

# METAPOST Engines 1.4.0

Nicola Vitacolonna <[nvitacolonna@gmail.com](mailto:nvitacolonna@gmail.com)>

These engines are meant as a replacement for the default T<sub>E</sub>XShop engine for METAPOST. They allow you to typeset METAFONT or METAPOST files and to see the result in T<sub>E</sub>XShop 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;
- usage of an `% !MPOST` directive to control the engine.

## Requirements

- `perl`, `mpost`, `mptopdf`, `pdftex` (everything already there if you have Mac OS X and MacT<sub>E</sub>X).
- Well, T<sub>E</sub>XShop is not strictly a requirement, as the scripts can be run from the command line.

## Installation

1. Close T<sub>E</sub>XShop.
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 T<sub>E</sub>XShop, 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 T<sub>E</sub>XShop 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`<sup>1</sup> is set to "eps" (which is the default) then a PDF preview of the resulting figures will be shown in T<sub>E</sub>XShop. If `outputformat` is set to "svg", then a valid XHTML file will be generated instead, which is best rendered by a browser such as Opera or Firefox.

## The % !MPOST directive

The engines can parse one or more % !MPOST directives<sup>2</sup> 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.

**Note.** The % !MPOST directive is *not* a feature of `mpost`, `mptopdf`, T<sub>E</sub>X, or T<sub>E</sub>XShop: it is useful only in combination with `nv-metapost.engine` or `nv-metafun.engine`.

- **Selecting the T<sub>E</sub>X processor**

By default, `nv-metapost` uses `mpost`'s default T<sub>E</sub>X processor (`etex`) and `nv-metafun` uses ConT<sub>E</sub>Xt's `texexec`. You can specify a different T<sub>E</sub>X 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 T<sub>E</sub>X 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 =
```

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

- **Enabling/disabling the conversion to PDF (`nv-metapost.engine` only)**

By default, `nv-metapost.engine` outputs figures in both EPS and PDF format.<sup>4</sup> 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:<sup>5</sup>

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

You may also choose to edit `nv-metapost.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 that, given the nature of the MetaFun format, disabling PDF output is not supported by `nv-metafun.engine`.

<sup>1</sup> The variable `outputformat` and support for SVG output have been available since `mpost` 1.200.

<sup>2</sup> The single space between % and !MPOST is optional.

<sup>3</sup> It is not necessary to use this directive with `nv-metapost.engine`, which parses `%&` lines by default.

<sup>4</sup> Unless the user sets SVG as the output format, in which case no PDF is made (the various T<sub>E</sub>X engines are not compatible with SVG). See `metapost-svg.mp` in the *examples* folder.

<sup>5</sup> You may use `yes` or `1` as an alternative to `on`, and `no` or `0` as an alternative to `off`.

- **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 6](#)); 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` L<sup>A</sup>T<sub>E</sub>X 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
```

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,<sup>6</sup>

```
% !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 L<sup>A</sup>T<sub>E</sub>X `color` package (including the 68 predefined internal colours of the `dvips` PostScript driver, e.g., `BurntOrange`, `Sepia`, etc...), while

---

<sup>6</sup> Use `% !MPOST backgroundcolor = rgb(255,0,0)` if `outputformat` is set to `"svg"`.

`nv-metafun.engine` uses the ConT<sub>E</sub>Xt 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 appearance of the preview: it does *not* modify your figures.

## Frequently Asked Questions

**Q.** I'm a T<sub>E</sub>X/L<sup>A</sup>T<sub>E</sub>X/ConT<sub>E</sub>Xt 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 T<sub>E</sub>X processor, while `nv-metafun.engine` uses `texexec`—but these defaults can be changed with `% !MPOST tex`.

That said, T<sub>E</sub>X and L<sup>A</sup>T<sub>E</sub>X users will mostly use `nv-metapost.engine`, and ConT<sub>E</sub>Xt users `nv-metafun.engine`, simply because T<sub>E</sub>X and L<sup>A</sup>T<sub>E</sub>X users typically include `mpost` output directly in their documents, while ConT<sub>E</sub>Xt users typically want the features provided by MetaFun and they want the textual labels in their figures to be typeset by ConT<sub>E</sub>Xt (which `nv-metafun.engine` does by default). L<sup>A</sup>T<sub>E</sub>X 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:

```
input 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<sup>7</sup>)

```
mptopdf foo.mp
```

**Q.** Please, I want the *same* behaviour as the original “mpost” option of T<sub>E</sub>XShop!

**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 T<sub>E</sub>XShop!

**A.** Use `nv-metafun.engine`. Really. And if you are a L<sup>A</sup>T<sub>E</sub>X user, read the next FAQ.

**Q.** I’m a L<sup>A</sup>T<sub>E</sub>X user and I want to use `nv-metafun.engine`. But when I try, it fails to compile my L<sup>A</sup>T<sub>E</sub>X labels, even when I use `%&latex!` But `mptopdf` can compile my file! What’s going on?!

**A.** Since `nv-metafun.engine` mainly addresses ConT<sub>E</sub>Xt users’ needs (by trying to be as close as possible to `mptopdf`, but without some of its limitations), it uses `texexec` as the default T<sub>E</sub>X 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?

<sup>7</sup> 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’m a L<sup>A</sup>T<sub>E</sub>X user. Why on earth would I need MetaFun?

A. Well, MetaFun is just a set of METAPOST macros, most of which are ConT<sub>E</sub>Xt-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 L<sup>A</sup>T<sub>E</sub>X 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` L<sup>A</sup>T<sub>E</sub>X package?

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

**Q.** Can I use LuaT<sub>E</sub>X 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 X<sub>Y</sub>T<sub>E</sub>X with METAPOST?

A. Alas, not with my engines. If you want your labels typeset by X<sub>Y</sub>T<sub>E</sub>X, the only way I know is to embed your METAPOST code in a ConT<sub>E</sub>Xt document, and compile it with

```
texexec --xtx
```

**Q.** How about ConT<sub>E</sub>Xt MKIV?

A. As far as I know, the `context` command outputs only PDF, so it is not possible to use it as a T<sub>E</sub>X processor in METAPOST. You may embed your METAPOST code in a ConT<sub>E</sub>Xt document, though.

## 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** (`nv-metapost.engine` only):  
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.
- **\$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

### 1.4.0

- 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 (Opera or Firefox recommended).
- 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 T<sub>E</sub>XShop in MacT<sub>E</sub>X 2009.

The ConT<sub>E</sub>Xt template used for this document is in the public domain, so that you can improve it, share it, and otherwise do what you want with it. Suggestions are welcome. Send them to [sanjoy@mit.edu](mailto:sanjoy@mit.edu) (Sanjoy Mahajan).