Computer Modern Typefaces as the Multiple Master Fonts

A.S. Berdnikov
Institute of Analytical Instrumentation
Rizsky pr. 26, 198103 St.Petersburg, Russia
berd@ianin.spb.su

O.A. Grineva
Institute of Analytical Instrumentation
Rizsky pr. 26, 198103 St.Petersburg, Russia
olga@ianin.spb.su

Introduction

Several years ago Adobe Inc. announced its new Multiple Master font format which enabled one to vary smoothly the font characteristics (say, *weight* from light to black, *width* from condensed to expanded, etc.) and create a unique font which suites the User's demands. Like many other "new" inventions in computer-assisted typography,[1] the roots of this idea can be found inside TEX — namely, in the Computer Modern font family created by D. Knuth in 1977 – 1985 [1]. These fonts are parametrized using 62 (!!!) parameters most of which are independent. It can be seen easily that it exceeds the flexibility of *any* multiple master font which has been created up to now or even *will be* created by somebody in future.

The METAFONT source code and the parameters for the *Computer Modern* series were created using the advice of such professional font designers as Hermann Zapf, Matthew Carter, Charles Bigelow and others. The METAFONT source code is designed so that the parameter files are separated from the main source code; all the parameters together with the relations between them are totally documented [1], and the font variations (provided that they have a name different from the original CM fonts) are encouraged by the author. The parametric representation of the parameters for canonical Computer Modern typefaces created by John Sauter and Karl Berry (SAUTER fonts [2]), and, on a different basis, by Jörg Knappen and Norbert Schwarz in *European Computer Modern* typefaces [3], enables one to vary smoothly the font size in a wide range without loosing high quality of the resulting fonts. All these facts provide the basis of the LATEX macros MFF.STY, which enables one to treat *Computer Modern typefaces* like multiple master fonts.

The MFF.STY macros follow the ideas implemented in the MFPIC package and allows specification of new fonts dynamically within a LATEX document without dealing with the details of METAFONT programming and without manual manipulations of *each* of 62 parameters used in METAFONT source files. Like MFPIC, the first pass of LATEX creates the METAFONT source file (substituting font `dummy` instead of the user-defined font), the METAFONT source file is processed by METAFONT, and at the second pass the generated font is used to format the document properly.

The user can vary the font shape smoothly between CMR, CMBX, CMSL, CMSS, CMTT and CMFF font families, specify the *weight*, *width*, *height* and *contrast* of the output font independently, and in addition he/she can play with the character characteristics so that the resulting output does not look like the canonical *Computer Modern* typefaces at all. Although originally the package was created for internal purposes to facilitate the investigation of the possibilities hidden inside Computer Modern source code, it can be useful for professional typographic purposes too.

Font series with the arbitrary design size

Suppose that there is a smooth approximation for *Computer Modern Roman* (CMR) which enables one to calculate the METAFONT font parameters for an *arbitrtary* design size even with weak TEX arithmetical capabilities. The desired font size is specified as the input parameter, all the internal calculations of the font parameters are performed by TEX, and the result is a METAFONT *ready-to-run* font header file for a new font. When this new font header file is processed by METAFONT, it can

[1] For example, *Microsoft Word* 6.0 was announced as the *first* program which enables to mark some place in the text by a special marker and then to refer to its position in a form: "see page ...".

be used in TEX documents like other generic TEX fonts.

This operation is performed by the command

`\FontMFF[`*fntscaled*`]{`*fntcmd*`}{`*filename*`}{`*size*`}`

The command *fntcmd* will switch to the desired font as is done by the LATEX commands `\bf`, `\sl`, `\sf`, etc. The file *filename*`.mf` will contain the METAFONT source for a new font (please, *never* use the font name which is already used in CM or DC or some other standard fonts!). The parameter *size* specifies the design size of the new font, and the optional parameter *fntscaled* specifies the additional scaling of this font in LATEX document. Using the commands described in the following sections the user can vary the shape of the font characters in a wide range.

It is interesting to compare the possibilities of this simplest form of parametrization of CMR fonts and the PostScript vector fonts. The nearly-proportional changing of the font dimensions with respect to the magnification parameter is the analog of the linear scaling of the PostScript fonts. The non-linear relationship of the inter-character spacing from the font size imitates the *tracking* mechanism implemented in PostScript fonts (which is not taken into account in most cases by text processors). The fact that the ratio height/width is a non-constant (and non-linear) function of the font size is a serious advantage of these pseudo-CMR fonts in comparison with the linearly scaled PostScript fonts since it enables one to make the font proportions more suitable for the human eye (it is well known that for good eye recognition, small letters are to be more expanded and have greater inter-character spacing).

There are at least *two* ready-to-use font approximations available from CTAN. The first one is the METAFONT SAUTER font package [2]. It uses the smooth functions composed from constant, linear and quadratic pieces which are constructed so that for *canonical* font sizes they produce nearly the same `*.mf` files as the ones used by the original *Computer Modern* typefaces. Although the latest version of SAUTER is dated 1992, and in 1995 the parameters of *Computer Modern* fonts were again slightly changed, it seems still to be the most reliable source of the fonts with intermediate design sizes.

The other approximation is realized in DC and TC fonts by by Jörg Knappen and Norbert Schwarz [3]. It is based on cubic splines — Lagrange cubic splines or canonical cubic splines — using the parameters of *Computer Modern* typefaces as the base points. Although generally piecewise-cubic functions produce good quality approximations, it is not so with the data extracted from *Computer Modern* METAFONT files. The plots of the parameters, with respect to the design size, are "noisy" functions with some abrupt jumps since these parameters were selected manually to optimize the font shape, not the mathematical plots. As a result the cubic smooth approximations obey parasitic local minima and maxima and do not work far outside the range of design sizes used as the base data points. The DC and TC fonts with intermediate font sizes are visually good even with these "mathematical" defects, but for some specific reasons these defects could work badly when implemented in MFF.STY.

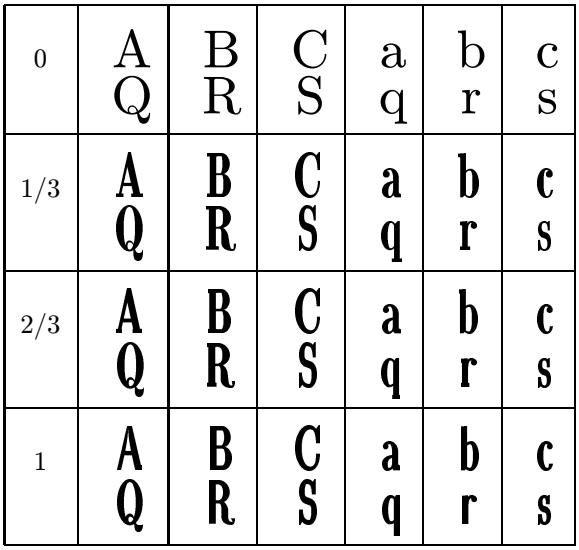

Figure 1: CMSS series

The first version of MFF.STY was based on piecewise-linear and piecewise-cubic (Lagrange splines) functions using *Computer Modern* typefaces as the reference data. To eliminate the parasitic local minima and maxima, some data points were slightly changed, and new data points were added to guarantee a good behavour of the approximating expressions outside the range `5pt` – `17.28pt`. The current version of MFF.STY is based on the SAUTER-type approximation with some modifications (especially for `cmff` and `cmfib` fonts).[2] As an option the piecewise-linear and the piecewise-cubic approximations based on DC data could be be included in futher versions of MFF.STY while the variated *Computer Modern* data used in the intermediate versions becames more-or-less obsolete.

[2] The reason to use SAUTER was the following: it is not a good idea to modify voluntary the original *Computer Modern* parameters.

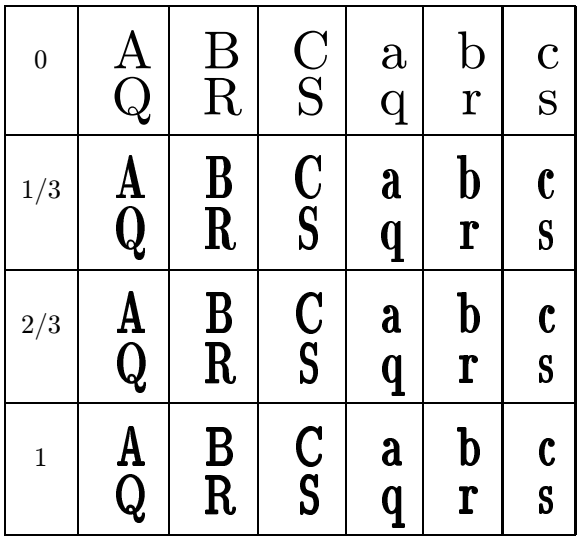

Figure 2: CMTT series

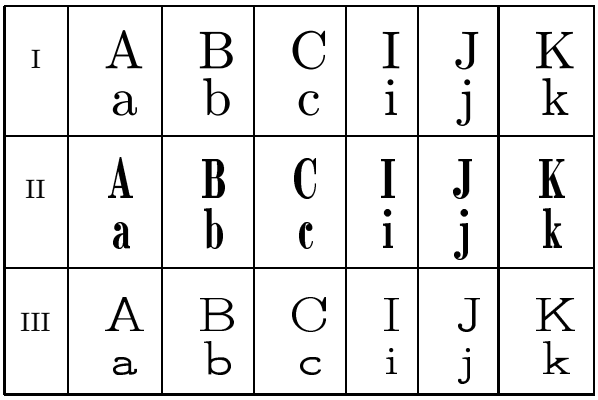

Figure 3: Font Modifications

Mixture of independent fonts

The *Computer Modern* fonts roman, **bold**, *slanted*, sans serif, typewriter, funny, dunhill, and quotation use the same METAFONT programs but with different values for font parameters. For each font series it is possible to construct the smooth approximations (similar to CMR font approximation mentioned in the previous section) which enables creation of a METAFONT header file with arbitrary design size.

Suppose that there *are* such smooth approximations, and it is possible to calculate for some fixed design size the parameters of *legal* CMR, CMBX, CMTT and CMSS fonts. All these fonts are generated by METAFONT without errors, and it can be supposed that the *weighted sum* of the parameters corresponding to these fonts also can be processed by METAFONT without error messages — at least we can expect it with good probability.

The CMTT fonts have nearly rectangular serifs, nearly no contrast between thin and thick lines, and are a little compressed in the vertical direction. The CMSS fonts have no serifs at all, their width is less than that of CMR fonts, and, although they also have no contrast, the thickness of their lines is greater as compared with CMTT. Other fonts have their own specific features, but in spite of this fact they can be "added" together — at least mathematically. The resulting font is no longer CMR, CMBX, CMTT, etc., but is something intermediate with a unique shape.

The CMBX fonts can be subdivided (to some extend artificially) into *two* independent font series — one for "boldness" (i.e., *weight*) and one for "extension" (i.e., *width*). It makes the total scheme more closely related to the NFSS realized in LaTeX2$_\epsilon$. So in MFF.STY the CMBX font sequence is decomposed into CMB′ (fonts which are as bold as CMBX and as wide as CMR, but which are different from the standard CMB10 font) and CMX (fonts which are as wide as CMBX and as bold as CMR).

The mixture of fonts is performed by the command `\MFFcompose{`α_1`}{`α_2`}...{`α_6`}` where $\alpha_1 - \alpha_6$ are some numerical values. The value α_1 corresponds to CMB, α_2 to CMX, α_3 to CMSS (sans serif font), α_4 to CMTT (typewriter font), α_5 to CMFIB ("Fibonacci" font), and α_6 to CMFF (funny font). If some parameter has the numerical value p_{cmr} for CMR font, p_{cmb} for CMB font with the same design size, etc., the *mixture* value is calculated as

$$\begin{aligned} p_* \;=\; & p_{cmr} + \alpha_1\,(p_{cmb} - p_{cmr}) + \alpha_2\,(p_{cmx} - p_{cmr}) \\ & + \alpha_3\,(p_{cmss} - p_{cmr}) + \alpha_4\,(p_{cmtt} - p_{cmr}) \\ & + \alpha_5\,(p_{cmfib} - p_{cmr}) + \alpha_6\,(p_{cmff} - p_{cmr}) \end{aligned}$$

It enables one, for example, to make the font "less bold" than CMR or "more bold" and "more extended" than CMBX by assignment of values which are less than 0 or greater than 1, and to create the "mutant" combinations of nearly incompatible font families. In Fig. 1 the result of a mixture of `cmr10` and `cmss10` is shown, and in Fig. 2 a similar series is constructed for `cmr10` and `cmtt10`.

Manual font modification

In addition to the weighted mixture of font ingredients it is possible to vary some font parameters which are responsible for the specific effects.

The inclination of the characters depends on the single font parameter `slant#`, and its value can be set explicitly as a ratio $\Delta y/\Delta x$ or as the inclination angle (specified in degrees).

The *width* of the CMR font can be varied by mixing with the CMX font, but it also can be

scaled explicitly by some factor specified by the user. Although this scaling skips some fine tuning of the font parameters, it can be advantageous to use it in some situations. For example, if we deal with CMTT fonts or CMSS fonts, mixing with CMX results also in some variation of the character shapes which can have an undesirable effect.

Similarly, the *weight*, i.e., the "boldness" of the characters, which can be controlled by mixing with CMB, can be defined also as a direct scaling of the font parameters which are responsible for this effect. The weight of "thin lines" and "thick lines" can be scaled independently, and in addition the user can specify explicitly the *contrast* — i.e., the ratio between thick and thin strokes of the characters.

The *height* of the vertical elements of the characters can be varied by the user with greater flexibility. That is, it is possible to scale independently

- the general height and the depth of the characters;
- the height of the capital characters, brackets, digits, etc., and the ascenders of the characters like 'b', 't';
- the depth of commas and the ascenders of the characters like 'Q', 'y';
- the height of digits and the position of the horizontal bar for mathematical signs like +, −.

If several height factors are specified, their effect is combined. For example, the font `cmdunh10` can be derived *exactly* from `cmr10` by proper specification of all these factors.

Finally, the special scaling factors can be used to produce the special effects:

- scale the fine connection between thin and thick lines in 'h', 'm', 'n';
- scale the thickness of sharp corners in letters 'A', 'V', 'w';
- scale the diameters of dots in 'i', ':' and bulbs in 'a', 'c';
- scale the curvature of the serif footnotes.

and the logical flags can control the level of ligatures and to switch on/off *square dots*, *sans serif* mode, *monospace* mode, etc.

Fig. 3 demonstrates the examples of such modifications:

- font (I) is the standard `cmr10` scaled at 1.8 times,
- font (II) differs from `cmr10` by scaling the width by 0.8, the height by 1.2 and the width of bold lines by 1.5;
- font (III) differs from `cmr10` by scaling the width by 1.2, the height by 0.8, the ascenders and descenders by 1.25, the width of bold lines by 0.8 and the width of thin lines by 2.4.

All the commands which allow performance of these modifications are described in MFF.STY manual in details.

Automatic check of font parameters

Each specification of MFF.STY parameters produces a unique font which belongs to the CMR/MF (Computer Modern Roman Master Font) font family and with a unique name specified by the user. Not all of these fonts are too pleasant, and not each variation of the parameters result to a font which is well-distinguished from the others. But at least it is an interesting toy for font maniacs.

There is a list of mutual relations between font parameters which are assumed implicitly in METAFONT programs for *Computer Modern* typefaces [1]. Although in reality most METAFONT source files *violate* these conditions, it is safer if the font parameters calculated by MFF.STY satisfy them. The command `\MFFcheck` sets the mode when these conditions are checked and the variable values are corrected if necessary. Nethertheless, several interesting effects can be achieved only without such automatic correction. The mode of automatic checking can be switched off by the command `\MFFcheck`.

Switch to other font classes

The LaTeX 2_ε NFSS classifies TeX font families in a way which is different from the logical structure of METAFONT programs. That is, the *italic* and SMALL CAPS are at the same family roman, together with **bold** and *slanted* fonts, although they are produced by a different driver files. Similarly, roman, `typewriter` and sans serif fonts are different families while they are generated by the same METAFONT script and their parameters can be varied so that one family is smoothly converted to another family.

In MFF.STY, there is no sharp boundary between roman, **bold**, *slanted*, `typewriter`, sans serif, quotation, funny and dunhill fonts — each font is smoothly converted to another one, while *italic* and SMALL CAPS fonts are quite different. The MFF.STY macros assign different *classes* to these fonts to distinguish such differences from the *font families* used in NFSS. The following font classes can be used:

CMR Computer Modern Roman;

CMTI Computer Modern Text Italic;

CMCSC Computer Modern Small Caps;

CMRZ *CMZ* Computer Modern Roman/Cyrillic by N. Glonti and A. Samarin;

CMRIZ *CMZ* Computer Modern Text Italic/Cyrillic;

CMCCSC *CMZ* Computer Modern Small Caps/Cyrillic;

LHR *LH* Computer Modern Roman/Cyrillic by O. Lapko and A. Khodulev;

LHTI *LH* Computer Modern Text Italic/Cyrillic;

LHCSC *LH* Computer Modern Small Caps/Cyrillic.

The interface for DC fonts [3] is under development.

The set of font classes can be extended easily when the METAFONT program is based on the same set of parameters as *Computer Modern* fonts: The only thing to do is to specify the macro which writes the `font_identifier` value and the operator `generate` with the corresponding file name.

Acknowledgements

The authors would like to express their warmest thanks to Kees van der Laan for organization and realization of the *Euro-Bus* project which enabled the Russian delegation to take part in *EuroTEX-95*, and to Ph.Taylor for his activity in breaking down the barriers between West and East.

This research was partially supported by a grant from the Dutch Organization for Scientific Research (NWO grant No 07-30-007).

References

[1] Donald E. Knuth. *Computer Modern Typefaces*, (*Computers & Typesetting* series). Addison-Wesley, 1986.

[2] John Sauter. *Building Computer Modern fonts. TUGboat*, **7** (1986), pp. 151 – 152.

[3] Jörg Knappen. *The release 1.2 of the Cork encoded DC fonts and the text companion symbol fonts.* Proceedings of the 9th EuroTEX Conference, Arnhem, 1995.

[4] A. Khodulev and I. Mahovaya. *On TEX experience in MIR Publishers.* Proceedings of the 7th EuroTEX Conference, Prague, 1992.

[5] O. Lapko. *MAKEFONT as a part of CyrTUG – EmTEX package.* Proceedings of the 8th EuroTEX Conference, Gdańsk, 1994.