fnlineno.sty

—

Numbering Footnote Lines*

March 8, 2026

Abstract

’fnlineno.sty’ extends `lineno` (created by Stephan I. Böttcher) such that even `\footnote` lines are numbered and can be referred to using `\linelabel`, `\ref`, etc.

Making the package was motivated as support for *critical editions* of *printed works with footnotes* as opposed to scholarly critical editions of *manuscripts.* For this purpose, an extension ’edfnotes’ of the `ednotes` package for critical editions, building on ’fnlineno’, is provided by the *ednotes* bundle.

’lineno.sty’ has also been used for the revision process of *submissions.* With ’fnlineno.sty’, reference to footnotes in the submitted work may become possible.

As to *implementation:* 1. Some included tools for *storing and restoring global settings* may be “exported” as standalone packages later. 2. The method of typesetting footnotes on the main vertical list may later lead to applying the line numbering method to several *parallel* texts (with footnotes) and to `inner` material such as table cells.

Keywords: line numbers; footnotes, pagewise, critical editions, revision

Contents

---

[*] This document describes version v0.55 of fnlineno.sty as of 2011/01/07.

1 Usage and Features

1.1 Package File Header (Legalize)

```
\NeedsTeXFormat{LaTeX2e}[1994/12/01]
\ProvidesPackage{fnlineno}[2011/01/07 v0.55
                           numbers to footnote lines (UL)]

%% Copyright (C) 2010 Uwe Lück
%%
%% This file can be redistributed and/or modified under
%% the terms of the LaTeX Project Public License; either
%% version 1.3c of the License, or any later version.
%% The latest version of this license is in
%%     http://www.latex-project.org/lppl.txt
%% We did our best to help you, but there is NO WARRANTY.
%%
%% Please report bugs, problems, and suggestions via
%%
%%   https://github.com/latex-lineno/lineno
```

%%

This work has been supported by the Deutsche Forschungsgemeinschaft (DFG), organized by Prof. Dr. Dr. Christian Tapp at Ruhr-Universität Bochum, Germany. Christian also has constructed some critical tests.

1.2 Installing and Calling

The file 'fnlineno.sty' is provided ready, installation only requires putting it somewhere where TEX finds it (which may need updating the filename data base).[1]

As usually, 'fnlineno.sty' is loaded by

```
\usepackage{fnlineno}
```

below the `\documentclass` line and before `\begin{document}`.

1.3 Limitations

v0.55 should really work the way users expect, but please consider:

1. Nothing is known about compatibility with packages (other than 'manyfoot' and 'bigfoot') providing footnote features beyond standard LATEX.
2. `\lipsum[<opt-arg>]` in main text produces a different number of paragraphs . . .
3. v0.41 tried supporting `\pagebreak` in footnotes for manual control of splitting footnotes. However, it wrongly assumed that `\pagebreak[4]` forces a footnote split, cf. Section 2.5.3; users better still don't use `\pagebreak` in footnotes!
4. Much of the code is "guessed" without complete knowledge of TEX internals and without having tested many possible cases.
5. *Local* switching to "pagewise" numbering won't be possible for a while; we rather assume that you *always* want "pagewise" numbering.
6. Nothing has been tried to offer choices about the *style* of numbering footnotes.

---

[1] `http://www.tex.ac.uk/cgi-bin/texfaq2html?label=inst-wlcf`

2 Implementation

2.1 Terms

“**OTR**” is short for *“output routine”*, “**MVL**” is short for *“main vertical list”*.

2.2 Basic Strategy

LaTeX’s `\@footnotetext` writes the footnote text into the insertion register. For numbering the footnote lines, *we here* do not execute this `\@footnotetext` immediately after placing `\@footnotemark`, but postpone its `\insert` a little so it is executed only after the main text paragraph has been broken into lines. Right below the line that contains the footnote mark, a special new “slot” of the **OTR** is called that interchanges “the page so far” with the footnote text. When the latter has been typeset, another “slot” of the OTR puts “the page so far” back to the MVL and immediately after that fills the footnote text as just typeset on the MVL into the `\insert` register.

Passing footnotes from horizontal mode to vertical mode resembles ’lineno”s `\PostponeVadjust`, but a different **list** `\FNLN@list` must store code (a) for the footnote **mark** and (b) for the footnote **text**.

2.3 Package Options

A package option —[check-latex]— for checking vital LaTeX internals may once be offered (TODO 2010/12/12) . . .

```
\newif\if@FNLN@check@
\DeclareOption{check-latex}{\@FNLN@check@true}
\ProcessOptions
```

2.4 Footnote Commands

2.4.1 Standard Footnotes

The following macro `\FNLN@ltx@fntext` is a copy of LaTeX’s `\@footnotetext` that we are varying. It may be used for a check if the `\@footnotetext` that ’fnlineno.sty’ encounters is the one expected (TODO). In line numbering mode, this code may never be needed all at once, rather we will have to see which material must be used at which point of our unusual way of processeing footnotes.

```
\if@FNLN@check@
  \long\def\FNLN@ltx@fntext#1{\insert\footins{%
      \reset@font\footnotesize
      \interlinepenalty\interfootnotelinepenalty
      \splittopskip\footnotesep
      \splitmaxdepth \dp\strutbox \floatingpenalty \@MM
      \hsize\columnwidth \@parboxrestore
      \protected@edef\@currentlabel{%
```

```
          \csname p@footnote\endcsname\@thefnmark
      }%
      \color@begingroup
        \@makefntext{%
          \rule\z@\footnotesep\ignorespaces#1\@finalstrut\strutbox}%
      \color@endgroup}}%
\fi
```

2.4.2 Modifying Footnote Commands

In order to number `\footnote` lines and make `\linelabel` available in footnotes, it seems to suffice (with standard LaTeX) to **redefine** the internal `\@footnotetext`. In line numbering mode, `\@footnotetext` will act as `\FNLN@text`, (*i*) placing a "signal" **output penalty** below the current line via `\vadjust` and (*ii*) appending the footnote text to the **list** `\FNLN@list` of **footnote texts**.

`\FNLN@@text` stores the `\@footnotetext` found, we might check if it is `\FNLN@ltx@fntext` ...

```
\let\FNLN@@text\@footnotetext
\def\@footnotetext{%
    \ifLineNumbers  \expandafter \FNLN@text
    \else           \expandafter \FNLN@@text
    \fi}
\def \FNLN@text {%                      %% 2010/12/31 arg read later
    \vadjust{\penalty-\FNLN@M@swap@codepen}%
```

Standard LaTeX's `\@footnotetext` expands `\@thefnmark` to produce the footnote mark at the page bottom, right after it has been determined for the mark in the main text. *Here* the footnote text will be typeset only when *other* footnote marks may have been formed for typesetting the main text paragraph before. In the **footnote list** macro `\FNLN@list`, the (&`\protect`ed) *current* expansion ¡mark¿ of `\@thefnmark` is stored as an item preceding the footnote text ¡text¿. One footnote entry in `\FNLN@list` thus has the form '`<mark>\@lt<text>\@lt`'. LaTeX's internal `\g@addto@macro` is used to *append* an entry to the list (at the right). The OTR will later take the entries from the left of the list.

The argument of the auxiliary/temporary `\@tempa` will contain the footnote text and thus must be able to carry `\par` tokens. We therefore need a `\long` version of `\protected@edef`:

```
    \let\@@protect\protect
    \let\protect\@unexpandable@protect
    \afterassignment\restore@protect
    \long \edef \@tempa ##1{%
         \noexpand\g@addto@macro \noexpand\FNLN@list {%
             \@thefnmark \noexpand\@lt ##1\noexpand \@lt}}%
```

… issuing ‘`\g@addto@macro\FNLN@list{<mark>\elt<text>\@lt}`’ …

```
   \@tempa                                %% reads arg
}
```

Here we initialize `\FNLN@list`:

```
\let\FNLN@list\@empty
```

2.5 Output Routines

2.5.1 ’lineno”s Output Routine

The following is a copy of ’lineno”s OTR that we are varying. It may be used for a check if the OTR that ’fnlineno.sty’ encounters is the one expected (TODO).

```
\if@FNLN@check@
  \def\FNLN@lno@output {%
    \LineNoTest
    \if@tempswa
      \ifnum\outputpenalty=-\@Mllbcodepen
        \WriteLineNo
      \else
        \ifnum\outputpenalty=-\@Mppvacodepen
          \PassVadjustList
        \else
          \LineNoLaTeXOutput
        \fi
      \fi
    \else
      \MakeLineNo
    \fi
  }
```

The “signal penalties” used here are

```
  \mathchardef\FNLN@M@llbl@codepen=11111
  \mathchardef\FNLN@M@ppva@codepen=11112
\fi
```

Their names should mean `\linelabel` code penalty” and `\PostponeVadjust` code penalty.”

`\TheLineNoLaTeXOutput`: It turns out to be inconvenient here that ’lineno’ sacrifices access to the *primitive* `\output` (‘‘`\@tempa`”; TODO: auxiliary package before loading ’lineno’!?; later change ’lineno.sty’ indeed). So to change the OTR we use `\LineNoLaTeXOutput` as a hook for adding additional cases of `\outputpenalties`. We take a copy of `\LineNoLaTeXOutput` here.

```
\let\TheLineNoLaTeXOutput\LineNoLaTeXOutput
```

2.5.2 Tools for Temporary Parameter Changes

`\GStoreReg{<register>}` (or `\GStoreReg<register>` when ¡register¿ is a single token—'`\count0`' being a counterexample . . .) stores the current content of ¡register¿ (*globally*) as an internal macro so that it can be restored later by

`\RestoreReg{<register>}` (or `\RestoreReg<register>`)

or *globally* by

`\GRestoreReg{<register>}` (`\GRestoreReg<register>`)

(The OTR runs in a local group!—Recall that assignments to "special dimens"—TEXbook p. 271—are automatically global.) ¡register¿ is something that can be prefixed by `\the` to read its content and to which you can assign a value ¡value¿ by '¡register¿¡value¿'. (TODO: could also be some `\catcode`!)

```
\newcommand*{\GStoreReg}[1]{%
    \expandafter \xdef \csname GS\string#1\endcsname {\the #1}}
\newcommand*{\RestoreReg}[1]{#1\csname GS\string#1\endcsname \relax}
\newcommand*{\GRestoreReg}{\global\RestoreReg}
```

`\GStoreSetReg{<register>}{<value>}` assigns ¡value¿ to ¡register¿ (locally) after executing `\GStoreSet`, `\GStoreGSetReg` does the same *globally* (and still argument braces aren't needed when a single token refers to the register).

```
\newcommand*{\g@storesetreg}[3]{\GStoreReg{#2}#1#2#3\relax}
\newcommand*{\GStoreSetReg} {\g@storesetreg\relax}
\newcommand*{\GStoreGSetReg}{\g@storesetreg\global}
```

(These preliminaries might go into an own new package, TODO! + loop on list of ¡register¿s . . .)

2.5.3 The basic hook

We use two more penalties triggering the "MVL swaps:"

```
\mathchardef\FNLN@M@swap@codepen  =11113
\mathchardef\FNLN@M@insert@codepen=11114
```

v0.41 deals with `\pagebreak` in footnote texts, using a flag `\if@FNLN@sw@` that must be set globally. It turned out not to work properly; however, the new switch has served a different purpose for "continuous line numbering," cf. section 2.6.

```
\newif\if@FNLN@sw@  \global\@FNLN@sw@false  %% v0.41
```

When a `\pagebreak` triggers the OTR while typesetting the footnote text, the page content is collected in a box `\FNLN@holdft`:

```
\newsavebox\FNLN@holdft                     %% v0.41
```

Using `\LineNoLaTeXOutput` for hooking into the OTR:

```
\renewcommand*{\LineNoLaTeXOutput}{%
  \ifnum\outputpenalty=-\FNLN@M@swap@codepen
    \SwapFootnoteMain
  \else
    \ifnum\outputpenalty=-\FNLN@M@insert@codepen
      \InsertFootnote
    \else
      \if@FNLN@sw@                          %% v0.41
%        \showthe\outputpenalty     %% 2010/12/20
        \global\setbox \FNLN@holdft \vbox{%
          \unvbox\FNLN@holdft
```

TODO from v0.41: `\pagebreak[4]` does not seem to force (reliably) splitting a footnote; if the footnote is not split here, at present the `\baselineskip` is lost, see the footnote paragraph starting with `C` in `edfndemo.pdf` as of 2010/12/21. We would need some measuring … `\pagebreak` might be redefined … resembling LaTeX's `\@specialoutput`!

```
          \unvbox\@cclv
```

TODO same problem here, see the footnote paragraph starting with `D` in `edfndemo.pdf` as of 2010/12/21.

```
          \penalty\outputpenalty}%
        %% TODO reset page book-keeping!?   %% v0.41
      \else
        \TheLineNoLaTeXOutput       %% "the real \LineNoLaTeXOuput"
      \fi
    \fi
  \fi
}
```

An idea: Instead of so many `\ifnum`, use

```
\csname<chars>\the\outputpenalty\endcsname
```

… in 'lineno.sty', when you really have a broad range of `\outputpenalties` useful to be described by `\ifnum` range checks …

2.5.4 Typesetting the Footnote Text

`\SwapFootnoteMain` is the slot of the OTR that our modified `\@footnotetext` calls with `\outputpenalty` = −`\FNLN@M@swap@codepen`. The "column so far" is stored in a new box register `\FLNL@holdcol`.

```
\newsavebox\FNLN@holdcol
\newcommand*{\SwapFootnoteMain}{%
    \global \setbox\FNLN@holdcol \vbox{\unvbox\@cclv}%
```

(… cf. `\@holdpg` in LaTeX.)

The entire text of a footnote is typeset on top of the MVL. `\vsize` is maximized temporarily to avoid that the footnote text is broken across pages.

```
    \GStoreGSetReg\vsize\maxdimen
```

However, the user may want to use `\pagebreak` in a footnote in order to control manually where a "long" footnote is split. v0.41 tries to support this:

```
    \global\@FNLN@sw@true                  %% v0.41
```

… cf. Section 2.5.3.

There shouldn't be any `\topskip`, the space on top of a footnote is controlled by `\footnotesep` entirely:

```
    \GStoreGSetReg\topskip\z@skip
```

(`\nointerlineskip` as well as setting `\topskip` locally instead fails … according to `\showlists` …)

Resetting `\pagegoal` (why doesn't it switch to `\vsize` = `\maxdimen` automatically?), `\pagetotal`, and the other "special dimens" (TeXbook p. 271; rather experimental … I think it is important to restore them later …)

```
    \GStoreSetReg\pagegoal \vsize
    \GStoreSetReg\pagetotal\z@
    \GStoreSetReg\pagestretch\z@
    \GStoreSetReg\pagefilstretch\z@
    \GStoreSetReg\pagefillstretch\z@
    \GStoreSetReg\pagefilllstretch\z@
    \GStoreSetReg\pageshrink\z@
    \GStoreSetReg\pagedepth\z@
```

We must choose certain settings from `\@footnotetext` such as font:

```
    \reset@font\footnotesize
    \interlinepenalty\interfootnotelinepenalty
```

LaTeX's `split` things here are relevant at `\insert\footins` only: (TODO!?)

```
%     \splittopskip\footnotesep
%     \splitmaxdepth \dp\strutbox \floatingpenalty \@MM
    \hsize\columnwidth \@parboxrestore
```

The previous lines were from LaTeX's `\@footnotetext`. Now we need to restore the `\@thefnmark` that belongs to the current footnote text. We use a macro that tears two items from `\FNLN@list` and executes the rest of LaTeX's `\@footnotetext`:

```
    \expandafter \FNLN@typeset \FNLN@list \@@
%     \showthe\vsize
```

… so a `\vsize` assignment without `\global` is noted here, and an analogous `\topskip` assignment is not!? TODO …

```
}
```

`\FNLN@typeset` first removes something from the list of footnotes, similarly to LaTeX's `\@xnext` and 'lineno''s `\@LN@xnext`, then executes a remaining portion of LaTeX's `\@footnotetext`. The footnote text may contain `\par` tokens, so the definition must be `\long`:

```
\long\def \FNLN@typeset #1\@lt #2\@lt #3\@@{%
    \gdef\FNLN@list{#3}%
    \def\@thefnmark{#1}%
```

This was our own, and next LaTeX continues:

```
    \protected@edef\@currentlabel{%
       \csname p@footnote\endcsname\@thefnmark
    }%
    \color@begingroup
```

We insert starting the 'lineno' settings …

```
      \linenumbers
      \setfootnotelinenumbers                   %% 2010/12/25
```

… LaTeX again (v0.41 exports dealing with closing `\par` to `finstrut.sty`):

```
      \@makefntext{%
        \rule\z@\footnotesep\ignorespaces
```

We replace `#1` by `#2\par` (`\linenumberpar`), so we really need `finstrut.sty`:

```
        #2\par
        \@finalstrut\strutbox}%
    \color@endgroup
```

Now we trigger the "swap back slot" of the OTR:

```
    \penalty-\FNLN@M@insert@codepen
}
\RequirePackage{finstrut}
```

2.5.5 Insert the Footnote Text

`\InsertFootnote` is the slot of the OTR that executes `\insert\footins` with the numbered footnote text. The `column` so far stored in `\FNLN@holdcol` is put onto the top of the MVL, and then parts of LaTeX's `\@footnotetext` are performed that haven't been done earlier, applied to the footnote text that the OTR should have found in `\box255`. Before however, the previous `\topskip`, `\vsize`, and the `\page`… book-keeping parameters are restored:

```
\newcommand*{\InsertFootnote}{%
    \GRestoreReg\topskip    \GRestoreReg\vsize
```

(… *global* restoring of `\vsize` proved vital with 'edfndemo' 2010/12/17 …)

```
    \RestoreReg \pagegoal     \RestoreReg\pagetotal
    \RestoreReg \pagestretch
    \RestoreReg \pagefilstretch
    \RestoreReg \pagefillstretch
    \RestoreReg \pagefilllstretch
    \RestoreReg \pageshrink   \RestoreReg\pagedepth
    \unvbox\FNLN@holdcol
    \insert\footins{%
        \splittopskip\footnotesep
        \splitmaxdepth \dp\strutbox \floatingpenalty \@MM
```

Support of `\pagebreak` with v0.41:

```
        \unvbox\FNLN@holdft                 %% v0.41
        \unvbox\@cclv}%
    \global\@FNLN@sw@false                  %% v0.41
```

With v0.5, global settings for "pagewise" numbering must be restored:

```
    \unsetfootnotelinenumbers
}
```

2.6 Continuous Numbering

2.6.1 Goal

With v0.5, for the first time we try to get a "pagewise" numbering such that, if a main text line has a footnote, (*i*) its printed number is just the natural successor of the printed number of the previous main text line (instead of continuing previous numbering with the lines of the footnote first), and (*ii*) the printed numbers of footnote lines just continue the printed numbers of the main text lines. This `obvious` desirement is not easy to achieve; already pagewise numbering of main text lines, without numbering footnote lines, has been somewhat ingenious.

2.6.2 How to Number Lines Pagewise

The basic idea of 'lineno''s pagewise numbering is:

1. Each numbered line of the document is identified by a unique counter value, an "absolute" number.
2. For each page (and column), the range of absolute line numbers occurring on them is recorded (or actually: the first and the last number).
3. The "public," "human-readable" ("pagewise") format of a given absolute line number l is generated by (*i*) finding the page (and column) with first

> number n and last number k such that $n \leq l \leq k$, (*ii*) "printing" $l-n+1$ in "columnwise" mode, otherwise $l-m+1$ where m is the first absolute line number in the left-hand column of the same page.

Generating the "pagewise" representation for a given absolute line number l thus may be summarized as *finding the corresponding* **offset** *value* to be subtracted (n, $n+1$, m, or $m+1$ ...).

When *footnote* lines are to be numbered as well, a little problem is the order in which main text and footnote lines increment the absolute counter. 'lineno''s mechanism for this is started immediately after a paragraph has been broken into lines. Each line of the paragraph then calls a macro generating the line number. 'fnlineno' now interrupts numbering of main text lines at a line issuing a footnote. The footnote text is typeset, including numbering its lines at each end of a footnote paragraph. When the footnote text has been sent into the `\insert` register, numbering of main text lines is resumed.

Up to v0.4 (a development version), we used the *same* absolute counter for main text and footnote lines. When a page p has more than one main text line and the first one has a long footnote continued on the next page $p+1$, there is no "range" of absolute line numbers characterizing page p any more, because the greatest absolute line number of page p exceeds the absolute line numbers of the footnote continued on page $p+1$.

'lineno''s procedure can be revived by numbering main text lines and footnote lines independently from each other. We use *two* absolute counters, one is incremented with main text lines only, the other with footnote lines only. Numbering of main text lines just will not be affected by numbering of the footnote lines.

Almost the same will hold for footnote lines. Each page (and column) will have a characteristic "range" of absolute footnote line numbers $\{n, \ldots, k\}$. The only notable difference will be that for footnote line l we print $(l-n+1)+(K-N+1) = (K+l)-(N+n)+2$ instead of $l-n+1$—where $\{N, \ldots, K\}$ is the range of *main text* line numbers of the page (and column).

The previous discussion of **generating** the printed line number from its absolute version has assumed that corresponding **offset** values have been given somehow, or that the "line number ranges" for pages are known from somewhere. In fact, these ranges are **computed** at the **start** of a LaTeX run *before* typesetting, when reading the `.aux` file for the first time. They are used in the entire document. While typesetting, each numbered line of main text leaves a record of its absolute number and page number in the new version of the `.aux` file that the run creates, a two-parameter macro `\@LN`. With 'fnlineno.sty', there will be new `\@FLN` entries of the same type. These `.aux` entries are used for building the page range data for the next run. When the document source has been changed, at least *two runs* will usually be required to get correct line numbers in page margins, and *another* run will be needed so references to line numbers by `\ref` and `\linelabel` are correct.

2.6.3 Summary of Changes

Variants of 'lineno.sty''s code for "pagewise" numbering are following. Sometimes we generalize `pagewise` stuff from 'lineno' and re-implement pagewise numbering of main text lines as a special case, the other special case being numbering of footnote lines.

Five things need modifications:

Building page info macros: Processing `\@LN` and `\@FLN` `.aux` entries will use shared building macros, the difference is obtained by switching **name spaces**. (It may be notable that a page may get one info macro for main text and another for footnote text, if it contains footnote text.)

Logging: While typesetting, the shared logging macro is switched to write either `\@LN` or `\@FLN` to the `.aux` file. Also, `\c@linenumber` may refer to either the main text or to the footnote text counter.

Generating "pagewise" format: The choice of `\c@linenumber` also determines which counter is incremented, and again name spaces for page info macros are switched. For footnote lines, a tail macro for adding the number of main text lines will be activated.

Referencing: The `.aux` file may have entries from `\linelabel` containing large numbers from an "absolute" counter. In generating the "human-readable" number, it must be known whether it is a main text or a footnote line number. An additional complication is referring to a main text line from a footnote and vice versa—thinking of global changes in generating the number. Or even think of the case referring from unnumbered text to numbered text! (I have wondered before if the entry couldn't be the ready human-readable number, TODO!)

Lists of "vertical tasks": 'lineno.sty' (v4) has introduced two lists of tasks that were issued in horizontal mode but only can be completed after breaking a paragraph into lines: one for `\linelabel`s and one for `\vadjust` items that must wait until the line number has been attached. It is essential that the tasks are processed in the same order in vertical mode as they were issued in horizontal mode. As we are now interrupting processing of main text paragraphs for processing footnotes, tasks for footnote text must be lined up in separate lists than tasks for main text. This is indeed essential for the previous issue of getting `\linelabel` work in footnotes as well as in main text.

2.6.4 Info Building

`\@LN`, `\@FLN`, and `\@FNLN` are processed at reading the `.aux` file before typesetting only. The **interface** to **generating** "pagewise" and footnote line numbers just are `\LN@Pfirst` and `\FLN@Pfirst`, eventually pointing to the first page/column with numbered main text lines or footnote lines, resp.

```
\def \FLN@Pfirst {\nextLN\relax}
```

This initialization of `\FLN@Pfirst` is just the same as the one of `verb|`LN@Pfirst— in 'lineno.sty'; their expansions are changed as soon as such a page is found, replacing the `\relax` by the corresponding page info macro.

`\LN@Pfirst` and `\FLN@Pfirst` are passed to `\testFirstNumberedPage` via the hook `\FNLN@first@numbered` that by default is the same as `\LN@first`:

```
\def \FNLN@first@numbered {\LN@Pfirst}
```

(oh, it must be `\def` here to recognize the change . . .). This must be changed by `\setfootnotelinenumbers` (`\let` then, as when called the change will have happened).

Moreover, they are passed to `\NumberedPageCache` (the page info macro where a search starts, "current" page/column) as its initialization; the "generating" macros then change the latter macro following `\nextLN` in the page info macros.

In this sense, no other "name space switching" is needed for communication with other functions.

'lineno.sty' has changed `\LastNumberedPage` globally . . . the last page with numbered *footnote* lines may well be another one than the last page with numbered *main* text lines . . . But fortunately, also `\LastNumberedPage` is needed in reading the `.aux` before typesetting only (`\@onlypreamble` is LaTeX's disabling command):

```
\@onlypreamble\LastNumberedPage
```

In 'lineno.sty', we have `\def\LastNumberedPage{first}`. We need the same for the footnote variant `\FNLN@last@numbered` (to be handled globally!):

```
\global   \let \FNLN@last@numbered \LastNumberedPage
\@onlypreamble \FNLN@last@numbered
```

`\@FNLN{<names>}<last-numbered>{<line>}{<page>}`

generalizes 'lineno.sty"s `\@LN{<line>}{<page>}` to re-implement it. There is an additional parameter argument ¡names¿ for choosing name spaces and a parameter ¡last-numbered¿ for choosing the macro storing the "last numbered page." (An argument without braces expects a macro name.)

```
\newcommand* \@FNLN [4]{{%
    \expandafter\@@LN
        \csname #1#4C\@LN@column \expandafter\endcsname
        \csname #1O#4\endcsname
        {#3}{#4}{#1}{#2}}}
\@onlypreamble\@FNLN
```

As in 'lineno.sty' `\@LN` calls `\@@LN`, a new variant of `\@@LN` is called by `\@FLN` here, but it gets one additional parameter for passing ¡names¿ and another for passing ¡last-numbered¿ from `\@FLN`. So the new syntax is

`\@@LN<info><first-page-line>{<line>}{<page>}{<names>}<last-numbered>`:

```
\renewcommand* \@@LN [6]{%
```

```
  \ifx#1\relax
    \ifx#2\relax\gdef#2{#3}\fi
    \expandafter\@@@LN\csname #5#6\endcsname#1%
    \xdef#1{\lastLN{#3}\firstLN{#3}%
            \pageLN{#4}{\@LN@column}{#2}\nextLN\relax}%
  \else
    \def\lastLN##1{\noexpand\lastLN{#3}}%
    \xdef#1{#1}%
  \fi
  \xdef#6{#4C\@LN@column}}
\@onlypreamble\@@LN
```

'lineno.sty"s `\@@@LN` does not need any adjustment.

'lineno.sty"s `\@LN{<line>}{<page>}` is reimplemented as

```
\def \@LN {\@FNLN{LN@P}\LastNumberedPage}
```

—so `\@LN` really does the same as before, including name spaces.

`\@FLN{<line>}{<page>}` is the other special case of the new `\@FNLN`—an `F` precedes the earlier names, and `\FNLN@last@numbered` is the storing macro initialized above:

```
\def \@FLN {\@FNLN{FLN@P}\FNLN@last@numbered}
```

For logging, we make both unexpandable:

```
% \AtBeginDocument{\let\@LN\relax \let\@FLN\relax}
```

. . . but this way nothing appears in the file!? TODO . . .

```
\@onlypreamble\@LN  \@onlypreamble\@FLN
```

For reading the `.aux` finally, we do what 'lineno' does with `\@LN`:

```
\AtEndDocument{\let\@FLN\@gobbletwo}
```

2.6.5 Tool for Reusing Global Operations with Macros

'lineno.sty' v4 provides list handling (changing lists globally) and global changes of `\NumberedPageCache`. We want to use them in "main text" mode as well as in "footnote" mode. To use such an operation on ¡ln-macro¿ for ¡fln-macro¿, we `\global\let<ln-macro><fln-macro>`, apply the operations, and finally `\global\let<fln-macro><ln-macro>`. However, we are not only interested in how ¡fln-macro¿ is changed this way, rather ¡ln-macro¿ also is used as input for some operations, and we can choose which ¡fln-macro¿ should be used as input. To switch from working on/with ¡fln-1¿ to ¡fln-2¿ using ¡ln-macro¿ with an option to use ¡fln-1¿ later again, a tool `\GStoreUse<ln-macro><fln-1><fln-2>` is provided (should render later switchings much better readable):

```
\newcommand* \GStoreUse [3]{\global\let#2#1\global\let#1#3}
```

I.e., current content of #1 is stored in #2, then #1 attains the content of #3.

2.6.6 General Settings for Typesetting Stage

Oh my dear, it seems that all the switching for the footnote variant of `pagewise` must be global (I can't find something useful using `\aftergroup` quickly). Therefore, I render 'lineno"s `\setpagewisenumbers` acting globally:

```
\renewcommand*\setpagewiselinenumbers{%
   \global\let \theLineNumber  \thePagewiseLineNumber
   \global\let \c@linenumber   \c@pagewiselinenumber
   \global\let \makeLineNumber \makePagewiseLineNumber
}
```

I just force this, hehe . . .

```
\setpagewiselinenumbers
```

As a counterpart to `\c@pagelinenumber`, `\c@footnotelinenumber` is reserved for the absolute footnote line numbers:

```
\newcount\c@footnotelinenumber
```

`\FNLN@@cache` stores `\NumberedPageCache` as from "main" mode:

```
\let \FNLN@@cache \NumberedPageCache
```

`\FNLN@cache` stores `\NumberedPageCache` as from "footnote" mode; its initial content is the counterpart or analogue to `\LN@Pfirst`:

```
\def \FNLN@cache {\FLN@Pfirst}
```

`\FNLN@foot@cache` and `\FNLN@main@cache` switch `\NumberedPageCache`:

```
\def \FNLN@foot@cache {%
    \GStoreUse \NumberedPageCache \FNLN@@cache \FNLN@cache}
\def \FNLN@main@cache {%
    \GStoreUse \NumberedPageCache \FNLN@cache \FNLN@@cache}
```

`\FNLN@labels` will be the counterpart to 'lineno.sty"s `\@LN@labellist`:

```
\global\let \FNLN@labels \@empty
```

`\FNLN@vadjusts` will be the counterpart to 'lineno"s `\@LN@vadjustlist`:

```
\global\let \FNLN@vadjusts \@empty
```

Settings for footnote line numbers first resemble `\setpagewiselinenumbers`; but more changes are needed, and results from main text numbering must be stored. Some of the settings are needed *locally* for generating numbers for labels, collected in `\setgetfootnotelinenumbers`; for this purpose nothing must be stored explicitly:

```
\newcommand* \setgetfootnotelinenumbers {%
```

Change of `\theLineNumber` is omitted as we are *reading*, not writing a label.

```
    \let\c@linenumber\c@footnotelinenumber
%     \let\makeLineNumber\makeFootnoteLineNumber
```

But in fact, `\makeFootnoteLineNumber` and `\makePagewiseLineNumber` will be the same. The difference is made by the choice of `\FNLN@first@numbered` and `\NumberedPageCache` for the line range searches.

```
    \let \FNLN@first@numbered \FLN@Pfirst
    \let \FNLN@finish \FNLN@add
}
```

`\setfootnotelinenumbers` performs all the settings for typesetting footnotes in line numbering mode *globally*, including storing results from typesetting main text:

```
\newcommand* \setfootnotelinenumbers {%
  \globaldefs\@ne
```

The previous line also renders `\setgetfootnotelinenumbers` global:

```
    \setgetfootnotelinenumbers
```

`\theLineNumber` is used for `\linelabel` entries. `\thePagewiseLineNumber` is replaced by `\theFootnoteLineNumber`:

```
    \let\theLineNumber\theFootnoteLineNumber
```

Logging to `.aux`:

```
    \def \FNLN@log {\string\@FLN}%
```

Starting range search: `\NumberedPageCache`

```
    \FNLN@foot@cache
```

Reusing 'lineno''s task list operations:

```
    \GStoreUse \@LN@labellist \FNLN@@labels \FNLN@labels
    \GStoreUse \@LN@vadjustlist \FNLN@@vadjusts \FNLN@vadjusts
  \globaldefs\z@
}
```

For switching back to "main text mode," again some settings may need a local variant—for processing line references from footnotes to main text! This is the purpose of `\setgetpagewiselinenumbers`:

```
\newcommand* \setgetpagewiselinenumbers {%
    \let \FNLN@first@numbered \LN@Pfirst
    \let \FNLN@finish         \@gobbletwo
}
```

`\unsetfootnotelinenumbers` stores the “current” page with footnote lines and loads the “most recent” page with main text lines—and more . . . :

```
\newcommand* \unsetfootnotelinenumbers {%
    \gdef \FNLN@log {\string\@LN}%
    \FNLN@main@cache
```

Task lists:

```
    \GStoreUse \@LN@labellist \FNLN@labels \FNLN@@labels
    \GStoreUse \@LN@vadjustlist \FNLN@vadjusts \FNLN@@vadjusts
    \globaldefs\@ne \setgetpagewiselinenumbers \globaldefs\z@ %% v0.53
    \setpagewiselinenumbers
}
```

`\makeFootnoteLineNumber` actually only copies `\makePagewiseLineNumber`, different results are obtained be changing hooks. The command first calls logging—`\logtheLineNumber`, then generating the “public” line number—`\getLineNumber` (which in turn only is a copy of `\testNumberedPage` in ’lineno.sty’).

```
\@ifdefinable\makeFootnoteLineNumber
    {\let \makeFootnoteLineNumber \makePagewiseLineNumber}
```

2.6.7 Logging

`\logtheLineNumber` is redefined to log both main text and footnote line numbers.

```
\def \logtheLineNumber {%
    \protected@write\@auxout{}{%
        \FNLN@log{\the\c@linenumber}{\noexpand\the\c@LN@truepage}}}
```

`\FNLN@log` is the hook for the difference, its default expansion `\@LN` is made for *main text* line numbers:

```
\gdef \FNLN@log {\string\@LN}
```

2.6.8 “Public” Line Numbers

Fortunately, these commands don’t need to know much about name spaces. The interfaces to them are `\NumberedPageCache`—changing globally—and `\FNLN@first@numbered`. Our `\FNLN@cache` is initialized by analogy to its counterpart `\NumberedPageCache` (a minute name space change):

```
\def \FNLN@cache {\FLN@Pfirst}
```

`\testFirstNumberedPage{<integer>}` from ’lineno.sty’ is modified by replacing `\LN@Pfirst` only:

```
\renewcommand* \testFirstNumberedPage [1]{%
```

```
  \ifnum#1>\c@linenumber
     \def\nextLN##1{%
        \testNextNumberedPage\FNLN@first@numbered}%
  \else
     \let\nextLN\@gobble
     \def\pageLN{\gotNumberedPage{#1}}%
  \fi}
```

`\testNumberedPage` and `\testNextNumberedPage` from 'lineno' don't need any modification. `\testLastNumberedPage` is modified in 'edfnotes.sty'.

`\gotNumberedPage` just needs a closing hook `\FNLN@finish` to allow for footnote lines.

```
\renewcommand* \gotNumberedPage [4]{%
  \oddNumberedPagefalse
  \ifodd \if@twocolumn #3\else #2\fi\relax\oddNumberedPagetrue\fi
  \advance\c@linenumber\@ne
  \ifcolumnwiselinenumbers
     \subtractlinenumberoffset{#1}%
  \else
     \subtractlinenumberoffset{#4}%
  \fi
%  \show\FNLN@finish
  \FNLN@finish{#2}{#3}%
}
```

`\FNLN@finish{<page>}{<column>}` gobbles both arguments with *main* text lines, but will add the number of main text lines to *footnote* line numbers:

```
\global\let \FNLN@finish \@gobbletwo
```

Then it will act as `\FNLN@add`. We run the page info macro for the same page (column; if defined).

```
\newcommand* \FNLN@add [2]{%
  \expandafter \let\expandafter \@tempa\csname LN@P#1C#2\endcsname
  \ifx\@tempa\relax
  \else
    \advance\c@linenumber\@ne
    \ifcolumnwiselinenumbers
      \let\firstLN\subtractlinenumberoffset
```

... rather assuming `\realpagewiselinenumbers`.

```
      \let\pageLN\@gobblethree
    \else
      \let\firstLN\@gobble
      \def\pageLN##1##2##3{\subtractlinenumberoffset{##3}}%
    \fi
    \def\lastLN##1{\subtractlinenumberoffset{-##1}}%
    \let\nextLN\@gobble
```

. . . TODO all needed?

```
    \@tempa
  \fi
}
```

2.6.9 Referencing

Now that we are using two separate counters for main text lines and footnote lines (v0.5), correct references to footnote lines using `\linelabel` and `\ref` need further adjustments. 'lineno.sty"s `\thePagewiseLineNumber` and `\getpagewiselinenumber{<integer>}` are generalized and re-implemented by macros that then serve to implement referring to footnote line numbers.

`\theWiseLineNumber{<trans>}` leaves a `\protect`ed call to a one-parameter macro ¡trans¿ in the `.aux` file:

```
\newcommand* \theWiseLineNumber [1]{\protect #1{\the\c@linenumber}}
```

`\getwiselinenumber{<choice>}{<integer>}` executes ¡choice¿ before applying `\testNumberedPage` to ¡integer¿—within a local group:

```
\newcommand* \getwiselinenumber [2]{{%
```

Some wisdom is needed to take account of the current ‘`numbering state" from which ‘\ref` was called.

Referring to main text line:

- Unless called from numbered footnote, no extra care is needed.
- If called from numbered footnote, `\setgetpagewiselinenumbers` and temporary switching of `\NumberedPageCache` is needed.

Referring to footnote line:

- If called from numbered footnote, no extra care is needed.
- Otherwise, `\setgetfootnotelinenumbers` and temporary switching of `\NumberedPageCache` is needed.

```
  \ifx#1\relax          %% to main text
    \if@FNLN@sw@        %% from footnote
      \setgetpagewiselinenumbers
      \FNLN@main@cache
      \let \FNLN@restore@cache \FNLN@foot@cache
    \fi
  \else                 %% to footnote
    \if@FNLN@sw@ \else  %% from elsewhere
```

```
      #1%
      \FNLN@foot@cache
      \let \FNLN@restore@cache \FNLN@main@cache
    \fi
  \fi
  \c@linenumber #2\relax\testNumberedPage
  \thelinenumber
  \FNLN@restore@cache
}}
\let \FNLN@restore@cache \relax
```

`\getpagewiselinenumber` doesn't need any ¡choice¿—we assume that the label was written in the default `pagewise` mode (but it is difficult, though, `\relax` is essential!):

```
% \renewcommand* \getpagewiselinenumber {\getwiselinenumber\relax} %!!
```

2010/12/31, a compatibility problem with 'ednotes" `\newlabel` mechanism shows up. 'ednotes' `‘undefines" ‘\getpagewiselinenumber` and restores it only `\AtBeginDocument`. We must ensure that 'ednotes' will not override our new version of `\getpagewiselinenumber`. (TODO in my view another motivation to write "ready" numbers without `\getpagewiselinenumbers` directly.)

We might assume that 'ednotes' (if at all) is loaded directly and loads 'lineno.sty' (that is the usual and recommended way of using 'ednotes') and that this will happen before 'fnlineno.sty' is loaded. But now that we have spent some time understanding the situation, we can deal with the case as well that 'lineno.sty' is loaded first, then 'fnlineno.sty' is loaded, and then 'ednotes'. (I have assumed earlier that 'fnlineno.sty' is loaded after 'lineno.sty' . . .)

```
\AtBeginDocument{%
    \def \getpagewiselinenumber {\getwiselinenumber\relax}% sic!
    \let \@EN@getpagewiselno \getpagewiselinenumber}
```

For `\thePagewiseLineNumber`, ¡trans¿ is `\getpagewiselinenumber`:

```
\renewcommand* \thePagewiseLineNumber {%
    \theWiseLineNumber\getpagewiselinenumber}
```

`\getfootnotelinenumber{<integer>}` considers ¡integer¿ the absolute number of a *footnote* line. The ¡choice¿ therefore is `\setgetfootnotelinenumbers`:

```
\newcommand* \getfootnotelinenumber {%
    \getwiselinenumber\setgetfootnotelinenumbers}
```

Finally, `\theFootnoteLineNumber` is how `\linelabel` refers to a *footnote* line. `\theWiseLineNumber` is called with ¡trans¿ being `\getfootnotelinenumber`:

```
\newcommand* \theFootnoteLineNumber {%
    \theWiseLineNumber\getfootnotelinenumber}
```

2.7 Leaving the Package File

```
\endinput
```

3 Acknowledgements

On the ’texhax’ mailing list, Boris Veytsman recommended using Victor Eijkhout’s *TeX by Topic* to me, and Andrej Lapshin pointed me to David Salomon’s work on output routines (TUGboat 1990 and 1994, also available as a book, as Ulrich Dirr tells me). It helped me a lot to read about output routines in these works, beyond the TeXbook. The abbreviations ‘OTR’ and ‘MVL’ are Salomon’s.—And recall Christian’s work and support by the DFG named at the start of the package file.—And . . . the ideas of how to implement (*i*) attaching line numbers, (*ii*) `\linelabel`, and (*iii*) numbering lines “pagewise”—so flexibly, compatibly with many other LaTeX packages, still are Stephan’s . . .

4 VERSION HISTORY

```
v0.1    2010/12/08  very first, \linelabel works in footnote
        SENT TO Christian, problems with "long" footnotes

v0.2    2010/12/08  corr. "manifoot"
        2010/12/09  moving doc. from .tex to here,
                    different doc. sectioning;
                    \@footnotetext modified (user feature!);
                    \@doclearpage NOT modified!; \if@FNLN@placing@
        2010/12/10  ignore dummy footnote split;
                    \FNLNpar, \AutoPars, \ExplicitPars,
                    more on limitations
        2010/12/11  more trying, almost anew ...
        JUST STORED

v0.3    2010/12/12  new approach, removed much before proceeding
        2010/12/13  -- this was putting \box\footins onto MVL,
                    bad with those penalties
        JUST STORED

v0.4    2010/12/14  another new approach:
                    typeset footnote on MVL immediately --
                    described strategy
        2010/12/15  ... continued, choice of hooking into \output
                    (...swap...)
        2010/12/16  ... continued; rearranged sections ...
                    \FNLN@@fntext vs. ...ltx...
        2010/12/17  success with \pagegoal ...; \GStoreReg etc.;
```

```
                    ...@fntext shortened
        2010/12/18  another two limitations: \pagebreak in fn.,
                    guessed/tested; another note to <register>;
                    ack. Christian; directed -> organized!?
        SENT TO Christian/Stephan

v0.41   2010/12/19  support of \pagebreak with \if@FNLN@sw@ etc.;
                    TODO on lists of <register>s
        2010/12/20  debugging: \if...true; \setbox...ft;
                    \@finalstrut in vmode exported to finstrut.sty;
                    notes on how v0.41 still fails with \pagebreak
        2010/12/21  additional notes on *two* \pagebreak's

v0.5    2010/12/21  restructuring doc., check@latex@ -> check@,
                    own account of lineno's pagewise mode
        2010/12/22  ... continued ...
        2010/12/23  ... continued ...
        2010/12/24  ... continued ...
        2010/12/25  moved this to pwlineno, replaced ...
                    more on \FNLN@typeset, + \setfootnotelinenumbers
        2010/12/26  new summary of implementation,
                    rearranged code sections; logging settled
v0.51   2010/12/27  "build" settled, typesetting, logging reformated;
                    ack.s: "recall"; all settings global,
                    "public" works
        JUST STORED, MARGINAL NUMBERS OK,
        \linelabel in footnote broken
        [2010/12/28]
v0.52   2010/12/28  own label and vadjust lists for footnotes;
                    local settings for referencing,
                    tool and care for global changes (...Cache)
                    (TODO write ready in .aux? needs another run)
        \linelabel's ok, MARGINAL NOTES MAIN BROKEN
v0.53   2010/12/28  debugging; OK; minor doc. modifications;
                    less "limitations"; \\[\smallskipamount]
        TO CHRISTIAN 2010-12-29
v0.54   2010/12/31  typo options; \FNLN@text without arg,
                    \getpagewiselinenumber with ednotes
        2011/01/01  \FNLN@cache, \FNLN@@cache initialized;
                    doc. "Typesetting Stage" qualification
        2011/01/02  that qualification was wrong
        2011/01/03  samepage@hook
        TO CHRISTIAN SAME DAY
v0.55   2011/01/04  samepage@hook emptied here as well;
        2011/01/06  edited version history
        2011/01/07  note on \if@FNLN@sw@ with v0.5;
```

```
                    finally without support for samepage@hook!
                    note on \testLastNumberedPage
        PART OF EDFN RELEASE r0.5 (together with edfnotes v0.2)
v0.55a  2011/02/09  corr. owner; "Limitations" updated; \pagebreak
```