⟨ Add '\bye' command to end of output and close both files 23 ⟩ Used in section 24.
⟨ Constants in the outer block 5 ⟩ Used in section 2.
⟨ Globals in the outer block 7, 12, 18 ⟩ Used in section 2.
⟨ Import List 4, 10, 11 ⟩ Used in section 2.
⟨ Initialize the file system 16 ⟩ Used in section 3.
⟨ Local variables for initialization 15 ⟩ Used in section 3.
⟨ Print statistics about line counts 26 ⟩ Used in section 24.
⟨ Print the job *history* 25 ⟩ Used in section 24.
⟨ Read the input 19 ⟩ Used in section 24.
⟨ Scan the line 21 ⟩ Used in section 20.
⟨ Search for '*.'; set *output_enabled* if found 22 ⟩ Used in section 21.
⟨ Set initial values 8, 17 ⟩ Used in section 3.
⟨ Terminate program, converting *history* to program exit status 9 ⟩ Used in section 24.
⟨ Types in the outer block 6 ⟩ Used in section 2.

Fonts

Blacker Thoughts

John S. Gourlay
Ohio State University

Like many owners of write-white laser printers, I found a few months ago that the "cm" series of Computer Modern fonts, as distributed, is unacceptably faint on my Xerox 2700. I began my search for more suitable METAFONT parameter settings with the "conjectural" settings for QMS printers, which share the same print engine as the 2700. I was immediately disappointed, however. Printed, the new bitmaps were acceptably black, but they didn't look anything like the Computer Modern in *Computer Modern Typefaces*, and not even very much like the original bitmaps printed on a write-black Canon engine.

Laser printers work by producing patterns of electric charge on a piece of paper. The charge attracts particles of black "toner," which eventually forms a permanent printed image. Write-black laser printers start with an uncharged piece of paper and in effect use a laser to place spots of charge on the paper. Write-white laser printers start with a fully charged piece of paper and then use a laser to remove the charge in places where the final image should remain white. In both cases the round spot produced by the laser is slightly larger than a pixel so that no gaps are left between spots in solid regions of black or white. For this reason, lines drawn on a write-black laser printer tend to be slightly thicker than one would expect given their width in pixels, and lines drawn on a write-white printer tend to be slightly thinner (the "white lines" are thicker).

The plain base file of METAFONT anticipates this kind of systematic difference between printers by providing a parameter called *blacker* whose value can be added to the thickness of pen strokes to compensate for any thinning inherent in the printing process. After some experimentation with various settings of *blacker* I decided empirically that the higher I made the value of *blacker* the smaller I found such lowercase letters as o and e to become. Also decreasing were the sizes of the bowls of such letters as p and b, the widths of m, n, and the lower part of h. The overall impression was that the "x-height" of the font was decreasing as *blacker* increased. At the conjectured setting of *blacker* = .75, the effect was great enough to make the font look entirely different and much less legible than the model in *Computer Modern Typefaces*.

Once I saw the problem it wasn't hard to see why it was happening. Looking at the METAFONT code for the roman lowercase o, one can see that it is drawn with a variable-width pen moving along a path through four points at the character's top, left, bottom, and right. Concentrating on point 1, the top point of the o, the relevant METAFONT statements are

$$penpos_1(vair, 90);$$

and

$$y_{1r} = h + \text{vround}\, 1.5oo;$$

The first says that the pen at point 1 has a nib of width *vair* and it is held vertically with the "right" edge of the nib at the top. The second says that the right (or top) edge of the pen should be at a distance $h + \text{vround}\, 1.5oo$ from the baseline. The parameter *blacker* figures into this because the pen width, *vair*, increases as *blacker* increases. Since the location of the top edge is fixed, an increase in *blacker* causes the whole pen to move down, and all the extra width appears at the bottom edge of the pen stroke. The same thing happens at the sides and bottom of the o, so the overall effect of an

increase in *blacker* is a decrease in the diameter of the path forming the o.

An increase in *blacker* should cause the pen width to increase, but it should not cause the pen's path to change. The extra width should be distributed equally on both sides of the stroke on the assumption that the printer will erode both sides of the stroke equally. Unfortunately, I do not see any easy way to modify the METAFONT code to make it behave this way. The best results would be obtained by moving all the points that are positioned relative to a pen edge outward by .5*blacker*. For example, we could change the second statement above to

$$y_{1r} - .5blacker = h + \text{vround}\, 1.5oo;$$

This would require an enormous amount of error prone work, however, because every point in the full set of Computer Modern fonts would have to be studied and a large proportion changed. A more tractable approach would be to remove all references to *blacker* in the definitions of pen widths and to modify such commands as **penstroke** and **filldraw stroke** to broaden their strokes by *blacker* automatically. This would limit the number of changes that would have to be made to the code, but it would have the disadvantage of nullifying the carefully planned rounding of pen widths, perhaps ruining the fonts in other ways.

Not wishing to tackle either of these projects immediately, I decided to live within the limits of the existing parameters. After many experiments I arrived at a compromise set of parameters:

$$blacker := .6;$$
$$fillin := -.3;$$
$$o\_correction := .6;$$

At this value of *blacker*, most of the characters keep their original sizes, but it is not quite enough to compensate for the thinning inherent in the printer. The rather extreme setting of *fillin*, which thickens diagonals, seems to correct the remaining faint spots. (At one point I tried to use *o_correction* to enlarge the shrunken bowls of *blacker* = .75, but with hindsight I should have known better. The results were not the kind of thing that I would take outside the privacy of my own home.) There are still some ugly features in the resulting fonts, particularly an inconsistency in the weights of characters. Nevertheless, I feel that this set of parameters is considerably better than the ones that result from the "conjectural" parameters, and also better than the "am" fonts they replace. There is room for improvement, however, as well as a challenge for the next generation of METAFONT designers.

Copies of these fonts ready for downloading to the Xerox 2700 can be obtained from me or preferably from

Margot Nelligan
Xerox Printing Systems Division
880 Apollo Street
El Segundo, CA 90245.

Updated Computer Modern Fonts for the LN03

John Sauter

Included with TUGboat volume 8 number 1 was the usual errata sheet for the TEX programs and documentation. Among the corrections to *Computers and Typesetting, volume E*, were changes to the parameters. The effect of these changes is to change the shapes of some of the Computer Modern characters.

I have been making available "alternative" versions of the Computer Modern parameter files since TUGboat volume 7 number 4, so these changes to volume E make my files obsolete. Fortunately, the files are easily fixed. Anyone who got a tape from me not marked METAFONT version 1.3 (or later) please make the following changes. I have taken some liberties with the spacing in order to fit the corrections into TUGboat's columns. You can use whatever spacing you wish when you change the files, except that a comment that starts with % must end on the same line as the %.

In COMPUTE_CMR.MF, starting at line 128, change four lines from

```
%elseif design_size < 12:
    ((design_size*15)+150)
else:  ((0.020812520812*
    design_size*design_size) +
(14.5421245421*design_size) +
(152.49750249))fi)/360pt#;
```

to

```
elseif design_size < 40:
    ((-0.23934398934*design_size*
    design_size) +
(20.265567765*design_size) +
(121.278721278))
else:  (548.951048934)fi)/360pt#;
```