My Way

May 2003

OpenType in ConTEXt

Adam T. Lindsay
Lancaster University

This is a summary of issues encountered and solutions implemented in order to support some advanced OpenType features in ConTEXt. This article describes an accompanying package that addresses installation (using TEXFONT), accommodating extended opticals families, and some "pro" font features. The extended character set afforded by pro fonts enables support for comprehensive small caps, old-style figures, and the use of greek glyphs to give workable math fonts. Although the typescripts and commands are described together, certain features (like variant encodings for TEXFONT and the idiosyncratic method for creating new math fonts) can be used independently of the other features described.

1 Introduction

This issue of My Way introduces some issues in installing, using, and integrating OpenType fonts in ConTEXt. OpenType has been discussed elsewhere in detail, but it should suffice to say that it is a modern melding of POSTSCRIPT and TrueType, enabling advanced typography features and Unicode support.

With some of the most complex OpenType fonts, one also sees integration with multiple design sizes. This is not necessarily unique to OpenType fonts, but is a co-occurring feature seen with "premium"-quality fonts. Some of the advanced typographical features seen in such fonts are integrated glyphs for SMALL CAPS (*across EVERY font variant*), old style and tabulation figures, and greek glyphs, all within the same font. This article introduces some strategies for taking advantage of the small caps and old style figures, and for making basic math fonts that include the greek glyphs where available.

It should be noted that this package owes its existence to Adobe's "Type Classics for Learning,"[1] an education-only package of very high quality OpenType fonts, for 99 USD. Many thanks to Bruce D'Arcus for pointing out the existence of this package, and his enthusiasm and cunning for encouraging me to do something with it. He also pointed me to the work of Achim Blumensath, whose efforts in the LATEX domain sparked my initial ideas about font installation. Otared Kavian provided the extended mathematics example. Obvious thanks to Hans Hagen and Don Knuth for providing such a rich foundation for this modest addition to the ConTEXt canon.

[1] see http://www.adobe.com/education/ed_products/typeclassics.html

2 Font Installation

The first issue to deal with is getting the fonts to be installed into the TeX tree. Early on, I decided that I should try to use TEXFONT, because it was clearly a ConTEXt-friendly tool, and, frankly, because I was slightly more familiar with it than other tools. It was pointed out that PfaEdit[2], by George Williams, was a powerful, freely-available, open-source, cross-platform tool that handled OpenType fonts as well as any other. As a result, using this package relies on you correctly installing PfaEdit on your machine.

I extended TEXFONT to include an optional pre-processing step that converts an OpenType font to a `.pfb` and `.afm` pair. From there, TEXFONT could work its usual magic.

The modified TEXFONT is included in the accompanying package as context/perltk /texfont.pl. The first additional option of interest is `--otf`. This command-line switch triggers a run of PfaEdit for each otf found in the current directory.

For example, if you wanted to install the Cronos Pro OpenType fonts, go into the directory containing the fonts, and issue the command (all on one line):

```
texfont --makepath --install --en=texnansi
         --otf --ve=adobe --co=CronosPro
```

If you prefer to work with batch files in TEXFONT, the following line should be a good place to start:

```
--en=? --ve=adobe --co=CronosPro --so=auto --otf
```

In order to support at least some of the many extended characters in a "Pro" font, another command-line option was added to TEXFONT, `--variant=<blah>` (abbreviated as `--va=`), which allows one `.enc` file to masquerade as another. For example, if I am working with the `texnansi` encoding, I can create a *variant* encoding that substitutes small caps for the lowercase letters. I name that encoding `texnansiSC.enc`, put it in a place where it can be found (like dvips/local), and run:

```
texfont --en=texnansi --va=SC --otf --ve=adobe --co=CronosPro
```

The `--variant` option appends the variant's name (`SC`, here) to the end of the name of the encoding (`texnansi`), and looks for that particular `.enc` file on the path. One variant, `texnansiOSFSC.enc`, is included in this package as a starter, in the dvips/local directory. It was the only variant I personally needed for extended support of the pro fonts, but you're welcome to create (and share!) your own.

It is worth noting that there is nothing about the `--variant` option that is tied to OpenType fonts. If you are working with any sort of font with extended glyphs (such as *Swash Caps*), you can create a font that accesses those extended glyphs (while masquerading as a known encoding) by using this method.

[2] see http://pfaedit.sourceforge.net/

3 Opticals

Most fonts that TEX users are familiar with include only two design axes, namely weight (e.g., light, regular, or bold) and shape (e.g., italic, slanted, or roman). Back in the days of metal type, each size of a given font had different design characteristics, because the eye is sensitive to different features at different scales. "Optical" fonts essentially add another design axis. This design axis was well-developed in the days of multiple master fonts, but since that technology appears to be dying, premium fonts are now being issued at discrete points along that axis. The font package that was used in the development of these macros and typescripts typically included four optical font sizes for each font: caption, regular, sub-head, and display. Their differences are shown in figure 3.1.

Figure 3.1 The four design sizes of Warnock Pro Opticals, shown at the same point size

At small design sizes (caption), there is typically lower contrast, a heavier stroke, a slightly larger x-height, and generally courser features. At large design sizes (display), strokes, tapers, serifs, and other details are much more refined.

This package supports these various design sizes with a series of extensive typescripts. It's essentially a brute-force method that defines font synonyms for each design size for each variant. That is, support for opticals is achieved by using a typescript that has small, regular, large, and extra-large fonts named for each of roman, italic, bold, and bold italic font variants, and adapting the `\tfa–\tfd` switching for each body font size.

For example, there are font synonyms declared for each of the following: SerifCaption, SerifText, SerifSubhead, SerifDisplay, SerifItalicCaption, SerifItalicText, SerifItalicSubhead, SerifItalicDisplay, and so on.... Each of these symbolic names is tied to font commands through the definition of a very large "Opticals" typescript, which generically associates a type variation and a type size with a design size. An extract follows:

```
\starttypescript [serif] [Opticals] [size]
\definebodyfont
  [12pt,11pt][rm]
  [tf=SerifText sa 1,
  tfa=SerifSubhead sa \magfactor1,
  tfb=SerifSubhead sa \magfactor2,
  tfc=SerifSubhead sa \magfactor3,
  tfd=SerifDisplay sa \magfactor4,
   it=SerifItalicText sa 1,
  ita=SerifItalicSubhead sa \magfactor1,
```

```
  itb=SerifItalicSubhead sa \magfactor2,
  itc=SerifItalicSubhead sa \magfactor3,
  itd=SerifItalicDisplay sa \magfactor4,
  ...]
\stoptypescript
```

As the bodyfont size changes (`12pt`, `11pt`, above), the relationships between the opticals change. This is handled in a huge typescript. In order to use this typescript yourself, you should define each of the SerifCaption ... SerifItalicDisplay synonyms in your own typescript, and include that with the Opticals typescript.

For example, I defined the various opticals for the Warnock font in a typescript called `WarnockProSiz`. I first created a typescript that associated the optical sizes with the actual font names as installed by TEXFONT:

```
\starttypescript [serif] [WarnockProSiz] [texnansi]
 \definefontsynonym[SerifCaption][texnansi-WarnockPro-Capt]
                   [encoding=texnansi,handling=pure]
 \definefontsynonym[SerifText][texnansi-WarnockPro-Regular]
                   [encoding=texnansi,handling=pure]
 \definefontsynonym[SerifSubhead][texnansi-WarnockPro-Subh]
                   [encoding=texnansi,handling=pure]
 \definefontsynonym[SerifDisplay][texnansi-WarnockPro-Disp]
                   [encoding=texnansi,handling=pure]
 ...
\stoptypescript
```

I then created a Warnock typescript to tie my size synonyms with the Opticals typescript:

```
\starttypescript[Warn]
 \usetypescript[serif][WarnockProSiz][texnansi]
 \usetypescript[serif][Opticals][size]
\stoptypescript
```

I then invoked the typescript at the top of this document, making sure to load the map files properly, as well:

```
\usetypescriptfile[type-atc]
\loadmapfile[texnansi-adobe-warnockpro.map]
\loadmapfile[texnansiOSFSC-adobe-warnockpro.map]
\usetypescript[Warn]
\switchtobodyfont[Warn,10pt]
```

4 Small Caps

Current support for SMALL CAPS, both inside and outside TEX, is generally very primitive. Most fonts – if they do offer it – only offer small caps support as a variation on the plain, roman font, and not for any italic/slanted fonts or bold fonts. This means that the small caps shape is of limited use with non-normal font alternatives (what ConTEXt calls `\it` and `\bf`). With a full complement of small caps shapes for each font alternative, small caps can be used more extensively.

The approach chosen for this package was to create another serif font style to exist inside the serif family, alongside the familiar roman (`rm`) style. The new font family, roman caps (`rc`) defines parallel font alternatives, using small caps variants. This means more work in terms of defining font synonyms, but it enables the small caps shape as a full design axis. The definitions begin as follows:

```
\definebodyfont
  [12pt,11pt][rc]
  [tf=SerifCapsText sa 1,
  tfa=SerifCapsSubhead sa \magfactor1,
  ...]
```

. . . and proceeds as the other definitions, above. There are sans serif equivalents defined, as well. The sans small caps family (`cs`, caps sans) is treated the same way. If you have installed a small caps type variant for a sans serif font, you should define and use this parallel family.

In order to use these font families, you may call them directly with macros like `{\rc\bf this}` (**THIS**). This is inconvenient, and requires you to recall the font alternative as well as the roman caps font style. To alleviate the inconvenience, this package defines a new font command, which switches from the normal style to the caps style while keeping the current alternative. The command is `\SmCap`, and it is used grouped, like other font commands. The first line, below, is achieved with the code:

```
{\it text {\SmCap text \em text} text \fontstyle\ \fontalternative}
```

text TEXT TEXT *text rm it*
text TEXT *TEXT* text rm bf
TEXT TEXT *TEXT* TEXT RC TF
text TEXT *TEXT* text ss tf

There is an identical command `\OldStyle`, which assumes that there are old style figures defined in the small caps family. It works in the same way as the above:

```
{\ss 0123456789 \OldStyle 0123456789}
```

0123456789 0123456789
0123456789 0123456789
0123456789 0123456789

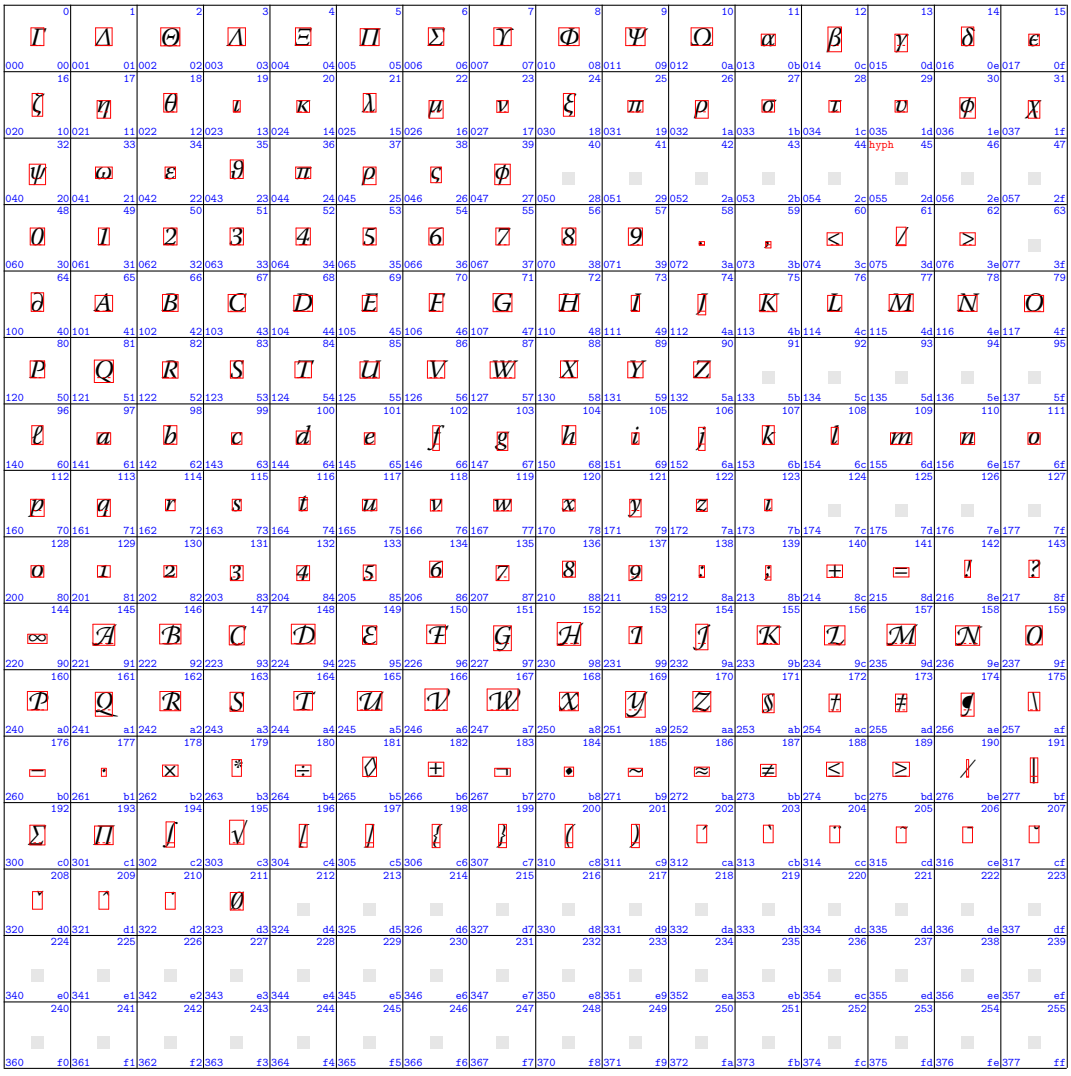
If you do not have fonts at different optical sizes, but you do have them in each of the font alternatives, you can use the roman caps and caps sans (\rc and \cs) type styles by including one or both of the following two lines in your typescript:

```
\usetypescript [serif] [romancaps] [size]
\usetypescript [sans] [sanscaps] [size]
```

. . . and defining the caps synonyms as suggested by those two typescripts.

5 Math

This package tries to take advantage of the greek characters included in certain OpenType Pro fonts. It does so by defining a new encoding that is applied to both roman and italic versions of a font. A math file then pulls the appropriate glyph from the math roman or math italic font, and makes it available to TEX. The encoding is shown for Warnock Math Italic.



name: math-atc-WarnockPro-It encoding: default mapping: default handling: default

As you can see, there is a strong complement of math-related glyphs. The encoding includes characters like old style figures and calligraphic letters (*Swash Caps*, taken from the italic font). The math-atc file consists of a math collection that is tuned for this pair

of fonts using this custom encoding. Where possible, the math collection pulls the glyph from the math roman/italic font. Otherwise, the character from the computer modern math font is used.

There exist some issues that probably prevent heavy-duty mathematics use. The biggest one is that spacing is still like a roman font. Math appears to be unkerned, normally, but characters in this font are "jammed together." It requires a level of font voodoo that I don't currently have. The second issue is that these fonts don't include a dotless j. Some of the math characters don't match perfectly with their computer roman cousins; I chose a scaling that is hopefully a plausible compromise. Let's see a sample for its suitability.

Here are some of the simple operations we can do with the O-notation:

$$f(n) = O(f(n)), \tag{5}$$
$$c.O(f(n)) = O(f(n)), \qquad \text{if } c \text{ is a constant,} \tag{6}$$
$$O(f(n)) + O(f(n)) = O(f(n)), \tag{7}$$
$$O(O(f(n))) = O(f(n)), \tag{8}$$
$$O(f(n))O(g(n)) = O(f(n)g(n)), \tag{9}$$
$$O(f(n)g(n)) = f(n)O(g(n)). \tag{10}$$

The O-notation is also frequently used with functions of a complex variable z, in the neighborhood of $z = 0$. We write $O(f(z))$ to stand for any quantity $g(z)$ such that $g(z) \le Mf(z)$ whenever $z < r$. (As before, M and r are unspecified constants, although we could specify them if we wanted to.) The context of O-notation should always identify the variable that is involved and the range of that variable. When the variable is called n, we implicitly assume that $O(f(n))$ refers to functions of a large integer n; when the variable is called z, we implicitly assume that $O(f(z))$ refers to functions of a small complex number z.

Suppose that $g(z)$ is a function given by an infinite power series

$$g(z) = \sum_{k\ge 0} a_k z^k$$

that converges for $z = z_0$. Then the sum of absolute values $\sum_{k\ge 0} a_k z^k$ also converges whenever $z < z_0$. If $z_0 \ne 0$, we can therefore always write

$$g(z) = a_0 + a_1 z + \cdots + a_m z^m + O(z^{m+1}). \tag{11}$$

For we have $g(z) = a_0 + a_1 z + \cdots + a_m z^m + z^{m+1}(a_{m+1} + a_{m+2} z + \cdots)$; we need only show that the parenthesized quantity is bounded when $z < r$, for some positive r, and it is easy to see that $a_{m+1} + a_{m+2} r + a_{m+3} r^2 + \cdots$ is an upper bound whenever $z \le r < z_0$.

For example, the generating functions listed in Section 1.2.9 give us many important asymptotic formulas valid when z is sufficiently small, including

$$e^z = 1 + \frac{1}{2!}z^2 + \cdots + \frac{1}{m!}z^m + O(z^{m+1}), \tag{12}$$

$$\ln(1+z) = z - \frac{1}{2}z^2 + \cdots + \frac{(-1)^{m+1}}{m}z^m + O(z^{m+1}) \tag{13}$$

$$(1+z)^\alpha = 1 + \alpha z + \binom{\alpha}{2} z^2 + \cdots + \binom{\alpha}{m} z^m + O(z^{m+1}), \tag{14}$$

$$\frac{1}{1-z}\ln\frac{1}{1-z} = z + H_2 z^2 + \cdots + H_m z^m + O(z^{m+1}), \tag{15}$$

for all nonnegative integers m. It is important to note that the hidden constants M and r implied by any particular O are related to each other. For ...

The normal math fonts switch to old style figures and calligraphic characters via a font switch. Since, with the newly-defined encoding, all of the characters are crammed into two fonts, and we do not create virtual fonts, the font switch approach doesn't work. As a result, there are two sets of "stupid" math commands that allow one to use these characters. They are not ideal, but until someone comes along with elegant virtual font support, they at least make the commands available.

The commands for old style figures in math mode are `\zero`, `\one`, ... `\nine`.

$$3 = 2\alpha + 5\beta$$

The commands for calligraphic capitals in math mode are `\calA`, `\calB`, ... `\calZ`.

$$\mathcal{A} = \mathcal{M} + \mathcal{N}$$

The math fonts used in this document don't take advantage of different design sizes (e.g., caption for scripts). That is mostly an oversight born of laziness. The underlying mechanism for taking advantage of caption and sub-head sizes is there, but untested.

In order to install a font for math usage, run TEXFONT on the regular and italic variants, using the math-atc encoding:

```
texfont --makepath --install --en=math-atc --otf
        --ve=adobe --co=WarnockPro WarnockPro-Regular
texfont --makepath --install --en=math-atc --otf
        --ve=adobe --co=WarnockPro WarnockPro-It
```

This will result in two fonts, named math-atc-WarnockPro-Regular and math-atc-WarnockPro-It, and a map file named math-atc-adobe-warnockpro.map. Make sure the map file is properly loaded by your source file, and that there is a typescript that properly associates MathRoman and MathItalic with your newly-created font files:

```
\starttypescript [math] [WarnockMath] [name]
 \definefontsynonym[MathRoman][math-atc-WarnockPro-Regular]
 \definefontsynonym[MathItalic][math-atc-WarnockPro-It]
\stoptypescript
```

Make sure that at some point you load the math-atc file:

```
\input math-atc
```

Finally, include the typescripts in your master typescript for the body font. If you do *not* define synonyms for MathRomanCaption, MathItalicSubhead, and the like, then you should use ATCMathFallback in your typescript as well:

```
\usetypescript[math][WarnockMath][name]
\usetypescript[math][ATCMathFallback][name]
\usetypescript[math][ATCMath][size]
\usemathcollection[atc]
```

6 Installation and Usage

The accompanying archive, OpenType.zip, is designed to be unpacked in a texmf tree, most likely texmf-local. If you have ConTEXt installed in that tree, be sure to back up your `texfont.pl` file first. The zip file installs a modified `texfont.pl` in the context /perltk directory.

There are two encoding files, `math-atc.enc` and `texnansiOSFSC.enc` installed in the dvips/local directory.

Finally, there are two ConTEXt files installed in the tex/context/third directory. `type-atc.tex` is the gigantic typescript that handles opticals, small caps, math, and the various other features and commands mentioned in this article. `math-atc.tex` contains the math collection that enables adapting to a math font.

After unzipping, make sure to run `texhash` or an equivalent to be sure that the files are found. Unless you are using the most spartan subset of the functionality described here, you will need to increase the font memory available to ConTEXt. Precise instructions vary with distribution.

From there, you should be able to install fonts, create your own typescripts linking names like SansCapsItalicText with the generated font files, and run your source file, using the `\usetypescriptfile[type-atc]` command. Don't forget to load your map files at the beginning of the document.

7 Known issues

The sheer number of fonts defined means that these typescripts are extremely memory-intensive. A modern computer should not struggle, as long as the font memory allocated to ConTEXt is sufficient.

The default compound hyphen that is used in composed–words is a hybrid character that doesn't work well in some pro fonts, including Warnock. Until someone comes up with a way of elegantly stretching the hyphen, it is best to replace the default compound character with a single hyphen (-), as I did in this article, with the command:

```
\setuphyphenmark[sign=-]
```

When using PfaEdit, I couldn't get the latest MacOSX version to work with Minion italic fonts. Rolling back to the previous version (Dec 2002) helped me. I doubt anyone else will encounter this problem.

This has only been tested with OpenType fonts that are of the POSTSCRIPT variety. Those that are TrueType kinds of OpenType fonts may act differently.

The Adobe Jenson Pro fonts really resisted any use with hanging alignment handling. TEX simply halted in that case. It's a bit of a mystery. As a result, the handling option is simply commented out in that typescript.

I don't define any of the typescripts in terms of Serif and SerifItalic, in order to avoid stomping on existing defaults. It causes a minor issue with files that attempt to conform to a pre-existing format, like this one. As a work-around, I defined the following synonyms to be sure the magazine title and margin decorations were set in the main font:

```
\definefontsynonym[Regular][SerifDisplay]
\definefontsynonym[RegularBold][SerifBoldSubhead]
```

source code of this document

```
\usemodule[mag-01]
\usemodule[abr-03]
\setvariables
  [magazine]
  [title={OpenType in \CONTEXT},
   author=Adam T.\ Lindsay,
   affiliation=Lancaster University,
   date=May 2003]

\startbuffer[abstract]
  This is a summary of issues encountered and solutions
  implemented in order to support some advanced OpenType
  features in \CONTEXT. This article describes an accompanying
  package that addresses installation (using \TEXFONT),
  accommodating extended opticals families, and some ``pro'' font
  features. The extended character set afforded by pro fonts
  enables support for comprehensive small caps, old-style
  figures, and the use of greek glyphs to give workable math fonts.
  Although the typescripts and commands are described
  together, certain features (like variant encodings for \TEXFONT\
  and the idiosyncratic method for creating new math fonts)
  can be used independently of the other features described.
\stopbuffer

\startbuffer[myscript]
\usetypescriptfile[type-atc]
\loadmapfile[texnansi-adobe-warnockpro.map]
\loadmapfile[texnansiOSFSC-adobe-warnockpro.map]
\usetypescript[Warn]
\switchtobodyfont[Warn,10pt]
\stopbuffer
\getbuffer[myscript]

\definefontsynonym[Regular][SerifDisplay]
\definefontsynonym[RegularBold][SerifBoldSubhead]
\setuphyphenmark[sign=-]
\setupalign[hanging]
\loadmapfile[math-atc-adobe-warnockpro.map]
\loadmapfile[texnansi-adobe-cronospro.map]
\loadmapfile[texnansiOSFSC-adobe-cronospro.map]

\hyphenation{Open-Type}
\def\PFAEDIT{{\SmCap PfaEdit}}

\starttext \setups [titlepage] \setups [title]

\chapter{Introduction}
```

source code of this document

```
This issue of My Way introduces some issues in installing,
using, and integrating OpenType fonts in \CONTEXT. OpenType
has been discussed elsewhere in detail, but it should suffice to
say that it is a modern melding of \POSTSCRIPT\ and TrueType,
enabling advanced typography features and Unicode support.

With some of the most complex OpenType fonts, one also sees
integration with multiple design sizes. This is not necessarily
unique to OpenType fonts, but is a co||occurring feature seen
with ``premium''||quality fonts. Some of the advanced
typographical features seen in such fonts are integrated glyphs
for {\SmCap small caps} ({\em across {\SmCap every} font variant}),
old style and tabulation
figures, and greek glyphs, all within the same font. This article
introduces some strategies for taking advantage of the small caps
and old style figures, and for making basic math fonts that include
the greek glyphs where available.

It should be noted that this package owes its existence to
Adobe{}'s ``Type Classics for Learning,''\footnote{see
\hyphenatedurl
{http://www.adobe.com/education/ed_products/typeclassics.html}}
an education||only package
of very high quality OpenType fonts, for {\SmCap 99 usd}. Many
thanks to Bruce D'Arcus for pointing out the existence of this
package, and his enthusiasm and cunning for encouraging me to
do something with it. He also pointed me to the work of
Achim Blumensath, whose efforts in the \LATEX\ domain
sparked my initial ideas about font installation. Otared Kavian
provided the extended mathematics example.
Obvious thanks to Hans Hagen and Don Knuth
for providing such a rich foundation for this modest addition
to the \CONTEXT\ canon.

\chapter{Font Installation}
The first issue to deal with is getting the fonts to be installed
into the TeX tree. Early on, I decided that I should try to use \TEXFONT,
because it was clearly a \CONTEXT||friendly tool, and, frankly, because
I was slightly more familiar with it than other tools. It was pointed
out
that \PFAEDIT\footnote{see
\hyphenatedurl{http://pfaedit.sourceforge.net/}},
by George Williams, was a powerful,
freely||available, open||source, cross||platform tool that handled
OpenType fonts as well as any other. As a result, using this package
relies on you correctly installing \PFAEDIT\ on your machine.
```

source code of this document

```
I extended \TEXFONT\ to include an optional
pre|-|processing step that converts an OpenType font to a
\type{.pfb} and \type{.afm} pair. From there, \TEXFONT\ could
work its usual magic.

The modified \TEXFONT\ is included in the accompanying
package as \hyphenatedfile{context/perltk/texfont.pl}. The
first additional option of interest is \type{--otf}. This command||line
switch triggers a run of \PFAEDIT\ for each otf found in the current
directory.

For example, if you wanted to install the {\ss Cronos Pro} OpenType
fonts, go into the directory containing the fonts, and issue the
command (all on one line):
\starttyping
texfont --makepath --install --en=texnansi
        --otf --ve=adobe --co=CronosPro
\stoptyping

If you prefer to work with batch files in \TEXFONT, the following line
should be a good place to start:
\starttyping
--en=? --ve=adobe --co=CronosPro --so=auto --otf
\stoptyping

In order to support at least some of the many extended characters in
a ``Pro'' font, another command||line option was added to \TEXFONT,
\type{--variant=<blah>} (abbreviated as \type{--va=}), which allows
one \type{.enc} file to masquerade as another. For example, if I am
working with the \type{texnansi} encoding, I can create a {\em
variant} encoding that substitutes small caps for the lowercase letters.
I name that encoding \type{texnansiSC.enc}, put it in a place where
it can be found (like \hyphenatedfile{dvips/local}), and run:
\starttyping
texfont --en=texnansi --va=SC --otf --ve=adobe --co=CronosPro
\stoptyping

The \type{--variant} option appends the variant's name (\type{SC},
here) to the end of the name of the encoding (\type{texnansi}), and
looks for that particular \type{.enc} file on the path. One variant,
\type{texnansiOSFSC.enc}, is included in this package as a starter,
in
the \hyphenatedfile{dvips/local} directory. It was the only variant
I personally needed for extended support of the pro fonts, but you're
welcome to create (and share!) your own.
```

source code of this document

```
It is worth noting that there is nothing about the \type{--variant}
option that is tied to OpenType fonts. If you are working with any
sort of font with extended glyphs (such as {\it
\mathematics{\calS}wash \mathematics{\calC}aps}), you can
create a font that accesses
those extended glyphs (while masquerading as a known encoding)
by using this method.

\chapter{Opticals}
Most fonts that \TEX\ users are familiar with include only two design
axes, namely weight (e.g., light, regular, or bold) and shape (e.g.,
italic,
slanted, or roman). Back in
the days of metal type, each size of a given font had different design
characteristics, because the eye is sensitive to different features
at
different scales. ``Optical'' fonts essentially add another design
axis.
This design axis was well||developed in the days of multiple master
fonts, but since that technology appears to be dying, premium fonts
are now being issued at discrete points along that axis. The font
package that was used in the development of these macros and
typescripts typically included four optical font sizes for each font:
caption, regular, sub||head, and display. Their differences are shown
in figure~\in[captiondisplay].

\placefigure[here][captiondisplay]
  {The four design sizes of Warnock Pro Opticals, shown at the same
  point size}
  {\startcombination[4*1]
   {\definedfont[SerifCaption sa 6]Ag}{caption}
   {\definedfont[SerifDisplay sa 6]Ag}{display}
   {\definedfont[SerifSubhead sa 6]Ag}{subhead}
   {\definedfont[SerifText sa 6]Ag}{regular}
  \stopcombination}

At small design sizes (caption), there is typically lower contrast,
a heavier stroke, a slightly larger x|-|height, and generally courser
features. At large design sizes (display), strokes, tapers, serifs,
and other
details are much more refined.

This package supports these various design sizes with a series of
extensive typescripts. It's essentially a brute||force method that
defines
font synonyms for each design size for each variant. That is, support
for opticals is achieved by using a typescript that has small, regular,
```

source code of this document

```
large, and extra||large fonts named for each of roman, italic, bold,
and
bold italic font variants, and adapting the \type{\tfa}--\type{\tfd}
switching for each body font size.

For example,
there are font synonyms declared for each of the following: Serif|*|Caption,
Serif|*|Text, Serif|*|Subhead, Serif|*|Display, Serif|*|Italic|*|Caption,
Serif|*|Italic|*|Text,
Serif|*|Italic|*|Subhead, Serif|*|Italic|*|Display, and so on\dots.
Each of these
symbolic names is tied to font commands through the definition
of a very large ``Opticals\/'' typescript, which generically associates
a type variation and a type size with a design size. An extract follows:
\startTEX
\starttypescript [serif] [Opticals] [size]
\definebodyfont
  [12pt,11pt][rm]
  [tf=SerifText sa 1,
  tfa=SerifSubhead sa \magfactor1,
  tfb=SerifSubhead sa \magfactor2,
  tfc=SerifSubhead sa \magfactor3,
  tfd=SerifDisplay sa \magfactor4,
   it=SerifItalicText sa 1,
  ita=SerifItalicSubhead sa \magfactor1,
  itb=SerifItalicSubhead sa \magfactor2,
  itc=SerifItalicSubhead sa \magfactor3,
  itd=SerifItalicDisplay sa \magfactor4,
  ...]
\stoptypescript
\stopTEX

As the bodyfont size changes (\type{12pt,11pt}, above), the
relationships between the opticals change. This is handled in
a huge typescript. In order to use this typescript yourself, you
should define each of the Serif|*|Caption \dots Serif|*|Italic|*|Display
synonyms in your own typescript, and include that with the
Opticals typescript.

For example, I defined the various opticals for the Warnock font
in a typescript called \type{WarnockProSiz}. I first created a
typescript that associated the
optical sizes with the actual font names as installed by
\TEXFONT:
\startTEX
\starttypescript [serif] [WarnockProSiz] [texnansi]
 \definefontsynonym[SerifCaption][texnansi-WarnockPro-Capt]
```

source code of this document

```
                    [encoding=texnansi,handling=pure]
 \definefontsynonym[SerifText][texnansi-WarnockPro-Regular]
                    [encoding=texnansi,handling=pure]
 \definefontsynonym[SerifSubhead][texnansi-WarnockPro-Subh]
                    [encoding=texnansi,handling=pure]
 \definefontsynonym[SerifDisplay][texnansi-WarnockPro-Disp]
                    [encoding=texnansi,handling=pure]
 ...
\stoptypescript
\stopTEX

I then created a Warnock typescript to tie my size synonyms
with the Opticals typescript:
\startTEX
\starttypescript[Warn]
 \usetypescript[serif][WarnockProSiz][texnansi]
 \usetypescript[serif][Opticals][size]
\stoptypescript
\stopTEX

I then invoked the typescript at the top of this document,
making sure to load the map files properly, as well:
\typebuffer[myscript]

\chapter{Small Caps}
Current support for {\SmCap small caps}, both inside and outside
\TEX, is generally very primitive. Most fonts -- if they do offer it
--
only offer small caps support as a variation
on the plain, roman font, and not for any italic||slanted fonts or
bold
fonts. This means that the small caps shape is of limited use with
non||normal font alternatives (what \CONTEXT\ calls \type{\it}
and \type{\bf}). With a full complement of small caps shapes
for each font alternative, small caps can be used more extensively.

The approach chosen for this package was to create another
serif font style to exist inside the serif family, alongside the
familiar roman (\type{rm})
style. The new font family, roman caps (\type{rc}) defines parallel
font alternatives, using small caps variants. This means more work
in terms of defining font synonyms, but it enables the small caps
shape as a full design axis. The definitions begin as follows:
\startTEX
\definebodyfont
  [12pt,11pt][rc]
  [tf=SerifCapsText sa 1,
```

```
  tfa=SerifCapsSubhead sa \magfactor1,
  ...]
\stopTEX
\dots and proceeds as the other definitions, above. There are sans
serif equivalents defined, as well. The sans small caps family (\type
{cs}, caps sans) is treated the same way. If you have installed a
small caps type variant for a sans serif font, you should define and
use this parallel family.

In order to use these font families, you may call them directly with
macros
like \type{{\rc\bf this}} ({\rc\bf this}). This is inconvenient,
and requires you to recall the font alternative  as well as the
roman caps font style. To alleviate the inconvenience, this
package defines a new font
command, which switches from the normal style to the caps style
while keeping the current alternative. The command is \type{\SmCap},
and it is used grouped, like other font commands. The first line, below,
is achieved with the code:
\startTEX
{\it text {\SmCap text \em text} text \fontstyle\ \fontalternative}
\stopTEX
{\it text {\SmCap text \em text} text \fontstyle\ \fontalternative}\crlf
{\bf text {\SmCap text \em text} text \fontstyle\ \fontalternative}\crlf
{\rc text {\SmCap text \em text} text \fontstyle\ \fontalternative}\crlf
%{text {\SmCap text \em text} text \fontstyle\ \fontalternative}\crlf
{\ss text {\SmCap text \em text} text \fontstyle\ \fontalternative}
%{\ssbi text {\SmCap text \em text} text \fontstyle\ \fontalternative}\crlf

There is an identical command \type{\OldStyle}, which assumes
that there are old style figures defined in the small caps family.
It works in the same way as the above:
\startTEX
{\ss 0123456789 \OldStyle 0123456789}
\stopTEX
{\ss 0123456789 \OldStyle 0123456789}\crlf
{\bi 0123456789 \OldStyle 0123456789}\crlf
{\itx 0123456789 \OldStyle 0123456789}\crlf

If you do not have fonts at different optical sizes, but
you do have them in each of the font alternatives, you can
use the roman caps and caps sans (\type{\rc} and
\type{\cs}) type styles by including one or both of
the following two lines in your typescript:
\startTEX
\usetypescript [serif] [romancaps] [size]
\usetypescript [sans] [sanscaps] [size]
```

source code of this document

```
\stopTEX
\dots and defining the caps synonyms as suggested by those
two typescripts.

\chapter{Math}
This package tries to take advantage of the greek characters included
in certain OpenType Pro fonts. It does so by defining a new encoding
that is applied to both roman and italic versions of a font. A math
file
then pulls the appropriate glyph from the math roman or math italic
font, and makes it available to \TEX. The encoding is shown for
Warnock Math Italic.

\showfont[math-atc-WarnockPro-It]

As you can see, there is a strong complement of math||related glyphs.
The encoding includes characters like old style figures and calligraphic
letters ({\it \mathematics{\calS}wash \mathematics{\calC}aps}, taken
from the italic font). The math-atc file consists of a math collection
that is
tuned for this pair of fonts using this custom encoding. Where possible,
the math collection pulls the glyph from the math roman||italic font.
Otherwise, the character from the computer modern math font is used.

There exist some issues that probably prevent heavy-duty mathematics
use. The biggest one is that spacing is still like a roman font. Math
appears to be unkerned, normally, but characters in this font are
``jammed together.'' It requires a level of font voodoo that I don't
currently have. The second issue is that these fonts don't include
a
dotless j. Some of the math characters don't match perfectly with their
computer roman cousins; I chose a scaling that is hopefully a plausible
compromise. Let's see a sample for its suitability.

\startnarrower
\hairline
\input math-knuth-shortest
\hairline
\stopnarrower

The normal math fonts switch to old style figures and calligraphic
characters via a font switch. Since, with the newly||defined encoding,
all of the characters are crammed into two fonts, and we do not
create virtual fonts, the font switch approach doesn't
work. As a result, there are two sets of ``stupid'' math commands
that allow one to use these characters. They are not ideal, but until
```

```
someone comes along with elegant virtual font support, they at least
make the commands available.

The commands for old style figures in math mode are \type{\zero},
\type{\one},  \dots \type{\nine}.
\startformula
\three = \two\alpha +\five\beta
\stopformula

The commands for calligraphic capitals in math mode are
\type{\calA}, \type{\calB}, \dots \type{\calZ}.
\startformula
\calA = \calM+\calN
\stopformula

The math fonts used in this document don't take advantage
of different design sizes (e.g., caption for scripts). That is mostly
an
oversight born of laziness. The underlying mechanism for
taking advantage of caption and sub||head sizes is there, but
untested.

In order to install a font for math usage, run \TEXFONT\ on the
regular and italic variants, using the math-atc encoding:
\starttyping
texfont --makepath --install --en=math-atc --otf
         --ve=adobe --co=WarnockPro WarnockPro-Regular
texfont --makepath --install --en=math-atc --otf
         --ve=adobe --co=WarnockPro WarnockPro-It
\stoptyping
This will result in two fonts, named math-atc-WarnockPro-Regular
and math-atc-WarnockPro-It, and a map file named
math-atc-adobe-warnockpro.map. Make sure the map file is
properly loaded by your source file, and that there is a
typescript that properly associates MathRoman and MathItalic
with your newly||created font files:
\startTEX
\starttypescript [math] [WarnockMath] [name]
 \definefontsynonym[MathRoman][math-atc-WarnockPro-Regular]
 \definefontsynonym[MathItalic][math-atc-WarnockPro-It]
\stoptypescript
\stopTEX

Make sure that at some point you load the math-atc file:
\startTEX
\input math-atc
\stopTEX
```

source code of this document

```
Finally, include the typescripts in your master typescript for
the body font. If you do {\em not} define synonyms for
MathRomanCaption, MathItalicSubhead, and the like, then you
should use ATCMathFallback in your typescript as well:
\startTEX
 \usetypescript[math][WarnockMath][name]
 \usetypescript[math][ATCMathFallback][name]
 \usetypescript[math][ATCMath][size]
 \usemathcollection[atc]
\stopTEX

\chapter{Installation and Usage}
The accompanying archive, OpenType.zip, is designed to be
unpacked in a texmf tree, most likely texmf|-|local. If you have
\CONTEXT\ installed in that tree, be sure to back up your
\type{texfont.pl} file first. The zip file installs a modified
\type{texfont.pl} in the \hyphenatedfile{context/perltk}
directory.

There are two encoding files, \type{math-atc.enc} and
\type{texnansiOSFSC.enc} installed in the
\hyphenatedfile{dvips/local} directory.

Finally, there are two \CONTEXT\ files installed in the
\hyphenatedfile{tex/context/third} directory.
\type{type-atc.tex} is the gigantic typescript that handles
opticals, small caps, math, and the various other features
and commands mentioned in this article. \type{math-atc.tex}
contains the math collection that enables adapting to a
math font.

After unzipping, make sure to run \type{texhash} or an
equivalent to be sure that the files are found. Unless you are
using the most spartan subset of the functionality described
here, you will need to increase the font memory available
to \CONTEXT. Precise instructions vary with distribution.

From there, you should be able to install fonts, create your
own typescripts linking names like SansCapsItalicText with
the generated font files, and run your source file, using
the \type{\usetypescriptfile[type-atc]} command.
Don't forget to load your map files at the beginning of the
document.

\chapter{Known issues}
The sheer number of fonts defined means that these
typescripts are extremely memory||intensive. A modern
```

source code of this document

```
computer should not struggle, as long as the font memory
allocated to \CONTEXT\ is sufficient.

The default compound hyphen that is used in
composed\compoundhyphen words is a hybrid character
that doesn't work well in some pro fonts, including Warnock.
Until someone comes up with a way of elegantly stretching
the hyphen, it is best to replace the default compound
character with a single hyphen (-), as I did in this article, with
the command:
\startTEX
\setuphyphenmark[sign=-]
\stopTEX

When using \PFAEDIT, I couldn't get the latest MacOSX version
to work with Minion italic fonts. Rolling back to the previous
version (Dec 2002) helped me. I doubt anyone else will encounter
this problem.

This has only been tested with OpenType fonts that are of the
\POSTSCRIPT\ variety. Those that are TrueType kinds of
OpenType fonts may act differently.

The Adobe Jenson Pro fonts really resisted any use with hanging
alignment handling. \TEX\ simply halted in that case. It's a bit of
a
mystery. As a result, the handling option is simply commented
out in that typescript.

I don't define any of the typescripts in terms of Serif and
Serif|*|Italic, in order to avoid stomping on existing defaults. It
causes
a minor issue with files that attempt to conform to a pre||existing
format, like this one. As a work||around, I defined the following
synonyms to be sure the magazine title and margin decorations
were set in the main font:
\startTEX
\definefontsynonym[Regular][SerifDisplay]
\definefontsynonym[RegularBold][SerifBoldSubhead]
\stopTEX

\setups [listing] \setups [lastpage] \stoptext
```

source code of this document