Dynamic Arabic Mathematical Fonts

Mostafa Banouni
mbanouni@voila.fr

Mohamed Elyaakoubi
m.elyaakoubi@ucam.ac.ma

Azzeddine Lazrek
lazrek@ucam.ac.ma
Department of Computer Sciences
Faculty of Sciences
University Cadi Ayyad
P.O. Box 2390
Marrakech, Morocco
http://www.ucam.ac.ma/fssm/rydarab

Abstract

This contribution describes a font family designed to meet the requirements of typesetting mathematical documents in an *Arabic presentation*. Thus, not only is the text written in an Arabic alphabet-based script, but specific symbols are used and mathematical expressions also spread out from right to left. Actually, this font family consists of two components: an Arabic mathematical font and a dynamic font. The construction of this font family is a first step of a project aiming at providing a complete and homogeneous Arabic font family, in the OpenType format, respecting Arabic calligraphy rules.
KEYWORDS: Mathematical font, Dynamic font, Variable-sized symbols, Arabic mathematical writing, Multilingual documents, Unicode, PostScript, OpenType.

1 Overview

The Arabic language is native for roughly three hundred million people living in the Middle East and North Africa. Moreover, the Arabic script is used, in various slightly extended versions, to write many major languages such as Urdu (Pakistan), Persian and Farsi (Iran, India), or other languages such as Berber (North Africa), Sindhi (India), Uyghur, Kirgiz (Central Asia), Pashtun (Afghanistan), Kurdish, Jawi, Baluchi, and several African languages. A great many Arabic mathematical documents are still written by hand. Millions of learners are concerned in their daily learning by the availability of systems for typesetting and structuring mathematics.

Creating an Arabic font that follows calligraphic rules is a complex artistic and technical task, due in no small part to the necessity of complex contextual analysis. Arabic letters vary their form according to their position in the word and according to the neighboring letters. Vowels and diacritics take their place over or under the characters, and that is also context dependent. Moreover, the *kashida*, a small flowing curve placed between Arabic characters, is to be produced and combined with characters and symbols. The *kashida* is also used for the text justification. The techniques for managing the *kashida* are similar to those that can be used for drawing curvilinear extensible mathematical symbols, such as *sum*, *product* or *limit*.

There are several Arabic font styles. Of course, it is not easy to make available all existing styles. The font style *Naskh* was the first font style adopted for computerization and standardization of Arabic typography. So far, only Naskh, Koufi, Ruqaa, and to a limited extent Farsi have really been adapted to the computer environment. Styles like Diwani or Thuluth, for example, don't allow enough simplification, they have a great variation in characters shapes, the characters don't share the same baseline, and so on. Considering all that, we have decided to use the Naskh style for our mathematical font.

The RyDArab [10] system was developed for the purpose of typesetting Arabic mathematical expressions, written from right to left, using specific symbols. RyDArab is an extension of the TEX system. It runs with K. Lagally's Arabic system ArabTEX [8] or with Y. Haralambous and J. Plaice's multilingual Ω [6] system. The RyDArab system uses characters belonging to the ArabTEX font `xnsh` or to the `omsea` font of Ω, respectively. Further Arabic alphabetic symbols in different shapes can be brought from the font `NasX` that has been developed, for this special purpose, using METAFONT. The RyDArab system also uses symbols from Knuth's Computer Modern family, obtained through adaptation to the right-to-left direction of Arabic.

Since different fonts are in use, it is natural that some heterogeneity will appear in mathematical expressions typeset with RyDArab [9]. Symbol sizes, shapes, levels of boldness, positions on the baseline will not quite be in harmony. So, we undertook building a new font in OpenType format with two main design goals: on the one hand, all the symbols will be drawn with harmonious dimensions, proportions, boldness, etc., and on the other hand, the font should contain the majority of the symbols in use in the scientific and technical writing based on an Arabic script.

Both Arabic texts and mathematical expressions need some additional variable-sized symbols. We used the CurExt [11] system to generate such symbols. This application was designed to generate automatically curvilinear extensible symbols for TEX with the font generator METAFONT. The new extension of CurExt does the same with the font generator PostScript.

While METAFONT generates bitmap fonts and thus remains inside the TEX environment, OpenType [14] gives outline and multi-platform fonts. Moreover, since Adobe and Microsoft have developed it jointly, OpenType has become a standard combining the two technologies TrueType and PostScript. In addition, it offers some additional typographic layout possibilities thanks to its multi-table feature.

2 A Mathematical Font

The design and the implementation of a *mathematical font* are not easy [5]. It becomes harder when it is oriented to Arabic presentation. Nevertheless, independent attempts to build an Arabic mathematical font have been undertaken. In fact, F. Alhargan [1] has sent us proofs of some Arabic mathematical symbols in TrueType format.

Now we will describe the way we constructed the OpenType Arabic mathematical font `RamzArab`. The construction of the font started by drawing the whole family of characters by hand. This task was performed by a calligrapher. Then the proofs were scanned to transform them into vectors. The scanning tools alone don't produce a satisfying result, so once the design is finalized, the characters are processed and analyzed using special software to generate the file defining the font.

In Arabic calligraphy, the feather's head (*kalam*) is a flat rectangle. The writer holds it so that the largest side makes an angle of approximately 70° with the baseline. Except for some variations, this orientation is kept all along the process of drawing the character. Furthermore, as Arabic writing goes from right to left, some boldness is produced around segments from top left toward the bottom right and conversely, segments from top right to the bottom left will rather be slim as in Figure 1.

The `RamzArab` font in Figure 4 contains only symbols specific to Arabic mathematics presentation plus some usual symbols found even in text mode. It is mainly composed of the following symbols:

- alphabetic symbols: Arabic letters in various forms, such as characters in isolated standard form, isolated double-struck, initial standard, initial with tail, initial stretched and with loop (e.g., ـب ـبـ ـنـ ـتـ ـثـ respectively);
- punctuation marks (e.g., ، ؛ : ! ؟);
- digits as used in the Maghreb Arab (North Africa), and as they are in the Machreq Arab (Middle East);
- accents to be combined with alphabetic symbols (e.g., ◌̂ ◌̃ ◌̈);
- ordinary mathematical symbols such as delimiters, arithmetic operators, etc.
- mirror image of some symbols such as sum, integral, etc.

In Arabic mathematics, the order of the alphabetic symbols differs from the Arabic alphabetic order. Some problems can appear with the alphabetic symbols in their multi-form.

Generally, in Arabic mathematical expressions, alphabetic symbols are written without dots (e.g., ◌̇ ◌̈ ◌⃛) or diacritics. This helps to avoid confusions with accents. The dots can be added whenever they are needed, however. Thus, few symbols are left.

Moreover, some deviation from the general rules will be necessary: in a mathematical expression, the isolated form of the letter ALEF can be confused with the Machreq Arab digit ONE. The isolated

form of the letter HEH can also present confusion with the Machreq Arab digit FIVE. The choice of the glyphs ۾ and ھ to denote respectively these two characters will help to avoid such confusions. Even though these glyphs are not in conformity with the homogeneity of the font style and calligraphic rules, they are widely used in mathematics. In the same way, the isolated form of the letter KAF ك, resulting from the combination of two other basic elements, will be replaced by the KAF glyph in Ruqaa style, ک.

For the four letters ALEF, DAL, REH and WAW, the initial and the isolated forms are the same, and these letters will be withdrawn from the list of letters in initial form. On the other hand, instead of a unique cursive element, the stretched form of each of the previous letters will result from the combination of two elements. It follows that these letters will not be present in the list of the letters in the stretched form.

The stretched form of a letter is obtained by the addition of a MADDA-FATHA or ALEF in its final form ـا to the initial form of the letter to be stretched (e.g., د + ـا ⟶ دا). The glyph of LAM-ALEF ﻻ has a particular ligature that will be added to the list. The stretched form of a character is used if there is no confusion with any usual function abbreviation (e.g., حا or حا for the sine function).

The form with tail is obtained starting from the initial form of the letter followed by an alternative of the final form of the letter HEH ـه (e.g., د + ـه ⟶ ـد). These two forms are not integrated into the font because they can be obtained through a simple composition.

The form with loop is another form of letters with a tail. It is obtained through the combination of the final form with a particular curl that differs from one letter to another (e.g., صع). This form will be integrated into the font because it cannot be obtained through a simple composition.

The following particular glyphs are also in use: ۾ ء ھ ر ے ك ک م ہ ء لا.

The elements that are used in the composition of the operator sum, product, limit and factorial in a conventional presentation (لـ ہـا حـد مـح) are added also. These symbols are extensible. They are stretched according to the covered expression, as we will see in the next section.

Reversed glyphs, with respect to the vertical — and sometimes also to the horizontal — axis, as in Figure 1, are taken from the Computer Modern font family. For example, there are:
∫ ∮ ∪ ∩ ∞ ∀ ∃ ⊂ ⊃ ∑.

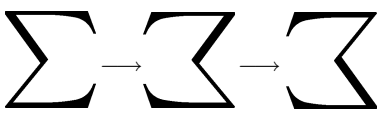

Figure 1: Sum symbol with vertical then horizontal mirror image

Other symbols with mirror image forms already in use[1] are not added to this font. Of course, Latin and Greek alphabetic symbols can be used in Arabic mathematical expressions. In this first phase of the project, we aren't integrating these symbols into the font. They can be brought in from other existing fonts.

3 A Dynamic Font

The composition of variable-sized letters and curvilinear symbols is one of the hardest problems in digital typography. In high-quality printed Arabic works, justification of the line is performed through using the kashida, a curvilinear variable lengthening of letters along the baseline. The composition of curvilinear extensible mathematical symbols is another aspect of dynamic fonts. Here, the distinction between fixed size symbols and those with variable width, length, or with bidimensional variability, according to the mathematical expression covered by the symbol, is of great importance.

Certain systems [11] solve the problem of vertical or horizontal curvilinear extensibility through the a priori production of the curvilinear glyphs for certain sizes. New compositions are therefore necessary beyond these already available sizes. This option doesn't allow a full consideration of the curvilinearity of letters or composed symbols at large sizes. A better approach to get curvilinear letters or extensible mathematical symbols consists of parameterizing the composition procedure of these symbols. The parameters then give the system the required information about the size or the level of extensibility of the symbol to extend. As an example, we will deal with the particular case of the opening and closing parenthesis as *vertically* extensible curvilinear symbol and with the kashida as a *horizontally* extensible curvilinear symbol. This can be generalized to any other extensible symbol.

The CurExt system was developed to build extensible mathematical symbols in a curvilinear way.

[1] The Bidi Mirrored property of characters used in Unicode.

The previous version of this system was able to produce automatically certain dynamic characters, such as parentheses, using METAFONT. In this adaptation, we propose to use the Adobe PostScript Type 3 format [13].

The PostScript language defines several types of font, 0, 1, 2, 3, 9, 10, 11, 14, 32, 42. Each one of these types has its own conventions to represent and to organize the font information. The most widely used PostScript font format is Type 1. However, a dynamic font needs to be of Type 3 [3].

Although the use of Type 3 loses certain advantages of Type 1, such as the possibility of producing hints for when the output device is of low resolution, and in the case of small glyphs, a purely geometrical treatment can't prevent the heterogeneity of characters. Another lost advantage is the possibility of using Adobe Type Manager (ATM) software. These two disadvantages won't arise in our case, since the symbols are generally without descenders or serifs and the font is intended to be used with a composition system such as TEX, not directly in Windows.

The PostScript language [7] produces a drawing by building a path. Here, a path is a set of segments (`lineto`) and third degree Bézier curves (`curveto`). The path can be open or closed on its origin (`closepath`). A path can contain several control points (`moveto`). Once a path is defined, it can be drawn as a line (`stroke`) or filled with a color (`fill`). From the graphical point of view, a glyph is a procedure defined by the standard operators of PostScript.

To parameterize the procedure, the form of the glyph has to be examined to determine the different parts of the procedure. This analysis allows determining exactly what should be parameterized. In the case of an opening or closing parenthesis, all the parts of the drawing depend on the size: the width, the length, the boldness and the end of the parenthesis completely depend on the size. Figure 2 shows the variation of the different parameters of the open parenthesis according to the height. We have chosen a horizontally-edged cap with a boldness equal to half of the boldness of the parenthesis. The same process is applied to the kashida.

Producing a dynamic parenthesis such as that in Figure 3 follows these steps:

- collecting the various needed sizes in a parameter file `par`;
- generating a file `pl` with the local tool `par2pl` starting from the `par` file;
- converting the file `pl` into a metric file `tfm` with the application `pltotf`;
- compiling the document to generate a `dvi` file;

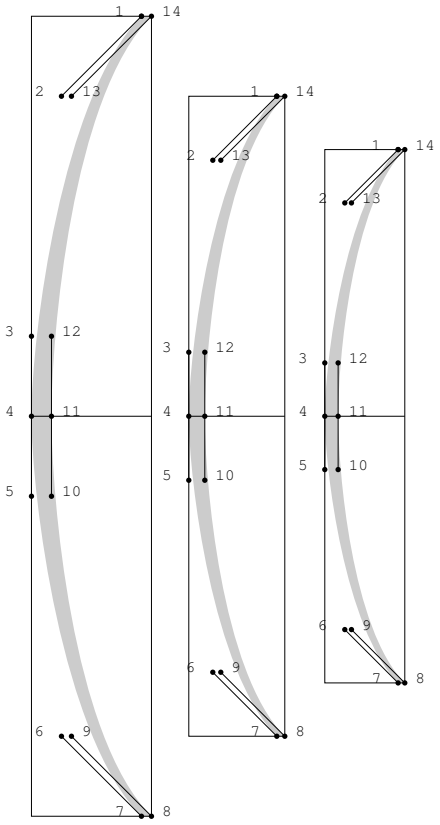

Figure 2: Parametrization of dynamic parenthesis

- converting the file from `dvi` to `ps` format.

This process should be repeated as many times as needed to resolve overlapping of extensible symbols.

The curvilinear parentheses is produced by CurExt with the following encoding:

```
$\parentheses{
\matrix{1 & 2 & 3\cr
        4 & 5 & 6\cr
        7 & 8 & 9\cr
        0 & 1 & 2\cr}
}$
```

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 0 & 1 & 2 \end{pmatrix}$$

instead of the straight parentheses given by the usual encoding in TEX:

```
$\left(
 \matrix{1 & 2 & 3\cr
         4 & 5 & 6\cr
         7 & 8 & 9\cr
         0 & 1 & 2\cr}
 \right)$
```

$$\left(\begin{matrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 0 & 1 & 2 \end{matrix}\right)$$

In the same way, we get the curvilinear kashida with CurExt:

```
\amarabmath
 ${\csum_{b=T-1}^{s}}$
```

$$\sum_{b=ط-1}^{ص} ب$$

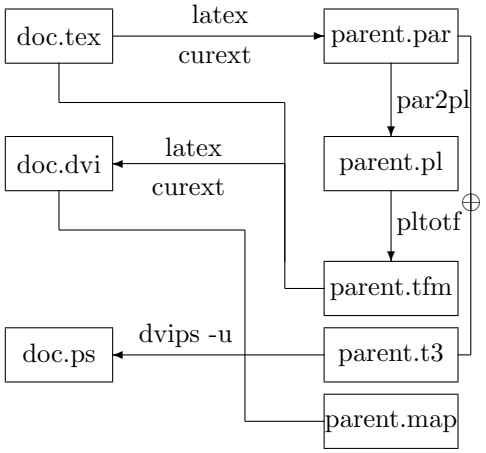

Figure 3: Generation of dynamic parentheses

instead of the straight lengthened one obtained by RyDArab:

```
\amarabmath
${\lsum_{b=T-1}^{s}}$
```

We can stretch Arabic letters in a curvilinear way through the kashida by CurExt:

مجـــــد مجـــد مجد

4 Conclusion

The main constraints observed in this work were:

- a close observation of the Arabic calligraphy rules, in the Naskh style, toward their formalization. It will be noticed, though, that we are still far from meeting all the requirements of Arabic calligraphy;
- the heavy use of some digital typography tools, rules and techniques.

RamzArab, the Arabic mathematical font in Naskh style, is currently available as an OpenType font. It meets the requirements of:

- homogeneity: symbols are designed with the same nib. Thus, their shapes, sizes, boldness and other attributes are homogeneous;
- completeness: it contains most of the usual specific Arabic symbols in use.

These symbols are about to be submitted for inclusion in the Unicode standard. This font is under test for Arabic mathematical e-documents [12] after having been structured for Unicode [2, 4].

The dynamic component of the font also works in PostScript under CurExt for some symbols such as the open and close parenthesis and the kashida. That will be easily generalized to other variable-sized symbols. The same adaptation can be performed within the OpenType format.

References

[1] `http://www.linux.org.sa`.

[2] Jacque André, Caractères numériques : introduction, *Cahiers GUTenberg*, vol. 26, 1997.

[3] Daniel M. Berry, Stretching Letter and Slanted-baseline Formatting for Arabic, Hebrew and Persian with `ditroff/ffortid` and Dynamic PostScript Fonts, *Software–Practice & Experience*, no. 29:15, 1999, pp. 1417–1457.

[4] Charles Bigelow et Kris Holmes, Création d'une police Unicode, *Cahiers GUTenberg*, vol. 20, 1995.

[5] Yannis Haralambous, Une police mathématique pour la Société Mathématique de France : le SMF Baskerville, *Cahiers GUTenberg*, vol. 32, 1999, pp. 5–19.

[6] Yannis Haralambous and John Plaice, Multilingual Typesetting with Ω, a Case Study: Arabic, *Proceedings of the International Symposium on Multilingual Information Processing (Tsukuba)*, 1997, pp. 137–154.

[7] Adobe Systems Incorporated, *PostScript Language Reference Manual*, Second ed., Addison-Wesley, 1992.

[8] Klaus Lagally, *ArabTEX — Typesetting Arabic with Vowels and Ligatures*, EuroTEX'92 (Prague), 1992.

[9] Azzeddine Lazrek, Aspects de la problématique de la confection d'une fonte pour les mathématiques arabes, *Cahiers GUTenberg*, vol. 39–40, Le document au XXIe siècle, 2001, pp. 51–62.

[10] Azzeddine Lazrek, A package for typesetting arabic mathematical formulas, *Die TEXnische Komödie*, DANTE e.V., vol. 13. (2/2001), 2001, pp. 54–66.

[11] Azzeddine Lazrek, CurExt, Typesetting variable-sized curved symbols, EuroTEX 2003 preprints, pp. 47–71 (to appear in *TUGboat*).

[12] Mustapha Eddahibi, Azzeddine Lazrek and Khalid Sami, *Arabic mathematical e-documents*, International Conference on TEX, XML and Digital Typography (TUG 2004, Xanthi, Greece), 2004.

[13] Włodzimierz Bzyl, The Tao of Fonts, *TUGboat*, vol. 23, 2002, pp. 27–39.

[14] Thomas W. Phinney, *TrueType, PostScript Type 1 & OpenType: What's the Difference?*, Version 2.00 (2001).

Figure 4: RamzArab Arabic mathematical font