Experiences Typesetting OpenType Math with LuaLaTeX and XeLaTeX

Zkušenosti se sazbou matematiky ve formátu OpenType math v LuaLaTeXu a XeLaTeXu

Ulrik Vieth

Abstract: When LuaTeX first provided support for OpenType math typesetting in version 0.40, high-level macro support for math typesetting was first developed for ConTeXt MkIV, while support for LuaLaTeX was initially limited to a very low-level or non-existent. In the meantime, this gap has been closed by recent developments on macro packages such as luaotfload, fontspec, and unicode-math, so LaTeX users are now provided with a unified high-level font selection interface for text and math fonts that can be used equally well with both LuaLaTeX and XeLaTeX. While a unified high-level interface greatly improves document interchange and eases transitions between systems, it does not guarantee that identical input will always produce identical output on different engines, as there are significant differences in the underlying implementations of math typesetting algorithms. While LuaTeX provides a full-featured implementation of OpenType math, XeTeX has taken a more limited approach based on a subset of OpenType parameters to provide the functionality of traditional TeX engines.

Given the possibility of running exactly the same test files on both engines, it now becomes feasible to study those differences in detail and to compare the results. Hopefully, this will allow to draw conclusions how the quality of math typesetting is affected and could be improved by taking advantage of a more sophisticated, full-featured OpenType math implementation.

Key words: LuaLaTeX, XeLaTeX, OpenType math, math typesetting, fontspec package, Cambria, Asana, XITS, Neo Euler.

Abstrakt: Jelikož LuaTeX podporuje Open Type math až od verze 0.40, byla podpora matematické sazby na vyšší úrovni vytvořena nejprve pro ConTeXt MkIV, zatímco podpora pro LuaLaTeX byla nízká nebo nebyla žádná. Další vývoj však tuto mezeru zacelil – uživatelé LaTeXu mají nyní k dispozici jednotné rozhraní pro připojení fontů pro běžný text i pro matematickou sazbu pomocí balíčků luaotfload, fontspec a unicode-math;

 doi: 10.5300/2011-2-4/116

obojí lze celkem stejně dobře využít v LuaLaTeXu i v XeLaTeXu. I když toto jednotné rozhraní značně zjednodušuje výměnu dokumentů i přenos mezi různými systémy, nezaručuje, že tentýž vstup vytvoří vždy tentýž výstup na různých počítačích kvůli významným odlišnostem v implementaci algoritmů pro matematickou sazbu. Zatímco LuaTeX poskytuje úplnou implementaci všech vlastností OpenType math, XeTeX převzal jen část z nich s ohledem na tradiční implementace TeXu.

Maje možnost překládat stejné testovací soubory v obou implementacích, bylo možné podrobně zkoumat jejich rozdíly a porovnat výslednou matematickou sazbu. Doufejme, že toto přispěje k zjištěním, co ovlivňuje kvalitu matematické sazby a jak ji zlepšit implementací výhod kompletního formátu OpenType math.

Klíčová slova: LuaLaTeX, XeLaTeX, formát OpenType math, sazba matematiky, balíček fontspec, Cambria, Asana, XITS, Neo Euler.

LuaTeX math

$$\Delta \boldsymbol{E} - \frac{1}{c^2}\frac{\partial^2 \boldsymbol{E}}{\partial t^2} = \frac{1}{\varepsilon_0}\boldsymbol{\nabla}\lambda + \mu_0\frac{\partial \boldsymbol{j}}{\partial t}\,,$$

$$\Delta \boldsymbol{B} - \frac{1}{c^2}\frac{\partial^2 \boldsymbol{B}}{\partial t^2} = -\mu_0 \operatorname{rot} \boldsymbol{j}\,.$$

```
% !TeX program = lualatex
\documentclass[fleqn]{article}
\usepackage{fontspec,unicode-math}
\setromanfont{Cambria}
\setmathfont{Cambria Math}
\begin{document}
  \input{luatex-fixes}
  \input{math-test}
\end{document}
```

XeTeX math

$$\Delta \boldsymbol{E} - \frac{1}{c^2}\frac{\partial^2 \boldsymbol{E}}{\partial t^2} = \frac{1}{\varepsilon_0}\boldsymbol{\nabla}\lambda + \mu_0\frac{\partial \boldsymbol{j}}{\partial t}\,,$$

$$\Delta \boldsymbol{B} - \frac{1}{c^2}\frac{\partial^2 \boldsymbol{B}}{\partial t^2} = -\mu_0 \operatorname{rot} \boldsymbol{j}\,.$$

```
% !TeX program = xelatex
\documentclass[fleqn]{article}
\usepackage{fontspec,unicode-math}
\setromanfont{Cambria}
\setmathfont{Cambria Math}
\begin{document}
  \input{xetex-fixes}
  \input{math-test}
\end{document}
```

Can you spot the difference?

$$\Delta \boldsymbol{E} - \frac{1}{c^2}\frac{\partial^2 \boldsymbol{E}}{\partial t^2} = \frac{1}{\varepsilon_0}\boldsymbol{\nabla}\lambda + \mu_0\frac{\partial \boldsymbol{j}}{\partial t}\,,$$

$$\Delta \boldsymbol{B} - \frac{1}{c^2}\frac{\partial^2 \boldsymbol{B}}{\partial t^2} = -\mu_0 \operatorname{rot} \boldsymbol{j}\,.$$

```
% !TeX program = pdflatex
\documentclass{article}
\usepackage{pdfpages}
\begin{document}
\includepdfmerge
   [nup=2x1,noautoscale=true,delta=-21cm 0]
   {xelatex-test.pdf,1,lualatex-test.pdf,1}
\end{document}
```

Introduction

In this paper, we will review the state of recent developments of new TEX engines and corresponding macro packages to support math typesetting with Unicode and OpenType math fonts, based on our experiences from testing the TEX Live 2010 pretest distribution [1].

In the first part, we will summarize the available choices of TEX engines and macro packages, as well as the available choices of math fonts, which can be used for testing OpenType math typesetting.

In the second part, we will report our experiences testing the various TEX engines, macro packages and fonts, and we will report our findings which kind of problems were encountered and how these problems were resolved or circumvented.

In the third part, we will analyze and compare the results of running identical documents through different TEX engines using different implementations of math typesetting algorithms, and we will try to draw conclusions how the quality of math typesetting is affected and could be improved.

Reviewing the state of OpenType math support in TEX Live 2010

In recent years, many developments related to new TEX engines and corresponding macro packages and fonts have focused on providing support for Unicode and OpenType, not just for text typesetting, but also for math typesetting (which is still important as one of the traditional strong-holds of TEX).

Unicode and OpenType math technology

While the pre-history of Unicode text support dates back to the mid-1990s, most activities related to Unicode math support only became possible during the last few years, after a suitable font technology was developed as an extensions to the OpenType font format.

When the efforts to bring math into Unicode were started in the late 1990s by a consortium of scientific publishers, the original focus was on identifying math symbols and getting them accepted into Unicode [2, 3]. Once this was done, the focus shifted to developing a reference font implementation, the so-called STIX fonts, to provide the necessary glyph shapes [4].

While the STIX project was spending nearly a decade waiting for fonts to be designed, the lack of a suitable font technology for math typesetting was overlooked for a long time. While traditional TEX font formats supported math typesetting in their own way, they suffered from limitations and questionable design decisions [5]. On the other hand, mainstream font formats such as OpenType did not provide any support for the semantics of math typesetting.

Ironically, the lack of a suitable math font technology was only resolved when Microsoft started to develop support for MS Office 2007. Given their influence as a vendor controlling the OpenType font specification [6], they simply went ahead and created an extension of the OpenType font format containing a MATH table [7] and commissioned the design of Cambria Math as a reference implementation of an OpenType math font [8, 9]. In addition, they also developed a simplified math input language known as 'linear math' [10].

While there are sometimes strong reservations about accepting a vendor-defined file format, especially an unreleased one coming from Microsoft, developers of font tools such as FontForge [11] as well as developers of TEX engines were willing to accept OpenType math as a *de facto* standard, because it filled a need and also because it turned out to be well-designed.

Upon closer analysis, many concepts of OpenType math could be seen as obvious extensions or generalizations of traditional concepts of math typesetting in TEX [12]. Moreover, most OpenType math font parameters could be identified to have a direct correspondence to TEX math font parameters [13], which had also been carefully analyzed in previous studies [14, 15].

OpenType math support in TEX engines

When XeTEX first added OpenType math support in version 0.997 as of 2007 [16], it kept TEX's traditional math typesetting algorithms essentially unchanged and only used a small subset of OpenType font parameters to initialize the required TEX font parameters.

When LuaTEX also added OpenType math support in version 0.40 as of 2009 [17, 18, 19], it introduced a number of extensions and generalizations to TEX's math typesetting algorithms, aiming to provide a full-featured implementation of OpenType math.

As of TEX Live 2010, both new TEX engines supporting Unicode and OpenType math typesetting have been added to the mainstream distributions and have become widely available for using and testing the new features. However, their acceptance also depends on providing adequate macro package and font support.

OpenType math support in macro packages

When XeTEX was first added to TEX distributions, it was easily accessible to LaTEX users with XeLaTEX. A high-level font selection interface for text fonts was developed with the `fontspec` package [20, 21], which became widely used as a standard package for XeLaTEX.

When XeTeX added math support, a corresponding high-level font selection interface for math fonts was also developed with the `unicode-math` package [22], but unlike `fontspec` it wasn't released until recently.

When LuaTeX added math support, high-level macro support was initially developed for ConTeXt MkIV [23], while macro support for LuaLaTeX (or Plain LuaTeX) was initially limited to the `luaotfload` package [24], which provided only a low-level interface.

As of TeX Live 2010, LaTeX macro package support for Unicode and OpenType math typesetting has been much improved, as both the `fontspec` and `unicode-math` packages have undergone a complete rewrite, adding support for LuaLaTeX (based on `luaotfload`) to the code originally developed for XeLaTeX.

As a result, LaTeX users are now provided with a unified high-level font selection interface for text and math fonts [25] that can be used equally well with both XeLaTeX and LuaLaTeX.

Given this interface, selecting a different math font (such as Cambria Math) can be as easy as this:

```
\usepackage{fontspec,unicode-math}
\setmainfont[<options>]{Cambria}
\setmathfont[<options>]{Cambria Math}
```

Using the options of `\setmathfont`, a number of details of math typesetting can be easily configured, including the behavior of math alphabets (such as upright vs. italic for normal and bold, uppercase and lowercase, Latin and Greek), which will make it much easier to support the specific requirements of math typesetting in various fields of sciences [26].

OpenType math fonts

Regardless of font technology, developing math fonts has always been far from easy and choices of math fonts have always been severely limited. In this respect, the situation of OpenType fonts today is not much different from the situation of PostScript fonts in the 1990s. While there are countless choices of text fonts, there are only very few math fonts available, and even fewer of them are freely available.

As of mid-2010, we have the following choices of OpenType math fonts at our disposal:

- Cambria Math [8], the original reference math font, commissioned by Microsoft for Office 2007,
- Asana Math [27], a Palatino-like math font derived from a repackaging of the `mathpazo` fonts,
- XITS Math [28], a Times-like math font derived from a repackaging of the STIX fonts [29],
- Neo Euler [30], an upright math font derived from Hermann Zapf's redesign of AMS Euler fonts [31].

Except for Cambria Math, all of these fonts all are freely available, either from CTAN (if already released) or from GitHub (if still under development).

As of TeX Live 2010, Asana Math and XITS Math are both included in the distribution, but Cambria Math and Neo Euler have to be obtained separately and need to be installed manually in your `texmf-local` tree.

Once installed, each of the fonts can be used in the same way, but the range of symbols and math alphabets available may differ significantly between fonts.

Experiences testing OpenType math support in TeX Live 2010

Testing the functionality and quality of OpenType math typesetting implies testing a complex system, consisting of typesetting engines, macro packages and fonts (with embedded intelligence), which have to interact properly to produce the desired results.

Given the inherent complexity, there are a large number of problems which can occur, and most likely will occur, so we have to consider the possibilities of engine problems, macro problems, font problems and font loading issues.

Problems with TeX engines

Problems with TeX engines can be of several different kinds, ranging from fatal ones (such as unexpected crashes or malfunctions) to more subtle ones (such as hidden bugs in the math typesetting algorithms producing incorrect results).

Traditionally, TeX engines have enjoyed a reputation of being extremely robust and totally free of bugs. Unfortunately, such expectations no longer hold true when it comes to new TeX engines, which are under active development and which don't have the luxury of two decades of time to eliminate all possible bugs.

Engine problems of the fatal kind are therefore a very real possibility, which may even prevent or delay further testing until the problems can be resolved.

While testing the TeX Live 2010 pretest distribution, a number of problems were encountered for XeTeX on some 64-bit Linux platforms, resulting in crashes or malfunctions upon loading OpenType math fonts, which are likely to be caused by incompatibilities with external library dependencies.

Unfortunately, there was not enough time to debug these problems before the deadline for the TeX Live 2010 binaries, so the problems remain unresolved for now and need to be revisited eventually. As a workaround, it may be possible to use 32-bit binaries on 64-bit Linux platforms, which do not exhibit such problems.

$$\gamma^\alpha\left(\frac{\hbar}{\mathrm{i}}\,\partial_\alpha - qA_\alpha\right)\psi + m_0 c\,\psi = 0\,,$$

$$\gamma^\alpha\left(\frac{\hbar}{\mathrm{i}}\,\partial_\alpha - qA_\alpha\right)\psi + m_0 c\,\psi = 0\,.$$

Figure 1: Comparison of the size of delimiters in Asana Math for LuaTEX 0.60.x and LuaTEX 0.61. Due to a bug in the math typesetting algorithms, the extensible version of delimiters was applied too soon. (The example shows the Dirac equation from relativistic quantum mechanics.)

$$\int_F \varepsilon_0 \boldsymbol{E}\cdot\mathrm{d}\boldsymbol{f} = \int_V \lambda\,\mathrm{d}V, \quad \int_F \boldsymbol{B}\cdot\mathrm{d}\boldsymbol{f} = 0\,,$$

$$\int_F \varepsilon_0 \boldsymbol{E}\cdot\mathrm{d}\boldsymbol{f} = \int_V \lambda\,\mathrm{d}V, \quad \int_F \boldsymbol{B}\cdot\mathrm{d}\boldsymbol{f} = 0\,.$$

Figure 2: Comparison of the size of displaystyle operators in Cambria Math for LuaTEX with incorrect parameter values of `DisplayOperatorMinHeight` and with a correction applied at the macro level. (The example shows the integral form of the Maxwell equations from electrodynamics.)

Engine problems of the more subtle kind were found in LuaTEX, when a bug was discovered for some math fonts (such as Asana Math), which resulted in applying the extensible version of delimiters before exhausting all available sizes of big delimiters. (An example of the incorrect behavior is illustrated in Figure 1.)

In the meantime, this bug has already been fixed in LuaTEX 0.61, but again it was too late to include the fix in TEX Live 2010 which still uses LuaTEX 0.60.x.

Problems with OpenType font metrics

Problems with OpenType fonts can also be of different kinds, ranging from fatal ones (such as containing malformed data structures causing TEX engines to crash) to more subtle ones (such as providing incorrect values of font metric parameters causing TEX engines to produce incorrect results). In addition, font problems can also include encoding issues or incorrect glyph shapes.

Font problems of the fatal kind did not occur while testing OpenType math fonts, but a similar kind of problem was recently reported for some OpenType text fonts. The problem was quickly addressed with a fix in LuaTEX 0.61 to make the font parsing algorithms more robust about handling unexpected values.

Font problems of the more subtle kind were found with incorrect parameter settings in the MATH table of Cambria Math and Asana Math, which caused LuaTEX to produce incorrect results.

The problem is related to the OpenType math parameter `DisplayOperatorMinHeight`, which is used in LuaTEX to determine the minimum size of displaystyle operators. If this parameter is incorrectly set too small in the font, displaystyle operators will appear in the same size as textstyle operators. (An example of the incorrect behavior is illustrated in Figure 2.)

As it turned out, the problem had already been found earlier when OpenType math support in LuaTEX was first tested with ConTEXt, and a workaround had been applied, but the same problem now reappeared when LuaTEX was tested with LuaLaTEX.

In ConTEXt, a patch for incorrect font parameters had been applied in the font loading code at the Lua level in `font-pat.lua`, but a similar patch was missing in `luaotfload`. Until this is fixed, a workaround to the same effect can be applied at the macro level by setting `\Umathoperatorsize\displaystyle=13.6pt`.

In any case, such kinds of patches for specific font parameter values only present a stop-gap solution until the actual fonts can be fixed. Whether or not this will be possible, critically depends on the cooperation of the font developer or distributor and may range between very quickly (for some open source projects) and next to impossible (for some commercial fonts).

Problems with OpenType font shapes

Font problems of yet another kind can occur when the assignment of glyph shapes to Unicode slots does not match the expectations, or when an incorrect font style is used for some glyphs.

One such problem was discovered for the partial differential symbol in Cambria Math and XITS Math. Since Unicode math provides a separate slot for a math italic version (U+1D715), one would expect the default slot (U+2202) to be reserved for the upright version, yet the Unicode font tables incorrectly happen to show an italic version in both slots and no upright version.

Given the confusion in the Unicode font tables, it is not surprising that several OpenType math fonts have inherited the same problem. Unfortunately, such font problems are unlikely to be fixed anytime soon.

∂	∂	$\boldsymbol{\partial}$	$\boldsymbol{\partial}$	Cambria Math
∂	∂	$\boldsymbol{\partial}$	∂	XITS Math
∂	∂	$\boldsymbol{\partial}$	∂	Asana Math

Figure 3: Comparison of different font shapes of the partial differential symbol (upright, italic, bold upright, bold italic) as they appear in Cambria Math, XITS Math, and Asana Math. Besides the confusion of upright vs. italic there are also some obvious problems for some of the bold italic versions.

Problems with TEX macro packages

Problems with TEX (or Lua) macro packages are usually easy to fix. In the course of the TEX Live 2010 pretest, a number of such issues were found in `luaotfload` and `unicode-math`, which have already been fixed.

Only one issue has remained unresolved, which is related to the use of the `\hbar` macro. In a traditional LATEX setting, `\hbar` is a macro which overlays the italic letter *h* with a bar accent from `cmr` to produce ħ. By contrast, `\hslash` is a macro to access a ready-made glyph from the AMS fonts to produce ħ.

In a Unicode math setting, only `\hslash` is assigned to a slot in an OpenType math font (U+210F), while there is no equivalent assignment for `\hbar`, which has somehow retained its original macro definition and still uses a glyph from `cmr` to create the overlay. For lack of a better solution, it would be better to define `\hbar` as an alias for `\hslash` in `unicode-math`.

Quite a different effect occurs in ConTEXt, where `\hbar` is interpreted as a diacritic text character (ħ) in upright shape (U+0127), which may be appropriate in text typesetting, but not necessarily in a math formula. As in `unicode-math`, it would be better to define `\hbar` as an alias for `\hslash` in ConTEXt as well.

Problems caused by interactions between TEX macro packages and TEX engines

Yet another kind of problem was discovered recently, which was caused by an interaction problem between macro packages and TEX engines, specifically between the `unicode-math` package and the XeTEX engine.

As it turned out, `unicode-math` allocated a new math family for the OpenType math font (such as family 4) while XeTEX (unlike LuaTEX) somehow still expected certain math font parameters to be taken from families 2 and 3 (as in traditional TEX engines).

As a result, the preloaded font metric parameters from `cmsy` and `cmex` were incorrectly used to determine the spacing of math instead of the font parameters from the OpenType math font.

A fix for this problem is still pending, but most likely it would involve changing the `unicode-math` package to account for different engine-specific behaviors of LuaTEX and XeTEX. As a workaround, we can apply a fix by reassigning the fonts in families 2 and 3 after loading an OpenType math font in family 4:

```
\ifxetex\everymath{
  \textfont3 = \textfont4
  \textfont2 = \textfont4
  \scriptfont2 = \scriptfont4
  \scriptscriptfont2 = \scriptscriptfont4
}\fi
```

Font-loading problems

Font-loading problems are usually easy to fix or avoid, once the cause of the problem has been understood. Nevertheless, such kinds of problems present a frequent source of frustration for unwary users, so it may well be useful to discuss our experiences regarding the font loading problems we encountered in the course of testing OpenType math with TEX Live 2010.

What is important here is to understand that different mechanisms are used to locate OpenType fonts in different TEX engines and macro packages.

In XeTEX, the `fontconfig` library is used to locate OpenType fonts, and this mechanism applies equally well for system fonts installed in the system font path as for fonts installed in your TEX Live distribution.

Depending on your installation, it may be necessary to adjust the `fonts.conf` config file to include the font directories in your `texmf-dist` or `texmf-local` tree, and to refresh the font cache with the `fc-cache` command. Once a font directory has been added to the search path, all kinds of font files will be found there, regardless of where the font files are located.

In LuaTEX, the `kpathsea` path searching library is used to locate fonts, which depends on the assignment of file extensions (such as `*.ttf` or `*.otf`) to different search paths. As a result of this, `cambria.ttc` will only be found in the `fonts/truetype` tree, while `euler.otf` will only be found in the `fonts/opentype` tree.

In addition to that, ConTEXt and `luaotfload` on LuaTEX use yet another font-loading mechanism based on a file cache implemented in Lua, which circumvents the `kpathsea` library completely. System fonts outside the TEXMF tree will be located using the `fonts.conf` config file to look up the font directories, but without using the the `fontconfig` library. Once a font directory has been added to the file cache, all kinds of font files will be found there, regardless of where the fonts are located, similar to the `fontconfig` library.

Comparing and testing the quality of OpenType math typesetting

To proceed with a study the quality of OpenType math font support as of TEX Live 2010, we have the following choices of typesetting engines and macro packages at our disposal (disregarding Plain LuaTEX and XeTEX which only provide low-level support):

- LuaTEX with ConTEXt MkIV
- LuaTEX with LuaLaTEX
- XeTEX with XeLaTEX

While both LuaLaTeX and ConTeXt share the same TeX engine and the same implementation of math typesetting algorithms, they differ in their high-level user interface and also in the intermediate levels of font loading code (such as `luaotfload`).

While both LuaLaTeX and XeLaTeX share the same user interface of `unicode-math` and `fontspec`, they are based on different TeX engines, LuaTeX and XeTeX, which differ significantly in their implementations of math typesetting algorithms.

Comparing the results of LuaLaTeX and ConTeXt should not be expected to expose any differences in the output from identical math typesetting algorithms, but if there are any differences, a closer analysis should help to detect bugs or inconsistencies in the different macro packages and/or in the font loading code.

Comparing the results of LuaLaTeX and XeLaTeX, however, should indeed be expected to expose some differences in the underlying typesetting algorithms. Hopefully, an analysis of these differences should allow to draw conclusions how the quality of math is affected and could be improved by taking advantage of a full-featured implementation of OpenType math.

Testing a sampling of OpenType math

When we began testing OpenType math support with TeX Live 2010, we didn't have time to do systematic and comprehensive testing, which would have been a very time-consuming and tedious task.

Instead, we wanted to get some quick impressions how well OpenType math support worked and if it would be ready for practical use, so we concentrated on testing just a sampling of mathematical notations. Given our personal background in typesetting mathematical physics, we started by creating a sample test document containing a selection of famous equations from various fields of physics, sampling various kinds of mathematical notations.

In addition to testing the available choices of TeX engines and macro packages, we also wanted to test a sampling of the available OpenType math fonts, so we proceeded to typeset identical copies of our test files with different TeX engines and with different choices of math fonts for each of Cambria Math, XITS Math, Asana Math, and Neo Euler.

Some examples of typesetting such test pages with different fonts are shown in Figures 8–11, except that each font sample was usually typeset at least twice with LuaLaTeX and XeLaTeX.

In some cases, we also tested an additional version with ConTeXt MkIV, but unfortunately we had to use a modified version of our test files in such cases.

$$\Delta\phi(\boldsymbol{r}) = \frac{\partial^2\boldsymbol{\phi}}{\partial\boldsymbol{x}^2} + \frac{\partial^2\boldsymbol{\phi}}{\partial\boldsymbol{y}^2} + \frac{\partial^2\boldsymbol{\phi}}{\partial\boldsymbol{z}^2}\,.$$

Figure 4: Comparison of math typesetting from XeLaTeX (red) and LuaLaTeX (blue) using Cambria Math. (The example shows the definition of the Laplace operator in vector analysis.)

$$R^{\mu\nu} - \frac{1}{2}Rg^{\boldsymbol{\mu\nu}} + \boldsymbol{\Lambda} g^{\boldsymbol{\mu\nu}} = -\frac{\boldsymbol{8\pi G}}{\boldsymbol{c}^{\boldsymbol{2}}}\mathbb{M}^{\mu\nu}\,.$$

Figure 5: Comparison of math typesetting from XeLaTeX (red) and LuaLaTeX (blue) using Cambria Math. (The example shows the Einstein field equation from general relativity.)

Analyzing large-scale effects

Comparing the different versions for the same font typeset with different engines or macro packages may reveal significant differences at various scales.

If there are any unexpected large-scale effects, such as using different sizes of delimiters or operators, it is usually easy to spot them simply by visual inspection. In most cases, such obvious differences will turn out to be unintentional and tend to indicate problems or bugs, such as those discussed earlier in this paper.

Analyzing small-scale effects

Once we have applied all the necessary workarounds and bug fixes to eliminate the unexpected large-scale effects, only small-scale effects should remain, affecting tiny micro-typographic details, which may be hard to see without closer inspection.

In order to the study the small-scale effects in more detail, we created another set of more sophisticated test files using PDF overlays between different versions of the same document using different colors.

These overlays were generated with PDFLaTeX using the `pdfpages` package as follows:

```
\documentclass[a4paper]{article}
\usepackage{pdfpages}

\begin{document}
\includepdfmerge[nup=2x1,noautoscale=true,
  delta=-21cm 0] % width of A4 paper
 {xelatex-test.pdf,1,lualatex-test.pdf,1,
  ...
  xelatex-test.pdf,n,lualatex-test.pdf,n}
\end{document}
```

This setup will put each page of the LuaLaTeX test file on top of the corresponding page of the XeLaTeX test file. For improved visibility, colors should be chosen in such a way that the darker colors (such as black or blue) will appear on top of the brighter colors (such as red). In our example illustrations we have usually used red for XeLaTeX overlaid by blue for LuaLaTeX.

$$\Delta\phi(\boldsymbol{r}) = \frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} + \frac{\partial^2\phi}{\partial z^2}.$$

Figure 6: Comparison of math typesetting from XeLaTeX (red) and LuaLaTeX (blue) after applying a workaround for XeLaTeX to circumvent inconsistent placement of superscripts.

$$R^{\mu\nu} - \frac{1}{2}Rg^{\mu\nu} + \Lambda g^{\mu\nu} = -\frac{8\pi G}{c^2}M^{\mu\nu}.$$

Figure 7: Comparison of math typesetting from XeLaTeX (red) and LuaLaTeX (blue) after applying a workaround for XeLaTeX to circumvent inconsistent placement of superscripts.

Analyzing the results of overlays

Once we have generated overlays of the results from typesetting the same equations with different engines, it is easy to detect if any differences occur. However, it is far from easy to understand how these differences come about and what their implications might be.

In our first series of tests, we originally noticed some very significant differences in vertical spacing around fraction bars. Once we detected the problem of XeTeX incorrectly using the preloaded set of font parameters and applied a workaround, most differences in vertical spacing disappeared and only few remained.

In our second series of tests, only relatively few differences remained. Moreover, the remaining effects only appeared for some fonts and not for others. While there were hardly any effects on vertical spacing for XITS Math, there were some notable differences in the placement of scripts for Asana Math or Cambria Math, as illustrated in Figures 4–5.

Upon closer inspection, we eventually found that the differences only affected some letters within an equation, but not all of them. There were no differences on letters without ascenders or descenders (as in x_0 or x^2), while there were differences for superscripts on letters with ascenders (as in ∂^2) and also for subscripts on letters with descenders (as in μ_0). The cause of this problem isn't clear yet, but it most likely indicates an unresolved engine problem in XeTeX.

In our third series of tests, the remaining effects on vertical spacing could be eliminated completely after we applied a workaround for the placement of scripts, and only some effects on horizontal spacing remained, as illustrated in Figures 6–7.

The remaining effects on horizontal spacing are most likely related to different interpretations of OpenType glyph metrics in different TeX engines (such as italic corrections and math kerning [18]), which certainly will have an effect on the quality of math typesetting, but only on a very small scale.

Summary and Conclusions

In this paper, we have reported our experiences, findings and observations from testing OpenType math support in TeX Live 2010 with different TeX engines, macro packages and fonts.

When we set out, we expected to gain some insights how the quality of math typesetting was affected by the use of additional font parameters in a more sophisticated implementation of OpenType math support.

In the end, however, it turned out that most of the differences were actually caused by unresolved bugs in both macro packages and TeX engines, while the use of additional math font parameters appears to be largely irrelevant for our selection of test cases.

It was only by direct comparison with LuaTeX that some long-standing bugs or inconsistencies in XeTeX engine and the `unicode-math` package could be found. Without a suitable baseline reference, it is obviously hard to tell if the spacing of math is exactly right or just slightly wrong, so it is not surprising that minor inconsistencies went unnoticed for a long time.

Once we applied workarounds or fixes for the problems we discovered, the remaining differences between different TeX engines turned out to be much smaller than expected and only affected the horizontal spacing, but no longer the vertical spacing.

Given the scale of the remaining effects, our studies regarding the quality of math typesetting in different TeX engines remain inconclusive for now. Both engines can produce very similar results, but XeTeX will do so only after applying a number of fixes and workarounds to arrive at what LuaTeX will do by default.

Acknowledgements

The author would like to thank the developers involved in math-related TeX engines, macro packages and fonts for their assistance and feedback during the testing of OpenType math font support in TeX Live 2010.

In particular, Will Robertson (`unicode-math`), Khaled Hosny (`luaotfload`), Taco Hoekwater (LuaTeX), Hans Hagen (ConTeXt), Jonathan Kew (XeTeX), Karl Berry and Peter Breitenlohner (TeX Live 64-bit Linux binaries) contributed to our testing and problem solving efforts in one way or another.

Fortunately, we were able to discover and eliminate a number of bugs before the deadline of TeX Live 2010 pretest. Unfortunately, not all known issues could be resolved in time, so some problems remain to be fixed in future releases. While such fixes for macro packages and fonts can be issued through the TeX Live update mechanism at any time, fixes for TeX engines may be delayed until next year's TeX Live release.

Cambria Math Example

Vector calculus:

$$\nabla\phi(\boldsymbol{r}) = \frac{\partial\phi}{\partial x}\boldsymbol{e}_x + \frac{\partial\phi}{\partial y}\boldsymbol{e}_y + \frac{\partial\phi}{\partial z}\boldsymbol{e}_z\,,$$

$$\Delta\phi(\boldsymbol{r}) = \frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} + \frac{\partial^2\phi}{\partial z^2}\,.$$

Maxwell equations (differential form):

$$\operatorname{div}\varepsilon_0\boldsymbol{E} = \lambda\,,\quad \operatorname{div}\boldsymbol{B} = 0\,,$$

$$\operatorname{rot}\boldsymbol{E} = -\frac{\partial\boldsymbol{B}}{\partial t}\,,\quad \operatorname{rot}\frac{\boldsymbol{B}}{\mu_0} = \boldsymbol{j} + \frac{\partial\varepsilon_0\boldsymbol{E}}{\partial t}\,.$$

Maxwell equations (integral form):

$$\int_F \varepsilon_0\boldsymbol{E}\cdot\mathrm{d}\boldsymbol{f} = \int_V \lambda\,\mathrm{d}V\,,\quad \int_F \boldsymbol{B}\cdot\mathrm{d}\boldsymbol{f} = 0\,,$$

$$\oint_C \boldsymbol{E}\cdot\mathrm{d}\boldsymbol{l} = -\frac{\mathrm{d}}{\mathrm{d}t}\int_F \boldsymbol{B}\cdot\mathrm{d}\boldsymbol{f}\,,$$

$$\oint_C \frac{\boldsymbol{B}}{\mu_0}\cdot\mathrm{d}\boldsymbol{l} = \int_F\left(\boldsymbol{j} + \frac{\partial\varepsilon_0\boldsymbol{E}}{\partial t}\right)\cdot\mathrm{d}\boldsymbol{f}\,.$$

Electromagnetic wave equations:

$$\Delta\boldsymbol{E} - \frac{1}{c^2}\frac{\partial^2\boldsymbol{E}}{\partial t^2} = \frac{1}{\varepsilon_0}\nabla\lambda + \mu_0\frac{\partial\boldsymbol{j}}{\partial t}\,,$$

$$\Delta\boldsymbol{B} - \frac{1}{c^2}\frac{\partial^2\boldsymbol{B}}{\partial t^2} = -\mu_0\operatorname{rot}\boldsymbol{j}\,.$$

Energy-mass equation (special relativity):

$$E = \frac{m_0c^2}{\sqrt{1 - v^2/c^2}}\,.$$

Einstein field equation (general relativity):

$$R^{\mu\nu} - \frac{1}{2}Rg^{\mu\nu} + \Lambda g^{\mu\nu} = -\frac{8\pi G}{c^2}M^{\mu\nu}\,.$$

Schrödinger equation (quantum mechanics):

$$\mathrm{i}\hbar\frac{\partial\psi}{\partial t} = \widehat{H}\,\psi = \frac{1}{2m}\left(\frac{\hbar}{\mathrm{i}}\nabla - q\boldsymbol{A}\right)^2\psi + q\phi\,\psi\,.$$

Dirac equation (relativistic quantum mechanics):

$$\gamma^\alpha\left(\frac{\hbar}{\mathrm{i}}\partial_\alpha - qA_\alpha\right)\psi + m_0c\,\psi = 0\,.$$

Figure 8: Sampling of equations typeset with LuaLaTeX using Cambria and Cambria Math.

Asana Math Example

Vector calculus:

$$\nabla\phi(\boldsymbol{r}) = \frac{\partial\phi}{\partial x}\boldsymbol{e}_x + \frac{\partial\phi}{\partial y}\boldsymbol{e}_y + \frac{\partial\phi}{\partial z}\boldsymbol{e}_z\,,$$

$$\Delta\phi(\boldsymbol{r}) = \frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} + \frac{\partial^2\phi}{\partial z^2}\,.$$

Maxwell equations (differential form):

$$\operatorname{div}\varepsilon_0\boldsymbol{E} = \lambda\,,\quad \operatorname{div}\boldsymbol{B} = 0\,,$$

$$\operatorname{rot}\boldsymbol{E} = -\frac{\partial\boldsymbol{B}}{\partial t}\,,\quad \operatorname{rot}\frac{\boldsymbol{B}}{\mu_0} = \boldsymbol{j} + \frac{\partial\varepsilon_0\boldsymbol{E}}{\partial t}\,.$$

Maxwell equations (integral form):

$$\int_F \varepsilon_0\boldsymbol{E}\cdot\mathrm{d}\boldsymbol{f} = \int_V \lambda\,\mathrm{d}V\,,\quad \int_F \boldsymbol{B}\cdot\mathrm{d}\boldsymbol{f} = 0\,,$$

$$\oint_C \boldsymbol{E}\cdot\mathrm{d}\boldsymbol{l} = -\frac{\mathrm{d}}{\mathrm{d}t}\int_F \boldsymbol{B}\cdot\mathrm{d}\boldsymbol{f}\,,$$

$$\oint_C \frac{\boldsymbol{B}}{\mu_0}\cdot\mathrm{d}\boldsymbol{l} = \int_F\left(\boldsymbol{j} + \frac{\partial\varepsilon_0\boldsymbol{E}}{\partial t}\right)\cdot\mathrm{d}\boldsymbol{f}\,.$$

Electromagnetic wave equations:

$$\Delta\boldsymbol{E} - \frac{1}{c^2}\frac{\partial^2\boldsymbol{E}}{\partial t^2} = \frac{1}{\varepsilon_0}\nabla\lambda + \mu_0\frac{\partial\boldsymbol{j}}{\partial t}\,,$$

$$\Delta\boldsymbol{B} - \frac{1}{c^2}\frac{\partial^2\boldsymbol{B}}{\partial t^2} = -\mu_0\operatorname{rot}\boldsymbol{j}\,.$$

Energy-mass equation (special relativity):

$$E = \frac{m_0c^2}{\sqrt{1 - v^2/c^2}}\,.$$

Einstein field equation (general relativity):

$$R^{\mu\nu} - \frac{1}{2}Rg^{\mu\nu} + \Lambda g^{\mu\nu} = -\frac{8\pi G}{c^2}M^{\mu\nu}\,.$$

Schrödinger equation (quantum mechanics):

$$\mathrm{i}\hbar\frac{\partial\psi}{\partial t} = \widehat{H}\,\psi = \frac{1}{2m}\left(\frac{\hbar}{\mathrm{i}}\nabla - q\boldsymbol{A}\right)^2\psi + q\phi\,\psi\,.$$

Dirac equation (relativistic quantum mechanics):

$$\gamma^\alpha\left(\frac{\hbar}{\mathrm{i}}\partial_\alpha - qA_\alpha\right)\psi + m_0c\,\psi = 0\,.$$

Figure 9: Sampling of equations typeset with LuaLaTeX using TeX Gyre Pagella and Asana Math.

XITS Math Example

Vector calculus:

$$\boldsymbol{\nabla}\phi(\boldsymbol{r}) = \frac{\partial\phi}{\partial x}\boldsymbol{e}_x + \frac{\partial\phi}{\partial y}\boldsymbol{e}_y + \frac{\partial\phi}{\partial z}\boldsymbol{e}_z\,,$$

$$\Delta\phi(\boldsymbol{r}) = \frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} + \frac{\partial^2\phi}{\partial z^2}\,.$$

Maxwell equations (differential form):

$$\operatorname{div}\varepsilon_0\boldsymbol{E} = \lambda\,,\quad \operatorname{div}\boldsymbol{B} = 0\,,$$

$$\operatorname{rot}\boldsymbol{E} = -\frac{\partial\boldsymbol{B}}{\partial t}\,,\quad \operatorname{rot}\frac{\boldsymbol{B}}{\mu_0} = \boldsymbol{j} + \frac{\partial\varepsilon_0\boldsymbol{E}}{\partial t}\,.$$

Maxwell equations (integral form):

$$\int_F \varepsilon_0\boldsymbol{E}\cdot\mathrm{d}\boldsymbol{f} = \int_V \lambda\,\mathrm{d}V\,,\quad \int_F \boldsymbol{B}\cdot\mathrm{d}\boldsymbol{f} = 0\,,$$

$$\oint_C \boldsymbol{E}\cdot\mathrm{d}\boldsymbol{l} = -\frac{\mathrm{d}}{\mathrm{d}t}\int_F \boldsymbol{B}\cdot\mathrm{d}\boldsymbol{f}\,,$$

$$\oint_C \frac{\boldsymbol{B}}{\mu_0}\cdot\mathrm{d}\boldsymbol{l} = \int_F\left(\boldsymbol{j} + \frac{\partial\varepsilon_0\boldsymbol{E}}{\partial t}\right)\cdot\mathrm{d}\boldsymbol{f}\,.$$

Electromagnetic wave equations:

$$\Delta\boldsymbol{E} - \frac{1}{c^2}\frac{\partial^2\boldsymbol{E}}{\partial t^2} = \frac{1}{\varepsilon_0}\boldsymbol{\nabla}\lambda + \mu_0\frac{\partial\boldsymbol{j}}{\partial t}\,,$$

$$\Delta\boldsymbol{B} - \frac{1}{c^2}\frac{\partial^2\boldsymbol{B}}{\partial t^2} = -\mu_0\operatorname{rot}\boldsymbol{j}\,.$$

Energy-mass equation (special relativity):

$$E = \frac{m_0c^2}{\sqrt{1 - v^2/c^2}}\,.$$

Einstein field equation (general relativity):

$$R^{\mu\nu} - \frac{1}{2}Rg^{\mu\nu} + \Lambda g^{\mu\nu} = -\frac{8\pi G}{c^2}M^{\mu\nu}\,.$$

Schrödinger equation (quantum mechanics):

$$\mathrm{i}\hbar\frac{\partial\psi}{\partial t} = \widehat{H}\,\psi = \frac{1}{2m}\left(\frac{\hbar}{\mathrm{i}}\boldsymbol{\nabla} - q\boldsymbol{A}\right)^2\psi + q\phi\,\psi\,.$$

Dirac equation (relativistic quantum mechanics):

$$\gamma^\alpha\left(\frac{\hbar}{\mathrm{i}}\partial_\alpha - qA_\alpha\right)\psi + m_0c\,\psi = 0\,.$$

Figure 10: Sampling of equations typeset with LuaLaTeX using XITS and XITS Math.

Neo Euler Example

Vector calculus:

$$\nabla\phi(\mathbf{r}) = \frac{\partial\phi}{\partial x}\mathbf{e}_x + \frac{\partial\phi}{\partial y}\mathbf{e}_y + \frac{\partial\phi}{\partial z}\mathbf{e}_z\,,$$

$$\Delta\phi(\mathbf{r}) = \frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} + \frac{\partial^2\phi}{\partial z^2}\,.$$

Maxwell equations (differential form):

$$\operatorname{div}\varepsilon_0\mathbf{E} = \lambda\,,\quad \operatorname{div}\mathbf{B} = 0\,,$$

$$\operatorname{rot}\mathbf{E} = -\frac{\partial\mathbf{B}}{\partial t}\,,\quad \operatorname{rot}\frac{\mathbf{B}}{\mu_0} = \mathbf{j} + \frac{\partial\varepsilon_0\mathbf{E}}{\partial t}\,.$$

Maxwell equations (integral form):

$$\int_F \varepsilon_0\mathbf{E}\cdot\mathrm{d}\mathbf{f} = \int_V \lambda\,\mathrm{d}V\,,\quad \int_F \mathbf{B}\cdot\mathrm{d}\mathbf{f} = 0\,,$$

$$\oint_C \mathbf{E}\cdot\mathrm{d}\mathbf{l} = -\frac{\mathrm{d}}{\mathrm{d}t}\int_F \mathbf{B}\cdot\mathrm{d}\mathbf{f}\,,$$

$$\oint_C \frac{\mathbf{B}}{\mu_0}\cdot\mathrm{d}\mathbf{l} = \int_F\left(\mathbf{j} + \frac{\partial\varepsilon_0\mathbf{E}}{\partial t}\right)\cdot\mathrm{d}\mathbf{f}\,.$$

Electromagnetic wave equations:

$$\Delta\mathbf{E} - \frac{1}{c^2}\frac{\partial^2\mathbf{E}}{\partial t^2} = \frac{1}{\varepsilon_0}\nabla\lambda + \mu_0\frac{\partial\mathbf{j}}{\partial t}\,,$$

$$\Delta\mathbf{B} - \frac{1}{c^2}\frac{\partial^2\mathbf{B}}{\partial t^2} = -\mu_0\operatorname{rot}\mathbf{j}\,.$$

Energy-mass equation (special relativity):

$$E = \frac{m_0c^2}{\sqrt{1 - v^2/c^2}}\,.$$

Einstein field equation (general relativity):

$$R^{\mu\nu} - \frac{1}{2}Rg^{\mu\nu} + \Lambda g^{\mu\nu} = -\frac{8\pi G}{c^2}M^{\mu\nu}\,.$$

Schrödinger equation (quantum mechanics):

$$\mathrm{i}\hbar\frac{\partial\psi}{\partial t} = \hat{H}\,\psi = \frac{1}{2m}\left(\frac{\hbar}{\mathrm{i}}\nabla - q\mathbf{A}\right)^2\psi + q\phi\,\psi\,.$$

Dirac equation (relativistic quantum mechanics):

$$\gamma^\alpha\left(\frac{\hbar}{\mathrm{i}}\partial_\alpha - qA_\alpha\right)\psi + m_0c\,\psi = 0\,.$$

Figure 11: Sampling of equations typeset with LuaLaTeX using TeX Gyre Pagella and Neo Euler.

References

[1] TEX Users Group: Testing TEX Live before release.
http://tug.org/texlive/pretest

[2] Barbara Beeton, Asmus Freytag, Murray Sargent III: Unicode Support for Mathematics.
Unicode Technical Report UTR#25. 2001.
http://www.unicode.org/reports/tr25/

[3] Barbara Beeton: Unicode and math, a combination whose time has come. *TUGboat*, 21(3):174–185, 2000.
Proceedings of TUG 2000, Oxford, UK.
http://www.tug.org/TUGboat/tb21-3/tb68beet.pdf

[4] Barbara Beeton: The STIX Project – From Unicode to fonts. *TUGboat*, 28(3):299–304, 2007.
Proceedings of TUG 2007, San Diego, CA, USA.
http://www.tug.org/TUGboat/tb28-3/tb90beet.pdf

[5] Ulrik Vieth: Math Typesetting in TEX: The Good, The Bad, The Ugly. *MAPS*, 26:207–216, 2001.
Proceedings of EuroTEX 2001, Kerkrade, Netherlands.
http://www.ntg.nl/maps/26/27.pdf

[6] OpenType Specification, Version 1.6.
http://www.microsoft.com/typography/otspec/

[7] Murray Sargent III: Math in Office Blog.
http://blogs.msdn.com/murrays/default.aspx

[8] John Hudson, Ross Mills: Mathematical Typesetting: Mathematical and scientific typesetting solutions.
Promotional Booklet, Microsoft, 2006.

[9] Daniel Rhatigan: Three typefaces for mathematics.
Dissertation for the MA in typeface design, 2007.
http://www.typeculture.com/academic_resource/articles_essays/pdfs/tc_article_47.pdf

[10] Murray Sargent III: Unicode Nearly Plain Text Encodings of Mathematics.
Unicode Technical Note UTN#28, 2006.
http://www.unicode.org/notes/tn28/

[11] George Williams: FontForge: Math typesetting information.
http://fontforge.sourceforge.net/math.html

[12] Ulrik Vieth: Do we need a 'Cork' math font encoding? *TUGboat*, 29(3):426–434, 2008.
Proceedings of TUG 2008, Cork, Ireland.
http://www.tug.org/TUGboat/tb29-3/tb93vieth.pdf

[13] Ulrik Vieth: OpenType Math Illuminated. Reprinted in *TUGboat*, 30(1):22–31, 2009.
Proceedings of BachoTEX 2009, Bachotek, Poland.
http://www.tug.org/TUGboat/tb30-1/tb94vieth.pdf

[14] Bogusław Jackowski: Appendix G Illuminated. *TUGboat*, 27(1):83–90, 2006.
Proceedings of EuroTEX 2006, Debrecen, Hungary.
http://www.tug.org/TUGboat/tb27-1/tb86jackowski.pdf

[15] Ulrik Vieth: Understanding the æsthetics of math typesetting. *Biuletyn GUST*, 5–12, 2008.
Proceedings of BachoTEX 2008, Bachotek, Poland.
http://www.gust.org.pl/projects/e-foundry/math-support/vieth2008.pdf

[16] Jonathan: Kew: XeTEX Live. *TUGboat*, 29(1):151–156, 2008.
Proceedings of BachoTEX 2007, Bachotek, Poland.
http://www.tug.org/TUGboat/tb29-1/tb91kew.pdf

[17] Taco Hoekwater: LuaTEX Reference Manual.
http://www.luatex.org/svn/trunk/manual/luatexref-t.pdf

[18] Taco Hoekwater: Math in LuaTEX 0.40.
MAPS, 38:22–31, 2009.
http://www.ntg.nl/maps/38/04.pdf

[19] Hans Hagen: Unicode Math in ConTEXt.
MAPS, 38:32–46, 2009.
http://www.ntg.nl/maps/38/05.pdf

[20] Will Robertson: Advanced font features with XeTEX: The fontspec package. *TUGboat*, 26(3):215–223, 2005.
http://www.tug.org/TUGboat/tb26-3/tb84robertson.pdf

[21] Will Robertson: The fontspec macro package.
http://www.ctan.org/pkg/fontspec
http://github.com/wspr/fontspec

[22] Will Robertson: The unicode-math macro package.
http://www.ctan.org/pkg/unicode-math
http://github.com/wspr/unicode-math

[23] Aditya Mahajan: Integrating Unicode and OpenType math in ConTEXt. *TUGboat*, 30(2):243–246, 2009.
Proceedings of TUG 2009, Notre Dame, IN, USA.
https://www.tug.org/members/TUGboat/tb30-2/tb95mahajan-cmath.pdf

[24] Khaled Hosny et al.: The luaotfload macro package.
http://www.ctan.org/pkg/luaotfload
http://github.com/khaledhosny/luaotfload

[25] Will Robertson: Unicode mathematics in LaTEX: advantages and challenges.
To appear in *TUGboat*, 31(2):???–???, 2010.
Proceedings of TUG 2010, San Francisco, CA, USA.
https://www.tug.org/members/TUGboat/tb31-2/tb98robertson.pdf

[26] Ulrik Vieth: Experiences typesetting mathematical physics. *MAPS*, 39:166–178, 2009.
Proceedings of EuroTEX 2009, Delft, Netherlands.
https://www.tug.org/members/TUGboat/tb30-3/tb96vieth.pdf

[27] Apostolos Syropoulos: Asana Math Font.
http://www.ctan.org/pkg/asana-math

[28] Khaled Hosny: XITS Fonts.
http://www.ctan.org/pkg/xits
http://github.com/khaledhosny/xits-math

[29] STIX Consortium: STIX Fonts.
http://www.stixfonts.org/
http://www.ctan.org/pkg/stix

[30] Khaled Hosny: Neo Euler Font.
http://github.com/khaledhosny/euler-otf

[31] Hans Hagen, Taco Hoekwater, Volker RW Schaa: Reshaping Euler: A collaboration with Hermann Zapf. *TUGboat*, 29(3):283–287, 2008.
http://www.tug.org/TUGboat/tb29-2/tb92hagen-euler.pdf

Ulrik Vieth
Vaihinger Straße 69
70567 Stuttgart
Germany
ulrik dot vieth (at) arcor dot de