Using graphics in scientific documents. (Notes from 2011)

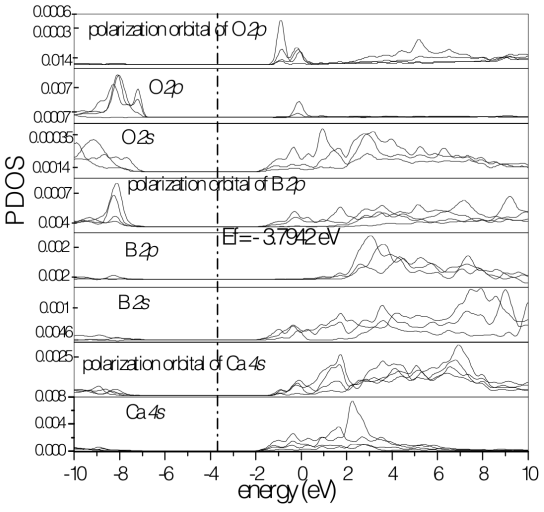
Andrei Postnikov
Université de Lorraine – Metz
andrei.postnikov@univ-lorraine.fr

This document does not address any particular tools to create scientific graphics, but summarizes my long-term observation of faults which repeatedly come about in manuscripts, publications and in scientific presentations, and which could have been so easily avoided. Even if they typically concern purely exhibitionistic side and not the contents of work, the damage they do to the work as whole is often quite pitiful. As the scientific activity demands certain degree of concentration and discipline, the clearness of the contents more often than otherwise imposes on the authors a desire, and an ability, to clearly express themselves by graphical means as well. No special talent nor a special eye is needed for this: a big number of great works appears in permanence, accompanied by clear and to-the-point graphics. Inversely, a "who cares" style of figures suggests sometimes – admittedly not always – a similar attitude for the scientific contents.

The aim of this document is to summarize some rather evident hints for those who care, and would like to make their graphics better. The text is illustrated by "negative" examples taken from genuine manuscripts. I did not ask for permission to reproduce these figures from the authors whose identity I do not disclose, and I offer my excuses to those of them who may feel offended.

First make a figure ready, then insert it into the document

It seems so easy to modify a figure, to assemble a figure from several parts, add inscripts to it, or hide unwanted parts of the figure, directly in your document, especially if you work in MS Word© or PowerPoint©. This practice haves big imminent risks, much more severe than you might at first think of. The document so composed, when sent elsewhere, might not necessarily be formatted the same way as you see it on your screen: some special symbols happen to be replaced by junk, arrows displaced to where they don't belong to, regular patched text not necessarily in the size or font you intended it to be.

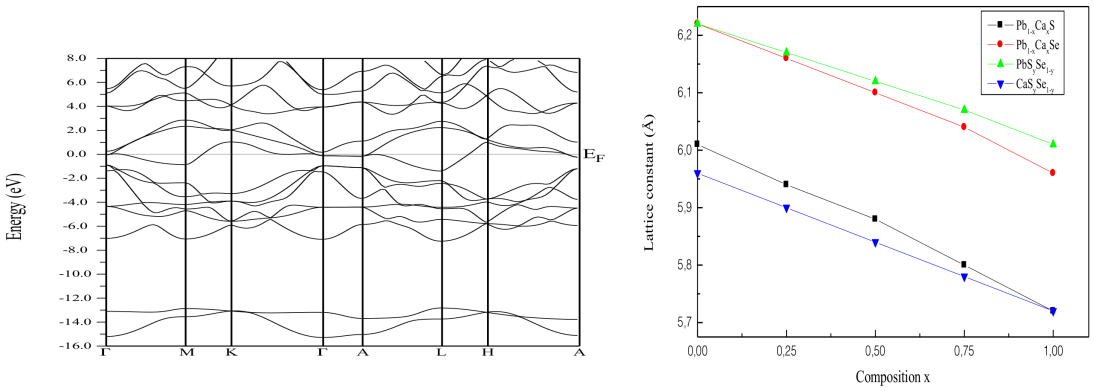


Probably you've seen such disaster, accompanied by big surprise by authors, in some PowerPoint© presentations at conferences. Similar happens with manuscripts sent to journals. Therefore, **rule number one:** assemble your figure ready with whatever graphic program (it may even be your favourite MS Word©, to this end), *then* export it as a whole in one of graphic

formats, and *then*, when nothing more would go astray in the assembled figure, insert it in your document. This means: even if you need just a small change in already existing figure, take your time to go one step back, redo the figure, save it in the graphic format, insert into the document. Never never apply patches to figures in the body of document.

Do not distort figures, preserve their proportions

It seems so easy to change the size of figure you insert in the document, just by clicking at its angle and tearing to the side. So you can fill the full width of page, or, if your figure doesn't fit into the rest of page you want it to reside, you shrink it downwards to the size that it finally fits. If you teared along the diagonal, so that the Y:X proportions of the figure were preserved, that's fine. However, if you teared along Y or X only, or made a sequence of such X/Y steps, the chances are that the proportions of your text, circles/squares of your data points, and even the widths of curves are so distorted that now look terrible. You might not really care; but please consider that people with more sensitive organization may read your document, and get a very bad impression. The result is too obviously of a non-professional quality. The typographic fonts are for good reason generated in different sizes; just tearing the letters upwards or sidewards does not do a good job. What to do, then? – If the original figure is out of your control (a copy from another publication, a photo, a logo) – leave its proportions as they are, be satisfied with the option of just changing the size – or, redraw its contents within the proportions of your preference. If the figure was your own, think over a possibility to generate it again, setting the proportions of the plot to those you want to ultimately have.



The left figure was considered by its author too narrow and hence was *a posteriori* expanded, whereas the right one was believed too broad, and underwent a slight compression. Look at the effect this had on the X/Y axes labeling – within the same figure, they are not anymore the same font! – and on data marks in the right plot – they are not anymore circles/squares... Whereas it would have been so easy to modify the figure sizes/proportions within the plotting routine!

Know the difference between vector and bitmap graphics, make a qualified choice between them when possible

There are two basically different ways to store the graphic information. Vector graphics store the information on graphic *objects*: a circle of such size, centered at such and such point, of specified colour, connected by line of such and such width to a square elsewhere, etc. Bitmap graphics describes a $(N \times M)$ raster, or matrix, of squares whose colours, or other properties, are specified in more or less compact way. When it comes to printing a figure of paper, or showing it on screen, it always comes down to putting the individual dots where they belong, i.e., the end device converts any graphic format into bitmap of some kind.[1] However, the intermediate format in which you can store the graphic information, send it to editorial office for publication etc., may differ. To be specific, graphic formats like `.png`, `.jpg`, `.tiff` are bitmap, whereas postscript format `.ps`, `.eps` are, *in principle*, vector ones. "In principle" – because a bitmap format can be converted into a postscript, which then stores the information about its bits as vector objects, i.e., it becomes as good or as bad as any other bitmap graphic format. Here comes the main difference between two ways of representing graphics. Vector graphics is perfectly scalable – a circle remains a circle, a line remains a line, – whereas the bitmap, when ultimately zoomed in or blown up, contains not more than huge colour squares. This becomes especially annoying if a small figure has to be considerably enlarged, e.g., for a poster. Another difference is the size of graphics file. As with photos, the higher the resolution (dots per inch) and hence the number of color bits to save, the larger the size of graphic bitmap file (with some variations between different formats). Whereas the size of the vector graphic file does not depend on the actual size of figure, which is simply influenced by the global scaling factor, but only on the number and complexity of vector objects described by it. Usually, postscript files (and hence vector graphics) are much more compact than bitmaps, but there are exceptions. For instance, a postscript figure might contain a description of very many graphical objects (e.g. from mathematical drawing of fancy surfaces, or "orbital-caracter band structure plots" over very many $E(\mathbf{k})$ values in electronic structure calculations), which won't be resolved by eye anyway, for any practical purpose.

In order to minimize the damage, and to get most from the graphic tools we have in our possession, I suggest the following system of reasoning.

1. Does your graphic program allow to generate the resulting figure as (vector) postscript? If it does (as **Grace**, **Origin**©, **Gnuplot**, **Xfig** do), we save it in this form, generating the bitmap of appropriate size as the next step.

2. What is the appropriate bitmap size? It depends on the physical size of the figure you'd like to ultimately have (in a journal or on a poster – it's not the same!) The spatial resolution 300 dpi (dots per inch, 1 inch $\approx$ 25.4 mm) is that of a very good

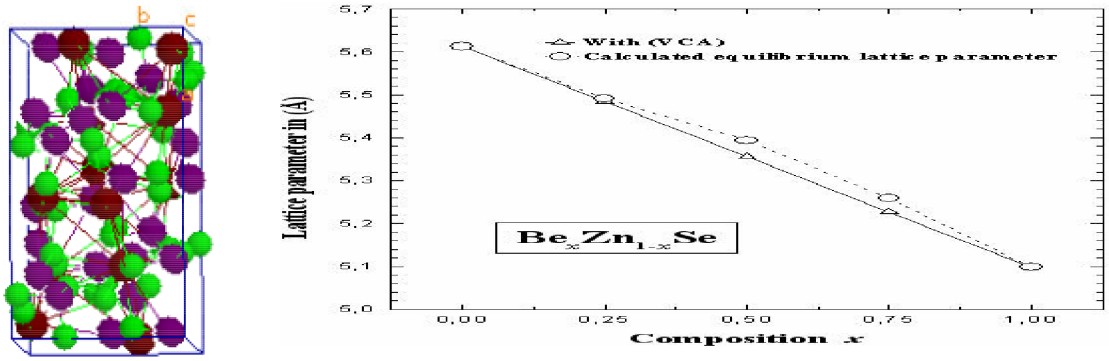
[1] With the possible exception, or at least peculiar standing, of plotting devices, probably not much common nowadays, in which the ultimate operation decomposes into individual events of moving, lifting, and putting down again the pen.

printer, in which the dot raster is not distinguishable for eye. If the figure is not very rich in fine details, the 200 dpi resolution could be almost equally good.

3. If the original figure is a bitmap of superior resolution, you keep it as such in your files, but for a publication you can safely reduce the resolution to the above numbers. For any kind of manipulating bitmap images (changing the resolution, size, converting from one format to another) I find the **Gimp** program (which is free and exists for MS Windows© and Linux) extremely useful.

Beware of bad resolution; if in doubt draw anew

Disappointingly often one sees, in manuscripts and in presentations, math plots done in such bad resolution, that hardly anything reasonable can be read from them. Sometimes this happens because a screenshot was downloaded instead of better-quality graphic file:



(In the second of these examples, a screenshot resolution was combined with ruthless Y/X distortion, the issue already addressed above). If no good resolution is available, but you want to demonstrate the essence of the figure, you might consider making the plot anew. Even if this is somebody else's figure which you borrow for presentation, there is nothing wrong in re-drawing it, as long as you indicate the source: Landolt-Brnstein does it all the time, and some figures in these fundamental reference volumes are of much better quality than the originals. Re-drawing might imply, among other options, re-drawing by hand, pen on paper as in good old times, with subsequent scanning the picture. This creates a fresh impression and sometimes helps to awake the audience.

Remove excessive white margins around figures

Figures which come "as they are" from your graphic program or from elsewhere do often include large non-transparent margins. This effectively shrinks the "useful" size of a figure in the document, and hinders placing two figures side by side. In presentations, one often sees the white margins of a figure cover and mask a part of useful text. Generally, a figure should always be stored/used "as large as possible" (i.e., with parasite white space removed); the extra space, if necessary (before the caption, or between two adjacent figures), is better be

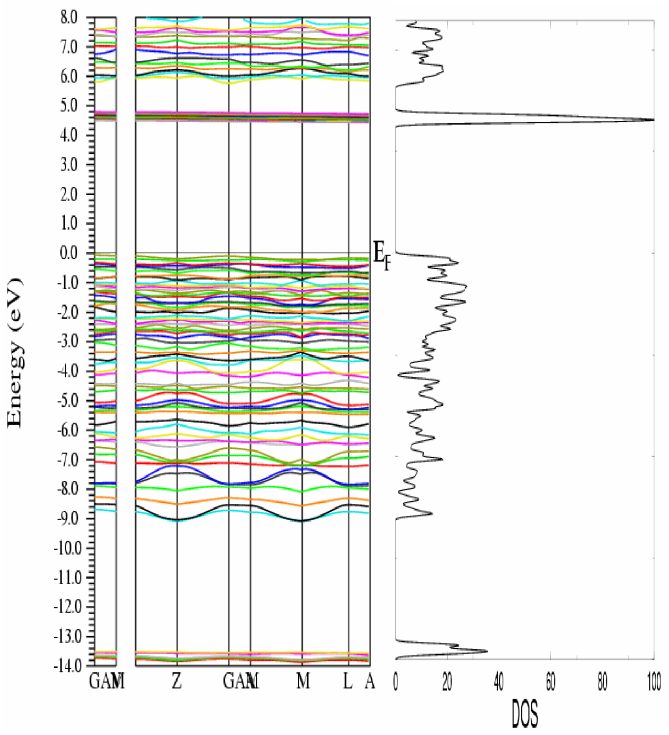
added and tuned in the final document by word processor means. If your figure is in postscript format, converting to eps (Encapsulated PostScript) automatically builds a Bounding Box (see below) around the "non-trivial" contents of figure, neglecting the rest. In Linux, this can be done on the command line as:

```
ps2eps -l MyFigure.ps
```

that creates the file `MyFigure.eps` with reduced BoundingBox; the optional flag `-l` (for "loose") leaves a "minimal" tiny white border around the figure. Bitmap graphics can be very easily "cut" and "pasted as a new figure" within **Gimp**; in this case the cutting of the figure size is done by hand.

Be prudent with colors

Scientific journals insist, and for a good reason, that submitted graphics should make sense also in black/white print. A tiny minority still reads, and stores, publications on paper, and may not always use color print, so this requirement is to protect their rights. Nobody expects a fancy spin density cloud painted with a color code for electron localization function to look as glorious in monochrome, but it should look decent. More common, if you want green and pink curves in your plot, please, please, make them also different in thickness, or in dash length. And, different atoms in your crystal structure may also differ a bit in their size, isn't it? Do not use colors excessively when they don't provide any useful information at all. In the band dispersion plot, every band does not have to be of different color (unless you have a special intention to emphasize some), even if **Grace** assigns it this way on reading.

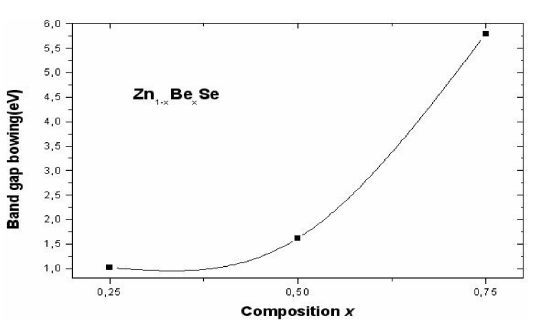


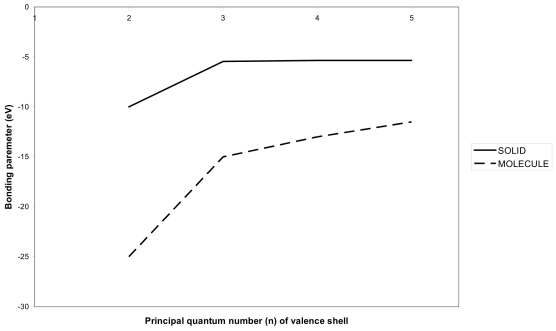
The figure on the left combines an abusive use of colors with excessive X-compression, to the point of illegibility of labels on the X axis.

And, last but not least: a really good contribution to the environment protection and to colleagues' budget is to avoid generating figures with a thick black background. Even if your favourite plotting program or crystal structure program likes it that way by default, there are certainly ways to overcome such setting for the final "to be published" version.

Avoid a figure when you don't need any

It's better to see once etc. – true, but sometimes it is easier to see three numbers in a table than three lonely points in a plot. Moreover, when *connected*, or even *interpolated*, the plot starts to imply some trends which may be nonexistent. A couple of examples where a figure could really have been spared:





Type of graphics you'd typically create yourself, appropriate tools, and things to consider

Besides the figures which come "as they are" (photos; figures borrowed from other sources), for a scientific publication you may need to create i) drawings like schemas, diagrams etc., probably with text; ii) data plots in two or three dimensions; iii) crystal structures and related properties, like electronic density etc.

For drawing purposes, I'd generally *discourage* from use of simple draw/paint programs, because their result is an integrated bitmap, on which it is very difficult to make subsequent changes – enlarging the boxes, reformatting text etc. Under Linux, I'd strongly recommend **Xfig** program for any drawing purposes, because it is very rich in options to create and manipulate vector objects (boxes, circles, arrows, interpolated curves, text, etc.). Take into account that the system of *layers* (the depth attributed to each element) which strictly control which element overlays which other one. The (postscript) output from **Xfig** is nice, strict and perfectly scalable to any size, so it looks perfectly on big posters. Should you need a bitmap of the resulting figure, my impression is that it is much better *first* to export it into postscript and *then* convert it to bitmap of the desired resolution, using **Gimp** or other software. Exporting the figure into a bitmap format directly from **Xfig**, in my impression, results in much inferior quality. A disadvantage of **Xfig** is that its color palette is rather poor and (in my knowledge) do not allow to choose other colors, or create color changes or other effects, like shadows etc. An interesting advantage is that one can import into **Xfig** an external figure in any bitmap format and draw *over it* arrows, boxes, comments or whatever you find necessary (just make sure that the layer of the graphic objects you create is *above* (less than) the layer of the imported figure).

As a simpler alternative to **Xfig** (which exists only under Unix X-Windows system, I

think), the drawing tools provided with MS Word© or Powerpoint©, or OpenOffice, – arrows, ovals, tex boxes etc. – would allow do basically the same type of job, even if offering not that rich set of tools.

For **data plots** in two dimensions, e.g. $Y(X)$ curves, I strongly recommend using **grace**, which runs however only under Unix, because it is very intuitive, rich in options, and stores the figure in its native readable ASCII format, which can be edited by hand if necessary. **Origin**© may probably do equally good job, but as a commercial software it cannot be installed at any computer at hand, and its native format is binary, that makes it difficult to repair corrupted data. Apart from this, **gnuplot** offers extreme richness of options, with a handicap that its command-line system almost precludes its use by a non-initialized user. Whatever the code in question, the people usually *know* how to make a $Y(X)$ plot. What they know less is how to make a *good* plot and to avoid usual mistakes, not confined to any given software. These usual mistakes are: • too small axis labels, inscripts; • too much empty space around too densely grouped data points; • wrong or badly chosen axis labeling. Examples of the latter are, e.g., tic labels like 10000, 20000, 30000 Wt, instead of putting "10^3 Wt" into the axis label, or tic with numbers along the axis which says "arbitrary units".

For **crystal structures**, among many commercial and freeware codes which can be really found suited to satisfy everyone's tastes, I'll single out just one, which I most frequently used so far. **XCrySDen** by Tone Kokalj allows to show isolated (molecular) and periodic structures and offers rich palette of options (colors, sizes, light parameters). (...) Note that the outut can be saved either bitmapped, or as vector Encapsulated PostScript. In the first regime, set by pressing "Display $\rightarrow$ Loghting On" (or F1), what is actually saved (on choosing File $\rightarrow$ Print) is the contents of the code's working screen. In order to not lose the resolution for nothing, enlarge the working field and (by pressing "Zoom +") the size of the molecule/crystal structure in it. Don't forget to change the black background (which sets on by default) to a more cartridge-friendly one. Among the formats suggested for saving, one finds EPS, but don't be misled – the output will be bitmapped anyway (with the resolution you've chosen by tuning the working screen size). On the contrary, in the "Display $\rightarrow$ Lighting Off" (or F2) regime, the "File $\rightarrow$ Print $\rightarrow$ Files of type: EPS" stores the output in scalable vector .eps, however, without the information on shading / light settings / arrows.