The Arsenal Math OpenType fonts

Rajeesh KV

1 Background

The Arsenal OpenType font was introduced to *TUGboat* readers in [4]. During TUG 2025, Boris Veytsman asked this author if an OpenType math counterpart for the Arsenal text font could be developed. The OpenType math font he had been using together with Arsenal was KpMath-Sans.

Detailed knowledge of TEX's math typesetting and math font dimensions, OpenType format and its `MATH` table specification, and mastery of tools to aid the building of such a font are necessary to develop an OpenType math font from scratch. The OpenType math model extends the TEX math font dimensions. There are a number of helpful resources, including Barbara Beeton's introduction [1], the article "OpenType Math Illuminated" by Ulrik Vieth [5], the Noto Sans Math documentation by Khaled Hosny [2], and the specification [3] itself.

A handful of OpenType math fonts are freely available now, such as Asana Math, Libertinus Math, and others. It also helps to inspect and learn from their implementations.

2 Implementation

Armed with the resources mentioned in the previous section, the task of developing an Arsenal Math OpenType font could be achieved by using KpMath-Sans as the base math font — thus handling the complicated math dimensions/constants — and integrating Latin letters, numerals, some punctuation glyphs and symbols (for all the upright, italic, bold and bold italic glyphs as relevant) from the Arsenal text fonts into it. Appropriate scaling and stem thickness adjustments were required, especially for `script` and `scriptscript` style glyphs. Using an existing OpenType math font as the base considerably simplifies the effort, yet care must be taken for a good integration. Both of these base fonts are under the OFL[1] license, hence this amalgamation is legally allowed.

FontForge has good support for editing the OpenType `MATH` table. This is the tool chosen by this author for editing the glyphs, creating math kerning, and building the binary font.

Inspecting the KpMath-Sans font in FontForge is the starting point, to become familiar with the math constants defined (via menu `Element` → `Other Info` → `MATH Info`), and this font's glyph naming convention: e.g. `mupA` for upright A, `mitalpha` for italic α, `mupA.st` for `script` style $_{\mathrm{A}}$, `mitA.sst` for `scriptscript` style $_{A}$, etc. The `ssty` lookup rule links these base glyphs and their script/scriptscript glyphs. The `ssty` lookup with OpenType `script` tag '`math`' is *required* in addition to the `MATH` table, in order for `fontspec` to recognize it as a math font.

[1] OFL: `openfontlicense.org`

2.1 Glyph merging

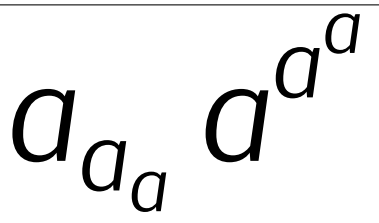
In order to copy glyphs properly from another font, one must open the target font and the 'source fonts' using the `Open` menu of the same window in FontForge. Care must be taken to copy and overwrite only the glyph outlines, not the Unicode codepoint, name, or other properties of the glyph; this can be ensured by unchecking the option `Glyph Metadata` via the menu `Edit` → `Copy From`. Then perform these steps:

1. Copy glyph outlines of `A`–`Z` and `a`–`z` from Arsenal to KpMath-Sans: including upright, italic, bold and bold italic, sans and monospace styles.
2. Copy glyph outlines of `0`–`9`: including upright, bold, oldstyle, and sans variants.
3. Copy some basic punctuation marks, such as `comma`, `colon`, `semicolon`; and some symbols such as `dollar`, `euro`, etc.

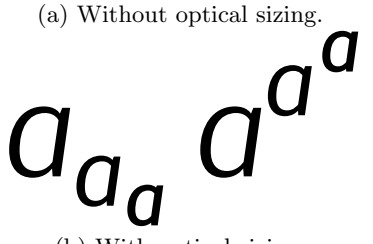
Even though the Arsenal Math font style in general is sans-serif, it still requires explicit sans-serif glyphs so that `\mathsf{A}` produces correct output glyph (for instance, the glyph named `msansA`).

2.2 Glyph scaling

Certain care must be taken for glyphs that get scaled (e.g. superscript and subscript glyphs), and also for bold style glyphs. The stroke widths of Arsenal and KpMath-Sans differ, thus weight adjustment is



(a) Without optical sizing.



(b) With optical sizing.

Figure 1: `script` and `scriptscript` style glyphs without and with optical size weight correction. Thickness of the base, sub- and super-script glyphs is uniform with optical size correction.

doi.org/10.47397/tb/46-3/tb144rajeesh-arsenalmath

necessary for such glyphs (see Fig. 1 below), leading to these further steps:

4. The `script` and `scriptscript` glyphs must be emboldened proportionally. They are named with the suffix `.st` and `.sst` respectively in the KpSans-Math base font. The OpenType lookup rule named `ssty` should be applied on the base glyph pointing to these corresponding styles. Those glyphs should be drawn with the same x-height as the base glyph, as they will be dynamically scaled down by the engine, by the math constants `ScriptPercentScaleDown` and `ScriptScriptPercentScaleDown` defined in the font.

   To compensate for the reduction in stroke width when scaling down, those glyphs are emboldened by `12` and `30 upem` (units per em) respectively, using the menu `Element` → `Style` → `Change Weight` in FontForge. Those glyphs might have shifted down from the baseline as a result of increasing the weight; they can be moved up to align on their baseline via the menu `Element` → `Transformations` → `Transform` and choose `Move` with `Y` shift by certain points, about `12 upem` in our case.
5. Glyphs of math bold (upright & italic) letters and numerals were also copied from corresponding Arsenal regular and italic glyphs, and their weight increased proportionally. Glyphs from Arsenal Bold were not used as those were thicker than KpMath-Sans bold glyphs, and reducing the weight of bold glyphs using FontForge (by a negative `em` value) produced unsatisfactory results. Increasing the weight of the regular-weight glyphs by a desired `em` unit, on the other hand, produced better glyphs.
6. The x-height of Arsenal glyphs is larger than KpMath-Sans glyphs. This caused collisions with accents on top of such glyphs. To fix this, the math constant `AccentBaseHeight` was reduced by `40 upem`, both in regular and bold fonts. This makes the typesetting engine place accents `40 upem` higher, thus avoiding collision.

2.3 Math kerning

Horizontal and vertical kerning of glyph pairs in OpenType are usually implemented using the GPOS feature — but not when it comes to math. Math kerning is enabled by a more advanced feature (also known as 'cut-ins' or 'staircase kerning') allowing for kerning adjustments at different heights of the four corners of a glyph (top-left, bottom-left, top-right, bottom-right).

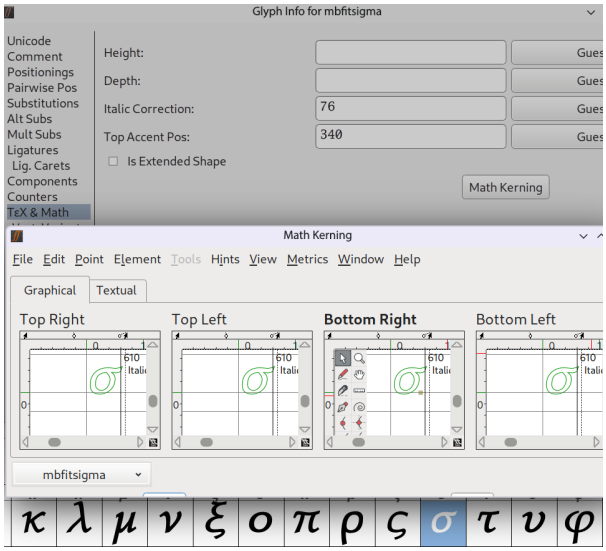


Figure 2: Editing math kerning in FontForge. Selecting each quadrant will show a visual indication of the kerning placement by the height of the glyph. The height-kerning value can also be set via the `Textual` tab.

$$\boldsymbol{\sigma}_C = \frac{\partial \psi_C}{\partial \boldsymbol{\Gamma}_C} \boldsymbol{\Gamma}_C^T$$

(a) Without math kerning.

$$\boldsymbol{\sigma}_C = \frac{\partial \psi_C}{\partial \boldsymbol{\Gamma}_C} \boldsymbol{\Gamma}_C^T$$

(b) With math kerning.

Figure 3: Greek symbols without and with math kerning. Note the kerning of subscripts to $\boldsymbol{\sigma}$, ψ, and $\boldsymbol{\Gamma}$.

KpMath-Sans defines math kerning for many glyphs, particularly for `\mathcal` and `\mathscr` letters, but it lacks kerning for Greek symbols. In Arsenal Math, kerning for many Greek symbols that need kerning is added to upright, italic, bold and bold italic glyphs. Math kerning can be added in FontForge either from the `Element` → `Other Info` → `MATH Info` → `Math Kern` menu for the entire font; or from the `Element` → `Glyph Info` → `TeX & Math` → `Math Kerning` menu for selected individual glyphs (see Fig. 2). The effect of math kerning (without & with) is demonstrated in Fig. 3.

2.4 Bold font

The ArsenalMath-Sans Bold font is created by integrating Latin glyphs from the Arsenal font into a copy

Suppose that α is a nonnegative real constant. We apply Proposition 3.5 with $\Phi(z) = \Phi_0(z)e^{\alpha|z|^2}$. If $u \in C_0^\infty(R^2 - \bigcup_\nu D_\nu(a))$, assume that $\mathscr{D}$ is a bounded domain containing the support of u and $A \subset \mathscr{D} \subset R^2 - \bigcup_\nu D_\nu(a)$. A calculation gives

$$\int_{\mathscr{D}} \left|\overline{\partial} u\right|^2 \Phi_0(z)e^{\alpha|z|^2} \geq c_4\alpha \int_{\mathscr{D}} |u|^2 \Phi_0 e^{\alpha|z|^2} + c_5\delta^{-2} \int_A |u|^2 \Phi_0 e^{\alpha|z|^2}.$$

The boundedness, property (1) of Φ_0, then yields

$$\int_{\mathscr{D}} \left|\overline{\partial} u\right|^2 e^{\alpha|z|^2} \geq c_6\alpha \int_{\mathscr{D}} |u|^2 e^{\alpha|z|^2} + c_7\delta^{-2} \int_A |u|^2 e^{\alpha|z|^2}.$$

Let $B(X)$ be the set of blocks of Λ_X and let $b(X) = |B(X)|$. If $\phi \in Q_X$ then ϕ is constant on the blocks of Λ_X.

$$P_X = \{\phi \in M \mid \Lambda_\phi = \Lambda_X\}, \qquad Q_X = \{\phi \in M \mid \Lambda_\phi \geq \Lambda_X\}. \tag{24}$$

If $\Lambda_\phi \geq \Lambda_X$ then $\Lambda_\phi = \Lambda_Y$ for some $Y \geq X$ so that

$$Q_X = \bigcup_{Y \geq X} P_Y.$$

Thus by Möbius inversion

$$|P_Y| = \sum_{X \geq Y} \mu(Y, X)\, |Q_X|\,.$$

Thus there is a bijection from Q_X to $W^{B(X)}$. In particular $|Q_X| = w^{b(X)}$.

Next note that $b(X) = \dim X$. We see this by choosing a basis for X consisting of vectors v^k defined by

$$v_i^k = \begin{cases} 1 & \text{if } i \in \Lambda_k, \\ 0 & \text{otherwise.} \end{cases}$$

```
\[v^{k}_{i}=
\begin{cases} 1 & \text{if $i \in \Lambda_{k}$},\\
0 &\text{otherwise.} \end{cases}
\]
```

Lemma 6.2. Let $\mathscr{A}$ be an arrangement. Then

$$\chi(\mathscr{A}, t) = \sum_{\mathscr{B} \subseteq \mathscr{A}} (-1)^{|\mathscr{B}|} t^{\dim T(\mathscr{B})}.$$

Figure 4: A page from `amsmath`'s `testmath.tex` typeset with Arsenal text and math fonts.

of the KpMath-Sans Bold font, and adding/improving math kerning. The general procedure is the same as for the Regular font, with the difference that Latin glyphs from Regular glyphs of Arsenal are made bold using FontForge. This approach is taken due to the fact that bold Latin glyphs from Arsenal Bold are thicker than KpMath-Sans Bold glyphs. FontForge can increase the weight of the Regular glyphs much better than reducing the weight of thick glyphs.

3 Availability

A typeset specimen, using Arsenal text and math fonts can be seen in Fig. 4.

The Arsenal Math font is licensed under the terms of the Open Font License (the same license as both of the fonts — Arsenal and KpMath-Sans — from which it is developed). The sources and binary fonts are freely available at the repository: `gitlab.com/rit-fonts/arsenal-math`.

To use the font in XeTeX or LuaTeX, one can add the following in the preamble:

```
\usepackage{unicode-math,fontspec}
\setmathfont[extension=otf,
  BoldFont=ArsenalMath-SansBold
]{ArsenalMath-Sans}
```

When using `unicode-math`, `\symbf` must be used instead of `\boldsymbol` for bold Greek symbols.

4 Acknowledgements

The author would like to thank Boris Veytsman for proposing the idea of developing a Math counterpart for the Arsenal font, and asking me to work on it. I would also like to thank the TeX development fund team who encouraged me to take up the work. For full disclosure, Veytsman donated the funding, but he had already paid me many times over by arranging for a copy of *The TeXbook* autographed by Don Knuth to me. Thus I requested the devfund team to use the fund for another worthy project.

Thanks to Daniel Flipo for the KpMath-Sans font, and Andrij Shevchenko for the Arsenal text font; both made available under the Open Font License. Vaishnavi Murthy reviewed the glyphs and provided many suggestions that helped to improve the font. Many thanks to Barbara Beeton and Karl Berry for the editorial corrections to the article.

References

[1] B. Beeton. Unicode and math, a combination whose time has come — finally! *TUGboat* 21(3), 2000. `tug.org/TUGboat/tb21-3/tb68beet.pdf`

[2] K. Hosny. Building OpenType math fonts. 28 Aug. 2024. `github.com/notofonts/math/blob/main/documentation/building-math-fonts/index.md`

[3] Microsoft. OpenType math — the mathematical typesetting table. 29 May 2024. `learn.microsoft.com/en-us/typography/opentype/spec/math`

[4] B. Veytsman. Packaging Arsenal fonts for (XeT/Lua)LaTeX. *TUGboat* 45(2), 2024. `doi.org/10.47397/tb/45-2/tb140veytsman-arsenal`

[5] U. Vieth. OpenType math illuminated. *TUGboat* 30(1), 2009. `tug.org/TUGboat/tb30-1/tb94vieth.pdf`

◇ Rajeesh KV
`rajeesh (at) rachana (dot) org (dot) in`
`https://gitlab.com/rit-fonts`