The package `epspdfconversion`

Daniel Becker
d.becker@jpberlin.de [*]

June 2, 2010
version 0.61

Contents

1 Purpose of `epspdfconversion`

This package enables the use of the epspdf tools (see http://tex.aanhet.net/epspdf/) from within (pdf)LaTeX "on the fly". It is similar to and based on the epstopdf package that uses the script `epstopdf` for the actual conversion while this packages uses `epspdf` (Note the `epsTOpdf` vs `epspdf`).[1] It is possible to pass several options to the `epspdf` conversion-command.

While this package can be used for the conversion of eps-files to pdf, the `epspdf`-tools itself can do the conversion both ways. Version of the 0.61 adds support for pdf->pdf and ps->pdf conversion, too.

---

[*] Many thanks to Siep Kroonenberg and Heiko Oberdiek for their help.

[1] You might also want to read the documentation of epstopdf. See http://www.ctan.org/tex-archive/macros/latex/contrib/oberdiek/epstopdf.pdf.

I am using this package for the inclusion of eps-figures (or .pdf or .ps) that are produced en-masse by a software packages like Stata, Mathematica or Maple and that are often updated. The package makes sure that I can include the eps-figures easily and the updating of the corresponding pdf's is done "on-the-fly". Using the `epspdf`-tools (and not `epstopdf`) helps a lot to prepare a final pdf that is, for example, accepted by your print shop (really grayscale, prepress-ready, ...).

2 Installation

If you are using a recent version of TeXLive ($\geqslant$ 2008), you can skip this section.

1. Install `epspdf`: Go to http://tex.aanhet.net/epspdf/ and follow the installation instructions there.
2. Check your `epspdf` installation: Make sure that you can use `epspdf` from the command line. I am using Mac OS X. After the installation of epspdf (or with TeXLive / MacTeX $\geqslant$ 2008), the following command is working from the command line (assuming the file `/Users/daniel/Desktop/testimage.eps` exists):

   ```
   macbook-daniel:~ daniel$ which epspdf
   /usr/texbin/epspdf
   macbook-daniel:~ daniel$ epspdf /Users/daniel/Desktop/testimage.eps
   ```

   It results in a file `/Users/daniel/Desktop/testimage.pdf`.
3. The package requires that shell-escape are enabled such that TeX is allowed to execute the command epspdf if needed. You should get a warning in your .log-file if this is not the case. Be aware that allowing that is a security risk.

   Enabling shell-escape means that you have to call pdflatex with additional options. In my case – I use TeXShop on Mac OS X with MacTeX/TeXLive as the TeX-installation in the background – the command specified for pdflatex is `pdflatex --file-line-error --shell-escape --synctex=1`. If you are using MikTeX on Windows as TeX-installation, use `--enable-write18` instead of `--shell-escape`. Look for "shell escape" and "write18" in the Help-Section of your preferred application for Typesetting on how to enable it (TeXShop, WinEdt, ...).
4. *Special Note for Windows-Users:* If you are using Windows and do not know how to install epspdf and use it from the command line: The trick is to create a batch-file `epspdf.bat` and place it somewhere where windows can find it (in your path, similar to pdflatex.exe etc.). This file should contain the line

   ```
   ruby "C:\Programme\epspdf\epspdf\epspdf.rb" %*
   ```

   where `path\_to\_epspdf.rb` needs to replaced the path to epspdf that is valid in your machine, for example by `<C:\Programme\epspdf\epspdf\epspdf.rb>`. If everything went ok, you should be able to execute the command epspdf from the command line in Windows ("Start > Programs > Accessories > Command Prompt".

   Stefan Pofahl (thanks!) suggested the following `epspdf.bat` batch file:

```
@ECHO OFF
REM ---
SET ruby= "C:\Programme\epspdf\rubysub\bin\ruby.exe"
SET rb="C:\Programme\epspdf\epspdf\epspdf.rb"
REM ---
%ruby% %rb% %*
```

Where, again, the path to `ruby.exe` and `\epspdf.rb` need to be adjusted to your local settings. The advantage is that epspdf.rb and ruby.exe not necessarily need to be in your path...

3 Usage

Put in the preamble of your .tex-file the line

```
\usepackage[OPTIONSHERE]{epspdfconversion}
```

where “OPTIONSHERE” can be either empty or be filled with the options described below. `epspdfconversion` loads, among others, the `graphics`-package and also `epstopdf`, so you don’t need to do that. If you prefer `graphicx` over `graphics`, load in before `epspdfconversion`, i.e.

```
\usepackage{graphicx}
\usepackage{epspdfconversion}
```

If you typeset your document, and (pdf)LaTeX detects that you want to use an eps-figure, the `epspdfconversion`-package makes sure that it is converted to a pdf that is then included.

By default, if you include your eps-figure *with* the `.eps` extension, as in

```
\includegraphics[width=0.5\textwidth]{image.eps}
```

, there will be a conversion of your `image.eps`-file to a pdf-file named `image-epspdf-to.pdf` that is then used in your document. The next run will only call a conversion if the original .eps-file is newer (has been updated in the meanwhile). This is to save typesetting time. You can change this behaviour with the option `update=false`, see below.

If you insist on using `\includegraphics` without the extension, as in

```
\includegraphics[width=0.5\textwidth]{image}
```

, the situation is more complicated. If you are using `\includegraphics` without the extension, pdfLaTeX when used with epstopdf or `epspdfconversion` looks for files that can be used in the following order:

```
Package grfext Info: Graphics extension search list:
(grfext)             [.png,.pdf,.jpg,.mps,.jpeg,.jbig2,.jb2,.PNG,.PDF,
.JPG,.JPEG,.JBIG2,.JB2,.eps]
```

If - for whatever reason, a file `image.png` exists, this one will be used, not the .eps or the converted pdf. However, you can use the option prepend the "Graphics extension search list" will look like this:

```
Package grfext Info: Graphics extension search list:
(grfext)             [.eps,.png,.pdf,.jpg,.mps,.jpeg,.jbig2,.jb2,.PNG,.PDF,
.JPG,.JPEG,.JBIG2,.JB2]
```

This implies that image.eps is found first and used if a conversion to pdf is necessary. Complicated? Consider to use `\includegraphics` *with* the extension. This avoids confusion which file actually makes it into your document. See options prepend, prefersuffix, update, suffix below and try to figure out how many different scenarios there are.

4 Options

`epspdfconversion` accepts several options. Table 1 below gives an overview. The explanations are more or less taken from the documentation of epspdf and epstopdf.

Table 1: Options for the package `epspdfconversion`

option	explanation
Options related to epspdf	
`help`	You will be shown the help of the epspdf command in your logfile. This option does not overrule all the others as previously.
`simple`	the epspdf-conversion will be done with no option at all. Don't use it together with any of the options below.
`gray | grey | GRAY | GREY`	gray | grey : convert eps-figures to grayscale (success not guaranteed); GRAY | GREY: Try harder to convert to grayscale (success still not guaranteed)
`nogray | nogrey`	nogray | nogrey : do not try to convert eps-figures to grayscale
`default | printer | prepress | screen | ebook | target=default | target=printer | target=prepess | target=screen | target=ebook`	target use of pdf

... continued on the next page ....

Table 1: Options for the package `epspdfconversion` – continued

option	explanation
`pdfversion=default` | `pdfversion=1.2` | `pdfversion=1.3` | `pdfversion=1.4`	Pdf version to be generated. Setting another version than those on the left will result in an error. 'default' means whatever Ghostscript's default is.
`bbox` | `bbox=true` | `bbox=false`	If true: Compute tight boundingbox
`nopdftops`	Ignore pdftops even if available (default: use if available)
`pdftops`	use pdftops if available
`hires`	compute hires-Boundingbox
`no-hires`	don't compute hires-Boundingbox
`custom={-dPDFX}`	This option allows you to pass a string to the ghostscript-commandline. On the left it would be Here you can set custom options for conversion to pdf, view Use.htm and ps2pdf.htm from the Ghostscript documentation set. The example on the left adds `-dPDFX` to the ghostscript-call by epstopdf
`psoptions={-level2}`	This sets the options for pdftops; the default is -level2, don't include -eps or page number options; these will be generated by epstopdf itself
`pagenumber={1}`	Page (in the source-file) to be converted
Options related to epstopdf.sty (the package)	
	These options are options that are passed over to epstopdf.sty that works in the background. You could also use `\epstopdfsetup{OPTIONSforEPSTOPDF}`, but you can also control the behaviour of epstopdf.sty by means of setting options of `epspdfconversion`. The explanation is borrowed from the documentation of epstopdf.
`prepend` | `prepend=true` | `prepend=false`	Determines whether .eps is appended (if false) or prepended (if true) to the Graphics extension search list. (default: false). (Note that there is no option append. Use `prepend=false` instead.)

... continued on the next page ....

Table 1: Options for the package `epspdfconversion` – continued

option	explanation
`update` | `update=true` | `update=false`	The conversion program is only called, if the target file does not exist or is older than the source image file. If false, the conversion is done with every run of pdflatex. `update=false` makes sense when you are not yet sure which settings for the conversion to pdf you are going to use.
`verbose` | `verbose=true` | `verbose=false`	prints some information about the image in the .log file (default: true).
`suffix={-mysuffix}`	defines a string that is put between the file name base and the extension of the output file. This avoids that existing pdf-files are overwritten. See the documentation of epstopdf for a more detailed explanation. (default: suffix=-epspdf-to)
`prefersuffix` | `prefersuffix=true` | `prefersuffix=false`	If a suffix is set by option suffix, then there can be two image file names that could be taken into account for inclusion: A image file name with the suffix string inside its name and a image file name without; e.g. for `foo.eps` the names could be: `foo-suffix.pdf`, `foo.pdf` If option prefersuffix is turned on, the file `foo-suffix.pdf` and its generation is preferred over using `foo.pdf`. Otherwise `foo.pdf` is included without generating `foo-suffix.pdf`. (default: true)
`outdir=./`	The converted file may put in another output directory. The value of outdir must include the directory separator. Example for the current directory: `\epstopdfsetup{outdir=./}` For other directories ensure that they can be found. See `\graphicspath` or `TEXINPUTS`. You might need to set `suffix=` to use another directory than the current. (default: outdir not set, converted images are saved in the same directory as the source-files.)
`pdftopdf` | `pdftopdf=true` | `pdftopdf=false`	Enable conversion also for .pdf-files. Detects an .pdf-file, converts it to an pdf, applying the other options that are set (grayscaling, ...) and used the converted .pdf-file. (default: `pdftopdf =false`)
`pstopdf` | `pstopdf=true` | `pstopdf=false`	Enable conversion also for .ps-files. Detects an .ps-file, converts it to an pdf and used this. (default: `pstopdf=false`)

When there are several options in the first column, divided by |, this means that you should *choose only one* of them. For example, it does not make sense have this in the preamble:

```
\usepackage[pdfversion=1.3,pdfversion=1.4]{epspdfconversion}
```

`\pdfminorversion`: When you choose the options pdfversion=1.2 or pdfversion=1.3, you need to set `\pdfminorversion` accordingly. The package checks if you have done that properly and shows a warning if not.

Changing the options somewhere in the middle of your .tex document is supported. Writing

```
\epspdfconversionsetup{target=prepress,bbox}
```

changes the options of `epspdfconversion` to `target=prepress,bbox`. Other options than `target=...` remain in effect.

5 `\epspdfconversioncmdline`

Many of the options described above change the command that is used to call epspdf for the conversion from .eps to .pdf.

Typing `\epspdfconversioncmdline` somewhere in your source-.tex file will output the call that you have defined in your preamble. For example, this file has in the preamble

```
\usepackage[pdfversion=1.3,GRAY]{epspdfconversion}
```

and the `\epspdfconversioncmdline` then is: `epspdf --GRAY --version=1.3`.

This means that you can use `\renewcommand` to define you own `\epspdfconversioncmdline`.

For example, to restore the behaviour of the epstopdf-package, you could write

```
\renewcommand{\epspdfconversioncmdline}%
{epstopdf }
```

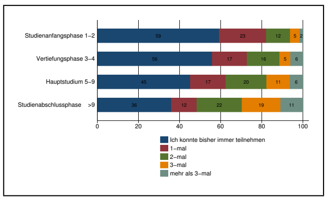
This allows you to use whatever tool you want for your eps->pdf conversion.

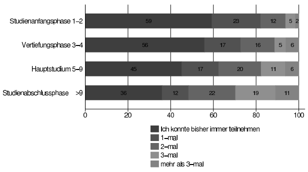
6 Switching options temporarily

It is possible to switch the options only temporarily using curly braces. Consider you have set the options `GRAY` such that all your figures appear in grayscale. Now you want color for a single figure. This can be done like this:

```
{% <= New group started
\epspdfconversionsetup{nogray,bbox=false}
\fbox{\includegraphics[width=4cm]{image2.eps}}
}% <= New group ended; grayscaling,bbox set to previous value.
\includegraphics[width=4cm]{image2.eps}
```

The figure `image2.eps` will be exceptionally in color, other figures in gray, according to the general rule for this document:





7 A note for users of latexmk

(pdf)latexmk is a perl script for running LaTeX, BibTex, Makeindex etc the correct number of times. See `http://www.phys.psu.edu/~collins/software/latexmk/`. It can be configured to run pdflatex if an eps-image has been updated (since version V. 3.21i) – hence updating an eps-file is considered to be an update of the document. In your local configuration file, you should have something like this:

```
# FOR USERS OF epstopdf v1.5 and later only:
add_cus_dep('eps', 'pdf', 0, 'cus_dep_require_primary_run');
```

This makes latexmk to pay attention for .eps-files. If they are updated, pdflatexmk triggers a run of pdflatex (uses the subroutine `cus_dep_require_primary_run`) and `epspdfconversion` can do the necessary conversion of the file.[2]

Note that both epstopdf and latexmk are under active development. If you have difficulties to use latexmk together with `epspdfconversion`, please let me know.

8 Version-history, ToDo's

Possible ToDo's add support for tif and others in pdflatex via convert / add support for pdf-inclusion in latex (not pdf-latex) / add support for more file-types (tif, jpeg,...) in latex (not pdf-latex) / add support for sam2p.

Please report errors, missing features and other suggestions.

v.0.61, 2010-06-1:

- new options pdftopdf and pstopdf. Uses epspdf to do pdf-to-pdf and ps-to-pdf conversions. Allows grayscaling, calculation of bounding boxes etc for pdf's that already exist an for .ps-files. Disabled by default.
- bugfix for the outdir-option (converted files in subdirectories are again saved in those subdirectories) (Thanks to Stefan Pofahl for the feedback.)

---

[2] If you enable the options pdftopdf and pstopdf, you should add the corresponding configuration.

- small improvement of the documentation (on the windows epspdf.bat file, on epstopdf's option 'outdir')
- now uses epstopdf's `\epstopdfDeclareGraphicsRule`

v.0.6, 2010-04-30: small improvements and documentation updates

- pdfversion now uses epspdf's –pdfversion. –version in epspdf is to print the version number of epspdf (currently, epspdf is at 0.5)
- new author email
- documentation updates

v.0.5, 2009-09-02: this update makes use of changes in the epstopdf-package v2.2

- new options update,verbose,prefersuffix,suffix,outdir (they are really epstopdf options, but can be set as options for this package)
- default is that converted files have a suffix
- info in logfile about the setup that is used for epstopdf
- new options hires, no-hires

v.0.4, 2007-11-24: the epstopdf-package is now loaded with options [update,prepend] (works only when epstopdf version 1.5 is used) An update of epstopd.sty (part of the oberdiek-bundle) is recommended. Added options nogrey,nogray

v.0.3, 2007-10-02:

- check whether `\pdfminorversion` has been set in accordance with option pdfversion=...
- Use the kvoptions-package for the implemention of options. It uses key value syntax that can be used both as package options and a separate setup macro.
- Almost all options of epstopdf are now available as an option of this package.
- The command `\epspdfconversionsetup` is new and allows a change of the options for this package anywhere in your document.
- The command `\epspdfconversioncmdline` has been renamed to `\epspdfconversioncmdline`.
- the documentation has been updated

v.0.2, 2007-09-21: the package is now simply based on epstopdf. It essentially defines `\@namedef{Gin@rule@.eps}#1{{pdf}{.pdf}{'\epspdfconversioncmdline #1}}` differently than epstopdf. The code has been cleaned up. Improvements of documentation and additional warning about pdfminorversion....

v.0.1, 2007-09-21: first try