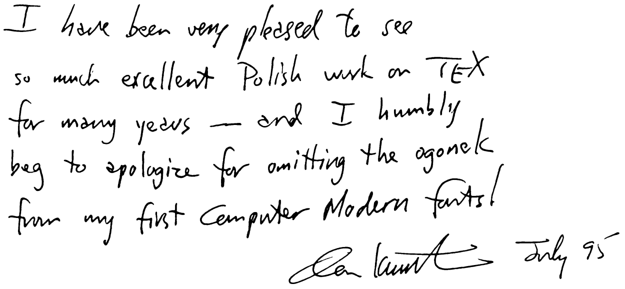
GUST e-foundry font projects

Bogusław Jackowski, Piotr Strzelczyk and Piotr Pianowski

What is a document? It is a sequence of rectangles containing a collection of graphic elements.
What is a font? It is a sequence of rectangles containing a collection of graphic elements.
— Marek Ryćko

1 Introduction

The Polish TEX Users Group (GUST) has paid attention to the issue of the fonts since the beginning of its existence. In a way, it was a must, because the repertoire of the diacritical characters of the *Computer Modern* family of fonts (CM), "canonical" TEX family defined as the Metafont programs (see [5]), turned out to be insufficient for the Polish language. The efforts of the GUST font team (GUST e-foundry), led by Bogusław Jackowski, were kindly acknowledged by the professor Donald E. Knuth:



I have been very pleased to see
so much excellent Polish work on TEX
for many years — and I humbly
beg to apologize for omitting the ogonek
from my first Computer Modern fonts!
Don Knuth July 95

Obviously, our first fonts were PK bitmap fonts programmed using Metafont. Alas, the TEX/Metafont bitmap font format never became the world-wide standard. Therefore, the next step were fonts in the PostScript Type 1 format which fairly soon became obsolescent and was replaced by the OpenType format (OTF, a joint enterprise of Microsoft and Adobe, 1996, see [31]) which is actually a common container for the Adobe PostScript Type 1 and Microsoft TrueType (TTF) formats. In 2007, Microsoft extended the OTF standard with the capability of typesetting math formulas, largely based on ideas developed for TEX, and implemented it in MS Office. Soon, TEX engines were adapted to process such math OTF fonts. Therefore our recent fonts are released in the OpenType format, which also makes them easily usable outside of the TEX realm.

So far, no OTF successor is in sight, which is both good and bad (cf. Section 7.4).

We published our partial results successively as the work progressed. This paper provides an overall summary of our work: it describes the collections of fonts prepared by the GUST e-foundry, deals with some technical issues related to the generation of fonts and their structure and puts forward a few proposals concerning future works.

This is not an overly strict report, but rather a story about our technical work on fonts, illustrated by representative examples which, we hope, show the essence of the matter. In order to keep our narration smooth, we decided not to use formal captions with explanations to figures and tables (only numbers of figures and tables are given). The relevant detailed descriptions always appear in the main text.

2 Historical background

PostScript and TEX are genetically related: their common ancestor is the ingenious idea of a programming language for the description of documents understood as a sequence of rectangular pages filled with letters and graphics. Both projects were devised nearly at the same time — at the turn of the 1970s to the 1980s.[1] And both are still alive and well, proving that the idea behind both projects was indeed brilliant.

From our perspective, the most important thing in common, and at the same time a key distinctive element, was the different handling of fonts and graphics; in other words, both systems clearly distinguished illustrations from fonts. That approach was justified by the computer technology at that time.

For both TEX and PostScript, fonts were external entities, both used metric files plus files defining glyph shapes, both defined contours as Bézier splines (planar polynomials of the 3rd degree), and for both fonts were to be prepared separately with dedicated font programs, prior to creating documents.

And this exhausts the list of similarities.

TEX worked with binary metric files, TFM; its output, a device independent file (DVI), was processed by so-called drivers which made use of the "proper" fonts, that is, the relevant bitmap collections, and produced output that could be sent to a printer or to a screen. The bitmaps were prepared independently with the Metafont program(s) which interpreted scripts written in the Metafont language and generated TFM and bitmap files.

In Metafont, the shapes of glyphs are defined as Bézier curves, stroked with a "pen" and/or filled.

Basic PostScript fonts (i.e., Type 1; see [28]) employ contours defined as non-intersecting closed Bézier outlines which can only be filled.[2] The "filled

[1] Formally, TEX was released a little earlier — TEX in 1982, PostScript in 1984, both with earlier work.

[2] The PostScript Type 1 documentation [28], p. 34, mentions the possibility of stroking: *a Type 1 font program can also be stroked along its outline when the user changes the PaintType entry in the font dictionary to 2. In this case,*

outline" paradigm relates also to the Microsoft TTF format and, thereby, to the OTF format.

PostScript Type 1 fonts are usually (but not necessarily) accompanied by corresponding ASCII metric files (AFM), not used by the interpreters of the PostScript language. In the Microsoft Windows operating system, making an already complex situation even more complex, binary printer font metric files (PFM) were introduced for Windows drivers that used PostScript Type 1 fonts.

For a long time, only commercial programs for generating PostScript Type 1 fonts were available. Only in 2001, George Williams released his remarkable FontForge program (initially dubbed PfaEdit; [25]). FontForge can generate outline fonts in many formats, including PostScript Type 1 and OTF.

PostScript was promptly (and rightly) hailed as *the* standard for printers and, more importantly, for phototypesetters, therefore a driver converting DVI files to PostScript became necessary. Fortunately for TEXies, PostScript is equipped also with Type 3 fonts; glyphs in Type 3 fonts can be represented by nearly arbitrary graphic objects, in particular, by bitmaps, therefore the making of a PostScript driver for converting DVI files to PostScript was possible already in 1986, when Dvips, the first and still most popular driver was released by Tomas Rokicki. (It's a pity that the idea of Type 3 fonts was not supported and developed by Adobe.)

There were a few unsuccessful attempts to convert the basic TEX font collection, CM, to the PostScript Type 1 format automatically, thus preserving the parameterization. The main hindrance was the excessive usage of stroked (both painted and erased) elements in the CM font programs, while, as was mentioned previously, the PostScript Type 1 and OTF formats accept only filled shapes.

The "filled outline" paradigm was a convenient optimization at the beginning of the computer typesetting era, when, for example, the generating of the complete collection of bitmaps for the CM fonts at resolution, say, 240 dots per inch (typical for dot matrix printers) took a few days. Nowadays, the paradigm still thrives by virtue of tradition: there is an abundance of such fonts and, and what is worse, all operating systems support only this kind of font.

3 First steps

Taking the above into account, we made up our minds to design our own programmable system for generating fonts in "world-compatible" formats.

3.1 Our tools

Our primary tool was MetaPost [4], a successor of Metafont, which promised well as a tool for making PostScript Type 1 fonts due to its native PostScript output. We called our MetaPost-based package MetaType 1 [13]. It was instantiated as a set of scripts using, besides MetaPost, T1utils, that is, Lee Hetherington's (dis)assembler for PostScript Type 1 fonts (cf. [3, 4]). A few scripts written in Gawk and Perl were also employed.

On the one hand, such a simple approach turned out to be insufficient for generating OTF fonts, in particular OTF math fonts. On the other hand, it turned out to be flexible enough to include an extra external step for making OTF fonts. For text fonts, we employed the *Adobe Font Development Kit for OpenType* (AFDKO [26]); for math fonts, the FontForge library governed by Python scripts [15, 25]. In the future, we want replace AFDKO by a FontForge+Python utility (cf. Section 7).

A set of MetaPost macros in the MetaType 1 package defines two important procedures, essential for generating non-intersecting outlines and heavily used in our font programs: finding a common outline for overlapping figures, known also as removing overlaps, and finding the outline of a pen stroke, known as expanding strokes or finding the pen envelope.

Another important feature, hinting, is implemented, but, in the end, we decided to avoid manual hinting, since it is difficult to control and yields mediocre results. Metafont has no notion of hinting — the Metafont language simply offers rounding. Moreover, the language for describing outlines in PostScript Type 1 fonts cannot express even as trivial a mathematical operation as rounding.

For low-resolution devices, controlled rounding is crucial — hence the idea of "hinting", that is, controlled rounding. Alas, hinting algorithms remain undisclosed, especially with regard to commercial typesetting devices such as phototypesetters. One can presume, however, that low-resolution devices are bound to disappear sooner rather than later: the resolution of display devices has reached almost 600 dpi and 1200 dpi (and more) for printers is nowadays nothing special. Therefore, running with the hare and hunting with the hounds, we decided to hint our recently released OTF fonts automatically with FontForge.

3.2 Trying our tools out

We tested our newborn MetaType 1 engine against a simple example, namely, Donald E. Knuth's `logo`

overlapping subpaths will be visible in the output; this yields undesirable visual results in outlined characters. In practice, this possibility is not used.

font [17]: the sources, originally written in the Metafont language, were adapted to MetaType 1's requirements. The distributed package contains both MetaType 1 sources and the resulting PostScript Type 1 files for the `logo` font [13].

The test proved the usefulness of the approach; hence, in 1999, we started a larger project: the programming of the long-established Polish typeface *Antykwa Półtawskiego* as a parameterized font. The preliminary family of fonts was released in 2000. Ten years later, prompted by the TEX community, we released the enhanced version of *Antykwa Półtawskiego* with the relevant OTF files [7]. In parallel, Janusz M. Nowacki used MetaType 1 to generate several replicas of Polish fonts, namely, *Antykwa Toruńska*, *Kurier*, *Iwona*, and *Cyklop* [22].

4 Latin Modern collection of fonts

Recall that the repertoire of diacritical characters in CM fonts was insufficient for most languages using diacritical characters. The TEX accenting mechanism (the `\accent` primitive), meant as a solution to this problem, was unsatisfactory — for example, accents and hyphenation conflicted. The problem was recognized relatively early and various approaches were used to remedy the situation.

For example, the Polish extension of CM in the PK format was prepared in Poland (PL fonts, [8]), but this worked only for Polish TEXies.

Also worthy of note is the *European Computer Modern Fonts* (EC) project led by Jörg Knappen and Norbert Schwarz, triggered during the TEX Users Group conference in Cork, 1990, and finished in 1997 [16]. The EC metric files, however, are slightly incompatible with CM metric files, by circa 0.025%.[3]

A rather general technique was applied by Lars Engebretsen, who attempted to eliminate the necessity of using the `\accent` primitive by making virtual fonts, dubbed *Almost European* (AE), containing quite a large set of the European diacritical characters [1].

Hyphenation worked with AE fonts, though still unsatisfactorily — for example, coinciding of such accents as cedilla or ogonek with a main glyph is inconvenient when a non-intersecting outline is required (for cutting plotters, for outlined titles, and so on). Moreover, the virtual fonts are obviously unusable outside the TEX realm.

4.1 Latin Modern fonts in the PostScript Type1 format

In 2002, during the EuroBachoTEX meeting, a proposal of converting Engebretsen's AE fonts into the PostScript Type 1 format and augmenting with them the set of necessary diacritical characters was put forward by representatives of European TEX user groups. We had no choice but to accept the proposal with delight.

Our initial plan was to use the AE fonts as our departure point; we even wanted to preserve the original name, *Almost European*, coined by Lars Engebretsen. It turned out, however, to be much more efficient to prepare the enhanced version of the CM fonts from scratch, and so sticking to Lars Engebretsen's name seemed inadequate, because the differences were too essential.

All in all, inspired by both EC and AE fonts, we came up with the *Latin Modern* (LM; see [11]) project which was accepted by the user groups.

Fortunately for us, freely available quality CM fonts in the PostScript Type 1 format already existed. In the 1980s and 90s, they were produced (from traced bitmaps improved by very solicitous manual tuning) for commercial purposes by Blue Sky Research and Y&Y. Nearly a decade later, they were released to the public thanks to the efforts of the American Mathematical Society.[4]

We converted the PostScript Type 1 files of the CM fonts to MetaType 1 and wrote the MetaPost software relevant for generating the characters we decided to add (mainly diacritical letters). The work had already been partially done by Janusz M. Nowacki, who prepared the PostScript Type 1 version of the PL fonts in 1997. The official version of the LM fonts, 1.000, was eventually released in 2006 (in the meantime, several unofficial versions were released for testing purposes). The LM collection of fonts consisted of 72 text fonts, each counting about 700 glyphs, plus 20 CM-like math fonts.

In 2009, an extensive revision of the LM fonts was carried out: the text fonts now contain more than 800 glyphs each (altogether more than 60,000 glyphs) and the glyphs conform to the changes introduced by Donald E. Knuth in 1992.

[3] The reason behind this discrepancy is a peculiarity of Metafont arithmetic: the formula $1/36 * i$ yields different results than the formulas $i/36$ and $1/36i$; for example, for $i = 3600$ the results are 99.97559 for the first formula and 100 for the latter formulas. The first formula was used in the EC fonts (in the `gendef` macro).

[4] In 1997, a consortium of scientific publishers (American Mathematical Society, Elsevier Science, IBM Corporation, Society for Industrial and Applied Mathematics, and Springer-Verlag) in cooperation with Blue Sky Research and Y&Y decided to release these excellent fonts non-commercially; in order to assure the authenticity of the fonts, copyright was assigned to the American Mathematical Society. (`http://www.ams.org/publications/type1-fonts`).

Almost all of the CM text fonts have counterparts in the LM family; the exceptions are one monospaced font, `cmtex10`, emulating Donald E. Knuth's keyboard layout, and the rarely used `cmff10`, `cmfi10`, `cmfib8`, and `cminch`. So far, nobody has complained about this inconsistency. Instead, encouraged by Hans Hagen, we decided to create 10 variants of typewriter LM fonts not having counterparts in the CM family: `lmtlc10`, `lmtk10`, `lmtl10`, `lmvtk10`, and `lmvtl10` (monospaced light condensed and monospaced and variable-width dark and light, respectively) plus their oblique variants `lmtko10`, `lmtlo10`, `lmtlco10`, `lmvtko10`, and `lmvtlo10`.

4.2 Latin Modern fonts in the OTF format

It was relatively easy to prepare the LM family of fonts in the OTF format using the AFDKO package: it mainly necessitated preparing a few extra data files in the *OpenType Feature File Specification* language [32]. Needless to say, the experience gathered at this stage came in handy during the work on the TG fonts (see Section 5).

There was trouble, however, with the 20 math fonts. We provided the respective LM equivalents in PostScript Type 1 format. For compatibility with the (obsolete) PL fonts, the symbol fonts, `lmsy*` and `lmbsy*`, contain two extra glyphs: slanted greater-or-equal and less-or-equal signs, used traditionally in Polish math typography. As the math extension for OpenType did not exist yet, we decided not to convert these fonts to OTF. We knew that the companies that had invented and maintained the OTF standard, in cooperation with the American Mathematical Society and the Unicode Consortium, were working on extending the standard with math typesetting capabilities. We expected that by using the enhanced OTF specification we would be able to create a TEX-compatible math OTF collection. Alas, the Unicode Consortium report on Unicode support for mathematics [37], followed by the initially confidential Microsoft specification [29], snuffed out our hopes. It turned out that OTF math and TEX math cannot be reconciled. More information on the interrelationships between OTF and TEX math can be found in Ulrik Vieth's thorough analysis [24].

4.3 Repertoire issues

Our primary aim was to provide a repertoire of diacritical letters rich enough to cover all European languages. We thoroughly exploited Michael Everson's comprehensive study of European alphabets [2], as well as other sources. Several other languages using Latin-based alphabets, such as Vietnamese, Navajo and Pali, are covered.

Initially, contrary to the *Latin Modern* name, we considered including Cyrillic alphabets also. Having thought the matter over, we decided, with regret, to abandon this idea and concentrated our efforts on OTF math fonts.

Besides diacritical characters, the Latin Modern fonts contain also a number of glyphs traditionally present in TEX fonts, such as Greek symbols, currency symbols, technical symbols, etc. Detailed description of the contents of the fonts can be found in the document entitled *The Latin Modern Family of Fonts. Technical Documentation*, included in the LM distribution package [11].

Two groups of glyphs are widely used in typography but neglected to a certain extent in the CM fonts, namely, *caps and small caps* and *old style numerals*, also known as *text figures* or *nautical digits*; the latter name originates from their widespread use in tables in nautical almanacs at one time. For reasons hard to explain, the caps and small caps were implemented in the CM family as a separate font, while the old style numerals (upright!) are in the math italic font (`cmmi*`).

The LM fonts incorporated caps and small caps from the CM family, together with its width idiosyncrasy: the `lmcsc10` font, like `cmcsc10`, has capital letters wider than the `lmr10` font by circa 8%. There are two caps and small caps fonts in the LM collection, namely, the regular and typewriter specimen, `lmcsc10` and `lmtcsc10`, as in CM, plus their oblique variants, `lmcsco10` and `lmtcsco10`, absent from CM. In principle, small caps glyphs could be transferred to other fonts, but we decided to not alter the framing of the original CM family, more so as CM has no sans-serif caps and small caps; however, the problem of extending the LM family with bold counterparts of `lmcsc10` and `lmtcsc10` (and their oblique variants), raised repeatedly by CM/LM users, needs serious consideration.

Concerning old style numerals, we could not accept the CM oddity and included them in all text fonts of the LM family. Further, all numerals come in 2 'flavors': normal (fixed-width a.k.a. tabular) and proportional (variable-width, having balanced sidebearings) which altogether yields 4 variants — see Figure 1.

The TFM format contains only 256 slots for glyphs, thus, the whole repertoire of glyphs cannot be accessed at once if TEX is used in a "traditional" way, that is, with TFM files. In particular, accessing the different kinds of numerals when using PostScript Type 1 fonts plus TFM metrics turns out to be clumsy; as a result, only tabular old style numerals are available in our package, in the TS1 encoding

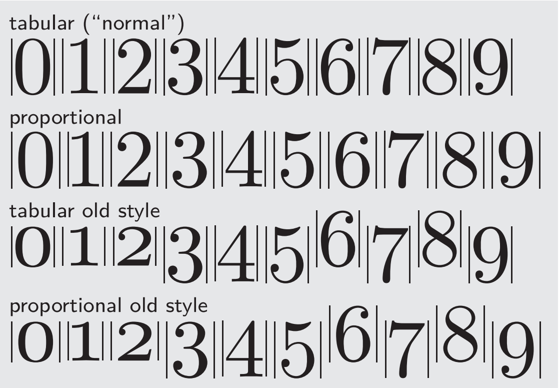


Figure 1

(see below). On the one hand, OTF fonts seem more convenient as they do not impose such a restriction; for example, all numerals can be accessed by using the OTF feature mechanism, more precisely, by the features `onum`, `lnum`, `pnum`, and `tnum` [33]. On the other hand, OTF metric data cannot, in general, be fully compatible with TFM metric data because the glyph widths in OTF fonts must be represented by integer quantities. This should be considered a drawback by TeXies — see Section 4.4.

Following the LaTeX tradition, we provided several encodings for the LM fonts, namely:

- ◇ CS (CS TUG) encoding (`cs-*.tfm`),
- ◇ EC (Cork) encoding (`ec-*.tfm`),
- ◇ L7X (Lithuanian) encoding (`l7x-*.tfm`),
- ◇ QX (GUST) encoding (`qx-*.tfm`),
- ◇ RM (“regular math”, used in OT1 and OT4) encodings (`rm-*.tfm`),
- ◇ Y&Y’s TeX’n’ANSI a.k.a. LY1 encoding (`texnansi-*.tfm`),
- ◇ T5 (Vietnamese) encoding (`t5-*.tfm`),
- ◇ Text Companion for EC fonts a.k.a. TS1 (`ts1-*.tfm`).

The LaTeX support for all these encodings, due to Marcin Woliński, is also part of the LM distribution.

TFM files nominally representing the same encoding do not always define the same set of characters; for example, the character sets of `cmr10.tfm` and `cmtt10.tfm` differ. The original CM fonts comprise 7 different character sets, with an idiosyncratic difference between the `cmr10` and `cmr5` layouts. As a remnant of the CM design, there are 5 different character sets of the LM text fonts:

1. 821 glyphs (basic set): `lmb10 lmbo10 lmbx10 lmbx12 lmbx5 lmbx6 lmbx7 lmbx8 lmbx9 lmbxi10 lmbxo10 lmdunh10 lmduno10 lmr10 lmr12 lmr17 lmr5 lmr6 lmr7 lmr8 lmr9 lmri10 lmri12 lmri7 lmri8 lmri9 lmro10 lmro12 lmro17 lmro8 lmro9 lmss10 lmss12 lmss17 lmss8 lmss9 lmssbo10 lmssbx10 lmssdc10 lmssdo10 lmsso10 lmsso12 lmsso17 lmsso8 lmsso9 lmu10 lmvtk10 lmvtko10 lmvtl10 lmvtlo10 lmvtt10 lmvtto10`
2. 824 glyphs: `lmssq8 lmssqbo8 lmssqbx8 lmssqo8`
   extra characters: `varI varIJ varIogonek`
3. 814 glyphs: `lmcsc10 lmcsco10`
   missing characters: `f_k ff ffi ffl fi fl longs`
4. 785 glyphs: `lmtk10 lmtko10 lmtl10 lmtlc10 lmtlco10 lmtlo10 lmtt10 lmtt12 lmtt8 lmtt9 lmtti10 lmtto10`
   missing characters: `f_k ff ffi ffl fi fl Germandbls hyphen.prop IJ ij permyriad servicemark suppress trademark varcopyright varregistered zero.oldstyle zero.prop one.oldstyle one.prop two.oldstyle two.prop three.oldstyle three.prop four.oldstyle four.prop five.oldstyle five.prop six.oldstyle six.prop seven.oldstyle seven.prop eight.oldstyle eight.prop nine.oldstyle nine.prop`
5. 784 glyphs: `lmtcsc10 lmtcso10`
   missing characters: as in 4, also `longs`

4.4 Compatibility issues

We did our best to provide outline fonts that can be used as a replacement for CM fonts. To a certain extent, we managed to achieve this goal, namely, the PostScript drivers which process TeX documents typeset with CM metric files, can use either CM or LM PostScript Type 1 fonts — special map files for PostScript Type 1 fonts are available for this purpose. The metric files, however, cannot be used replaceably, because the typesetting algorithms are intrinsically unstable — even tiny (rounding) errors may yield glaringly different results.

Therefore, LM users also cannot expect PostScript Type 1 and OTF fonts to be used replaceably. Recall that the OTF format requires integer number representation for glyph widths. The “reference” quantity is the *em* unit: $1\,em = 2048$ units for fonts using splines of the 2$^{\text{nd}}$ degree, $1\,em = 1000$ units for fonts using splines of the 3$^{\text{rd}}$ degree (e.g., our LM and TG fonts). Therefore, in our case, the difference in width is on average 1/2000 *em* (twice as large as the variation in the EC widths), that is, circa $0.005\,pt$ for 10-point fonts.

Because the MetaType 1 sources of the LM fonts are the result of conversion from PostScript Type 1, the widths stored in the LM TFM files are not identical to the respective original CM widths. They are closer, however, to the original quantities by an order of magnitude compared to the EC and OTF widths. At the cost of great effort (by referring to the Metafont sources), we might have eliminated rounding errors in LM widths. But it would not cure the problem of (non-)replaceability, as widths are not the only source of trouble. Differences in heights and depths of glyphs may also yield unexpected behavior of the TeX typesetting algorithm.

The problem of heights and depths in TeX turns out to be unavoidable, and quite serious: the TFM format permits by design only 16 different heights and depths, including the obligatory entries containing the value zero. If there are in fact more heights and depths in a given font, their number is cleverly reduced to 16 by Metafont (as well as by the MetaPost and TFtoPL programs). One of the certainly unwanted results is that the same glyph in different encodings may have different heights and/or depths! For example, the height of the letter ‘A’ is 6.88875 pt in the `rm-lmr10.tfm` file (this layout is an extension to 256 slots of the `cmr10` layout), it is 6.99648 pt in the `t5-lmr10.tfm` file (Vietnamese layout), while in the canonical `cmr10.tfm` file it is 6.83331 pt.

This is not the end of the list of possible sources of incompatibility between CM and LM fonts. Positioning of the accents is also a long story. These and related aspects are explained minutely in [12].

Finally, let us consider a somewhat atypical example of incompatibility between the LM and CM fonts related to Donald E. Knuth’s mistake in a CM `ligtable` program, uncorrectable for obvious reasons but basically harmless; namely, `roman.mf` contains the following:

```
% three degrees of kerning
k#:=-.5u#; kk#:=-1.5u#; kkk#:=-2u#;
ligtable "k":
  if serifs: "v": "a" kern -u#, fi
  "w": "e" kern k#, "a" kern k#,
       "o" kern k#, "c" kern k#;
```

The culprit is the `if serifs` clause: the kern pair ‘ka’ appears twice in the TFM files of serif fonts with the values $-u\#$ and $-0.5u\#$, respectively, as is easily seen in the following fragment of the `cmr10.pl` file (the respective lines are marked with arrows):

```
(CHARACTER C k
   (CHARWD R 0.527781)
   (CHARHT R 0.694445)
   (COMMENT
      (KRN C a R -0.055555)⇐
      (KRN C e R -0.027779)
      (KRN C a R -.027779) ⇐
      (KRN C o R -0.027779)
      (KRN C c R -0.027779)
      )
   )
```

Moreover, there are no ‘va, ‘vc’, ‘ve’, and ‘vo’ kern pairs in sans-serif fonts, although there are ‘kc’, ‘ka’, ‘ke’, ‘ko’, ‘wa’, ‘wc’, ‘we’, and ‘wo’ kern pairs in these fonts. We could not see the reason for ignoring ‘v’ in this context, thus we decided to add the relevant kern pairs in the LM fonts; we also added quite a few other kern pairs missing, in our opinion, from the CM fonts, for example, ‘eV’ and ‘kV’.

Summing up, we believed that we had good reasons for giving up the struggle for a “100-percent compatibility” between LM and CM metrics, whatever that would mean, and to confine ourselves to providing the mentioned replaceability of outlines.

5 The TeX Gyre collection of fonts

Heartened by the results of the LM enterprise, we accepted without hesitation the next proposal: the “LMization” of the family of fonts provided by Ghostscript as a replacement for the renowned *Adobe base 35* fonts, generously released by the URW++ company under free software licenses.

- ◇ ITC Avant Garde Gothic (book, book oblique, demi, demi oblique)
- ◇ ITC Bookman (light, light italic, demi, demi italic)
- ◇ Courier (regular, regular oblique, bold, bold oblique)
- ◇ Helvetica (medium, medium oblique, bold, bold oblique)
- ◇ Helvetica Condensed (medium, medium oblique, bold, bold oblique)
- ◇ New Century Schoolbook (roman, roman italic, bold, bold italic)
- ◇ Palatino (regular, regular italic, bold, bold italic)
- ◇ Symbol
- ◇ Times (regular, regular italic, bold, bold italic)
- ◇ ITC Zapf Chancery (medium italic)
- ◇ ITC Zapf Dingbats

Since our aim was “LMization”, we excluded the Symbol and ITC Zapf Dingbats non-text fonts from the scope of our interest.

After a brief (but heated) debate, the name of the project and of its constituent fonts were coined. The project was dubbed TeX Gyre (TG) and the following names were accepted (the respective file name kernels, original Adobe names and Ghostscript, that is, URW, names are given in parentheses):

- ◇ TG Adventor (`qag` / ITC Avant Garde Gothic / URW Gothic L)
- ◇ TG Gyre Bonum (`qbk` / ITC Bookman / URW Bookman L)
- ◇ TG Cursor (`qcr` / Courier / Nimbus Mono L)
- ◇ TG Heros (`qhv` / Helvetica / Nimbus Sans L)
- ◇ TG Heros Condensed (`qhvc` / Helvetica Condensed / Nimbus Sans L Condensed)
- ◇ TG Schola (`qcs` / New Century Schoolbook / Century Schoolbook L)
- ◇ TG Pagella (`qpl` / Palatino / URW Palladio L)
- ◇ TG Termes (`qtm` / Times / Nimbus Roman No9 L)
- ◇ TG Chorus (`qzc` / ITC Zapf Chancery / URW Chancery L)

We initially considered including Cyrillic alphabets, but, as in the case of the LM fonts, we eventually abandoned this idea, also with regret, the more so as the Ghostscript fonts at that time contained an (apparently unfinished) set of Cyrillic glyphs.

We expected that the main effort would be the making of extra glyphs plus maybe correcting outlines here and there. To our surprise, quite a few glyphs required tuning because of evident errors in outlines. One of the most striking examples is the glyph 'eight' from the URW Schoolbook bold font[5] — see Figure 2.

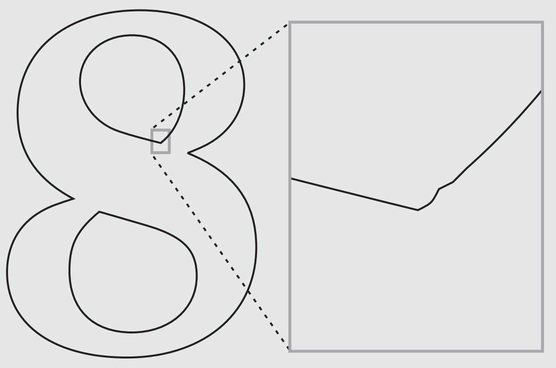


Figure 2

Mostly, we removed redundant or wrong nodes (points) from the outline definitions, deleting more than 5% of them in all. In the case of TG Pagella, however, the insertion of extra nodes turned out necessary. The chart in Figure 3 shows some statistics for the upright TG fonts. The diagram concerns the version of the Ghostscript fonts which we used as our starting point. We used circa 350 glyphs from each font. The total number of nodes in these glyphs varied from circa 10,000 to 25,000 per font. In the current release, the TG text fonts count almost 1100 glyphs each with the number of nodes varying from circa 30,000 to 65,000 (for sans-serif and serif italic variants, respectively; see [14]).

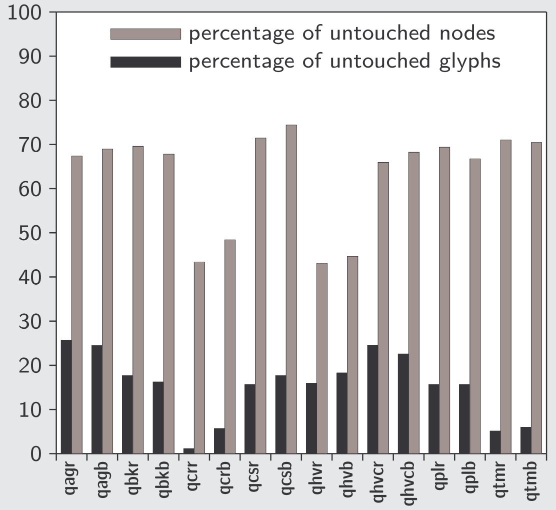


Figure 3

5.1 Repertoire issues

The difference in the number of glyphs between the TG and LM text fonts (the former having circa 250 glyphs more per font) is due mainly to the presence of small caps in the TG fonts (225 glyphs per font). Also unlike the LM fonts, each TG font contains the complete Greek alphabet and a few technical glyphs, such as 'lozenge' and 'lscript'.[6] Except for these, the LM and TG fonts share the same repertoire of glyphs and the same set of TFM encodings (see Section 4.3).

The LaTeX support for these encodings was also provided by Marcin Woliński.

5.2 Compatibility issues

The consistency of the widths of the original Adobe and the respective TG glyphs was one of our main concerns, as the TG fonts were meant as potential replacements for the Adobe fonts. It turned out, however, that the original font metric files [27], contained apparent metric flaws which we decided, with some hesitation, not to retain.

A typical example (concerning Helvetica, a.k.a. Nimbus Sans L, a.k.a. TG Heros) is depicted in Figure 4. Both Spanish '¡' and Scandinavian 'ø' glyphs belong to the Adobe Standard Encoding set, hence

[5] Recently, the font has been renamed to 'C059 bold'; the bug was removed from the Ghostscript distribution only in 2015, although the TeX Collection 2016 distribution still contains (due to the legacy reasons) the faulty glyph.

[6] For historical reasons, the LM fonts contain a few additional variants of the base, left, and right double quotes, absent from the TG fonts.

one can expect that they should be considered important and thus unchangeable. Nevertheless, we could not see a reason for using widths different from the '!' and 'o' widths, respectively, and certainly there is no substantiation for the asymmetry of sidebearings, especially in the case of 'ø'; therefore, we decided to alter the metrics.

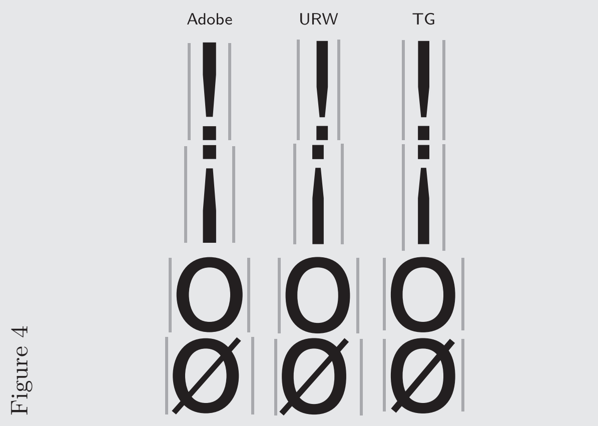


Figure 4

Fortunately, there are few such cases in the TG collection; each is mentioned in the documentation of the TG fonts [14].

Because the original widths for the TG fonts, unlike in the LM collection, were integer numbers, there is no metric discrepancy between the PostScript Type 1 and OTF font formats, as far as widths are concerned. Heights and depths, however, are subject to the same restrictions as discussed in Section 4.4.

6 OTF math fonts

The chronic problem of lack of math support for the TG collection became the impetus for our third venture: math fonts for the LM and TG collections in the OTF format. The recent update of the LM and TG fonts took place at the end of 2009. The math extension for the OTF format ([29]) had been released and there existed the FontForge font editor ([25]) capable of generating such fonts. So we embarked upon an expedition into unknown regions — since then we have focused our attention on the work on OTF math fonts, again with the benevolent encouragement and support from the TEX users groups.

As we should have expected, the task turned out interesting and absorbing, and, according to Hofstadter's Law,[7] we spent more time on it than we expected. From the very beginning, we aimed at making a collection of mutually consistent math OTF fonts and we underestimated the heterogeneity of the sources of additional alphabets and the problem of interrelationships — works on subsequent fonts entailed moving backwards to the fonts which we had prematurely considered ready. Nevertheless, in 2011, we happily announced the release of our first math OTF font, namely, Latin Modern Math. Altogether, six math fonts have been released by the GUST e-foundry so far [9, 10]:

◇ TG Latin Modern Math
◇ TG Bonum Math
◇ TG Schola Math
◇ TG Pagella Math
◇ TG Termes Math
◇ TG DejaVu Math

This amounts to nearly half of all OTF math fonts released in the world. Besides these, the following OTF math fonts have been released: Asana by Apostolos Syropoulos, Neo-Euler and XITS by Khaled Hosny, STIX by the STI Pub companies,[8] Cambria Math by Microsoft,[9] Lucida Math by Bigelow & Holmes, and Minion Math by Johannes Küster; the latter three fonts are distributed commercially.

6.1 OTF math font contents

Math OTF fonts, as we expounded in [10], are truly nasty beasts. In accordance with [35] and [37], they are expected to contain a plethora of glyphs: letters, arrows, math operators and delimiters, geometrical shapes, technical symbols, etc. The presence of some of them, particularly the (over)abundance of peculiar geometrical shapes and arrows, is hard to substantiate in our opinion.

Initially, we planned also releasing the math companion to the TEX Gyre sans-serif fonts, TG Adventor and TG Heros, but the Unicode specification for the contents of math fonts, [37], turned out definitely "serif-oriented". Let us take, for example, the arrangement of the LM Math font shown in Table 1: following the cited specification, we combined several LM source fonts into a single complex font. As the table clearly shows, the basic subsets, that is, plain, bold, italic and bold italic are assumed to consist of serif glyphs by default. It is not obvious how

[7] Hofstadter's Law: *It always takes longer than you expect, even when you take into account Hofstadter's Law* — Douglas R. Hofstadter.

[8] The STIX project began through the joint efforts of American Mathematical Society (AMS), American Institute of Physics Publishing (AIP), American Physical Society (APS), American Chemical Society (ACS), Institute of Electrical and Electronic Engineers (IEEE), and Elsevier Science; these companies are collectively known as the STI Pub companies.

[9] Cambria Math was the first math font published, conforming to the specification *MATH — The mathematical typesetting table* [29]. It was released by Microsoft in 2007, along with a MS Office version equipped with the capability of handling the math font and editing math formulas.

the table should be adjusted to suit sans-serif math fonts. Work on this issue is in progress.

category	charset	source fonts
plain (upright, serif)	L*, G, D	`lmr`, `lmmi` (upright)
bold	L, G, D	`lmbx`, `lmmib` (upright)
italic	L, G	`lmmi`
bold italic	L, G	`lmmib`
sans-serif	L, D	`lmss`
sans-serif bold	L, G, D	`lmssbx`
sans-serif italic	L	`lmsso`
sans-serif bold italic	L, G	`lmssbo`
calligraphic	L	`eusm` (slanted)
bold calligraphic	L	`eusb` (slanted)
Fraktur	L	`eufm`
bold Fraktur	L	`eufb`
double-struck	L, D	`bbold` (by Alan Jeffrey)
monospace	L, D	`lmtt`

L, G, D — Latin, Greek and digits, respectively
L* — contains also diacritical letters and punctuation
All the alphanumeric glyphs specified in the table, except for the "plain" ones (first row), are given special mathematical Unicode slots — see [37]

Table 1

The original CM fonts, and thus the LM fonts, do not contain the complete sans-serif Greek. We generated the missing glyphs using modified Metafont sources in order to generate outlines instead of bitmaps and tuning the result manually, if required.

Moreover, a math font is bound to contain many other characters, most notably glyphs used for subscripts of the 1$^{\text{st}}$ and 2$^{\text{nd}}$ order, used also for superscripts; we'll refer to them shortly *pars pro toto* as subscripts. They are accessed by the OTF feature mechanism, more precisely by the math extension feature `ssty` [33].

Extensible symbols, like large brackets or radicals, are another important group of math-oriented glyphs. An extensible symbol consists of a collection of a few components (the left part of Figure 5) assembled by the typesetting engine into a seemingly single character (the right part of Figure 5).

First, a glyph of adequate size is searched for in the so-called chain of glyph variants (here: the three leftmost curly braces). If a proper glyph is not found, the typesetting engine assembles a respectively large symbol from the relevant pieces using a fairly complex algorithm — everybody who has attempted unsuccessfully to fit braces around a formula according to one's desire probably knows it.

In the case of the radical symbol, the situation is still more complex, because the OTF and TEX geometric structure, as well as the relevant metric data of the components differ, as is shown in Figure 6 which visualises "stages" of the assembling of an extensible radical symbol.

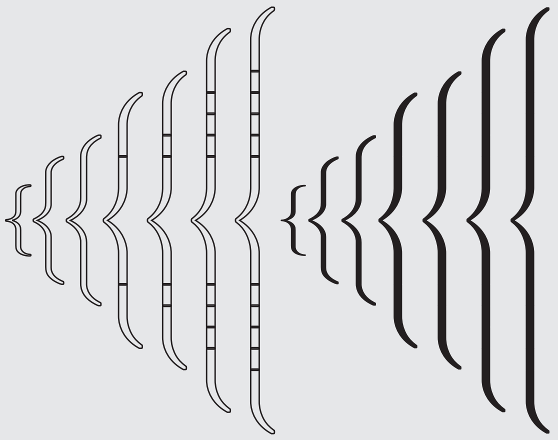


Figure 5

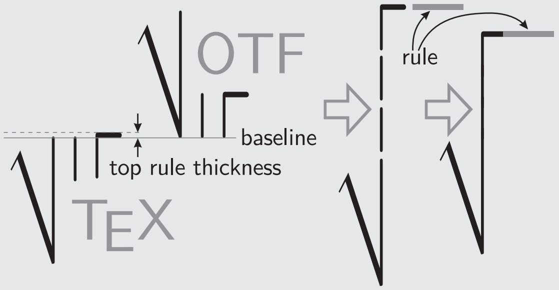


Figure 6

In both cases, the radical symbol is assembled from glyphs taken from a relevant font and a line (rule) drawn by the typesetting program at the top of the symbol (marked with a gray color). The thickness of the rule, however, is given explicitly as a parameter in math OTF files, while it is inferred from the height of the top element of the radical symbol in TEX (recall the problem of the limited number of heights in a TEX metric file).

There is an important difference between the TEX and OTF math font specification concerning extensible glyphs: TEX is equipped with the ability to assemble compound glyphs from pieces only vertically, while the OTF format offers both horizontal and vertical assembling. In TEX (i.e., in the `plain` TEX format), such glyphs as horizontal braces are defined by special macros using rules and a few glyphs from a relevant font:

```
\def\downbracefill
  {$\m@th \setbox\z@\hbox{$\braceld$}%
  \braceld\leaders\vrule height\ht\z@
     depth\z@\hfill\braceru
  \bracelu\leaders\vrule height\ht\z@
     depth\z@\hfill\bracerd$}
\def\upbracefill [...]
```

Incidentally, it seems that diagonal extensible glyphs have not yet been invented. Could it be that the conundrum is too difficult to solve?

More on the differences between the TEX and OTF specifications of the structure of math fonts can be found in Ulrik Vieth's survey [24].

In the case of LM Math, we were fortunate to have well-known clean sources for a base, already containing 7-point and 5-point variants, suitable for typesetting subscripts, and components for assembling extensible glyphs.

We have to confess, however, that we were not especially delighted with the Computer Modern calligraphic script. More pleasingly designed, to our eyes, are the calligraphic letters of the renowned Euler family. Therefore, we decided to transfer the glyphs from the Euler fonts (slanting them slightly) to the LM Math font:

$\mathcal{ABCPQT}\boldsymbol{\mathcal{ABCPQT}}$ Computer Modern
$\mathcal{ABCPQT}\boldsymbol{\mathcal{ABCPQT}}$ Euler
$\mathcal{ABCPQT}\boldsymbol{\mathcal{ABCPQT}}$ Latin Modern Math

In the case of the TG math fonts, the situation was slightly worse. The basic sources, that is, the text fonts, were already improved by us, but the sources of the relevant additional character sets were highly heterogeneous. We used freely available fonts of the best possible quality as our base; nevertheless, much manual tuning was necessary (recall the tuning of the sources of the TG text fonts).

Moreover, not all suitable fonts had acceptable free licenses. In a few cases, we had to contact the authors personally. It should be emphasized that in all cases the authors, if we managed to reach them, courteously agreed to make their fonts available for our purposes. The following external fonts were used in the TG math fonts project (in alphabetical order):

◇ Lato by Łukasz Dziedzic (TG Schola Math)
◇ Kerkis by Apostolos Syropoulos and Antonis Tsolomitis (TG Bonum Math)
◇ Leipziger Fraktur replica by Peter Wiegel (TG Bonum Math and TG Termes Math)
◇ Math Pazo by Diego Puga (TG Pagella Math)
◇ Odstemplik by Grzegorz Luk (gluk) (TG Pagella Math)
◇ Theano Modern by Alexey Kryukov (TG Schola Math)

We are most grateful to all the authors for their prominent aid.

6.2 Visual issues

Observe that the same monospaced alphanumeric characters (excerpted from TG Cursor) are shared by TG Bonum Math, TG Schola Math, and TG Termes Math. In such cases one must be carefully check whether the size of the glyphs being included fits the size of the basic set of glyphs. Nominally, all fonts (both in the PostScript Type 1 and OTF formats) have the same design size, 10 typographic points (recall that 1 point = 1/72 inch; cf. Section 4.4). Working on the TG math fonts, we had to adjust the size of the subsets a few times. For example, we enlarged the borrowed monospaced glyphs to circa 112.5% in TG Schola Math; otherwise the monospaced alphabet looked too small in combination with Schola's brawny glyphs.

The visual harmonizing of the supplementary alphabets with the main font face concerns not only alphanumeric glyphs. Even essentially geometrical shapes should also reflect the characteristic features of the main font, for example, the thickness of stems, the ending of arms, etc. Seemingly trivial glyphs, such as arrows, serve as a convenient example: they have slightly different shapes in each of our math fonts — see Figure 7.

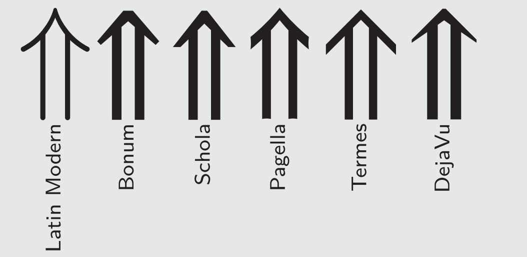


Figure 7

Another example is the shape of integrals. Historically, the integral symbol originates from the letter 'long s'[10] which is nowadays identical with a barless 'f'. Therefore, we did our best to preserve some characteristic features of the letter 'f' in the design of the integral shape — see Figure 8.

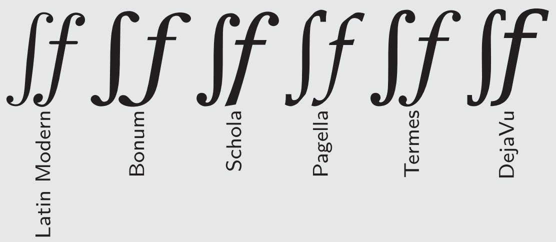


Figure 8

We are not going to dwell on the visual aspects of the math font design, as the number of details relevant for a font with over 4000 glyphs (and

[10] The cursive long 's' for denoting an integral operation, i.e., infinite summation, was introduced by Gottfried Wilhelm Leibniz in 1675 in an unpublished paper *Analyseos tetragonisticæ pars secunda* (*Second part of analytical quadrature*).

counting) would perhaps become rather overwhelming. Notwithstanding, one aspect needs emphasizing, namely, the problem of sidebearings and kerning for alphanumeric symbols.

It is generally accepted by typographers that math symbols should have larger sidebearings than the respective text glyphs. In particular, TEX math italic glyphs are a little bit broader and have larger sidebearings than the text italic glyphs. It is not obvious, however, how large such sidebearings should be. We have already experimented with a few sizes but further tuning will probably be necessary. Of course, the broadening of sidebearings should not be applied to the basic font, that is, regular upright, because of multiletter names of functions and operators, such as ‘sin’ or ‘max’, which are traditionally typeset with regular upright letters.

As regards the problem of kerning, we decided not to include kerns in math fonts, although providing kerns for the upright regular alphabet might be reasonable. Actually, we consider introducing special math kerning (so-called “staircase” kerns a.k.a. “cut-ins”; see Section 7). Thankfully, nobody has complained yet because of the lack of kerning in our math OTF fonts.

6.3 Repertoire issues

At present, a common standard for the repertoire of characters that should be present in an OTF math font does not exist. After several debates (mostly during TEX conferences) and many experiments, we came up with the tentative repertoire scheme presented in Table 2, which can be considered a detailed specification of Table 1.

Table 2

B – basic letters, A – accented letters, G – Greek letters, D – digits, O – other symbols, P – punctuation

	B	A	G	D	O	P
plain (upright, serif)	$+_{s}$	$+_{s}$	$+^{d}_{s}$	$+_{s}$	$+$	$+_{s}$
italic	$+_{s}$		$+_{s}$			
bold	$+_{s}$		$+^{d}_{s}$	$+_{s}$		
bold italic	$+_{s}$		$+_{s}$			
sans-serif	$+$			$+$		
sans-serif italic	$+$					
sans-serif bold	$+$		$+$	$+$		
sans-serif bold italic	$+$		$+$			
calligraphic	$+$					
bold calligraphic	$+$					
Fraktur	$+$					
bold Fraktur	$+$					
double-struck	$+$			$+$		
monospace	$+$			$+$		

d — digamma excluded from relevant Unicode blocks
s — subscripts are to be added

We would like all TG math fonts (Bonum, DejaVu, Pagella, Schola, and Termes) to share this pattern. For LM Math, however, we adopted another scheme because of legacy concerns. Currently, LM Math contains circa 4800 glyphs while the remaining fonts contain circa 4250 glyphs each. LM Math contains more subscripts, as the original sources already provided them. On the other hand, the TG math fonts contain more size variants of integral symbols. This yields circa 550 glyphs more (net) in LM Math. At the moment, TEX Gyre DejaVu Math contains (in accordance with Table 2) subscript variants for bold upright glyphs, while the remaining TG math fonts need to be complemented with these glyphs. It is a typical instance of the frequently occurring “backtracking” procedures: the final decision was undertaken only after a few fonts had been released.

Concerning subscripts, it should be emphasized that subscript glyphs for the TG math fonts are obtained using the approach applied in the Metafont sources of the Euler family of fonts, that is, by non-uniform rescaling of the respective normal-size glyphs. In a way, such “optical scaling” can be considered undesirable; nevertheless, it proved acceptable for the Euler fonts, thus we decided to apply it also to the TG math fonts.

Some rather quirky characters which received Unicode slots are spaces. Unicode [35] defines quite a few space-related glyphs:

```
 0008  BACKSPACE (<control>)
*0020  SPACE
*00A0  NO-BREAK SPACE
*2002  EN SPACE
*2003  EM SPACE
 1361  ETHIOPIC WORDSPACE
 1680  OGHAM SPACE MARK
*2004  THREE-PER-EM SPACE
*2005  FOUR-PER-EM SPACE
*2006  SIX-PER-EM SPACE
*2007  FIGURE SPACE
*2008  PUNCTUATION SPACE
*2009  THIN SPACE
*200A  HAIR SPACE
*200B  ZERO WIDTH SPACE
*202F  NARROW NO-BREAK SPACE
*205F  MEDIUM MATHEMATICAL SPACE
 2408  SYMBOL FOR BACKSPACE
 2420  SYMBOL FOR SPACE
 3000  IDEOGRAPHIC SPACE
 303F  IDEOGRAPHIC HALF FILL SPACE
*FEFF  ZERO WIDTH NO-BREAK SPACE
1DA7F  SIGNWRITING LOCATION-WALLPLANE SPACE
1DA80  SIGNWRITING LOCATION-FLOORPLANE SPACE
E0020  TAG SPACE
```

We decided to include a subset of the Unicode-defined spaces (marked with asterisks in the above list), even though their meaning and usage seems vague. The problem of spaces touches a general problem of the relation between the Unicode standard and typography. We discuss this topic in more detail in the next section.

The next two pages show the representative subset (for LM Math and TG Termes Math) of the repertoire we adopted as the GUST e-foundry "private standard"; gray squares denote zero-width characters. As one can see, the repertoires are very similar. One can assume that the difference in the repertoire will be imperceptible in practical applications.

6.4 Unicode: the typographer's friend or enemy?

An important subject, closely related to the contents and repertoire issues, is briefly mentioned in Sections 4.2, 6.1, and 6.3: the problem of the character set defined by the Unicode standard [35], and specified for math fonts in *Unicode Technical Report #25* [37].

The Unicode Consortium claims that "the Unicode standard follows a set of fundamental principles" and gives, among others, the following "principle": *characters, not glyphs* and *semantics*. We are not able to reconcile these principles with the cases discussed in this section.

It should be emphasized that not all glyphs used extensively in typography, in particular, in or out of math formulas, are assigned Unicode numbers. Examples of such glyphs are small caps and old style numerals. Nothing in these two classes of glyphs has received Unicode numbers, although there are codes for much more narrowly used double struck characters or monospaced and sans-serif digits.

Another example of "unicodeless" glyphs are the pieces used for assembling extensible characters (cf. Section 6.1, Figures 5 and 6).

It is not so bad if whole blocks of characters are included or excluded. The worse situation is an inconsistency of including only some glyphs. The abovementioned double struck glyphs are a good example of such a situation. Here is the group of double struck glyphs assigned Unicode slots, without apparent rhyme or reason:

```
2102 DOUBLE-STRUCK CAPITAL C
210D DOUBLE-STRUCK CAPITAL H
2115 DOUBLE-STRUCK CAPITAL N
2119 DOUBLE-STRUCK CAPITAL P
211A DOUBLE-STRUCK CAPITAL Q
211D DOUBLE-STRUCK CAPITAL R
2124 DOUBLE-STRUCK CAPITAL Z
213C DOUBLE-STRUCK SMALL PI
213D DOUBLE-STRUCK SMALL GAMMA
213E DOUBLE-STRUCK CAPITAL GAMMA
213F DOUBLE-STRUCK CAPITAL PI
2140 DOUBLE-STRUCK N-ARY SUMMATION
2145 DOUBLE-STRUCK ITALIC CAPITAL D
2146 DOUBLE-STRUCK ITALIC SMALL D
2147 DOUBLE-STRUCK ITALIC SMALL E
2148 DOUBLE-STRUCK ITALIC SMALL I
2149 DOUBLE-STRUCK ITALIC SMALL J
```

The remaining letters of the alphabet (upper and lower case) and digits received mathematical codes (1D538–1D56B), with gaps at the glyphs above.

Another notable example are sub- and superscripts, theoretically not needed in math fonts, because the sub- and superscript characters ("unicodeless") are accessed there by the OTF feature mechanism (the `ssty` feature, cf. Section 6.1). In practice, for legacy reasons, sub- and superscript glyphs are expected to be present in a text font, and, consequently, in the basic (plain) charset of a math font too — see Table 1. The Unicode standard [35] enumerates the following Latin letters in this context:

```
1D62 LATIN SUBSCRIPT SMALL LETTER I
1D63 LATIN SUBSCRIPT SMALL LETTER R
1D64 LATIN SUBSCRIPT SMALL LETTER U
1D65 LATIN SUBSCRIPT SMALL LETTER V
2090 LATIN SUBSCRIPT SMALL LETTER A
2091 LATIN SUBSCRIPT SMALL LETTER E
2092 LATIN SUBSCRIPT SMALL LETTER O
2093 LATIN SUBSCRIPT SMALL LETTER X
2094 LATIN SUBSCRIPT SMALL LETTER SCHWA
2095 LATIN SUBSCRIPT SMALL LETTER H
2096 LATIN SUBSCRIPT SMALL LETTER K
2097 LATIN SUBSCRIPT SMALL LETTER L
2098 LATIN SUBSCRIPT SMALL LETTER M
2099 LATIN SUBSCRIPT SMALL LETTER N
209A LATIN SUBSCRIPT SMALL LETTER P
209B LATIN SUBSCRIPT SMALL LETTER S
209C LATIN SUBSCRIPT SMALL LETTER T
2C7C LATIN SUBSCRIPT SMALL LETTER J
2071 SUPERSCRIPT LATIN SMALL LETTER I
207F SUPERSCRIPT LATIN SMALL LETTER N
```

Besides the somewhat surprising presence of the 'schwa' character and the striking asymmetry between the number of sub- and superscripts, most questionable here is the incompleteness of the Latin alphabet. So far, we have not included these glyphs in neither text nor math fonts.

Another example concerns math italic symbols. The Unicode standard reads:

```
1D44E MATHEMATICAL ITALIC SMALL A
1D44F MATHEMATICAL ITALIC SMALL B
1D450 MATHEMATICAL ITALIC SMALL C
1D451 MATHEMATICAL ITALIC SMALL D
1D452 MATHEMATICAL ITALIC SMALL E
1D453 MATHEMATICAL ITALIC SMALL F
1D454 MATHEMATICAL ITALIC SMALL G
                                 ⇐?
1D456 MATHEMATICAL ITALIC SMALL I
1D457 MATHEMATICAL ITALIC SMALL J
1D458 MATHEMATICAL ITALIC SMALL K
1D459 MATHEMATICAL ITALIC SMALL L
1D45A MATHEMATICAL ITALIC SMALL M
1D45B MATHEMATICAL ITALIC SMALL N
1D45C MATHEMATICAL ITALIC SMALL O
1D45D MATHEMATICAL ITALIC SMALL P
1D45E MATHEMATICAL ITALIC SMALL Q
1D45F MATHEMATICAL ITALIC SMALL R
```

Glyph repertoire of Latin Modern Math

ASCII: ! " # $ % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ? @ A B C D E F G H I J K L M N O P Q R S T U V W X Y Z [\ ] ^ _ ` a b c d e f g h i j k l m n o p q r s t u v w x y z { | } ~

Latin-1: ¡ ¢ £ ¤ ¥ ¦ § ¨ © ª « ¬ ® ¯ ° ± ´ µ ¶ · ¸ º » ¿ À Á Â Ã Ä Å Æ Ç È É Ê Ë Ì Í Î Ï Ð Ñ Ò Ó Ô Õ Ö × Ø Ù Ú Û Ü Ý Þ ß à á â ã ä å æ ç è é ê ë ì í î ï ð ñ ò ó ô õ ö ÷ ø ù ú û ü ý þ ÿ

Latin-1 Supplement: ĀāĂăĄąĆćČčĎďĐđĒēĖėĘęĚěĞğĢģĪīĮįİıĲĳĶķĹĺĻļĽľŁłŃńŅņŇň ŊŋŌōŐőŒœŔŕŖŗŘřŚśŞşŠšŢţŤťŨũŪūŮůŰűŲųŸŹźŻżŽžſƠơƯưȘșȚțȷ ˆ ˇ ˘ ˙ ˚ ˛ ˜ ˝

Combining Diacritical Marks: ̀ ́ ̂ ̃ ̄ ̅ ̆ ̇ ̈ ̉ ̊ ̋ ̌ ̏ ̑ ̣ ̦ ̧ ̨ ̬ ̭ ̮ ̯ ̰ ̱ ̲ ̳ ̸ ̿ ͍

Greek: Α Β Γ Δ Ε Ζ Η Θ Ι Κ Λ Μ Ν Ξ Ο Π Ρ Σ Τ Υ Φ Χ Ψ Ω α β γ δ ε ζ η θ ι κ λ μ ν ξ ο π ρ ς σ τ υ φ χ ψ ω ϑ ϕ ϖ ϰ ϱ ϴ ϵ

Latin Extended Additional: Ạ ạ Ả ả Ấ ấ Ầ ầ Ẩ ẩ Ẫ ẫ Ậ ậ Ắ ắ Ằ ằ Ẳ ẳ Ẵ ẵ Ặ ặ Ẹ ẹ Ẻ ẻ Ẽ ẽ Ế ế Ề ề Ể ể Ễ ễ Ệ ệ Ỉ ỉ Ị ị Ọ ọ Ỏ ỏ Ố ố Ồ ồ Ổ ổ Ỗ ỗ Ộ ộ Ớ ớ Ờ ờ Ở ở Ỡ ỡ Ợ ợ Ụ ụ Ủ ủ Ứ ứ Ừ ừ Ử ử Ữ ữ Ự ự Ỳ ỳ Ỵ ỵ Ỷ ỷ Ỹ ỹ

General Punctuation: ‐ ‒ – — ― ‖ ‗ ‘ ’ ‚ “ ” „ † ‡ • … ‰ ‱ ′ ″ ‴ ‵ ‶ ‷ ‹ › ※ ‽ ⁄ ⁒ ⁗

Currency Symbols: ₡ €

Combining Diacritical Marks for Symbols: ⃐ ⃑ ⃒ ⃓ ⃔ ⃕ ⃖ ⃗ ⃛ ⃜ ⃝ ⃞ ⃟ ⃡ ⃤ ⃥ ⃦ ⃨ ⃩ ⃪ ⃫ ⃬ ⃭ ⃮ ⃯ ⃰

Letterlike Symbols: ℂ ℃ ℇ ℉ ℋ ℌ ℍ ℎ ℏ ℐ ℑ ℒ ℓ ℕ № ℗ ℘ ℙ ℚ ℛ ℜ ℝ ℞ ℠ ™ ℤ Ω ℧ ℨ K Å ℬ ℭ ℮ ℰ ℱ ℳ ℵ ℶ ℷ ℸ ℼ ℽ ℾ ℿ ⅀ ⅅ ⅆ ⅇ ⅈ ⅉ

Arrows: ← ↑ → ↓ ↔ ↕ ↖ ↗ ↘ ↙ ↚ ↛ ↞ ↟ ↠ ↡ ↢ ↣ ↤ ↥ ↦ ↧ ↩ ↪ ↫ ↬ ↭ ↮ ↰ ↱ ↲ ↳ ↴ ↵ ↶ ↷ ↺ ↻ ↼ ↽ ↾ ↿ ⇀ ⇁ ⇂ ⇃ ⇄ ⇅ ⇆ ⇇ ⇈ ⇉ ⇊ ⇋ ⇌ ⇍ ⇎ ⇏ ⇐ ⇑ ⇒ ⇓ ⇔ ⇕ ⇖ ⇗ ⇘ ⇙ ⇚ ⇛ ⇜ ⇝ ⇦ ⇧ ⇨ ⇩ ⇳ ⇵ ⇶

Mathematical Operators: ∀ ∁ ∂ ∃ ∄ ∅ ∆ ∇ ∈ ∉ ∊ ∋ ∌ ∍ ∎ ∏ ∐ ∑ − ∓ ∔ ∕ ∖ ∗ ∘ ∙ √ ∝ ∞ ∟ ∠ ∡ ∢ ∣ ∤ ∥ ∦ ∧ ∨ ∩ ∪ ∫ ∬ ∭ ∮ ∯ ∰ ∱ ∲ ∳ ∴ ∵ ∶ ∷ ∸ ∹ ∺ ∻ ∼ ∽ ∾ ∿ ≀ ≁ ≂ ≃ ≄ ≅ ≆ ≇ ≈ ≉ ≊ ≋ ≌ ≍ ≎ ≏ ≐ ≑ ≒ ≓ ≔ ≕ ≖ ≗ ≘ ≙ ≚ ≛ ≜ ≝ ≞ ≟ ≠ ≡ ≢ ≣ ≤ ≥ ≦ ≧ ≨ ≩ ≪ ≫ ≬ ≭ ≮ ≯ ≰ ≱ ≲ ≳ ≴ ≵ ≶ ≷ ≸ ≹ ≺ ≻ ≼ ≽ ≾ ≿ ⊀ ⊁ ⊂ ⊃ ⊄ ⊅ ⊆ ⊇ ⊈ ⊉ ⊊ ⊋ ⊌ ⊍ ⊎ ⊏ ⊐ ⊑ ⊒ ⊓ ⊔ ⊕ ⊖ ⊗ ⊘ ⊙ ⊚ ⊛ ⊜ ⊝ ⊞ ⊟ ⊠ ⊡ ⊢ ⊣ ⊤ ⊥ ⊦ ⊧ ⊨ ⊩ ⊪ ⊫ ⊬ ⊭ ⊮ ⊯ ⊲ ⊳ ⊴ ⊵ ⊶ ⊷ ⊸ ⊹ ⊺ ⊻ ⊼ ⊽ ⊾ ⊿ ⋀ ⋁ ⋂ ⋃ ⋄ ⋅ ⋆ ⋇ ⋈ ⋉ ⋊ ⋋ ⋌ ⋍ ⋎ ⋏ ⋐ ⋑ ⋒ ⋓ ⋕ ⋖ ⋗ ⋘ ⋙ ⋚ ⋛ ⋜ ⋝ ⋞ ⋟ ⋠ ⋡ ⋢ ⋣ ⋤ ⋥ ⋦ ⋧ ⋨ ⋩ ⋪ ⋫ ⋬ ⋭ ⋮ ⋯ ⋰ ⋱

Miscellaneous Technical: ⌀ ⌅ ⌆ ⌈ ⌉ ⌊ ⌋ ⌐ ⌙ ⌜ ⌝ ⌞ ⌟ ⌠ ⌡ ⌢ ⌣ 〈 〉 ⎛ ⎜ ⎝ ⎞ ⎟ ⎠ ⎡ ⎢ ⎣ ⎤ ⎥ ⎦ ⎧ ⎨ ⎩ ⎪ ⎫ ⎬ ⎭ ⎲ ⎳ ⎴ ⎵ ⎷ ⎸ ⏜ ⏝ ⏞ ⏟ ⏠ ⏡

Box Drawing, Block Elements: ─ │ ┌ ┐ └ ┘ ├ ┤ ┬ ┴ ┼ ▁ █ ░ ▒ ▓

Geometric Shapes: ■ □ ▪ ▫ ▬ ▭ ▲ △ ▶ ▷ ▼ ▽ ◀ ◁ ◊ ○ ● ◦ ◯

Miscellaneous Symbols, Dingbats: ♠ ♡ ♢ ♣ ♤ ♥ ♦ ♧ ♪ ♭ ♮ ♯ ⚭ ⚮ ✓ ✠ ❘ ➡

Miscellaneous Mathematical Symbols-A: ⟂ ⟘ ⟙ ⟚ ⟛ ⟜ ⟝ ⟞ ⟠ ⟡ ⟢ ⟣ ⟦ ⟧ ⟨ ⟩ ⟪ ⟫ ⟮ ⟯

Supplemental Arrows-A, B: ⟴ ⟵ ⟶ ⟷ ⟸ ⟹ ⟺ ⟻ ⟼ ⟽ ⟾ ⟿ ⤆ ⤇

Supplemental Mathematical Operators: ⨀ ⨁ ⨂ ⨃ ⨄ ⨅ ⨆ ⨉ ⨌ ⨑ ⨯ ⨿ ⩽ ⩾ ⪅ ⪆ ⪇ ⪈ ⪉ ⪊ ⪋ ⪌ ⪕ ⪖ ⪯ ⪰

Miscellaneous Symbols and Arrows: ⬄ ⬅ ⬆ ⬇ ⬌ ⬍ ⬚ ⬱ ⬳

CJK Symbols and Punctuation: 〚 〛 Alphabetic Presentation Forms: ﬀ ﬁ ﬂ ﬃ ﬄ

Mathematical Alphanumeric Symbols: 𝐀𝐁𝐂𝐃𝐄𝐅𝐆𝐇𝐈𝐉𝐊𝐋𝐌𝐍𝐎𝐏𝐐𝐑𝐒𝐓𝐔𝐕𝐖𝐗𝐘𝐙𝐚𝐛𝐜𝐝𝐞𝐟𝐠𝐡𝐢𝐣𝐤𝐥𝐦𝐧𝐨𝐩𝐪𝐫𝐬𝐭𝐮𝐯𝐰𝐱𝐲𝐳𝐴𝐵𝐶𝐷𝐸𝐹𝐺𝐻𝐼𝐽𝐾𝐿𝑀𝑁𝑂𝑃𝑄𝑅𝑆𝑇𝑈𝑉𝑊𝑋𝑌𝑍𝑎𝑏𝑐𝑑𝑒𝑓𝑔𝑖𝑗𝑘𝑙𝑚𝑛𝑜𝑝𝑞𝑟𝑠𝑡𝑢𝑣𝑤𝑥𝑦𝑧𝑨𝑩𝑪𝑫𝑬𝑭𝑮𝑯𝑰𝑱𝑲𝑳𝑴𝑵𝑶𝑷𝑸𝑹𝑺𝑻𝑼𝑽𝑾𝑿𝒀𝒁𝒂𝒃𝒄𝒅𝒆𝒇𝒈𝒉𝒊𝒋𝒌𝒍𝒎𝒏𝒐𝒑𝒒𝒓𝒔𝒕𝒖𝒗𝒘𝒙𝒚𝒛𝒜𝒞𝒟𝒢𝒥𝒦𝒩𝒪𝒫𝒬𝒮𝒯𝒰𝒱𝒲𝒳𝒴𝒵𝓐𝓑𝓒𝓓𝓔𝓕𝓖𝓗𝓘𝓙𝓚𝓛𝓜𝓝𝓞𝓟𝓠𝓡𝓢𝓣𝓤𝓥𝓦𝓧𝓨𝓩𝔄𝔅𝔇𝔈𝔉𝔊𝔍𝔎𝔏𝔐𝔑𝔒𝔓𝔔𝔖𝔗𝔘𝔙𝔚𝔛𝔜𝔞𝔟𝔠𝔡𝔢𝔣𝔤𝔥𝔦𝔧𝔨𝔩𝔪𝔫𝔬𝔭𝔮𝔯𝔰𝔱𝔲𝔳𝔴𝔵𝔶𝔷𝔸𝔹𝔻𝔼𝔽𝔾𝕀𝕁𝕂𝕃𝕄𝕆𝕊𝕋𝕌𝕍𝕎𝕏𝕐𝕒𝕓𝕔𝕕𝕖𝕗𝕘𝕙𝕚𝕛𝕜𝕝𝕞𝕟𝕠𝕡𝕢𝕣𝕤𝕥𝕦𝕧𝕨𝕩𝕪𝕫𝕬𝕭𝕮𝕯𝕰𝕱𝕲𝕳𝕴𝕵𝕶𝕷𝕸𝕹𝕺𝕻𝕼𝕽𝕾𝕿𝖀𝖁𝖂𝖃𝖄𝖅𝖆𝖇𝖈𝖉𝖊𝖋𝖌𝖍𝖎𝖏𝖐𝖑𝖒𝖓𝖔𝖕𝖖𝖗𝖘𝖙𝖚𝖛𝖜𝖝𝖞𝖟𝖠𝖡𝖢𝖣𝖤𝖥𝖦𝖧𝖨𝖩𝖪𝖫𝖬𝖭𝖮𝖯𝖰𝖱𝖲𝖳𝖴𝖵𝖶𝖷𝖸𝖹𝖺𝖻𝖼𝖽𝖾𝖿𝗀𝗁𝗂𝗃𝗄𝗅𝗆𝗇𝗈𝗉𝗊𝗋𝗌𝗍𝗎𝗏𝗐𝗑𝗒𝗓𝗔𝗕𝗖𝗗𝗘𝗙𝗚𝗛𝗜𝗝𝗞𝗟𝗠𝗡𝗢𝗣𝗤𝗥𝗦𝗧𝗨𝗩𝗪𝗫𝗬𝗭𝗮𝗯𝗰𝗱𝗲𝗳𝗴𝗵𝗶𝗷𝗸𝗹𝗺𝗻𝗼𝗽𝗾𝗿𝘀𝘁𝘂𝘃𝘄𝘅𝘆𝘇𝘈𝘉𝘊𝘋𝘌𝘍𝘎𝘏𝘐𝘑𝘒𝘓𝘔𝘕𝘖𝘗𝘘𝘙𝘚𝘛𝘜𝘝𝘞𝘟𝘠𝘡𝘢𝘣𝘤𝘥𝘦𝘧𝘨𝘩𝘪𝘫𝘬𝘭𝘮𝘯𝘰𝘱𝘲𝘳𝘴𝘵𝘶𝘷𝘸𝘹𝘺𝘻𝘼𝘽𝘾𝘿𝙀𝙁𝙂𝙃𝙄𝙅𝙆𝙇𝙈𝙉𝙊𝙋𝙌𝙍𝙎𝙏𝙐𝙑𝙒𝙓𝙔𝙕𝙖𝙗𝙘𝙙𝙚𝙛𝙜𝙝𝙞𝙟𝙠𝙡𝙢𝙣𝙤𝙥𝙦𝙧𝙨𝙩𝙪𝙫𝙬𝙭𝙮𝙯𝙰𝙱𝙲𝙳𝙴𝙵𝙶𝙷𝙸𝙹𝙺𝙻𝙼𝙽𝙾𝙿𝚀𝚁𝚂𝚃𝚄𝚅𝚆𝚇𝚈𝚉𝚊𝚋𝚌𝚍𝚎𝚏𝚐𝚑𝚒𝚓𝚔𝚕𝚖𝚗𝚘𝚙𝚚𝚛𝚜𝚝𝚞𝚟𝚠𝚡𝚢𝚣𝚤𝚥𝚨𝚩𝚪𝚫𝚬𝚭𝚮𝚯𝚰𝚱𝚲𝚳𝚴𝚵𝚶𝚷𝚸𝚹𝚺𝚻𝚼𝚽𝚾𝚿𝛀𝛁𝛂𝛃𝛄𝛅𝛆𝛇𝛈𝛉𝛊𝛋𝛌𝛍𝛎𝛏𝛐𝛑𝛒𝛓𝛔𝛕𝛖𝛗𝛘𝛙𝛚𝛛𝛜𝛝𝛞𝛟𝛠𝛡𝛢𝛣𝛤𝛥𝛦𝛧𝛨𝛩𝛪𝛫𝛬𝛭𝛮𝛯𝛰𝛱𝛲𝛳𝛴𝛵𝛶𝛷𝛸𝛹𝛺𝛻𝛼𝛽𝛾𝛿𝜀𝜁𝜂𝜃𝜄𝜅𝜆𝜇𝜈𝜉𝜊𝜋𝜌𝜍𝜎𝜏𝜐𝜑𝜒𝜓𝜔𝜕𝜖𝜗𝜘𝜙𝜚𝜛𝜜𝜝𝜞𝜟𝜠𝜡𝜢𝜣𝜤𝜥𝜦𝜧𝜨𝜩𝜪𝜫𝜬𝜭𝜮𝜯𝜰𝜱𝜲𝜳𝜴𝜵𝜶𝜷𝜸𝜹𝜺𝜻𝜼𝜽𝜾𝜿𝝀𝝁𝝂𝝃𝝄𝝅𝝆𝝇𝝈𝝉𝝊𝝋𝝌𝝍𝝎𝝏𝝐𝝑𝝒𝝓𝝔𝝕𝝖𝝗𝝘𝝙𝝚𝝛𝝜𝝝𝝞𝝟𝝠𝝡𝝢𝝣𝝤𝝥𝝦𝝧𝝨𝝩𝝪𝝫𝝬𝝭𝝮𝝯𝝰𝝱𝝲𝝳𝝴𝝵𝝶𝝷𝝸𝝹𝝺𝝻𝝼𝝽𝝾𝝿𝞀𝞁𝞂𝞃𝞄𝞅𝞆𝞇𝞈𝞉𝞊𝞋𝞌𝞍𝞎𝞏𝞐𝞑𝞒𝞓𝞔𝞕𝞖𝞗𝞘𝞙𝞚𝞛𝞜𝞝𝞞𝞟𝞠𝞡𝞢𝞣𝞤𝞥𝞦𝞧𝞨𝞩𝞪𝞫𝞬𝞭𝞮𝞯𝞰𝞱𝞲𝞳𝞴𝞵𝞶𝞷𝞸𝞹𝞺𝞻𝞼𝞽𝞾𝞿𝟀𝟁𝟂𝟃𝟄𝟅𝟆𝟇𝟈𝟉𝟎𝟏𝟐𝟑𝟒𝟓𝟔𝟕𝟖𝟗𝟘𝟙𝟚𝟛𝟜𝟝𝟞𝟟𝟠𝟡𝟢𝟣𝟤𝟥𝟦𝟧𝟨𝟩𝟪𝟫𝟬𝟭𝟮𝟯𝟰𝟱𝟲𝟳𝟴𝟵𝟶𝟷𝟸𝟹𝟺𝟻𝟼𝟽𝟾𝟿

Glyph repertoire of TeX Gyre Termes Math

ASCII: ! " # $ % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ? @ A B C D E F G H I J K L M N O P Q R S T U V W X Y Z [\ ] ^ _ ` a b c d e f g h i j k l m n o p q r s t u v w x y z { | } ~

Latin-1: ¡ ¢ £ ¤ ¥ ¦ § ¨ © ª « ¬ ® ¯ ° ± ´ µ ¶ · ¸ º » ¿ À Á Â Ã Ä Å Æ Ç È É Ê Ë Ì Í Î Ï Ð Ñ Ò Ó Ô Õ Ö × Ø Ù Ú Û Ü Ý Þ ß à á â ã ä å æ ç è é ê ë ì í î ï ð ñ ò ó ô õ ö ÷ ø ù ú û ü ý þ ÿ

Latin-1 Supplement: Ā ā Ă ă Ą ą Ć ć Č č Ď ď Đ đ Ē ē Ė ė Ę ę Ě ě Ğ ğ Ģ ģ Ĩ ĩ Ī ī Į į İ ı Ĳ ĳ Ķ ķ Ĺ ĺ Ļ ļ Ľ ľ Ł ł Ń ń Ņ ņ Ň ň Ŋ ŋ Ō ō Ő ő Œ œ Ŕ ŕ Ŗ ŗ Ř ř Ś ś Ş ş Š š Ţ ţ Ť ť Ũ ũ Ū ū Ů ů Ű ű Ų ų Ÿ Ź ź Ż ż Ž ž ſ ƒ Ơ ơ Ư ư Ș ș Ț ț ȷ ˆ ˇ ˘ ˙ ˚ ˛ ˜ ˝

Combining Diacritical Marks: ◌̀ ◌́ ◌̂ ◌̃ ◌̄ ◌̅ ◌̆ ◌̇ ◌̈ ◌̉ ◌̊ ◌̋ ◌̌ ◌̏ ◌̑ ◌̣ ◌̦ ◌̬ ◌̭ ◌̮ ◌̯ ◌̰ ◌̱ ◌̲ ◌̳ ◌̸ ◌̿ ◌͍

Greek: Α Β Γ Δ Ε Ζ Η Θ Ι Κ Λ Μ Ν Ξ Ο Π Ρ Σ Τ ϒ Φ Χ Ψ Ω α β γ δ ε ζ η θ ι κ λ μ ν ξ ο π ρ ς σ τ υ φ χ ψ ω ϑ ϕ ϖ ϰ ϱ ϴ ϵ

Latin Extended Additional: Ạ ạ Ả ả Ấ ấ Ầ ầ Ẩ ẩ Ẫ ẫ Ậ ậ Ắ ắ Ằ ằ Ẳ ẳ Ẵ ẵ Ặ ặ Ẹ ẹ Ẻ ẻ Ẽ ẽ Ế ế Ề ề Ể ể Ễ ễ Ệ ệ Ỉ ỉ Ị ị Ọ ọ Ỏ ỏ Ố ố Ồ ồ Ổ ổ Ỗ ỗ Ộ ộ Ớ ớ Ờ ờ Ở ở Ỡ ỡ Ợ ợ Ụ ụ Ủ ủ Ứ ứ Ừ ừ Ử ử Ữ ữ Ự ự Ỳ ỳ Ỵ ỵ Ỷ ỷ Ỹ ỹ

General Punctuation: ◌ ◌ ◌ ‐ ‒ – — ― ‖ ‗ ‘ ’ ‚ “ ” „ † ‡ • … ‰ ‱ ′ ″ ‴ ‵ ‶ ‷ ‹ › ※ ‽ ⁄ ⁒ ⁗ ◌ ⁡ ◌ ◌ ◌

Currency Symbols: ₡ ₤ ₦ ₩ ₫ € ₱ ₲

Combining Diacritical Marks for Symbols: ◌⃐ ◌⃑ ◌⃒ ◌⃓ ◌⃔ ◌⃕ ◌⃖ ◌⃗ ◌⃘ ◌⃛ ◌⃜ ◌⃝ ◌⃞ ◌⃟ ◌⃡ ◌⃤ ◌⃥ ◌⃦ ◌⃨ ◌⃩ ◌⃪ ◌⃫ ◌⃬ ◌⃭ ◌⃮ ◌⃯ ◌⃰

Letterlike Symbols: ℂ ℃ ℇ ℉ ℊ ℋ ℌ ℍ ℎ ℏ ℐ ℑ ℒ ℓ ℕ № ℗ ℘ ℙ ℚ ℛ ℜ ℝ ℞ ℠ ™ ℤ Ω ℧ ℨ K Å ℬ ℭ ℮ ℯ ℰ ℱ ℳ ℴ ℵ ℶ ℷ ℸ ℼ ℽ ℾ ℿ ⅀ ⅅ ⅆ ⅇ ⅈ ⅉ

Arrows: ← ↑ → ↓ ↔ ↕ ↖ ↗ ↘ ↙ ↚ ↛ ↞ ↟ ↠ ↡ ↢ ↣ ↤ ↥ ↦ ↧ ↩ ↪ ↫ ↬ ↭ ↮ ↰ ↱ ↲ ↳ ↴ ↵ ↶ ↷ ↺ ↻ ↼ ↽ ↾ ↿ ⇀ ⇁ ⇂ ⇃ ⇄ ⇅ ⇆ ⇇ ⇈ ⇉ ⇊ ⇋ ⇌ ⇍ ⇎ ⇏ ⇐ ⇑ ⇒ ⇓ ⇔ ⇕ ⇖ ⇗ ⇘ ⇙ ⇚ ⇛ ⇜ ⇝ ⇦ ⇧ ⇨ ⇩ ⇳ ⇵ ⇶

Mathematical Operators: ∀ ∁ ∂ ∃ ∄ ∅ ∆ ∇ ∈ ∉ ∊ ∋ ∌ ∍ ∎ ∏ ∐ ∑ − ∓ ∔ ∕ ∖ ∗ ∘ ∙ √ ∝ ∞ ∟ ∠ ∡ ∢ ∣ ∤ ∥ ∦ ∧ ∨ ∩ ∪ ∫ ∬ ∭ ∮ ∯ ∰ ∱ ∲ ∳ ∴ ∵ ∶ ∷ ∸ ∹ ∺ ∻ ∼ ∽ ∾ ∿ ≀ ≁ ≂ ≃ ≄ ≅ ≆ ≇ ≈ ≉ ≊ ≋ ≌ ≍ ≎ ≏ ≐ ≑ ≒ ≓ ≔ ≕ ≖ ≗ ≘ ≙ ≚ ≛ ≜ ≝ ≞ ≟ ≠ ≡ ≢ ≣ ≤ ≥ ≦ ≧ ≨ ≩ ≪ ≫ ≬ ≭ ≮ ≯ ≰ ≱ ≲ ≳ ≴ ≵ ≶ ≷ ≸ ≹ ≺ ≻ ≼ ≽ ≾ ≿ ⊀ ⊁ ⊂ ⊃ ⊄ ⊅ ⊆ ⊇ ⊈ ⊉ ⊊ ⊋ ⊌ ⊍ ⊎ ⊏ ⊐ ⊑ ⊒ ⊓ ⊔ ⊕ ⊖ ⊗ ⊘ ⊙ ⊚ ⊛ ⊜ ⊝ ⊞ ⊟ ⊠ ⊡ ⊢ ⊣ ⊤ ⊥ ⊦ ⊧ ⊨ ⊩ ⊪ ⊫ ⊬ ⊭ ⊮ ⊯ ⊲ ⊳ ⊴ ⊵ ⊶ ⊷ ⊸ ⊹ ⊺ ⊻ ⊼ ⊽ ⊾ ⊿ ⋀ ⋁ ⋂ ⋃ ⋄ ⋅ ⋆ ⋇ ⋈ ⋉ ⋊ ⋋ ⋌ ⋍ ⋎ ⋏ ⋐ ⋑ ⋒ ⋓ ⋕ ⋖ ⋗ ⋘ ⋙ ⋚ ⋛ ⋜ ⋝ ⋞ ⋟ ⋠ ⋡ ⋢ ⋣ ⋤ ⋥ ⋦ ⋧ ⋨ ⋩ ⋪ ⋫ ⋬ ⋭ ⋮ ⋯ ⋰ ⋱

Miscellaneous Technical: ⌀ ⌅ ⌆ ⌈ ⌉ ⌊ ⌋ ⌐ ⌙ ⌜ ⌝ ⌞ ⌟ ⌠ ⌡ ⌢ ⌣ 〈 〉 ⎛ ⎜ ⎝ ⎞ ⎟ ⎠ ⎡ ⎢ ⎣ ⎤ ⎥ ⎦ ⎧ ⎨ ⎩ ⎪ ⎫ ⎬ ⎭ ⎮ ⎲ ⎳ ⎴ ⎵ ⎷ ⎸ ⏜ ⏝ ⏞ ⏟ ⏠ ⏡

Box Drawing, Block Elements: ─ │ ┌ ┐ └ ┘ ├ ┤ ┬ ┴ ┼ ▁ █ ░ ▒ ▓

Geometric Shapes: ■ □ ▪ ▫ ▬ ▭ ▲ △ ▶ ▷ ▼ ▽ ◀ ◁ ◊ ○ ● ◦ ◯

Miscellaneous Symbols, Dingbats: ♠ ♡ ♢ ♣ ♤ ♥ ♦ ♧ ♪ ♭ ♮ ♯ ⚭ ⚮ ✓ ✠ ➡

Miscellaneous Mathematical Symbols-A: ⟂ ⟘ ⟙ ⟚ ⟛ ⟜ ⟝ ⟞ ⟠ ⟡ ⟢ ⟣ ⟦ ⟧ ⟨ ⟩ ⟪ ⟫ ⟮ ⟯

Supplemental Arrows-A, B: ⟴ ⟵ ⟶ ⟷ ⟸ ⟹ ⟺ ⟻ ⟼ ⟽ ⟾ ⟿ ⤆ ⤇

Supplemental Mathematical Operators: ⨀ ⨁ ⨂ ⨃ ⨄ ⨅ ⨆ ⨉ ⨌ ⨏ ⨯ ⨿ ⩽ ⩾ ⪅ ⪆ ⪇ ⪈ ⪉ ⪊ ⪋ ⪌ ⪕ ⪖ ⪯ ⪰

Miscellaneous Symbols and Arrows: ⬄ ⬅ ⬆ ⬇ ⬌ ⬍ ⬚ ⬱ ⬳

CJK Symbols and Punctuation: 〚 〛 Alphabetic Presentation Forms: ﬀ ﬁ ﬂ ﬃ ﬄ

Mathematical Alphanumeric Symbols: 𝐀𝐁𝐂𝐃𝐄𝐅𝐆𝐇𝐈𝐉𝐊𝐋𝐌𝐍𝐎𝐏𝐐𝐑𝐒𝐓𝐔𝐕𝐖𝐗𝐘𝐙𝐚𝐛𝐜𝐝𝐞𝐟𝐠𝐡𝐢𝐣𝐤𝐥𝐦𝐧𝐨𝐩𝐪𝐫𝐬𝐭𝐮𝐯𝐰𝐱𝐲𝐳𝐴𝐵𝐶𝐷𝐸𝐹𝐺𝐻𝐼𝐽𝐾𝐿𝑀𝑁𝑂𝑃𝑄𝑅𝑆𝑇𝑈𝑉𝑊𝑋𝑌𝑍𝑎𝑏𝑐𝑑𝑒𝑓𝑔𝑖𝑗𝑘𝑙𝑚𝑛𝑜𝑝𝑞𝑟𝑠𝑡𝑢𝑣𝑤𝑥𝑦𝑧𝑨𝑩𝑪𝑫𝑬𝑭𝑮𝑯𝑰𝑱𝑲𝑳𝑴𝑵𝑶𝑷𝑸𝑹𝑺𝑻𝑼𝑽𝑾𝑿𝒀𝒁𝒂𝒃𝒄𝒅𝒆𝒇𝒈𝒉𝒊𝒋𝒌𝒍𝒎𝒏𝒐𝒑𝒒𝒓𝒔𝒕𝒖𝒗𝒘𝒙𝒚𝒛𝒜𝒞𝒟𝒢𝒥𝒦𝒩𝒪𝒫𝒬𝒮𝒯𝒰𝒱𝒲𝒳𝒴𝒵𝒶𝒷𝒸𝒹𝒻𝒽𝒾𝒿𝓀𝓁𝓂𝓃𝓅𝓆𝓇𝓈𝓉𝓊𝓋𝓌𝓍𝓎𝓏𝓐𝓑𝓒𝓓𝓔𝓕𝓖𝓗𝓘𝓙𝓚𝓛𝓜𝓝𝓞𝓟𝓠𝓡𝓢𝓣𝓤𝓥𝓦𝓧𝓨𝓩𝓪𝓫𝓬𝓭𝓮𝓯𝓰𝓱𝓲𝓳𝓴𝓵𝓶𝓷𝓸𝓹𝓺𝓻𝓼𝓽𝓾𝓿𝔀𝔁𝔂𝔃𝔄𝔅𝔇𝔈𝔉𝔊𝔍𝔎𝔏𝔐𝔑𝔒𝔓𝔔𝔖𝔗𝔘𝔙𝔚𝔛𝔜𝔞𝔟𝔠𝔡𝔢𝔣𝔤𝔥𝔦𝔧𝔨𝔩𝔪𝔫𝔬𝔭𝔮𝔯𝔰𝔱𝔲𝔳𝔴𝔵𝔶𝔷𝔸𝔹𝔻𝔼𝔽𝔾𝕀𝕁𝕂𝕃𝕄𝕆𝕊𝕋𝕌𝕍𝕎𝕏𝕐𝕒𝕓𝕔𝕕𝕖𝕗𝕘𝕙𝕚𝕛𝕜𝕝𝕞𝕟𝕠𝕡𝕢𝕣𝕤𝕥𝕦𝕧𝕨𝕩𝕪𝕫𝕬𝕭𝕮𝕯𝕰𝕱𝕲𝕳𝕴𝕵𝕶𝕷𝕸𝕹𝕺𝕻𝕼𝕽𝕾𝕿𝖀𝖁𝖂𝖃𝖄𝖅𝖆𝖇𝖈𝖉𝖊𝖋𝖌𝖍𝖎𝖏𝖐𝖑𝖒𝖓𝖔𝖕𝖖𝖗𝖘𝖙𝖚𝖛𝖜𝖝𝖞𝖟𝖠𝖡𝖢𝖣𝖤𝖥𝖦𝖧𝖨𝖩𝖪𝖫𝖬𝖭𝖮𝖯𝖰𝖱𝖲𝖳𝖴𝖵𝖶𝖷𝖸𝖹𝖺𝖻𝖼𝖽𝖾𝖿𝗀𝗁𝗂𝗃𝗄𝗅𝗆𝗇𝗈𝗉𝗊𝗋𝗌𝗍𝗎𝗏𝗐𝗑𝗒𝗓𝗔𝗕𝗖𝗗𝗘𝗙𝗚𝗛𝗜𝗝𝗞𝗟𝗠𝗡𝗢𝗣𝗤𝗥𝗦𝗧𝗨𝗩𝗪𝗫𝗬𝗭𝗮𝗯𝗰𝗱𝗲𝗳𝗴𝗵𝗶𝗷𝗸𝗹𝗺𝗻𝗼𝗽𝗾𝗿𝘀𝘁𝘂𝘃𝘄𝘅𝘆𝘇𝘈𝘉𝘊𝘋𝘌𝘍𝘎𝘏𝘐𝘑𝘒𝘓𝘔𝘕𝘖𝘗𝘘𝘙𝘚𝘛𝘜𝘝𝘞𝘟𝘠𝘡𝘢𝘣𝘤𝘥𝘦𝘧𝘨𝘩𝘪𝘫𝘬𝘭𝘮𝘯𝘰𝘱𝘲𝘳𝘴𝘵𝘶𝘷𝘸𝘹𝘺𝘻𝘼𝘽𝘾𝘿𝙀𝙁𝙂𝙃𝙄𝙅𝙆𝙇𝙈𝙉𝙊𝙋𝙌𝙍𝙎𝙏𝙐𝙑𝙒𝙓𝙔𝙕𝙖𝙗𝙘𝙙𝙚𝙛𝙜𝙝𝙞𝙟𝙠𝙡𝙢𝙣𝙤𝙥𝙦𝙧𝙨𝙩𝙪𝙫𝙬𝙭𝙮𝙯𝙰𝙱𝙲𝙳𝙴𝙵𝙶𝙷𝙸𝙹𝙺𝙻𝙼𝙽𝙾𝙿𝚀𝚁𝚂𝚃𝚄𝚅𝚆𝚇𝚈𝚉𝚊𝚋𝚌𝚍𝚎𝚏𝚐𝚑𝚒𝚓𝚔𝚕𝚖𝚗𝚘𝚙𝚚𝚛𝚜𝚝𝚞𝚟𝚠𝚡𝚢𝚣𝚤𝚥𝚨𝚩𝚪𝚫𝚬𝚭𝚮𝚯𝚰𝚱𝚲𝚳𝚴𝚵𝚶𝚷𝚸𝚹𝚺𝚻𝚼𝚽𝚾𝚿𝛀𝛁𝛂𝛃𝛄𝛅𝛆𝛇𝛈𝛉𝛊𝛋𝛌𝛍𝛎𝛏𝛐𝛑𝛒𝛓𝛔𝛕𝛖𝛗𝛘𝛙𝛚𝛛𝛜𝛝𝛞𝛟𝛠𝛡𝛢𝛣𝛤𝛥𝛦𝛧𝛨𝛩𝛪𝛫𝛬𝛭𝛮𝛯𝛰𝛱𝛲𝛳𝛴𝛵𝛶𝛷𝛸𝛹𝛺𝛻𝛼𝛽𝛾𝛿𝜀𝜁𝜂𝜃𝜄𝜅𝜆𝜇𝜈𝜉𝜊𝜋𝜌𝜍𝜎𝜏𝜐𝜑𝜒𝜓𝜔𝜕𝜖𝜗𝜘𝜙𝜚𝜛𝜜𝜝𝜞𝜟𝜠𝜡𝜢𝜣𝜤𝜥𝜦𝜧𝜨𝜩𝜪𝜫𝜬𝜭𝜮𝜯𝜰𝜱𝜲𝜳𝜴𝜵𝜶𝜷𝜸𝜹𝜺𝜻𝜼𝜽𝜾𝜿𝝀𝝁𝝂𝝃𝝄𝝅𝝆𝝇𝝈𝝉𝝊𝝋𝝌𝝍𝝎𝝏𝝐𝝑𝝒𝝓𝝔𝝕𝝖𝝗𝝘𝝙𝝚𝝛𝝜𝝝𝝞𝝟𝝠𝝡𝝢𝝣𝝤𝝥𝝦𝝧𝝨𝝩𝝪𝝫𝝬𝝭𝝮𝝯𝝰𝝱𝝲𝝳𝝴𝝵𝝶𝝷𝝸𝝹𝝺𝝻𝝼𝝽𝝾𝝿𝞀𝞁𝞂𝞃𝞄𝞅𝞆𝞇𝞈𝞉𝞐𝞑𝞒𝞓𝞔𝞕𝞖𝞗𝞘𝞙𝞚𝞛𝞜𝞝𝞞𝞟𝞠𝞡𝞢𝞣𝞤𝞥𝞦𝞧𝞨𝞩𝞪𝞫𝞬𝞭𝞮𝞯𝞰𝞱𝞲𝞳𝞴𝞵𝞶𝞷𝞸𝞹𝞺𝞻𝞼𝞽𝞾𝞿𝟀𝟁𝟂𝟃𝟄𝟅𝟆𝟇𝟈𝟉𝟎𝟏𝟐𝟑𝟒𝟓𝟔𝟕𝟖𝟗𝟘𝟙𝟚𝟛𝟜𝟝𝟞𝟟𝟠𝟡𝟢𝟣𝟤𝟥𝟦𝟧𝟨𝟩𝟪𝟫𝟬𝟭𝟮𝟯𝟰𝟱𝟲𝟳𝟴𝟵𝟶𝟷𝟸𝟹𝟺𝟻𝟼𝟽𝟾𝟿

```
1D460 MATHEMATICAL ITALIC SMALL S
1D461 MATHEMATICAL ITALIC SMALL T
1D462 MATHEMATICAL ITALIC SMALL U
1D463 MATHEMATICAL ITALIC SMALL V
1D464 MATHEMATICAL ITALIC SMALL W
1D465 MATHEMATICAL ITALIC SMALL X
1D466 MATHEMATICAL ITALIC SMALL Y
1D467 MATHEMATICAL ITALIC SMALL Z
```

Math italic ‘h’ is apparently missing. In fact, the Unicode Consortium decided to leave the slot `1D455` unused forever and “inflate” the meaning of the slot formerly assigned to the Planck constant:

```
210E PLANCK CONSTANT
     = height, specific enthalpy, ...
     * simply a mathematical italic h;
       this character’s name results
       from legacy usage
```

More on Unicode’s ambiguities, inconsistencies, riddles, curiosities, etc. concerning typography applications, especially math typesetting, can be found in Piotr Strzelczyk’s ruminations [23].

From the typographer’s point of view, the enumeration of all signs used in the world does not seem to be a good idea. Therefore, it should be no surprise that it turned out to be impossible to create a math OTF font that has an internally logical and coherent structure and, at the same time, is practically useful. Conforming rigorously to the mentioned specifications does not help too much — it would lead, in our opinion, to fonts containing mostly seldom used glyphs. Needless to say, research examining which characters from the Unicode repertoire are actually used in practice, would be welcome. Lacking such empirical data, we adopted Cambria Math as our “reference point”. Cambria Math contains circa 6500 glyphs; we decided to reduce this number to at most 4500 for the TEX Gyre series of math fonts. We can reluctantly consider proposals for suitable extensions, if truly needed.

6.5 Compatibility issues

We already explained why 100-percent compatibility of the LM text fonts with the parent Computer Modern fonts is unfeasible; thereby, the same applies, even to a greater extent, to the LM Math font (cf. the case of horizontal extensible braces in Section 6.1).

Recall that we used an alternative calligraphic alphabet in LM Math (Section 6.1). This incompatibility can be relatively easily patched, by including two calligraphic scripts in the font and using OTF ‘stylistic sets’ features, implemented as the OTF features `ss01`–`ss20` (see [33]).

There is, however, a flaw shared by both Computer Modern and Euler calligraphic alphabets and thus inherited by the Latin Modern Math: the lack of a corresponding lower case alphabet. The Unicode specification assigns slots to lower case mathematical calligraphic letters, implying that they are expected to be present in math fonts.

We could not find a calligraphic font with a proper license optically matching Euler or Computer Modern upper case calligraphic letters, and we gave up in advance the idea of designing the respective matching lower case letters ourselves. Instead, we began to consider the inclusion of yet another stylistic set, a “home-made” calligraphic font inspired by (but not based on) the original Computer Modern calligraphic script. Although we would never dare to make a text calligraphic font without close cooperation by a professional type designer, we took a chance and attempted to prepare a symbol calligraphic font for TG DejaVu Math.

There was an additional reason for doing our own calligraphic script. Anticipating future work (see Section 7 below), we looked for a calligraphic script matching a sans-serif font. Having not found any, we decided to prepare our own. Because our calligraphic alphabet for TG DejaVu Math was, of course, defined by a parametric MetaType 1 program, it was possible to prepare a sans-serif (linear) variant of the calligraphic script with reasonable effort. A tentative, experimental linear calligraphic alphabet is shown in Figure 9: the calligraphic script used in TG DejaVu Math (top) and its linear variant to be used in a sans-serif math font (bottom).

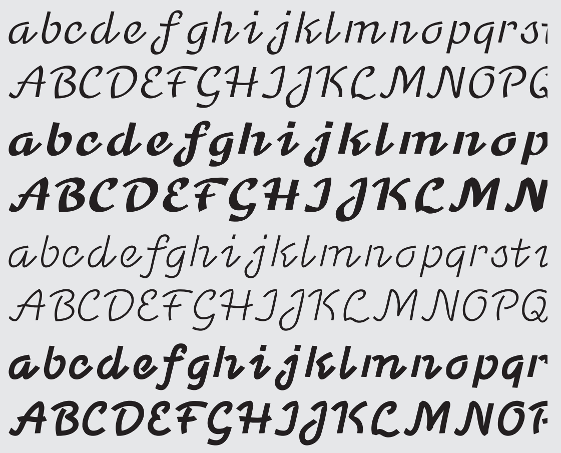


Figure 9

Fortunately, there is no issue of compatibility regarding the TG math fonts, as there are no predecessors. The only question is the similarity of repertoire between the LM and TG math fonts. As explained in Section 6.3, the repertoires are bound to differ slightly, for the usual legacy reasons.

For the same reasons, some technical details of the font structure also cannot be implemented similarly. Worthy of mentioning in this context is a peculiar difference between the LM and TG math fonts related to integrals; namely, the TG math fonts contain extensible integrals, which are definable within the OTF math format — but we are not aware of any typesetting engine that can take advantage of this possibility.

7 Plans for the future

7.1 Testing and maintenance

Tasks that are important today and will be forever important in the future are maintenance and testing. There is, of course, neither a single tool for testing nor a unique maintenance procedure. Each case demands a specific approach.

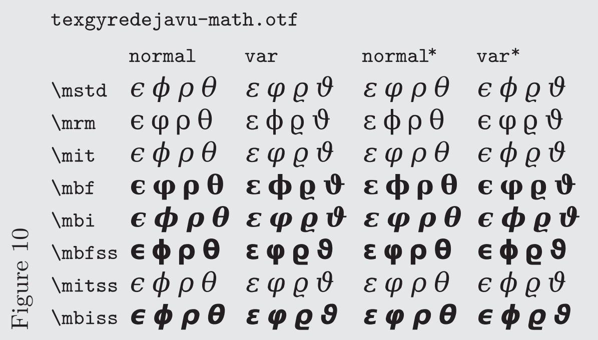
It is Piotr Pianowski who is responsible for testing fonts and preparing adequate tools. Tests refer to both the appearance of the fonts and their internal structure. In particular, the intermediate PostScript Type 1 code needs checking. The following example shows a case where the inspection of the code revealed an error in the `parenright.ex` procedure (describing an extender of the extensible right parenthesis, which should be just a rectangle), probably due to the wrong rounding procedure: the number ‘1’ means that the line drawn by the command `rlineto` is not strictly vertical. The left column contains the correct code for the corresponding component of the extensible left parenthesis:

```
/parenleft.ex {       /parenright.ex {
  143 609 hsbw          338 609 hsbw
  127 418 rmoveto       128 418 rmoveto
  -127 hlineto          -128 hlineto
  -418 vlineto          1 -418 rlineto
  127 hlineto           126 hlineto
  closepath             closepath
  endchar               endchar
  } ND                  } ND
```

It is next to impossible to perceive such a tiny deformity on a printout of glyph shapes, which does not mean that the bug should not be fixed.

The next example, Figure 10, shows one of the tests we use for checking a vexing problem regarding Greek letter names. Some Greek letters have a shape variant, and there is an unfortunate discrepancy between TEXies and the rest of the world (the rightmost two columns marked with asterisks) as to which glyphs are considered “normal” and which “variant”.

Almost every time we deal with the Greek alphabet, a mistake in variant names tries to creep in. Without tests of this kind we would be lost.



texgyredejavu-math.otf

	normal	var	normal*	var*
\mstd	ϵ ϕ ρ θ	ε φ ϱ ϑ	ε φ ρ θ	ϵ ϕ ϱ ϑ
\mrm	ϵ φ ρ θ	ε ϕ ϱ ϑ	ε ϕ ρ θ	ϵ φ ϱ ϑ
\mit	ϵ ϕ ρ θ	ε φ ϱ ϑ	ε φ ρ θ	ϵ ϕ ϱ ϑ
\mbf	ϵ φ ρ θ	ε ϕ ϱ ϑ	ε ϕ ρ θ	ϵ φ ϱ ϑ
\mbi	ϵ ϕ ρ θ	ε φ ϱ ϑ	ε φ ρ θ	ϵ ϕ ϱ ϑ
\mbfss	ϵ ϕ ρ θ	ε φ ϱ ϑ	ε φ ρ θ	ϵ ϕ ϱ ϑ
\mitss	ϵ ϕ ρ θ	ε φ ϱ ϑ	ε φ ρ θ	ϵ ϕ ϱ ϑ
\mbiss	ϵ ϕ ρ θ	ε φ ϱ ϑ	ε φ ρ θ	ϵ ϕ ϱ ϑ

Figure 10

The last example concerning maintenance and testing issues, Figure 11, shows a typical test of the structure of extensible characters, “embellished” horizontal arrows in this case: ‘vh 2’ means that there are 2 size variants, ‘ch 3’ and ‘ch 5’ — that there are 3 and 5 horizontal components of a given glyph, respectively. Tests of this kind are essential for maintaining uniformity across a font collection as well as inside a single font.

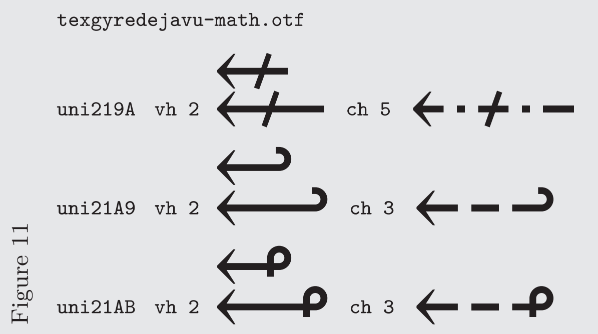


Figure 11

As a result of remarks to date from the users of our fonts, we gathered a list of recommended fixes and improvements — some trivial, like reports on malformed glyphs or wrongly assigned Unicode slots, some fairly difficult, like suggestions to implement anchors or math staircase kerns.

The latter two potential improvements are still pending, mainly because of vague specification and the question of practical application. Being unsure whether all relevant typesetting programs would be able to handle such improvements, we preferred to linger till the engines would become stable. It seems that the time is ripe to attempt these extensions, especially as the staircase kerns were ultimately implemented in XƎTEX in the middle of 2014.

We are also not sure which OTF features should be present in math fonts. At the moment, only math-oriented OTF features are implemented. Perhaps we should consider the inclusion of text-oriented OTF features too, like the mentioned `onum`, `lnum`, `pnum`, and `tnum` for switching between numeral variants, or stylistic sets for switching between calligraphic alphabets.

Also as suggested by users, we consider excerpting a number of glyphs, mostly geometrical shapes and arrows but also selected math operators and relational symbols, etc. (but excluding extensible and subscript characters), from math fonts and transferring them into the respective text fonts. Such glyphs, though prepared for math-oriented applications, can also be fruitfully used in conventional fonts, that is, those not equipped with the MATH table, for the typesetting of technical documents. This, obviously, requires a proper specification and careful selection of the glyphs in question. This is, in fact, one of the planned stages of work on new fonts.

Of course, MetaType 1 itself also requires maintenance: enhancing, modifying, fixing, etc. For example, we had to extend the otherwise stable set of MetaPost macros used in MetaType 1 in order to handle the math extension of the OTF specification.

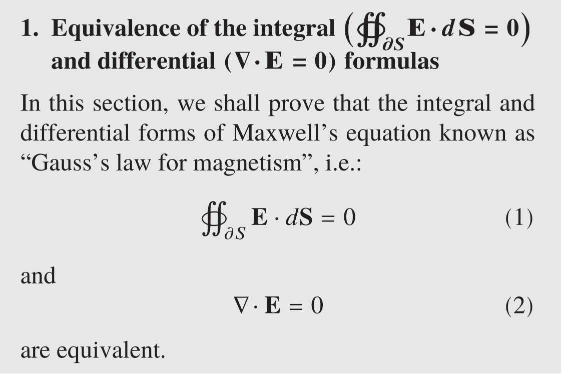
Commencing our works for the GUST e-foundry, we underestimated the inexorable Hofstadter's Law (see footnote 7 on page 324) which apparently applies not only to time resources. We believed that MetaType 1 could be kept simple: just MetaPost, a few trivial Gawk scripts and a stand-alone, stable converter to PostScript Type 1 fonts, T1utils, and that's all. In accordance with Hofstadter's Law, the task has unavoidably turned out to be much more complex than we expected and the implementation — too heterogeneous.

In order to remedy this, we intend to unify MetaType 1 by eliminating Gawk, Perl, T1utils, and, as we mentioned in Section 3.1, AFDKO. We aim at employing two basic "subengines", namely, MetaPost (for generating outlines) and Python (for arranging the MetaPost output) plus one external, possibly replaceable, "subengine", for now, the FontForge Python library (for generating OTF fonts and the theoretically obsolescent but still widely used PostScript Type 1 fonts).

7.2 More GUST e-foundry fonts

In the nearest future, we would like to elaborate and release three experimental math fonts, namely: bold math, sans-serif math, and monospaced math. These fonts fit neither the Microsoft nor Unicode specifications ([29] and [37], respectively). Therefore, we have to start by working out an altered specification, adjusted to our purposes, for these non-standard cases. For example, it is not at all clear what to do with the sans-serif alphabet in a sans-serif math font: should it be omitted, left intact, or modified somehow (how?).

Bold math fonts can be used in titles containing math formulas, as shown in Figure 12 (a tentative version of TG Termes Bold Math is used).



1. Equivalence of the integral $\left(\oiint_{\partial S}\mathbf{E}\cdot d\mathbf{S}=0\right)$ and differential $(\nabla\cdot\mathbf{E}=0)$ formulas

In this section, we shall prove that the integral and differential forms of Maxwell's equation known as "Gauss's law for magnetism", i.e.:

$$\oiint_{\partial S}\mathbf{E}\cdot d\mathbf{S}=0 \tag{1}$$

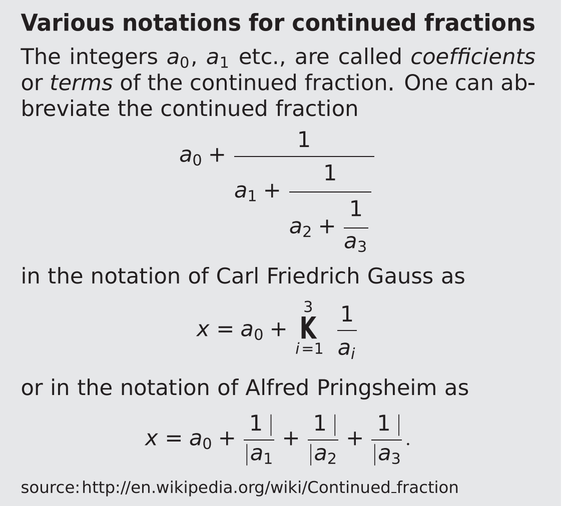
and

$$\nabla\cdot\mathbf{E}=0 \tag{2}$$

are equivalent.

Figure 12

Nowadays, many documents are being typeset in sans-serif, even schoolbooks. Computer presentations may serve as another example. For such applications sans-serif math seems plausible. An example of such an application is shown in Figure 13 (a tentative version of TG DejaVu Sans Math is used).



Various notations for continued fractions

The integers a_0, a_1 etc., are called *coefficients* or *terms* of the continued fraction. One can abbreviate the continued fraction

$$a_0+\cfrac{1}{a_1+\cfrac{1}{a_2+\cfrac{1}{a_3}}}$$

in the notation of Carl Friedrich Gauss as

$$x=a_0+\underset{i=1}{\overset{3}{\mathrm{K}}}\frac{1}{a_i}$$

or in the notation of Alfred Pringsheim as

$$x=a_0+\frac{1|}{|a_1}+\frac{1|}{|a_2}+\frac{1|}{|a_3}.$$

source: http://en.wikipedia.org/wiki/Continued_fraction

Figure 13

TeX users are accustomed to the use of control sequences for math-oriented glyphs such as `\infty`, `\sum` or `\pm`. Some of these glyphs can be accessed (typed in and displayed) directly in text editors, provided the font used by the editor contains the relevant glyphs. The lion's share of nominally math-oriented glyphs can also be prepared as monospaced glyphs, usable in text editors, provided they are assigned Unicode numbers. One can expect that it will improve the legibility of sources and, thereby, the efficiency of preparation of such documents — see Figure 14 (a tentative version of TG DejaVu Mono Math is used).

```
\section{Алгоритм де Кастельє}

Заданий многочлен Бернштейна $B$ ступеня $n$
з опорними точками $β₀,…,βₙ$
$$
  B(t) = ∑ⁿᵢ₌₀ βᵢb_{i,n}(t),
$$
де $b$ — базис многочлена Бернштейна, многочлен
в точці $t₀$ може бути визначений за допомогою
рекурентного співвідношення:
$$\eqalign{
  βᵢ⁽⁰⁾& ≔ βᵢ \mbox{, } i=0,…,n\cr
  βᵢ^{(j)}& ≔ βᵢ^{(j−1)}(1−t₀) + βᵢ₊₁^{(j−1)}t₀
     \mbox{, } i=0,…,n−j \mbox{, } j=1,…,n\cr
}$$
Тоді визначення $B$ в точці $t₀$ може бути
визначено в $n$ кроків алгоритму.
Результат $B(t₀)$ дано за:
$$
  B(t₀)=β₀⁽ⁿ⁾.
$$
Також, крива Безьє $B$ може бути розділена
в точці $t₀$ на дві криві з відповідними
опорними точками:
$$
  β₀⁽⁰⁾,β₀⁽¹⁾,…,β₀⁽ⁿ⁾
  β₀⁽ⁿ⁾,β₁⁽ⁿ⁻¹⁾,…,βₙ⁽⁰⁾
$$
```

source: https://uk.wikipedia.org/wiki/Алгоритм_де_Кастельє

Figure 14

Note that treating subscripts incoherently by the Unicode Standard (Section 6.4) may prove to be a significant impediment in the latter case, that is, for fonts without full subscript support. Perhaps the defining of the complete set of subscripts and superscripts (partially "unicodeless") and accessing them via stylistic set features (`ss01`–`ss20`) can be a solution, provided a given text editor handles the relevant OTF features.

7.3 Legal issues

The problem of copyrights and licenses has clung to us like a leech from the very beginning and still persists. We are not copyright experts and we do not want to be. Being sick of trouble with releasing our work due to discussions about legal aspects of distributing free fonts, we coined a pun *lice-sense*. Just one example: the release of TG DejaVu Math was delayed by about a year because of doubts raised concerning legal matters.

Having said this, we should emphasize that it does not mean that there has been no activity from the side of the GUST e-foundry regarding legal issues. On the contrary, our magnificent liaison officer, Jerzy Ludwichowski, has made Herculean efforts in order to provide appropriate licenses for GUST fonts. In particular, he prepared a proposal of a license for the URW fonts that would suit GUST e-foundry needs, managed to contact in person the URW++ managing director, Dr. Peter Rosenfeld, and, after long negotiations, received the gracious approval for the additional license.

All the fonts released so far are licensed under the GUST Font License (GFL; see [30]), except for TG DejaVu Math which has a somewhat complex license (see [9], the Manifest file for TG DejaVu Math). Recently, many fonts are being released under the SIL Open Font License (OFL; see [34]); therefore, we consider dual-licensing GUST fonts (GFL+OFL).

More details concerning legal matters relevant to GUST e-foundry fonts can be found in a series of publications by Jerzy Ludwichowski — see, for example [18, 19, 20, 21].

7.4 Constraints of tradition vs. our dreams

We are not particularly enthusiastic about font technology nowadays, but we are not going to spit into the wind, and try to make the best of what is available. This does not mean, however, that we are going to relinquish dreams of a successor to the currently prevalent "Knuth–Gutenberg" model of typesetting, that is, the model of stiff rectangles (types) arranged within a larger rectangle (page; a series of pages being called a document), deeply ingrained in Johannes Gutenberg's technology of movable type, and transferred by Donald E. Knuth, among others, to the realm of computers.

Computers facilitate some operations, such as kerning and ligatures, thereby accelerating work on documents. The ease of use of affine transformations is also considered an advantage of computers by many graphic designers. We are inclined to consider both these aspects as being of rather equivocal benefit. Leaving aside these philosophical questions and a far-reaching yet obvious idea, in fact also philosophical, that a font could be defined as a structured collection of general purpose object programs, we confine ourselves to pointing out two examples of aspects that might be improved within the framework of the contemporary edifice of typesetting.

Shrinkable and stretchable spaces, as in TEX, would supposedly be convenient also in OTF fonts and seem not too hard to implement. This simple problem touches on a fairly general and not in the least trivial problem which could be called "the conflict of competence": which "knowledge" should be implemented in a font and which — in a typesetting program.

Another improvement, this time hard to implement, is related to a naïve question: why must additional alphabets be embedded into a single math font? The answer is simple: operating systems are poorly designed with regard to font management. In particular, a user is not allowed to define a "private" family of fonts — the commonplace pattern "regular, regular italic, bold, bold italic" is very difficult to dislodge. One can imagine other variants of this idea, namely, borrowing components for extensible characters or kerns from several fonts. As far as we know, such an approach has not been implemented in any typesetting program or any operating system. TEX is no exception (unless virtual fonts are used); note, for example, that changing fonts within a word switches off the TEX hyphenation mechanism.

It should be emphasized that several such problems were tackled in LuaTEX by Hans Hagen and team; it does not seem, however, as if amendments of this kind are to be introduced worldwide in any near future. This is understandable, as the constraints of tradition and compatibility usually slow down the innovative processes. For example, recently announced advancements in the OTF format specification (see, e.g., [6]), can actually be considered a step backward towards a previously abandoned idea, temporarily as it turns out, of *multiple master fonts*. Anyway, this announcement augurs both ill and well for us: it means, on the one hand, that we will probably have to reimplement MetaType 1 in order to keep pace with the surrounding world and, on the other hand, that we still have something to do and something to think about.

8 Acknowledgements

We are indebted to all the people and TEX groups that have supported our font enterprises. Almost all the GUST e-foundry projects have been kindly supported by the Czechoslovak TEX user group CSTUG, the German-speaking TEX user group DANTE e.V., the Polish TEX Users Group GUST, the Dutch-speaking TEX user group NTG, TUG India, UK-TUG, and, last but not least, TUG. In a few cases, GUTenberg, the French-speaking TEX Users Group, supported us too.

We are very grateful to Karel Píška and Ulrik Vieth, who performed extensive tests of our fonts and inspired us with insightful comments, and to Marcin Woliński, who provided the indispensable LATEX support for our fonts.

The exceptional, personal thanks we owe to our friends who kept our spirits up for many years and tirelessly encouraged us to work on fonts: Hans Hagen, Johannes Küster, Jurek Ludwichowski, Volker RW Schaa, Jola Szelatyńska — hearty thanks!

References

Presentations, publications and packages

[1] Lars Engebretsen, *Almost European Fonts*
`https://ctan.org/pkg/ae`

[2] Michael Everson, *The Alphabets of Europe*
`http://www.evertype.com/alphabets/`

[3] Lee Hetherington, Eddie Kohler, *T1utils*
`http://www.lcdf.org/type/t1asm.1.html`
`http://www.lcdf.org/type/t1disasm.1.html`

[4] John D. Hobby, *MetaPost*
`https://ctan.org/pkg/metapost`

[5] Donald E. Knuth, *The TEXbook*, *TEX: The Program*, *The Metafontbook*, *Metafont: The Program*, *Computer Modern Typefaces*, Computers & Typesetting, vol. A–E, Addison-Wesley, Reading, Massachusetts, 1986

[6] John Hudson, *Introducing OpenType Variable Fonts*, 2016
`https://medium.com/@tiro/https-medium-com-tiro-introducing-opentype-variable-fonts-12ba6cd2369`

[7] Bogusław Jackowski, Janusz M. Nowacki, Piotr Strzelczyk, *Antykwa Półtawskiego: a parameterized outline font*, Proceedings of the EuroTEX Conference, Heidelberg, Germany, 1999
article: `http://www.staff.uni-giessen.de/partosch/eurotex99/jackowski`
download: `http://www.gust.org.pl/projects/e-foundry/poltawski`

[8] Bogusław Jackowski, Marek Ryćko, *Polish extension of Computer Modern fonts*
`https://ctan.org/pkg/pl-mf`

[9] Bogusław Jackowski, Piotr Strzelczyk, Piotr Pianowski, GUST e-foundry math fonts
The Latin Modern Math (LM Math) font:
`http://www.gust.org.pl/projects/e-foundry/lm-math`
The TEX Gyre (TG) Math Fonts:
`http://www.gust.org.pl/projects/e-foundry/tg-math`
`https://ctan.org/pkg/tex-gyre-math`

[10] Bogusław Jackowski, Piotr Strzelczyk, *How to make more than one math OpenType font or the Beasts of Fonts*, DANTE 2011 Meeting, Bremen, Germany, 2011
`http://www.gust.org.pl/projects/e-foundry/math/beasts05.pdf`

[11] Bogusław Jackowski, Janusz M. Nowacki, Piotr Strzelczyk, *The Latin Modern (LM) Family of Fonts*
`http://www.gust.org.pl/projects/e-foundry/latin-modern`

[12] Bogusław Jackowski, Janusz M. Nowacki, *Latin Modern fonts: how less means more* Proceedings of the EuroTEX conference, Pont-à-Mousson, France, 2005
`http://tug.org/TUGboat/tb27-0/jackowski.pdf`

[13] Bogusław Jackowski, Janusz M. Nowacki, Piotr Strzelczyk, *MetaType 1: a MetaPost-based engine for generating Type 1 fonts*, 2001
article: `http://www.ntg.nl/maps/26/15.pdf`
presentation: `http://ntg.nl/eurotex/JackowskiMT.pdf`
download: `https://ctan.org/pkg/metatype1`

[14] Bogusław Jackowski, Janusz M. Nowacki, Piotr Strzelczyk, *The TeX Gyre (TG) Collection of Fonts*
`http://www.gust.org.pl/projects/e-foundry/tex-gyre`

[15] Bogusław Jackowski, Janusz M. Nowacki, Piotr Strzelczyk, *TeX Gyre Pagella Math or Misfortunes of Math Typographer*, BachoTeX XX, Bachotek, Poland, 2012
`http://www.gust.org.pl/projects/e-foundry/math/misfortunes02.pdf`

[16] Jörg Knappen, Norbert Schwarz, *European Computer Modern Fonts*
`https://ctan.org/pkg/ec`

[17] Donald E. Knuth, *Metafont and MetaPost logo fonts.* `https://ctan.org/pkg/mflogo-font`

[18] Jerzy B. Ludwichowski, *GUST font licenses*, BachoTeX XIV, Bachotek, Poland, 2006
`http://tug.org/fonts/licenses/gfl.pdf`

[19] Jerzy B. Ludwichowski, Karl Berry, *GUST Font License: An application of the LaTeX Project Public License*, XVII European TeX Conference and BachoTeX XV, Bachotek, Poland, 2007; *TUGboat*, Volume 29 (2008), No. 1
`http://www.gust.org.pl/projects/e-foundry/licenses/tb91Berry_Ludwichowski.pdf`

[20] Jerzy B. Ludwichowski, *Licensing of the TeX Gyre family of fonts*, XIX European TeX Conference and 3rd International ConTeXt User Meeting, The Hague, The Netherlands, 2009
`https://www.ntg.nl/EuroTeX/2009/slides/jerzy-slides.pdf`

[21] Jerzy B. Ludwichowski, *Is there life besides licensing?*, DANTE e.V. General Meeting, Bremen, Germany, 2011
`https://www.dante.de/events/Archiv/dante2011/programm/vortraege/folien-jl.pdf`

[22] Janusz M. Nowacki, *Polish fonts*
`http://jmn.pl/en/`

[23] Piotr Strzelczyk, *Standard Unicode w typografii*, Acta Poligraphica nr 1/2013 (in Polish; to be published also in English under the title *Standard Unicode in typography*)
`http://www.cobrpp.com.pl/actapoligraphica/uploads/pdf/AP2013_01_Strzelczyk.pdf`
see also: Piotr Strzelczyk, *(uni)coding of math fonts*, BachoTeX XIX, Bachotek, Poland, 2011
`http://www.gust.org.pl/bachotex/2011-en/presentations/Strzelczyk_1_2011`

[24] Ulrik Vieth, *OpenType math illuminated*, *TUGboat*, Volume 30 (2009), No. 1
`http://tug.org/TUGboat/tb30-1/tb94vieth.pdf`

[25] George Williams and FontForge project contributors, *FontForge*
`http://fontforge.github.io/en-US/`

General purpose documentation:

[26] *Adobe Font Development Kit for OpenType*
`http://adobe.com/devnet/opentype/afdko.html`

[27] *Adobe base 35 fonts*
Adobe original AFM files: `ftp://ftp.adobe.com/pub/adobe/type/win/all/afmfiles/base35/`
URW replacement: `https://ctan.org/pkg/urw-base35`

[28] *Adobe Type 1 Font Format*
`https://partners.adobe.com/public/developer/en/font/T1_SPEC.PDF`

[29] *MATH — The mathematical typesetting table*
`https://www.microsoft.com/typography/OTSPEC/math.htm`

[30] *GUST Font License*
`http://www.gust.org.pl/fonts/licenses/GUST-FONT-LICENSE.txt`
`http://tug.org/fonts/licenses/GUST-FONT-LICENSE.txt`

[31] *OpenType specification (full)*
`http://www.microsoft.com/en-ph/download/details.aspx?id=1144`

[32] *OpenType Feature File Specification*
`http://www.adobe.com/devnet/opentype/afdko/topic_feature_file_syntax.html`

[33] *Registered features — definitions and implementations (p–t)*
`https://www.microsoft.com/typography/otspec/featuretags.htm`

[34] *SIL Open Font License*
`https://scripts.sil.org/OFL`

[35] *The Unicode Standard 9.0.0, 2016*
`http://unicode.org/versions/Unicode9.0.0`

[36] *The Unicode Standard: A Technical Introduction*
`http://unicode.org/standard/principles.html`

[37] Barbara Beeton, Asmus Freytag, Murray Sargent III, *Unicode Technical Report #25. Unicode Support for Mathematics*
`http://unicode.org/reports/tr25/`

All links above were tentatively accessed 05.07.2015.

◇ Bogusław Jackowski
Gdańsk, Poland
`b_jackowski (at) gust dot org dot pl`

◇ Piotr Strzelczyk
Sopot, Poland
`p.strzelczyk (at) gust dot org dot pl`

◇ Piotr Pianowski
Trąbki Wielkie, Poland
`p.pianowski (at) gust dot org dot pl`