The microtype package

Subliminal refinements towards typographical perfection

R Schlicht
w.m.l@gmx.net

v3.2b
2025/07/09

https://github.com/schlcht/microtype

The microtype package provides a LaTeX interface to the micro-typographic extensions that were introduced by pdfTeX and some of which have since also propagated to LuaTeX and XeTeX: most prominently, character protrusion and font expansion, furthermore the adjustment of interword spacing and additional kerning, as well as hyphenatable letterspacing (tracking) and the possibility to disable all or selected ligatures. These features may be applied to customisable sets of fonts, and all micro-typographic aspects of the fonts can be configured in a straight-forward and flexible way. Settings for various fonts are provided.

Note that character protrusion requires pdfTeX (version 0.14f or later), LuaTeX, or XeTeX (at least version 0.9997). Font expansion works with pdfTeX (version 1.20 for automatic expansion) or LuaTeX. The package will by default enable protrusion and expansion if they can safely be assumed to work. Disabling ligatures requires pdfTeX ($\geq$ 1.30) or LuaTeX, while the adjustment of interword spacing and of kerning only works with pdfTeX ($\geq$ 1.40). Letterspacing is available with pdfTeX ($\geq$ 1.40), LuaTeX ($\geq$ 0.62) or XeTeX.

The alternative package letterspace, which also works with plain TeX, provides the user commands for letterspacing only, omitting support for all other extensions (see section 7).

Contents

List of Tables

Implementation (external document)

1 Micro-typography with TEX

Micro-typography is the art of enhancing the appearance and readability of a document while exhibiting a minimum degree of visual obtrusion. It is concerned with what happens between or at the margins of characters, words or lines. Whereas the macro-typographical aspects of a document (i.e., its layout) are clearly visible even to the untrained eye, micro-typographical refinements should ideally not even be recognisable. That is, you may think that a document looks beautiful, but you might not be able to tell exactly why: good micro-typographic practice tries to reduce all potential irritations that might disturb a reader.

Some essential micro-typographical aspects are already taken care of by TEX out of the box – and in an outstanding manner – namely, hyphenation and justification, as well as kerning and ligatures. Other aspects are in the user's scope of responsibilities, e.g., to specify the right amounts of spacing around punctuation characters, numbers, or quotation marks. On top of this, a number of long-standing micro-typographic techniques have been introduced to the TEX world relatively recently with pdfTEX, and have since also propagated to LuaTEX and XETEX. These features make them the tool of choice not only for the creation of electronic documents but also of works of outstanding time-honoured typography: most prominently, *character protrusion* (also known as margin kerning) and *font expansion*. Quoting Hàn Thế Thành, the author of pdfTEX, who writes in his thesis:

After you have read the text on the right, you can view the effect of the features it describes by clicking on the links:

Protrusion	*off*
Expansion	*off*

Both features are enabled throughout this document.

> 'Margin kerning is the adjustments of the characters at the margins of a typeset text. A simplified employment of margin kerning is hanging punctuation. Margin kerning is needed for optical alignment of the margins of a typeset text, because mechanical justification of the margins makes them look rather ragged. Some characters can make a line appear shorter to the human eye than others. Shifting such characters by an appropriate amount into the margins would greatly improve the appearance of a typeset text.
>
> Composing with font expansion is the method to use a wider or narrower variant of a font to make interword spacing more even. A font in a loose line can be substituted by a wider variant so the interword spaces are stretched by a smaller amount. Similarly, a font in a tight line can be replaced by a narrower variant to reduce the amount that the interword spaces are shrunk by. There is certainly a potential danger of font distortion when using such manipulations, thus they must be used with extreme care. The potentiality to adjust a line width by font expansion can be taken into consideration while a paragraph is being broken into lines, in order to choose better breakpoints.' [Thành 2000, p. 323]

Another micro-typographic technique, which has always been extremely difficult to achieve in TEX, is robust and hyphenatable *letterspacing (tracking)*.[1] Whereas letterspacing can easily be, and often is, abused when applying it to lowercase letters, readability may be increased by slightly letterspacing (small) capitals or by decreasing the tracking of very large uppercase type.

Setting *additional kerning* for individual characters is especially (but not only) useful for languages whose typographical tradition requires certain characters to be separated by a space. For example, it is customary in French typography to add a small space before question mark, exclamation mark and semi-colon, and a bigger space before the colon and the guillemets. Until now, this could only be achieved

1 The `soul` package undertakes great efforts, but may still fail in certain circumstances; even to systematically adjust the tracking of a font throughout the document remains impossible.

by making these characters active (as is done, for example, by the `babel` package), which may not always be a robust solution. In contrast to the standard kerning built into the fonts (which will of course apply as usual), this additional kerning relates to single characters, not to character pairs.

Adjustment of interword spacing is based upon the idea that in order to achieve a uniform greyness of the text, the space between words should also depend on the surrounding characters. For example, if a word ends with an 'r', the following space should be a tiny bit smaller than that following, say, an 'm'. You can think of this concept as an extension to TEX's 'space factors'. This feature may enhance the appearance of paragraphs even more. Emphasis in the last sentence is on the word 'may': this extension is still highly experimental – in particular, only ending characters will currently influence the interword space. Also, the settings shipped with `microtype` are but a first approximation, and I would highly welcome corrections and improvements. I suggest reading the reasoning behind the settings in the Implementation part, section 2.9.

The possibility, finally, to *disable all or selected ligatures* is particularly useful for typewriter fonts.

The `microtype` package provides an interface to all these micro-typographic extensions. All micro-typographic aspects may be customised to your taste and needs in a straight-forward and systematic manner. The next chapters present a survey of all options and customisation possibilities. Should the micro-typographic extension discussed in a section work only with certain TEX engines, this requirement is marked inside a grey text box on the right.

2 Getting started

There is nothing surprising in loading this package:

```
\usepackage{microtype}
```

This will be sufficient in most cases, and if you are not interested in fine-tuning the micro-typographic appearance of your document (however unlikely this would seem, since using this package is proof of your interest in typographic issues), you may actually skip the rest of this document. If this, on the other hand, does not satisfy you – be it for theoretical or practical reasons – this manual will guide you on the path to the desired results along the following milestones:

- Enable the desired micro-typographic features, either via the respective package option or with the `\microtypesetup` command (section 3).
- Select the fonts to which the features should be applied by declaring and activating 'sets of fonts'. A number of sets are predefined, which may be activated directly in the package options (section 4).
- Fine-tune the micro-typographic settings of the fonts or sets of fonts (section 5).
- If you're of the kind who always wants to march on, you will certainly be interested in the possibility of context-sensitive setup (section 6).
- You are even countenanced to leave the path of typographic virtue and steal some sheep (section 7) or trespass in other ways (section 8).

- For the pedantic or the perfectionist, sections 9 and 10 are warmly recommended.
- Should you encounter any obstacles, follow the hints and caveats (section 11).

3 Options

Like many other LaTeX packages, the `microtype` package accepts options in the well-known `key=value` syntax. In the following, you will find a description of all `keys` and their possible `values` ('`true`' may be omitted; multiple values, where allowed, must be enclosed in braces; the default value is shown on the right – if preceded by an asterisk, this default only applies when running an up-to-date pdfTeX in PDF mode).

3.1 Enabling the micro-typographic features

`protrusion` `true`, `false`, `compatibility`, `nocompatibility`, ⟨*font set name*⟩ * `true`

`expansion` These are the main options to control the level of micro-typographic refinement which the fonts in your document should gain. By default, the package is moderately greedy: character protrusion will always be enabled, font expansion will only be disabled when the fonts cannot be expanded automatically, that is, with pdfTeX versions older than 1.20 or in DVI output mode (see section 3.5), or with XeTeX. In other words, `microtype` will try to apply as much micro-typography as can safely be expected to work under the respective conditions (hence, it is usually not necessary to load the package with different options, e.g., for PDF resp. DVI mode).

`activate` Protrusion and expansion may be enabled or disabled independently from each other by setting the respective key to `true` resp. `false`. The `activate` option is a shortcut for setting both options at the same time. Therefore, the following lines all have the same effect (when creating PDF files with a recent version of pdfTeX):

```
\usepackage[protrusion=true,expansion]{microtype}
```

```
\usepackage[activate={true,nocompatibility}]{microtype}
```

```
\usepackage{microtype}
```

With activated font expansion and/or character protrusion, line breaks (and consequently, page breaks) may turn out differently. If this is not desired – because you are re-typesetting a book whose pagination must not change – you may pass the value `compatibility` to the `protrusion` and/or `expansion` options. Typographically, however, the results will be suboptimal, hence the default value is `nocompatibility`.

Finally, you may also specify the name of a font set to which character protrusion and/or font expansion should be restricted. See section 4 for a detailed discussion. Specifying a font set for a feature implicitly activates this feature.

`tracking` `true`, `false`, ⟨*font set name*⟩ `false`

This option will systematically change the tracking of the fonts specified in the active font set (by default, all small capitals). With pdfTeX, it is only available in PDF mode.

Table 1: Availability of micro-typographic features

TEX engine			Micro-typographic features					
Engine	Version	Output	Protrusion	Expansion		Tracking	Kerning	Spacing
				manual	automatic			
pdfTEX	< 0.14f	DVI/PDF	∅	∅	∅	∅	∅	∅
	≥ 0.14f	DVI/PDF	★	⊠	∅	∅	∅	∅
	≥ 1.20	DVI	★	⊠	∅	∅	∅	∅
		PDF	★	⊠	★	∅	∅	∅
	≥ 1.40	DVI	★	⊠	∅	∅	⊠	⊠
		PDF	★	⊠	★	⊠	⊠	⊠
LuaTEX	≥ 0.30	DVI	★	⊠	∅	∅	∅	∅
		PDF	★	⊠	★	∅	∅	∅
	≥ 0.62	DVI	★	∅	(⊠)[a]	⊠	∅	∅
		PDF	★	∅	★	⊠	∅	∅
XETEX	≥ 0.9997	PDF	★	∅	∅	⊠	∅	∅

★ = enabled ⊠ = not enabled ∅ = not available *a* by means of variable tracking

`kerning` `true`, `false`, ⟨*font set name*⟩ `false`

`spacing` These features do not unconditionally improve the quality of the typeset text: the `spacing` feature is still considered experimental, while the `kerning` feature only makes sense in special cases. Therefore, neither feature is enabled by default. They are not available with XETEX or LuaTEX.

Table 1 presents an overview of which micro-typographic features are available and enabled by default for the relevant TEX versions and output modes.

Whether ligatures should be disabled cannot be controlled via a package option but by using the `\DisableLigatures` command, which is explained in section 8.

3.2 Character protrusion

pdfTEX 0.14f | LuaTEX 0.30 | XETEX 0.9997

`factor` ⟨*integer*⟩ 1000

Using this option, you can globally increase or decrease the amount by which the characters will be protruded. While a value of 1000 means that the full protrusion as specified in the configuration (see section 5.1) will be used, a value of 500 would result in halving all protrusion factors of the configuration. This might be useful if you are generally satisfied with the settings but prefer the margin kerning to be less or more visible (e.g., if you are so proud of being able to use this feature that you want everybody to see it, or – to mention a motivation more in compliance with typographical correctness – if you are using a large font that calls for more modest protrusion).

`patch` `all`, `none`, ⟨*list of patches*⟩ `all`

`nopatch` These options provide control over various patches meant to fix protrusion at margins other than the text block margins. They are described in detail in section 9.

`unit` `character`, ⟨*dimension*⟩ `character`

This option is described in section 5.1, apropos of the command `\SetProtrusion`. Use with care.

3.3 Font expansion

pdfTEX 0.14f | LuaTEX 0.30

`auto` `true`, `false` * `true`

Beginning with pdfTEX version 1.20 (inherited by LuaTEX), the expanded instances of the fonts may be calculated automatically and at run-time instead of the user having to prepare them in advance. This option is true by default provided that you are using a TEX engine with this capability and the output mode is PDF.[2] If `auto` is set to false, the font instances for all expansion steps must exist (with files called ⟨*font name*⟩`±`⟨*expansion value*⟩, e.g., `cmr12+10`, as described in the pdfTEX manual). With recent versions of LuaTEX, expansion is always automatic.

When generating DVI files, font expansion has to be enabled explicitly. With pdfTEX, *automatic* font expansion will not work because the postprocessing drivers (`dvips`, `dvipdfm`, etc.) resp. the DVI viewer are not able to generate the fonts on the fly. With LuaTEX, on the other hand, expansion in DVI mode is realised by modifying the inter-letter spacing (tracking) instead of the glyphs themselves, which may or may not be desired.

`stretch` ⟨*integer*⟩ 20

`shrink` You may specify the stretchability and shrinkability of a font, i.e., the maximum amount that a font may be stretched or shrunk. The numbers will be divided by 1000, so that a stretch limit of 10 means that the font may be expanded by up to 1%. The default stretch limit is 20. The shrink limit will by default be the same as the stretch limit.

`step` ⟨*integer*⟩ * 1

Fonts are not expanded by arbitrary amounts but only by certain discrete steps within the expansion limits. With recent versions of pdfTEX (1.40 or newer) or LuaTEX, this option is by default set to 1, in order to allow trying the maximum number of font instances, and hence to guarantee the best possible output.[3] Older pdfTEX versions, however, had to include every font instance in the PDF file, which may increase the file size quite dramatically. Therefore, in case you are using a pre-1.40 pdfTEX version, `step` is by default set to one fifth of the smaller value of `stretch` and `shrink`.

`selected` `true`, `false` `false`

When applying font expansion, it is possible to restrict the expansion of some characters that are more sensitive to deformation than others (e.g., the 'O', in contrast to the 'I'). This is called *selected expansion*, and its use allows increasing the stretch and shrink limits (to, say, 30 instead of 20); however, the gain is limited since at the same time the average stretch variance will be decreased. Therefore,

2 With pdfTEX, automatic font expansion does not work with bitmap fonts. Therefore, if you are using the Computer Modern Roman fonts in `T1` encoding, you should either install the `cm-super` package or use the Latin Modern fonts (package `lmodern`).

3 The downside with this default is that pdfTEX may run out of memory with huge documents; in this case, read about the error messages in the 'Hints and caveats' section (11), or try with a larger `step`.

this option is by default set to `false`, so that all characters will be expanded by the same amount. See section 5.2 for a more detailed discussion.

3.4 Tracking

pdfTeX 1.40 | LuaTeX 0.62 | XeTeX 0.9997

`letterspace` ⟨*integer*⟩ 50

This option changes the default amount for tracking (see section 5.3) resp. letterspacing (see section 7). The amount is specified in thousandths of 1 em; admissible values are in the range of −1000 to +1000.

3.5 Miscellaneous options

`disable` `true`, `false`, `ifdraft` `false`

If the `disable` option is passed to the package, all micro-typographic extensions will be disabled, which may lead to different line, and hence even page, breaks. This option replaces the `draft` option from earlier versions, which could be inherited from the class options; to restore the previous behaviour, you may pass the value `ifdraft`: in this case, the `disable` option will be set to `true` if and only if the document class has been loaded with the `draft` option.

`verbose` `true`, `false`, `errors`, `silent` `false`

Information on the settings used for each font will be written into the log file if you enable the `verbose` option. This is useful for debugging. When `microtype` encounters a problem that is not fatal (e.g., an unknown character in the settings, or non-existent settings), it will by default only issue a warning and try to continue. Loading the package with `verbose=errors` will turn all warnings into errors, so that you can be sure that no problem will go unnoticed. If, on the other hand, you have investigated all warnings and decide to ignore them, you may silence `microtype` with `verbose=silent`.

`babel` `true`, `false` `false`

Loading the package with the `babel` option will adjust the typesetting according to the respective selected language. This also works with the `polyglossia` package. Read section 6 for further information.

`config` ⟨*file name*⟩ `microtype`

Various settings for this package will be loaded from a main configuration file, by default `microtype.cfg` (see section 5.7). You can have a different configuration file loaded instead by specifying its name *without the extension*, e.g., `config=mycrotype`.

`DVIoutput` `true`, `false` * `false`

pdfTeX and LuaTeX are not only able to generate PDF output but can also spit out DVI files. In fact, all recent TeX systems are using pdfTeX as the default engine also for DVI output, and LuaTeX too can be called in DVI mode. However, since changing the output mode inside the document may have undesired effects, this option should be considered deprecated; instead, it is recommended to just call the respective program (`latex` resp. `dvilualatex`). For XeTeX, this option is not applicable.

3.6 Changing options later

`\microtypesetup` {⟨*key = value list*⟩}

Inside the preamble, this command accepts all package options described above (except for `config`). In the document body, this command may be used to change the general settings of the micro-typographic extensions. It then accepts all options from section 3.1: `expansion`, `protrusion` and `activate`, which in turn may receive the values `true`, `false`, `compatibility` or `nocompatibility`, and `tracking`, `kerning` and `spacing` with the admissible values `true` or `false`. Passing the name of a font set is not allowed. Additionally, it accepts the options `patch` and `nopatch` (see section 9). Using this command, you could for instance temporarily disable font expansion by saying:

```
\microtypesetup{expansion=false}
```

4 Selecting fonts for micro-typography

By default, character protrusion will be applied to all text fonts used in the document, and a basic set of fonts will be subject to font expansion. You may want to customise which fonts should get the benefit of micro-typographic treatment. This can be achieved by declaring and activating 'font sets'; these font sets are specified via font attributes that have to match.

`\DeclareMicrotypeSet` [⟨*features*⟩] {⟨*set name*⟩} {⟨*set of fonts*⟩}
`\DeclareMicrotypeSet*`

This command declares a new set of fonts to which the micro-typographic extensions should be applied. The optional argument may contain a comma-separated list of features to which this set should be restricted. The starred version of the command declares *and* activates the font set at the same time.

The set of fonts is specified by assigning values to the NFSS font attributes: encoding, family, series, shape and size (cf. LaTeX 2ε font selection). Let's start with an example. In the main configuration file `microtype.cfg`, a font set called '`basictext`' is defined as follows:

```
\DeclareMicrotypeSet{basictext}
   { encoding = {OT1,T1,T2A,LY1,OT4,QX,T5,EU1,EU2,TU},
     family   = {rm*,sf*},
     series   = {md*},
     size     = {normalsize,footnotesize,small,large}
   }
```

If you now call

```
\UseMicrotypeSet[protrusion]{basictext}
```

in the document's preamble, only fonts in the text encodings, roman or sans serif families, normal (or 'medium') series, and in sizes called by `\normalsize`, `\footnotesize`, `\small` or `\large`, will be protruded. Math fonts, on the other hand, will not, since they are in another encoding. Neither will fonts in bold face, or huge fonts. Etc.

If an attribute list is empty or missing – like the 'shape' attribute in the above example – it does not constitute a restriction. In other words, this is equivalent

to specifying *all* possible values for that attribute. Therefore, the predefined set ‘alltext’, which is declared as:

```
\DeclareMicrotypeSet{alltext}
   { encoding = {OT1,T1,T2A,LY1,OT4,QX,T5,TS1,EU1,EU2,TU} }
```

is far less restrictive. The only condition here is that the encoding must match.

If a value is followed by an asterisk (like ‘rm*’ and ‘sf*’ in the first example), it does not designate an NFSS code, but will be translated into the document’s \⟨*value*⟩default, e.g., \rmdefault.[4] A single asterisk means \⟨*attribute*⟩default, e.g., \encodingdefault, respectively \normalsize for the size axis. Sizes may either be specified as a dimension (‘10’ or ‘10pt’), or as a size selection command *without* the backslash. You may also specify ranges (e.g., ‘small-Large’); while the lower boundary is included in the range, the upper boundary is not. Thus, ‘12-16’ would match 12 pt, 13.5 pt and 15.999 pt, for example, but not 16 pt. You are allowed to omit the lower or upper bound (‘-10’, ‘large-’).

Additionally to this declaration scheme, you can add single fonts to a set using the ‘font’ key, which expects the concatenation of all font attributes, separated by forward slashes, i.e., ‘font = ⟨*encoding*⟩/⟨*family*⟩/⟨*series*⟩/⟨*shape*⟩/⟨*size*⟩’. This allows you to add fonts to the set that are otherwise disjunct from it. For instance, if you wanted to have the roman family in all sizes protruded, but only the normal sized, possibly italic, typewriter font (in contrast to, say, the small one), this is how you could declare the set:

```
\DeclareMicrotypeSet[protrusion]
   { myset }
   { encoding = T1,
     family   = rm*,
     font     = {T1/tt*/m/n/*,
                 T1/tt*/m/it/*} }
```

As you can tell from the example, the asterisk notation is also permitted for the font key. A single asterisk is equivalent to ‘*/*/*/*/*’, i.e., the normal font. Size selection commands are possible, too, however, ranges are not allowed.

Table 2 lists the eleven predefined font sets. They may also be activated by passing their name to the feature options protrusion, expansion, tracking, kerning and spacing when loading the package, for example:

```
\usepackage[protrusion=allmath,tracking=smallcaps]{microtype}
```

\UseMicrotypeSet [⟨*features*⟩] {⟨*set name*⟩}

This command activates a font set previously declared by \DeclareMicrotypeSet. Using the optional argument, you can limit the application of the set to one or more features. This command only has an effect if the feature was activated in the package options.

\DeclareMicrotypeSetDefault [⟨*features*⟩] {⟨*set name*⟩}

If a feature is enabled but no font set has been chosen explicitly, the sets declared by this command will be activated. By default, the ‘alltext’ font set will be activated

4 These translations will take place \AtBeginDocument, which means that changes to the defaults inside the preamble will also be taken into account. Only in cases where you change font defaults \AtBeginDocument yourself, you need to load microtype after these changes.

Table 2:

Predefined font sets

Set name	Font attributes				
	Encoding	Family	Series	Shape	Size
`all`	∅	∅	∅	∅	∅
`alltext` (`allmath`)	Text encodings, TS1 (OML, OMS, U)	∅	∅	∅	∅
`alltext-nott` (`allmath-nott`)	Text encodings, TS1 (OML, OMS, U)	`\rm*`, `\sf*`	∅	∅	∅
`basictext` (`basicmath`)	Text encodings (OML, OMS)	`\rm*`, `\sf*`	`\md*`	∅	`\normalsize`, `\footnotesize`, `\small`, `\large`
`smallcaps`	Text encodings	∅	∅	`\sc*`,si,scit	∅
`footnotesize`	Text encodings, TS1	∅	∅	∅	-`\small`
`scriptsize`	Text encodings, TS1	∅	∅	∅	-`\footnotesize`
`normalfont`	`\encoding*`	`\family*`	`\series*`	`\shape*`	`\normalsize`

'Text encodings' = OT1, T1, T2A, LY1, OT4, QX, T5, EU1, EU2, TU '`\...*`' = '`\...default`'

for character protrusion and additional kerning, the '`alltext-nott`' set for font expansion and interword spacing, and the '`smallcaps`' set for tracking.

These commands may only be used in the preamble or in the main configuration file. Their scope is global to the document. Only one set per feature may be activated.

5 Micro fine tuning

Every character asks for a particular protrusion, kerning or spacing amount. It may also be desirable to restrict the maximum expansion of certain characters. Furthermore, since every font looks different, settings have to be specific to a font or set of fonts. This package offers flexible and straight-forward methods of customising these finer aspects of micro-typography.

All fine-tuning commands follow basically the same syntax: they all take three arguments; the first one is optional and may contain additional options; in the second argument, you specify the set of fonts to which the settings should apply; the third argument contains the actual settings. Here, as in all configuration commands, all spaces are ignored.

The set of fonts to which the settings should apply is declared using the same syntax of ⟨*font axis*⟩ = ⟨*value list*⟩ pairs as for the command `\DeclareMicrotypeSet` (see section 4), with the only difference that values including asterisks (which, as you may recall, stand for the respective default) will be translated immediately instead of at the end of the preamble. To find the matching settings for a given font the package will try all combinations of font encoding, family, series, shape and size, with decreasing significance in this order. For instance, if settings exist for both the current family (say, `T1/cmr///`) and for italic fonts in the normal weight (`T1//m/it/`), the settings for the `cmr` family would apply. The encoding must always match.

The characters may be specified either as a single letter (`A`), as a text symbol command (`\textquoteleft`), or as a slot number (resp. Unicode number for LuaTEX or XETEX): three or more digits for decimal notation, prefixed with `"` for hexadecimal, with `'` for octal numerals (e.g., the 'fl' ligature in `T1` encoding: `029`, `"1D`, `'35`). 8-bit (and even UTF-8) characters may be entered directly or in LATEX's traditional 7-bit notation: both `\"A` and Ä are valid, provided the character is actually declared in both the input and the font encoding. With LuaTEX or XETEX, you may additionally specify a (font-specific) glyph name, prefixed with '/' (e.g., the 'fl' ligature as `/f_l`). Note that you also have the possibility to declare lists of characters that should inherit settings (see section 5.6).

5.1 Character protrusion

pdfTEX 0.14f | LuaTEX 0.30 | XETEX 0.9997

`\SetProtrusion` [⟨*options*⟩] {⟨*set of fonts*⟩} {⟨*protrusion settings*⟩}

Using this command, you can set the protrusion factors for each character of a font or a set of fonts. A very incomplete example would be the following:

```
\SetProtrusion
   { encoding = T1,
     family   = cmr }
   { A              = {50,50},
     \textquoteleft = {700, } }
```

which would result in the character 'A' being protruded by 5% of its width on both sides, and the left quote character by 70% of its width into the left margin. This would apply to all font shapes, series and sizes of the `T1` encoded Computer Modern Roman family.

The protrusion settings consist of ⟨*character*⟩ = ⟨*protrusion factors*⟩ pairs. The protrusion factors designate the amount that a character should be protruded into the left margin (first value) respectively into the right margin (second value). By default, the values are relative to the character widths, so that a value of 1000 means that the character should be shifted fully into the margin, while, for example, with a value of 50 it would be protruded by 5% of its width. Negative values are admitted, as well as numbers larger than 1000 (but effectively not more than 1 em of the font). You may omit either number if the character should not be protruded on that side, but must not drop the separating comma.

Options:

`name` You may assign a name to the protrusion settings, so that you are able to load it by another list.

`load` You can load another list (provided, you assigned a name to it) before the current list will be loaded, so that the fonts will inherit the values from the loaded list.

In this way, the configuration may be simplified considerably. You can for instance create a default list for a font; settings for other shapes or series can then load these settings, and extend or overwrite them (since the value that comes last will take precedence). Font settings will be loaded recursively. The following options will affect all loaded lists, in other words, any options from the loaded lists will be ignored:

`factor` This option can be used to influence all protrusion factors of the list, overriding any global `factor` setting (see section 3.2). For instance, if you want fonts in larger sizes to be protruded less, you could load the normal lists, just with a different factor applied to them:

```
\SetProtrusion
   [ factor   = 700,
     load     = cmr-T1 ]
   { encoding = T1,
     family   = cmr,
     size     = large- }
   { }
```

`unit` By default, the protrusion factors are relative to the respective character's width. The `unit` option may be used to override this and make `microtype` regard all values in the list as thousandths of the specified width. Issuing, for instance, '`unit=1em`' would have the effect that a value of, say, 50 now results in the character being protruded by 5% of an em of the font (thus simulating the internal measuring of pdfTEX's `\lpcode` and `\rpcode` primitives). The default behaviour can be restored with `unit=character`.[5]

`preset` Presets the protrusion codes of all characters to the specified values (`=`{⟨*left*⟩`,`⟨*right*⟩}), possibly scaled by a `factor`. A `unit` setting will only be taken into account if it is not `=character`.

`inputenc` Selects an input encoding that should apply to this list, regardless of what the document's input encoding is. You may specify any encoding that can be loaded via the `inputenc` package, e.g., `ansinew`, `koi8-r`, `utf8`.

`context` The scope of the list may be limited to a certain context. For further details, see section 6.

5.2 Font expansion

pdfTEX 0.14f | LuaTEX 0.30

`\SetExpansion` [⟨*options*⟩] {⟨*set of fonts*⟩} {⟨*expansion settings*⟩}

By default, all characters of a font are allowed to be stretched or shrunk by the same amount. However, it is also possible to limit the expansion of certain characters if they are more sensitive to deformation. This is the purpose of the `\SetExpansion` command. Note that it will only have an effect if the package has been loaded with the `selected` option (cf. section 3.3). Otherwise, the expansion settings will be ignored – unlike the options in the optional first argument, which will still be evaluated. If the `selected` option has been set to true, and settings for a font don't exist, font expansion will not be applied to this font at all. Should the extraordinary situation arise that you want to employ selected expansion in general but for a particular font (set) all characters should be expanded or shrunk by the same amount, you would have to declare an empty list for these fonts.

The expansion settings consist of ⟨*character*⟩`=`⟨*expansion factor*⟩ pairs. You may specify one number for each character, which determines the amount that a character may be expanded. The numbers denominate thousandths of the full expansion.

5 The `unit` option can even be passed globally to the package (cf. section 3.2). However, all provided settings are created under the assumption that the values are relative to the character width. Therefore, you should only change it if you are certain that the default settings will not be used in your document.

For example, if you set the expansion factor for the character 'O' to 500, it will only be expanded or shrunk by one half of the amount that the rest of the characters will be expanded or shrunk. While the default value for character protrusion is 0 – that is, if you didn't specify any characters, none would be protruded – the default value for expansion is 1000, which means that all characters would be expanded by the same amount.

Options:

`name`, `load`, `preset`, `inputenc`, `context` Analogous to `\SetProtrusion`, the optional argument may be used to assign a name to the list, to load another list, to preset all expansion factors, to set the input encoding, or to determine the context of the list (expansion contexts are only possible with pdfTEX version 1.40.4 or newer).

`auto`, `stretch`, `shrink`, `step` These keys can be used to override the global settings from the package options (see section 3.3). If you don't specify either one of `stretch`, `shrink` and `step`, their respective global value will be used (that is, no calculation will take place).

As a practical example, suppose you have a paragraph with a lone widow that could be avoided by shrinking the font a bit more. In conjunction with the `context` option (see section 6 for further details), you could thus allow for more expansion in this particular paragraph:

```
\SetExpansion
   [ context = sloppy,
     stretch = 30,
     shrink  = 60,
     step    =  5  ]
   { encoding = {OT1,T1,TS1} }
   { }
% ... END PREAMBLE
{\microtypecontext{expansion=sloppy}%
 This paragraph contains a `fussy' widow.}
```

This method of employing contexts to temporarily apply different expansion parameters only works with pdfTEX version 1.40.4 or later,[6] or with LuaTEX. Also note that both pdfTEX and LuaTEX prohibit the use of fonts with different expansion limits or steps (even of different fonts) within one paragraph, hence the `sloppy` context would have to be applied to complete paragraphs.

`factor` This option provides a different method to alter expansion settings for certain fonts, working around the restriction just mentioned. The `factor` option influences the expansion factors of all characters (in contrast to the overall stretchability) of the font. For instance, if you want the italic shape to be expanded less, you could declare:

```
\SetExpansion
   [ factor   = 500 ]
   { encoding = *,
     shape    = it }
   { }
```

6 For older versions, a dirty trick is laid out in the Implementation part, section 1.2, page 40.

The `factor` option can only be used to *decrease* the stretchability of the characters, that is, it may only receive values smaller than 1000. Also, it can only be used for single fonts or font sets; setting it globally in the package options wouldn't make much sense – to this end, you use the package's `stretch` and `shrink` options.

5.3 Tracking

pdfTeX 1.40 | LuaTeX 0.62 | XeTeX 0.9997

`\SetTracking` [⟨*options*⟩] {⟨*set of fonts*⟩} {⟨*tracking amount*⟩}

An important typographic technique – which was missing in TeX for a long time – is the adjustment of tracking, i.e., the uniform addition or subtraction of letter space to/from all the characters in a font. For example, it is good typographic practice to slightly space out text set in all capitals or small capitals (as in this document). Legibility may also be improved by minimally increasing the tracking of smaller and decreasing that of larger type.[7] The `\SetTracking` command allows specifying the tracking amount for different fonts or font sets. It will also be evaluated by the `\textls` command, which may be used for letterspacing shorter pieces of text (see section 7).

The tracking amount is specified in thousandths of 1 em (or the given unit); negative values are allowed, too.

Options:

`name`, `unit`, `context` These options serve the same functions as in the previous configuration commands. The unit may be any dimension, default is 1 em.

`spacing` When the inter-*letter* spacing is altered, the inter-*word* spacing probably also needs to be adjusted. This option expects three numbers for interword space, stretch and shrink respectively, which are given in thousandths of 1 em (or of the current `unit`). If a value is followed by an asterisk, it denotes thousandths of the respective font dimension which will be added to it. For instance, with

```
\SetTracking[ spacing = {25*,166, } ]{ encoding = *, shape = sc }{ 25 }
```

the interword space will be increased by 2.5%, the stretch amount will be set to 0.166 em, while the shrink amount will be left untouched. If you don't specify the `spacing` option, the interword space will be scaled by the current letterspace amount (as in the above example), while stretch and shrink will not be changed.

`outer spacing` If an interword space immediately precedes or follows letterspaced text, it will by default be equal to that within the text. With this option, which accepts the same values as `spacing`, it may be adjusted independently.

`outer kerning` If, on the other hand, no interword space precedes or follows, you may still want to slightly set off the first and last letter from adjoining letters. This option expects the kerning amounts for left and right hand side, separated by a comma, in thousandths of 1 em (or the current `unit`). If a value is followed by an asterisk, it denotes thousandths of the current letterspacing amount. A single asterisk means '`500*`'; this is also the default, i.e., the sum of the outer kerns is by default equal to the current letterspace amount. To remove kerning on both sides, you would write '`outer kerning={0,0}`'.

7 With full-featured fonts like Computer Modern, this is usually not necessary, though, since they come in optical sizes, and the tracking of the small-capitals font is already adjusted.

`no ligatures` In the absence of this option, ligatures in letterspaced fonts would be constructed as usual, which may be advisable when changing the tracking by only a small amount. For larger letterspacing amounts, on the other hand, the normal letter space within ligatures would have displeasing effects. This key expects a comma-separated list of characters for which ligatures should be disabled; only the character that begins a ligature must be specified. If the key is given without a value (or with the value 'all'), *all* ligatures of the font will be disabled. (With pdfTEX, this is not recommended, however, since it entails that kerning will be switched off, too. With LuaTEX, there is no such limitation.) Given the value 'none', none of the ligatures will be disabled, or, put positively, all of the font's ligatures will stay intact. The default settings (in `microtype.cfg`) disable ligatures for the character 'f' only, i.e., 'ff', 'fi', ffi', etc. In exceptional situations, you can manually break up a ligature by inserting '{\kern0pt}' resp. `babel`'s `"|` shortcut, or protect it by enclosing it in `\lslig` (see section 7). XETEX does not allow disabling ligatures selectively, or generally, for that matter.

`features` What it does allow, though, is enabling or disabling OpenType font features pertaining to ligatures. With XETEX or LuaTEX, the `features` key accepts any number of `fontspec` options for the 'Ligatures' font feature, e.g., `NoCommon`, `Rare`, `RequiredOff` or `ResetAll` (see the `fontspec` documentation for details). With LuaTEX, the `features` and the `no ligatures` keys may even be combined, where the latter will take precedence.

Since a picture is worth a thousand words, probably even more if, in our case, it depicts a couple of letterspaced words, let's bring one to sum up these somewhat confusing options. Suppose you had the following settings (which are in no way recommended; they only serve illustrative purposes):

```
\SetTracking
  [ no ligatures  = {f},
    spacing       = {600*,-100*, },
    outer spacing = {450,250,150},
    outer kerning = {*,*} ]
  { encoding = * }
  { 160 }
```

and then write:

```
Stop \textls{stealing sheep}!
```

this would be the (typographically dubious) outcome:

Click on the image to show the kerns and spacings involved. Click on emphasised words in the text below to reveal the relation of image and code.

While the word 'Stop' is not letterspaced, the space between the letters in the other two words is expanded by the *tracking amount* of 160/1000 em = 0.16 em. The *inner space* within the letterspaced text is increased by 60%, while its *stretch* amount is decreased by 10% and the *shrink* amount is left untouched. The *outer space* (of 0.45 em) immediately before the piece of text may *stretch* by 0.25 em and *shrink* by 0.15 em. Note that there is no outer space after the text, since the exclamation mark immediately follows; instead, the default *outer kern* of half the letterspace amount (0.08 em) is added. Furthermore, one *ligature* wasn't broken up, because we neglected to specify the 's' in the `no ligatures` key.

As another, more realistic example, suppose you want to space out all small capitals by 50/1000 em, fonts smaller than `\small` by 0.02 em, and to decrease the tracking of large type by 0.02 em. This could be achieved with the following settings:

```
\usepackage[tracking=true]{microtype}
\DeclareMicrotypeSet*[tracking]{my}
   { encoding = *,
     size     = {-small,Large-},
     font     = */*/*/sc/* }
\SetTracking[ no ligatures = f ]{ encoding = *, shape = sc}{ 50 }
\SetTracking{ encoding = *, size = -small }{  20 }
\SetTracking{ encoding = *, size = Large- }{ -20 }
```

Letterspaced fonts for which settings don't exist will be spaced out by the default of 0.05 em (adjustable with the package option `letterspace`, see section 3.4). Suppose your editor wants you to shorten your 1000-pages chef-d'œuvre by a handful of pages, you could load `microtype` with (fingers crossed):

```
\usepackage[tracking=alltext,letterspace=-40]{microtype}
```

5.4 Additional kerning

pdfTEX 1.40

`\SetExtraKerning` [⟨*options*⟩] {⟨*set of fonts*⟩} {⟨*kerning settings*⟩}

With this command, you can fine tune the extra kerning. In contrast to standard kerning, which is always associated with a *pair* of characters, and to tracking, which specifies the space between *all* characters of a font, the extra kerning relates to single characters, that is, whenever a particular character appears in the text, the specified kerning will be inserted, regardless of which character precedes resp. follows it. (Put differently, this feature allows modifying the left or right *sidebearings* of specific glyphs.)

It should not be neglected to mention a limitation of this feature: words *immediately following* such a kern (not separated by a space) will not be hyphenated, unless you insert the breakpoints manually, e.g., for kerning after the apostrophe, '`l'apos\-trophe`'. Furthermore, additional kerning will not be applied in math mode. These restrictions of pdfTEX will hopefully be lifted some time.

The kerning settings are specified as pairs of ⟨*character*⟩ = ⟨*kerning values*⟩, where the latter consist of two values: the kerning added before the character, and the kerning appended after the respective character. Once again, either value may be omitted, but not the separating comma.

Options:

`name`, `load`, `factor`, `preset`, `inputenc` These options serve the same function as in the previous configuration commands.

`unit` Admissible values are: `space`, `character` and a ⟨*dimension*⟩. By default, the values denote thousandths of 1 em.

`context` When it comes to kerning settings, this option is especially useful, since it allows applying settings depending on the current language.

For example, you can find the following settings, intended to be used for documents written in French, in the main configuration file:

```
\SetExtraKerning
   [ name     = french-default,
     context  = french,
     unit     = space   ]
   { encoding = {OT1,T1,LY1} }
   {
     :  = {1000,}, % = \fontdimen2
     ;  = {500, }, % ~ \thinspace
     !  = {500, },
     ?  = {500, }
   }
```

What is the result of these settings ? If they are active, like in the current paragraph, a thin space will be inserted in front of each question mark, exclamation mark and semicolon ; a normal space in front of the colon. Read section 6 to learn how to activate these settings ! This paragraph was input like this :

```
\begin{microtypecontext}{kerning=french}
 What is the result of these settings? If they are active, like in the
 current paragraph, a thin space will be inserted in front of each
 question mark, exclamation mark and semicolon; a normal space in front
 of the colon. Read section~\ref{sec:context} to learn how to activate
 these settings! This paragraph was input like this:
\end{microtypecontext}
```

5.5 Interword spacing

pdfTEX 1.40

`\SetExtraSpacing` [⟨*options*⟩] {⟨*set of fonts*⟩} {⟨*spacing settings*⟩}

This command allows you to fine tune the interword spacing (also known as glue). A preliminary remark on what a 'space' is may be in order: between two words, TEX will insert a so called glue, which is characterised by three parameters – the normal distance between two words, the maximum amount of space that may be added to it, and the maximum amount that may be subtracted. The latter two parameters come into effect whenever TEX tries to break a paragraph into lines and does not succeed; it can then stretch or shrink the spaces between words. These three parameters are specific to each font.

On top of these glue dimensions, TEX has the concept of 'space factors'. They may be used to increase the space after certain characters, most prominently the punctuation characters. pdfTEX's additional spacing adjustment may be considered as an extension to space factors with much finer control: while space factors will influence all three parameters of interword space (or glue) by the same amount – the kerning, the maximum amount that the space may be stretched and the maximum amount that it may be shrunk – you may modify these parameters independently from one another. Furthermore, the values may be set differently for each font. And, probably most importantly, the parameters may not only be increased but also decreased. Note that when interword spacing adjustment is in effect, space factors are ignored.

The spacing settings are declared as pairs of ⟨*character*⟩ = ⟨*spacing factors*⟩, where the latter consist of three numbers: first, the additional kern inserted after this

character if it appears before an interword space, second, the additional stretch amount, and third, the additional shrink amount. All values may also be negative, in which case the dimensions will be decreased. Not all values have to be specified, but the settings must always contain the two separating commas.

Options:

`name`, `load`, `factor`, `preset`, `inputenc`, `context` These options serve the same function as in the previous configuration commands.

`unit` You can specify the unit by which the specified numbers are measured. Possible values are: `character`, a ⟨*dimension*⟩ and, additionally, `space`. The latter will measure the values in thousandths of the respective space dimension set by the font. By default, the unit is measured by the space dimensions. For example, with the following (nonsensical) settings:

```
\SetExtraSpacing
   [ unit = space  ] % default
   { font = */*/*/* }
   {
       .  = {1000,1000,1000},
   }
```

the space inserted after a full stop would be doubled (technically speaking: 2 × `\fontdimen` 2), as would the maximum stretch and shrink amounts of the interword space (`\fontdimen` 3 and 4). Conversely, setting all three values to −1000 would completely cancel a space after the respective character.

5.6 Character inheritance

`\DeclareCharacterInheritance` [⟨*features*⟩] {⟨*set of fonts*⟩} {⟨*inheritance lists*⟩}

In most cases, accented characters should inherit the settings from the respective base character. For example, all of the characters À, Á, Â, Ã, Ä, Å and Ă should probably be protruded by the same (absolute) amount as the character A. Using the command `\DeclareCharacterInheritance`, you may declare such classes of characters, so that you then only have to set up the respective base character. With the optional argument, which may contain a comma-separated list of features, you can confine the scope of the list. Additionally, it accepts the `inputenc` key to set the input encoding for this list. The font set can be declared in the usual way. The inheritance lists are declared as pairs of ⟨*base character*⟩ = ⟨*list of inheriting characters*⟩.

With pdfTEX, there should be no need to change the default character inheritance settings, unless you are using a different encoding or a very peculiarly shaped font. The situation is different with LuaTEX and XETEX, however: the default inheritance settings only contain those glyhps that can safely be assumed to exist in any font; but since OpenType fonts may contain many more glyphs for different scripts (languages), it is quite probable that font-specific settings are necessary, which should be specified in the font's configuration file (see next section).

Additionally, this command provides the possibility to have the desired protrusion amount calculated automatically based on the difference of character widths. If the base character is preceded with either '`(l)`', '`(r)`' or '`(lr)`', the characters in the list

will not only inherit the protrusion of the base character, but will be additionally protruded by the result of `charwidth(`⟨*inheriting character*⟩`)`−`charwidth(`⟨*base character*⟩`)`: for '`(l)`' and '`(r)`', this amount will be added to the given protrusion on the respective side, for '`(lr)`' half of the amount will be added to both sides. So that, e.g., with

```
\DeclareCharacterInheritance
  { encoding = TU, family = {GFS Porson} }
  {  (l)H = {᾽H},
    (lr)I = {T}  }
```

the accent before the 'H' would be fully protruded into the left margin, while the stem of the 'T' would be aligned with that of the 'I' if at either margin. This feature is especially useful for Greek typography, where capital letters at the beginning of a line should be placed as if they had no accent (the 'H' in the example above). Base characters may have both unprefixed and prefixed inheritance lists; characters in the latter will be implicitly added to the corresponding unprefixed one. Inheriting characters may appear in both '`(l)`' and '`(r)`'-prefixed lists at the same time, whereas lists prefixed with '`(l)`' or '`(r)`' and those prefixed with '`(lr)`' are mutually exclusive. For all other features except protrusion, these prefixed lists will be ignored.

5.7 Configuration files

The default configuration, consisting of inheritance settings, declarations of font sets and alias fonts, and generic protrusion, expansion, spacing and kerning settings, will be loaded from the file `microtype.cfg`. You may extend this file with custom settings (or load a different configuration file with the '`config`' option, see section 3.5).

If you embark on creating new settings for a font family, you should put them into a dedicated file, whose name must be: '`mt-`⟨*font family*⟩`.cfg`' (e.g., '`mt-cmr.cfg`'; any spaces in the font name should be removed, e.g., '`mt-MinionPro.cfg`'), and may contain all commands described in the current section 5. These files will be loaded automatically if you are actually using the respective fonts. The `microtype` package ships with configuration files for a number of font families. They are listed in tables 3 (Type 1 fonts) and 4 (OpenType fonts).

`\DeclareMicrotypeVariants` {⟨*list of suffixes*⟩}
`\DeclareMicrotypeVariants*`

On its search for a configuration file, the package will also try to remove from the font name a suffix of one or more letters that denotes a 'variant' of the base font (cf. Karl Berry's Fontname). It is thus possible to put settings for, e.g., the fonts `pplx` (expert set), `pplj` (oldstyle numerals) and `ppl` (plain) into one and the same file `mt-ppl.cfg`. This command expects a comma-separated list of variant suffixes. The starred version appends the suffix(es) to the existing list. The default declaration in `microtype.cfg` is:

```
\DeclareMicrotypeVariants{x,j,w,a,d,0,1,-LF,-TLF,-OsF,-TOsF}
```

`\DeclareMicrotypeAlias` {⟨*font name*⟩} {⟨*alias font*⟩}

This command may be used for fonts that are very similar, or actually the same (for instance if you did not stick to the Berry naming scheme when installing a font).

Table 3: Type 1 fonts with tailored protrusion settings

Font family (NFSS code)	Features	
	Encodings	Shapes
Generic	OT1, T1, T2A, LY1, QX, (TS1) *l*	n, (it, sl, sc) *l*
Computer Modern Roman (cmr) *a*	OT1, OT4, T1, T2A, T5, LY1, TS1	n, it, sl, sc
Bitstream Charter (bch) *b*	OT1, T1, T5, LY1, TS1	n, it, (sl) *m*, sc
EB Garamond *c*	OT1, T1, LY1, LGR, TS1	n, it, sc, si
URW Garamond (ugm) *d*	OT1, T1, TS1	n, it
Bitstream Letter Gothic (blg) *e*	OT1, T1, TS1	n, it
Adobe Minion (pmnx, pmnj) *f*	OT1, T1, T2A, LY1, TS1	n, it, (sl) *m*, sc, si
Palatino (ppl, pplx, pplj) *g*	OT1, OT4, T1, LY1, (TS1) *l*	n, it, (sl) *m*, sc
Times (ptm, ptmx, ptmj) *h*	OT1, OT4, T1, LY1, QX, (TS1) *l*	n, it, (sl) *m*, sc
Computer Modern math (cmsy, cmm) *i*	OML/OMS	n/it
AMS symbols (msa, msb)	U	n
Euler (eur, eus, euf) *j*	U	n
Euro symbols (Adobe, ITC, marvosym)	U/OT1	n, it

a Aliases: Latin Modern Roman (lmr), ae (aer), zefonts (zer), eco (cmor), hfoldsty (hfor), mlmodern (mlmr)
b Aliases: mathdesign/Charter (mdbch), MicroPress's chmath (chr), XCharter
c Alias: Adobe Garamond (pad, padx, padj)
d Aliases: mathdesign/URW Garamond (mdugm), garamondx (zgmx, zgmj)
e Alias: ulgothic (ulg)
f Alias: Crimson
g Aliases: pxfonts (pxr), qfonts/QuasiPalatino, TEX Gyre Pagella (qpl), newpx, FPL Neu (fp9x, fp9j), domitian
h Aliases: txfonts (txr), qfonts/QuasiTimes, TEX Gyre Termes (qtm), newtx, tempora, step, stix/stix2/stickstoo
i Aliases: Latin Modern (lmsy, lmm), mlmodern (mlmsy, mlmm)
k Alias: eulervm (zeur, zeus)
l Incomplete
m Settings inherited from italic shape

An example would be the Latin Modern fonts, which are derived from Computer Modern, so that it is not necessary to create new settings for them – you could say:

```
\DeclareMicrotypeAlias{lmr}{cmr}
```

which would make the package, whenever it encounters the font lmr and does not find settings for it, also try the font name cmr. In fact, you will find this very line, along with some others, in the default configuration file.

`\LoadMicrotypeFile` {⟨*font name*⟩}

In rare cases, it might be necessary to load a font configuration file manually, for instance, from within another configuration file, or to be able to extend settings defined in a file that would otherwise not be loaded automatically, or would be loaded too late.[8] This command will load the file '`mt-`⟨*font name*⟩`.cfg`'.

`\DeclareMicrotypeFilePrefix` {⟨*prefix*⟩}

Or, in fact, it will load the file '⟨*prefix*⟩`-`⟨*font name*⟩`.cfg`'. The ⟨*prefix*⟩ may be

8 Font package authors might also want to have a look at the hook `\Microtype@Hook`, described in the Implementation part, section 1.4.4.

Table 4:
OpenType fonts with tailored protrusion settings

Font family	Features	
	Scripts	Shapes
Generic	Latin	n, (it, sl, sc) [c]
Latin Modern Roman	Latin	n, it, (sl) [d]
New Computer Modern [a]	Latin, Greek	n, it, (sl) [d]
Charis SIL	Latin, Cyrillic	n, it, sc
EB Garamond	Latin, Cyrillic, Greek	n, it, sc, si
Palatino [b]	Latin	n, it, sc

a Alias: CMU Serif
b Aliases: Palatino Linotype, Palatino LT Std, TEX Gyre Pagella, Domitian
c Incomplete
d Settings inherited from italic shape

changed, which, however, you should only do if you have your own set of configuration files that you want to be loaded *instead of* any of those provided with the package.

6 Context-sensitive setup

pdfTEX 1.40 | LuaTEX 0.30

The `microtype` package also allows applying different micro-typographic settings to the fonts depending on the context in which they occur. This opens up the space for infinite possibilities of tweaking the document's appearance.

`\microtypecontext` {⟨*context assignments*⟩}

This command may be used anywhere in the document (also in the preamble) to change the micro-typographic context in the current group. To each feature (`protrusion`, `expansion`, (or `activate` as a shortcut for both), `tracking`, `spacing` and `kerning`), one context may be assigned. Consequently, only settings with the corresponding '`context`' keyword will be applied.

`\begin{microtypecontext}` {⟨*context assignments*⟩}
`\end{microtypecontext}`

Like many LATEX commands, it is also available in the form of an environment.

`\textmicrotypecontext` {⟨*context assignments*⟩} {⟨*general text*⟩}

As another possibility, the command `\textmicrotypecontext` sets the context(s) for the text given in the second argument.

Suppose you want the footnote markers in the text to be protruded by a larger amount. You could define settings for the numbers:

```
\SetProtrusion
   [ context = footnote ]
   { font    = */*/*/*/scriptsize } % adapt if necessary
   { 1 = { ,650}, 2 = { ,400}, 3 = { ,400}, 4 = { ,400}, 5 = { ,400},
     6 = { ,400}, 7 = { ,500}, 8 = { ,400}, 9 = { ,400}, 0 = { ,400} }
```

and have the context changed in the footnote marker command. This command differs among the various classes; for the base classes, e.g., `article`, it would be:

```
\newcommand*\new@makefnmark{\hbox{\@textsuperscript{\normalfont
   \microtypecontext{protrusion=footnote}\@thefnmark}}}
\renewcommand*\@footnotemark{%
   \leavevmode \ifhmode\edef\@x@sf{\the\spacefactor}\nobreak\fi
   \new@makefnmark \ifhmode\spacefactor\@x@sf\fi \relax}
```

For the `memoir` class, you would additionally have to disable auto-detection of multiple footnotes, which prevents protrusion:

```
\renewcommand*\@makefnmark{\hbox{\@textsuperscript{\normalfont
   \microtypecontext{protrusion=footnote}\@thefnmark}}}
\let\m@mmf@prepare\relax
\let\m@mmf@check\relax
```

Another possibility would be to employ contexts for a language-dependent setup. For instance, if you are writing a text in French, you could add:

```
\microtypecontext{kerning=french}
```

to the preamble. This would have the effect that kerning settings for the French context would be applied to the document. Should parts of the document be in English, you could write:

```
\textmicrotypecontext{kerning=}{English text!}
```

to reset the context, so that the punctuation characters in these parts will not receive any extra kerning.

Instead of adding these commands manually to your document, you may also load `microtype` with the `babel` option (see section 3.5). The current language will then be automatically detected and the contexts set accordingly.

`\DeclareMicrotypeBabelHook` {⟨*list of babel/polyglossia languages*⟩} {⟨*context list*⟩}

Naturally, `microtype` does not know about the typographic specialties of every language. This command is a means of teaching it how to adjust the context when a particular language is selected. The main configuration file contains among others the following declaration:

```
\DeclareMicrotypeBabelHook
   {french,francais,acadian,canadien}
   {kerning=french, spacing=}
```

Consequently, whenever you switch to the French language, the kerning context will be changed to 'french' and the spacing context will be reset. This hook only has an effect if the package was loaded with the `babel` option. Currently, `microtype` supports French and Turkish kerning and English spacing (aka. `\nonfrenchspacing`). For unknown languages, all contexts will be reset.

7 Letterspacing revisited

pdfTEX 1.40 | LuaTEX 0.62 | XETEX 0.9997

`\textls` [⟨*amount*⟩] {⟨*general text*⟩}

While the `tracking` feature, described in section 5.3, will apply to sets of fonts, you may also want to letterspace shorter pieces of text, regardless of the font in

which they are typeset.[9] For such ad-hoc letterspacing, `microtype` introduces two commands that can be used (independently of whether the `tracking` option is enabled) in the same way as LaTeX's text commands: `\textls` – which also works in math mode – expects the text in the mandatory argument, while `\lsstyle` will switch on letterspacing for all subsequent fonts until the end of the current group. The starred version of `\textls` does not add any extra kerning before or after the text, which may be useful, e.g., for section titles. By default, each character will be spaced out by 50/1000 em = 0.05 em; this amount may be altered in the optional argument to `\textls`, using the `\SetTracking` command, or globally with the `letterspace` package option, with decreasing significance in this order.

`\lsstyle` `\textls*`

`\lslig` {⟨*ligature*⟩}

Since the commands `\textls` and `\lsstyle` will also evaluate the 'no ligatures' key for the respective font (as well as all other options mentioned in section 5.3), you usually need not worry about protecting or breaking ligatures with most fonts. However, in certain situations, there may be a conflict of ligatures beginning with the same letter, where some of them should be inhibited, while others should not. For example, when letterspacing text typeset in Fraktur fonts, the ligatures 'ch', 'ck', 'tz' and 'sz' ('ß') should never be broken up; you also usually see the 'st' ('ſt') ligature in letterspaced text. Furthermore, at least the `yfonts` package realises the short s ('ſ') as the pseudo-ligature 's:'. On the other hand, the 'ct' ligature and the other 'long s' ligatures often found in Fraktur fonts should be suppressed. There are two ways of solving this problem: either don't disable the 's' and/or 'c' ligatures and break those that need to be broken up by inserting '{\kern0pt}' or `babel`'s `"|` shortcut; or disable them and protect those ligatures that need to be protected by enclosing them in the `\lslig` command. So, the following two solutions have the same result (namely, 'Aus ſ ichts ſ o ſ igkeit', with ligatures here highlighted in green, inhibited ligatures in red):

```
\SetTracking[no ligatures={f}]{encoding = LY, family = yfrak}{120}
\textfrak{\lsstyle Aus:s{\kern0pt}ichts:los{\kern0pt}igkeit}
```

```
\SetTracking[no ligatures={f,s,c}]{encoding = LY, family = yfrak}{120}
\textfrak{\lsstyle Au\lslig{s:}si\lslig{ch}t\lslig{s:}losigkeit}
```

`letterspace.sty`

These three commands (plus the `letterspace` option, described in section 3.4) are also available with the alternative `letterspace` package, which is in fact a much stripped-down version of `microtype`, omitting support for all the other extensions (and also omitting the possibilities of the `\SetTracking` command – all 'f' ligatures will be disabled, inner and outer spacing and outer kerning will be set to the default values described in section 5.3). If you would rather forgo `microtype`'s specialties, you may load the `letterspace` package instead. Both packages should not be used at the same time.

In contrast to `microtype`, which requires LaTeX, the `letterspace` package also works with `eplain` or even only `miniltx`: for use with `eplain`, load the package with `\usepackage` inside the `\beginpackages ... \endpackages` environment; with `miniltx` (which does not support package options) simply `\input letterspace.sty`.

9 Letterspacing should be used cautiously; in particular, letterspacing lowercase text is held in abhorrence by honourable typographers. Unless you know what you are doing, you should probably only letterspace capitals or small capitals. Another just cause may be emphasis in texts typeset in Fraktur fonts.

8 Disabling ligatures

pdfTEX 1.30 | LuaTEX 0.30

\DisableLigatures [⟨*characters*⟩] {⟨*set of fonts*⟩}

While completely disabling all ligatures of a font (which will also switch off kerning for this font), purposely *lowers* the micro-typographic quality instead of raising it, it is especially useful for typewriter fonts, so that, e.g., in a T1 encoded font, '\texttt{--}' will indeed be printed as '--', not as '–'. \DisableLigatures may be used to specify, in the usual way, a set of fonts for which ligatures should be disabled, for example, of the typewriter font in T1 encoding:

```
\DisableLigatures{encoding = T1, family = tt* }
```

It is also possible to disable selected ligatures only. The optional argument may contain a comma-separated list of characters for which the ligature mechanism should be inhibited:

```
\DisableLigatures[?,!]{encoding = T1} % inhibit ?‘ and !‘, but not fi, –, », etc.
```

Only the character that begins the ligature(s) should be specified. This command may only be used in the preamble, and only once. [10]

9 Being pedantic about protrusion

e-TEX

Protrusion works well in running paragraphs, but may also be desirable in situations where TEX, in contrast to human eyes, would not see a margin, [11] e.g., in itemize or tabular environments. The microtype package offers two commands that may be inserted at such an effective inner margin to make the first respectively last glyph protrude as if it were located at a normal outer margin:

\leftprotrusion {⟨*general text*⟩}

This command will add left protrusion for the following text. You may also just say \leftprotrusion (without an argument), and microtype will gather the next glyph (possibly a ligature) before adding protrusion on its left hand side. For instance, you could add this command to tabular cell definitions (using the array package):

```
\begin{tabular}{l>{\leftprotrusion}p{9cm}r}
```

in order to get protrusion at the beginning of the p cell.

\rightprotrusion {⟨*general text*⟩}

will typeset ⟨*general text*⟩ and then add protrusion on the right side. (Unfortunately, TEX cannot look backwards at what it has already typeset, so this command requires that the text be given in the argument.)

\noprotrusion is a command from LATEX proper, so it's just mentioned apropos. When added to the beginning or end of a line, protrusion at the respective margin will be prevented.

\noprotrusionifhmode While the latter command will always leave vertical mode first (which in some situations may result in unwanted vertical space), this variant prevents protrusion only if already in horizontal mode.

10 With LuaTEX, you have to load the fonts with the fontspec option 'Renderer=Basic'.

11 Possibly because TEX tends to look through its gastro-intestinal tract.

The microtype package defines a number of patches in order to get protrusion right (indicated in parentheses are dedicated patches for classes and packages beyond the standard ones; they may still work with other classes or packages):

`item` Affects the first line of `\item`s in various environments, so that, e.g., the two A's here – the one in the current line and the one in the line above – are neatly aligned; without the patch, the first A would be slightly shifted to the right. This patch also affects environments such as `quote` or `flushleft`, which are implemented in LaTeX as lists starting with an implicit empty item. (beamer, simplecv)

`toc` Adds protrusion at the left margins of sectioning titles in the Table of Contents and similar Lists of ⟨*Things*⟩. (memoir)

`footnote` Protrusion for the first line of footnote text (visible in particular when the footnote text is set in block paragraphs, like in this document). (memoir, KOMA classes, beamer, changebar, fnbreak, hyperref)

`verbatim` Disables protrusion and expansion in verbatim environments. (alltt)

`eqnum` Protrusion for equation numbers on either side. If you are using the interface provided by `mathtools`, insert the relevant commands yourself, e.g.:

```
\newtagform{mytag}[\textbf]{\leftprotrusion{(}}{\rightprotrusion{)}}
```

(amsmath, IEEEtran, showkeys)

By default, all of the above patches will be applied. Should this not be desired, or in case you are running into problems (or just want to silence a warning that a patch cannot be applied – which in itself only means that no harm is done, but no good either), you may prevent or undo the patches through the `patch` and `nopatch` options, either when loading the package:

```
\usepackage[nopatch=toc]{microtype}
```

or, possibly temporarily, within the document:

```
\microtypesetup{nopatch=item}
```

Help wanted! These pedantic protrusion patches are still work-in-progress. I suspect many use cases where they either do not work as advertised, or worse, where they may even lead to errors. Also, I am aware that there are myriads of classes and packages out there that modify internal commands in their own ways, possibly rendering the patches useless. Finally, there will certainly be many more situations where protrusion would be appropriate. I would be happy to include more patches, or enhance the existing ones, so I welcome any suggestions and problem reports you may have.

10 Creating configurations and contributing

I would also be glad to include configuration files for more fonts. Preparing such configurations is quite a time-consuming task and requires a lot of patience. To alleviate this process, microtype includes the companion package `microtype-show`, which offers some tools to visually debug protrusion settings. It should not be used in production contexts.

`microtype-show.sty`

\ShowCharacterInheritance
\ShowProtrusion

These commands show all defined inheritance resp. protrusion settings (the latter relative to character width, to 1 em, and as the effective kerning amount) for the current font. They are most useful for consistency checks.

\ShowMissingGlyphstrue
\ShowGlyphIndextrue
\GlyphScaleFactor

If the boolean `\ifShowMissingGlyphs` is set to true, the glyphs *not* included in the configuration will also be shown. Setting the switch `\ifShowGlyphIndex` to true will additionally display the glyph (pdfTEX) respectively Unicode (LuaTEX) index number. Furthermore, you may alter the glyphs' display size by redefining `\GlyphScaleFactor` (default: 2).

\ShowProtrusionAll
\ShowProtrusionDefined
\ShowProtrusionMissing

When preparing the actual protrusion settings, these commands may prove helpful. Conveniently placing the glyphs at the margins, they show, respectively: all glyphs in the current font; only those with defined protrusion settings; or those without any protrusion settings. (Note that here, the protrusion amounts are given in 1000ths of 1 em, not in 1000ths of character widths). The package also includes a test file that shows all of these commands in action (`test-microtype.tex`).

If you have created a configuration file for another font, or if you have any suggestions for enhancements in the default configuration files, I would gratefully accept them: w.m.l@gmx.net.

Development of the package takes place on GitHub, which also provides an issue tracker for submitting bug reports and feature requests: https://github.com/schlcht/microtype/issues.

11 Hints and caveats

Use settings that match your font. Although the default settings should give reasonable results for most fonts, the particular font you happen to be using may have different character shapes that necessitate more or less protrusion. In particular, italic letter shapes may differ wildly in different fonts, hence I have decided against providing default protrusion settings for them. See the previous section for some tools for the preparation of protrusion settings.

Don't use too large a value for expansion. Font expansion is a feature that is supposed to enhance the typographic quality of your document by producing a more uniform greyness of the text block (and potentially reducing the number of necessary hyphenations). When expanding or shrinking a font too much, the effect will be turned into the opposite. Expanding the fonts by more than 2%, i.e., setting a `stretch` limit of more than 20, should be justified by a typographically trained eye. If you are so lucky as to be in the possession of multiple instances of a Multiple Master font, you may set expansion limits to up to 4%.

Settings for Greek/Thai/Armenian etc. encodings are not yet included. The default sets of fonts for which the micro-typographic features will be enabled (see table 2) only contain those encodings for which configurations exist. Therefore, if you are using any other encoding (e.g., LGR, T2B, etc.), `microtype` will not apply to these fonts. You have to define and activate a new font set including the encoding(s) you are using (for details, see section 4). For protrusion at least, you would also have to create settings for the fonts in question (see section 5.1). It goes without saying that contributions for these encodings are more than welcome.

Only employ kerning adjustment if it is customary in the language's typographic tradition. In contrast to protrusion and expansion, additional kerning does not unconditionally improve the micro-typographical quality of your document. You should only switch it on if you are writing a document in a language whose typographic tradition warrants such kerning. If you are, for example, writing an English text, your readers would probably be rather confused by additional spaces before the punctuation characters.

Adjustment of interword spacing is still experimental. The implementation of this feature in pdfTEX is not complete, and may not yield the positive effects on the typographical quality you might expect – in certain situations, there may even be undesired side effects, in particular, when used together with the `ragged2e` package. Therefore, the `spacing` option should not be chosen blindly; it is also recommended to experiment with the settings in order to understand the workings of this feature.

Compatibility and interaction with other packages: The `microtype` package is supposed to work happily together with all other LATEX packages (except for `pdfcprot`). However, life isn't perfect, so problems are to be expected. Currently, I am aware of the following issues:

- Even though most configuration files are still provided in legacy (7-bit) format, using multi-byte (Unicode) characters in the settings will run smoothly with an up-to-date LATEX system. For older systems or documents in legacy encodings, in contrast, the `inputenc` package must be loaded first. Furthermore, when using multiple input encodings in a document, 8-bit characters in the settings will only work reliably if you specify the `inputenc` key.
- When loading the package with the `babel` option, you must load the `babel` or `polyglossia` package before `microtype`.
- Before this package was fully compatible with LuaTEX, the following method of enabling expansion and protrusion with the `fontspec` package was most often found to be recommended:

```
\newfontfeature{Microtype}{protrusion=default;expansion=default}
\defaultfontfeatures{Microtype}
```

  This code should *not* be used with this package, as it will basically override all of the settings made by `microtype` – despite the naming, the above lines have nothing to do with this package.[12]
- With pdfTEX, it is currently not possible to create character-specific settings for Chinese/Japanese/Korean fonts. Therefore, the only micro-typographic extension that can be made to work with `CJK` fonts is (non-selected) font expansion.
- When used with the `xeCJK` package or the `luatexja` package, text commands (e.g., `\'A`, `\textless`) in the configuration will not be understood. You therefore have to ensure that `microtype` will encounter none of them. This requires, firstly, that the glyphs be specified only as single (possibly Unicode) characters, as numbers, or as glyph names (cf. section 5); and secondly, if you are using a font for which pre-defined settings do not exist, that you create these settings yourself (because otherwise, the default settings will be loaded, which do contain text commands). Furthermore, you should load `microtype` late.

12 They make use of features provided by `luaotfload` (via `fontspec`).

Possible warning and error messages and how to get rid of them (specs may differ):

- `Package microtype Warning: Unknown slot number of character`
  `(microtype)               `\k A'`
  `(microtype)               in font encoding `TU' in inheritance list`
  `(microtype)               `microtype.cfg/424(protrusion)'.`

  If you receive lots of warnings like the above, and you are running LuaTEX or XETEX, this probably means that the font you are using contains less glyphs than are defined in the default protrusion and/or inheritance settings. For such fonts, the `microtype` package defines basic inheritance settings to which you may alias your font (the generic protrusion settings will still be loaded). Try adding the line:

  ```
  \DeclareMicrotypeAlias{⟨your font⟩}{TU-basic}
  ```

  to the preamble. For symbol fonts, you may even use the alias `TU-empty`, which is, well, empty, meaning that no protrusion will be applied at all to this font. Note that these aliases are just meant to provide an easy way of getting rid of the warnings, while the proper solution would be to create inheritance and protrusion settings specific to the font in question.

- `Package microtype Warning: Unable to apply patch `footnote' on input line 29.`

  As described in chapter 9 above, `microtype` tries to patch a number of commands to enable or disable protrusion in certain situations. If you receive a warning like the above, you should first make sure (as always) that the package is up to date. If it is, consider filing a bug report. Until a fix is available, or if you are, for whatever reason, unable or unwilling to update, the easiest solution to get rid of the warning is to disable the problematic patch with the `nopatch` option, e.g.:

  ```
  \microtypesetup{nopatch=footnote}
  ```

- `! Font csnameendcsname=cmr10+20 at 10.0pt not loadable: Metric (TFM) file not found.`

  This error message will occur if you are trying to employ font expansion while creating DVI output. Remember that *automatic* font expansion only works when running pdfTEX or LuaTEX in PDF mode. Although expansion is also possible in DVI mode with pdfTEX, it requires that all instances of the expanded fonts exist on your TEX system.

- `! pdfTeX error (font expansion): auto expansion is only possible with scalable fonts.`

  Beginning with pdfTEX 1.40, font expansion not only works with Type 1 fonts but also with TrueType, OpenType and even non-embedded fonts. The above error message indicates either that you are trying to apply expansion to a bitmap (`pk`) font, which is still not possible, or that the font isn't found at all, e.g., because of missing map entries.

- `Warning: pdflatex: font ptmr8r cannot be expanded (not an included Type1 font)`

  and the PDF viewer complains about a missing font, e.g., Adobe Reader thusly:

  `Could not find a font in the Resources dictionary - using Helvetica instead.`

  With pdfTEX versions older than 1.40, font expansion can only be applied if the font is actually embedded in the PDF file. If you get the above error message, your TEX system is not set up to embed (or 'download') the base PostScript fonts (e.g., Times, Helvetica, Courier). In most TEX distributions, this can be changed in the file `updmap.cfg` by setting `pdftexDownloadBase14` to `true`.

- `Warning: pdflatex (file ecrm1000+20): Font ecrm1000+20 at 1200 not found`

  Furthermore, pdfTEX versions older than 1.40 require Type 1 fonts for automatic font expansion. When you receive a message like the above, you are probably trying to apply font expansion to a bitmap or TrueType font. With older pdfTEX versions, this is only possible if you manually create expanded instances of the fonts.

- `! Font T1/cmr/m/n/10=ecrm1000 at 10.0pt not loaded: Not enough room left.`

  Memory parameter '`font_mem_size`' too small.

- `! TeX capacity exceeded, sorry [maximum internal font number (font_max)=2000].`

  Memory parameter '`font_max`' too small.

- `! TeX capacity exceeded, sorry [PDF memory size (pdf_mem_size)=65536].`

  Memory parameter '`pdf_mem_size`' too small (pdfTEX versions older than 1.30).

  When applying micro-typographic enhancement to a large document with a lot of fonts, pdfTEX may be running out of some kind of memory. It can be increased by setting the respective parameter to a larger value. For web2c-based systems, e.g., TEX Live, change the settings in `texmf.cnf`, for MiKTEX, in the file `miktex.ini` (2.4 or older) resp. `pdflatex.ini` (2.5 or newer).

- `pdfTeX warning (font expansion): font should be expanded before its first use`

  This warning will occur with pdfTEX versions older than 1.40.4, if tracking *and* expansion is applied to a font. It is harmless and can be ignored.

The source code of this document is freely available. If you wonder how this document was created, just have a look at the source code in `microtype.dtx`, which is either already included in your TEX distribution, or else can be downloaded from CTAN. For the source code of the logo on the title page and of the letterspacing sample from section 5.3, see the Implementation part, appendices A and B. If you want to re-typeset the documentation, read the comments at the end of `microtype.dtx`.

12 Acknowledgments

This package would be pointless if *Hàn Thế Thành* hadn't created the pdfTEX program in the first place, which introduced the micro-typographic extensions and made them available to the TEX world. Furthermore, I thank him for helping me to improve this package, and not least for promoting it in Thành 2004, Thành 2008 and elsewhere. I also thank him and the rest of the pdfTEX team, and more recently also the LuaTEX and XƎTEX teams, for refuting the idea that TEX is dead, and for fixing the bugs I find.

Harald Harders has contributed protrusion settings for Adobe Minion. I would also like to thank him for a number of bug reports and suggestions he had to make. *Andreas Bühmann* has suggested the possibility to specify ranges of font sizes, and resourcefully assisted in implementing this. He also came up with some good ideas for the management of complex configurations. *Ulrich Dirr* has made numerous suggestion, especially concerning the new extensions of interword spacing adjustment and additional character kerning. *Georg Duffner* has patiently tested `microtype` under XƎTEX and LuaTEX with his beautiful OpenType font EB Garamond. My thanks also go to *Maciej Eder* for contributing settings for the QX encoding, as

well as to *Karl Karlsson* for providing settings for the Cyrillic T2A encoding, and to *Hendrik Vogt*, who made substantial improvements to the Computer Modern Roman italic settings. I thank *Loren B. Davis* for providing protrusion settings for OpenType versions of Palatino Linotype, as well as *Antonis Tsolomitis* for settings for his New Computer Modern font. The latter also showcase the new feature of automatically calculated protrusion, which was an original idea by *Daniel Benjamin Miller*. I am also very much indebted to *Élie Roux*, who not only contributed the `lua` module in the first place, but also, together with *Philipp Gesang*, took care of updating it for the developments in LuaTEX land.

I thank *Philipp Lehman* for adding to his `csquotes` package the possibility to restore the original meanings of all activated characters, thus allowing for these characters to be used in the configuration files. *Peter Wilson* kindly provided a hook in his `ledmac`/`ledpar` packages (inherited by their successors `reledmac`/`reledpar`), so that critical editions can also benefit from character protrusion. Likewise, *Donald Arseneau* patched his `shapepar` package to accommodate protrusion.

Additionally, the following people have reported bugs, made suggestions or helped otherwise (in chronological order, quotes indicate TeX.SX and/or GitHub user names): *Tom Kink*, *Herb Schulz*, *Michael Hoppe*, *Gary L. Gray*, *Georg Verweyen*, *Christoph Bier*, *Peter Muthesius*, *Bernard Gaulle*, *Adam Kucharczyk*, *Mark Rossi*, *Stephan Hennig*, *Michael Zedler*, *Herbert Voß*, *Ralf Stubner*, *Holger Uhr*, *Peter Dyballa*, *Morten Høgholm*, *Steven Bath*, *Daniel Flipo*, *Michalis Miatidis*, *Sven Naumann*, *Ross Hetherington*, *Wiebke Petersen*, *Geoff Vallis*, *Steven E. Harris*, *Karl Berry*, *Peter Meier*, *Nathan Rosenblum*, *Wolfram Schaalo*, *Vasile Gaburici*, *Sveinung Heggen*, *Axel Berger*, *Colin Rourke*, *Maverick Woo*, *Silas S. Brown*, *Lars Rönnbäck*, *Christian Stark*, *Leo*, *Marcin Borkowski*, *hscm*, *George Gratzer*, *Josep Maria Font*, *Juan Acevedo*, *Heiko Oberdiek*, *Till A. Heilmann*, *Rolf Dieterich*, *Seamus Bradley*, *Meho R*, *Steffen Hoffmann*, *Scott Pakin*, *Maïeul Rouquette*, *Jonas Hogstrom*, *Gabriel Kerneis*, *'RazorXsr'*, *Sebastian Schubert*, *'Dave'*, *Giuseppe Palma*, *Stephan Stiller*, *Christopher Schramm*, *'uli'*, *Sam Mason*, *'kleenstar'*, *'Henning'*, *Ronnie Marksch*, *David Carlisle*, *'web-stranger'*, *'Max'*, *'HcN'*, *Will Robertson*, *'user11126'*, *Ulrike Fischer*, *'Daniel'*, *'ltcomdata'*, *Reinhard Kotucha*, *'jcr'*, *Nils Anders Danielsson*, *Paolo Ney*, *Frank Mittelbach*, *Franz Wexler*, *Moritz Wemheuer*, *'Andy N'*, *Phelype Oleinik*, *Falk Hanisch*, *Markus Kohm*, *Paolo Polesana*, *Oliver Kopp*, *Hironori Kitagawa*, *Aman Mehra*, *Md Ayquassar*, *Holger Gerhardt*, *Marcel Krüger*, *Ekkehart Schlicht*, *'Canageek'*, *'dsedivec'*, *'DORpapst'*, *'chsk'*, *'tnull'*, *'azur'*, *'Safron'*, *Joseph Wright*, *Gustavo Barros*, *Torsten Schuetze*, *'florian'*, *Liang-Bo Wang*, *'MisterFiLou'*, *Akira Yokosawa*, *'theufman'*, *'frafl'*, *Joel Coffman*, *'user182849'*, *'NightShade'*, *Nelson Lago*, *Brian Dunn*, *Ralf Steinle*, *Denis Bitouzé*, *Christophe Dervieux*, *Scott Kostyshak*, *Shen Zhou Hong*, *David Purton*, *'rallg'*, *Artur A. Marczok*, *Uwe Siart*, *'simon-codes-something'*, *'scholnik'*, *Ulrich Schwarz*, *Bruno Victal*, *Linas Stonys*, *'user202729'*, *Oliver Beery*, *Bernhard Fisseni*, *Aleksandr Petrosyan*, *Didier Verna*, *'nowox'*, *'hpvd'*, *Mark Collins*, *Nick Bart*, *'Rimole'*, *Clea F. Rees* and *Joerg Klein*.

13 References

Hàn Thế Thành, 'Micro-typographic extensions to the TEX typesetting system', Diss. Masaryk University Brno 2000, in: *TUGBoat*, vol. 21 (2000), no. 4, pp. 317–434. (Online at https://www.tug.org/TUGboat/Articles/tb21-4/tb69thanh.pdf)

Hàn Thế Thành, 'Micro-typographic extensions of pdfTEX in practice', in: *TUGBoat*, vol. 25 (2004), no. 1: 'Proceedings of the Practical TEX 2004 Conference', pp. 35–38. (Online at https://www.tug.org/TUGboat/Articles/tb25-1/thanh.pdf)

Hàn Thế Thành, 'Font-specific issues in pdfTEX', in: *TUGBoat*, vol. 29 (2008), no. 1: 'EuroBachoTEX 2007 Proceedings', pp. 36–41. (Online at https://www.tug.org/TUGboat/Articles/tb29-1/tb91thanh-fonts.pdf)

Hàn Thế Thành and others, *The pdfTEX user manual*, 12 February 2025. (Available from CTAN at /systems/doc/pdftex/manual/pdftex-a.pdf)

Karl Berry, *Fontname: Filenames for TEX fonts*, July 2009. (Available from CTAN at /info/fontname/fontname.pdf)

LATEX Project Team, *LATEX 2ε font selection*, September 2024. (Available from CTAN at /macros/latex/base/fntguide.pdf)

Will Robertson, *The fontspec package: Font selection for XETEX and LuaLATEX*, 11 May 2024. (Available from CTAN at pkg/fontspec)

LATEX3 Project, Élie Roux, Khaled Hosny, Philipp Gesang, Ulrike Fischer, Marcel Krüger, *The luaotfload package*, 3 December 2024. (Available from CTAN at pkg/luaotfload)

Carsten Schurig, Tobias Schlemmer, *The pdfcprot.sty package*, 10 June 2005. (Available from CTAN at pkg/pdfcprot)

[Melchior Franz,] Heiko Oberdiek, *The soul package*, 14 June 2023. (Available from CTAN at pkg/soul).

14 Changes

The comprehensive list of changes can be found in the Implementation part, appendix C. The following is a list of all changes relevant in the user land; bug and compatibility fixes are swept under the rug. Numbers in brackets indicate the relevant section in this manual.

3.2 (2024/12/12)

- Support for tracking/letterspacing with XETEX [3.1, 5.3, 7]
- New default for letterspacing: 50 [3.4]
- New key 'features' for \SetTracking to enable/disable fontspec 'Ligatures' features (LuaTEX/XETEX only) [5.3]
- New values 'none' and 'all' for the 'no ligatures' key of \SetTracking [5.3]

3.1 (2023/03/06)

- New command \DeclareMicrotypeFilePrefix to change the prefix of the configuration files [5.7]
- New protrusion patch verbatim to switch off protrusion and expansion in verbatim environments [9]

3.0 (2021/10/31)

- New option disable, replacing the draft option; deprecate option final [3.5]
- Possibility of automatical protrusion based on difference of character widths [5.6]

- New commands `\leftprotrusion` and `\rightprotrusion`; various patches to get protrusion right [9]
- New package `microtype-show` for visual debugging of protrusion settings [10]
- Protrusion settings for New Computer Modern (OpenType)
- Protrusion settings for EB Garamond (OpenType)
- New generic protrusion settings for LuaTeX/XeTeX [11]
- Move development to GitHub

2.8 (2020/12/07)

- New default font sets for expansion and spacing: '`alltext-nott`' [4, table 2]

2.7 (2017/07/07)

- Allow automatic expansion and letterspacing with LuaTeX in DVI mode (aka. `dvilualatex`) [3.1, 3.3, table 1]

2.6 (2016/05/01)

- Support for LuaTeX ≥ 0.85
- Improvements for tracking/letterspacing with LuaTeX (`Renderer=Basic` no longer required)
- New font sets: '`alltext-nott`', '`allmath-nott`' [4, table 2]

2.5 (2013/03/13)

- Support for the `fontspec` package, viz. for OpenType fonts with LuaTeX and XeTeX
- Support for protrusion with XeTeX ≥ 0.9997
- Support for tracking/letterspacing with LuaTeX ≥ 0.62
- Allow context-sensitive setup with LuaTeX
- Info instead of warning if protrusion settings are generic
- Protrusion settings for Latin Modern Roman (OpenType)
- Protrusion settings for Charis SIL (OpenType)
- Protrusion settings for Palatino Linotype (OpenType)

2.4 (2010/01/10)

- Protrusion settings for T2A encoded Minion

2.3e (2009/11/09)

- Support for the Cyrillic T2A encoding (protrusion, expansion, spacing)

2.3d (2009/03/27)

- New default for expansion option '`step`': 1, if pdfTeX ≥ 1.40 [3.3]

2.3c (2008/11/11)

- Support for LuaTeX enabled by default

2.3 (2007/12/23)

- New option '`verbose=silent`' to turn all warnings into mere messages [3.5]
- New key '`outer kerning`' for `\SetTracking` to customise outer kerning [5.3]
- Adjust protrusion settings for tracking even if protrusion is not enabled
- The `letterspace` package also works with `eplain` or `miniltx` [7]

2.2 (2007/07/14)

- Improvements to tracking/letterspacing: retain kerning (pdfTeX ≥ 1.40.4); automatically adjust protrusion settings

- Possibility to expand a font with different parameters (pdfTeX ≥ 1.40.4) [5.2]
- New key 'no ligatures' for \SetTracking to disable selected or all ligatures (pdfTeX ≥ 1.40.4) [5.3]
- New keys 'spacing' and 'outer spacing' for \SetTracking to customise interword spacing [5.3]
- New command \DeclareMicrotypeVariants to specify variant suffixes [5.7]
- New command \textmicrotypecontext as a wrapper for \microtypecontext [6]
- New optional argument for \DisableLigatures to disable selected ligatures [8]
- Protrusion settings for Bitstream Letter Gothic

2.1 (2007/01/21)

- New command \lslig to protect ligatures in letterspaced text [7]

2.0 (2007/01/14)

- Support for the new extensions of pdfTeX ≥ 1.40: tracking/letterspacing, additional kerning, and adjustment of interword spacing (glue) (new commands \SetTracking, \SetExtraKerning, \SetExtraSpacing; new options 'tracking', 'kerning', 'spacing') [5.3, 5.4, 5.5]
- New commands \textls and \lsstyle for letterspacing, new option 'letterspace' [3.4, 7]
- New option 'babel' for automatic micro-typographic adjustment to the selected language [3.5, 6]
- New font sets: 'smallcaps', 'footnotesize', 'scriptsize' [4, table 2]
- New package 'letterspace' providing the commands for robust and hyphenatable letterspacing [7]

1.9e (2006/07/28)

- New key 'inputenc' to specify the lists' input encodings [5]
- Protrusion settings for Euler math fonts

1.9d (2006/05/05)

- Support for the Central European QX encoding (protrusion, inheritance)
- Protrusion settings for various Euro symbol fonts (Adobe, ITC, marvosym)
- Support for Unicode input in the configuration (inputenc/utf8)

1.9c (2006/02/02)

- Protrusion settings for URW Garamond

1.9a (2005/12/05)

- Defer setup until the end of the preamble
- Inside the preamble, \microtypesetup accepts all package options [3.6]
- Protrusion settings for T5 encoded Charter

1.9 (2005/10/28)

- New command \microtypecontext to change the configuration context; new key 'context' for the configuration commands [6]
- New command \DisableLigatures to disable ligatures (pdfTeX ≥ 1.30) [8]
- New key 'font' to add single fonts to the font sets [4]
- New key 'preset' to set all characters to the specified value before loading the lists
- Value 'relative' renamed to 'character' for 'unit' keys
- Support for the Polish OT4 encoding (protrusion, expansion, inheritance)

- Support for the Vietnamese T5 encoding (protrusion, expansion, inheritance)

1.8 (2005/06/23)

- New option 'config' to load a different configuration file [3.5]
- New command \DeclareMicrotypeSetDefault to declare the default font sets [4]
- New option 'unit' to measure protrusion factors relative to a dimension instead of the character width [5.1]
- Renamed commands from \..MicroType.. to \..Microtype..
- Protrusion settings for AMS math fonts
- The 'allmath' font set also includes U encoding
- Support for protrusion with the ledmac package (pdfTeX ≥ 1.30)

1.7 (2005/03/23)

- Possibility to specify ranges of font sizes in the set declarations [4, 5]
- New command \LoadMicrotypeFile to load a configuration file manually [5.7]
- New command \Microtype@Hook for font package authors [Implementation, 1.4.4]
- New option 'verbose=errors' to turn all warnings into errors
- Warning when running in disable mode

1.6 (2005/01/24)

- When pdfTeX is too old to expand fonts automatically, expansion has to be enabled explicitly, automatic expansion will be disabled [3.1]
- New option 'factor' to influence protrusion resp. expansion of all characters of a font or font set [3.2, 5]
- Use e-TeX extensions, if available

1.5 (2004/12/15)

- When output mode is DVI, font expansion has to be enabled explicitly, automatic expansion will be disabled [3.1]
- New option 'selected' to enable selected expansion, default: false [3.3, 5.2]
- New default for expansion option 'step': 4 (min(stretch,shrink)/5) [3.3]
- Protrusion settings for Bitstream Charter

1.4 (2004/11/12)

- Set up fonts independently from LaTeX font loading

1.2 (2004/10/03)

- New font sets: 'allmath' and 'basicmath' [4, table 2]
- Protrusion settings for Computer Modern Roman math symbols

1.1 (2004/09/21)

- New command: \DeclareCharacterInheritance [5.6]
- Characters may also be specified as octal or hexadecimal numbers [5]
- Protrusion settings for Adobe Minion

1.0 (2004/09/11)

- First CTAN release

15 Index