OpenType math font development: Progress and challenges*

Ulrik Vieth

Abstract

A main reason for the development of the LuaTeX and XeTeX engines has been to provide support for Unicode and OpenType font technology, which implies support for Unicode math and OpenType math as well. One important ingredient is the development of full-featured OpenType math fonts, which are needed to replace traditional math fonts. In this paper, we review recent progress in OpenType math font development as well as the many challenges faced by font developers of OpenType math fonts.

1 Introduction

In this paper, we will discuss technical details of OpenType math font development, so we will assume that readers have some familiarity with the basic concepts of Unicode math and OpenType math font technology.

When we speak about Unicode math, we refer to an effort that was undertaken between 1998 and 2002 by a group of scientific and technical publishers to get math symbols and alphabets accepted into the Unicode standard. As a result of this activity, hundreds of math symbols as well as dozens of math alphabets have been added to Unicode, and have become part of the official standard ever since Unicode 3.2 was released in 2002 [1, 2].

From a technical point of view, Unicode math is nothing special, just a convenient term for a subset of Unicode that is relevant for typesetting math.

When we speak about OpenType math, we refer to an extension of the OpenType font format [3] that was developed by Microsoft when they introduced support for math typesetting in Office 2007 [4, 5]. As a result of this, a new optional MATH table has been added to the OpenType font format, which is used to store all the additional information needed for proper typesetting of math, such as font metric parameters controlling the spacing of math as well as additional lookup mechanisms of glyph variants [6, 7].

From a technical point of view, OpenType math does represent an extension of the OpenType font format, but it uses a well-defined extension mechanism, so the optional MATH table will only be seen by typesetting engines which happen to know about math, while other engines will safely ignore it.

Finally, it is helpful to understand how Unicode math, OpenType math, fonts and typesetting engines work together.

Unicode math, by itself, only defines the encoding of mathematical input. It does not define any semantics of how a math formula is arranged or spaced. That is a matter left to the font technology (OpenType) and the typesetting engine (LuaTeX or XeTeX or MS Office).

In Unicode, each math symbol is usually represented only once, regardless of how many sizes may be needed for proper typesetting. Letters of math alphabets are the exceptions: since a font change in math usually also conveys a different meaning, each variation of a letter has a separate slot.

OpenType, as a font technology, provides the glyphs and metric information for mathematical output. OpenType math fonts are encoded based on Unicode, but they can extend beyond the scope of Unicode by taking advantage of the private use area.

Where Unicode math defines only a single slot for each symbol, OpenType math provides lookup mechanisms for multiple sizes of glyph variants or glyph substitutions for constructed symbols.

Where Unicode math does not define any provisions for the semantics of math, OpenType math provides a table to store the font metric information controlling the spacing, but leaves interpretation of these parameters to the typesetting engine.

In the end, it all depends on having an OpenType math-capable typesetting system to take advantage of the information in OpenType math fonts and to properly arrange math formulas.

When OpenType math was first introduced, MS Office 2007 was the only available OpenType math engine, but both XeTeX and LuaTeX have since implemented OpenType math capabilities [8, 9].

While XeTeX provides only a partial implementation, LuaTeX aims to provide a full-featured OpenType math engine. (As of 2012, work on improving math typesetting in XeTeX has been ongoing, so hopefully both engines will eventually be able to produce the same quality of math typesetting.)

2 Progress in OpenType math fonts

When OpenType math was introduced, only a single math font was available: Cambria Math, which was developed by Tiro Typeworks on behalf of Microsoft and bundled with MS Office 2007. In some sense, the situation was reminiscent of the early days of TeX, when Computer Modern was the only available math font in METAFONT format.

* First submitted for publication in the Proceedings of the 5th ConTeXt Meeting 2011 (to appear). Updated and revised for BachoTeX 2012. Updated and revised again for *TUGboat*. Reprinted with permission.

In recent years, several more OpenType math fonts have been added, so by the time of writing the first revision of this paper (September 2011) we had at least 4 math fonts available as released versions:

- Cambria Math, by Tiro Typeworks on behalf of Microsoft [11],
- Asana Math, by Apostolos Syropoulos [12],
- XITS Math, by Khaled Hosny, derived from the STIX fonts [13, 14],
- Latin Modern Math, by the GUST e-foundry [15, 16] (released in June 2011).

In the meantime, several additional choices of math fonts that were under development have been completed and released as well:

- Minion Math, by Johannes Küster [17],
- Lucida Math, by Khaled Hosny on behalf of TUG, designed by Bigelow & Holmes [18, 19] (released in March 2012),
- TEX Gyre Pagella Math, by the GUST e-foundry [20] (released in June 2012),
- TEX Gyre Termes Math, by the GUST e-foundry [20] (released in October 2012).

Finally, two more math font projects are under development or have been announced:

- Neo Euler, by Khaled Hosny on behalf of DANTE, designed by Hermann Zapf [21, 22],
- Maxwell Math, by Tiro Typeworks [23].

Given all these recent and ongoing font projects, we now have OpenType math font support for a number of popular text typefaces, such as Latin Modern, Times, Palatino, Lucida Bright, and Minion, although some of these are non-free, but subject to the usual industry proprietary licensing.

In some sense, the situation is now reminiscent of the early days of PostScript font support for TEX, when choices of math fonts were still very few, but several popular typefaces were already supported.

An interesting note is that these fonts have been developed by relatively few teams and individuals: Tiro Typeworks (Cambria, Maxwell), the GUST e-foundry (Latin Modern, TEX Gyre), and Khaled Hosny (XITS (with STIX), Neo Euler, Lucida), in addition to the solo efforts by Johannes Küster (Minion Math), and Apostolos Syropoulos (Asana Math).

We may conclude that OpenType math font development remains a very challenging task, that has been mastered by only a few.

3 References for OpenType math

Before we consider the challenges faced by font developers of OpenType math fonts, it may be worthwhile to consider the question: What is the basis for OpenType math font development?

First, there is a specification of the OpenType MATH table, developed by the Microsoft typography group. The specification is officially considered experimental and is available only on request, so it has remained unpublished for years, despite the fact that it has been widely adopted as a *de facto* standard by typesetting engines and font tools.

Next, there is a reference implementation of an OpenType math font, Cambria Math, developed by Tiro Typeworks on behalf of Microsoft. This font is widely available, and can be viewed with font editors such as FontForge, making it easily possible to study how it is constructed and what it contains.

Finally, there is a reference implementation of an OpenType math typesetting engine, namely MS Office, developed by the Microsoft Office group. Unlike the specification, which is at least somewhat open, the reference implementation is completely closed and off-limits, so it is impossible to see how the specification is actually implemented.

Given this scenario, developers of OpenType math fonts or engines face the problem of determining what defines the reference behavior and what may be needed to make their fonts behave correctly with different typesetting engines.

First, the OpenType math specification may not be perfect, leaving some gray areas open to questions or interpretations. For example, there is hardly any description when to apply italic correction.

Next, the reference OpenType math font may not be perfect either. For example, it may have some incorrect parameter settings, which may confuse some engines when interpreting the parameter values literally.

Finally, the reference OpenType math engine may not be perfect either. For example, it may have interpreted the specification in certain ways, or it may have included some workarounds to patch certain problems in the reference font.

In LuaTEX, the implementation of the math engine has followed the specification as much as possible, but in case of doubt, it has chosen to follow the reference implementation. OpenType math fonts developed and tested with LuaTEX should work with MS Office equally well, although they may not work quite as well with earlier versions of XeTEX.

4 OpenType math development challenges

Development of OpenType math fonts is challenging for many reasons. Besides the inherent complexity, the size of the project is also a factor. Typical examples of OpenType math fonts may include between

1500 and 2500 symbols or letters, not counting size variants or optical design sizes. Besides technical issues and design issues, achieving some level of completeness is already a challenge by itself.

4.1 Completeness of math symbols

Unicode math defines thousands of math symbols in all. However, developers of OpenType math fonts can choose what to implement and will typically implement only a subset of the most important symbols, accepting some level of incompleteness.

Most OpenType math fonts include a common subset, comparable to what is in traditional TEX math fonts based on 7-bit or 8-bit encodings, but very few OpenType math fonts will provide the complete set of math symbols defined in Unicode.

At the moment, XITS Math is the most complete of all available OpenType math fonts, because it is based on the STIX fonts [24], which have taken nearly a decade to design and review all the glyphs.

At the other end of the spectrum, Neo Euler is the least complete of all OpenType math fonts, which is understandable given that Euler always had to rely on borrowing symbols from other fonts.

All the other available OpenType math fonts rank somewhere in between these extremes, with each of Asana Math, Lucida Math, and Minion Math providing some ranges of additional symbols that go beyond the scope of Cambria Math. By comparison, Latin Modern Math is still far less complete.[1]

One important factor to consider when converting TEX math fonts to OpenType format is that a number of macros need to be replicated by designed symbols. This will include symbols such as triple dots, double and triple integrals, negated or stacked symbols, arrows with hooks or tails, long arrows, over- and underbraces.

In the end, how much incompleteness can be tolerated depends on actual usage. If you are using only basic math, the symbol coverage of any available font will suffice, but if you need some special notations, it may be worthwhile to check to see which fonts provide the required symbols.

Probably the best reference of Unicode math symbols for various OpenType math fonts can be found in the documentation of the `unicode-math` package [25].

4.2 Completeness of math alphabets

Unicode math defines more than a dozen shapes of math alphabets:

- 4 shapes of a serif alphabet (regular, italic, bold, bold italic), each including Latin and Greek,
- 4 shapes of sans-serif (regular, italic, bold, bold italic), some including Latin and Greek,
- 2 shapes of Script/Calligraphic (regular, bold), each including upper- and lowercase,
- 2 shapes of Fraktur/Blackletter (regular, bold), each including upper- and lowercase,
- 1 shape of open face or Blackboard bold (regular), also including upper- and lowercase.

Once again, developers of OpenType math fonts can choose what to implement and will typically implement a common subset of the most important alphabets, but will not necessarily provide all the shapes.

Except for Neo Euler, which has only an upright shape by design, most fonts include at least 4 shapes of the main serif alphabet, but the completeness of other math alphabets varies considerably.

Some fonts may not include any sans-serif at all, some may include only an incomplete range of sans-serif (only Latin, no Greek), some may be missing bold Script and Fraktur, and some may be missing lowercase Script or lowercase and numerals in Blackboard Bold.

Besides missing some alphabets, some fonts may also provide some additional alphabets, such as an alternative italics, or a different style of Script or Calligraphic. Typically, these alphabets will have to be accessed via OpenType features using numbered stylistic sets.

As mentioned above, how much incompleteness is tolerable depends on your usage. If you are typesetting physics, you may well be interested in having a bold sans-serif alphabet for typesetting tensors, but you may need them only a few times in a series of books. In such cases, you may ask if you really need Greek for tensors, or if you can do with Latin only. And if you do need Greek for tensors, you may ask if you really need lowercase Greek, or if you can do with uppercase Greek only.

Depending on your requirements, your choices of math fonts providing the required alphabets may be limited, or you may be able to avoid the limitations. Finally, you may also consider substituting another font for certain math alphabets.

Taking advantage of stylistic sets or range substitutions depends on support by macro packages, but such features are already provided (for LATEX) by the `unicode-math` and `fontspec` packages (on top of `luaotfload`) [26, 27, 28, 29].

4.3 Choosing typefaces for math alphabets

Unicode math combines a number of different shapes of math alphabets into a single font, including a

[1] However, the updates of Latin Modern Math and TEX Gyre Math in October 2012 have made them much more complete than their original releases.

matching serif and sans-serif typeface, a Script or Calligraphic, a Fraktur or Blackletter, a Blackboard bold, and a typewriter design (which we will ignore).

In the case of comprehensive font families, such as Latin Modern or Lucida, the choice of matching typeface designs will be obvious, as there is already a set of serif, sans-serif, and other font shapes that have been designed to be used together.

In other cases, however, choosing a set of matching typeface designs leads to a non-trivial design question: What is the proper choice of a sans-serif to be combined with a given serif font?

For a Times-like serif font (as in XITS Math), Arial or Helvetica may be an obvious choice of a sans-serif font (although this is debatable), but what should be used with Palatino or Euler? Should the sans-serif be another Hermann Zapf design (such as Optima)? What should be used with Minion? Should the sans-serif be another Adobe design (such as Myriad)? Or should it be something entirely different?

Should the sans-serif be a matching design with similar characteristics or should it be a contrasting design? How much similarity is needed to achieve consistency? How much contrast is needed to make individual letters clearly distinguishable?

Answers to such fundamental design questions may not be clear or obvious, but at some point font designers or developers will have to make a choice, or choose not to make a choice.

In some cases, such as for Minion Math or Neo Euler, decisions have been deliberately left open, leaving the fonts incomplete without any sans-serif alphabets. In other cases, such as for Asana Math (derived from `pxfonts` and `cbgreek`), decisions seem to have been taken based on what was available or which sans-serif fonts offered a suitable set of Greek besides Latin.

Besides the choice of sans-serif, similar design decisions may arise for the choice of Script, Calligraphic, Fraktur, or Blackboard Bold designs.

For Script, Calligraphic, or Fraktur, there seems to be considerable agreement among different font designers regarding the typical look of these shapes. Several different fonts seem to be very similar in the overall style, although each is still different in its individual design, as was discussed in a separate article by Michael Sharpe [30].

For Blackboard Bold, however, some very different approaches have been favored by different designers. In some cases, such as Cambria Math or Minion Math, the Blackboard Bold design is derived from an open face version of the main serif font. In other cases, such as XITS Math, Lucida Math, and Latin Modern Math, the Blackboard Bold is a very different style (typically sans-serif), which may be unrelated to the main sans-serif font.

4.4 Choices of Script (Calligraphic)

Design choices of Script alphabets fall into several groups, being fairly similar within a group. One group favors a very embellished style of Script:

XITS Math	$\mathscr{ABCXYZ}\ \mathscr{abcxyz}$
Asana Math	$\mathscr{ABCXYZ}\ \mathscr{abcxyz}$
Lucida Math	$\mathscr{ABCXYZ}\ \mathscr{abcxyz}$
TG Termes	$\mathscr{ABCXYZ}\ \mathscr{abcxyz}$

Another group favors a restrained style of Script:

Cambria Math	$\mathscr{ABCXYZ}\ \mathscr{abcxyz}$
LM Math	$\mathscr{ABCXYZ}$
Neo Euler	$\mathscr{ABCXYZ}$
TG Pagella	$\mathscr{ABCXYZ}\ \mathscr{abcxyz}$

Finally, several fonts also provide a Calligraphic as an alternative to Script (usually for uppercase only):

XITS Math	$\mathcal{ABCXYZ}$ (StylisticSet=1)
Lucida Math	$\mathcal{ABCXYZ}$ (StylisticSet=4)

It is noteworthy that Latin Modern Math currently does not provide the traditional Calligraphic style from Computer Modern as an alternative set, but that might be added in the future.

4.5 Choices of Fraktur (Blackletter)

Design choices of Fraktur alphabets are also similar among different fonts:

XITS Math	$\mathfrak{ABCXYZ}\ \mathfrak{abcxyz}$
Asana Math	$\mathfrak{ABCXYZ}\ \mathfrak{abcxyz}$
Cambria Math	$\mathfrak{ABCXYZ}\ \mathfrak{abcxyz}$
TG Termes	$\mathfrak{ABCXYZ}\ \mathfrak{abcxyz}$
TG Pagella	$\mathfrak{ABCXYZ}\ \mathfrak{abcxyz}$
LM Math	$\mathfrak{ABCXYZ}\ \mathfrak{abcxyz}$
Neo Euler	$\mathfrak{ABCXYZ}\ \mathfrak{abcxyz}$

In this case, LM Math, TG Pagella, and Neo Euler are all based on the same design of Euler Fraktur, whereas TG Termes is based on another source.

The only exception is Lucida Math, which features a completely different style of Blackletter:

Lucida Math	$\mathfrak{ABCXYZ}\ \mathfrak{abcxyz}$

This may be seen as a example that not every Blackletter font is equally well suited for use in math.

4.6 Choices of Blackboard Bold

Design choices of Blackboard Bold alphabets again fall into multiple groups. One group favors a serif design which is derived from the main serif font:

Cambria Math	𝔸𝔹ℂℕ𝕆ℙℚℝ𝕏𝕐ℤ 𝕒𝕓𝕔 𝟘𝟙𝟚
Asana Math	𝔸𝔹ℂℕ𝕆ℙℚℝ𝕏𝕐ℤ 𝕒𝕓𝕔 𝟘𝟙𝟚
Minion Math	𝔸𝔹ℂℕ𝕆ℙℚℝ𝕏𝕐ℤ 𝟘𝟙𝟚
TG Termes	𝔸𝔹ℂℕ𝕆ℙℚℝ𝕏𝕐ℤ 𝕒𝕓𝕔 𝟘𝟙𝟚
TG Pagella	𝔸𝔹ℂℕ𝕆ℙℚℝ𝕏𝕐ℤ 𝕒𝕓𝕔 𝟘𝟙𝟚

Another group favor a sans-serif design which may be unrelated to the main sans-serif font:

XITS Math	𝔸𝔹ℂℕ𝕆ℙℚℝ𝕏𝕐ℤ 𝕒𝕓𝕔 𝟘𝟙𝟚
Lucida Math	𝔸𝔹ℂℕ𝕆ℙℚℝ𝕏𝕐ℤ
LM Math	𝔸𝔹ℂℕ𝕆ℙℚℝ𝕏𝕐ℤ 𝕒𝕓𝕔 𝟘𝟙𝟚

Finally, the designs of individual letters can vary significantly among different math fonts, and are an additional consideration in font choice. For example, some users may have fairly strong preferences regarding such details as to whether the stem or the diagonal of the letter 'N' is double-struck.

4.7 Design issues of math alphabets

Besides the high-level design questions regarding the choice of matching typefaces for math alphabets to be assembled in an OpenType math font, there also some low-level design questions regarding the glyph shapes of individual typefaces.

In particular, we may ask: How should an upright Greek look, and how should a bold sans-serif Greek look compared to a bold serif Greek?

Unicode defines a number of math alphabets, many of which are supposed to come with a complete set of Latin and Greek for upper- and lowercase. This applies to all 4 shapes of the main serif typeface and to 2 out of 4 shapes of sans-serif.

4.8 Design of upright Greek alphabets

Unlike Unicode math, traditional TEX math fonts did not provide a complete set of Greek in all shapes.

Whereas uppercase Greek, just like uppercase Latin, came in several different shapes, including serif and sans-serif versions, lowercase Greek was only available in italics and bold italics.

As it turns out, creating an upright version of lowercase Greek by removing the slant while reusing the same designs of the italic version (as for Latin Modern Math) does not guarantee good results.

Comparing the results to other designs clearly shows that some letters in the unslanted Greek appear unbalanced, in particular for γ, ν, π, ϵ.

LM Math (upright = unslanted)

αβγδεζηθικλμνξοπρςστυφξψωϵϑφϱϖ
αβγδεζηθικλμνξοπρςστυφξψωϵϑφϱϖ

Cambria Math (upright = designed)

αβγδεζηθικλμνξοπρςστυφξψωεϑκφϱϖ
αβγδεζηθικλμνξοπρςστυφξψωεϑκφϱϖ

Obviously, font projects aiming for good results will need to take glyph design of individual math alphabets seriously, at which point a skilled font designer may be needed in addition to a font developer working on the technical aspects of font assembly.

4.9 Design of sans-serif Greek alphabets

Besides the design of upright Greek letter shapes, the design of sans-serif Greek alphabets may pose another challenge to font developers.

Strictly speaking, lowercase Greek letter shapes do not have serifs at all, so whether a Greek typeface design matches the look of a serif or sans-serif largely depends on matching the typical proportions and the typical stroke thickness of such fonts.

Usually, a sans-serif design exhibits a uniform stroke thickness, whereas a serif design exhibits contrast between thin or thick strokes, but the amount of contrast may vary between different fonts.

For the purposes of typesetting physics, letters from serif and sans-serif alphabets may be used next to each other to distinguish between different types of entities (vectors or tensors) by subtle differences in font shape (serif or sans-serif).

If the serif font exhibits a high contrast (as in the case of XITS Math), it is easy to tell apart from a sans-serif font, but if the serif font has a fairly uniform thickness itself (as in the case of Lucida Math), it becomes difficult to tell which one is which.

Lucida Math

αβγδεζηθικλμνξοπρςστυφξψωϵϑκφϱϖ
αβγδεζηθικλμνξοπρςστυφξψωϵϑκφϱϖ

XITS Math

αβγδεζηθικλμνξοπρςστυφξψωϵϑκφϱϖ
αβγδεζηθικλμνξοπρςστυφξψωϵϑκφϱϖ

Depending on the characteristics of the font, design of a clearly distinct sans-serif Greek may depend on more factors than just stroke thickness and may also require further adjustments to glyph shapes.

4.10 Technical issues regarding font metrics

Finally, besides achieving completeness and finding solutions to various design issues, there remain some technical issues to consider.

Most notably, there is an important difference in how glyph metrics are stored in OpenType math fonts as opposed to traditional TEX math fonts, and how those glyph metrics are interpreted in OpenType math engines following the reference behavior, such as LuaTEX (as opposed to XƎTEX).

In traditional TEX fonts, the actual glyph width used to be represented by the combination of the nominal width and the italic correction, but in Open-

Type fonts, the italic correction is disregarded, and only the nominal width is taken into account.

When converting traditional TEX math fonts to OpenType, it becomes necessary to adjust the glyph metrics to match the interpretation in OpenType math engines to ensure proper rendering in LuaTEX, while sacrificing the rendering in current versions of XƎTEX. (As of 2012, work on improving math typesetting in XƎTEX has been ongoing, so hopefully both engines will eventually adopt the same interpretation of glyph metrics.)

In recent developments, several font projects besides Cambria Math have adopted the OpenType interpretation of glyph metrics, such as Lucida Math and XITS Math, while others such as Latin Modern remain to be revised. Hopefully, other font projects will eventually follow to ensure consistent behavior when switching between different OpenType math fonts or different typesetting engines.

5 Conclusion

In this paper, we have reviewed recent progress in OpenType math font development as well as the many challenges faced by font developers of OpenType math fonts, including completeness of math symbols and math alphabets, design issues, and technical issues regarding the glyph metrics.

While significant progress has been made in recent years, resulting in recent or upcoming releases of several important OpenType math font projects, math font development remains a challenging task and more work remains to be done on developing new fonts or improving existing fonts.

References

[1] Barbara Beeton: Unicode and math, a combination whose time has come: Finally! *TUGboat*, 21(3), 174–185, 2000. `http://tug.org/TUGboat/tb21-3/tb68beet.pdf`

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

[3] Microsoft Typography: OpenType specification, version 1.6, 2009. `http://www.microsoft.com/typography/otspec/`

[4] Murray Sargent: High-quality editing and display of mathematical text in Office 2007. `http://blogs.msdn.com/murrays/archive/2006/09/13/752206.aspx`

[5] John Hudson, Ross Mills: Mathematical Typesetting: Mathematical and scientific typesetting solutions from Microsoft. Promotional Booklet, Microsoft, 2006. `http://www.tiro.com/projects/`

[6] Ulrik Vieth: Do we need a Cork math font encoding? *TUGboat*, 29(3), 426–434, 2008. Reprinted in *MAPS*, 38, 3–11, 2009. `http://tug.org/TUGboat/tb29-3/tb93vieth.pdf` `http://www.ntg.nl/maps/38/02.pdf`

[7] Ulrik Vieth: OpenType Math Illuminated. *TUGboat*, 30(1), 22-31, 2009. Reprinted in *MAPS*, 38, 12–21, 2009. `http://tug.org/TUGboat/tb30-1/tb94vieth.pdf` `http://www.ntg.nl/maps/38/03.pdf`

[8] Jonathan Kew: XƎTEX Live. *TUGboat*, 29(1), 151–156, 2008. `http://tug.org/TUGboat/tb29-1/tb91kew.pdf`

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

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

[11] Tiro Typeworks: Projects — Cambria Math. `http://tiro.com/projects.html`

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

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

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

[15] GUST: e-foundry. `http://www.gust.org.pl/projects/e-foundry`

[16] GUST: The `lm-math` font package. `http://www.ctan.org/pkg/lm-math`

[17] Johannes Küster: Minion Math 1.020. `http://typoma.de/en/fonts.html`

[18] Ulrik Vieth, Mojca Miklavec: Another incarnation of Lucida: Towards Lucida OpenType. *TUGboat*, 32(2), 169–176, 2011. `http://tug.org/TUGboat/tb32-2/tb101vieth.pdf`

[19] Karl Berry: Lucida OpenType fonts available from TUG. *TUGboat*, 33(1), 11, 2012.

http://tug.org/TUGboat/tb33-1/tb103lucida.pdf

[20] GUST: The tex-gyre-math font package. http://www.ctan.org/pkg/tex-gyre-math

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

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

[23] Tiro Typeworks: Fonts — Maxwell Math. http://tiro.com/fonts.html

[24] Barbara Beeton: The STIX Project: From Unicode to fonts. *TUGboat*, 28(3), 299–304, 2007. http://tug.org/TUGboat/tb28-3/tb90beet.pdf

[25] Will Robertson: Symbols defined by unicode-math. http://mirror.ctan.org/macros/latex/contrib/unicode-math/unimath-symbols.pdf

[26] Will Robertson: Unicode mathematics in LaTeX: Advantages and challenges. *TUGboat*, 31(2), 211–220, 2010. http://tug.org/TUGboat/tb31-2/tb98robertson.pdf

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

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

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

[30] Michael Sharpe: Math alphabets and the mathalpha package. *TUGboat*, 32(2), 164–168, 2011. http://tug.org/TUGboat/tb32-2/tb101sharpe.pdf

◇ Ulrik Vieth
Stuttgart, Germany
ulrik dot vieth (at) arcor dot de