## Parametric math symbol fonts

Bogusław Jackowski, Piotr Strzelczyk and Piotr Pianowski

### 1 Introduction

In 2007, Microsoft released their math-equipped MS Office along with the math OpenType (OTF) font Cambria. In the past 10 years, a dozen more OTF math fonts have been released — half of which were developed by the GUST e-foundry [4, p. 908].

Given the huge number of font vendors (see, e.g., [2]) and the correspondingly huge number of offered fonts, the nearly negligible number of math OTF fonts is somewhat puzzling. Leaving aside the reasons for such a state of the art, one conclusion seems obvious: math OTF fonts, despite having a well-defined standard which is undoubtedly an important advantage, are not particularly popular.

Thus, the question arises: is concentrating efforts on generating more math fonts reasonable? As far as the TEX society is considered, the answer is equivocal: yes and no. Certainly, TEXies are interested in typesetting math texts, as TEX is still the best tool for this purpose, therefore they would gladly use a broad variety of math fonts. However, TEXies do not actually need complete OTF math fonts. Thanks to new TEX engines, notably LuaTEX, math fonts can be assembled out of already existing text fonts and a "math trunk" — a set of math symbols from another font.

Below we present the idea of assembling math fonts on the fly using the LuaTEX engine. We will try to justify that this approach is less laborious than the making of a regular math font, yet general enough for TEX users.

### 2 What is a math font?

The contents of an OTF (also called Unicode) math font is specified by Microsoft documentation [9], and the Unicode Consortium report on Unicode support for mathematics [12]. The former specifies a special MATH table, a pivotal table for math OTF fonts. It contains information about glyph chains, stretchable glyphs, positioning of subscripts and superscripts, fractions, etc. The latter defines component alphabet sets (scripts) that are expected to be present in a math OTF font. The required components of a typical math OTF font are schematically shown in Figure 1.


Figure 1

As one can see, a math OTF font is, in fact, a collection of various fonts assembled into one entity. One of the reasons, the most important in our opinion, behind this arrangement is that nowadays operating systems do not enable flexible handling of user-defined families (collections) of fonts — formatting editors usually handle 4-member families comprising regular, regular italic, bold and bold italic variants. TEX users, however, are not bound to follow that restriction. The solution proposed in this paper follows from this observation.

### 3 Subscripts and superscripts

Subscripts and superscripts (by tradition, of the 1$^{\text{st}}$ and 2$^{\text{nd}}$ order) are obligatory for typesetting math; therefore, math fonts are expected to contain special glyphs which can be used for this purpose, also used in fractions and as root degree in radicals; for the sake of brevity, we'll call these glyphs *pars pro toto* subscripts. They are accessed by the OTF feature mechanism, more precisely by the math extension feature `ssty` [10, 11].

Neither the Microsoft documentation nor the Unicode Consortium report ([9] and [12]) mentioned above specify which glyphs should be accompanied by subscripts; in the GUST e-foundry fonts, we have tried to limit their number, nevertheless, they make up about 30 percent of all glyphs.

Computer Modern, optical (fancy) scaling:

abc abc abc

AMS Euler, non-uniform scaling:

abc abc abc

TeX Gyre, non-uniform scaling:

abc abc abc

default, uniform scaling:

abc abc abc

Figure 2

The outlines of the subscripts have the normal size. The MATH table contains the scaling coefficient by which the rendering engine is expected to scale down the glyph uniformly before the placement. If the respective subscript glyph is absent from the font, the original glyph is used.

A pity that only uniform scaling is allowed. If non-uniform scaling were allowed, the extra glyphs for subscripts would be unnecessary.

Donald E. Knuth decided to scale all fonts non-linearly (typographers prefer the term "optically") in his Computer Modern family of fonts [1]: corresponding glyphs from fonts having different design sizes have different proportions, thus including the fonts having the design size 7 pt and 5 pt, used as subscripts of the 1st and 2nd order, respectively.

It turns out, however, that fairly decent visual results can be achieved by non-uniform scaling, i.e., by scaling down and widening the glyph at the same time. This approach was successfully used in the renowned Euler font [6], designed by Hermann Zapf, belonging to the basic collection of TeX fonts. We are not aware of any complaints about the inelegance or illegibility of Euler subscripts. Following the Euler project, we have employed the same method in our GUST e-foundry fonts. The appearance of subscripts generated with various methods is shown in Figure 2.

The point is that the non-uniform scaling can be done by the modern TeX engines on the fly; therefore, the presence of subscript glyphs in a math font is not essential for TeX users.

## 4 LuaTeX as a "font assembler"

In order to assemble several component (sub)fonts into a single math OTF font, advanced software is generally needed, such as, e.g., the excellent Font-Forge editor [7]. TeX wizards, however, or more precisely LuaTeX wizards, are in a better position — they do not need a font editor at all. Font assembly can be programmed in TeX (using the package `unicode-math` [5]). Once a wizard devises (and releases) a relevant script, others can adapt it to their needs. This is exactly our case: starting from LuaTeX code for loading OTF fonts, we were able to prepare a script for "blending" a given math OTF font with a few other selected fonts. Figure 3 shows an example, the beginning of our LuaTeX script for

```
\input otf-math
\OMtrm{DejaVuSans.ttf}
\OMtbf{DejaVuSans-Bold.ttf}
\OMtit{DejaVuSans-Oblique.ttf}
\OMtbi{DejaVuSans-BoldOblique.ttf}
\OMmat{texgyredejavu-math-1106.otf}
\OMsize{12pt} % fix size
\OMfix % fix 12-point "blended font"
```

Figure 3

```
$$ \delta_{ij}=\cases{
0& for \ $i=j$\cr
1& for \ $i\ne j$\cr
}\;=\; \left(\matrix{
1& 0& 0\cr
0& 1& 0\cr
0& 0& 1\cr
}\right) $$
```

$$\delta_{ij}=\begin{cases}0 & \text{for } i=j\\ 1 & \text{for } i\ne j\end{cases} \;=\; \begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix}$$

Figure 3a

```
\def\frac#1#2{{#1\over#2}}
$$ \sqrt{\frac{1}{2}}
\approx 0{,}7\qquad
f\,'\!(x)=
\left(\frac{1}{x^2}\right) $$
```

$$\sqrt{\frac{1}{2}}\approx 0{,}7\qquad f\,'\!(x)=\left(\frac{1}{x^2}\right)$$

Figure 3b

```
$$ \left\langle\left\langle
\left\langle\left\langle
  x
\right\rangle^2\right\rangle^2
\right\rangle^2\right\rangle
\quad
\left(\left(\left(
\left(\left(\left(\left(
  x
\right)^x\right)^x\right)^x
\right)^x\right)^x\right)^x\right) $$
```

$$\left\langle\left\langle\left\langle\left\langle x\right\rangle^2\right\rangle^2\right\rangle^2\right\rangle\quad\left(\left(\left(\left(\left(\left(\left(x\right)^x\right)^x\right)^x\right)^x\right)^x\right)^x\right)$$

Figure 3c

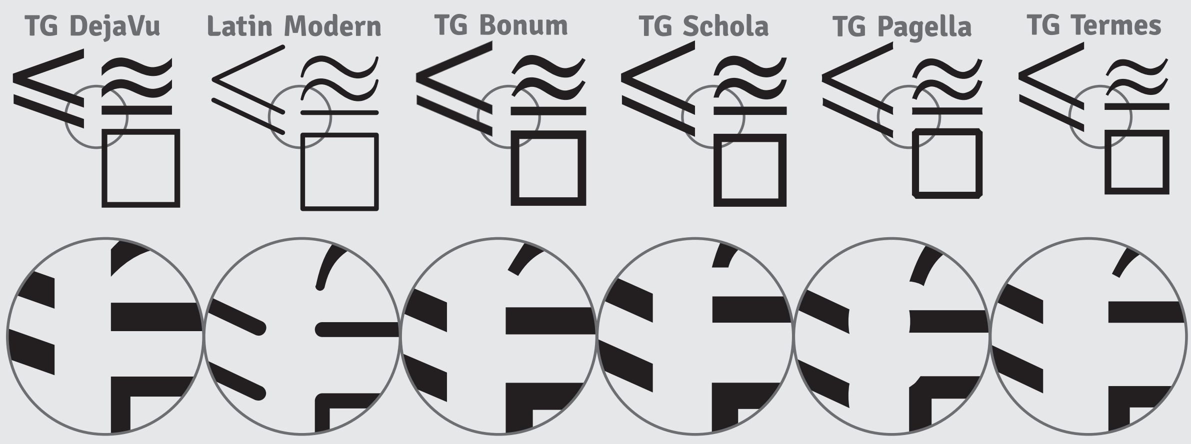


Figure 4

mixing the TEX Gyre DejaVu OTF math font with the DejaVu sans-serif variant.

Such a relatively simple header allows even inexperienced users to easily type math formulas with a chosen main font (in general, an arbitrary text font, DejaVu sans-serif in this case) along with the math symbols, i.e., braces, radicals, etc., taken from a chosen math font (in general, a math OTF font, TG DejaVu Math in this case) — see Figures 3a–3c.

Not only subscript sizes and proportions can be defined on the fly; also sidebearings can be controlled by appropriate font family definitions using the LuaTEX font loading option `extend` and the LuaTEX (originally from pdfTEX) primitive command `\letterspacefont`, respectively.

## 5 What else do we need?

In the previous section we substantiated the statement that LuaTEX can be used, in a sense, as a "poor man's font editor". What cannot be easily handled from within LuaTEX? The answer is: subtle details should be taken into account, provided that one cares — we do.

As we emphasized in our paper on the GUST e-foundry font projects [4, p. 326], an important aspect of a math font is the visual harmonizing of the alphanumeric glyphs and the symbols. Seemingly trivial glyphs, such as operator and relational symbols, may serve as a convenient example: they have slightly different shapes in each of our math fonts — see Figure 4 above. Another example is the optical similarity between the shape of integrals and the letter 'long s', which in turn is similar to the letter 'f' [4, p. 326].

Such details, in principle, could be controlled from within LuaTEX; however, we would consider this to be overloading the functionality of LuaTEX. Furthermore, we prefer to fiddle around with glyph shapes using MetaType 1 [3], our favorite MetaPost-based tool.

## 6 How to tackle the problem?

We can pinpoint the problem to solve as follows: *given (say, by a customer) a font, add an adequate, i.e., optically consistent, math companion to be used in LuaTEX with the given font.* The solution consists of a few more or less obvious steps:

- prepare a generic set of LuaTEX macros;
- prepare a generic set of MetaPost/MetaType 1 macros for generating the basic set of math symbol glyphs;
- for this set of MetaPost/MetaType 1 macros, prepare a set of relevant parameters for a given font controlling ovalness, incisions, thickness of stems, x-height, etc.

The good news is that all the steps listed above are to a great extent accomplished or at least commenced:

- we use LuaTEX with the `unicode-math` package [5, 8], in our office (heavily exploiting Hans Hagen's font handling macros — thanks!);
- a lion's share of MetaType 1 macros which we use for generating GUST e-foundry fonts can also be used for this purpose;
- moreover, the MetaType 1 macros are, of course, parameterized — this is why we were able to release a new math OTF font once a year on average.

Our experience is thus optimistic, although it does not mean that nothing remains to be done. On the contrary. Putting it figuratively: it takes a few minutes to saw a plank, burnishing it takes a few hours. So far, we "have sawn the plank".

## 7 Conclusions

A canonical math OTF font has many advantages, such as, e.g., universality — it can be used with various programs and various operating systems. At the same time, it is a "frozen" (unmodifiable) object — it is impossible to modify it without employing a font editor; e.g., none of the subfonts can be replaced with a user-chosen variant.

The method described in this paper is, on one hand, certainly less universal as it is restricted to the TeX environment, but, on the other hand, provides a flexible tool that may prove useful (we hope) in practical applications.

Our thinking about implementing such an approach was triggered by customers' demands, who (rarely, but still) wanted to have math formulas typeset with their "flagship" font; unfailingly, it was none of the dozen math fonts mentioned in Section 1. Needless to say, the making of a respective complete math OTF font was not feasible.

Thus, we have a natural motivation to continue the work on this subject. We believe that before long we will be able to notify the TeX community about some results.

## 8 Acknowledgements

Permanent and hearty thanks go to Hans Hagen for providing us with a marvelous pastime.



## References

Presentations, publications and packages:

[1] Donald E. Knuth, *Computer Modern Typefaces*, Computers & Typesetting, vol. E, Addison-Wesley, Reading, Massachusetts, 1986

[2] Luc Devroye, *Font vendors*, `http://luc.devroye.org/vendors.html`

[3] 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://www.gust.org.pl/bachotex/2015/presentations/2015BJackowski.pdf`
Download: `https://ctan.org/pkg/metatype1`

[4] Bogusław Jackowski, Piotr Strzelczyk, Piotr Pianowski, *GUST e-foundry font projects*, *TUGboat*, Vol. 37 (2016), No. 3, pp. 317–336, `https://tug.org/TUGboat/tb37-3/tb117jackowski.pdf`

[5] Will Robertson, Philipp Stephani, Khaled Hosny, *Experimental Unicode mathematical typesetting: The unicode-math package*, ver. 0.8d, 2017 `https://ctan.org/pkg/unicode-math`

General purpose documentation:

[6] *AMS Euler typeface*, `https://en.wikipedia.org/wiki/AMS_Euler`

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

[8] *LuaTeX Reference Manual*, ver. 1.0.4, 2017 `http://www.luatex.org/svn/trunk/manual/luatex.pdf`

[9] *MATH — The mathematical typesetting table*, updated 2017 `https://www.microsoft.com/typography/OTSPEC/math.htm`

[10] *OpenType specification*, ver. 1.8.1, updated 2017 `https://www.microsoft.com/en-us/Typography/OpenTypeSpecification.aspx`

[11] *Registered features — definitions and implementations (p–t)*, updated 2017 `https://www.microsoft.com/typography/otspec/features_pt.htm`

[12] *Unicode Technical Report #25. Unicode Support for Mathematics*, revision 15, 2017 — Barbara Beeton, Asmus Freytag, Murray Sargent III `http://unicode.org/reports/tr25/`

All links were tentatively accessed 2017-06-09.

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

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

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