LaTeX 2ε Prepared Documents

Martin Comyn
TRIUMF

Talk Outline

- Background
  - PAC'97 experience
- What goes wrong?
  - Bad fonts
  - Incorrect templates
  - Private class and style files
  - Private macros
  - Devious tricks
  - File corruptions
  - Different LaTeX installations
- Processing LaTeX 2ε files
  - Correct installation
  - Reprocessing using source files
  - Correcting common errors in figures
- Acrobat 3 and 4
- Templates

Background

PAC'97 experience

- 428/1261 published papers were produced using LaTeX
- All had to be reprocessed due to:
  - very few used Type 1 Computer Modern (CM) fonts;
  - many used the wrong template;
  - numerous figure problems.

What goes wrong?

Bad fonts

- Old LaTeX installations only had Type 3 bitmap CM fonts.
- Times fonts were always Type 1.
- Many installations did not have the *times* package.
- If the wrong template was used (PAC'95, IEEE, APS, etc.) then the entire document was produced in Type 3 CM fonts.
- Type 3 CM bitmap fonts only have 300 dpi resolution and the resulting *.pdf* files are almost illegible when viewed on a screen. The resulting print quality is also very poor.

How to tell whether Type 1 or 3 fonts have been used:

Look in the PostScript file to see whether:

there is a TeXDict dictionary with a long block of hex code (Type 3 CM fonts):

```
TeXDict begin 40258431 52099146 1000 300 300 @start /Fa 1 121
df<1CF023182310060006000600 4610CE2073C00D097F8810>120 D E /Fb
⋮
```

or a series of font dictionaries (Type 1 CM fonts):

```
%%BeginFont: CMR10
%!PS-AdobeFont-1.1: CMR10 1.00B
%%CreationDate: 1992 Feb 19 19:54:52

% Copyright (C) 1997 American Mathematical Society. All Rights Reserved.

11 dict begin
⋮
```

What goes wrong?

Incorrect templates

- Many papers submitted using the incorrect template and class files.
- PAC'97 had many examples of EPAC'92, EPAC'96, PAC'95, IEEE, APS, etc.

Solution:

Cut and paste the *.tex* file into the correct template and reprocess.

What goes wrong?

Private class and style files

- Beware of disk and ftp submissions which include private class or style files, or (difficult to spot) an edited conference class file retaining the same name.
- Almost certainly the author has been fighting to save space and the paper may not conform to the mandated margins or style.

Solution:

Reprocess using the correct class file.

What goes wrong?

Private macros

- May be implemented via an additional style file, or included in the preamble of the *.tex* file.
- Many are okay — just shorthand for frequently used strings.
- Others are more devious — introduced to reduce whitespace around headings or change font sizes.

Solution:

Look here first if the paper does not conform to the margins.

What goes wrong?

Devious tricks

- The *.ps* file was produced using an edited class file, but only the *.ps* and *.tex* files were submitted. It is impossible to reproduce the submitted layout if the paper has to be reprocessed and the page limit is often exceeded.
- The preamble contains new or redefined commands in order to override the mandated style of the document. Often used to reduce whitespace around section headings.
- Extensive use of `\vspace*{-xmm}` or `\hspace*{-xmm}` to gain extra space for the text and figures.

What goes wrong?

File corruptions

- Cross platform problems introduced when sending source files via binary rather than ASCII.
- PC (CRLF) / Mac (CR) / UNIX (LF) linebreak convention problems.
  - Use a Perl script to fix the problem.
  - The most severe problems arise when processing a *.tex* file on a UNIX machine which was produced on (or submitted via) a PC.

    Every other line is blank, causing each line of text to be typeset as a separate paragraph.

    Always use a Perl script to fix the problem as it is very easy to make mistakes if deleting blank lines by hand using an editor. Genuine paragraph breaks are sometimes missed.

What goes wrong?

Different LaTeX installations

- UNIX (various platforms), PC, Mac, VAX/VMS, Open VMS, etc.
- Some installations cannot run the latest version of LaTeX 2_ε.
- The latest TeTeX releases solve this problem by having common class, style and font libraries. Only the binaries for TeX, LaTeX 2_ε, *xdvi* and *dvips* differ.
- Few problems were experienced during reprocessing if the original document worked correctly on the original platform — so long as a complete set of source files was provided. Some page layout changes may occur.

Processing LaTeX 2ε files

Correct installation

- Ensure that the latest version of LaTeX 2ε is installed by checking the *.log* file produced when running LaTeX 2ε.

Old LaTeX2.09 or LaTeX 2ε:

```
This is TeX, Version 3.141 [PD VMS 3.4/CERN 1.0]
(preloaded format=latex 94.9.26 ) DATE TIME
⋮
LaTeX2e <1994/06/01> patch level 4a undistributed
```

LaTeX 2ε used for PAC'97:

```
This is TeX, Version 3.14159 (C version 6.1) (format=latex 97.3.1)
DATE TIME
⋮
LaTeX2e <1996/12/01> patch level 1
```

Latest LaTeX 2ε:

```
This is TeX, Version 3.14159 (Web2C 7.3.1) (format=latex 1999.5.27)
DATE TIME
⋮
LaTeX2e <1998/12/01> patch level 1
```

Processing LaTeX 2ε files

Reprocessing using source files

- Use correct template and class file.
- Switch on margins overlay to confirm that the document conforms to the correct margins.
- Check effect of any user defined macros.
- Check for acceptable behaviour of any `\vspace*` or `\hspace*` tricks to reduce whitespace.
- Check for LaTeX 2ε warnings and errors.
- Solve any LaTeX 2ε errors.
- View using *Ghostview* or *Ghostscript*.
- Any "virtual video" effects in figures?
- Does the *.ps* file print okay, but not view okay?
- Any viewing problems must be fixed or the resulting *.pdf* file will be defective with blank figures or pages.
- Okay?
  - YES:
    - Turn off margins overlay.
    - Re-run LaTeX 2ε (more than once to resolve citations and references).
    - Run *dvips* to get *.ps* file.
    - Produce *.pdf* file.
  - NO:
    - See correcting common errors in figures section.

Processing LaTeX 2ε files

Correcting common errors in figures

- General comments
- EPS file structure
- Bounding box — *ps2epsi*
- Cannot view/print — head & tail
- CTRL D or Z — EOF
- Printer setup
- Whiteout of rectfill areas
- White fill of clip areas
- Unwanted frame box
- Unwanted headings
- Incorrect caption text or positioning
- Incorrect linewidths
- Incorrect colours
- Cropping scanned images
- Not EPS — LaTeX & *dvips*
- Large files take too long to display
- Scan

Processing LaTeX 2ε files

Correcting common errors in figures

General comments

- 99% of problems are caused by bad PostScript code at the head or tail of the file, or in the %%Begin(End)Setup and %%Begin(End)PageSetup blocks.
- Many figure files are not true EPS files. They were created by saving to file with parameters set for a specific PostScript printer. This often results in viewing and printing problems due to explicit printer setup commands being embedded in the PostScript code.

  A generic PostScript printer driver is required, or correct EPS creation software.

  Bounding box values are also often incorrect.
- Some software packages have standard %%BoundingBox: values, irrespective of the image size or position. Incorrect scaling results. Use *ps2epsi* to determine the correct values.
- Commented out code above %!PS-Adobe... prevents *Ghostview* or *Ghostscript* from displaying images with the correct window (bounding box) size. However the figures scale properly in the LaTeX 2ε document.

  Delete the code above %!PS-Adobe... or move it to the tail of the file.
- Any viewing problems seen when using *Ghostview* or *Ghostscript* always result in bad *.pdf* files.

See:

`http://www.triumf.ca/annrep/ar99www/ar98_figures.html`

for full details.

Processing LaTeX 2ε files

Correcting common errors in figures

EPS file structure

Required comments:

```
%!PS-Adobe-x.0 EPSF-y.z
```

(x=1,2,3 ; y=1,2,3 ; z=0,1,2,...)

```
%%BoundingBox: llx lly urx ury
```

Recommended comments:

```
%%Title:
%%Creator:
%%CreationDate:
%%EndComments
```

Conditionally required comments:

```
%%Begin(End)Preview:
%%Extensions:
%%LanguageLevel:
```

Allowed comments:

```
%%DocumentNeededFonts
%%DocumentNeededResources
```

Illegal and restricted operators:

```
statusdict
userdict
clear
cleardictstack
initmatrix
setpagedevice
...
```

... PostScript code for image ...

```
showpage
%%Trailer
%%EOF
```

Processing LaTeX 2ε files

Correcting common errors in figures

Bounding box — ps2epsi

Problem:

The bounding box values specified for the figure are wrong and the image does not scale correctly. Often too much whitespace around figure.

Solution:

Run the *ps2epsi* package to determine the correct bounding box values from the resulting *.epsi* file which is created (see the `%%BoundingBox:` line at the head of the file). Comment out the incorrect `%%BoundingBox:` line in the *.eps* file and insert the correct one (edit the *.eps* file rather than using the *.epsi* file because the *.epsi* file is larger due to the fact that it also includes a preview of the image).

Sometimes the *ps2epsi* package runs but produces an error message; the resulting *.epsi* file is not viewable. However, in many cases the bounding box values have been computed correctly and can be copied from the head of the file.

If *ps2epsi* fails completely, the bounding box must be determined via trial and error.

ps2epsi also fails to calculate the urx coordinate correctly if an image created in landscape orientation was saved in portrait orientation. In this case urx=612 (8.5" in points) for US letter paper mode or urx=595 (210 mm in points) for A4 paper mode. Use trial and error to find the correct coordinate value.

Warning:

After using *ps2epsi* and editing the *.eps* file, re-run LaTeX 2ε and *dvips* and then use *Ghostview* or *Ghostscript* to check that the image has scaled properly. In addition, ensure that neighbouring text or images have not been clipped by an extended whitespace around the image. See the instructions for how to suppress whiteout.

Processing LaTeX 2ε files

Correcting common errors in figures

Bounding box — ps2epsi — trial and error

Problem:

ps2epsi or other solutions fail to determine the correct bounding box values.

Solution:

A good estimate can be gained by printing the *.eps* file, drawing a box around the image which just touches the top, bottom, left and right-most pixels of the image, and then computing the coordinates of the lower left and upper righthand corners of the box to determine the llx, lly, urx, ury values in points (72 pt = 1 inch). The origin of the coordinate system is the lower lefthand corner of the paper.

Edit the *.eps* file by inserting the new values in the `%%BoundingBox:` line, then view the resulting file using *Ghostview* or *Ghostscript*. If the bounding box is correct, the *Ghostview* or *Ghostscript* window will scale to exactly enclose the image. Edit the estimated values to correct for any whitespace or clipping.

Processing LaTeX 2ε files

Correcting common errors in figures

Cannot view/print — head & tail

Problem:

The figure cannot be viewed using *Ghostview* or *Ghostscript* and cannot be printed.

Inability to view or print is caused by the existence of unwanted lines at the head of the file, or unwanted characters at the beginning of the first line, and unwanted lines appended at the end of the file.

Often the lines at the head and tail of the file correspond to escape sequences which are sent to the printer to explicitly force it into and out of PostScript printing mode.

Solution:

The first line of the file must begin with

`%!PS-Adobe-...`

Strip off any lines above this, or any characters before it on the first line.

The file should end with

`%%EOF` (or `%%Trailer` with perhaps valid lines in the trailer block).

Strip off any lines after this.

CAUTION: many of these lines are extremely long binary strings (may be 1 MB long) and editors tend to freeze if an attempt is made to go to the end of the file and start deleting backwards. Find the EOF— line, move to the next line and then start deleting by word or line. Normally there are only about 5–50 lines of data to delete.

Processing LaTeX 2ε files

Correcting common errors in figures

CTRL D or Z — EOF

Problem:

`%%EOF` line followed by `CTRL D` or `CTRL Z` character.

Causes viewing and printing problems.

Caused by the software package used to create the EPS file or using the `-F1` option in *dvips*.

Solution:

Delete the `CTRL D` or `CTRL Z` character at the end of the file.

Processing LaTeX 2ε files

Correcting common errors in figures

Printer setup

Problem:

The file is not a true EPS file, but a PostScript file which contains explicit printer setup commands.

Explicit printer setup commands are contained in the

`%%Begin(End)Setup` or `%%Begin(End)PageSetup`

sections of the PostScript file which cause viewing and printing problems.

Solution:

Find the

`%%BeginSetup ... %%EndSetup`
and
`%%BeginPageSetup ... %%EndPageSetup`

sections of the file.

Look for sets of lines like:

```
%%BeginFeature: *OPTION
... (SETUP)
%%EndFeature
```

where `*OPTION` can be:

```
*PageSize US Letter
*Resolution 300dpi
*InputSlot AutoSelect Tray
*ManualFeed
*Duplex
```
etc.

Either delete or comment out each line corresponding to `... (SETUP)`

(Comment out a line by placing a single % at the start of the line.)

Blank lines may be left as no printer setup is specified.

Processing LaTeX 2ε files

Correcting common errors in figures

Whiteout of rectfill areas

Problem:

The software package used did not create a true EPS file, but a PostScript file where the image was placed on a full size page which was explicitly initialized with a white colour fill.

Almost all problems are due to the use of the *Microsoft Windows PostScript Driver* PSCRIPT.DRV.

The PostScript macro `rf` (often multiply defined in the various PostScript dictionaries in the file) is defined to execute a `rectfill` (rectangular fill) of the defined area (full US Letter or A4 page) in the defined colour (white).

Even if the bounding box values for the image are correct, when the scaled image is laid down on the page it still retains a white border corresponding to a whiteout area of a full US Letter or A4 page scaled down by the figure scaling factor. This whiteout area obliterates any underlying text or images which have already been defined in the PostScript file (text or images defined later in the file can overprint this whiteout area).

Solution:

Find the
`%%BeginPageSetup ... %%EndPageSetup`
section of the file and then the line

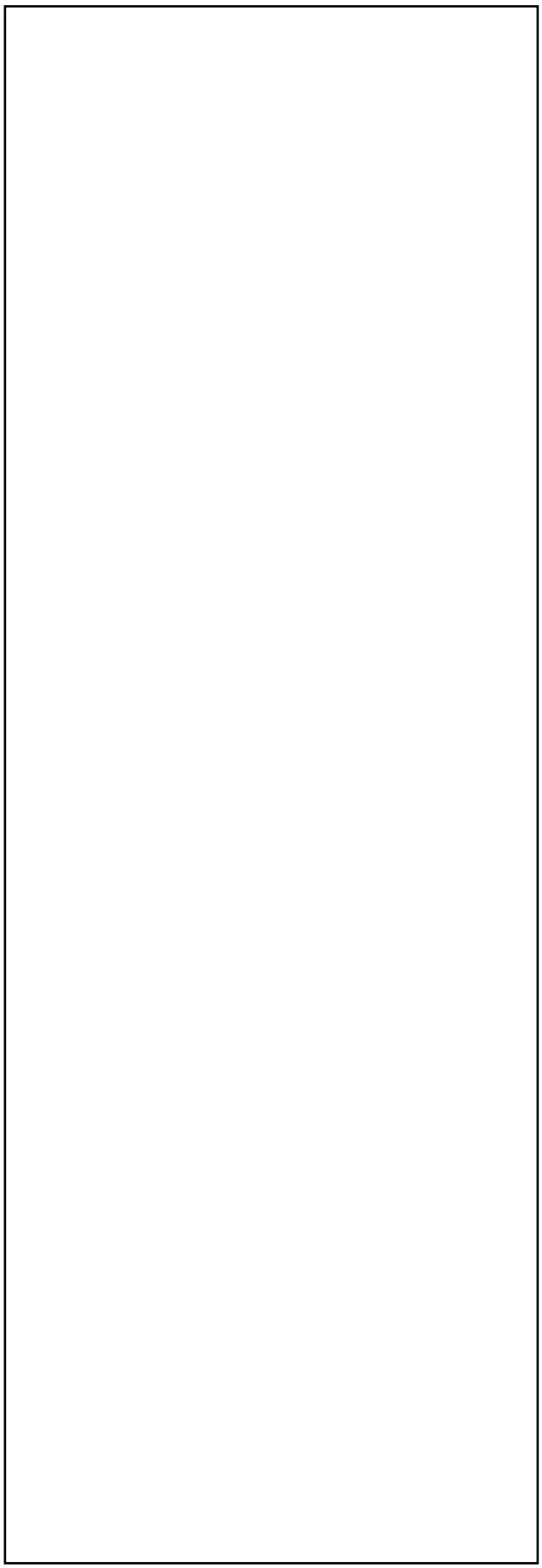
```
userdict begin /pagesave save def end mysetup concat colspRefresh
: 1.000 1.000 1.000 sco 0 0 XXXX YYYY rf ;
```

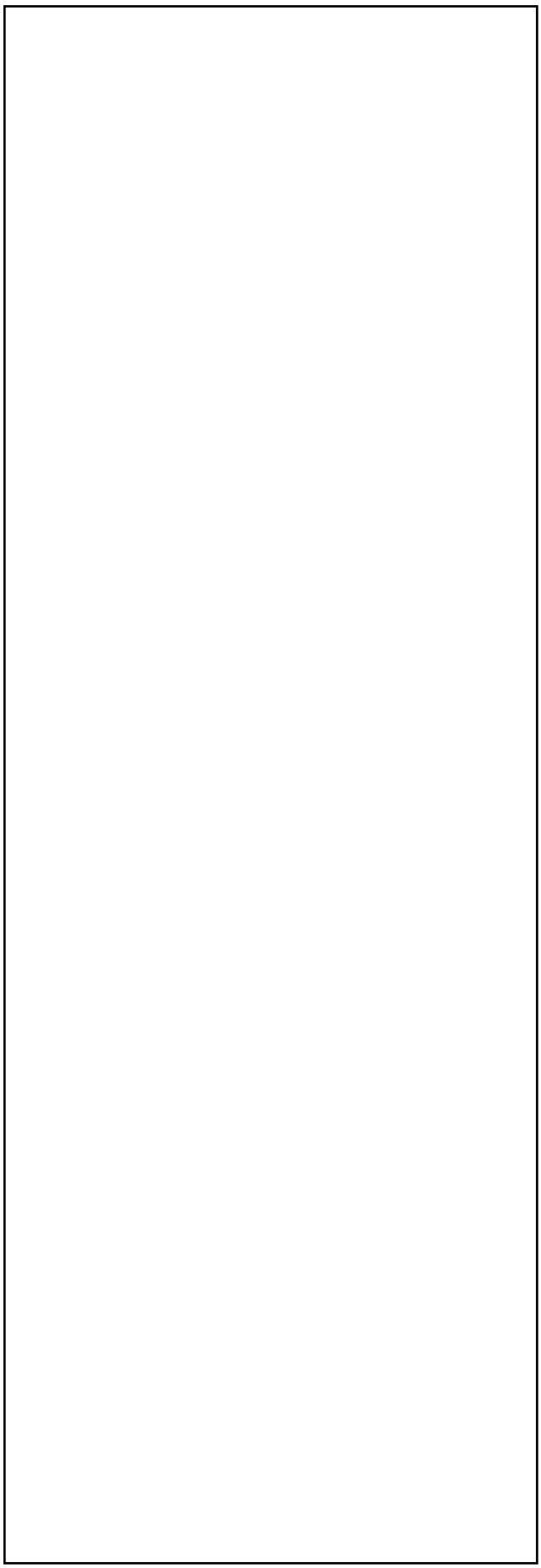
where `XXXX YYYY` =

2550 3300 (300 dpi, US Letter, 8.5" x 11", portrait)
3300 2550 (300 dpi, US Letter, 8.5" x 11", landscape)
5100 6600 (600 dpi, US Letter, 8.5" x 11", portrait)
6600 5100 (600 dpi, US Letter, 8.5" x 11", landscape)
2480 3508 (300 dpi, A4, 210 mm x 297 mm, portrait)
etc.

Edit the line by replacing both `XXXX` and `YYYY` with `0`.

Now the `rf` rectangular fill command is filling a zero area rectangle with white.





Processing LaTeX 2ε files

Correcting common errors in figures

White fill of clip areas

Problem:

Clip area defined and filled with white before laying down the image, obliterating any underlying text or images which have already been defined in the PostScript file.

Solution:

Normally the offending code will be found at the head of the PostScript code which draws the image.

Look for code such as:

```
xxxx yyyy N M XXXX YYYY rr : 1.000 1.000 1.000 sco 0 ;
```

which starts a newpath and moves to `xxxx yyyy` before defining a relative clip area of dimensions `XXXX YYYY` and performing a white fill. Edit the line by changing both `XXXX` and `YYYY` to `0`.

Or look for code such as:

```
: 0 0 XXXX YYYY rc
```

which defines a rectangular relative clip area of dimensions `XXXX YYYY` and performs a white fill. Edit the line by changing both `XXXX` and `YYYY` to `0`.

Or look for code which defines a newpath, moves to a point, defines a box by a series of four lines, and then performs a white fill using the `F` macro. In this case, comment out or delete these particular lines of code.

In each case, `XXXX` and `YYYY` are large numbers corresponding to the overall size of the image. There may be many instances of such strings of code in the file because many software packages clip and white fill areas before laying down each element. However, the offending set of code is normally at the head of the PostScript code defining the image.

Processing LaTeX 2ε files

Correcting common errors in figures

Unwanted frame box

Problem:

The image has been produced with an unwanted frame box drawn around it.

Solution:

In the case of a scanned image file, the hex code of each affected scan line has to be edited.

For a normal EPS file, the PostScript code defining the frame box lines has to be found and deleted, commented out, or modified. The code is normally near the head or tail of the PostScript code which defines the image.

Occasionally the frame box is drawn twice, so two sets of code have to be modified.

Processing LaTeX 2ε files

Correcting common errors in figures

Unwanted headings

Problem:

The figure includes an unwanted heading which is duplicated in the figure caption, or an incorrect heading, or other unwanted or incorrect text.

Solution:

Find the relevant PostScript code (normally easy to find by searching for the actual words or fragments of the words) and either comment out (sometimes fails) or replace by null fields.

If the text needs changing, modify the relevant code. However, positioning problems may result if major modifications are made.

Some software packages do not use PostScript fonts for text, but draw a series of lines instead. These are far more difficult to edit, but sometimes the correct sections of code can be found and removed.

Processing LaTeX 2ε files

Correcting common errors in figures

Incorrect caption text or positioning

Problem:

The captions on graphs are incorrectly labelled or are in the wrong position.

Solution:

Find the relevant code and alter the wording or positioning.

To alter the position of a part of the image, use the `x y translate` command before the relevant section of code, followed by a compensating translation in the opposite sense before the remainder of the code in the figure.

Processing LaTeX 2ε files

Correcting common errors in figures

Incorrect linewidths

Problem:

Some or all linewidths are too thick or too thin in the figure.

Solution:

Find the `n setlinewidth` command (or the corresponding macro command) for the relevant part of the image and alter the value of `n`. Some software packages assign global linewidths, others explicitly assign values for each set of line elements.

Processing LaTeX 2ε files

Correcting common errors in figures

Incorrect colours

Problem:

Colours need to be changed in the figure for some reason. In particular, pale colours such as yellow or cyan reproduce poorly when printed on monochrome laser printers. Lines, arrows and labelling should not use pale colours.

Solution:

Find and edit the `r g b setrgbcolor` values for the relevant parts of the image. Some software packages define global macros for a palette of colours, in which case only one set of rgb values has to be changed to alter all such instances of the colour in the image. Other software packages explicitly assign values for each set of line elements.

Processing LaTeX 2ε files

Correcting common errors in figures

Cropping scanned images

Problem:

The file has been created by scanning an image or by conversion from another file format using a package such as *XV*. The image has unwanted features such as a frame, imperfections, or whitespace.

Solution:

If removing whitespace, edit the preamble defining the number of pixels in the horizontal and vertical directions, and then either remove the unwanted lines of hex code at the top and/or bottom of the image, or edit every line of hex code to remove pixels on the left or right.

If removing unwanted lines or imperfections, edit the corresponding hex code for each scan line.

(NOTE: This procedure requires a detailed knowledge of the hex code syntax, which differs depending upon which software package was used to produce the image.)

Warning:

If the scanned image has unwanted whitespace around it, beware of merely using *ps2epsi* to find the correct bounding box values, editing the *.eps* file to insert the correct bounding box values, and then rescaling the image in the document. The white pixels forming the unwanted whitespace border will also be scaled and may overwrite any neighbouring text or images that have already been defined in the PostScript file. Therefore the steps outlined above should be followed in order to remove any unwanted whitespace from the image.

Processing LaTeX 2ε files

Correcting common errors in figures

Not EPS — LaTeX & *dvips*

Problem:

Not a true EPS file but a PostScript file defining an image placed somewhere on a one page document.

In many instances, PostScript files masquerading as EPS files can be used successfully in LaTeX 2ε documents, either unaltered or by using one of the fixes described in this table.

However, PostScript files that have been created by writing LaTeX code to place an image on a page, then running LaTeX 2ε and *dvips* to create a PostScript file, have the problem that they have incorrect bounding box values in the heading and embedded code specifying a full US Letter or A4 page size. Therefore the images will not scale properly. The irony is that a (perhaps) perfectly good EPS file had to be used to create an unusable PostScript file.

Solution:

If the original EPS figure file cannot be obtained, edit the PostScript file by stripping out all of the code after the `%%BoundingBox:` line in the heading until `%%Page: 1 1` is reached.

Use the `llx lly urx ury` values in the `@beginspecial` command as the possibly correct bounding box values and edit the `%%BoundingBox:` line in the heading.

Then delete the code starting at `%%Page: 1 1` until after the `%%BeginDocument:` line.

The PostScript code defining the image follows.

At the tail of the file, comment out the lines created by LaTeX 2ε and *dvips* and leave the `%%Trailer` and `%%EOF` lines and relevant code in the Trailer block.

View the file using *Ghostview* or *Ghostscript* to see if it works, and run *ps2epsi* to confirm the bounding box values.

Processing LaTeX 2ε files

Correcting common errors in figures

Large files take too long to display

Problem:

Figures contain too much data, often due to superposition of colour fills or plotting of an extremely large number of data points. The resulting *.pdf* files take too long to display.

Solution:

Import into *Adobe Illustrator*, save as a bitmap, and then in EPS format.

Processing LaTeX 2ε files

Correcting common errors in figures

Scan

Problem:

All attempts to edit submitted EPS file fail, or no electronic version of the figure exists.

Solution:

Scan, using *Adobe Photoshop* or similar package, and save as an EPS file.

Acrobat 3 and 4

Problem:

Type 3 CM bitmap fonts only have 300 dpi resolution and the resulting *.pdf* files are almost illegible when viewed on a screen. The resulting print quality is also very poor.

Acrobat Distiller 3.02 claimed to have solved the Type 3 font problem for screen rendering. However tests with PostScript files produced by old LaTeX installations contradict this claim.

Likewise, Acrobat Distiller 4.0 does not solve the problem for these files.

Solution:

Reprocess the files using LaTeX 2_ε and Type 1 fonts.

Templates

- There is a convergence between the EPAC and PAC class files, but several features added during the PAC'97 processing phase were not implemented for EPAC'98 or PAC'99.
- Formulation and adoption of a single set of templates is vital. The more familiar authors become with the format, the less errors there will be.
- Useful features embedded in the class file are not advertised well enough.
  - Itemize, Enumerate, Description dense list environments.
  - Macros for correct formatting of commonly used units.

Templates

Features added for PAC'97

- Remove whitespace above title.
- Remove whitespace at lefthand and righthand ends of title so that the title can span the full page width.
- Allow option of automatic uppercase for the title via the template (could not be implemented in the class file).
- Force boldmath in the title and section headings. Otherwise normal font Greek characters, superscripts and subscripts look terrible.
- Fonts sizes reduced for most sectioning levels and shapes changed. This gained several lines of text in a typical three page paper.