PKscale

`PKscale` changes the magnification of TEX `PK` files

Version 1.12

Every reasonable effort has been made to assure the quality and completeness of this program, if you have any questions, comments, or suggestions or if you believe that you have found a bug please contact the author at the address given at the end of this document.

Changes (what's new and exciting)

Version 1.12

- More minor documentation changes

Version 1.11

- Minor documentation changes

Version 1.1

- Initial release
- Added abbreviations for common \magsteps

Version 1.0a,b

- Beta test releases

Usage

PKscale's name may be misleading. PKscale changes some of the internal values in the PK file to make TEX believe that the font is a scaled version of some other font; however, PKscale does not, and cannot, change the actual size of the bitmapped characters in the font. For a better idea of how PKscale can be used, and what it actually does, consult the "Learning by Example" and "What's Really Going On" sections below. If you really want to resize bitmapped fonts, you can use Small Planet Software's Sfware package to accomplish the task. To resize TEX fonts, you will also need PKtoSFP and SFPtoPK.

PKscale is run from the DOS command prompt. In order to keep the program small and simple, a command-line interface has been chosen instead of something more user-friendly. The general format for running PKscale looks like this:

PKSCALE pkfile[.pk] ⟨tfmfile[.tfm]⟩ ⟨/option1:value1⟩ ... ⟨/optionn:valuen⟩

Where:

pkfile[.pk] is the name of the TeX PK file.

tfmfile[.tfm] is the name of the TeX TFM file. If no path is given, PKscale will search the tfmpath specified in the configuration file. By default, PKscale will use the root name of the PK file as the root name for the TFM file (i.e. the default TFM file when converting D:\PATH\MYPKFILE.PK is MYPKFILE.TFM). PKscale needs the TFM file in order to convert the PK file.

/option:value is one of the following options: /designsize, /scale, or /threshold. These options allow you to specify what scale of PK file to create and how large an error to allow before printing a warning message.

Options

/designsize:size

The /designsize option sets the design size of the PK font (in points). This value should reflect the base size of the font that you are scaling. For example, if you are creating a PK file that is a scaled version of CMR12, the design size is 12pt (you should specify /designsize:12).

/scale:scalefactor

The /scale option sets the scaling factor of the PK file. PKscale determines this value based upon information in the PK file and the requested /designsize, however, the computed value may not match the desired scale precisely. This can be caused, for example, by rounding errors in the conversion of a softfont file to a PK file by SFPtoPK. You should not use the /scale option to make gross changes in the scale factor since it is bound to produce an ugly, poorly spaced font.

The /scale option accepts MAGSTEPHALF (or simply H), MAGSTEP1...MAGSTEP5, and scale values of less than 10 as abbreviations for standard \magstep sizes.

`/threshold:`pixelcount

The /threshold value adjusts `PKscale`'s sensitivity to errors. If the difference between the character width in the `PK` file and the character width information in the `TFM` file is larger than the /threshold value, a warning message will be printed.

Learning by Example

If it is not immediately obvious how `PKscale` works, have no fear; a few examples should make it much clearer.

The examples that follow concentrate on making `PK` files in one or more of the standard \magstep sizes. Keep in mind that these are not sacred sizes, `PKscale` will allow you to create fonts that are scaled by any amount. If the concept of a scaled font is not clear, you might want to re-read chapter 4 of the TEXbook as a refresher.

Both of these examples assume that you are creating `PK` files from softfonts using the `SFPtoPK` program and that it is the `PK` files produced by `SFPtoPK` that you are scaling. This is by no means the only use for `PKscale` but it is the only use that I have ever encountered. Sure, you *could* use `PKscale` to create a version of `CMR5 scaled 2000` from `CMR10`, but what would be the point?

In the first example, suppose that we have HP Type Director or ZSoft TypeFoundry, or some other program capable of creating softfont files from scaleable outline fonts. Our task is to produce a group of `PK` files that represent a 10pt font scaled to all of the standard \magsteps.

Begin by creating softfont files at the following sizes: 10pt, 10.95pt, 12pt, 14.4pt, 17.28pt, 20.74pt, and 24.88pt. These softfonts represent the original 10pt font at each of the standard \magsteps (*i.e.* a 10pt font at \magstep1 is 1.2 times as large so it's a 12pt font, a 10pt font at \magstep2 is 1.2^2 times as large so it's a 14.4pt font, etc).

Next, convert the 10pt font into a `PK` file with `SFPtoPK` and convert the `PL` file created by `SFPtoPK` into a `TFM` file with the standard TEXware utility `PLtoTF`. The documentation included with `SFPtoPK` describes in greater detail how to convert softfonts into `PK` files. Now you have your 10pt font at \magstep0 completed. Move the `PK` file to wherever you keep your \magstep0 `PK` fonts and move the `TFM` file to wherever you keep your `TFM` files.

Convert the remaining softfonts into `PK` files with `SFPtoPK` but discard the `PL` files since these fonts won't have `TFM` files of thier own.

Beginning with the 12pt font, run `PKscale` to create the scaled `PK` file. The command-line for `PKscale` will look like this:

`PKSCALE` *pkfile tfmfilefrom10ptfont* `/designsize:10 /scale:magstep1`

The *pkfile* specified above will now be recognized by TEX as a scaled version of the 10pt font at \magstep1. The last thing you must do before you can use the font is rename it to *the same* name as your 10pt font and move it to wherever you keep your \magstep1 font files.

Convert the remaining `PK` files with `PKscale`, moving each scaled `PK` file the the appropriate directory for your implementation of TEX. When you have moved the last `PK` file, the fonts will be ready for use.

Here is the sequence of steps in psuedo-batch file format:

```
SFPTOPK TR1000 TR10 TR10
PLTOTF TR10
MOVE TR10.PK C:\TEXFONTS\LJ300DPI\TR10.PK
MOVE TR10.TFM C:\TEX\TFM\TR10.TFM
SFPTOPK TR1095 /KRN:NONE /MAP:NONE /LIG:NONE
SFPTOPK TR1200 /KRN:NONE /MAP:NONE /LIG:NONE
SFPTOPK TR1440 /KRN:NONE /MAP:NONE /LIG:NONE
SFPTOPK TR1728 /KRN:NONE /MAP:NONE /LIG:NONE
SFPTOPK TR2074 /KRN:NONE /MAP:NONE /LIG:NONE
SFPTOPK TR2488 /KRN:NONE /MAP:NONE /LIG:NONE
DEL TR*.PL
PKSCALE TR1095 /DESIGNSIZE:10 /SCALE:H
PKSCALE TR1200 /DESIGNSIZE:10 /SCALE:1
PKSCALE TR1440 /DESIGNSIZE:10 /SCALE:2
PKSCALE TR1728 /DESIGNSIZE:10 /SCALE:3
PKSCALE TR2074 /DESIGNSIZE:10 /SCALE:4
PKSCALE TR2488 /DESIGNSIZE:10 /SCALE:5
MOVE TR1095.PK C:\TEXFONTS\LJ329DPI\TR10.PK
MOVE TR1200.PK C:\TEXFONTS\LJ360DPI\TR10.PK
MOVE TR1440.PK C:\TEXFONTS\LJ432DPI\TR10.PK
MOVE TR1728.PK C:\TEXFONTS\LJ518DPI\TR10.PK
MOVE TR2074.PK C:\TEXFONTS\LJ622DPI\TR10.PK
MOVE TR2488.PK C:\TEXFONTS\LJ746DPI\TR10.PK
```

As another example, suppose that we have the standard CM fonts but we wish to manipulate them in some way; perhaps we would like to create outline or shaded versions of a particular font. Although few, if any, programs manipulate TEX `PK` fonts directly, many programs will edit HP softfonts. We can use `PKtoSFP`, `SFPtoPK`, and `PKscale` to take advantage of the HP softfont format to perform the manipulations.

Begin by using `PKtoSFP` to convert the `PK` files into HP softfonts. If you convert fonts at different magnifications, you should be careful to give each softfont a name that reflects its magnified size. For example, you might create `CMR8-H.SFP`, `CMR8-0.SFP`, `CMR8-1.SFP`, etc from the various `CMR8.PK` fonts. `PKtoSFP` will create HP softfonts at the correct point sizes based upon the scaling information in the `PK` file.

Make whatever changes you wish to the HP softfont files and follow the procedures detailed above to convert them back to `PK` files. Here is a short of example of what you might do:

```
PKTOSFP C:\TEXFONTS\LJ746DPI\CMR10 CMR10 CMR10-5
SFFX CMR10-5 CMRH10.5 HOLLOW
SFPTOPK CMRH10.5 /KRN:NONE /MAP:NONE /LIG:NONE
PKSCALE CMRH10 /DESIGNSIZE:10 /SCALE:MAGSTEP5
MOVE CMRH10.PK C:\TEXFONTS\LJ746DPI\CMRH10.PK
COPY C:\TEX\TFM\CMR10.TFM C:\TEX\TFM\CMRH10.TFM
DEL CMRH10*.*
```

Notice that I've used the same `TFM` file for both fonts because the `hollow` effect doesn't change any of the character widths.

What's Really Going On?

This section is probably for TEXnicians only.

PKscale changes the design size value in the PK font header to match the design size requested and makes the checksum value in the header equal to the checksum in the TFM file that PKscale is using. PKscale adjusts the hppp and vppp values in the header to make the font resolution appropriate for the scale requested.

For each character, PKscale changes the TFMWidth value to match the width in the TFM file that PKscale is using. If the difference between the TFMWidth value (expressed in pixels at 300dpi) and the DX value of the character exceeds the threshold, a warning message is issued but conversion continues.

This program *may* have implicit requirements that the PK files it manipulates be designed for 300×300dpi output devices. Since the only output device that I have is a LaserJet printer, I don't have any good way of testing this possibility. If you have a different kind of output device, and you experience difficulty with PKscale, let Small Planet Software know about the problem and (if time permits) corrective action will be considered.

Configuration File

To make PKscale easier to use, default parameters can be stored in a configuration file. The configuration file is a simple DOS text file. Each line in the file defines exactly one parameter. Every parameter has a program name, a parameter name and a value. PKscale searches for parameters by program name and parameter name. The program name is optional, if it is not specified the parameter matches all program names. This is a very general format with a full potential that is not realized by PKscale. This format is designed so that different programs can share the same configuration file and some or all of the same configuration parameters.

In cases where a parameter can be specified in *both* the configuration file *and* as a command line option, the command line option will take precedence.

If PKscale is run under DOS 3.00 or later, it looks for the configuration file in the same directory as the PKSCALE.EXE file. Otherwise, PKscale looks in the current directory. In either case, you can tell PKscale explicitly what configuration file to use by setting the DOS environment variable "PKSCALE" equal to the fully qualified name of the configuration file.

Configuration Parameters

PKSCALE TFMPATH=path

The TFMPATH is a DOS path string that PKscale uses to find the TFM file if you do not specify a path on the TFM filename. Currently, this is the only configuration variable that PKscale uses.

Bye Bye

I hope that you find `PKscale` useful. `PKscale` is absolutely free. You may copy it and give it away to anyone that you think might benefit from it. However, you *may not* sell it or profit from it's distribution in any way, shape, or form.

If you wish to contact the author, you may write to:

Norman Walsh
#42I Southwood Apartments
Brittany Manor Drive
Amherst, MA 01002
USA

or send electronic mail to:

walsh@cs.umass.edu

This electronic mail address requires access to Internet domains (through BITNET or UUCP, for example). This is possible from CompuServe and from several large national BBS systems as well as many colleges and universities.

Other Programs by Small Planet Software

`PKtoSFP`

`PKtoSFP` converts TEX `PK` fonts into HP LaserJet softfonts. `PKtoSFP` can produce HP AutoFont Support files to provide accurate spacing and kerning information to other applications. `PKtoSFP` is the inverse of `SFPtoPK`.

`SFPtoPK`

`SFPtoPK` converts HP LaserJet softfonts into TEX fonts. `SFPtoPK` can use AutoFont Support files to provide accurate spacing and kerning information to TEX. `SFPtoPK` is the inverse of `PKtoSFP`.

`MergeSFP`

Merges multiple LaserJet softfonts into a single file. If you are generating TEX fonts, you may discover that you need characters from several different symbol sets (and, hence, several different LaserJet softfonts) in order to create a complete TEX character set. `MergeSFP` allows you to construct a single LaserJet softfont containing the appropriate characters from several different softfonts.

Sfware

The Sfware utilities allow you to download, rotate, compress, expand, view, and perform special effects on softfonts. The effects provided include bold, fill, convert to fixed spacing, halftone, hollow, invert, mirror, outline, convert to proportional spacing, resize, reverse, shade, shadow, slant, stripe, tilt, three-d, hollow-three-d, and filled-three-d effects. The effects can be tailored and customized for any font with various parameters and shading patterns. Sfware is distributed under a shareware license agreement.

SFP2Auto

SFP2Auto reads HP LaserJet softfonts and produces HP AutoFont Support files directly. Many applications that cannot use softfonts directly, can install them with HP AutoFont Support. For example, this program allows you to install arbitrary softfonts into WordPerfect using only the PTR program!