>\Pages</code>	1
v2.2.0.	
General: <code>astanza</code> environment can take an optional argument, which will be the op- tional argument of <code>\pstart</code> started by this environment.	1
New tools to number stanza	1