METAPOST **Engines 1.4.7**

Nicola Vitacolonna ⟨**nvitacolonna@gmail.com**⟩

These engines are meant as a replacement for the default TEXShop engine for METAPOST. They allow you to typeset METAFONT or METAPOST files and to see the result in TEXShop preview window. Main features:

- support for METAFONT, METAPOST, MetaFun (shadows, transparencies, etc...);
- support for files containing multiple figures;
- support for METAPOST `outputtemplate` and deprecated `filenametemplate`;
- support for PDF output;
- support for SVG output (generates an HTML5 document);
- support for PNG output (generates an HTML5 document);
- usage of an `% !MPOST` directive to control the engine.

Requirements

- `perl`, `mpost`, `mptopdf`, `pdftex` (everything already there if you have Mac OS X and MacTEX).
- Well, TEXShop is not strictly a requirement, as the scripts can be run from the command line.

Installation

1. Close TEXShop.
2. Copy the files `nv-metapost.engine` and `nv-metafun.engine` into

   ```
   ~/Library/TeXShop/Engines
   ```

3. Open Terminal.app and type:

   ```
   cd ~/Library/TeXShop/Engines
   chmod +x nv-metapost.engine nv-metafun.engine
   ```

Usage

Open a METAFONT or METAPOST file in TEXShop, select “nv-metapost” or “nv-metafun” from the engine dropdown menu and typeset! (METAFONT files can be typeset only with “nv-metapost”.)

As for any other TEXShop document, you may put one of the following lines at the beginning of your file to have it typeset with the right engine automatically:

```
% !TEX TS-program = nv-metapost
```

or

```
% !TEX TS-program = nv-metafun
```

If METAPOST internal variable `outputformat`[1] is set to `"eps"` (which is the default) then a PDF preview of the resulting figures will be shown in TEXShop. If `outputformat` is set to `"svg"` or `"png"`, then a valid HTML5 file will be generated instead, which is best rendered by a browser such as Safari, Chrome, Opera or Firefox.

The `% !MPOST` directive

The engines can parse one or more `% !MPOST` directives[2] that occur no later than the first non-comment line of your source file (that is, somewhere *before* the first non-comment line). Each `% !MPOST` directive must be on a line by itself. The precise syntax of each directive is given below. For a summary of the possible directives, see **Table 1**.

Note. The `% !MPOST` directive is *not* a feature of `mpost`, `mptopdf`, TEX, or TEXShop: it is useful only in combination with `nv-metapost.engine` or `nv-metafun.engine`.

`% !MPOST` **Directive**	**Values**	**Notes**
`tex`	`etex`, `latex`, `texexec`, etc...	The value can be left empty.
`pdf`	`on`, `off`, `yes`, `no`, `0`, `1`	
`draft`	`on`, `off`, `yes`, `no`, `0`, `1`	Only with `nv-metafun`.
`preview`	`on`, `off`, `yes`, `no`, `0`, `1`	
`titles`	`on`, `off`, `yes`, `no`, `0`, `1`	`title` is a synonym.
`backgroundcolor`	`rgb`(r, g, b), `cmyk`(c, m, y, k), `gray`(g)	Color names can also be used.
`bin`	Path to the `mpost` binary	
`texmfcache`	Value of `TEXMFCACHE`	
`mem`	Path to `mem` file	Only with `nv-metapost`.
`numbersystem`	`scaled`, `double`, `binary`, `decimal`	MetaPost 1.800 or later only.

Table 1

- **Selecting the TEX processor**

  By default, `nv-metapost` uses `mpost`'s default TEX processor (`etex`) and `nv-metafun` uses ConTEXt's `texexec`. You can specify a different TEX processor using the `% !MPOST tex` directive, e.g., if you want to use `latex`, put

  ```
  % !MPOST tex = latex
  ```

  near the beginning of your file. If you use `nv-metafun.engine` and you want to choose the TEX processor with `%&` (e.g., you want to use `%&latex` inside `verbatimtex`) instead of using the method above, you must add the following line to your source file:

  ```
  % !MPOST tex =
  ```

[1] The variable `outputformat` and support for SVG output have been available since `mpost` 1.200. Support for PNG output has been added in version 1.800.
[2] The single space between `%` and `!MPOST` is optional.

without specifying any program name.[3] This will cause `mpost` to be run without the `-tex` option. See the *examples* folder for more examples.

- **Enabling/disabling the conversion to PDF**

  By default, the engines output figures in both EPS and PDF format.[4] If you don't want the conversion to PDF to take place, you may add the following line near the beginning of your source file:[5]

  ```
  % !MPOST pdf = off
  ```

  You may also choose edit an engine to disable PDF output permanently (see "**Engine Customization**"). In that case, PDF output can still be enabled on a per file basis by writing

  ```
  % !MPOST pdf = on
  ```

  near the beginning of the file.

> **Note for MetaFun users:** given the nature of the MetaFun format, disabling PDF output in `nv-metafun` will cause the preview not to render special features such as shadows and transparencies. On the other hand, compilation will be (much) faster. Hence, you may disable PDF conversion to quickly get a draft of your pictures, and enable it only when you want to see the final result. If you use `nv-metafun`, the directive
>
> ```
> % !MPOST draft = on
> ```
>
> has the same meaning as
>
> ```
> % !MPOST pdf = off
> ```
>
> and vice versa.

- **Toggling the preview on and off**

  If you don't need the preview, you may add the following line to your source file:

  ```
  % !MPOST preview = off
  ```

  Note that the preview is off by default when you typeset a `.tex` document (see the FAQ about `mfpic` on **page 7**); it is on by default otherwise. You may force the preview to be shown with

  ```
  % !MPOST preview = on
  ```

  It is not recommended to turn the preview on if you are using the `mfpic` LaTeX package, however, because some filename conflicts may prevent correct compilation.

- **Showing or suppressing the titles**

  By default, a single page preview has no title and no additional margins, while a multiple page preview shows a title for each figure and contains some extra margins. You may force the titles to be displayed unconditionally with

  ```
  % !MPOST titles = on
  ```

[3] It is not necessary to use this directive with `nv-metapost.engine`, which parses `%&` lines by default.
[4] Unless the user sets the output format to SVG or PNG, in which case no PDF is generated. See `metapost-svg.mp` in the *examples* folder.
[5] You may use `yes` or `1` as an alternative to `on`, and `no` or `0` as an alternative to `off`.

Similarly, you may suppress all the titles (and margins) unconditionally with

```
% !MPOST titles = off
```

Note that `nv-metapost.engine` and `nv-metafun.engine` produce slightly different multiple page previews when titles are off. Using the latter, each page has the size of the figure within; using the former, each page has the width and height of the figure(s) with maximum width and height.

- **Changing the background color in the preview**

  The `% !MPOST backgroundcolor` directive can be used to set a different background color in the preview. This may be useful to see how your figures look like over a certain background. For example,

  ```
  % !MPOST backgroundcolor = rgb(1,0,0)
  ```

  sets a red background,[6]

  ```
  % !MPOST backgroundcolor = cmyk(0,0,.4,0)
  ```

  sets a yellowish background, and

  ```
  % !MPOST backgroundcolor = gray(.8)
  ```

  sets a light gray background. In the above, the parentheses are optional (but the commas are not!). Besides the `rgb`, `cmyk` and `gray` color models, you may also use some color names, e.g.,

  ```
  % !MPOST backgroundcolor = red
  ```

  Which color names are available depends on the engine and the output format. If METAPOST internal variable `outputformat` is set to `"eps"`, then `nv-metapost.engine` uses the same names as provided by the LaTeX `color` package (including the 68 predefined internal colours of the `dvips` PostScript driver, e.g., `BurntOrange`, `Sepia`, etc...), while `nv-metafun.engine` uses the ConTeXt predefined names for colors. If `outputformat` is set to `"svg"`, you may use the standard **CSS color names** or the `rgb` model with arguments between 0 and 255 (see `metapost-svg.mp` in the *examples* folder).

  Note that `% !MPOST backgroundcolor` affects only the appearence of the preview: it does *not* modify your figures.

- **Changing the `mpost` executable**

  The `% !MPOST bin` directive can be used to explicitly specify the full path to `mpost`. For example, if you have a ConTeXt Minimals installation, say, in `/Users/me/ConTeXt`, you may use `mpost` from ConTeXt Minimals by writing in your source file something like:

  ```
  % !MPOST bin = /Users/me/ConTeXt/tex/texmf-osx-64/bin
  ```

  In some cases, you may want to adjust the value of the environment variable `TEXMFCACHE`. You can do so using the directive `% !MPOST texmfcache`, e.g.

  ```
  % !MPOST texmfcache = /Users/me/ConTeXt/tex/texmf-cache
  ```

- **Specifying a `mem` file**

  The `% !MPOST mem` directive can be used with `nv-metapost` to load a custom `mem` file. For example, to load `/Users/me/Desktop/custom.mem`, use

[6] Use `% !MPOST backgroundcolor = rgb(255,0,0)` if `outputformat` is set to `"svg"`.

```
% !MPOST mem = /Users/me/Desktop/custom
```

- **Setting the number system**

  Starting with MetaPost 1.800, MetaPost supports floating-point arithmetic through the `-numbersystem` command-line switch. You may set the number system using `% !MPOST numbersystem`, e.g.,

  ```
  % !MPOST numbersystem = double
  ```

Frequently Asked Questions

Q. I'm a TeX/LaTeX/ConTeXt user. Which engine should I use?

A. The short answer is: you can use either, possibly together with an `% !MPOST tex` specification (see above). The long explanation: both engines output EPS and PDF files that can be included in other documents. The main difference between the two engines is the METAPOST *format file* used. The `nv-metapost.engine` script uses Plain METAPOST and `nv-metafun.engine` uses MetaFun, which adds a bunch of useful macros. Another difference is that `nv-metapost.engine` uses `mpost`'s default TeX processor, while `nv-metafun.engine` uses `texexec`—but these defaults can be changed with `% !MPOST tex`.

That said, TeX and LaTeX users will mostly use `nv-metapost.engine`, and ConTeXt users `nv-metafun.engine`, simply because TeX and LaTeX users typically include `mpost` output directly in their documents, while ConTeXt users typically want the features provided by MetaFun and they want the textual labels in their figures to be typeset by ConTeXt (which `nv-metafun.engine` does by default). LaTeX users may use `nv-metafun.engine` by putting this line at the beginning of their files:

```
% !MPOST tex = latex
```

Then, they should include the resulting PDF files, not the EPS ones, in their documents, otherwise some special features, like shadows and transparencies, will get lost. As an alternative, they may still use `nv-metapost.engine` by including:

```
% !MPOST mem = metafun;
```

at the beginning of their source files (but the preview, in this case, will lack all MetaFun special features).

If you are still confused, then it may help to know that

- running `nv-metapost.engine` on `foo.mp` is equivalent to executing the following commands:

  ```
  mpost -jobname="foo" -recorder foo.mp
  mptopdf foo.0
  mptopdf foo.1
  ...
  ```

  where `foo.0`, `foo.1`, ... are assumed to be the EPS files output by `mpost`;
- running `nv-metafun.engine` on `foo.mp` is equivalent to executing

  ```
  mpost -mem=metafun -tex="texexec --dvi" -jobname="foo"
         -recorder foo.mp
  mptopdf foo.0
  mptopdf foo.1
  ```

```
...
```

which in turn is essentially equivalent to (yet more flexible than[7])

```
mptopdf foo.mp
```

Q. Please, I want the *same* behaviour as the original “mpost” option of TEXShop!

A. Edit `nv-metapost.engine` and make the following changes:

```
my $CONVERTPDF = 0;
my $PREVIEW = 0;
```

And if you are really allergic to `.fls` files, change the line

```
my $extra_options = '-recorder';
```

so that it looks like

```
my $extra_options = '';
```

Q. Give me back the functionality of the old “mptopdf” option of TEXShop!

A. Use `nv-metafun.engine`. Really. And if you are a LATEX user, read the next FAQ.

Q. I'm a LATEX user and I want to use `nv-metafun.engine`. But when I try, it fails to compile my LATEX labels, even when I use `%&latex`! But `mptopdf` can compile my file! What's going on?!

A. Since `nv-metafun.engine` mainly addresses ConTEXt users' needs (by trying to be as close as possible to `mptopdf`, but without some of its limitations), it uses `texexec` as the default TEX processor, ignoring any `%&` directive. You may change this behaviour on a per-file basis or permanently.

- On a per-file basis:

  put this line near the beginning of your file:

  ```
  % !MPOST tex =
  ```

  See also `metapost-mixed-labels` and `metafun-latex-label.mp` in the *examples* folder.

- Permanent change:

  edit `nv-metafun.engine` and change

  ```
  my $TEX = 'texexec --dvi';
  ```

  into

  ```
  my $TEX = '';
  ```

After that, any `%&` directive will be parsed as expected.

Q. I don't need all those PDF files, EPS output is just fine. How can I get rid of them?

[7] It is more flexible because it allows you to use `outputtemplate`, which would most probably confuse `mptopdf`. For other examples, search the *examples* folder for `metafun-mixed-labels.mp` and `metafun-latex-label.mp`.

A. To disable PDF output permanently, edit `nv-metapost.engine` and change the value of the variable `$CONVERTPDF` as follows:

```
my $CONVERTPDF = 0;
```

To enable/disable PDF output on a per file basis, use the `% !MPOST pdf` directive (see "**The `% !MPOST` directive**").

Q. I often compile files with 50+ figures while I actually only change the source code of at most two figures at a time. Most of the compiling time is spent on converting METAPOST's output to PDF. However, only 2 out of 50 figures change and produce different PDF files... Is there a way to skip PDF generation for those remaining 48 figures?

A. Determining which figures in your source file have changed since the last compilation is beyond the ability of my engines (and beyond the laziness of their programmer). An approximation of the result, however, can be obtained by using the `% !MPOST pdf` directive (see "**The `% !MPOST` directive**"). If you are using `nv-metafun`, then try `% !MPOST draft = on` (which is the same as `% !MPOST pdf = off`): special MetaFun features like shadows and transparencies will be lost in the preview, but compilation will be faster.

Q. I'm a LaTeX user. Why on earth would I need MetaFun?

A. Well, MetaFun is just a set of METAPOST macros, most of which are ConTeXt-independent and provide useful functionality. Examples of macros that I have used in the past and I have found invaluable are `externalfigure` (to include external images) and the macros for shadows and transparency—but there are *a lot* more: I recommend that you read the **MetaFun Manual**! You may also want to take a look at `metafun-latex-label.mp` in the *examples* folder, to see a specific instance of how LaTeX users can benefit from MetaFun.

Q. I'm a METAFONT user. How can I use your engines?

A. You may use `nv-metapost.engine` to generate a preview of a font, one glyph per page. And, of course, you may use METAFONT primitives for making `.tfm` files. The *examples* folder contains a couple of examples. Just keep in mind that your font is typeset by `mpost`, not by the original `mf` program (see the METAPOST **User's Manual** about the differences).

Q. Can your engines be used with the `mfpic` LaTeX package?

A. Yes. As for the original TeXShop script, the name in the `\opengraphsfile{...}` command must be the same as the `.tex` document name to use the following procedure. Just open your `.tex` document, run it once through LaTeX, then once through one my engines (note that no PDF preview will be shown in this case), then again through LaTeX. If needed, you may write `% !MPOST` directives at the beginning of your `.tex` source.

The compilation process may be simplified by using a custom engine that performs all the needed passes:[8]

```
#!/bin/tcsh
set path= ($path /usr/texbin /usr/local/bin $HOME/Library/TeXShop/Engines)
set filename = "$1"
pdflatex --shell-escape -synctex=1 "$filename"
nv-metapost.engine "$filename"
```

[8] Courtesy of Frank Pastor.

```
pdflatex --shell-escape -synctex=1 "$filename"
```

Q. Can I use LuaTEX with METAPOST?

A. Yes, you may use `dviluatex` or `dvilualatex`. Just specify the relevant engine at the beginning of your source file, e.g.,

```
% !MPOST tex = dviluatex
```

Search the *examples* folder for `luatex-label-test.mp`.

Q. Can I use XƎTEX with METAPOST?

A. Alas, not with my engines. If you want your labels typeset by XƎTEX, the only way I know is to embed your METAPOST code in a ConTEXt document, and compile it with

```
texexec --engine=xetex
```

Q. How about ConTEXt MkIV?

A. As far as I know, the `context` command outputs only PDF, so it is not possible to use it as a TEX processor in METAPOST. You may embed your METAPOST code in a ConTEXt document, though.

Q. How do I use `nv-metapost.engine` and `nv-metafun.engine` with **TeXworks**?

A. (Courtesy of Benoît Rivet) In TeXworks, create a new typesetting tool as described in **AdvancedTypesettingTools**. For example, for `nv-metapost.engine` (make sure that it is executable):

```
Program:   <path to nv-metapost.engine goes here>
Arguments: $fullname

[X] View PDF after running
```

If you use TeXworks under Windows, create a `nv-metapost.bat` file with the following content:

```
Rem Execute nv-metapost.engine, searching for it in the PATH
Rem Assume perl is in the PATH. If not, you should replace 'perl' by
Rem the full pathname of the perl executable, e.g., 'C:\perl\bin\perl.exe'

perl -S nv-metapost.engine %1
```

The TeXworks configuration becomes:

```
Program:   <path to nv-metapost.bat goes here>
Arguments: $fullname

[X] View PDF after running
```

Engine Customization

Some parameters can be configured in the scripts' source files. Most relevant to the end user are the following variables:

- `$TITLES`:

  determines whether each page of the preview should have a title and some added margins. This is set to 1 by default. Set this variable to 0 to suppress all titles and additional margins. The value of this variable can also be changed through the `% !MPOST titles` directive.

- `$ONENOTITLE`:

  determines whether a single page preview has a title and some additional margins or not. This is set to 1 by default (no title and no margins). If set to 0, then whether the title appears or not depends on the value of `$TITLES`. The value of `$ONENOTITLE` can also be changed through the `% !MPOST titles` directive.

- `$CONVERTPDF`:

  if set to 1, then the engine processes `mpost`'s output to additionally make a PDF file for each figure, suitable for inclusion in other documents. Requires `mptopdf`. This is set to 1 by default. Set it to 0 if you need only the standard METAPOST output. The value of this variable can also be changed through the `% !MPOST pdf` directive (or through the `% !MPOST draft` directive in `nv-metafun`).

- `$PREVIEW`:

  if set to 0, then the preview will not be shown. By default, this is set to 1. The value of this variable can also be changed through the `% !MPOST preview` directive.

- `$TEX`:

  the default TEX processor for textual labels. The value of this variable can also be changed through the `% !MPOST tex` directive.

- `$VERBOSE`:

  set this to 1 to output more verbose information.

Known Issues

When typesetting a document that makes use of the `mfpic` package, say `foo.tex`, the generated METAPOST file should be called `foo.mp` for `nv-metapost` (or `nv-metafun`) to recognize it. If the used engine has PDF conversion turned on (which is always the case for `nv-metafun` and the default for `nv-metapost`), the conversion process will cause an existing `foo.pdf` file to be deleted. This shouldn't be a problem, however, because typesetting `foo.tex` with LATEX again will re-generate the file.

This issue may also affect users of the LATEX `emp` package.

Version History

1.4.7

- Use `File::Copy` for portable file operations (thanks to Benoît Rivet).
- Minor updates to error messages.

1.4.6

- Added support for PNG output. If MetaPost is instructed to output images in PNG format, then the engines will generate an HTML5 preview of the images, which can be opened in any browser.

1.4.5

- Added `% !MPOST numbersystem` directive.

1.4.4

- Added an example of a custom engine to the README.
- Fixed a typo in `nv-metafun.engine`.

1.4.3

- When outputting SVG, generate a self-contained HTML5 document instead of using XHTML.
- Updated some examples and the README, to reflect recent changes in MetaFun. (See this discussion: `http://permalink.gmane.org/gmane.comp.tex.metapost/2312`.)

1.4.2

- Added the `%! MPOST mem` directive to `nv-metapost`.

1.4.1

- Added the `%! MPOST bin` directive to specify the full path to `mpost`.
- Added the `%! MPOST texmfcache` directive to set the environment variable `TEXMFCACHE`.
- The `%! MPOST pdf` directive works also in `nv-metafun`. The directive `%! MPOST draft` can also be used (`%! MPOST draft = on` is equivalent to `%! MPOST pdf = off`).

- In some examples, added a workaround for `mpost` 1.210 or later, which causes `%&` directives to be ignored in `TEX` commands.
- The engines print information about the paths to the executables and enviroment variables when debugging is enabled.

1.4.0

- Added support for SVG output. The engines can generate an XHTML file for previewing SVG figures, which can be opened by any SVG compliant browser.
- Improved preview layout. Figures are centered both horizontally and vertically in their page. Using `nv-metafun.engine`, each figure should now always appear in the same page as its title. A multipage preview without titles output by `nv-metafun.engine` causes each page to have the same size as the picture within, whereas using `nv-metapost.engine` all the pages have the height and width of the biggest figure(s).
- Now `% !MPOST` (with a single space between `%` and `!MPOST`) can be used instead of `%!MPOST`. This to be consistent with TEXShop syntax for `% !TEX`.
- New `% !MPOST pdf` directive, to toggle conversion to PDF on and off (`nv-metapost.engine` only).
- New `% !MPOST preview` directive, to toggle the preview on and off.
- New `% !MPOST backgroundcolor` directive, to select a background color for the preview.
- New `% !MPOST titles` directive, to show or hide all the titles.
- You can invoke a METAPOST engine on a `.tex` file. This will cause the engine to automatically search for an `.mp` file with the same name as the `.tex` document and to compile such METAPOST file. This workflow was introduced mainly to support people using the LATEX `mfpic` package and for compatibility with the original "mpost" option in TEXShop.
- The `mpost`'s log file is now renamed only if necessary.
- The preview in `nv-metafun.engine` is created using `texexec` instead of `context`.
- Added (untested) support for `troff` in `% !MPOST tex` (is anybody in the world still using `troff`?)
- Added a "debug" option in the source code, for debugging.
- Added some more example files.
- Updated documentation (be sure to read the FAQ!).

1.3.2b1

- Introduced the `%!MPOST` directive. In this release, it can be used to specify which TEX processor to use for the text labels.
- `nv-metafun.engine` uses `texexec` by default for the text labels.
- Before this release, converting a single figure (say, `foo.0`) into PDF would result in the file `foo.pdf`. Starting with this release, the PDF output is called `foo-0.pdf`, consistently with the behaviour in the general case (`foo.pdf` is still there for the preview, of course).

- The README file is now a PDF document, no more a text file, and it includes a new FAQ section.
- More examples (the `test` folder has been renamed `examples`).

1.3.0

This is the first official version, released under the GPL licence and bundled with TEXShop in MacTEX 2009.