

PRACE
Winter School
2013

Scientific Visualization

SOME CONCEPTS, TOOLS & LIBRARIES

Nicolas P. Rougier

Introduction

Audience

- Yourself
- Scientific community
- Students
- Media

Criterion

- Quality
- Speed
- Development time

Usage

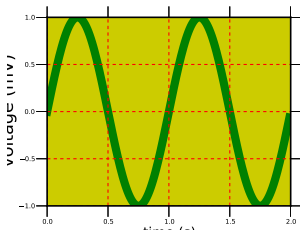
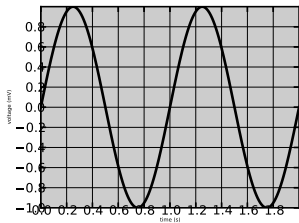
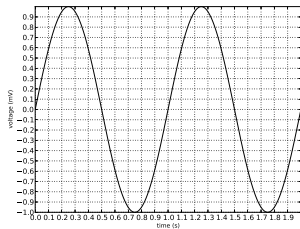
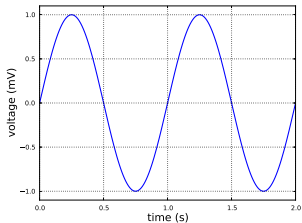
- Runtime visualization
- Final visualization
- Illustration
- Demonstration

Nature of data

- 2D, 3D, 4D, ...
- Continuous, discrete, ...
- Numeric, symbolic, ...

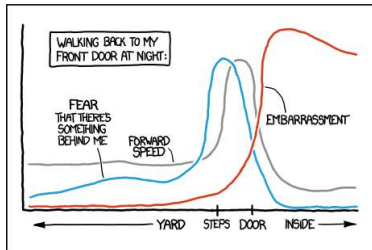
The good, the bad & the ugly...

...and the very ugly

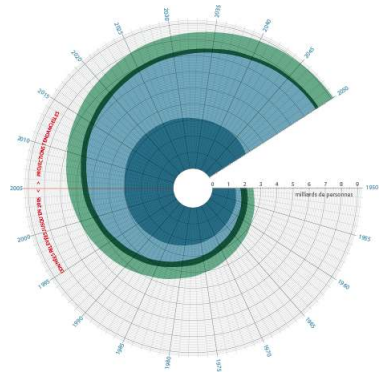


Readability first

Beauty is an option



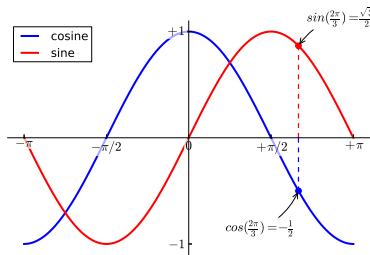
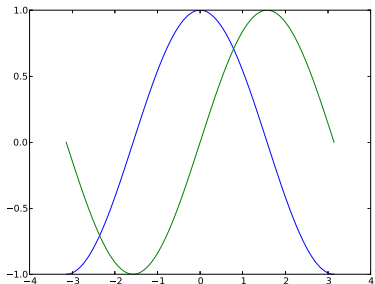
(<http://xkcd.com/1064/>)



What's the point of this polar axis ?

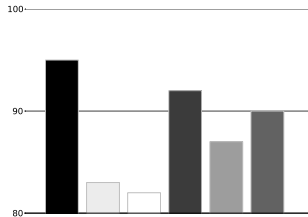
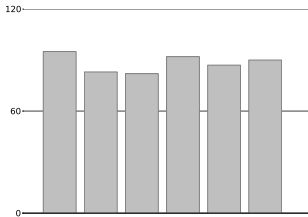
Do not trust the defaults

Defaults are never good for a specific case



Be fair to your data

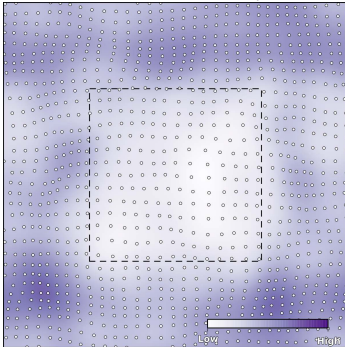
Don't hide reality



Get the right tool

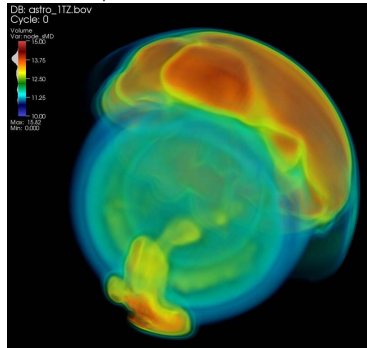
32 000 cores to plot $\sin(x)$ may be slightly overkill...

Model of somatosensory cortex



≈ 1 000 points, matplotlib, single core

Supernova simulation



≈ 2 trillion points, Vist, 32000 cores

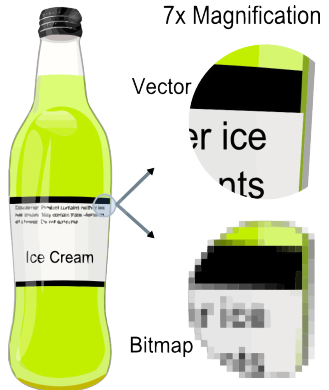
Image formats

Bitmap/Raster Image format

- Matrix of pixels
- Fixed native resolution
- B&W, grayscale, color, HDR
- PNG, JPG, TIFF

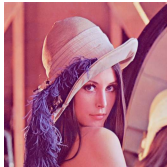
Vector Image format

- Geometrical primitives
- No fixed resolution
- B&W, grayscale, color
- SVG, PDF, PS

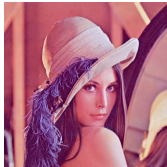


Bitmap Image Compression

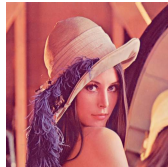
Lossless compression (png, bmp, tiff)



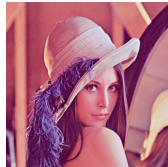
quality=0 (507k)



quality=10 (702k)

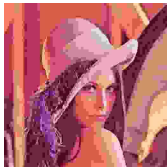


quality=50 (712k)

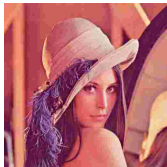


quality=100 (717k)

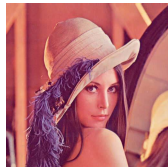
Lossy data compression (jpg)



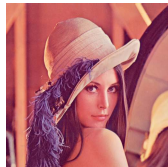
quality=0 (3k)



quality=10 (7k)



quality=50 (30k)



quality=100 (400k)

Bitmap Image Resolution

DPI (dots per inch)

- 1 inch = 2.54 cm
- 1000x1000 pixels at 250dpi = 4 inches x 4 inches area at most

Figures should be rendered at 600dpi

- Double-column article on A4 paper
 - $(21 - 2 \times 2 (\approx \text{margins}) - 1 (\approx \text{col. sep.})) / 2 \approx 8 \text{ cm}$
 - $8 / 2.54 * 600 = 1889 \text{ pixels} \approx \mathbf{2000 \text{ pixels wide}}$
- Single-column article on A4 paper
 - $(21 - 2 \times 2 (\approx \text{margins})) \approx 17 \text{ cm}$
 - $17 / 2.54 * 600 = 4015 \text{ pixels} \approx \mathbf{4000 \text{ pixels wide}}$

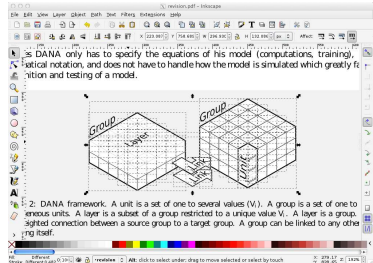
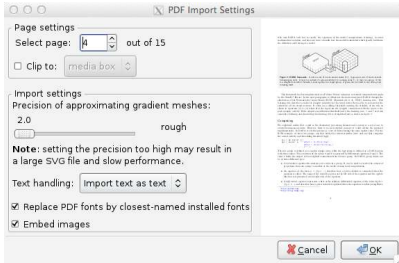
Drawing software

Vector

- xfig
 - Old-school and limited font support
 - pdf/svg/eps/bitmap import/export
- inkscape
 - Unix standard
 - pdf/svg/eps/bitmap import/export

Bitmap

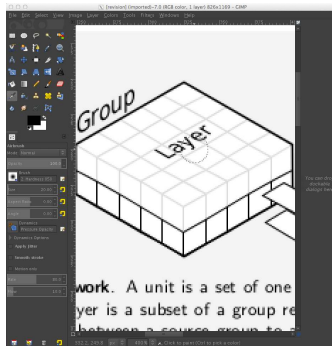
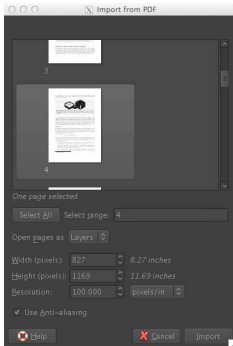
- gimp
 - Unix standard
 - bitmap import/export, vector import



An Open Source vector graphics editor, with capabilities similar to Illustrator, CorelDraw, or Xara X, using the W3C standard Scalable Vector Graphics (SVG) file format.

Gimp

www.gimp.org



Gimp can be used as a simple paint program, an expert quality photo retouching program, an online batch processing system, a mass production image renderer, an image format converter, etc.

Drawing tools

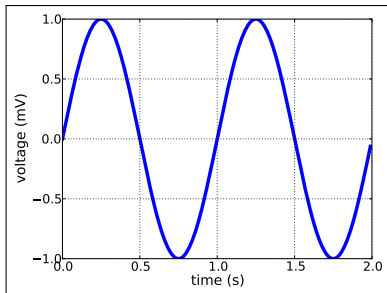
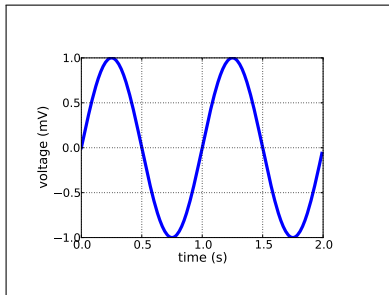
Bitmap

- ImageMagick
 - convert between image formats as well as resize an image, blur, crop, despeckle, dither, draw on, flip, join, re-sample
- ffmpeg (ffmpeg.org)
 - record, convert and/or stream audio and/or video.

Vector

- pdjam
- pdfcrop
- graphviz
- tikz

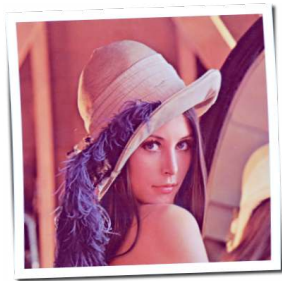
\$ pdfcrop figure.pdf figure.pdf



ImageMagick

www.imagemagick.org

```
$ convert lena.png +polaroid polaroid.png
```

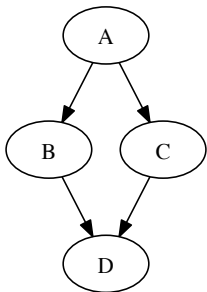


More effects at www.fmwconcepts.com/imagemagick/index.php

graph.dot:

```
digraph G { A->B; A->C; B->D; C->D }
```

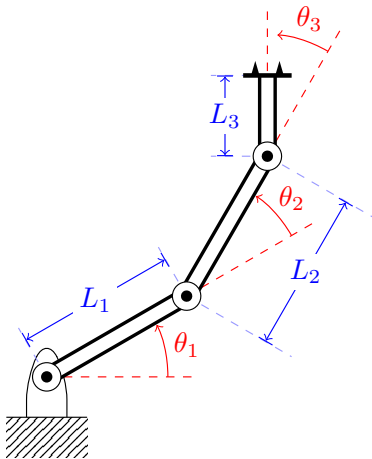
```
$ dot graph.dot -Tpdf -o graph.pdf
```



Tikz

www.texample.net/tikz

```
% Define commands for links, joints and such
\def\link{\draw [double distance=1.5mm,
                very thick] (0,0)--}
\def\joint{%
  \filldraw [fill=white] (0,0) circle (5pt);
  \fill[black] circle (2pt);
}
\def\grip{%
  \draw[ultra thick](0cm,\dg)--(0cm,-\dg);
  \fill (0cm, 0.5\dg)+(0cm,1.5pt) --
    +(0.6\dg,0cm) -- +(0pt,-1.5pt);
  \fill (0cm, -0.5\dg)+(0cm,1.5pt) --
    +(0.6\dg,0cm) -- +(0pt,-1.5pt);
}
\def\robotbase{%
  \draw[rounded corners=8pt]
    (-\dw,-\dh)-- (-\dw, 0) --
    (0,\dh)--(\dw,0)--(\dw,-\dh);
  \draw (-0.5,-\dh)-- (0.5,-\dh);
  \fill[pattern=north east lines]
    (-0.5,-1) rectangle (0.5,-\dh);
}
```



Plotting tools

Free

- gnuplot
`www.gnuplot.info`
- matplotlib
`matplotlib.sourceforge.net`
- R
`www.r-project.org`
- mayavi
`mayavi.sourceforge.net`

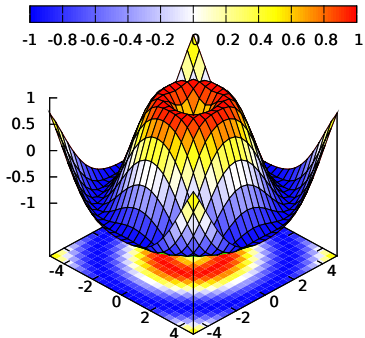
Not so free

- grapher (mac only)
`wikipedia.org/wiki/Grapher`
- maple
`www.maplesoft.com`
- matlab
`www.mathworks.com`
- mathematica
`www.wolfram.com`

gnuplot

www.gnuplot.org

```
set style line 100 lt -1 lw 0.1
set pm3d
set pm3d at b
set palette defined ( 0 "blue", .5 "white", \
                    .75 "yellow", 1 "red")
set colorbox horiz user origin .1,.9 size .8,.04
set view 55,45
set nokey
set hidden3d
set isosamples 25
set term pdf size 3in,3in
set output 'surface-gnuplot.pdf'
set xrange [-5:+5]
set yrange [-5:+5]
set zrange [-1:+1]
set multiplot
splot sin(sqrt(x*x+y*y)) with dots
set pm3d
set pm3d solid hidden3d 100
splot sin(sqrt(x*x+y*y)) with lines
unset multiplot
```

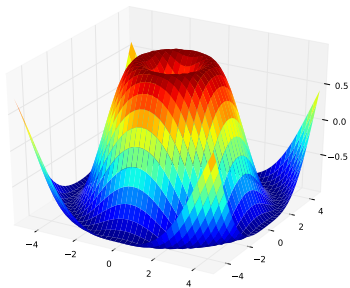


matplotlib

matplotlib.sourceforge.net

```
from pylab import *
from mpl_toolkits.mplot3d import Axes3D

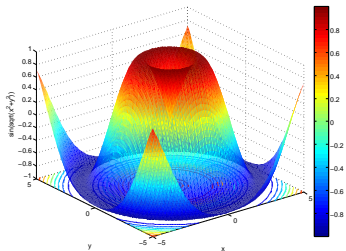
ax = Axes3D(fig)
T = np.arange(-5, 5, 0.25)
X, Y = np.meshgrid(T,T)
Z = np.sin(np.sqrt(X**2 + Y**2))
ax.plot_surface(X, Y, Z, rstride=1, cstride=1, cmap='jet')
```



matlab

www.mathworks.com

```
[x,y]=meshgrid(-5:0.1:5,-5:0.1:5);  
surf(x,y,sin(sqrt(x.^2+y.^2)),  
      'EdgeColor','none')  
colorbar  
xlabel('x')  
ylabel('y')  
zlabel('sin(sqrt(x^2+y^2))')
```



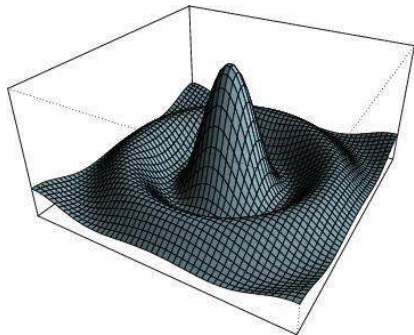
R

www.r-project.org

```
x <- seq(-10, 10, length = 50)
y <- x
rotsinc <- function(x,y)
{
  sinc <- function(x)
  {
    y <- sin(x)/x ; y[is.na(y)] <- 1; y
  }
  10 * sinc( sqrt(x^2+y^2) )
}
sinc.exp <- expression(z == Sinc(sqrt(x^2 + y^2)))

z <- outer(x, y, rotsinc)

par(bg = "white",mfrow=c(1,2),mar=rep(1.5,4))
persp(x, y, z, theta = 30, phi = 30,
      expand = 0.5, col = "lightblue",
      ltheta = 120, shade = 0.75,
      xlab = "X", ylab = "Y", zlab = "Z")
```



Drawing/plotting libraries

2D (points, lines, bezier curves, etc.)

- cairo (www.cairo.org, c/c++/python)
- agg (www.antigrain.com, c+/c++)
- gnuplot (c/c++/python), 2d/2.5d
- matplotlib (python), 2d/2.5d
- d3 (d3js.org, javascript)

3D

- OpenGL (www.opengl.org, c/python)
- VTK (c/c++/python, not for the faint of heart)
- mayavi.mlab (python)

Visualization tools

Free

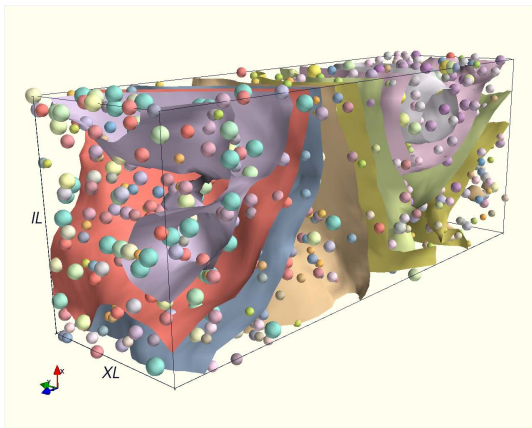
- Mayavi
`mayavi.sourceforge.net`
- VisIt
`matplotlib.sourceforge.net`
- Paraview
`www.scilab.org`

Not so free

- matlab
`www.mathworks.com`
- mathematica
`www.wolfram.com`
- etc.

Mayavi

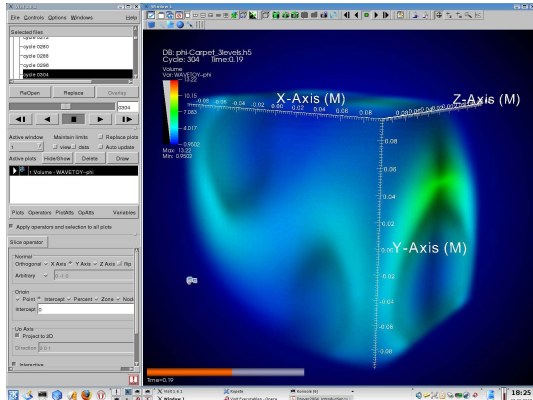
code.enthought.com/projects/mayavi/



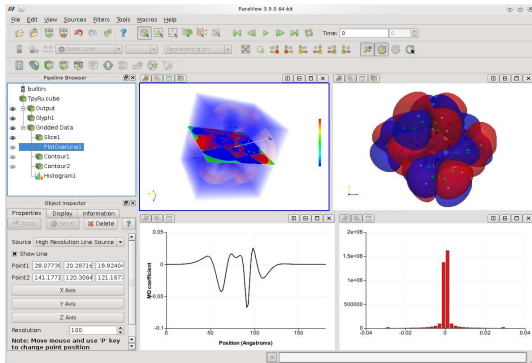
MayaVi is a scientific data visualizer written in Python and uses the Visualization Toolkit (VTK) for the visualization.

VisIt

wci.llnl.gov/codes/visit/



VisIt is a free interactive parallel visualization and graphical analysis tool for viewing scientific data on Unix and PC platforms.



ParaView is an open-source, multi-platform data analysis and visualization application.

First Aid Kit I

Tools

- ImageMagick (conversion)
- Gimp (bitmap images creation/manipulation)
- Inkscape (vector creation/images manipulation)
- Gnuplot (visualization/illustration, 2d/2.5d)
- Matplotlib (visualization/illustration 2d/2.5d)
- ffmpeg (movie creation/manipulation)

Environment

- IPython
→ IPython provides a rich toolkit to help you make the most out of using Python.

First Aid Kit 2

Libraries

- Drawing/plotting
 - matplotlib (python, 2d/2.5d)
 - d3 (javascript, interactive, 2d)
 - R (R, 2d/3d)
- Visualization (heavy duty)
 - Mayavi
 - VisIt
 - Paraview

Questions ?

