开源地球物理学校（第一期）
2025年8月19日-21日 | 吉林 • 长春

Reproducible Paper:

Writing technical papers using Madagascar and LaTeX

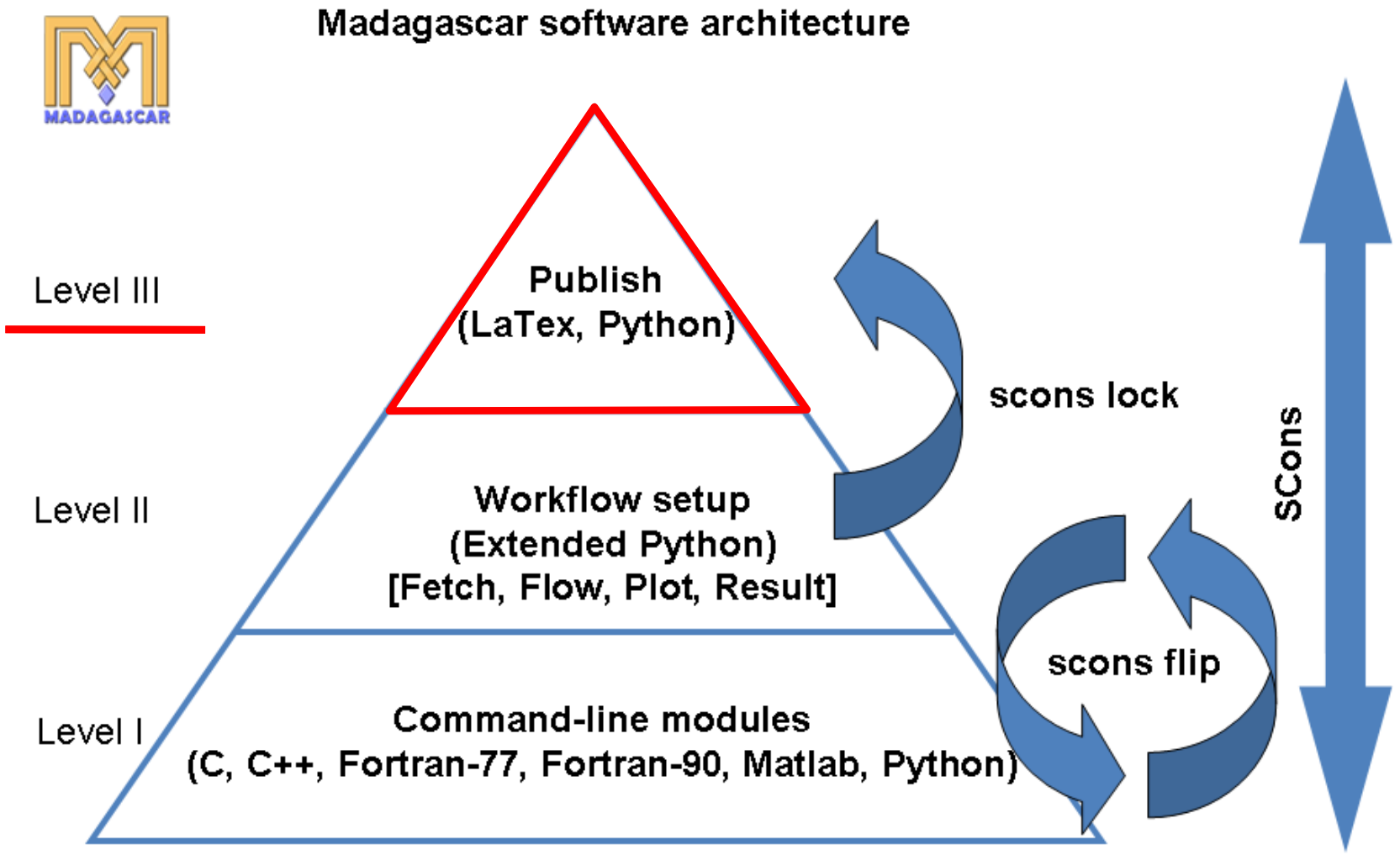
Jiawei Chen*, Yang Liu
College of Geo-exploration Sciences and Technology
Jilin University
chenjw22@mails.jlu.edu.cn, yangliu1979@jlu.edu.cn

Aug 19, 2025

Outline

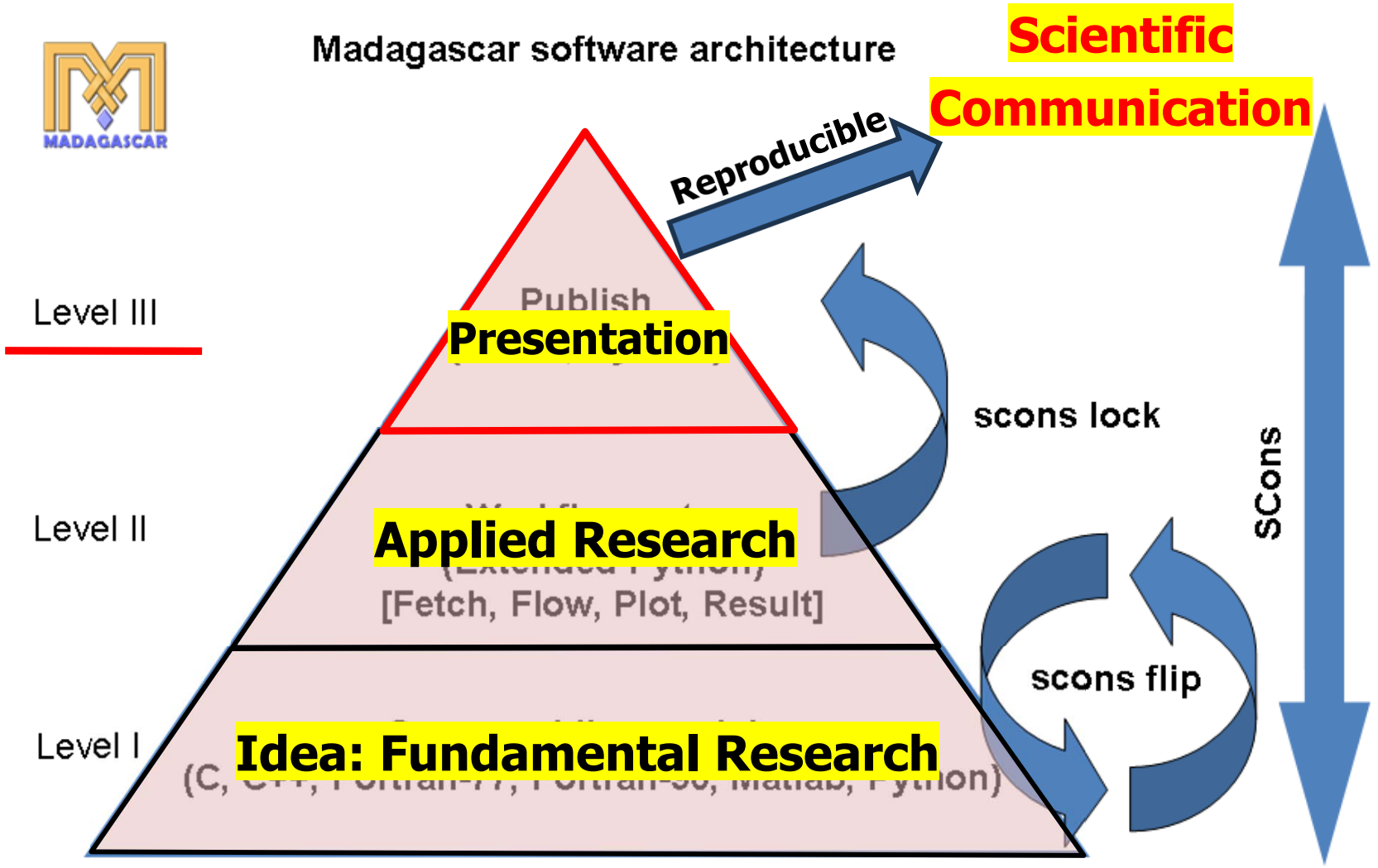
- **Motivations**
- Prerequisites
- Figure preparation
  - Plotting with M8R
  - Plotting with Python
  - Plotting with Matlab/Octave
  - Plotting with GMT
- SEGTeX
- Alternative LaTeX template
- Simple Example

Motivations



Fomel, S, Sava, P, Vlad, I, Liu, Y, and Bashkardin, V 2013. Madagascar: open-source software project for multidimensional data analysis and reproducible computational experiments. Journal of Open Research Software 1(1): e8, DOI: http://dx.doi.org/10.5334/jors.ag

Motivations



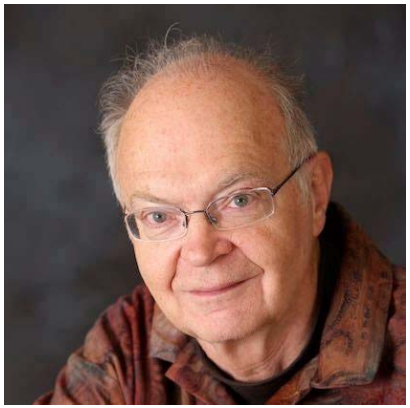
Fomel, S, Sava, P, Vlad, I, Liu, Y, and Bashkardin, V 2013. Madagascar: open-source software project for multidimensional data analysis and reproducible computational experiments. Journal of Open Research Software 1(1): e8, DOI: http://dx.doi.org/10.5334/jors.ag

Motivations

- MS-word--> LaTeX

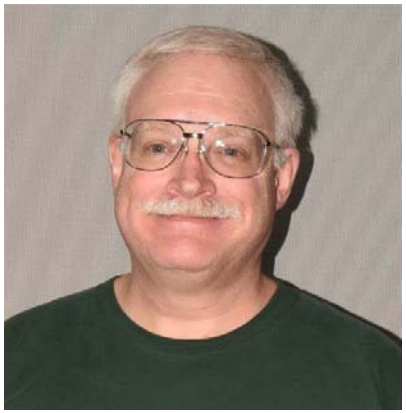
"I hope to die before I have to use Microsoft Word"

– Donald Ervin Knuth



Reasons:

- 1) Focus on paper structure and content;
- 2) Better mathematic style;
- 3) More efficient (easy comments);
- 4) Better revision for long document;
- 5) Sole high-quality text software in Linux;
- 6) Compiling by command line, supported by Madagascar and suitable for reproducible research.



- "All technical papers should be "reproducible" in the sense that someone of reasonable skill ought to be able to read the paper and then reproduce the results." – Joe Dellinger
- Remember, LaTeX is not perfect, either MS-word or LaTeX is just a tool, choose any of them you need.

Vibe/LLM Coding makes it easier than ever

ChatGPT

deepseek

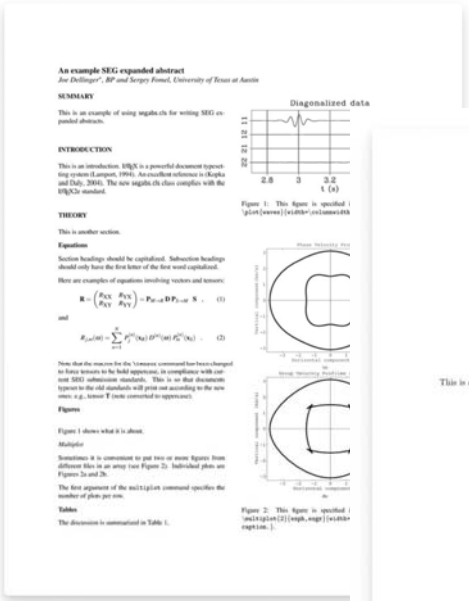
Gemini

- **Fully or largely give in to the vibes.**
- **Embrace exponentials.**
- **Forget that the code even exists.**

SEGTeX : When LaTeX meets Madagascar

https://library.seg.org/page/gpysa7/ifa/latex-package

SEGTeX : When LaTeX meets Madagascar



Template for SEG abstract

Template for writing SEG abstract.

Sergey Fomel

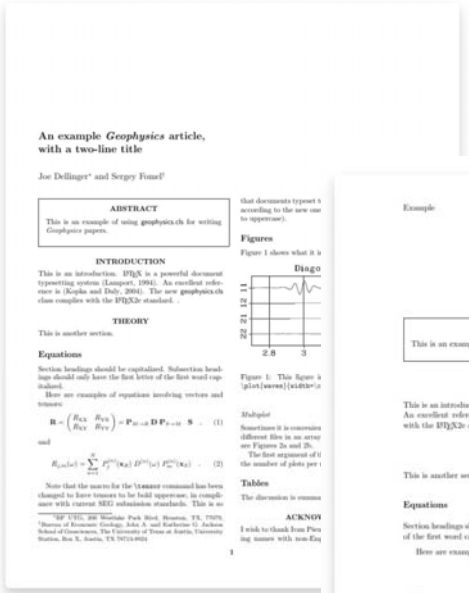
Template for GEOPHYSICS

GEOPHYSICS, published by the Society of Geophysicists since 1936, is an archival jo

Sergey Fomel

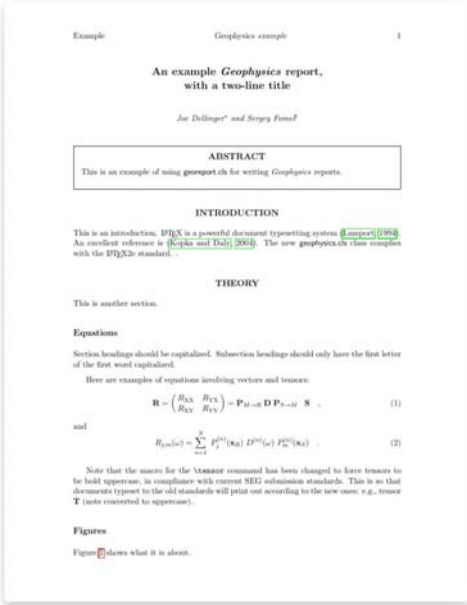
An example EAGE expanc abstract

SEGTeX is a LaTeX package for geophys publications. This template consists of



An example 2-column Ge article

This is an example of using geophysic two-column Geophysics papers.



An example Geophysics report

This is an example of using geopreport.cls for writing Geophysics reports.

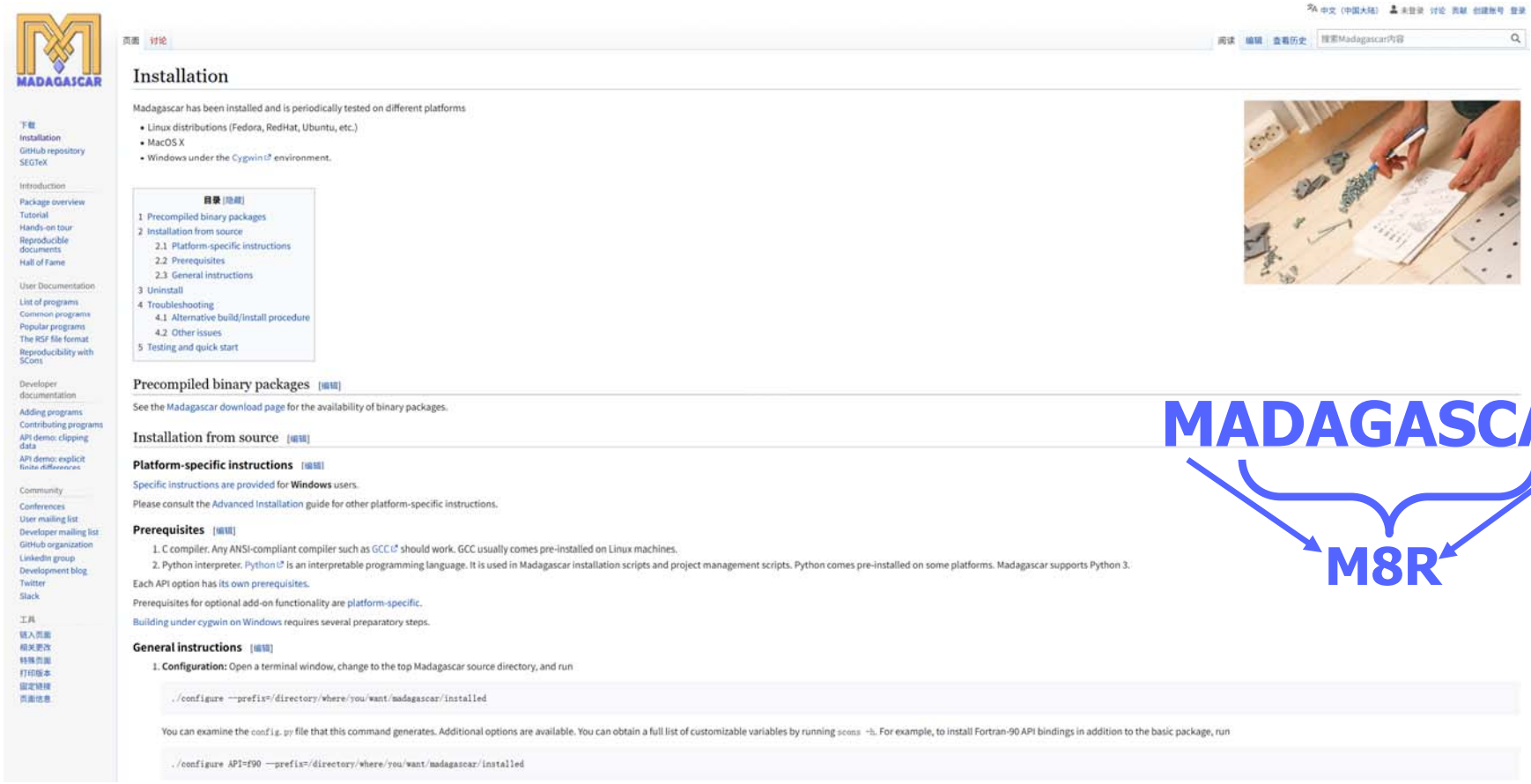
SEGTeX

Outline

- Motivations
- **Prerequisites**
- Figure preparation
  - Plotting with M8R
  - Plotting with Python
  - Plotting with Matlab/Octave
  - Plotting with GMT
- SEGTeX
- Alternative LaTeX template
- Simple Example

Prerequisites

1) Download and install Madagascar



MADAGASCAR

Installation

Madagascar has been installed and is periodically tested on different platforms

- Linux distributions (Fedora, RedHat, Ubuntu, etc.)
- MacOS X
- Windows under the Cygwin environment.

目录 [隐藏]

1 Precompiled binary packages
2 Installation from source
2.1 Platform-specific instructions
2.2 Prerequisites
2.3 General instructions
3 Uninstall
4 Troubleshooting
4.1 Alternative build/install procedure
4.2 Other issues
5 Testing and quick start

Precompiled binary packages [编辑]

See the Madagascar download page for the availability of binary packages.

Installation from source [编辑]

Platform-specific instructions [编辑]

Specific instructions are provided for **Windows** users.

Please consult the Advanced Installation guide for other platform-specific instructions.

Prerequisites [编辑]

1. C compiler. Any ANSI-compliant compiler such as GCC should work. GCC usually comes pre-installed on Linux machines.
2. Python interpreter. Python is an interpretable programming language. It is used in Madagascar installation scripts and project management scripts. Python comes pre-installed on some platforms. Madagascar supports Python 3.

Each API option has its own prerequisites.

Prerequisites for optional add-on functionality are platform-specific.

Building under cygwin on Windows requires several preparatory steps.

General instructions [编辑]

1. **Configuration:** Open a terminal window, change to the top Madagascar source directory, and run

```
./configure --prefix=/directory/where/you/want/madagascar/installed
```

You can examine the config.py file that this command generates. Additional options are available. You can obtain a full list of customizable variables by running scons -h. For example, to install Fortran-90 API bindings in addition to the basic package, run

```
./configure API=f90 --prefix=/directory/where/you/want/madagascar/installed
```


MADAGASCAR

M8R

https://ahay.org/wiki/Installation

2) Download and install LaTeX Package (TeXLive)

a. Manual download:

https://tug.org/texlive/

b. Package manager (Ubuntu 22.04)

```
sudo apt update
sudo apt install texlive-full
```

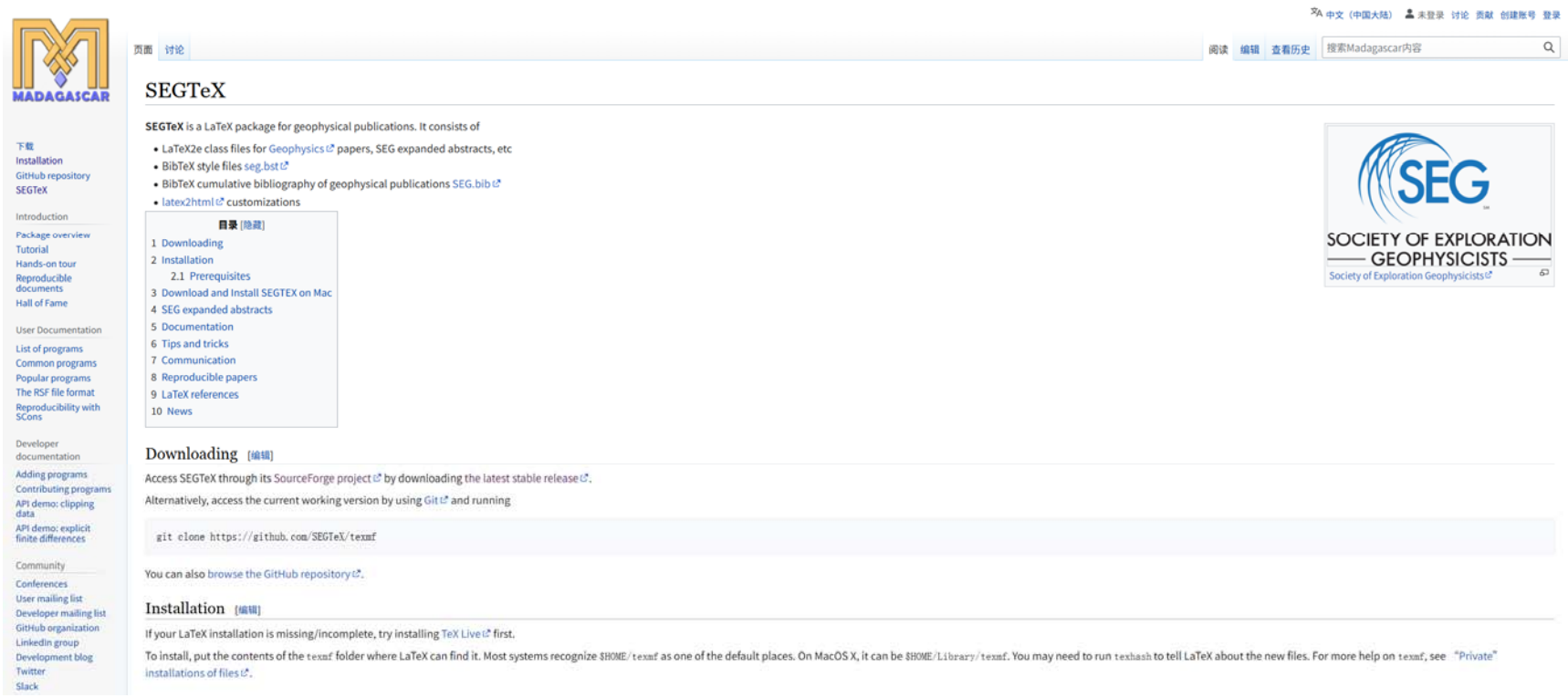
install texlive is OK, if you know exactly which packages you'll need

Package	Installed Size	Main Dependencies
texlive	~350 Mb	texlive-latex-base, texlive-latex-recommended, texlive-fonts-recommended
texlive-full (recommended)	~5.3 Gb	Entire TeXLive distribution

Prerequisites

3) Download and configure SEGTeX

(1) Download from: http://www.ahay.org/wiki/SEGTeX



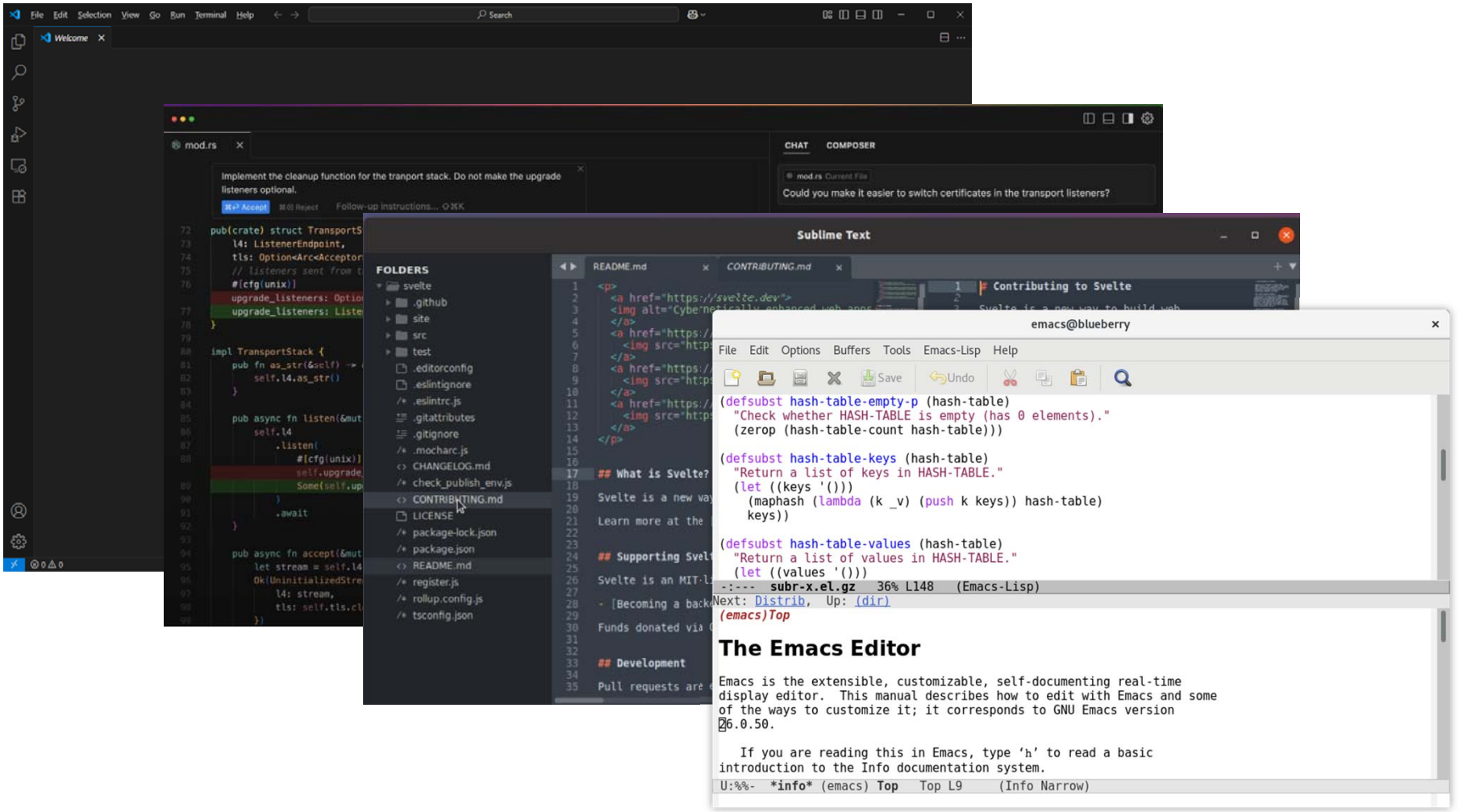
(2) Add environment variable TEXMFHOME:

```
echo "export TEXMFHOME=$HOME/texmf" >> ~/.bashrc
```

Prerequisites

4) Choose your favorite text editor

VSCode, Cursor, Sublime Text, Emacs, Gedit, Vim, ...



Outline

- Motivations
- Prerequisites
- Figure preparation
  - Plotting with M8R
  - Plotting with Python
  - Plotting with Matlab/Octave
  - Plotting with GMT
- SEGTeX
- Alternative LaTeX template
- Simple Example

Plotting with M8R

Plotting with Madagascar

- **Plotting tools: rsf → vpl**
  - sfgraph
  - sfgraph3
  - sfcontour
  - sfcontour3
  - sfwiggle
  - sfgrey
  - sfgrey3
  - sfgrey4

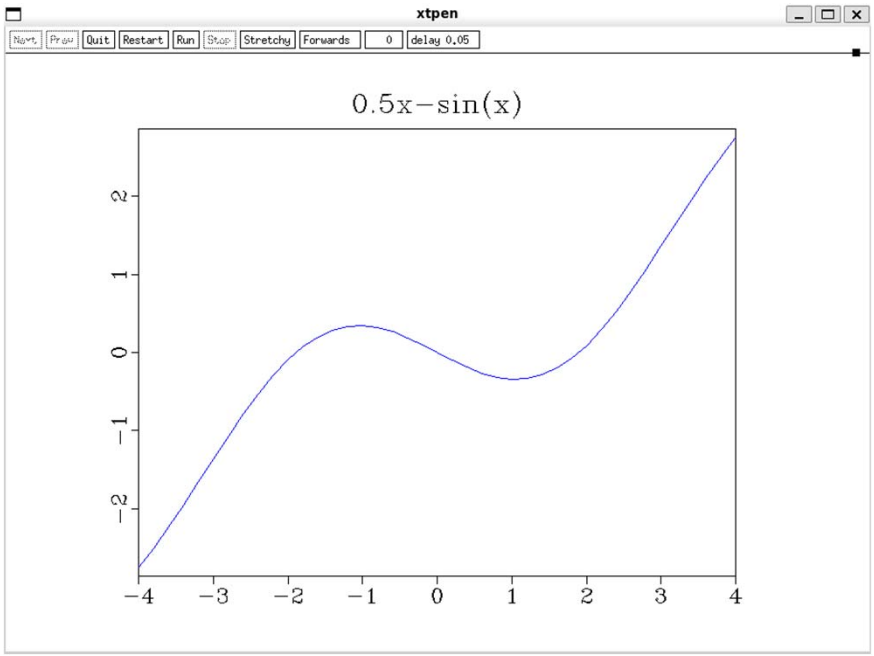
- **Pens: vpl → jpg/png/tif/gif…**
  - vppen
  - pspen
  - xtpen
  - ……

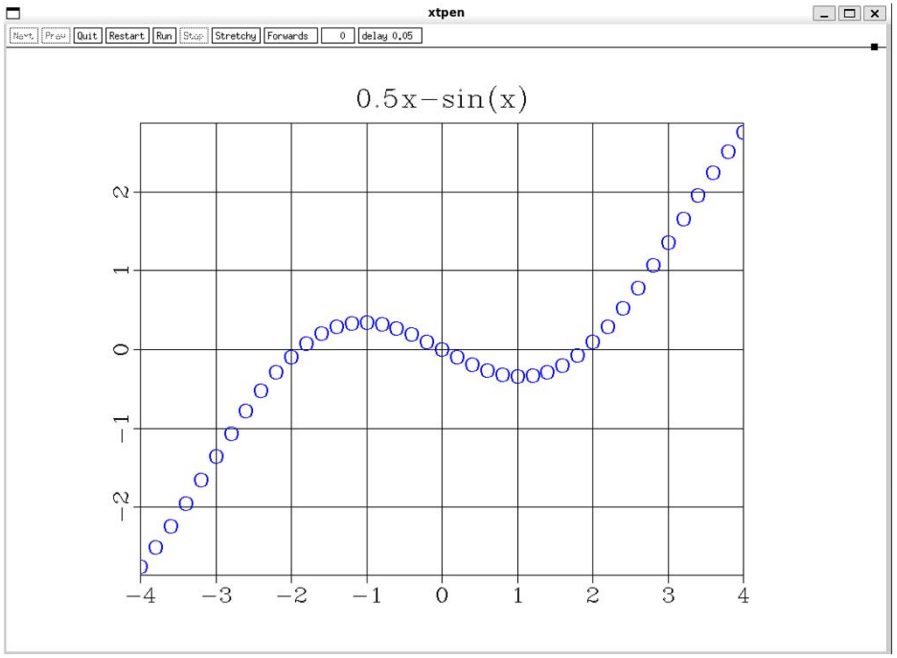
Convert/Show/Filter figures

Find doc also from sfsdtplot:
http://ahay.org/RSF/sfstdplot.html

sfgraph

```
sfmath n1=41 o1=-4 d1=.2 output=".5*x1-sin(x1)" > y.rsf

< y.rsf sfgraph title="0.5x-sin(x)" min1=-4 max1=4 | xtpen
< y.rsf sfgraph title="0.5x-sin(x)" symbol=o symbolsz=12 grid=y min1=-4 max1=4 | xtpen
```

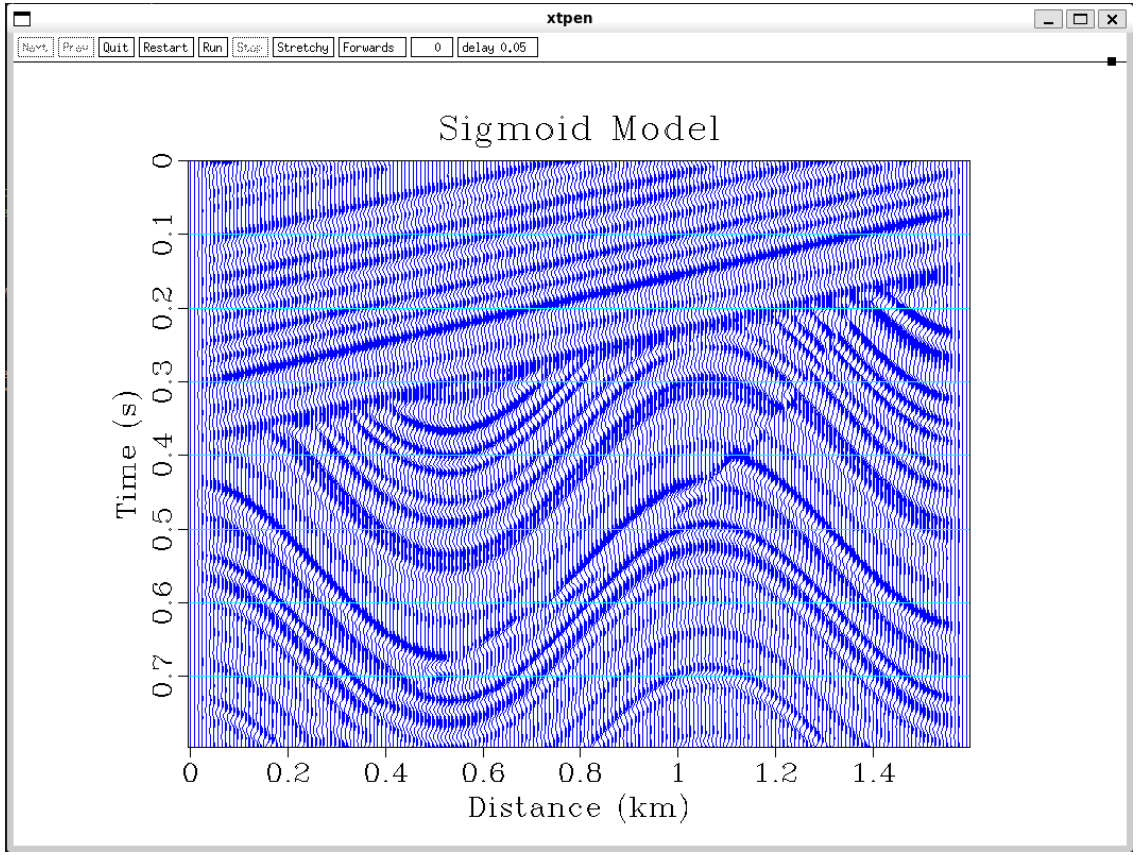




export SFPENOPTS='bgcolor=w'

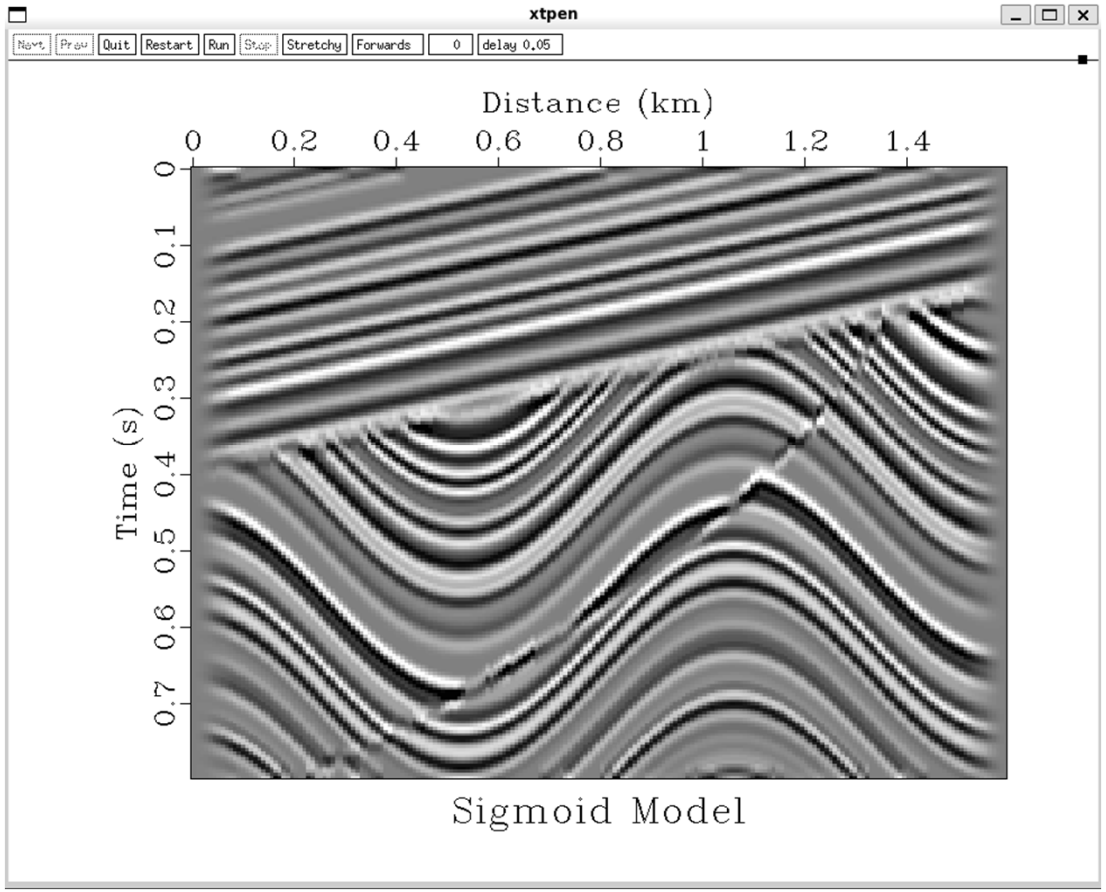
sfwiggle

```
sfsigmoid n1=200 n2=200 d1=.004 d2=.008 \
    | sfsmooth rect1=3 diff1=1 | sfsmooth rect1=3 \
    | sfput label2=Distance > model.rsf
< model.rsf sfwiggle yreverse=y transp=y poly=y title="Sigmoid Model" | xtpen
```



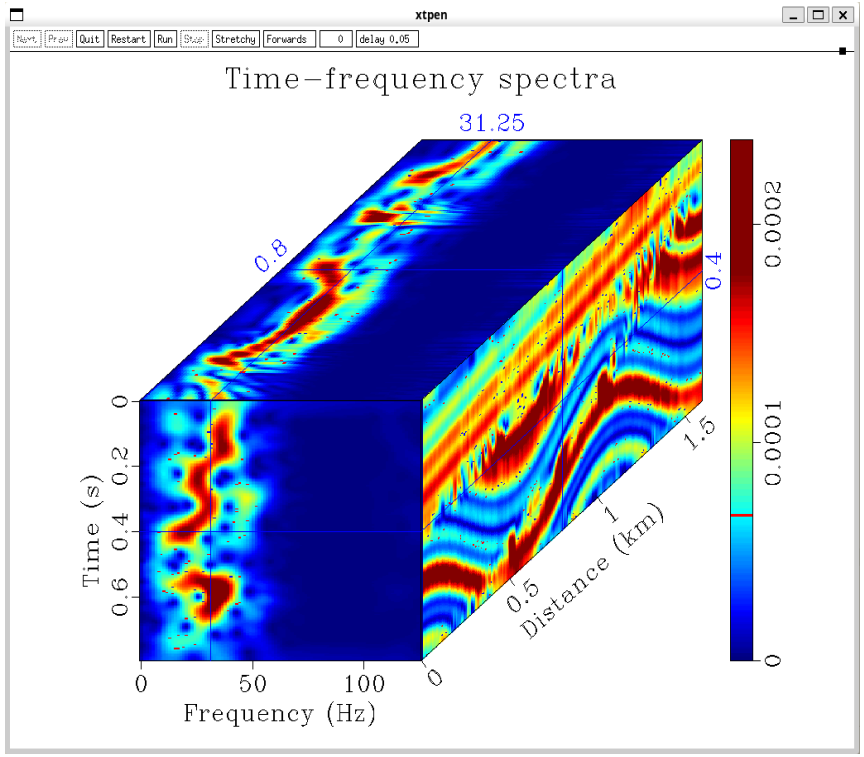
sfgrey

```
sfsigmoid n1=200 n2=200 d1=.004 d2=.008 \
    | sfsmooth rect1=3 diff1=1 | sfsmooth rect1=3 \
    | sfput label2=Distance > model.rsf
< model.rsf sfgrey title="Sigmoid Model" | xtpen
```

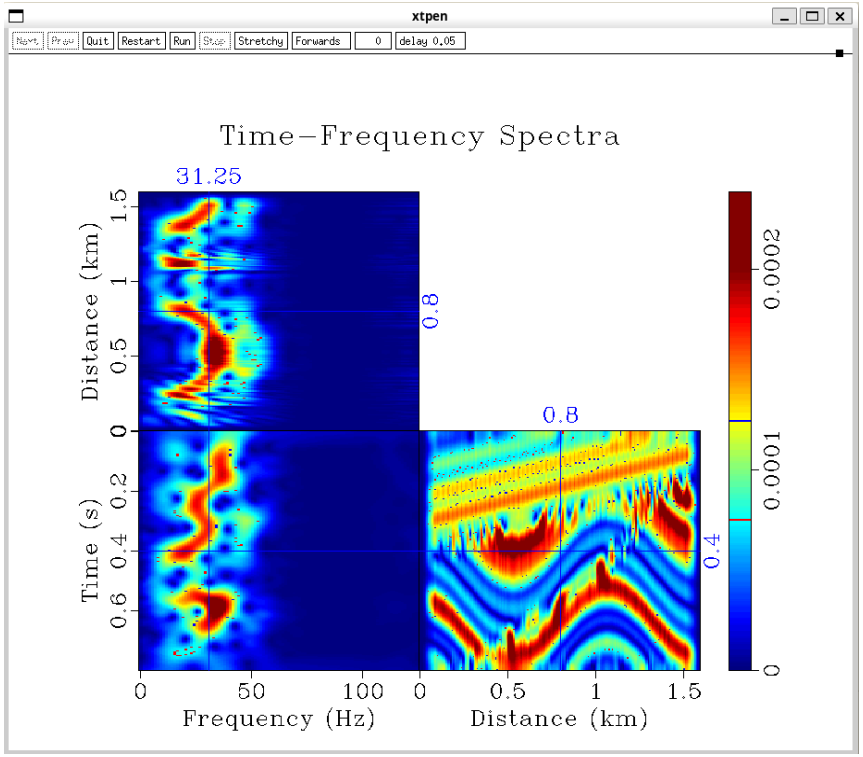


sfgrey3

```
< model.rsf sfltft niter=10 rect=5 | sfcabs > ltft.rsf
< ltft.rsf sfbyte bar=bar.rsf allpos=y clip=2e-4 \
    | sfgrey3 title="Time-frequency spectra" bar=bar.rsf scalebar=y \
      color=j frame1=100 frame2=25 frame3=100 flat=n | xtpen
```



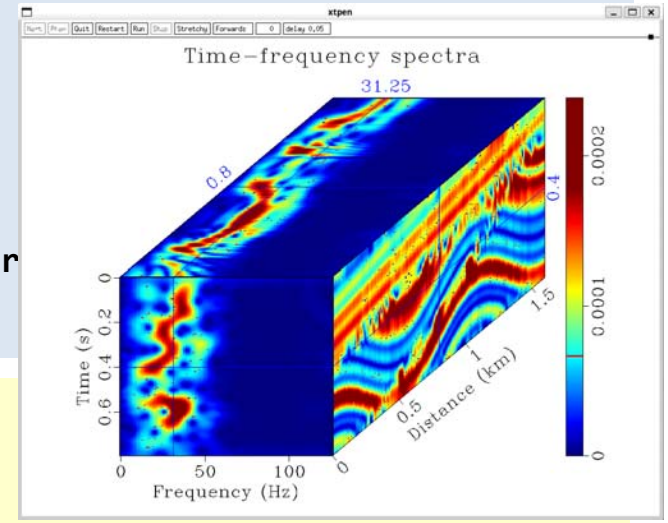
flat=n



Default: flat=y

Command line versus SCons

```
sfsigmoid n1=200 n2=200 d1=.004 d2=.008 \
    | sfsmooth rect1=3 diff1=1 | sfsmooth rect1=3 \
    | sfput label2=Distance > model.rsf
< model.rsf sfltft niter=10 rect=5 | sfcabs > ltft.rsf
< ltft.rsf sfbyte bar=bar.rsf allpos=y clip=2e-4 \
    | sfgrey3 title="Time-frequency spectra" bar=bar.rsf scalebar
      color=j frame1=100 frame2=25 frame3=100 flat=n | sfpen
```




```
from rsf.proj import *

Flow('model',None,
     '''
      sigmoid n1=200 n2=200 d1=.004 d2=.008 | smooth rect1=3 diff1=1 |
      smooth rect1=3 | put label2=Distance
     ''')

Flow('ltft', 'model', 'ltft niter=10 rect=5 | cabs')

Result('ltft','ltft',
       '''
        byte bar=bar.rsf allpos=y clip=2e-4 |
        grey3 title="Time-Frequency Spectra" bar=bar.rsf scalebar=y
        color=j frame1=100 frame2=25 frame3=100 flat=n
       ''')

End()
```

scons ltft.view

sfdoc stdplot

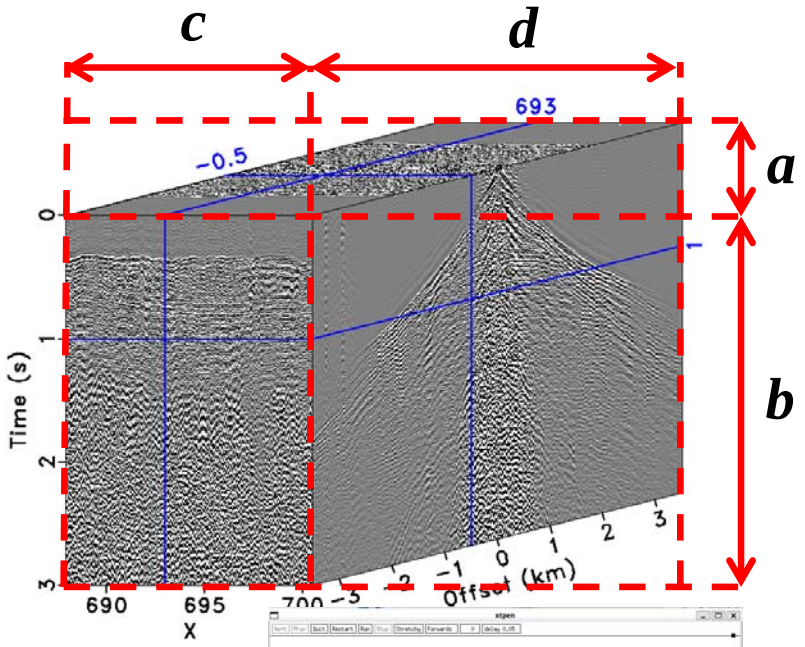
```
NAME
        sfstdplot
DESCRIPTION
        Setting up frames for a generic plot.
SYNOPSIS
        sfstdplot backcol= fillcol= dash= plotfat= plotcol= xreverse=xreverse1 yreverse=yreverse1 pad=pad1 scalebar=n barmove=n tickscale=0.5
tickscale1=tickscale tickscale2=tickscale tickscale3=tickscale tickscale4=tickscale min1=umin1 min2=umin2 max1=umax1 max2=umax2 font=-1 screen
ratio=VP_SCREEN_RATIO screenht=VP_STANDARD_HEIGHT screenwd=screenht / screenratio crowd=0.75 xinch= crowd1=crowd yinch= crowd2=crowd xll= xur=
 yll= yur= barwidth=0.36 axiscol=VP_WHITE framelabelcol=VP_YELLOW cubelinecol=framelabelcol labelsz=8. larnersz=0.0f labelrot=n grid1=transp?
false: grid grid2=transp? grid: false gridcol=grid? VP_RED: framecol gridfat=1 griddash=0.0f titlesz=10. barlabelsz= framelabel1=(bool) (NULL
!= label1) framelabel2=(bool) (NULL != label2) framelabel3=(bool) (NULL != label3) axisfat=0 axiscol=7 labelfat=0 labelsz=8. wantaxis= screenr
atio=VP_SCREEN_RATIO screenht=VP_STANDARD_HEIGHT screenwd=screenht / screenratio crowd=0.75 xinch= crowd1=crowd yinch= crowd2=crowd xll= xur=
yll= yur= transp=transp1 xreverse=n yreverse=yreverse1 labelrot=n min1= min2= max1= max2= wanttitle=y titlefat=0 titlesz=10. wantaxis= wantaxi
s1= wantaxis2= wantaxis3= labelfat= label1= unit1= label3= unit3= label2= unit2= nbartic= dbarnum= obarnum= wherebartics= n1tic= d1num= o1num=
 n2tic= d2num= o2num= n3tic= d3num= o3num= n4tic= d4num= o4num= wheretics= grid1= g1num0= g1num= grid2= g2num0= g2num= title= barlabelfat= bar
label= barunit= bartype= wherexlabel= whereylabel= formatbar= format2= format1= format3= wheretitle= wherebarlabel=
PARAMETERS
        int     axiscol=7
        int     axisfat=0
        floats  backcol=         [3]
        string  barlabel=       ( barlabel bar label )(bar label)
        int     barlabelfat=    bar label fatness
        float   barlabelsz=     bar label font size
        bool    barmove=n [y/n] adjust scalebar position, if bartype=h
        string  bartype=        [v,h] vertical or horizontal bar (default is v)
        string  barunit=        ( barunit bar unit )(bar unit)
        float   barwidth=0.36   scale bar size
        float   crowd=0.75
        float   crowd1=crowd
        float   crowd2=crowd
        int     cubelinecol=framelabelcol       cube lines color
        float   d1num=  axis1 tic increment
        float   d2num=  axis2 tic increment
        float   d3num=  axis3 tic increment
        float   d4num=  axis4 tic increment
        floats  dash=   line dash type
                    0 continuos (default)
                    1 fine dash
                    2 fine dot
                    3 dash
                    4 large dash
                    5 dot dash
                    6 large dash small dash
                    7 double dot
                    8 double dash
                    9 loose dash  The part after the decimal point determines the pattern repetition interval  [n2]
        string  dbarnum=        scalebar tic increment
        floats  fillcol=         [3]
        int     font=-1         font to use in text
        string  format1=        tick mark format
        string  format2=        tickmark format ()
 line 1 (press h for help or q to quit)
```

Tips for M8R plotting

1) Aspect ratio settings for sfgrey3

$$\text{point1} = \frac{b}{a+b} = 0.8$$

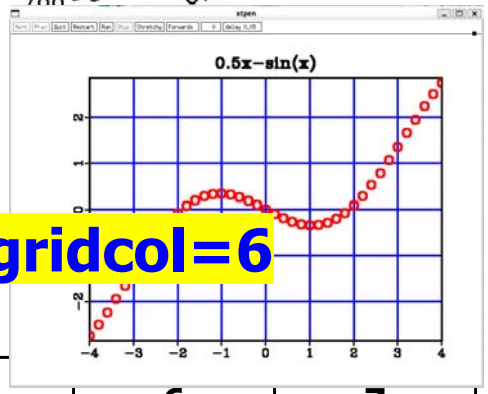
$$\text{point2} = \frac{c}{c+d} = 0.4$$



2) Element color

Lines, labels, frames, grids…

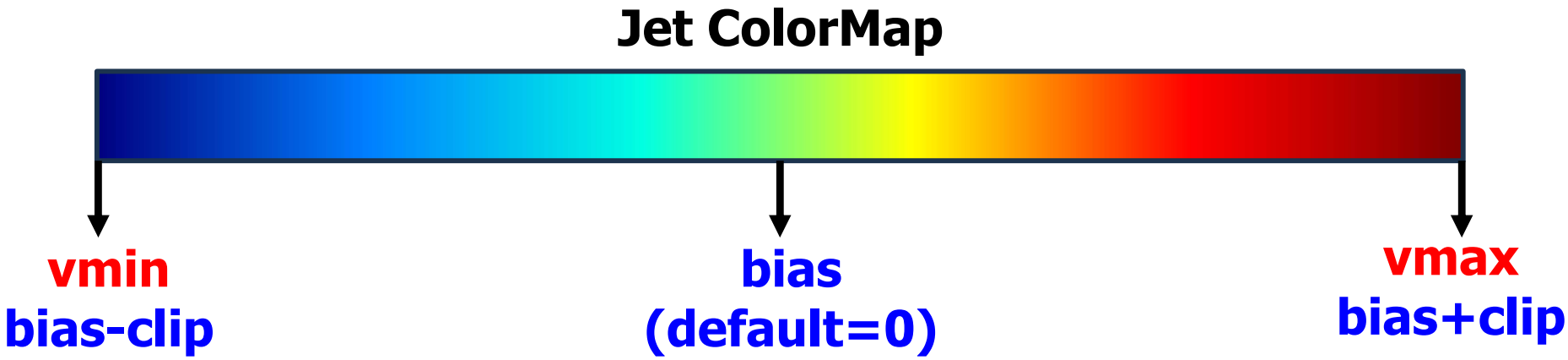
plotcol=5 gridcol=6



Index	0	1	2	3	4	5	6	7
bgcolor=w	white	yellow	cyan	green	magenta	red	blue	black
bgcolor=b	black	blue	red	magenta	green	cyan	yellow	white

Tips for M8R plotting

3) Clip/bias (m8r), vmin/vmax (Python) and caxis/CLim (Matlab)



allpos=y: ignore bias=, vmin=0, vmax=clip

Matlab: caxis([vmin, vmax]);
or: ax=gca; ax.CLim = [vmin, vmax];

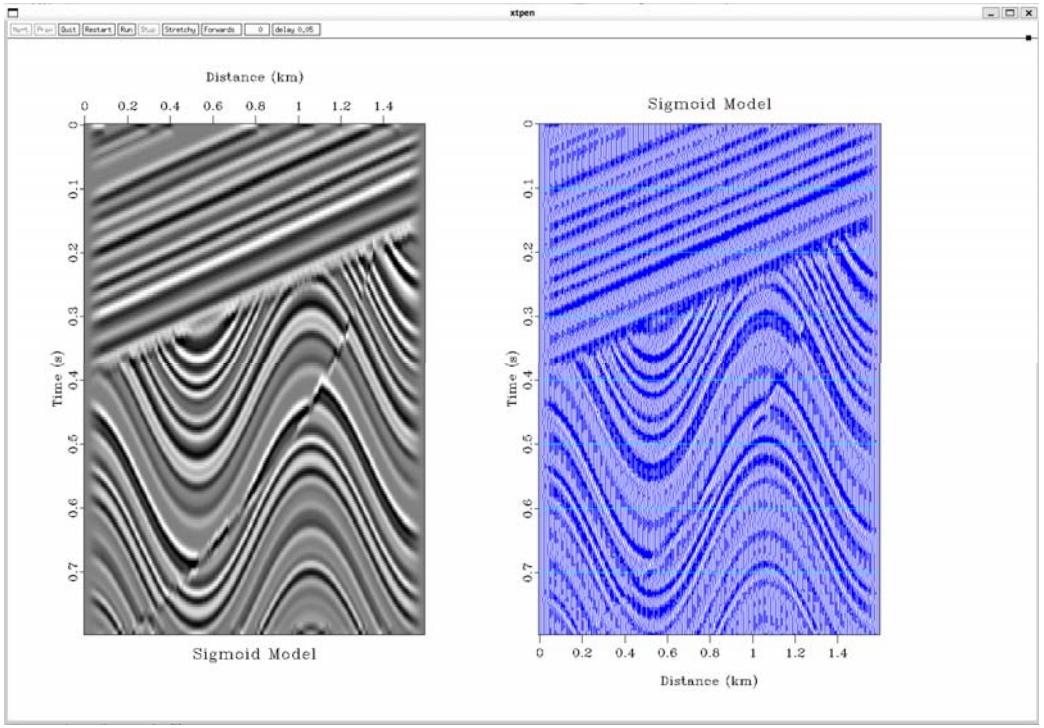
Tips for M8R plotting

4) Figure combinations

```
Plot('subfig1', 'model', 'grey')
Plot('subfig2', 'model', 'wiggle transp=y yreverse=y poly=y')

Result('combined','subfig1 subfig2','SideBySideAniso')
```

Combinations

- Movie
- Overlay
- SideBySideAniso
- OverUnderAniso
- SideBySideIso
- OverUnderIso
- TwoRows
- TwoColumns



Outline

- Motivations
- Prerequisites
- Figure preparation
  - Plotting with M8R
  - Plotting with Python
  - Plotting with Matlab/Octave
  - Plotting with GMT
- SEGTeX
- Alternative LaTeX template
- Simple Example

Plotting with Python: sfmatplotlib

Plotting with matplotlib

September 12, 2019 Programs 1 comment

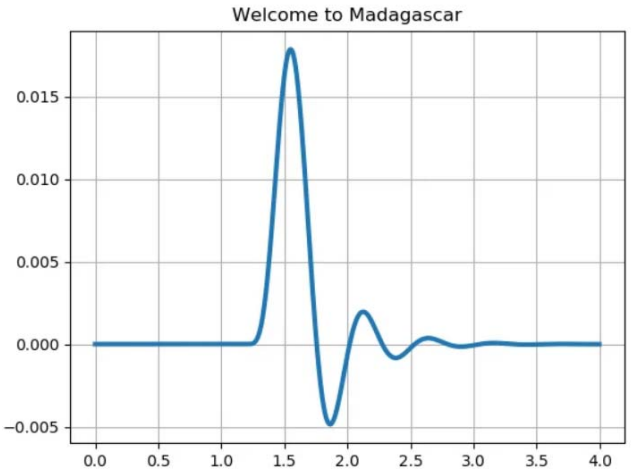
Matplotlib is a popular Python library for 2D plotting, which emulates (and exceeds) the plotting functionality of MATLAB.

At the Madagascar Working Workshop in 2018, a couple of new scripts were added to simplify plotting with Matplotlib in Madagascar.

- sfmatplotlib can be used to plot RSF files.You can use it as follows:

```
sfmatplotlib <matplotlib function> <plot options> [format=eps] < inp.rsf [ > out.eps]
```

Without the standard output, the figure is displayed on the screen.

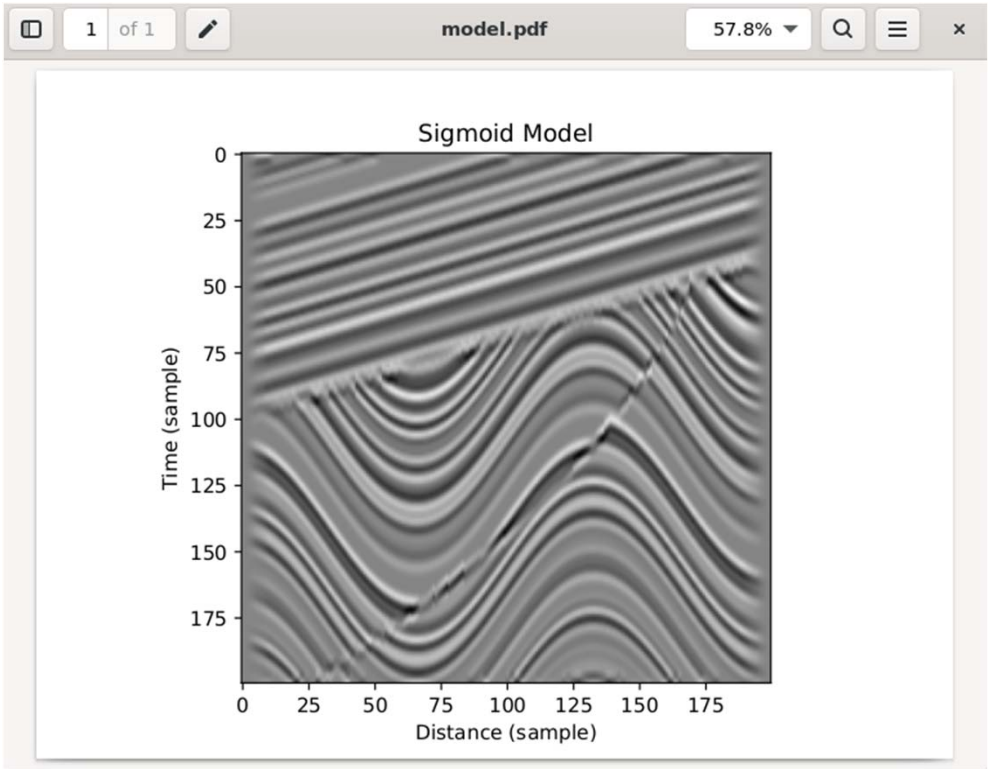


The figure above was generated with

```
sfspike n1=1000 k1=300 | sfbandpass fhi=2 phase=y| sfmatplotlib plot title="Welcome to Maagascar" grid=y linewidth=3 forma
t=png > figure.png
```

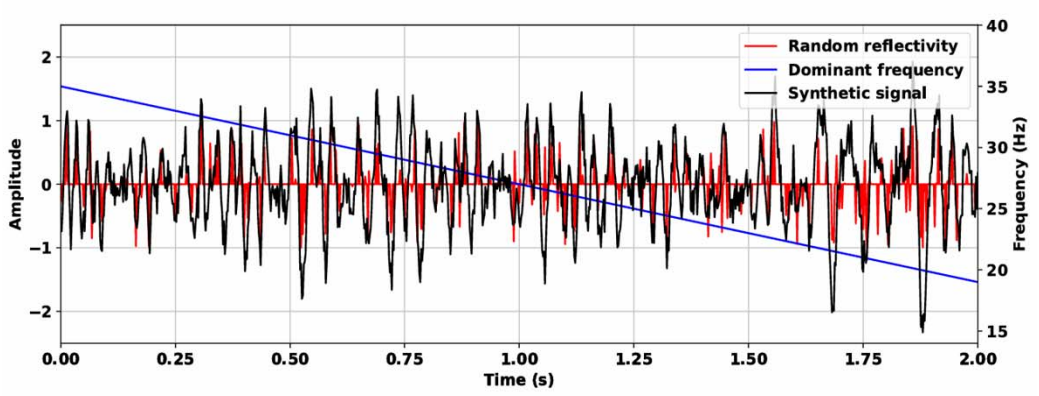
Plotting with Python: sfmatplotlib

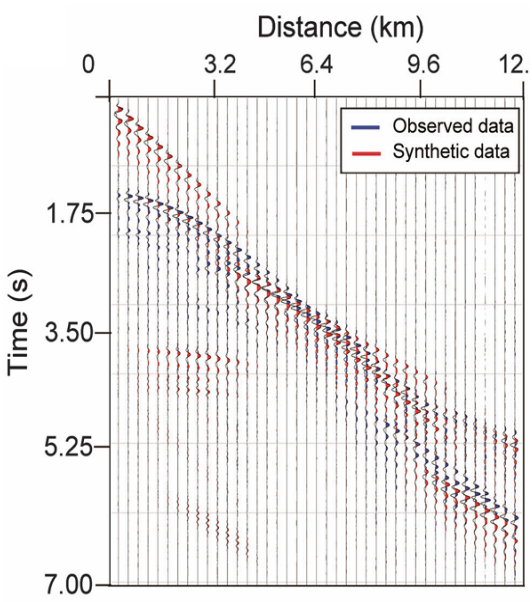
```
sfsigmoid n1=200 n2=200 d1=.004 d2=.008 \
    | sfsmooth rect1=3 diff1=1 | sfsmooth rect1=3 \
    | sfput label2=Distance > model.rsf
< model.rsf sftransp | sfmatplotlib imshow cmap=gray title="Sigmoid Model" xlabel="Distance
(sample)" ylabel="Time (sample)"  format=pdf > model.pdf
```

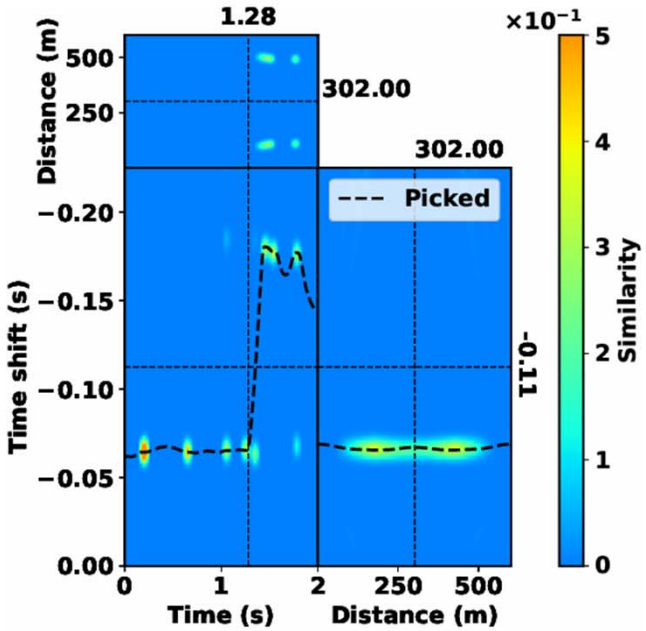


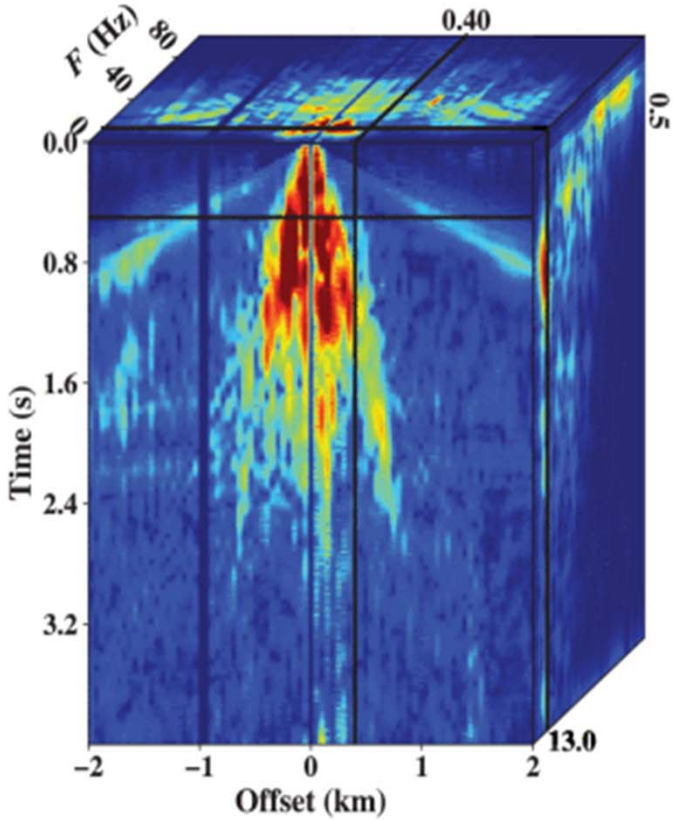
Plotting with Python: write your own script

Gallery









How to implement

1) **Deal with SConstruct files.**

   ➔ **Choose a command-line style.**

2) **Deal with M8R data (rsf data files).**

3) **Write your script.**

Plotting with Python: write your own script

1) Choose a command-line style:

(M8r style) Using stdin/stdout

```
< data.rsf python your_script.py arg1=val1 \
        arg2=val2 ... > Fig/figure.pdf
```

```
Result("figure", "data", """
    python your_script.py arg1=val1 arg2=val2 …
""", suffix=".pdf")
```

- **Output directory: Fig/**
- **ideal for use with shell pipes (|) and chaining multiple processing modules.**

Plotting with Python: write your own script

1) Choose a command-line style:

Ignore stdin or stdout

```
python your_script.py in=data.rsf \
out=Fig/figure.pdf ...
```

```
Result("figure", "data","""
    python your_script.py in=${SOURCES[0]} out=${TARGETS[0]} …
""", suffix=".pdf", stdin=0, stdout=-1)
```

- **stdout=0 : >/dev/null**

Plotting with Python: write your own script

2) Deal with M8r data file:

- ✓ **Use m8r to read rsf data.**

3) Write your script:

- ✓ **Use numpy to deal with data volume and calculation.**
- ✓ **Use matplotlib to visualize your results.**
- ✓ **(Optionally) use AI to assist your coding.**

```
from rsf.proj import *

# Sigmoid model
Flow('model',None,
     '''
      sigmoid n1=200 n2=200 d1=.004 d2=.008 |
      smooth rect1=3 diff1=1 | smooth rect1=3 |
      put label2=Distance title='"Sigmoid Model"'
     ''')

Result('figure', 'model', 'python script.py', suffix='.pdf')

End()
```

< model.rsf python script.py > Fig/figure.pdf

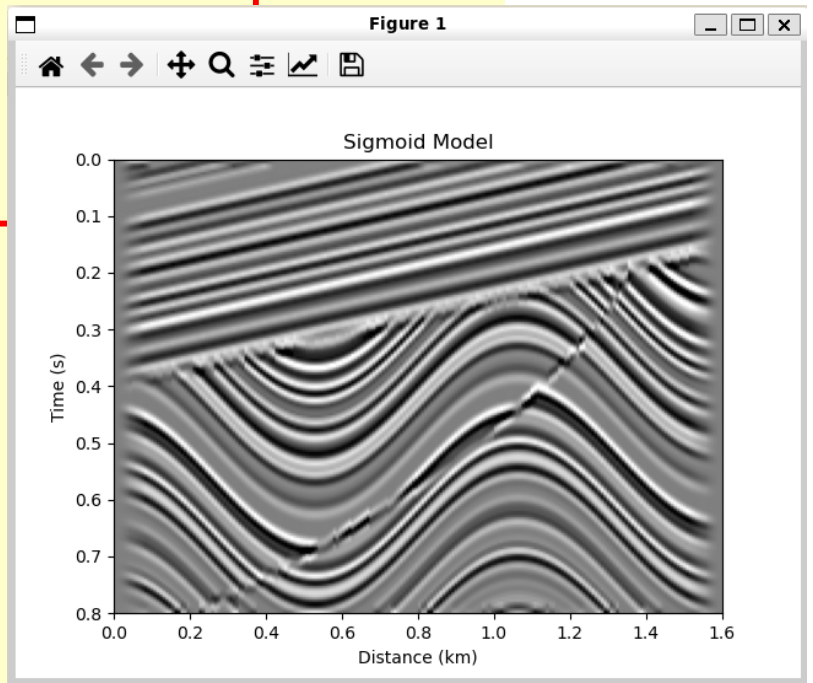
Plotting with Python: simple example

```
$ sfin model.rsf
model.rsf:
    in="/var/tmp/Desktop/school2025/plots/model.rsf@"
    esize=4 type=float form=native
    n1=200          d1=0.004        o1=0            label1="Time" unit1="s"
    n2=200          d2=0.008        o2=0            label2="Distance" unit2="km"
        40000 elements 160000 bytes
(main)
```

```python
import m8r, numpy, sys
import matplotlib.pyplot as plt

if __name__ == "__main__":
    # check stdin
    if sys.stdin.isatty():
        print("Usage: python greytest.py < data.rsf [> output.pdf]", file=sys.stderr)
        sys.exit(1)
    # read data from stdin
    inp = m8r.Input()
    # read data and axes
    n1, n2 = inp.int('n1'), inp.int('n2')
    d1, d2 = inp.float('d1'), inp.float('d2')
    o1, o2 = inp.float('o1'), inp.float('o2')
    l1, l2 = inp.string('label1'), inp.string('label2')
    u1, u2 = inp.string('unit1'), inp.string('unit2')
    title = inp.string('title')
    data = inp.read(datatype=numpy.float32).T
    # calculate clip
    clip = numpy.percentile(numpy.abs(data), 99)
    # display data
    plt.imshow(data, aspect='auto', cmap='gray',
               vmin=-clip, vmax=clip,
               extent=(o2, o2 + d2 * n2, o1 + d1 * n1, o1))
    plt.ylabel(f"{l1} ({u1})")
    plt.xlabel(f"{l2} ({u2})")
    if title: plt.title(title)
    # check stdout
    if sys.stdout.isatty(): plt.show()
    else: plt.savefig(sys.stdout.buffer, format='pdf')
    sys.exit(0)
```

Use m8r.Input() to read rsf data and header.

Always float32 for rsf float data.



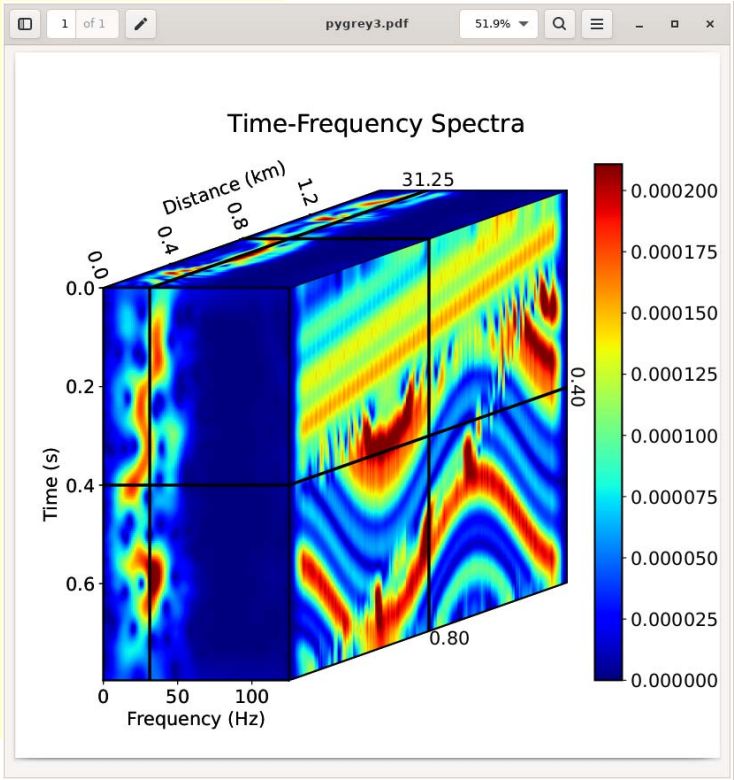
Plotting with Python: simple example

```
.....

# read header from stdin
inp = m8r.Input()
n1, n2, n3 = inp.int('n1'), inp.int('n2'),  inp.int('n3')
d1, d2, d3 = inp.float('d1'), inp.float('d2'), inp.float('d3')
o1, o2, o3 = inp.float('o1'), inp.float('o2'), inp.float('o3')
l1, l2, l3 = inp.string('label1'), inp.string('label2'), inp.string('label3')
u1, u2, u3 = inp.string('unit1'), inp.string('unit2'), inp.string('unit3')
title = inp.string('title')

# Read parameters from command line
pars = m8r.Par()
frame1 = pars.int('frame1', 0)
frame2 = pars.int('frame2', 0)
frame3 = pars.int('frame3', 0)
point1 = pars.float('point1', 0.8)
point2 = pars.float('point2', 0.4)
clip = pars.float('clip', None)
pclip = pars.float('pclip', 99.)
isflat = pars.bool('flat', True)
allpos = pars.bool('allpos', False)
colorbar = pars.bool('scalebar', False)
color = pars.string('color', 'gray')
title = pars.string('title', title)
.....
```

Use m8r.Input() to read parameters

pars.par_type(par_key, default)

```
69  Result('pygrey3',['ltft', grey3script],
70         '''
71         ${SOURCES[1]} frame1=100 frame2=25 frame3=100
72         point1=0.8 point2=0.4 clip=2e-4 flat=n scalebar=y
73         title="Time-Frequency Spectra" allpos=y color=jet
74         ''', suffix='.pdf')
```



Outline

- Motivations
- Prerequisites
- Figure preparation
  - Plotting with M8R
  - Plotting with Python
  - Plotting with Matlab/Octave
  - Plotting with GMT
- SEGTeX
- Alternative LaTeX template
- Simple Example

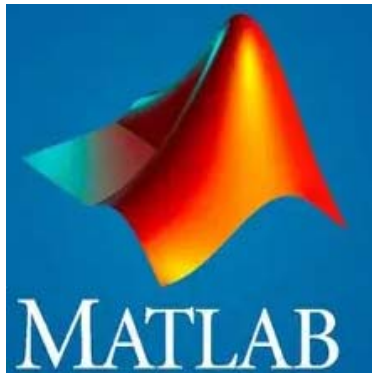
Plotting with Matlab: M8R API

a. **Use Madagascar Matlab api to read rsf data.**

- **Require proper installation.**

  Check by `ls $RSFROOT/lib/*.mexa64`

- **Matlab version > R2019a.**




```
from rsf.proj import *

# Sigmoid model
Flow('model',None,
     '''
      sigmoid n1=200 n2=200 d1=.004 d2=.008 |
      smooth rect1=3 diff1=1 | smooth rect1=3 |
      put label2=Distance title='"Sigmoid Model"'
     ''')

Result('matgrey','model',
       '''
        matlab -batch "matgrey('${SOURCES[0]}', '${TARGETS[0]}', 'Sigmoid Model')"
       ''', suffix='.pdf', stdin=0, stdout=-1)

End()
```

Plotting with Matlab: simple example (M8R API)

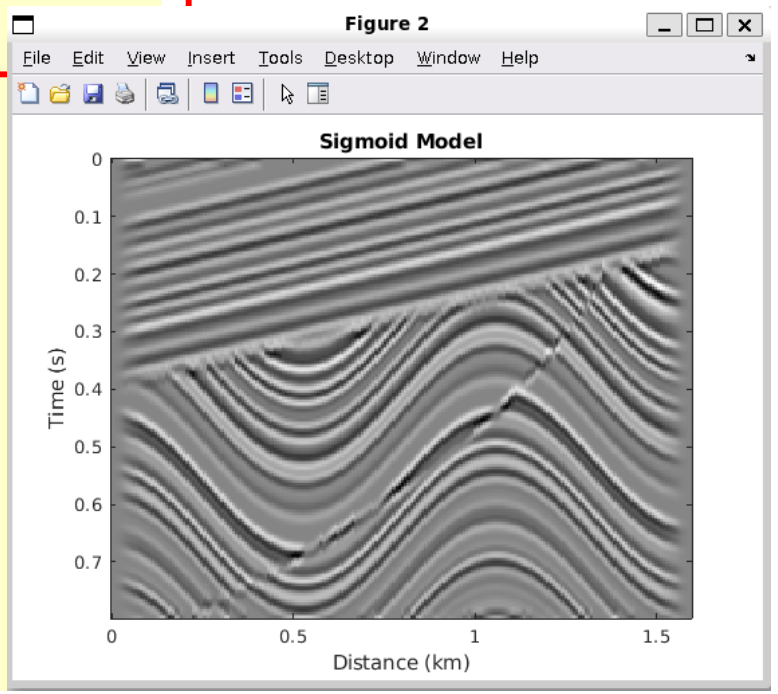
```
Result('matgrey','model',
       '''
        matlab -batch
"matgrey('${SOURCES[0]}', '${TARGETS[0]}',
'Sigmoid Model')"
       ''', suffix='.pdf', stdin=0,
stdout=-1)
```

scons matgrey.view

```
function matgrey(rsf_file, outfig, figtitle)
    % Require 2 or 3 input arguments
    narginchk(2,3);
    if isempty(outfig)
        error('The output filename "outfig" cannot be empty.')
    end
    if nargin < 3
        figtitle = '';
    end
    %--- 1. Read RSF header and data
    % Add the RSF library to MATLAB path
    addpath(fullfile(getenv('RSFROOT'), 'lib'));
    [shape, delta, origin, label, unit] = rsf_read_header(rsf_file);
    data = zeros(shape);
    rsf_read(data, rsf_file);
    time_axis = origin(1) + (0:shape(1)-1) * delta(1);
    spatial_axis = origin(2) + (0:shape(2)-1) * delta(2);
    %--- 2. Display the data in a grayscale figure
    hFig = figure('Color', 'w');
    imagesc(spatial_axis, time_axis, data);
    colormap gray;
    xlabel(sprintf('%s (%s)', label{2}, unit{2}));
    ylabel(sprintf('%s (%s)', label{1}, unit{1}));
    if ~isempty(figtitle)
        title(figtitle, 'Interpreter', 'none');
    end
    %--- 3. Save the figure as a PDF
    exportgraphics(hFig, outfig, 'ContentType', 'vector');
    fprintf('Saved figure to %s\n', outfig);
end
```

Use rsf_read to read rsf data.

Use rsf_read_header to read data header.



Plotting with Matlab/Octave

b. Read rsf data manually.

- **Write your own I/O module.**
- **Minimum requirements.**

GNU Octave

https://octave.org/

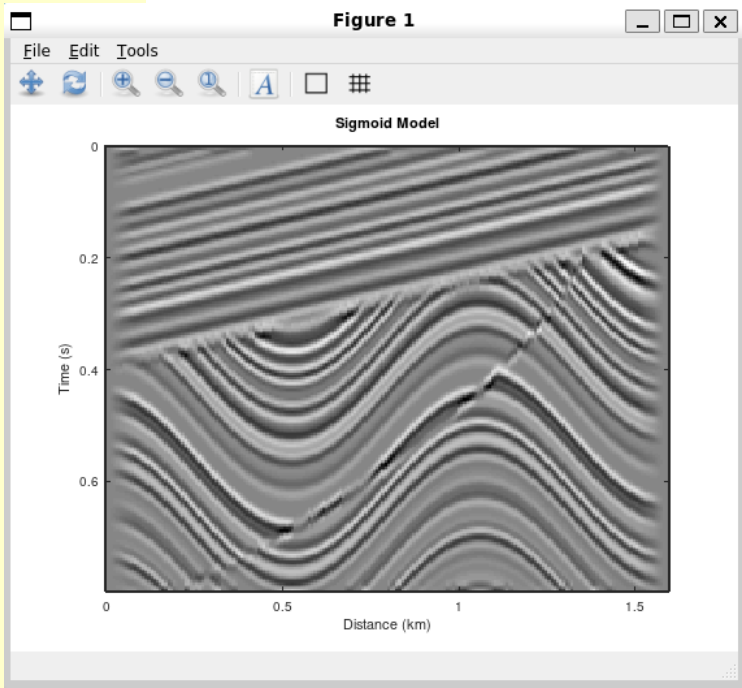
sudo apt update
sudo apt install octave

```
from rsf.proj import *
# Sigmoid model
Flow('model',None,
     '''
      sigmoid n1=200 n2=200 d1=.004 d2=.008 |
      smooth rect1=3 diff1=1 | smooth rect1=3 |
      put label2=Distance title='"Sigmoid Model"'
     ''')
# Matlab version
Result('matgrey','model',
       '''
        matlab -batch "matgrey('${SOURCES[0]}', '${TARGETS[0]}', 'Sigmoid Model')"
       ''', suffix='.pdf', stdin=0, stdout=-1)
# Octave version
Result('octgrey','model',
       '''
        octave --eval "matgrey('${SOURCES[0]}', '${TARGETS[0]}', 'Sigmoid Model'); exit;"
       ''', suffix='.pdf', stdin=0, stdout=-1)
End()
```

Plotting with Matlab/Octave: simple example

```
function matgrey(rsf_file, outfig, figtitle)
    narginchk(2,3); % Require 2 or 3 input arguments
    if isempty(outfig)
        error('The output filename "outfig" cannot be empty.');
    end
    if nargin < 3
        figtitle = '';
    end
    %--- 1. Read RSF header and data
    [data, header] = read_rsf(rsf_file);

    taxis = header.o1 + (0:header.n1-1) * header.d1;
    xaxis = header.o2 + (0:header.n2-1) * header.d2;
    %--- 2. Display the data in a grayscale figure
    hFig = figure('Color', 'w');
    imagesc(xaxis, taxis, data);
    colormap gray; axis tight;
    xlabel(sprintf('%s (%s)', header.label2, header.unit2));
    ylabel(sprintf('%s (%s)', header.label1, header.unit1));
    if ~isempty(figtitle)
        title(figtitle, 'Interpreter', 'none');
    end
    %--- 3. Save the figure as a PDF
    if exist('OCTAVE_VERSION','builtin') ~= 0
        print(hFig, outfig, '-dpdf', '-painters');
    else
        exportgraphics(hFig, outfig, 'ContentType', 'vector');
    end
    fprintf('Saved figure to %s\n', outfig);
end
```

Use read_rsf to read data and header.



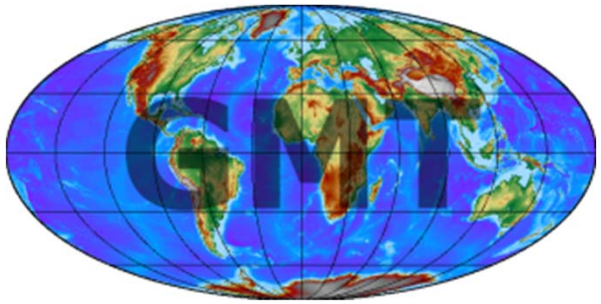
Outline

- Motivations
- Prerequisites
- Figure preparation
  - Plotting with M8R
  - Plotting with Python
  - Plotting with Matlab/Octave
  - Plotting with GMT
- SEGTeX
- Alternative LaTeX template
- Simple Example

Plotting with GMT: GMT example 02

sudo apt update
sudo apt install gmt

```
from rsf.proj import *

Result('gmt1', 'gmt-1.sh', 'bash ${SOURCES[0]}',
suffix='.pdf', stdin=0, stdout=-1)

End()
```

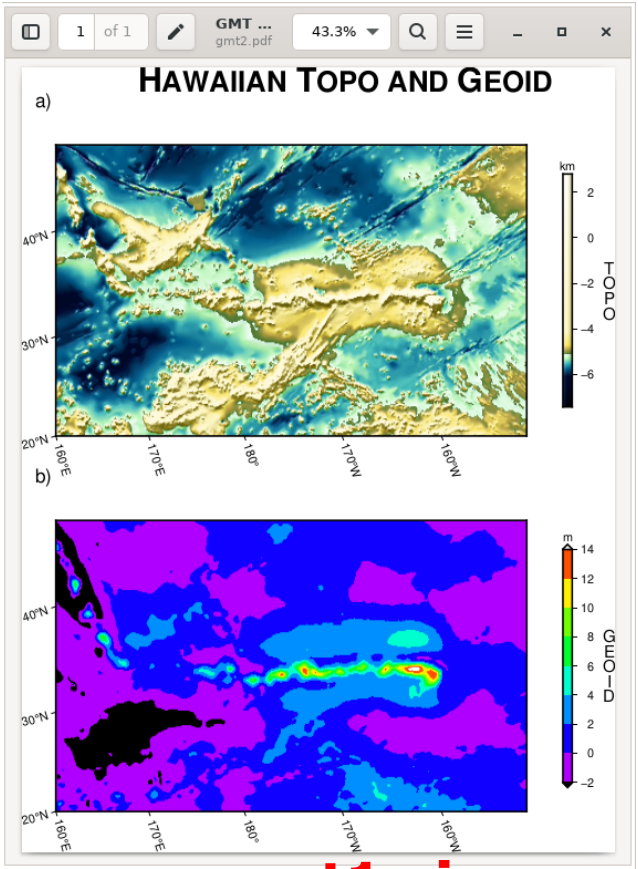


(2) Image presentations

As our second example we will demonstrate how to make color images from gridded data sets (again, we will defer the actual making of grid files to later examples). We have prepared two 2-D grid files of bathymetry and Geosat geoid heights from global grids and will put the two images on the same page. The region of interest is the Hawaiian Islands, and due to the oblique trend of the island chain we prefer to rotate our geographical data sets using an oblique Mercator projection defined by the hotspot pole at (68W, 69N). We choose the point (190, 25.5) to be the center of our projection (e.g., the local origin), and we want to image a rectangular region defined by the longitudes and latitudes of the lower left and upper right corner of region. In our case we choose (160, 20) and (220, 30) as the corners. We twice use grdimage to make the illustration:

```
#!/usr/bin/env bash
#   GMT EXAMPLE 02
#
# Purpose:  Make two color images based gridded data
# GMT modules:  set, grd2cpt, grdimage, makecpt, colorbar, subplot
#
gmt begin gmt1 02
  gmt set MAP_ANNOT_OBLIQUE separate
  gmt subplot begin 2x1 -A+JTL -Fs16c/9c -M0 -R160/20/220/30+r -JOc190/25.5/292/69/16c -B10 -T"H@#awaiian@# T@#opo and @#G@#eoid@#"
    gmt subplot set 0,0 -Ce3c
    gmt grd2cpt @HI_topo_02.nc -Crelief -Z
    gmt grdimage @HI_topo_02.nc -I+a0
    gmt colorbar -DJRM+o1c/0+mc -I0.3 -Bx2+lTOPO -By+lkm

    gmt subplot set 1,0 -Ce3c
    gmt makecpt -Crainbow -T-2/14/2
    gmt grdimage @HI_geoid_02.nc
    gmt colorbar -DJRM+o1c/0+e+mc -Bx2+lGEOID -By+lm
  gmt subplot end
gmt end show
```

Fig/gmt1



https://docs.generic-mapping-tools.org/6.6/

scons gmt1.view

Plotting with GMT: GMT example 13

```
from rsf.proj import *

Result('gmt2', 'gmt-2.sh', 'bash ${SOURCES[0]}',
suffix='.pdf', stdin=0, stdout=-1)

End()
```

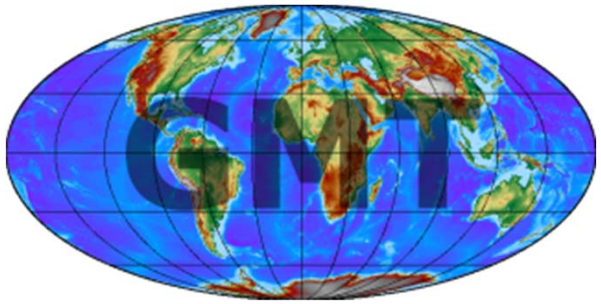


Illustration Gallery / (13) Plotting of vector fields — Edit on GitHub

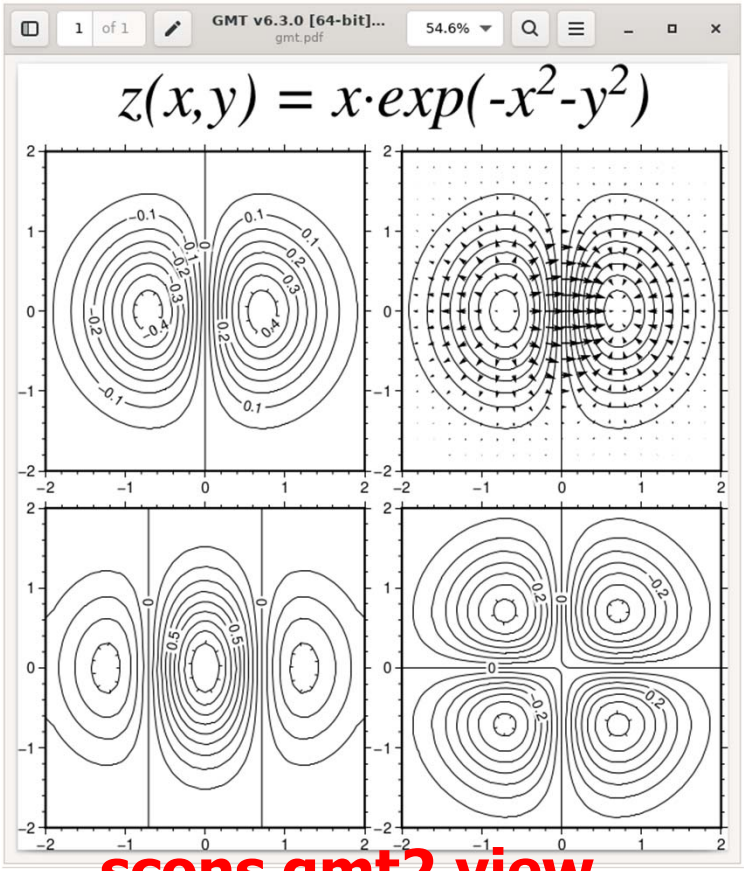
(13) Plotting of vector fields

In many areas, such as fluid dynamics and elasticity, it is desirable to plot vector fields of various kinds. GMT provides a way to illustrate 2-component vector fields using the grdvector utility. The two components of the field (Cartesian or polar components) are stored in separate grid files. In this example we use grdmath to generate a surface $z(x,y) = x \cdot \exp(-x^2 - y^2)$ and to calculate ∇z by returning the *x*- and *y*-derivatives separately. We superpose the gradient vector field and the surface *z* and also plot the components of the gradient in separate windows. A text call to place a header finishes the plot

```
#!/usr/bin/env bash
#   GMT EXAMPLE 13
#
# Purpose:  Illustrate vectors and contouring
# GMT modules:  set, grdmath, grdcontour, grdvector, subplot
# Unix progs: rm
#
gmt begin ex13
  gmt set FONT_HEADING 40p,Times-Italic
  gmt grdmath -R-2/2/-2/2 -I0.1 X Y R2 NEG EXP X MUL = z.nc
  gmt grdmath z.nc DDX = dzdx.nc
  gmt grdmath z.nc DDY = dzdy.nc
  gmt subplot begin 2x2 -M0.1c -Ff15c -BWSne -T"z(x,y) = x@~\327@~exp(-x@+2@+-y@+2@+)"
    gmt grdcontour z.nc -C0.05 -A0.1 -Gd5c -S4 -T+d8p/2p -c0,0
    gmt grdcontour z.nc -C0.05 -Gd5c -S4 -c0,1
    gmt grdvector dzdx.nc dzdy.nc -I0.2 -Q0.25c+e+n0.25i/0+h0.5 -Gblack -W1p -S2c
    gmt grdcontour dzdx.nc -C0.10 -A0.5 -Gd5c -S4 -T+d8p/2p -c1,0
    gmt grdcontour dzdy.nc -C0.05 -A0.2 -Gd5c -S4 -T+d8p/2p -c1,1
  gmt subplot end
gmt end show
rm -f z.nc dzdx.nc dzdy.nc
```

Fig/gmt2

https://docs.generic-mapping-tools.org/6.6/

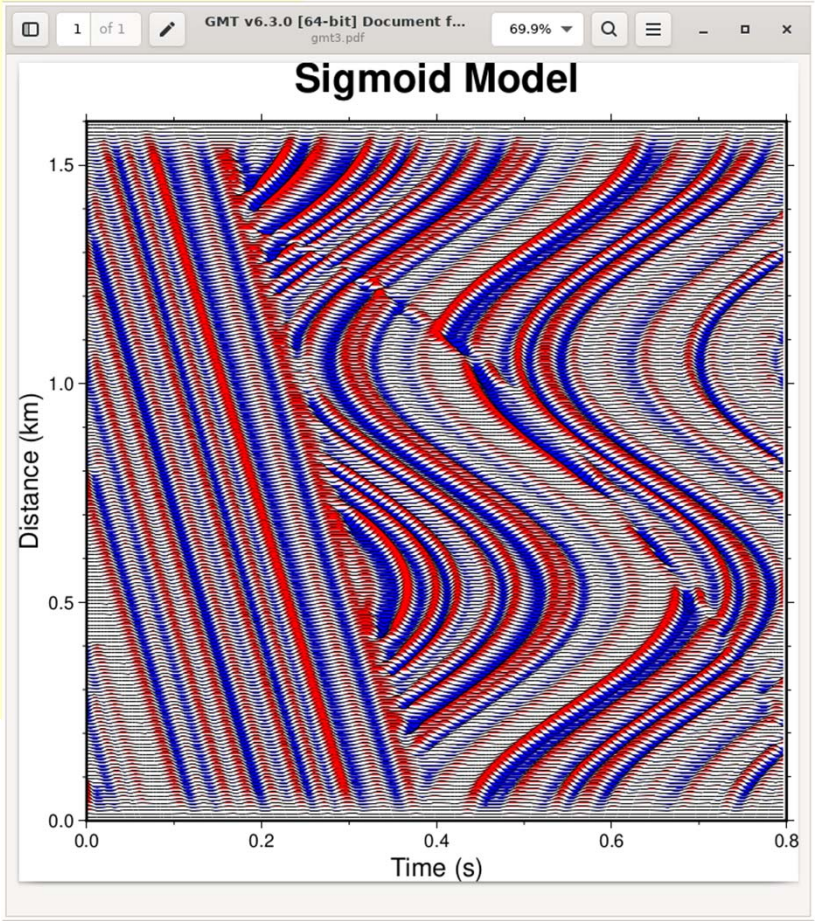


scons gmt2.view

Plotting with GMT: try wiggle plot

```
# GMT Wiggle

## Build T-X Grid
Flow('model1','model','dd form=ascii line=1 --
out=${TARGETS[0]}', stdout=0)
Flow('tgrid','model','math output="x1" | dd form=ascii
line=1 --out=${TARGETS[0]}', stdout=0)
Flow('xgrid','model','math output="x2" | dd form=ascii
line=1 --out=${TARGETS[0]}', stdout=0)
Flow('gmtmodel','tgrid xgrid model1',
    '''
     paste ${SOURCES[0:3]}
    ''')
## GMT One-line Commands
Result('gmt3', 'gmtmodel',
       '''
        gmt wiggle -R0/0.8/0/1.6 -JX5i/5i
        -BWSne+t"Sigmoid Model" -B -Z0.1i
        -Gred+p -Gblue+n -Wdefault -gy0.001
        -By+l"Distance (km)" -Bx+l"Time (s)"
        -pdf Fig/gmt3
       ''', suffix='.pdf',  stdout=-1)
```

T	X	Data
0	0	d(0,0)
4e-3	0	d(4e-3,0)
…	…	…



scons gmt3.view

Make it a try

1) Choose your favorite language.

2) Edit the SConstruct and the corresponding script provided before.

3) [Optional] Ask LLM (Large Language Model) for help.

4) scons your_figure.view

Make it a try: LLM coding

For example, let's ask the Copilot to generate a wiggle plot:

Here is a python script used to generate a grey-scale image. Please use the framework and make it generate a wiggle plot instead: import m8r…… (dump the previous code here)

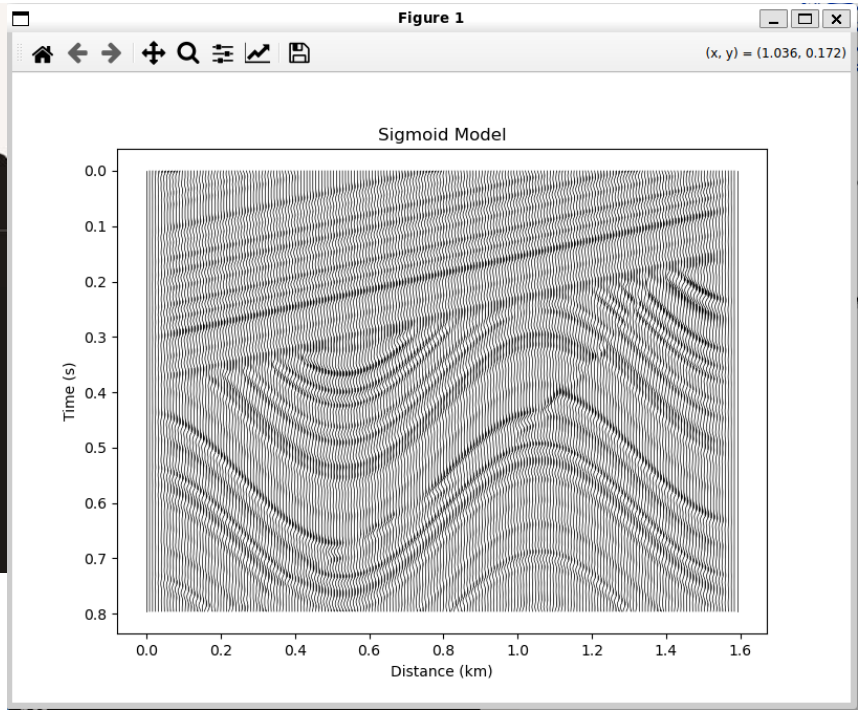
And the Copilot replies:

Here's a drop-in replacement of your script that reads the same RSF data but renders a classic seismic-style wiggle plot instead of a grey-scale image. You can adjust the `scale_factor` or `fill_positive` flags to taste.

Python 复制

```python
import m8r, numpy, sys
import matplotlib.pyplot as plt

def wiggle_plot(data, time, o2, d2, scale_factor=1.0,
                color='black', linewidth=0.5, fill_positive=True):
    """
    Draws a wiggle plot for 2D array `data` of shape (n_samples, n_traces).
    time : 1D array of length n_samples
    o2, d2 : origin and sampling interval on the horizontal axis
    scale_factor : multiplies amplitudes before plotting
    """
```

……



Make it a try: LLM coding

Avoid directly asking AI like this:

Please write a python script that read rsf data from stdin and then plot its grey-scale image. Save the image as a pdf to stdout.

And the Copilot replies:

Here's a self-contained Python script that reads an RSF volume from stdin, plots it in greyscale with Matplotlib, and writes the PDF image to stdout:

Python 复制

```
#!/usr/bin/env python3
import sys
import numpy as np
import matplotlib.pyplot as plt
from rsf.api import rsf_input

def main():
    # Read RSF from stdin
    a = rsf_input()              # by default reads from sys.stdin
    n1 = a.getint('n1')          # number of samples in fast axis
    n2 = a.getint('n2', 1)       # number of samples in slow axis (default 1

    # Load the full volume into a 2D numpy array
    data = a.read()              # shape: (n2, n1) or (n1,) if 1D

    # If 1D, force into a 2D for imshow
    if data.ndim == 1:
        data = data[np.newaxis, :]
```

.....

When there isn't precise context, AI often won't ask you for additional information and instead tends to outright fabricate things that don't exist.

Outline

- Motivations
- Prerequisites
- Figure preparation
  - Plotting with M8R
  - Plotting with Python
  - Plotting with Matlab/Octave
  - Plotting with GMT
- **SEGTeX**
- Alternative LaTeX template
- Simple Example

Implement steps

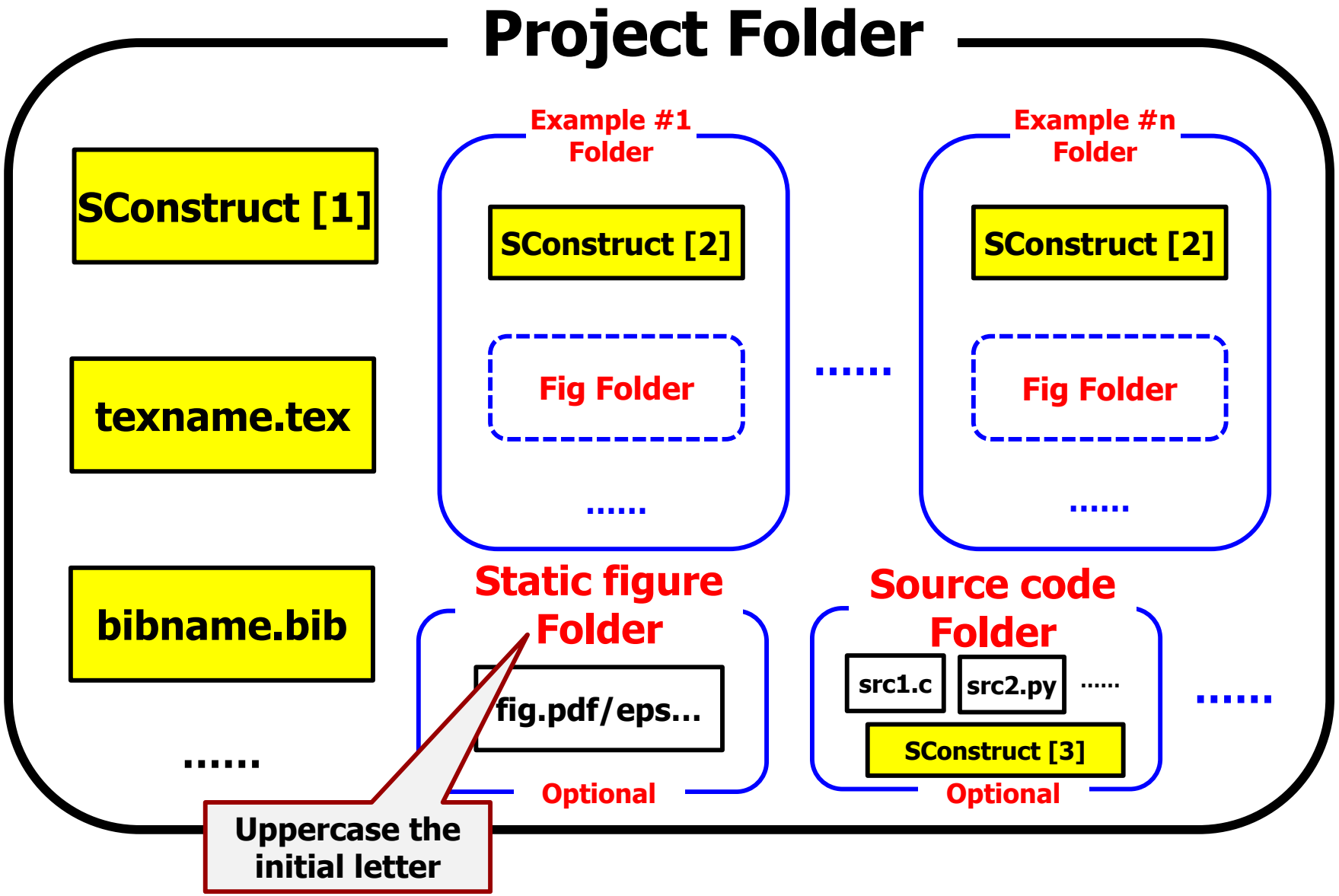
1) Ideas

2) Implement ideas in Madagascar and lock Figures (scons fig.lock)

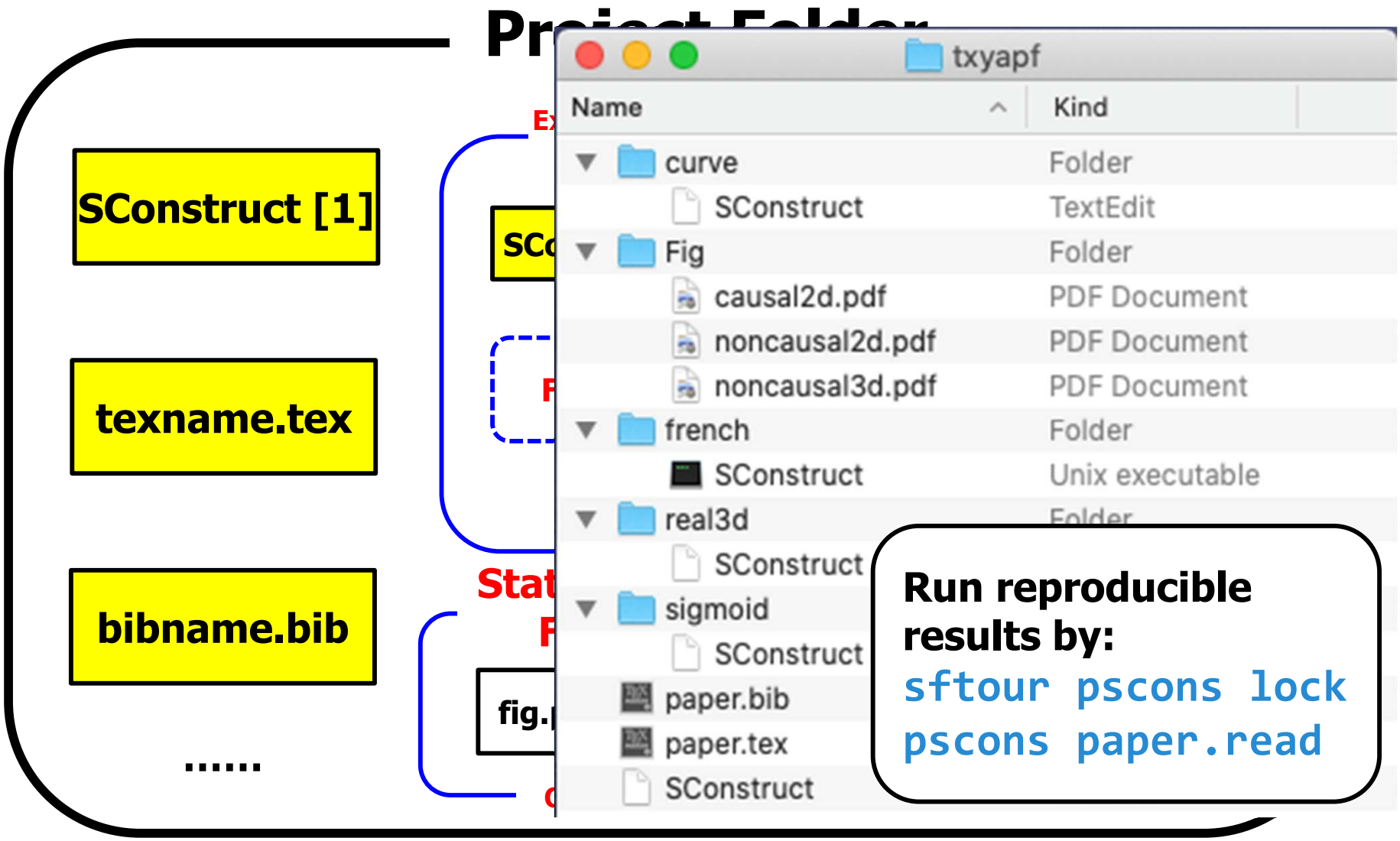
3) Write a paper text by following SEGTeX rules

4) Insert Figures into paper by following SEGTeX rules

Rules for project structure



Rules for project structure



Look inside example: $RSFSRC/book/jlu/txsopf

Rules for the tex file (structure)

Similar to common LaTeX format, but with a few differences

LaTeX tex file

Preamble

\documentclass[options]{class}

\usepackage[options]{package}

Custom Definitions
\newcommand,...

\begin{document}
Document Body
Bibliography
\end{document}

Madagascar tex file

Document Body
Bibliography

Rules for SConstruct [1]

- **Always starts with:** `from rsf.tex import *`
- **Configure your paper by:**

```
Paper('pdfname','texname',
      lclass='geophysics',
      options='manuscript,revised',
      use='''amsmath empheq graphicx
      float threeparttable multirow
      algorithm [caption=false]subfig
      '''.split(),
      include='')
```

texname.tex

Document class

options

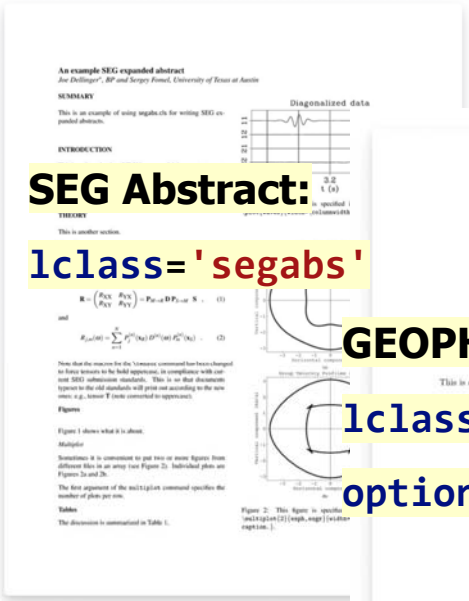
Used packages
\usepackage

Custom definitions

\usepackage[caption=false]{subfig}

- **Ends with** `End()`
- **Specially, if paper name is "paper.tex", you can directly use** `End()` **similar to** `Paper()`.

Commonly-used document class in SEGTeX



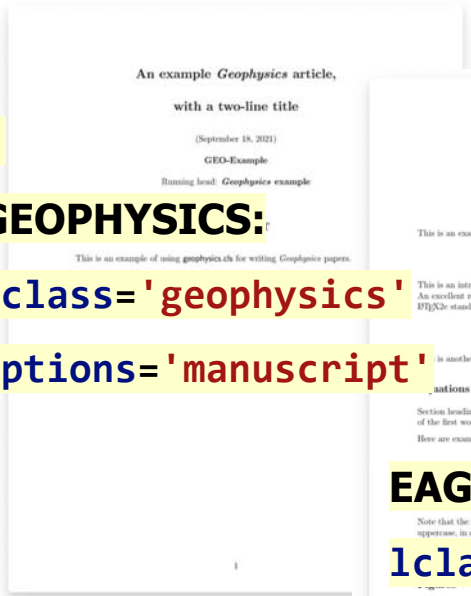
SEG Abstract:

```
lclass='segabs'
```

Template for SEG abstract

Template for writing SEG abstract.

Sergey Fomel

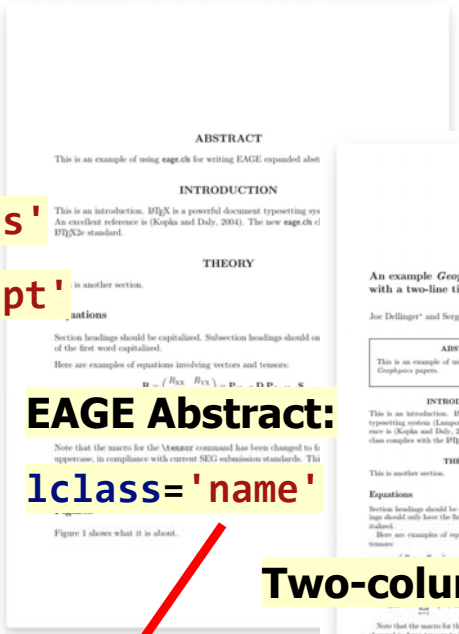


GEOPHYSICS:

```
lclass='geophysics'

options='manuscript'
```

Template for GEOPHYSICS

GEOPHYSICS, published by the Society of Geophysicists since 1936, is an archival jo

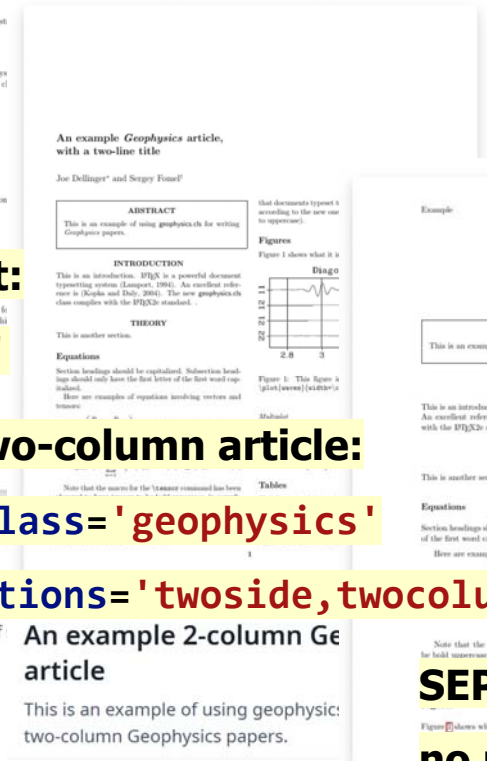


EAGE Abstract:

```
lclass='name'
```

An example EAGE ex
abstract

SEGTeX is a LaTeX package for g
publications. This template consists of

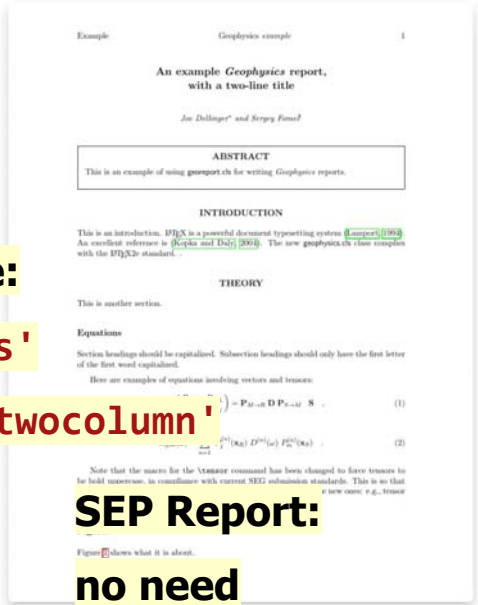


Two-column article:

```
lclass='geophysics'

options='twoside,twocolumn'
```

An example 2-column Ge
article

This is an example of using geophysics
two-column Geophysics papers.



SEP Report:

no need

lclass and options

An ex
This is an example of using geopreport.cls for writing Geophysics reports.

SEGTeX

Geophysical Prospecting:

```
lclass='geophysics',

options='manuscript,a4paper'
```

Need edit name.cls

See template "texmf/tex/latex/cwp/adam2009.cls"

Rules for SConstruct [2]

- Always starts with: `from rsf.proj import *`, and ends with `End()`.
- Use `Fetch()` to fetch open datasets in Madagascar server:

```
Fetch('Txx.HH', 'septour')
```

- Use `Flow()` to handle processing flow:

```
Flow('windowed', 'Txx.HH',
     '''
      window n2=10 min1=0.4 max1=0.8
     ''')
```

- Use `Result()` or `Plot()` to plot results:

```
Result('wiggle', 'windowed',
     '''
      wiggle transp=y poly=y yreverse=y pclip=100 nc=100
allpos=n
     ''')
```

Example of SConstruct [2]

```
from rsf.proj import *


Fetch('Txx.HH','septour')

Result('wiggle0','Txx.HH','wiggle')

Flow('windowed','Txx.HH','window n2=10 min1=0.4 max1=0.8')

plotpar = '''
      transp=y poly=y yreverse=y pclip=100 nc=100 allpos=n'''

for plot in ('wiggle','contour','grey'):
      Result(plot,'windowed',plot + plotpar)


End()
```

Python tricks work here.

Rules for SConstruct [3]

- **Always starts with: `from rsf.proj import *`, and ends with `End()`.**
- **[Optional] Use `Program()` to compile source codes (C/C++):**

```
Program(target='Mprogram.exe',
        source=['Mprogram.c', 'other.c'])
```

- **[Optional] Scripts written in interpreted languages (Python, GMT, Shell, …) do not require compilation. However, we can use a shebang (#!) to simplify commands. Put `#! /usr/bin/env python` at the very top of a python script file:**

`python script.py` → `./script.exe`

```
Flow('script.exe', 'script.py', '''
     echo "#! /usr/bin/env python" > ${TARGETS[0]} &&
     cat ${SOURCES[0]} >> ${TARGETS[0]} &&
     chmod +x ${TARGETS[0]}
    ''', rsfflow=0, stdin=0, stdout=-1)
```

Example of Geophysics paper

1) Title page and abstract

```
\title{paper title}

\address{\footnotemark[1] address1 \footnotemark[2]
address2 \footnotemark[3] address3}
\author{Name1\footnotemark[1], Name2\footnotemark[2], and
Name3\footnotemark[3]}
\footer{GEO-2020-XXXX}
\lefthead{Surname1 et al.}
\righthead{short title}
\maketitle

\begin{abstract}
Write paper abstract here …
\end{abstract}
```

Example of Geophysics paper

2) Sections, citations and equations

```
\section{section name, e.g., Introduction}
The paper starts from old idea \cite[]{refname1}.
\cite{refname2} also improve it. …

\section{section name, e.g., Theory}
  \subsection{subsection name}

The theory is shown as follows …
   \begin{equation}
     \label{eq:eq1}
          \mathbf{C} = \frac {\mathbf{A}}{b} \;,
    \end{equation}

Refer to equation~\ref{eq:eq1}.
```

(Refname1, year)

Refname2 (year)

Write equation label here

Refer to the equation by its label

Example of Geophysics paper

3) Figures and their references

```
\section{section name, e.g., Synthetic examples}
The paper has several synthetic tests.
\inputdir{example#1 folder name}
    \multiplot{2}{fig1,fig2}{width=0.5\textwidth}{figure1
name (a) and figure2 name (b).}

One can also refer to different figures, e.g.,
Figure~\ref{fig:fig1,fig2}a ...

\section{section name, e.g., Field data tests}
The method is also used to deal with field data
(Figure~\ref{fig:fig3}).
\inputdir{example#2 folder name}
    \plot{fig3}{width=0.75\textwidth}{figure3 name.}
```

Declare figure folder first

Figure 1a

Example of Geophysics paper

3) Ending and bibliography

```
\section{Conclusion}
Write conclusion here…

\section{Acknowledgments}
Should thank anyone for useful help…

\appendix
\section{Appendix: name}
Write the content of appendix here…

\bibliographystyle{seg}
\bibliography{bibname}
```

Rules for revision

- **Revised mode:** `options='manuscript, revised'`
- **Use** `\new{New contents}` **and** `\old{Old contents}` **to indicate "revised contents" and "~~replaced contents~~" corresponding to SConstruct[1] "options='revised' "**

Revised mode on

You can use \new and \old commands to mark the changes in the revised version.

For example, the revised text is *revised text* and the replaced text is ~~replaced text~~.

Revised mode off

You can use \new and \old commands to mark the changes in the revised version.

For example, the revised text is revised text and the replaced text is .

Rules for revision

- **Equations, citations, etc. don't work inside \new and \old.**

However, you can enclose citations in \mbox and equations in \parbox or minipage environment:

```
\old{wrong citation \cite{wrong}}
\new{revised citation \cite{revised}}
```

```
\old{wrong citation \mbox{\cite{wrong}}}
\new{revised citation \mbox{\cite{revised}}}
```

Rules for revision

- **Equations, citations, etc. don't work inside \new and \old.**

However, you can enclose citations in \mbox and equations in \parbox or minipage environment:

```
\old{wrong equation
\begin{equation}
2*2 = 5
\end{equation}}
```

```
\new{revised equation \\
\begin{equation}
2*2 = 4
\end{equation}}
```

```
\old{wrong equation \\
\begin{minipage}{\textwidth}
\begin{equation}
2*2 = 5
\end{equation}\end{minipage}}
```

```
\new{revised equation \\
\parbox{\textwidth}{
\begin{equation}
2*2 = 4
\end{equation}}}
```

Example of bibliography file

```
@Article{Abma05,
  author  =  {R[] Abma and N[] Kabir},
  title   =  {Comparisons of interpolation methods},
  journal =  {The Leading Edge},
  year    =  2005,
  volume  =  24,
  pages   =  {984-989}
}

@Article{Naghizadeh09,
  author  =  {M[ostafa] Naghizadeh and M[auricio] D[] Sacchi},
  title   =  {f-x adaptive seismic-trace interpolation},
  journal =  {Geophysics},
  year    =  2009,
  volume  =  74,
  pages   =  {V9-V16}
}
```

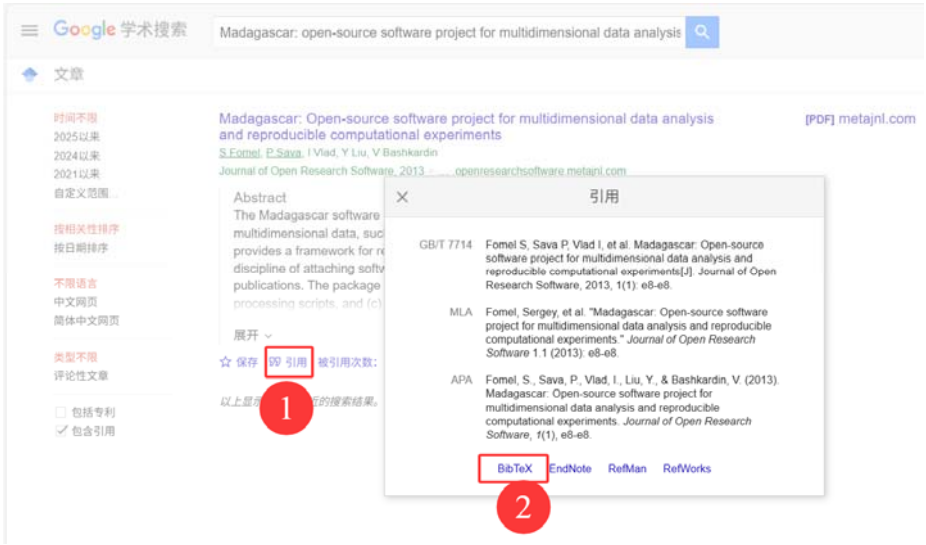
Example of bibliography file

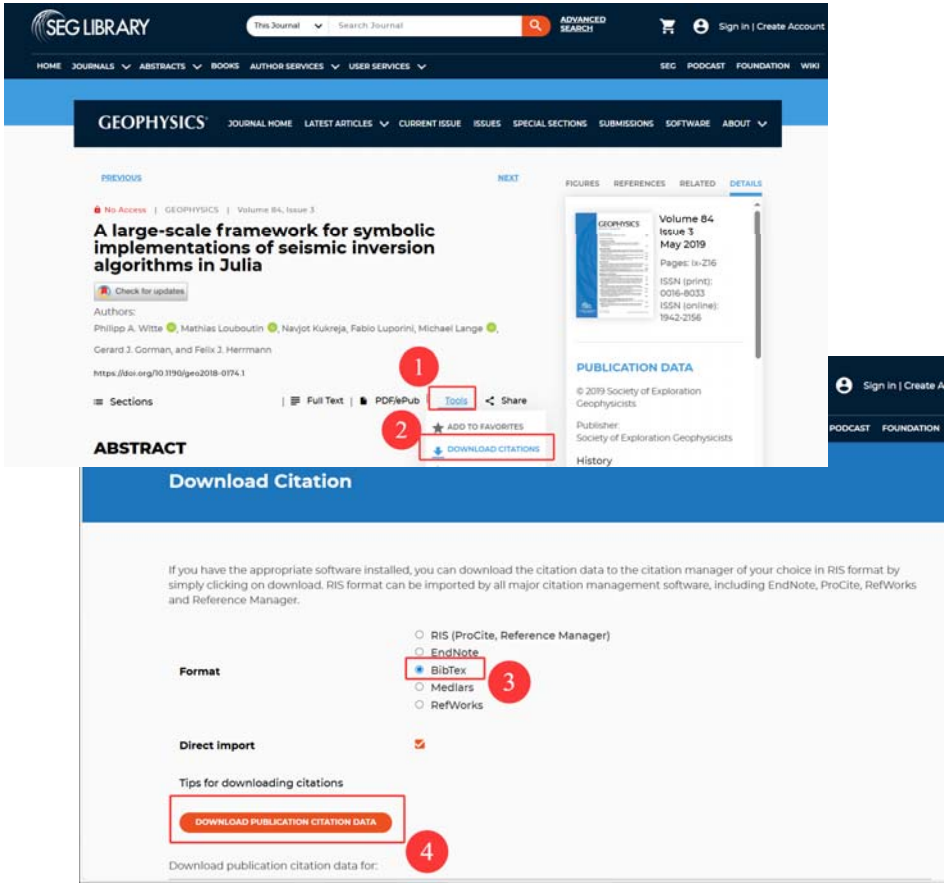
```
@book{Claerbout92,
    author    =  {J[on] F[] Claerbout},
    publisher =  {Blackwell Scientific Publications},
    title     =  {{Earth Soundings Analysis: Processing Versus
Inversion}},
    year      =  {1992}
}

@inproceedings{Curry04,
  author    =  {W[illiam] Curry},
  title     =  {Interpolation with multi-shifted-scale
prediction-error filters},
  booktitle =  {74th Annual International Meeting},
  year      =  {2004},
  publisher =  {SEG, Expanded Abstracts},
  pages     =  {2005-2008}
}
```

Check "texmf/bibtex/bib/seg/SEG.bib" before you start to type the references.

Download citations

It is also very convenient to download bibtex citations, for example:





Tips for Madagascar plot modules

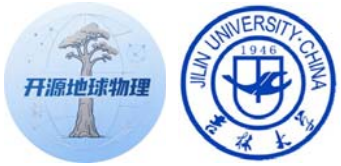
1) Label parameter:

- http://ahay.org/blog/2007/10/27/how-do-i-change-fonts-in-vplot-graphics/
- http://ahay.org/blog/2008/03/26/a-journal-requires-tick-labels-on-my-plots-to-be-oriented-vertically-and-aligned-on-the-left-how-do-i-achieve-that/
- http://ahay.org/blog/2008/10/15/a-journal-requires-a-particular-font-e-g-arial-or-helvetica-in-figure-labels-how-do-i-achieve-that/

2) Figure color:

- http://ahay.org/blog/2005/03/28/color-schemes/
- http://ahay.org/blog/2013/03/19/color-palettes/
- http://ahay.org/blog/2013/08/23/cube-helix-color-palette/
- http://ahay.org/blog/2014/05/15/light-bartlein-color-palette/
- http://ahay.org/blog/2014/10/18/tutorial-on-colormaps/
- http://ahay.org/blog/2015/07/12/more-colormaps/

Tips for Madagascar plot modules



3) Control figure color and resolution (SConstruct [1]):

```
Paper('name',lclass='geophysics', options='manuscript',
      use='listings',hires='fig1 fig2 fig3',color='fig4 fig5 fig6')
```

4) Insert only grey figures in LaTeX (SConstruct [1]):

```
from rsf.tex import *
import os
os.environ['PSTEXPENOPTS'] = 'color=n'

End(use='amsmath,hyperref',options='manuscript')
```

Tips for SCons scripts

- **You can write multiple `Paper()` with different options in SConstruct[1] to build different papers.**

- **Use pscons or scons -j # for parallel building.**

Tips for SCons scripts

- **Most parameters in the SCons file are specified as strings.**

Therefore, Python's string manipulation capabilities are extremely useful when writing SCons scripts.

```
plotpar = '''
      transp=y poly=y yreverse=y pclip=100 nc=100 allpos=n'''
for plot in ('wiggle','contour','grey'):
      Result(plot,'windowed',plot + plotpar)
```

```
f'{plot} {plotpar}'
'%s %s'%(plot, plotpar)
```

Outline

- Motivations
- Prerequisites
- Figure preparation
  - Plotting with M8R
  - Plotting with Python
  - Plotting with Matlab/Octave
  - Plotting with GMT
- SEGTeX
- Alternative LaTeX template
- Simple Example

How to use alternative templates

1) Download a LaTeX template file and corresponding class file.

2) Open the template file. According to the preamble contents in template .tex file, configure `Paper()` in SConstruct [1] following its rules.

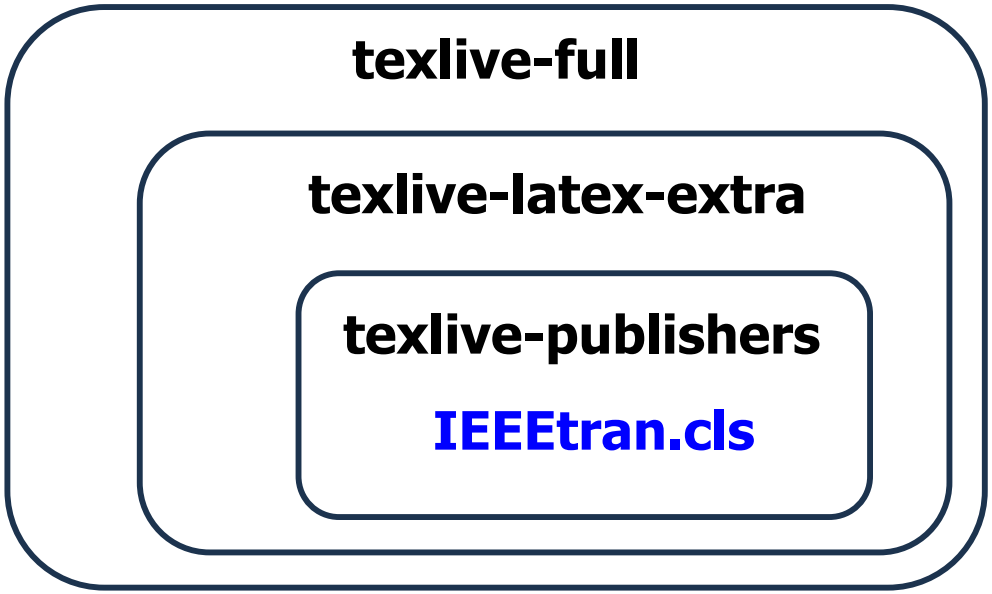
3) Remove all preamble contents (before \begin{document}) including \begin{document} and \end{document}.

4) scons paper.read.

1 Download template file and class file

a. The class file **IEEEtran.cls** is included in TeXLive packages: texlive-publishers, texlive-latex-extra and texlive-full.

```
sudo apt update
sudo apt install texlive-full
```

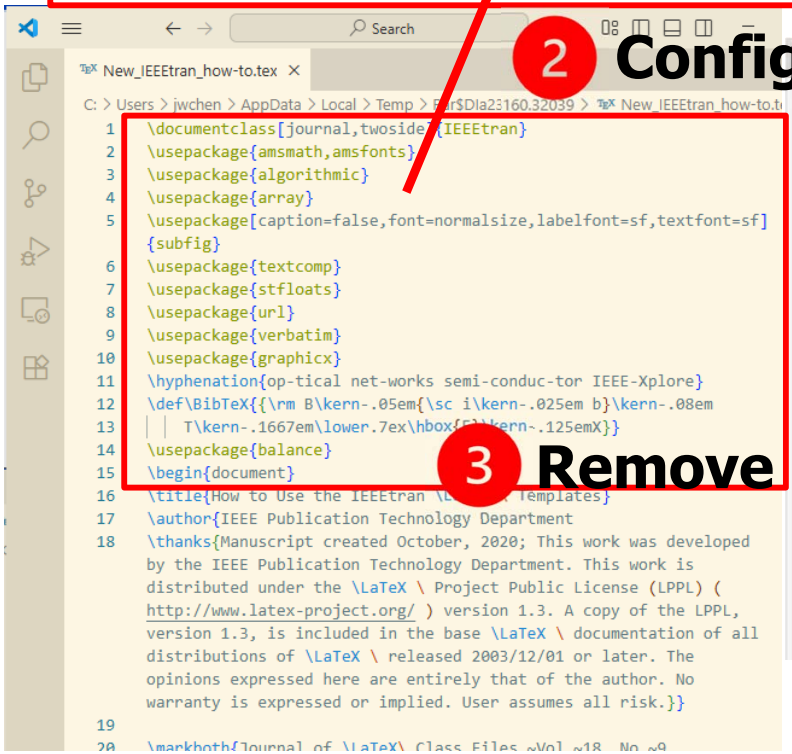


b. Or manually download it from https://template-selector.ieee.org/ and put it under your project directory.

IEEE Transactions LaTeX Template

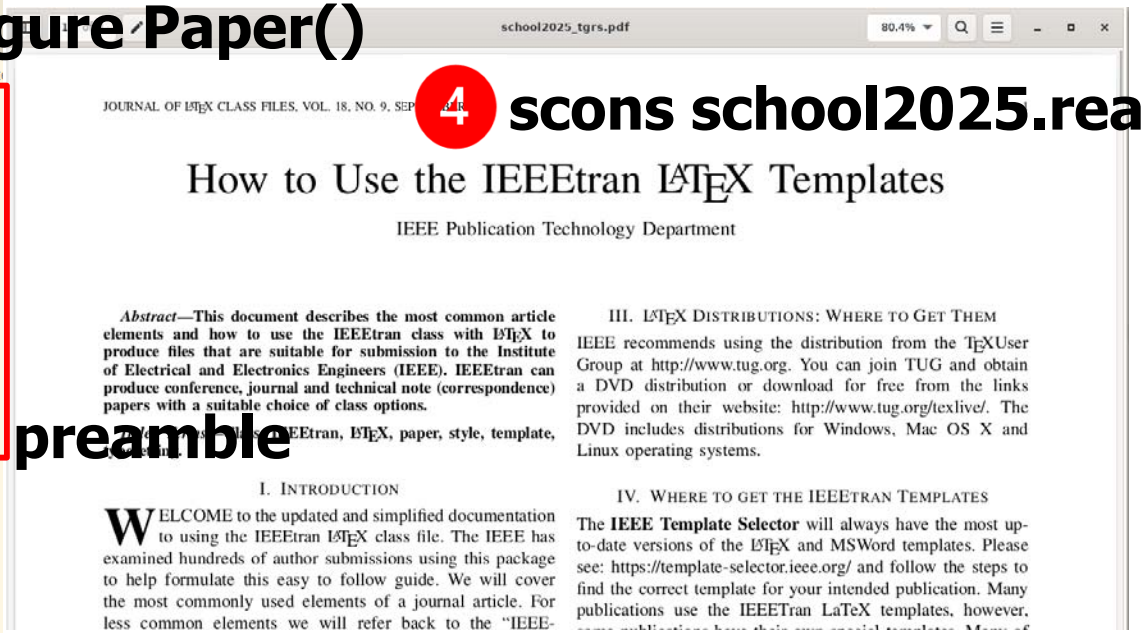
```
Paper('school2025','school2025',
        lclass='IEEEtran',
        options='journal, twoside',
        use='''amsmath,amsfonts algorithmic array
        [caption=false,font=normalsize,labelfont=sf,textfont=sf]subfig
        textcomp stfloats url verbatim graphicx balance
       '''.split(),
       include=r'''\hyphenation{op-tical net-works semi-conduc-tor IEEE-Xplore}
\def\BibTeX{{\rm B\kern-.05em{\sc i\kern-.025em b}\kern-.08em
    T\kern-.1667em\lower.7ex\hbox{E}\kern-.125emX}}''')
```

2 Configure Paper()



3 Remove preamble

4 scons school2025.read



Outline

- Motivations
- Prerequisites
- Figure preparation
  - Plotting with M8R
  - Plotting with Python
  - Plotting with Matlab/Octave
  - Plotting with GMT
- SEGTeX
- Alternative LaTeX template
- Simple Example

Simple Example

Never start from scratch!

STEP 1:

$git clone https://github.com/cplusv2023/school2025.git ~/school2025

$cd ~/school2025

or

$cd /directory/where/your/madagascar/book/rsf/school2025

STEP 2:

$sftour pscons lock

STEP 3:

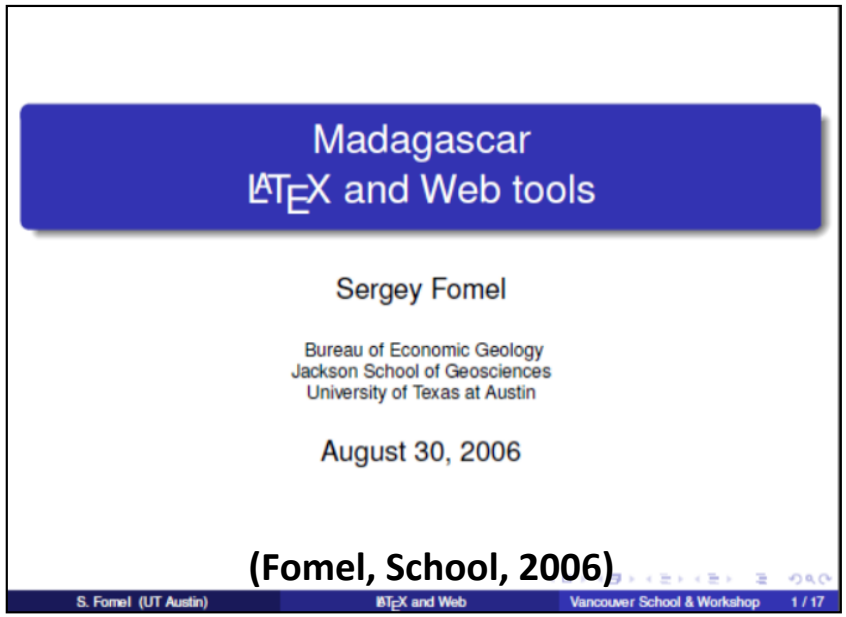
$scons school2025.read

STEP 4:

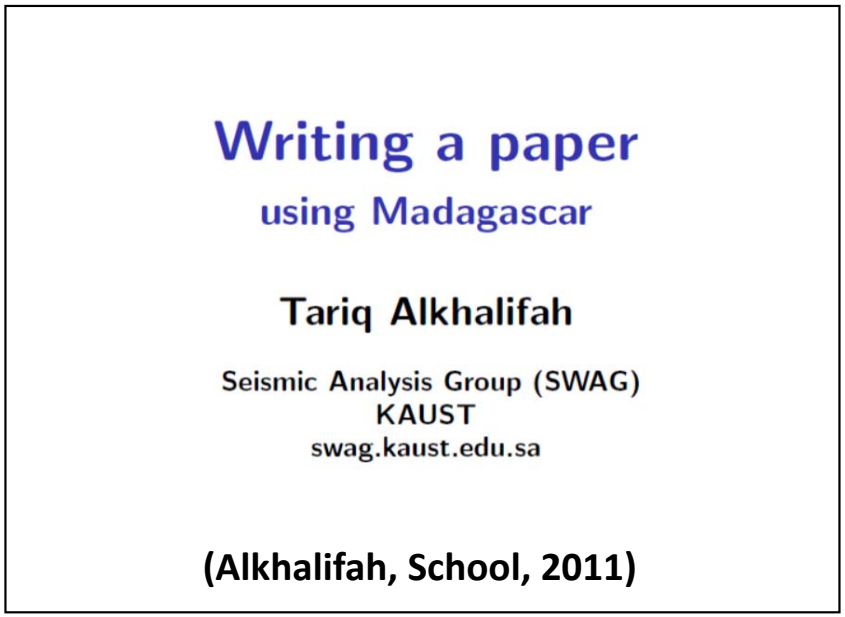
- ✓ **Change examples in SConstruct[2] and LaTeX files**
- ✓ **Change parameters in SConstruct[1] to get different template**
- ✓ **Loop over STEP 2-4**

Asking AI helps!

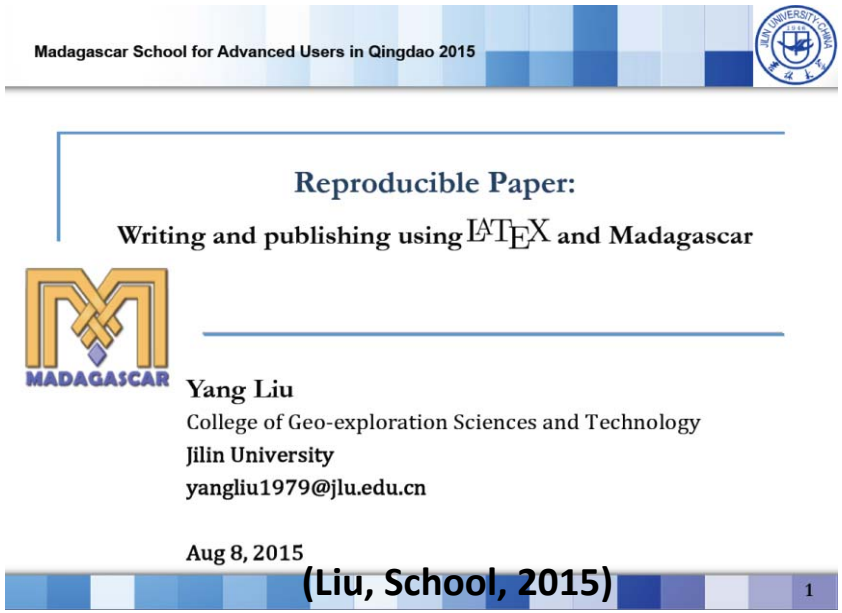
References



(Fomel, School, 2006)



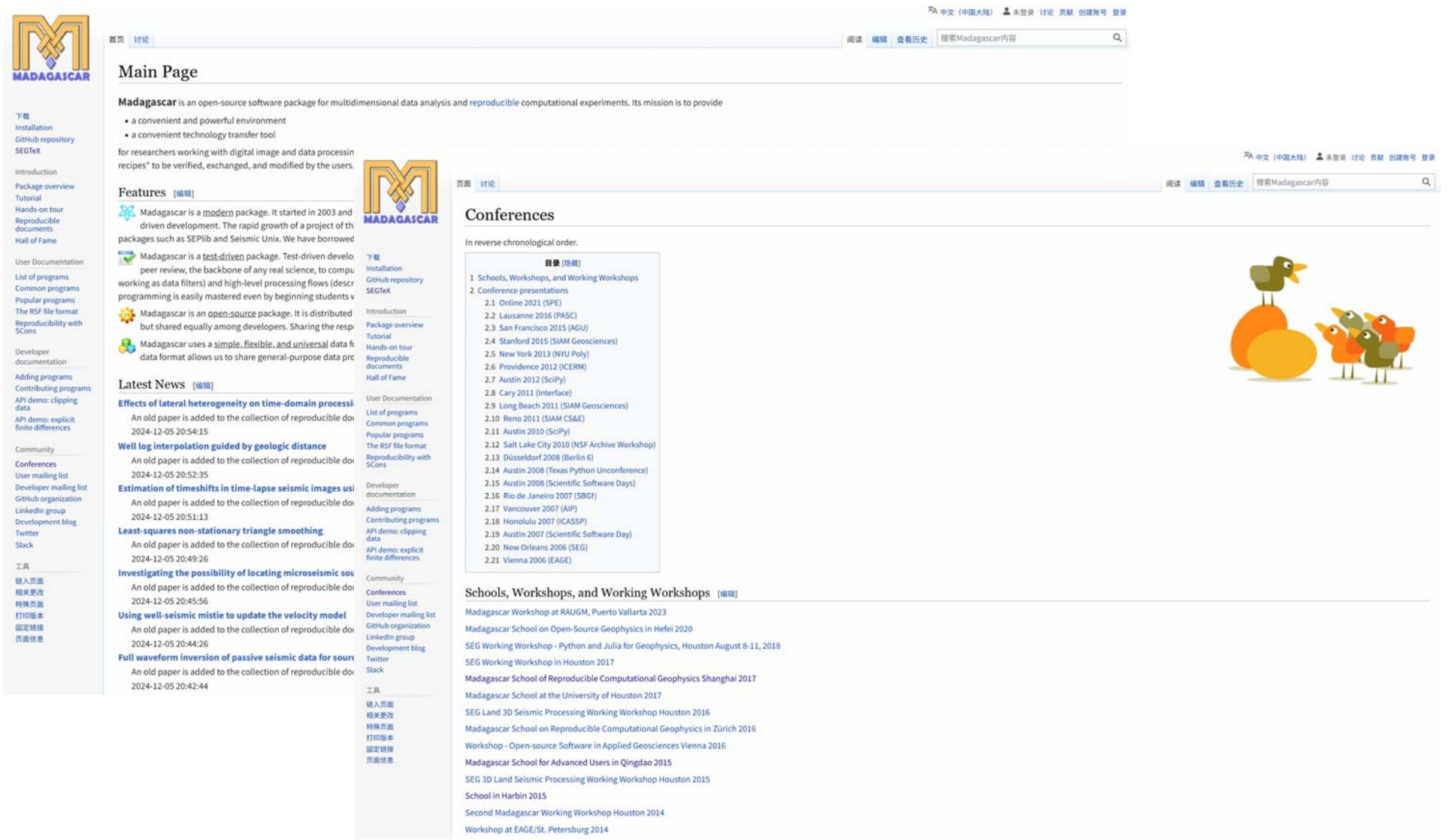
(Alkhalifah, School, 2011)



(Liu, School, 2015)

(Cheng and Liu, School, 2017)

References



https://www.ahay.org

https://www.ahay.org/wiki/Conferences

THANKS FOR ATTENTION AND HAVE A FUN!

Stand on each other's shoulders
(not each other's toes)!

Modify/add your own data tests, write your own papers …

http://www.ahay.org