



Reproducible Paper:

Writing technical papers using
Madagascar and L^AT_EX

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

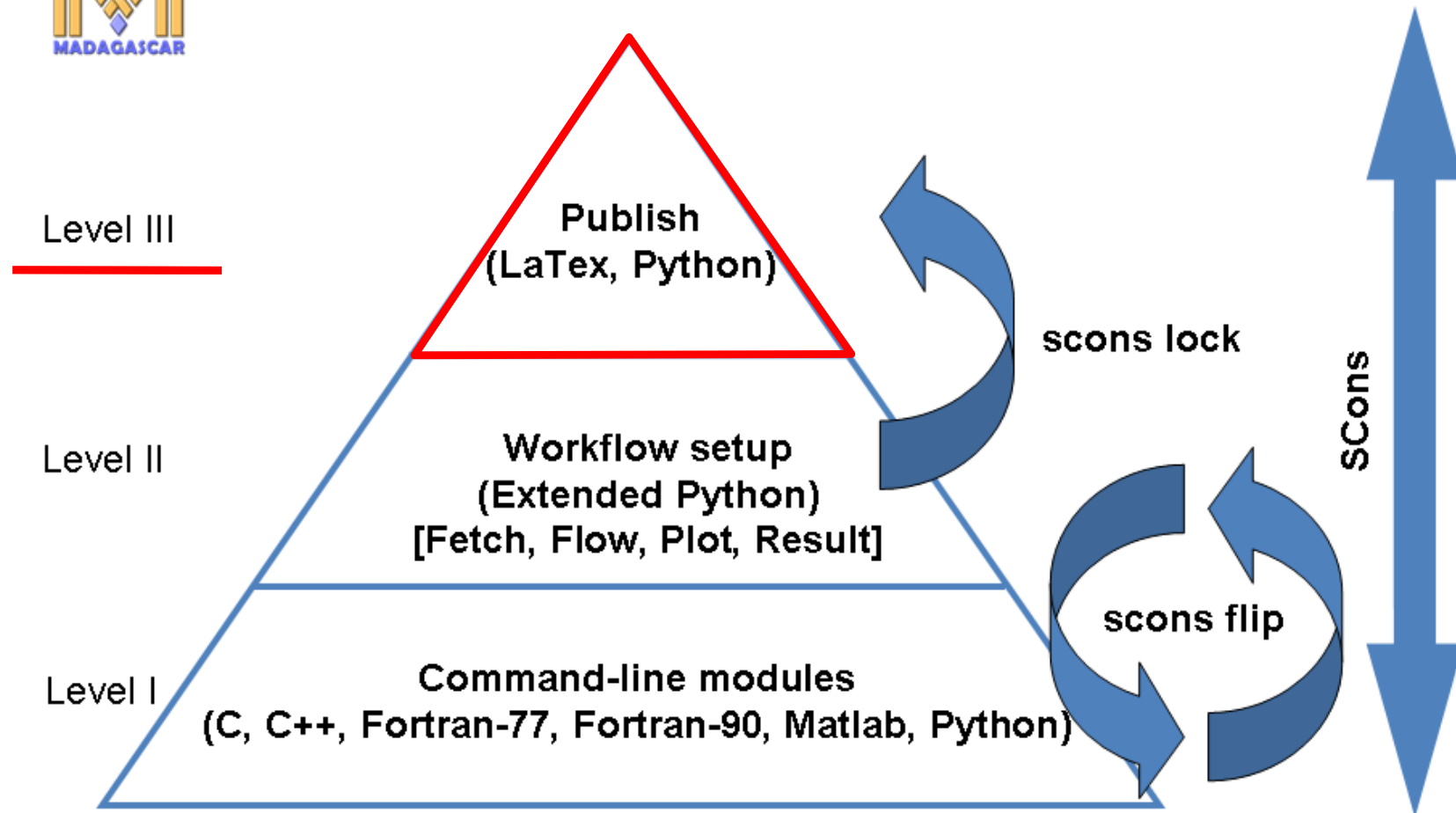


- **Motivations**
- Prerequisites
- Figure preparation
 - Plotting with M8R
 - Plotting with Python
 - Plotting with Matlab/Octave
 - Plotting with GMT
- SEGTeX
- Alternative $\text{\LaTeX}$ template
- Simple Example

Motivations

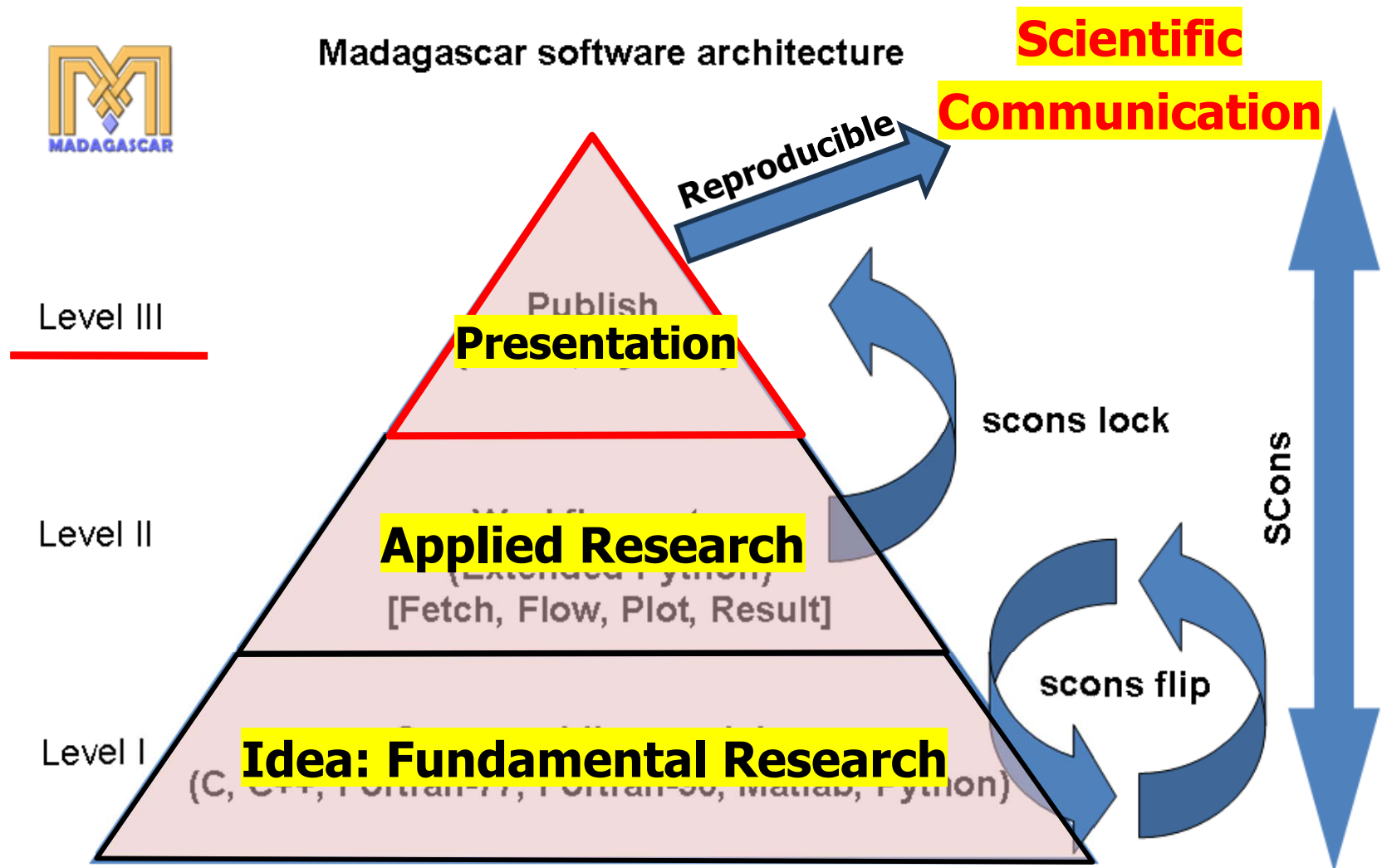


Madagascar software architecture



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>

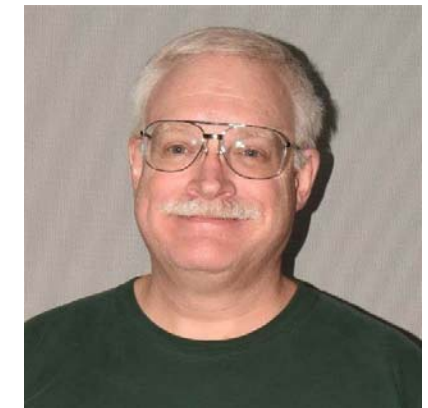
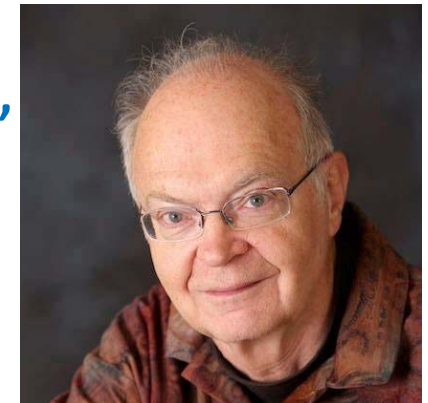
- MS-word--> $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$

“I hope to die before I have to use Microsoft Word”

– Donald Ervin Knuth

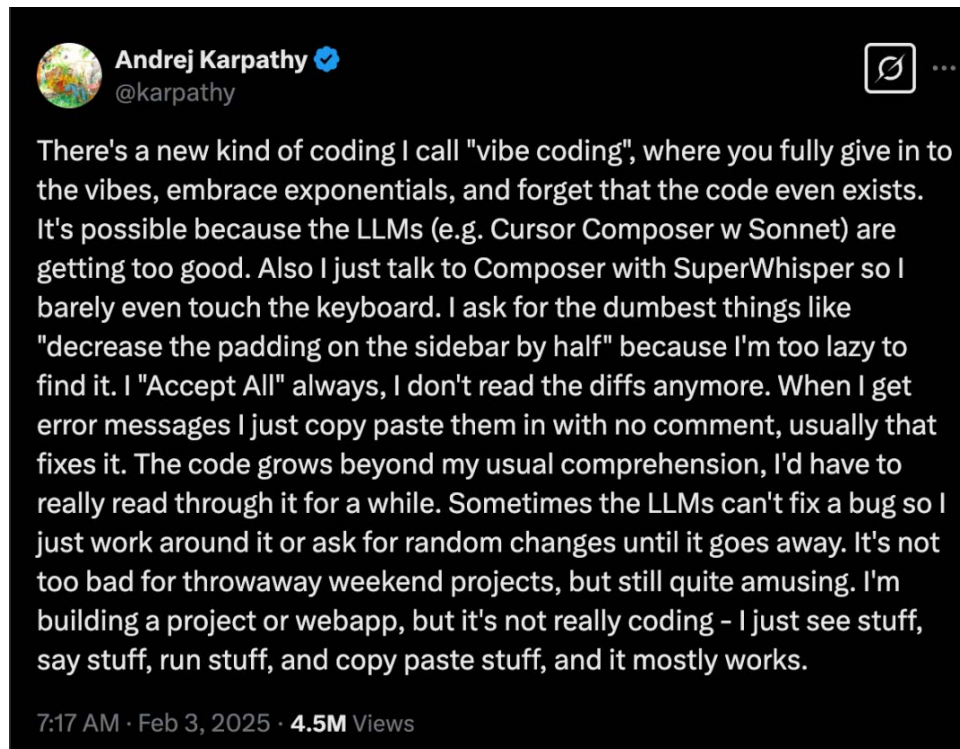
Reasons:

- 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.

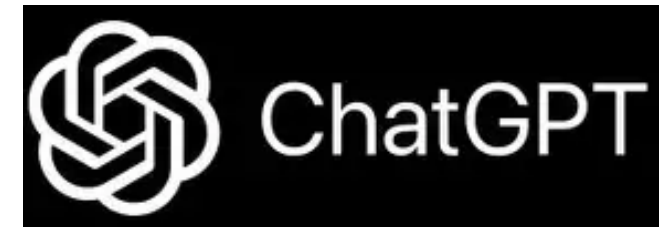


- “All technical papers should be "reproducible" in the sense that someone of reasonable skill ought to be able to read the paper and then reproduce the results.”
– Joe Dellinger
- Remember, LaTeX is not perfect, either MS-word or LaTeX is just a tool, choose any of them you need.

Vibe/LLM Coding makes it easier than ever



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



SEGTex : When LaTeX meets Madagascar





download
Installation
GitHub repository
SEGTex

Introduction

- Package overview
- Tutorial
- Hands-on tour
- Reproducible documents

User Documentation

- List of programs
- Common programs
- The RSF file format
- Reproducibility with SCons

Developer documentation

- Adding programs
- Contributing programs
- API demo: clipping data
- API demo: explicit finite differences

Community

- User mailing list
- Developer mailing list
- GitHub organization
- SourceForge project
- Conferences
- LinkedIn group
- Development blog

<http://www.ahay.org/wiki/SEGTex>

SEGTex

SEGTex is a LaTeX package for geophysical publications. It consists of

- LaTeX2e class files for [Geophysics](#) papers, SEG expanded abstracts, etc
- BibTeX style files [seg.bst](#)
- BibTeX cumulative bibliography of geophysical publications [SEG.bib](#)
- [latex2html](#) [cu](#)

Contents [hid]

- Downloading
- Installation
- 2.1 Prerequisite
- SEG expanded at
- Documentation
- Tips and tricks
- Communication
- Reproducible pap
- LaTeX references
- News

Downloadin

Access SEGTex th

Alternatively, acces

```
git clone ht
```

or using Subversio

```
svn co https
```

LaTeX package for paper submission to GEOPHYSICS

A typesetting package is available to help authors prepare papers for GEOPHYSICS. The package consists of a set of macros prepared for LaTeX, a popular document preparation system. The SEG has developed several macros designed specifically for GEOPHYSICS and the SEG Annual Meeting *Expanded Abstracts*.

Authors who use the LaTeX typesetting program to prepare their manuscripts can use the SEG macros (called SEGTex) to format the text, equations, references, and appendices so that they conform to GEOPHYSICS guidelines for submission. If using BibTeX to create references, authors must run BibTeX before submitting the .tex file and read in or paste the resulting contents of the generated .bbl file within the bibliography section of the .tex file. All LaTeX submissions must include only one .tex file and a PDF of that file. You may send questions concerning LaTeX files to the SEGTex mailing list.

To submit papers to GEOPHYSICS, follow the procedures described in the SEG *Instructions to Authors*" (in the January-February issue of GEOPHYSICS and on this Web site at the link above). The manuscript will undergo the standard review process. Once all revisions and changes to the manuscript are made and the manuscript is accepted, LaTeX files will be converted to Microsoft Word documents for production.

SEG wiki
Layering the Science of Applied Geophysics
Advertise with SEG

المؤتمرات العلمية العالمية
الاجتماعات العلمية العالمية
Their manuscript conference
on Engineering Geophysics
15-18 November | Al Ain, UAE

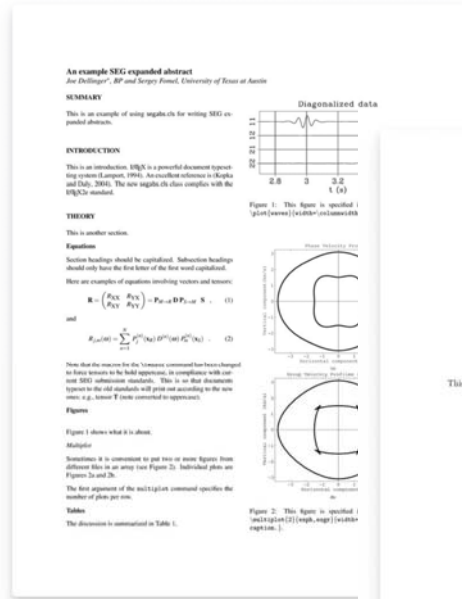
MAY 2015

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

SEGTex : When LaTeX meets Madagascar



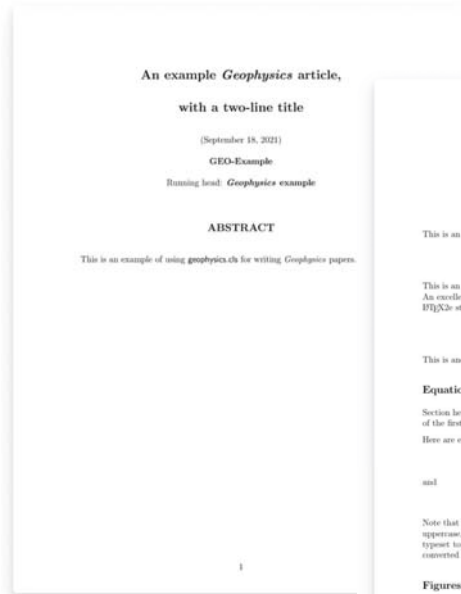
TEX



Template for SEG abstract

Template for writing SEG abstract.

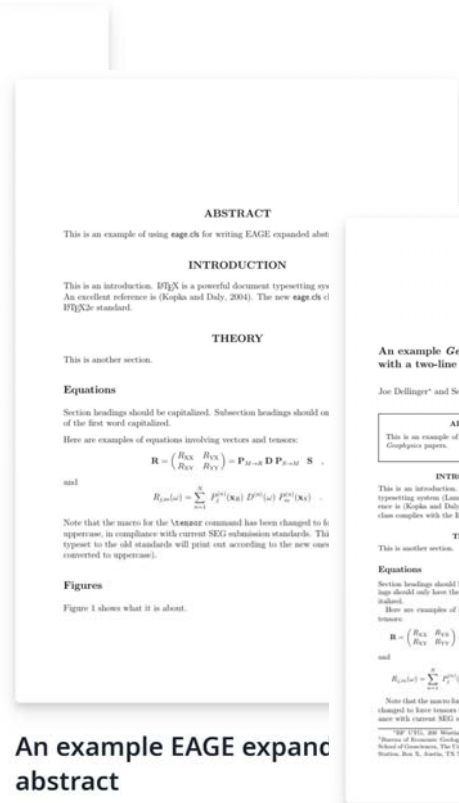
Sergey Fomel



Template for GEOPHYSICS

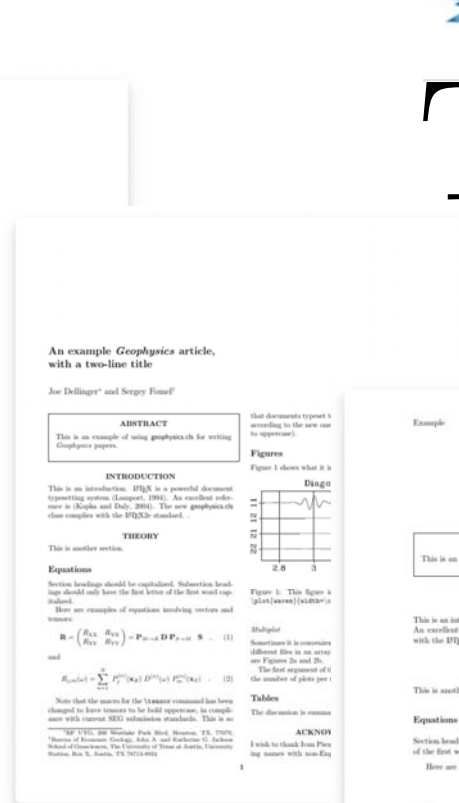
GEOPHYSICS, published by the Society of Geophysicists since 1936, is an archival journal.

Sergey Fomel



An example EAGE expanded abstract

SEGTex is a LaTeX package for geophysics publications. This template consists of



An example 2-column Geophysics article

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



An example Geophysics report

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

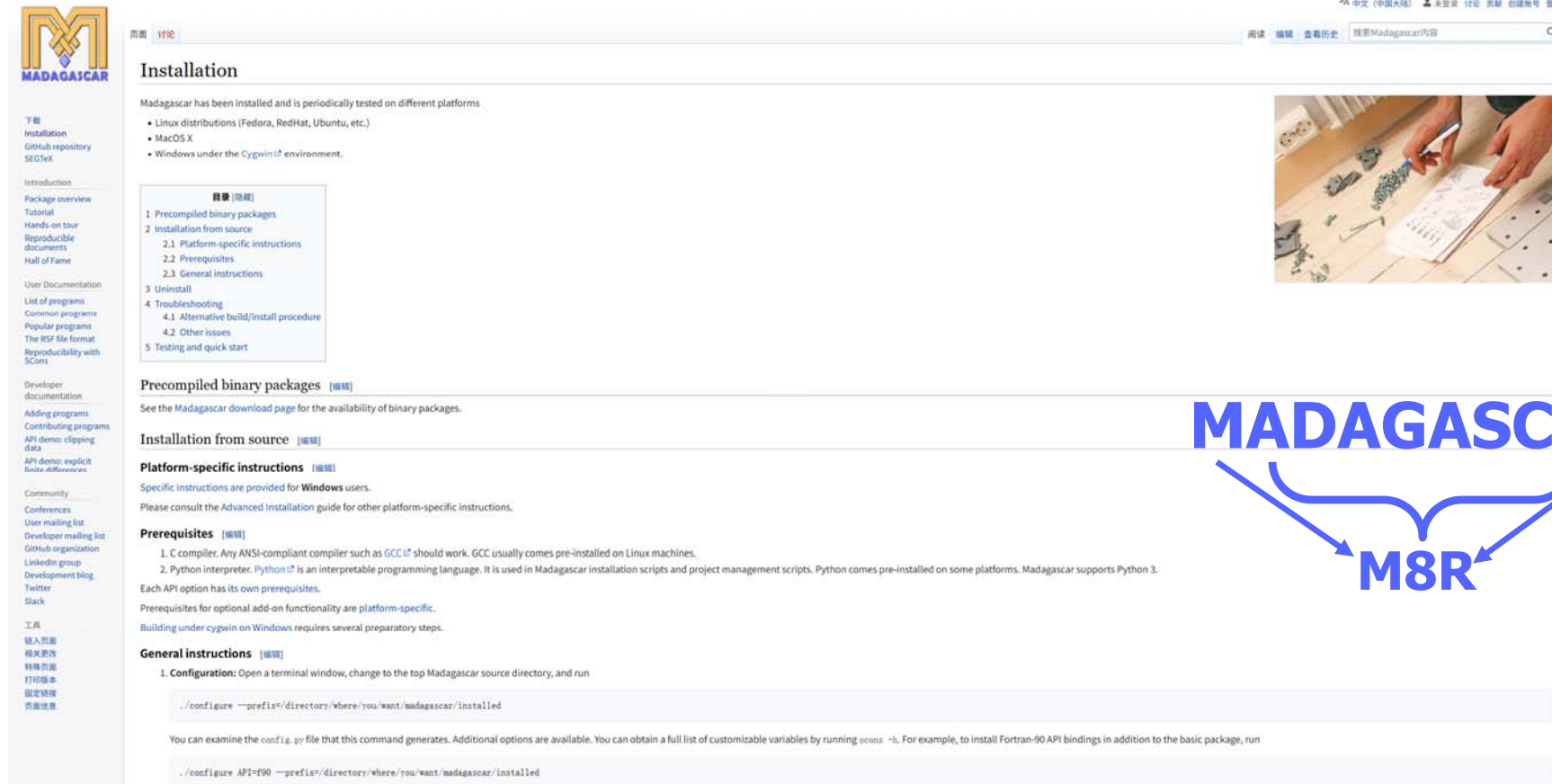
SEGTex



- Motivations
- Prerequisites
- Figure preparation
 - ▣ Plotting with M8R
 - ▣ Plotting with Python
 - ▣ Plotting with Matlab/Octave
 - ▣ Plotting with GMT
- SEGTeX
- Alternative $\text{\LaTeX}$ template
- Simple Example

Prerequisites

1) Download and install Madagascar



The screenshot shows the Madagascar installation page on the ahay.org website. The page is titled 'Installation' and contains the following sections:

- Precompiled binary packages**: See the [Madagascar download page](#) for the availability of binary packages.
- Installation from source**: Specific instructions are provided for **Windows** users. Please consult the [Advanced Installation guide](#) for other platform-specific instructions.
- Platform-specific instructions**: Specific instructions are provided for **Windows** users. Please consult the [Advanced Installation guide](#) for other platform-specific instructions.
- Prerequisites**:
 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**:
 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
```

On the right side of the page, there is a diagram showing 'MADAGASCAR' in large blue letters, with two blue arrows pointing from it to 'M8R' in large blue letters.

<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

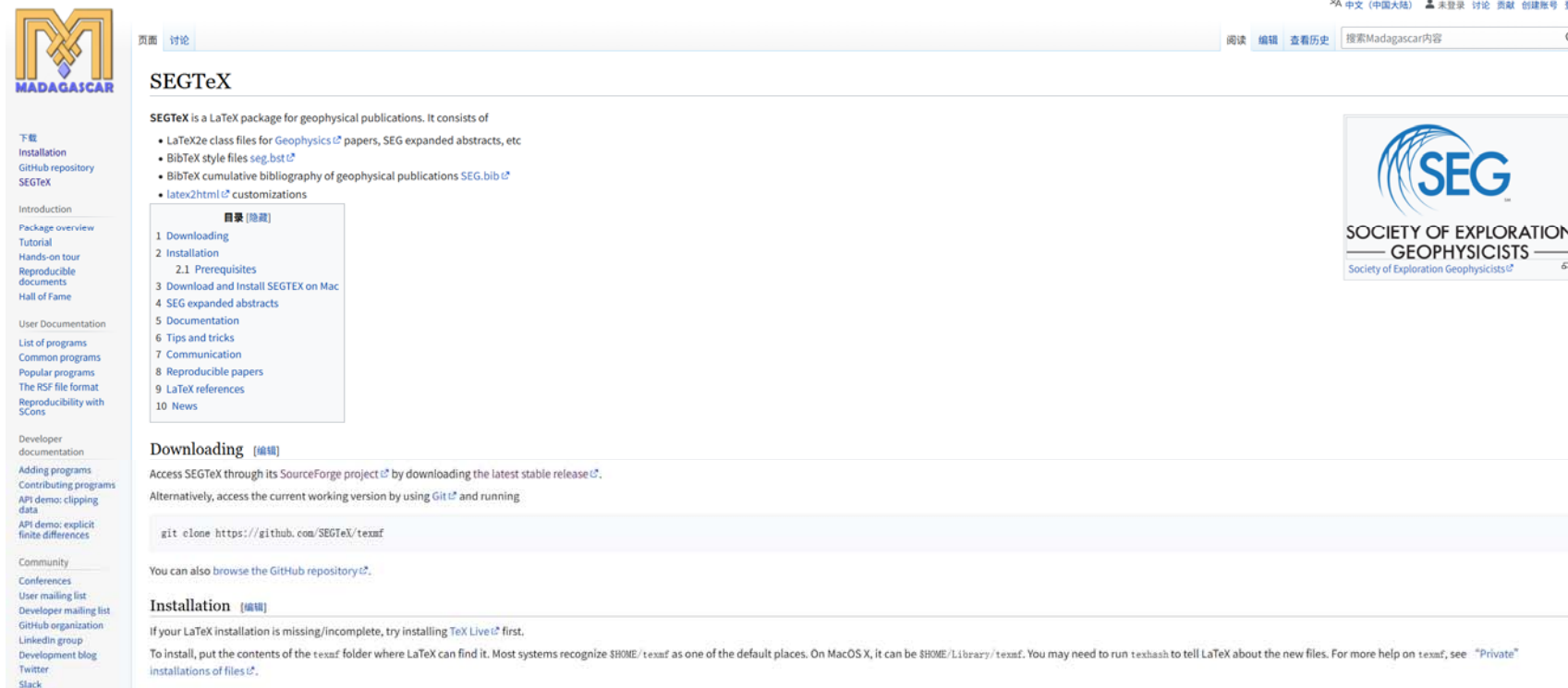
sudo apt install texlive-full

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

3) Download and configure SEGTeX

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



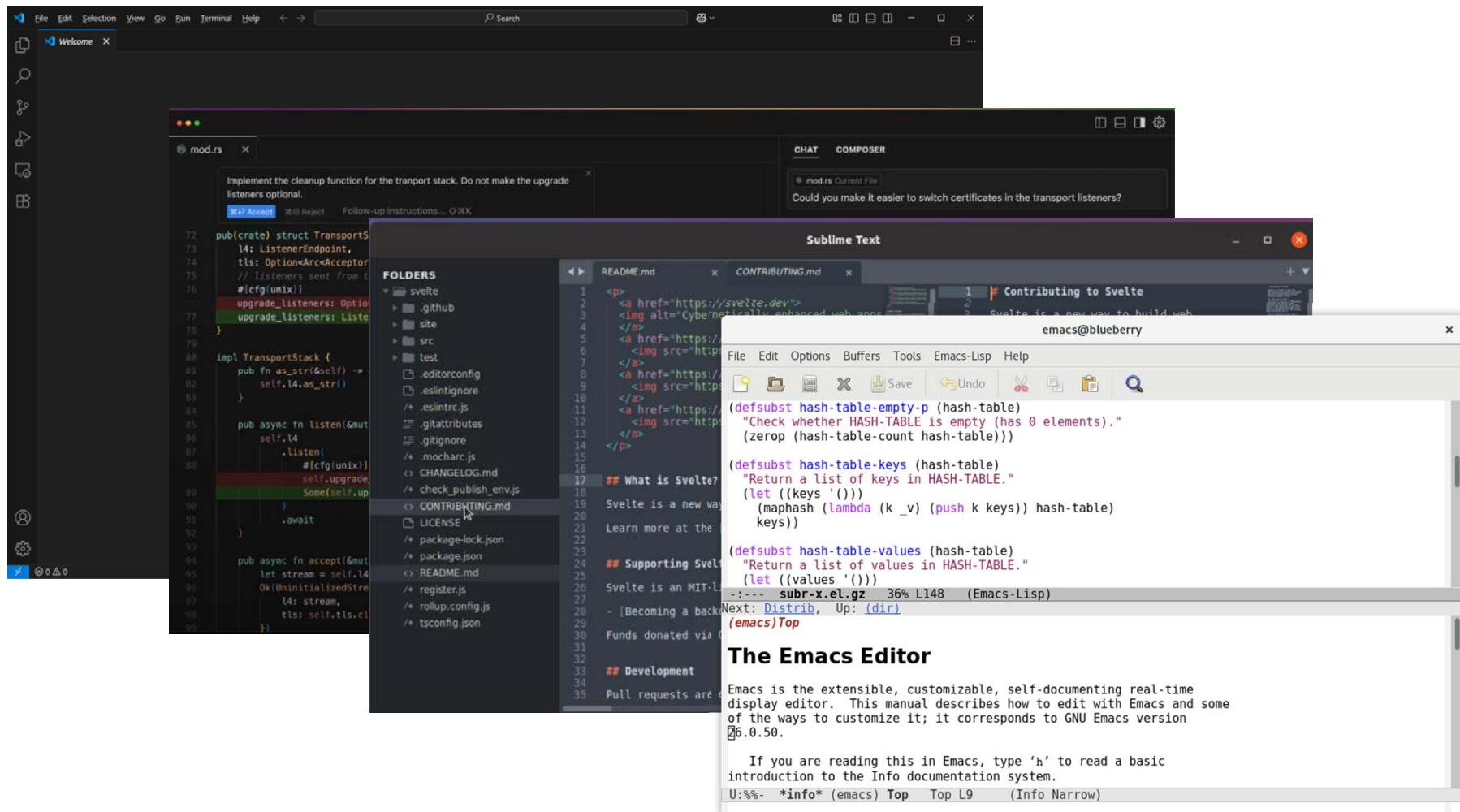
The screenshot shows the SEGTeX website. The header includes the MADAGASCAR logo and navigation links. The main content area is titled "SEGTeX" and describes it as a LaTeX package for geophysical publications. It lists the components: LaTeX class files, BibTeX style files, BibTeX cumulative bibliography, and LaTeX customizations. A table of contents is provided, listing sections from Downloading to News. The "Downloading" section is expanded, showing instructions on how to access SEGTeX through its SourceForge project or by using Git. A code block shows the command to clone the repository: `git clone https://github.com/SEGTeX/texmf`. The "Installation" section is also visible, providing instructions on how to install the package.

(2) Add environment variable TEXMFHOME:

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

4) Choose your favorite text editor

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





- Motivations
- Prerequisites
- **Figure preparation**
 - ❑ Plotting with M8R
 - ❑ Plotting with Python
 - ❑ Plotting with Matlab/Octave
 - ❑ Plotting with GMT
- SEGTeX
- Alternative $\text{\LaTeX}$ template
- Simple Example

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 sfsdtplo:
<http://ahay.org/RSF/sfstdplot.html>

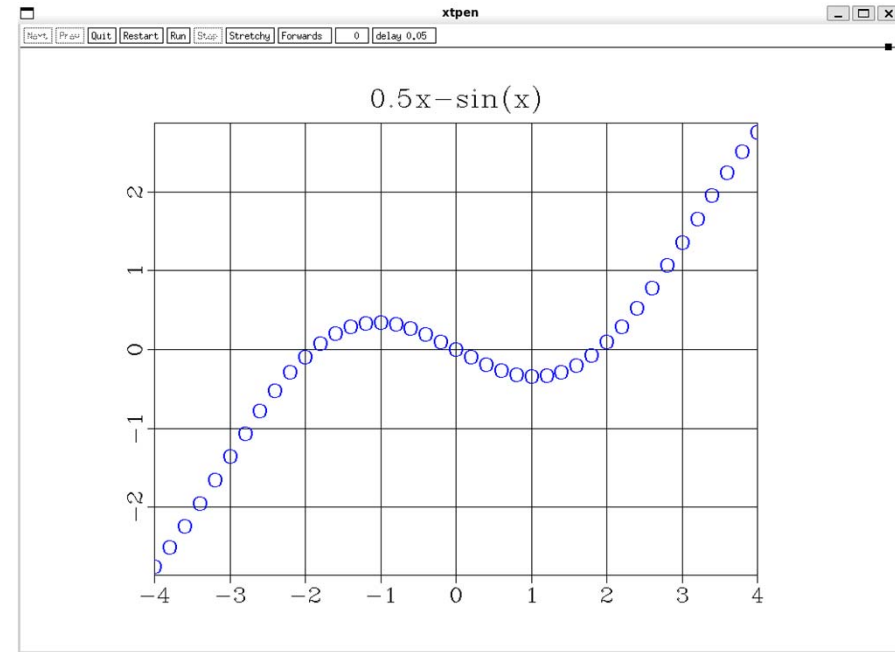
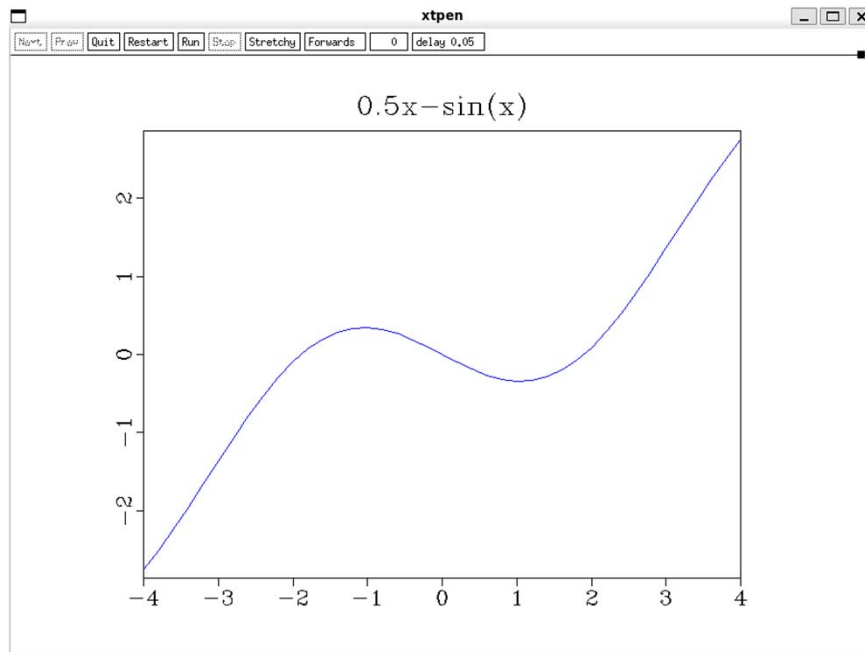
sfgraph



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

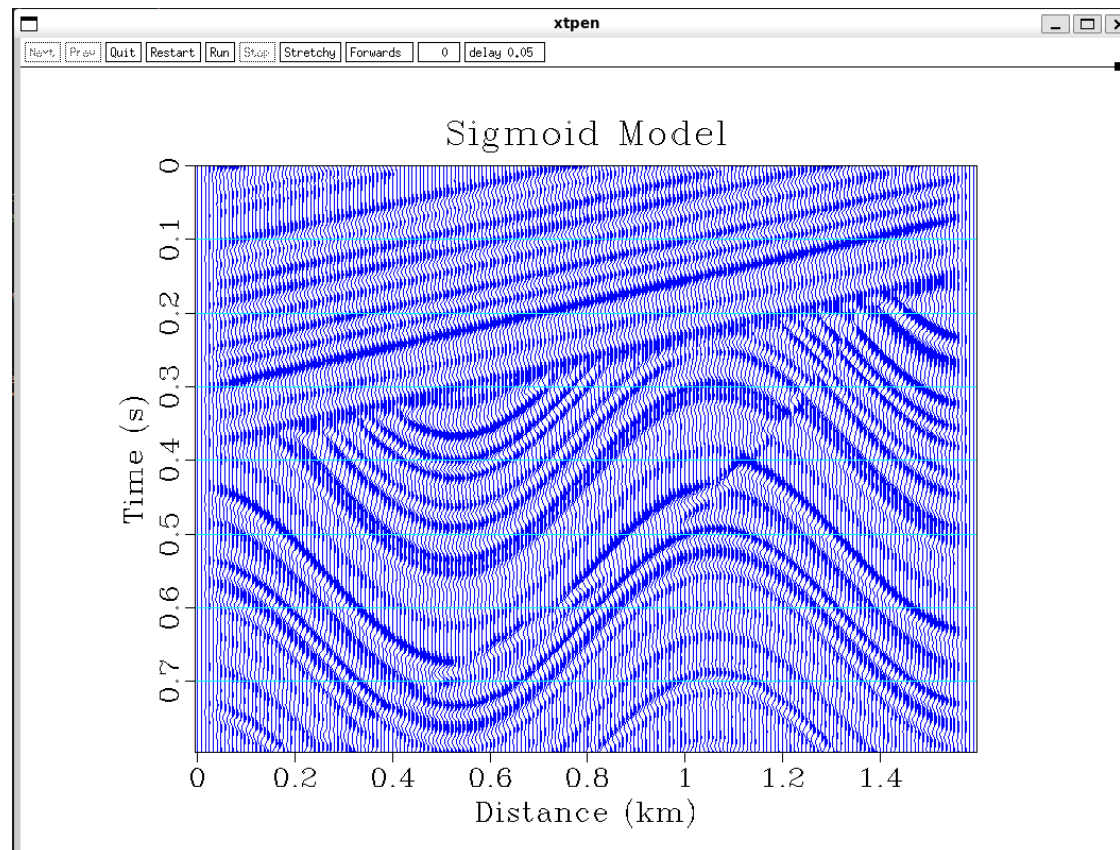
```
< y.rsf sfgraph title="0.5x-sin(x)" min1=-4 max1=4 | xtpen
```

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

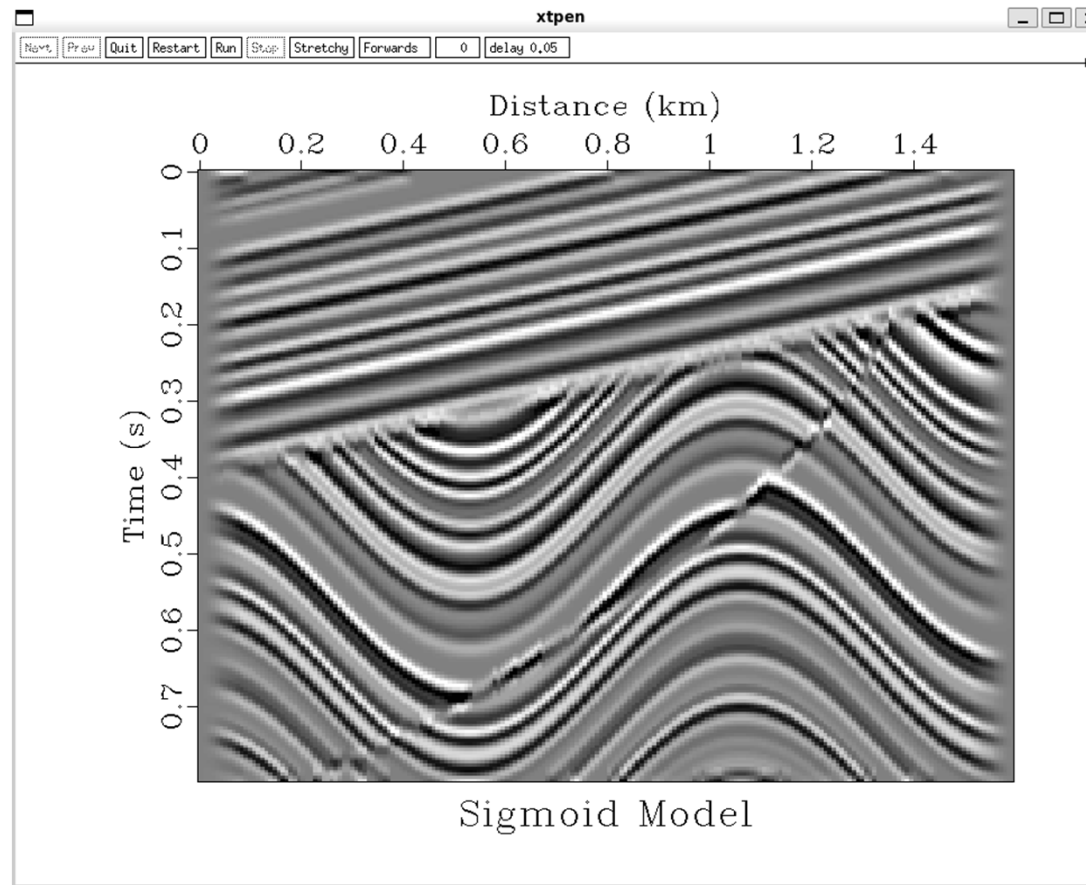


```
export SFPENOPTS='bgcolor=w'
```

```
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
```



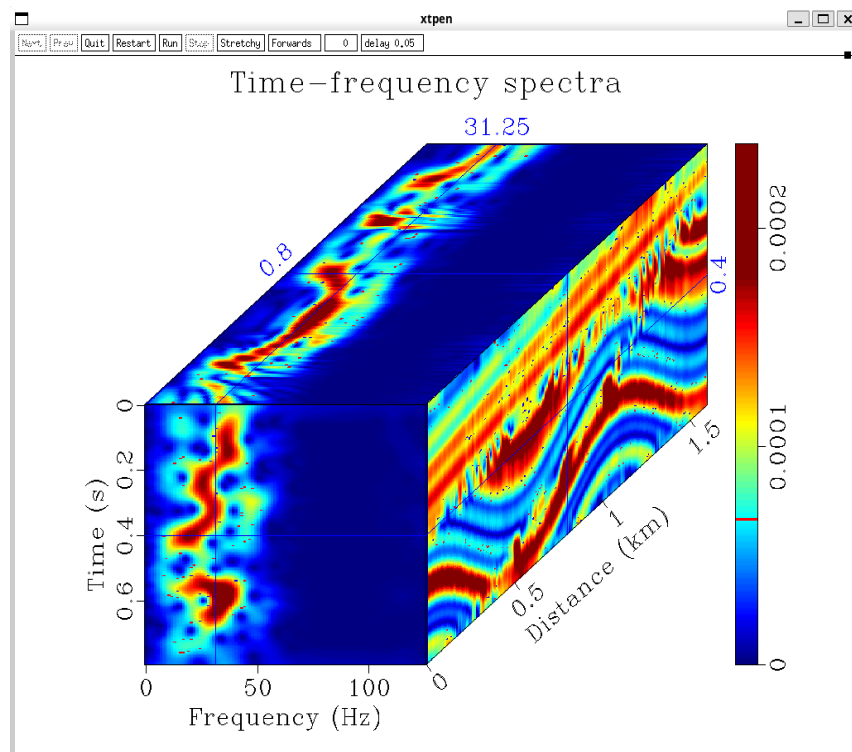
```
sfsigmoid n1=200 n2=200 d1=.004 d2=.008 \  
  | sfsmooth rect1=3 diff1=1 | sfsmooth rect1=3 \  
  | sfput label2=Distance > model.rsf  
< model.rsf sfgrey title="Sigmoid Model" | xtpen
```



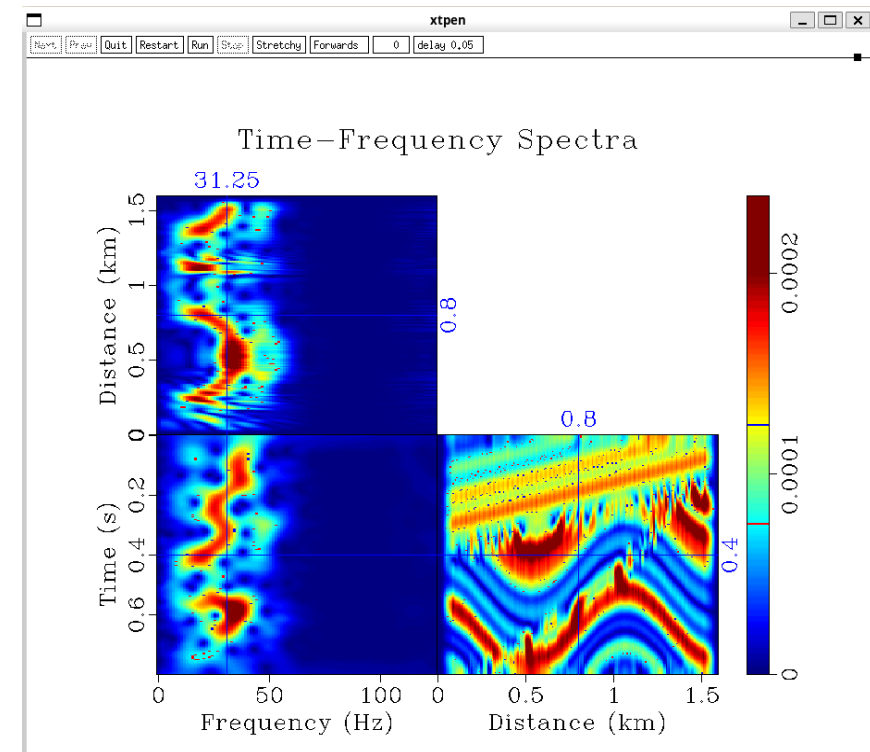
sfgrey3



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



flat=n

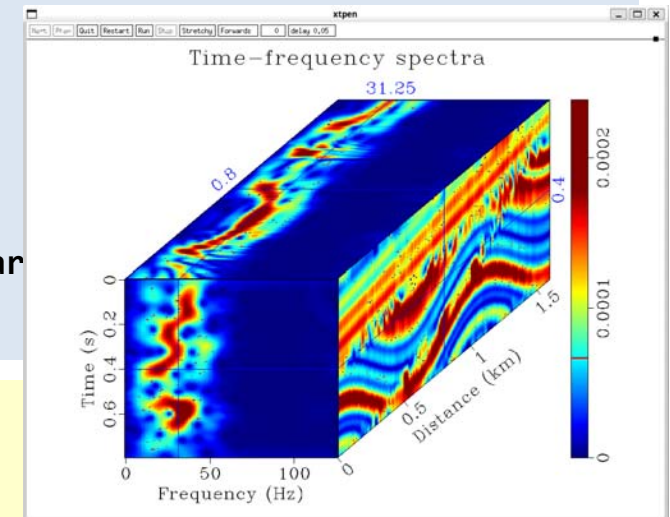


Default: flat=y

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 sflttf 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= nbartics= dbarnum= obarnum= wherebartics= n1tic= d1num= o1num=
    n2tic= d2num= o2num= n3tic= d3num= o3num= n4tic= d4num= o4num= whereticks= 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)
```

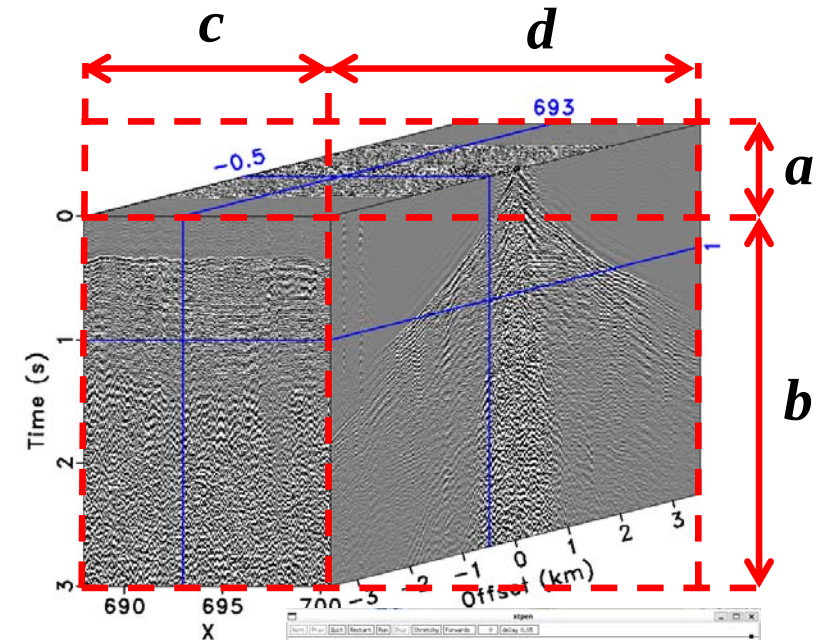
Tips for M8R plotting



1) Aspect ratio settings for `sfgrey3`

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

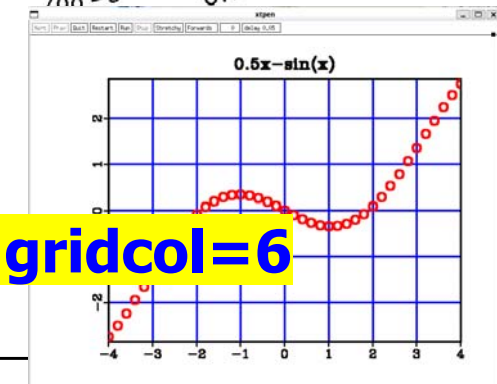
$$\text{point2} = \frac{c}{c+d} = 0.4$$



2) Element color

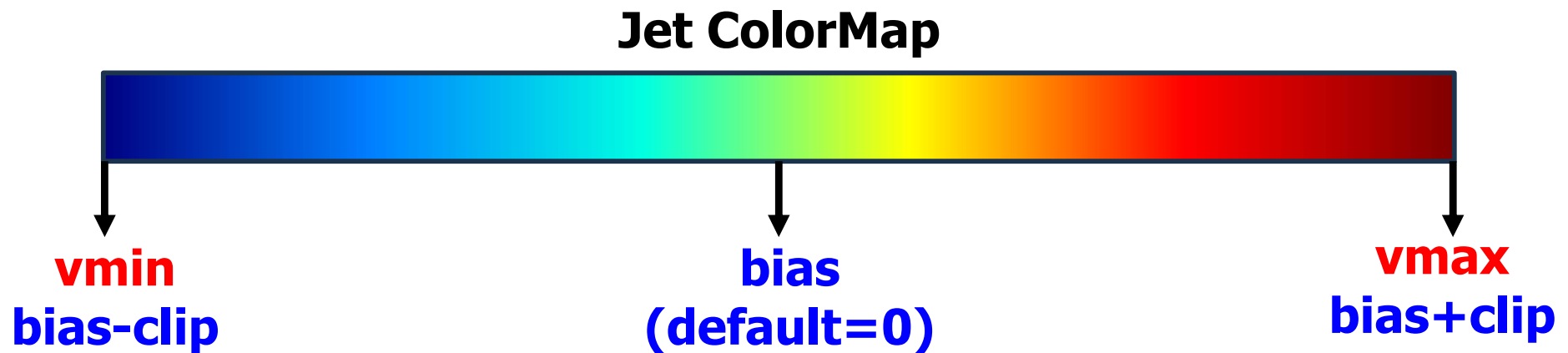
Lines, labels, frames, grids...

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

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];**

4) Figure combinations

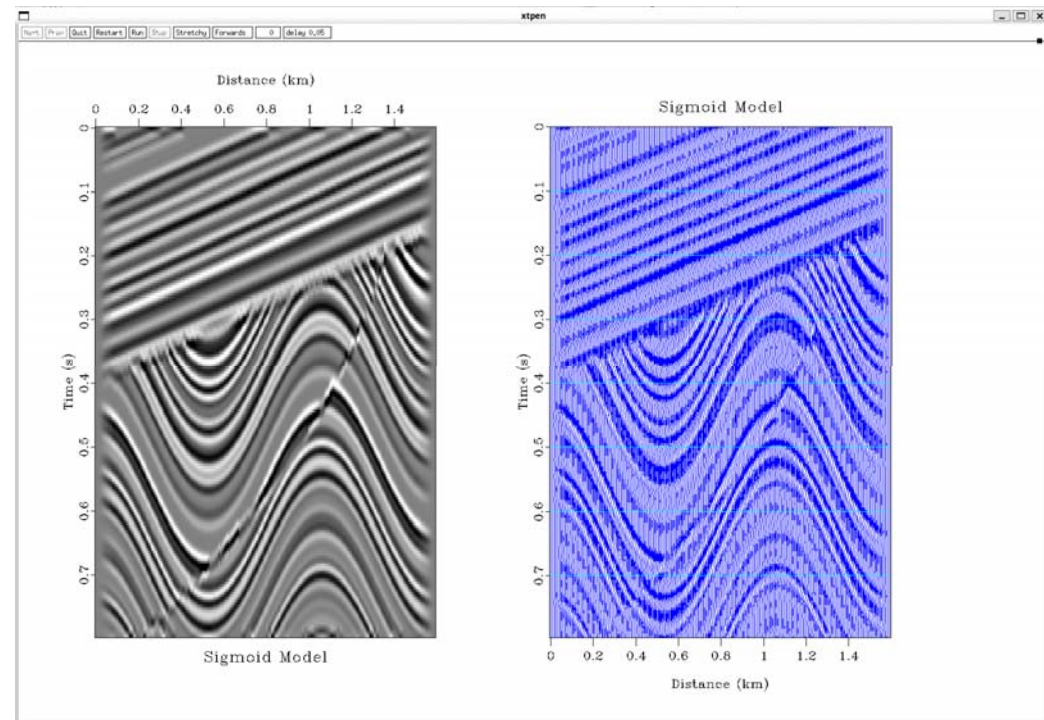
```
Plot('subfig1', 'model', 'grey')
```

```
Plot('subfig2', 'model', 'wiggles transp=y yreverse=y poly=y')
```

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

Combinations

Movie
Overlay
SideBySideAniso
OverUnderAniso
SideBySideIso
OverUnderIso
TwoRows
TwoColumns





- Motivations
- Prerequisites
- **Figure preparation**
 - Plotting with M8R
 - Plotting with Python
 - Plotting with Matlab/Octave
 - Plotting with GMT
- SEGTeX
- Alternative $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ template
- Simple Example

Plotting with Python: sfmatplotlib



Plotting with matplotlib

September 12, 2019 Programs 1 comment

matplotlib

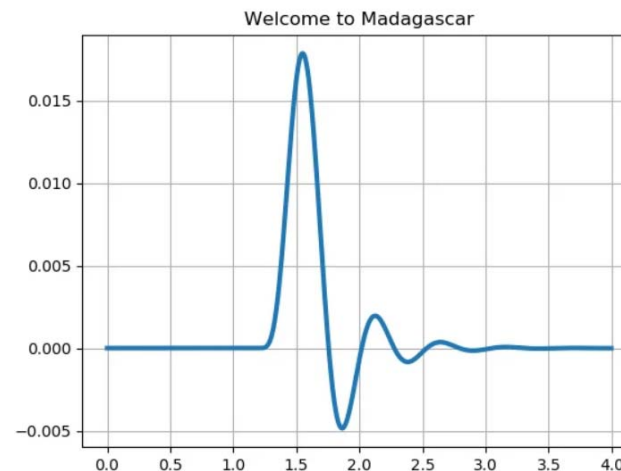
Matplotlib is a popular Python library for 2D plotting, which emulates (and exceeds) the plotting functionality of MATLAB.

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.



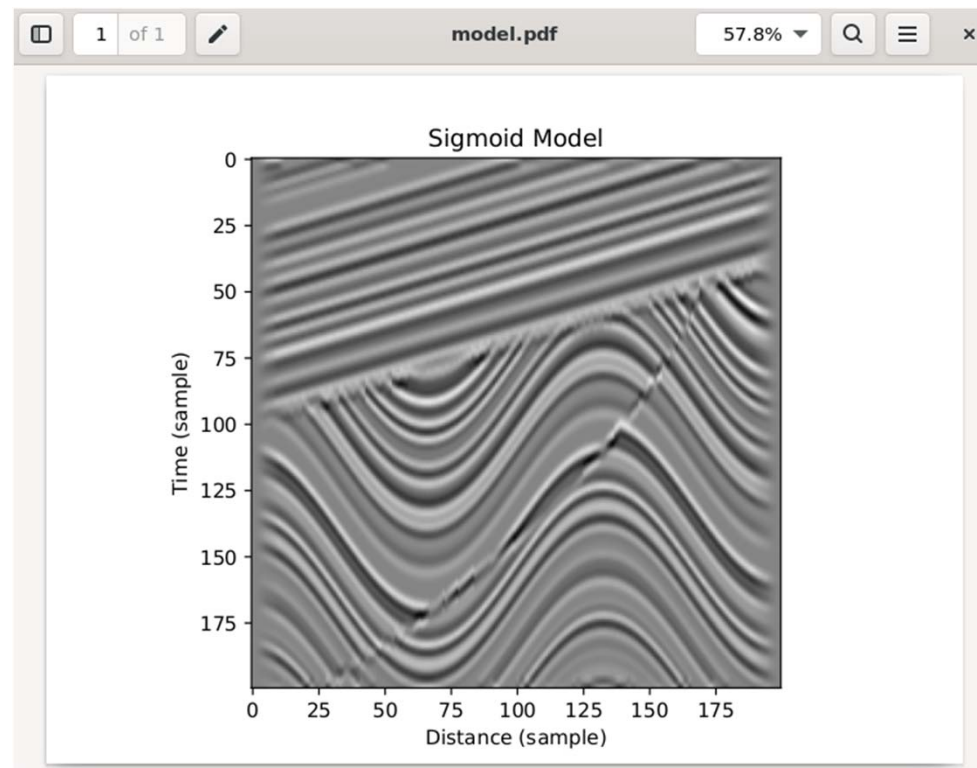
The figure above was generated with

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

Plotting with Python: sfmatplotlib



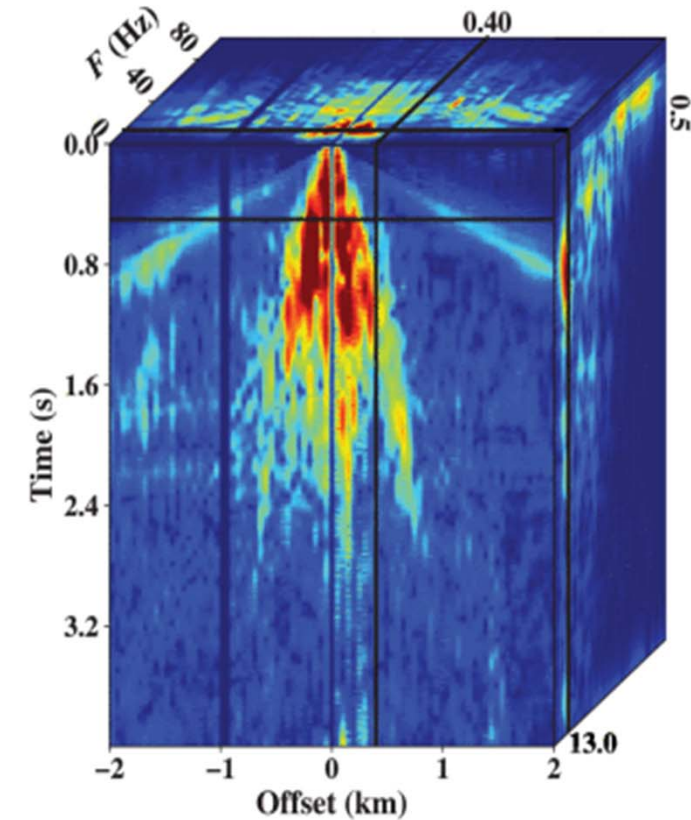
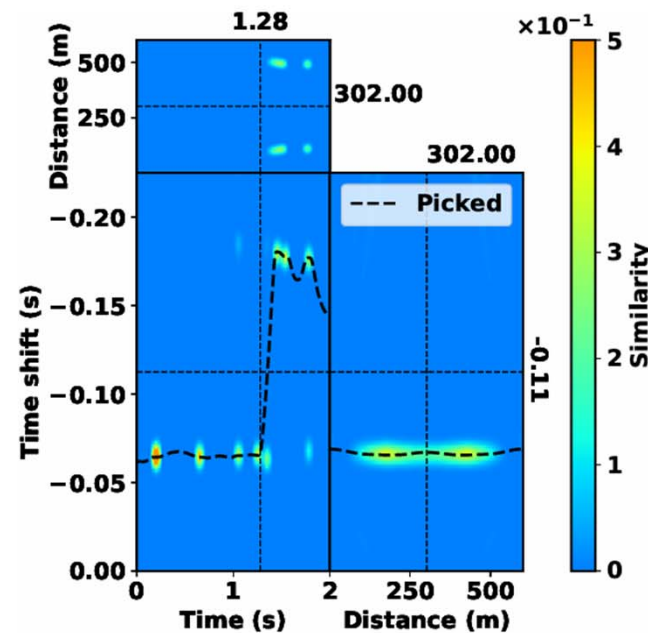
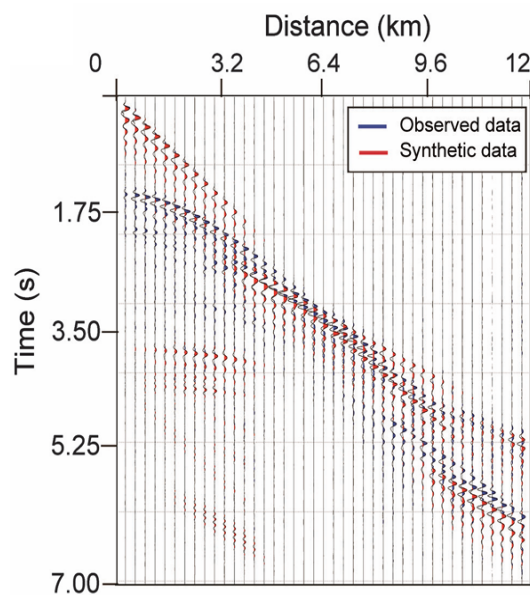
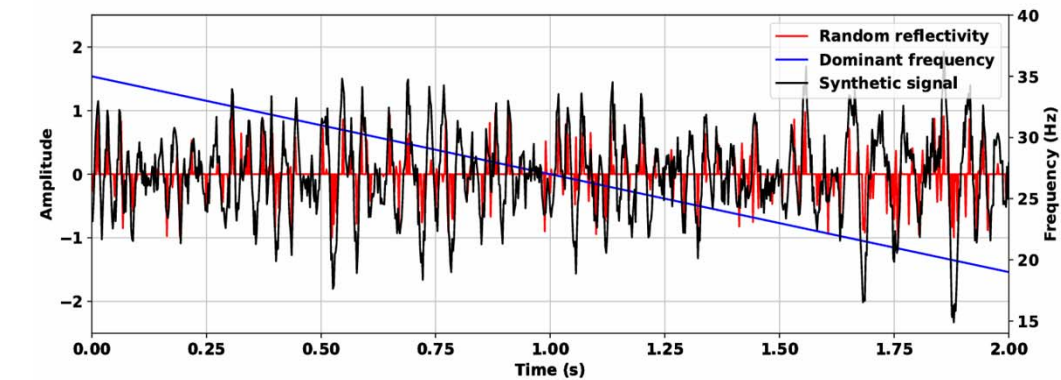
```
sfsigmoid n1=200 n2=200 d1=.004 d2=.008 \  
  | sfsmooth rect1=3 diff1=1 | sfsmooth rect1=3 \  
  | sfput label2=Distance > model.rsf  
< model.rsf sftransp | sfmatplotlib imshow cmap=gray title="Sigmoid Model" xlabel="Distance  
(sample)" ylabel="Time (sample)" format=pdf > model.pdf
```



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.



1) Choose a command-line style:

(M8r style) Using stdin/stdout

```
< data.rsfc 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.

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**

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.rsfc python script.py > Fig/figure.pdf

Plotting with Python: simple example



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

```
if __name__ == "__main__":
```

```
    # check stdin
```

```
    if sys.stdin.isatty():
```

```
        print("Usage: python greytest.py < data.rsfs [> 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()
```

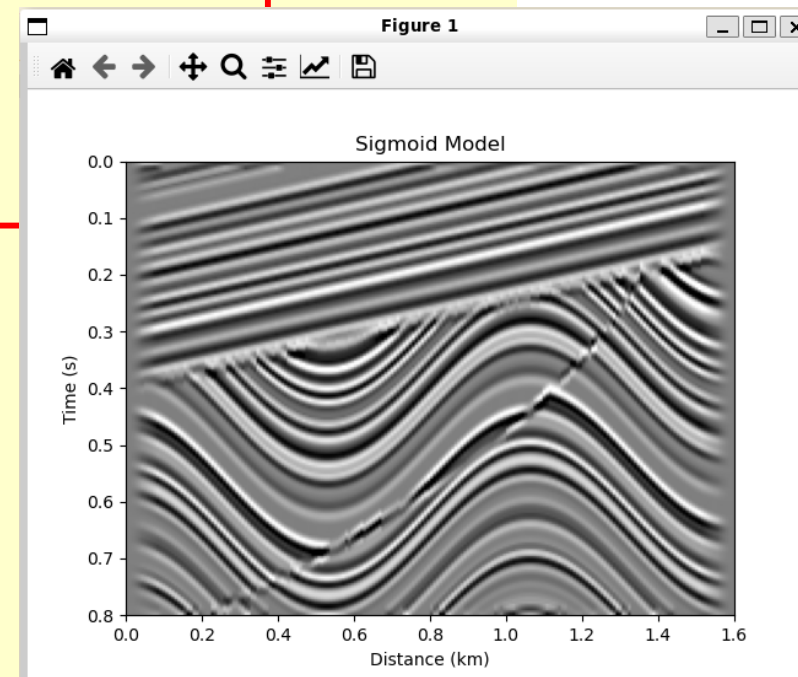
```
    else: plt.savefig(sys.stdout.buffer, format='pdf')
```

```
    sys.exit(0)
```

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

Always **float32** for rsf float data.

```
$ sfin model.rsfs
model.rsfs:
  in="/var/tmp/Desktop/school2025/plots/model.rsfs@"
  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)
```



Plotting with Python: simple example



.....

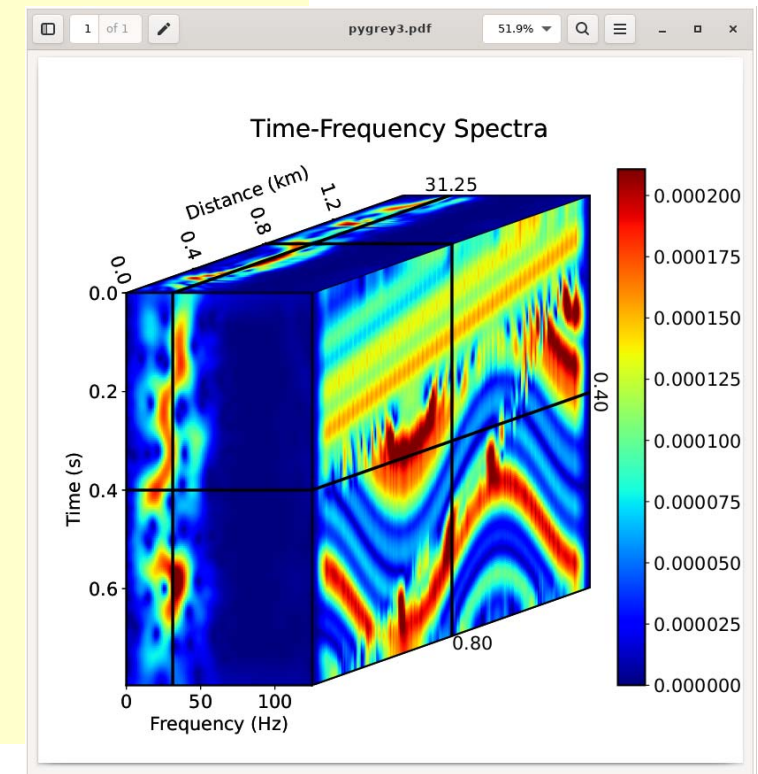
```
# read header from stdin
inp = m8r.Input()
n1, n2, n3 = inp.int('n1'), inp.int('n2'), inp.int('n3')
d1, d2, d3 = inp.float('d1'), inp.float('d2'), inp.float('d3')
o1, o2, o3 = inp.float('o1'), inp.float('o2'), inp.float('o3')
l1, l2, l3 = inp.string('label1'), inp.string('label2'), inp.string('label3')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

Use `m8r.Input()` to read parameters

```
# 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)
```

`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')
```





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

Plotting with Matlab: M8R API

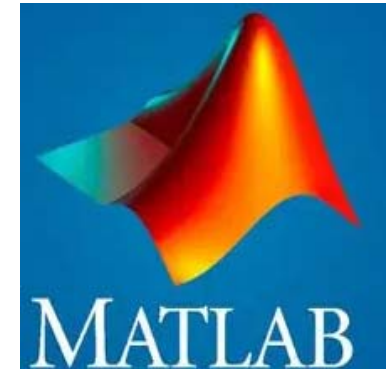


a. Use Madagascar Matlab api to read rsf data.

➤ **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)



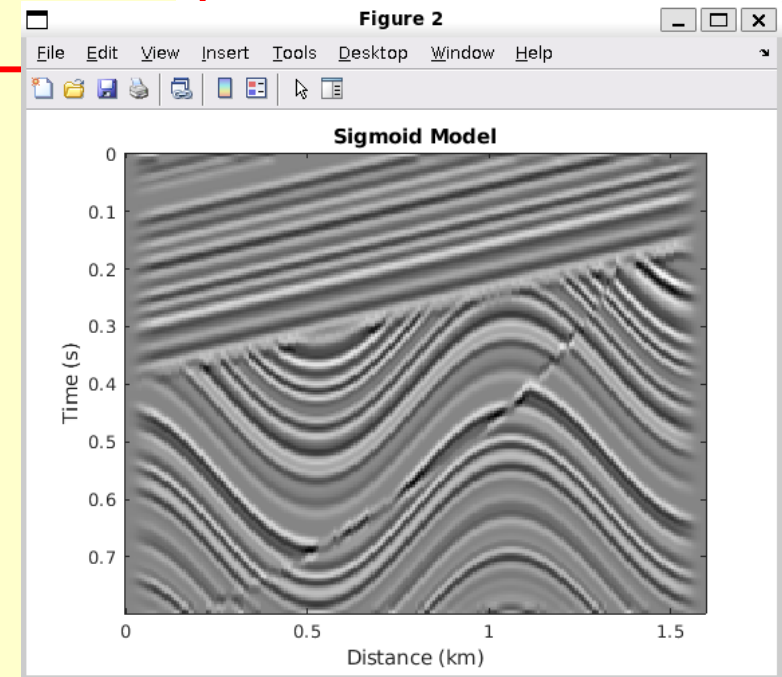
```
function matgrey(rsf_file, outfig, figtitle)
    % Require 2 or 3 input arguments
    narginchk(2,3);
    if isempty(outfig)
        error('The output filename "outfig" cannot be empty.')
    end
    if nargin < 3
        figtitle = '';
    end
    %--- 1. Read RSF header and data
    % Add the RSF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

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

scons matgrey.view



b. Read rsf data manually.

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



<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

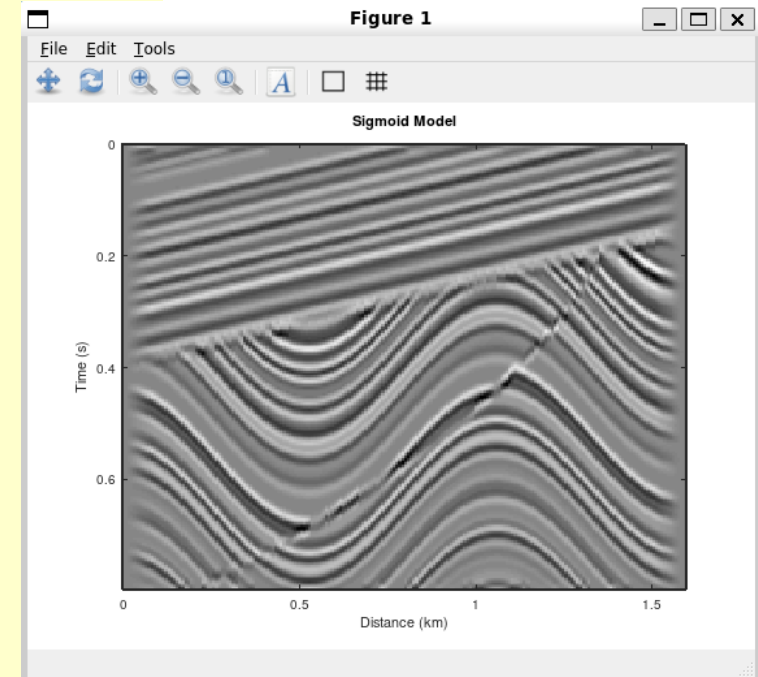


```
function matgrey(rsf_file, outfig, figtitle)
    narginchk(2,3); % Require 2 or 3 input arguments
    if isempty(outfig)
        error('The output filename "outfig" cannot be empty.');
```

Use read_rsf to read data and header.

```
    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
```





- Motivations
- Prerequisites
- **Figure preparation**
 - ❑ Plotting with M8R
 - ❑ Plotting with Python
 - ❑ Plotting with Matlab/Octave
 - ❑ Plotting with GMT
- SEGTeX
- Alternative $\text{\LaTeX}$ 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()
```

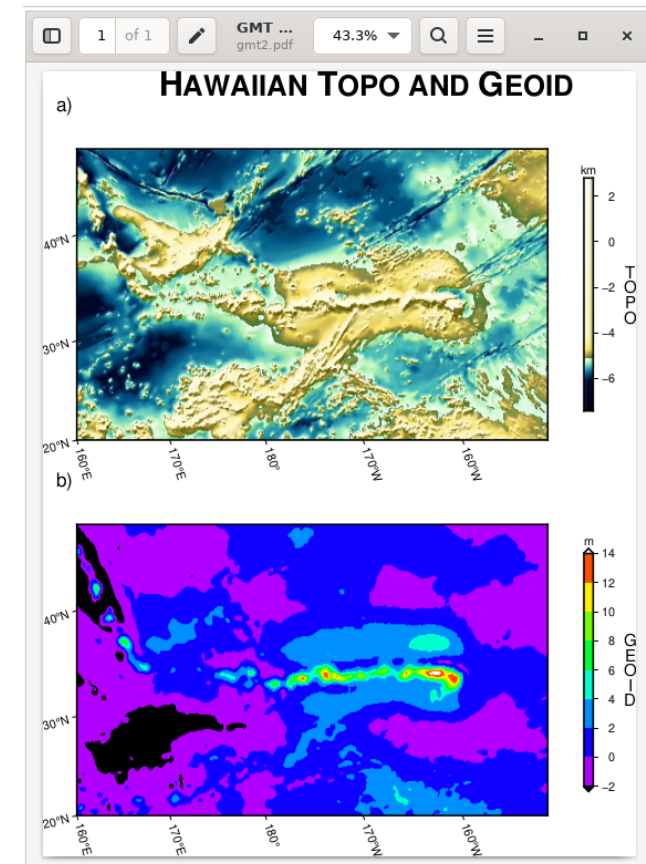
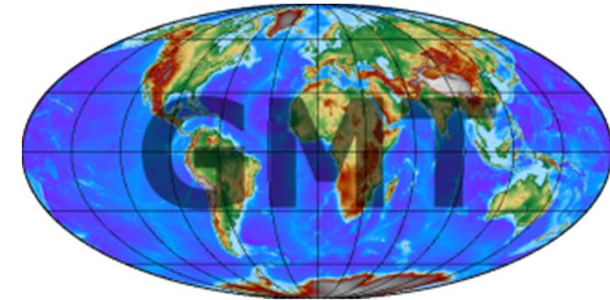
(2) Image presentations

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 02 Fig/gmt1
gmt set MAP_ANNOT_OBLIQUE separate
gmt subplot begin 2x1 -A+JTL -Fs16c/9c -M0 -R160/20/220/30+r -J0c190/25.5/292/69/16c -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+1TOPO -By+1km

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+1GEOID -By+1m
gmt subplot end
gmt end show
```

<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()
```

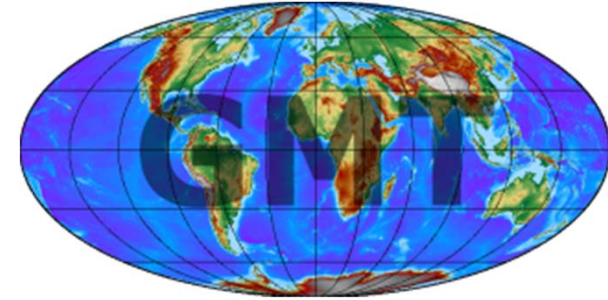


Illustration Gallery / (13) Plotting of vector fields

Edit on GitHub

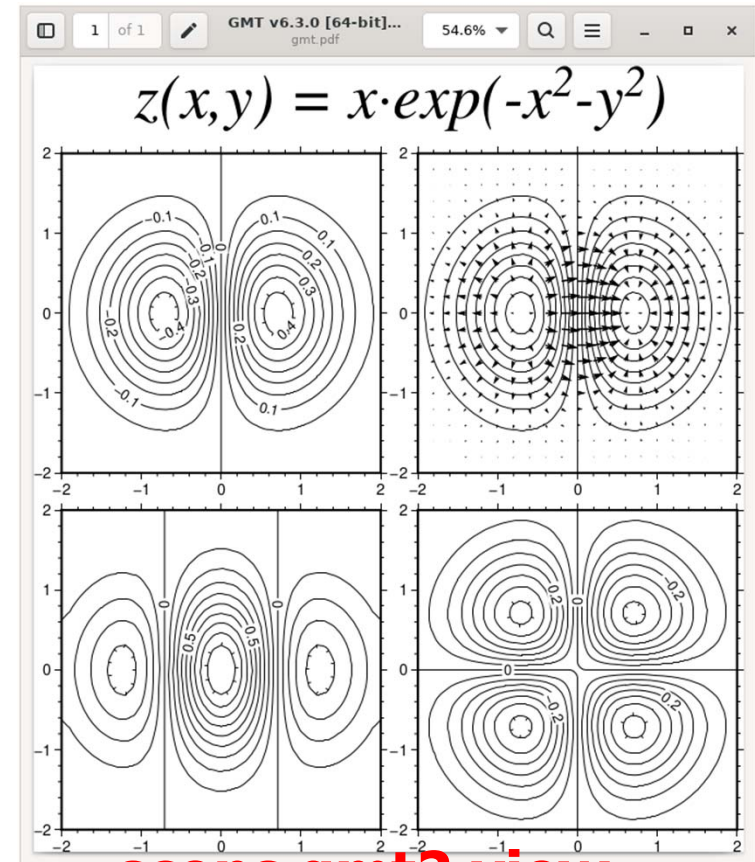
(13) Plotting of vector fields

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@-1327@~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 sheet  
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



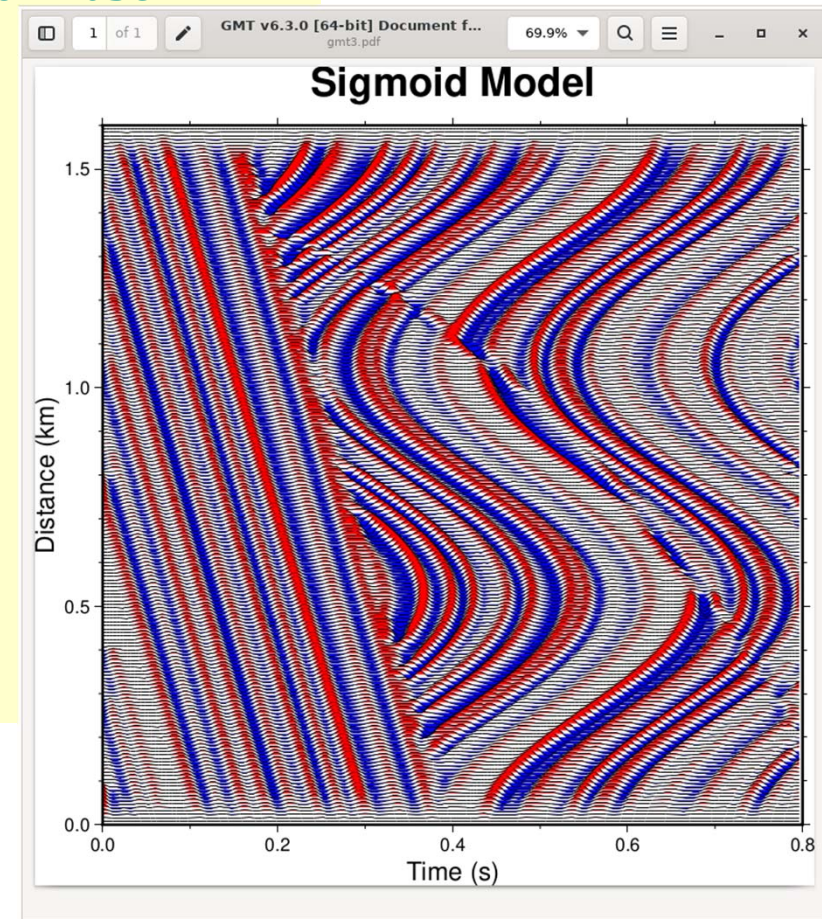
```
# GMT Wiggle
```

```
## 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.

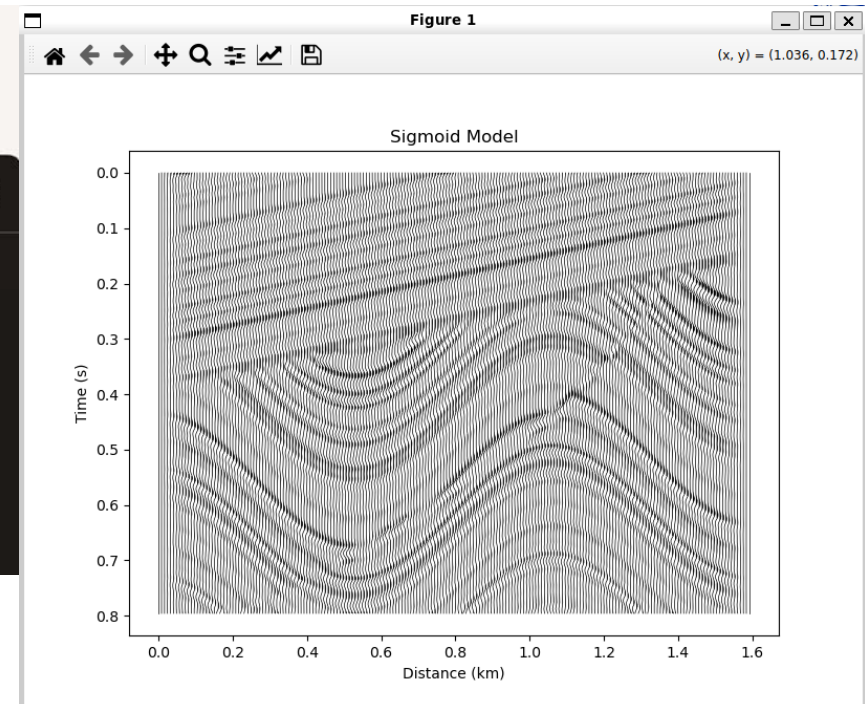
Python

复制

```
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.**



- Motivations
- Prerequisites
- Figure preparation
 - ▣ Plotting with M8R
 - ▣ Plotting with Python
 - ▣ Plotting with Matlab/Octave
 - ▣ Plotting with GMT
- **SEGT_EX**
- Alternative L^AT_EX template
- Simple Example

Implement steps



1) Ideas

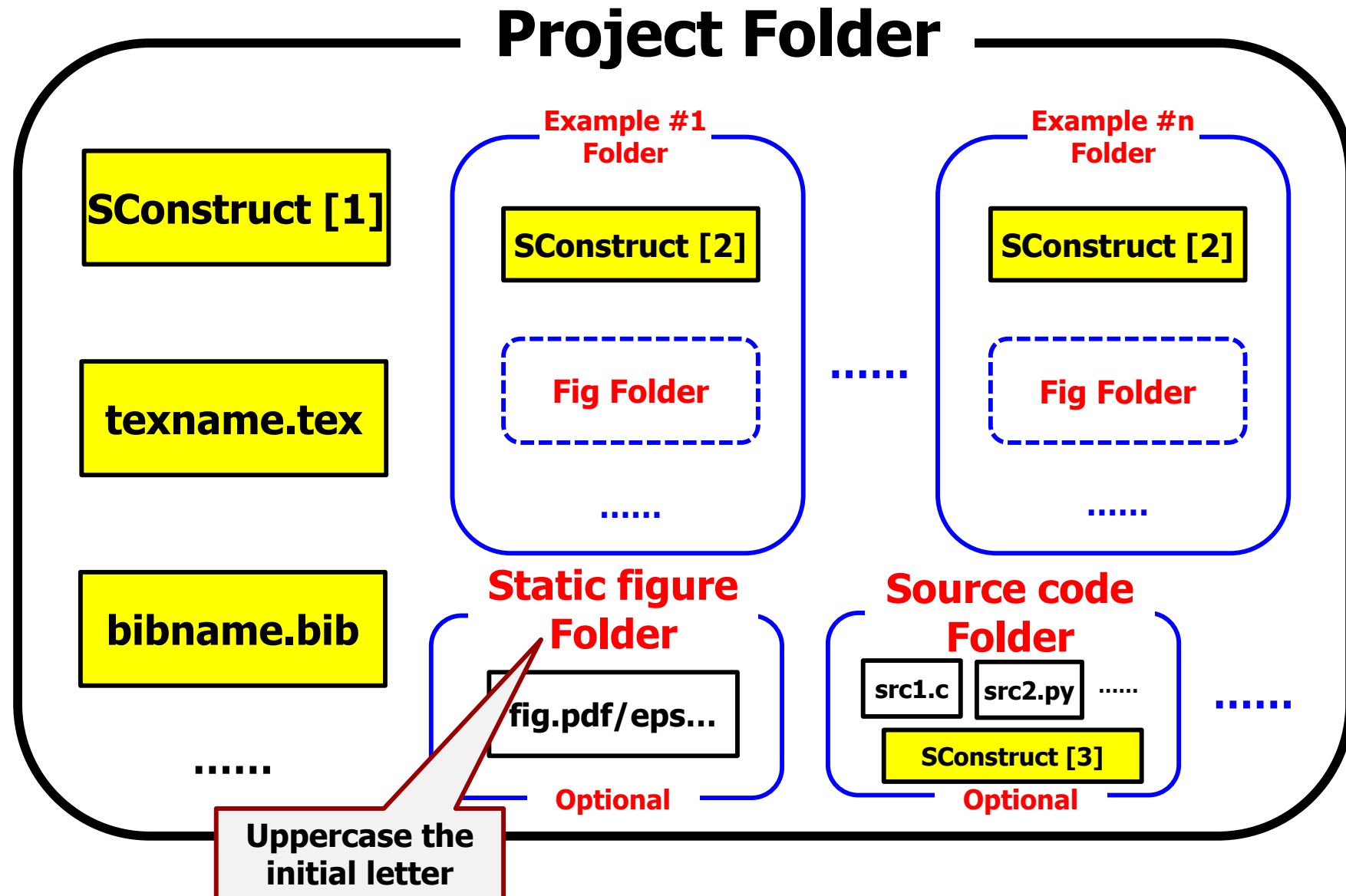


**2) Implement ideas in Madagascar and lock Figures
(scons **fig.lock**)**

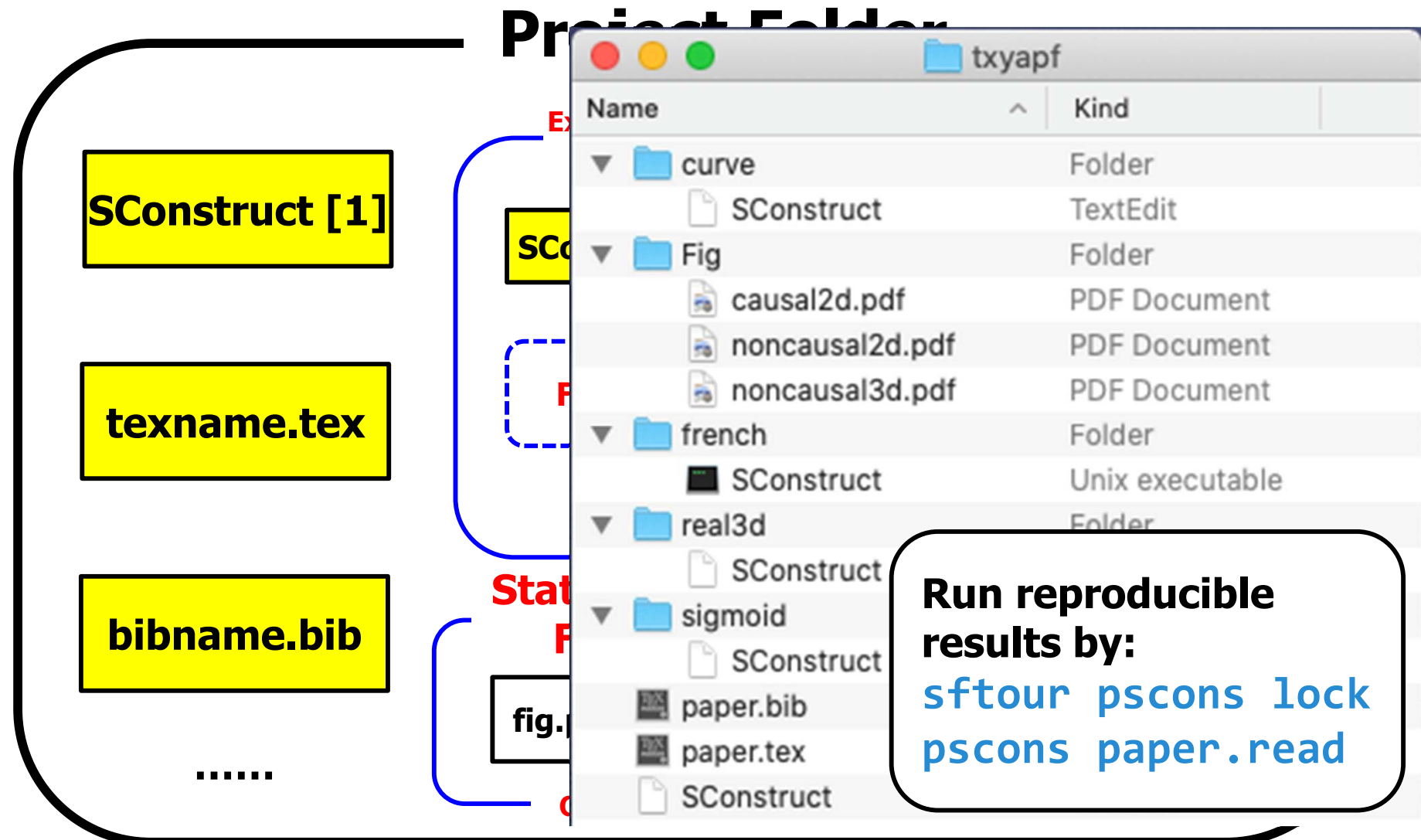
3) Write a paper text by following SEGTeX rules

4) Insert Figures into paper by following SEGTeX rules

Rules for project structure



Rules for project structure



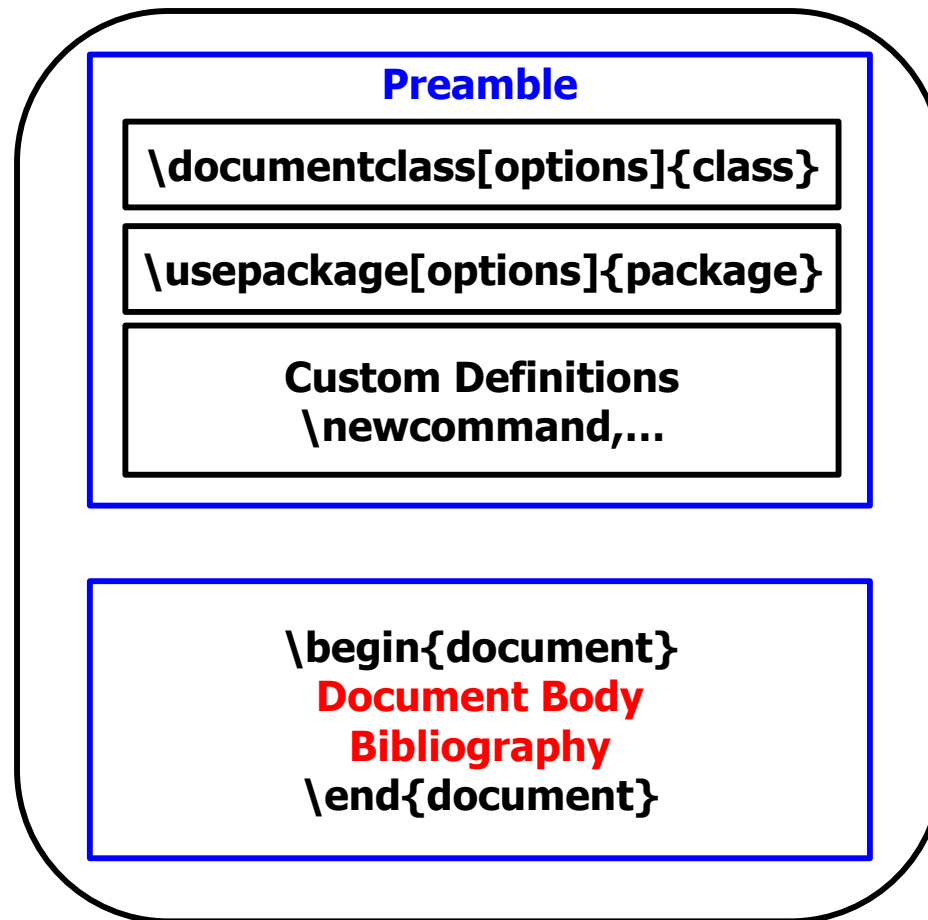
Look inside example: `$RSFSRC/book/jlu/txsopf`

Rules for the tex file (structure)

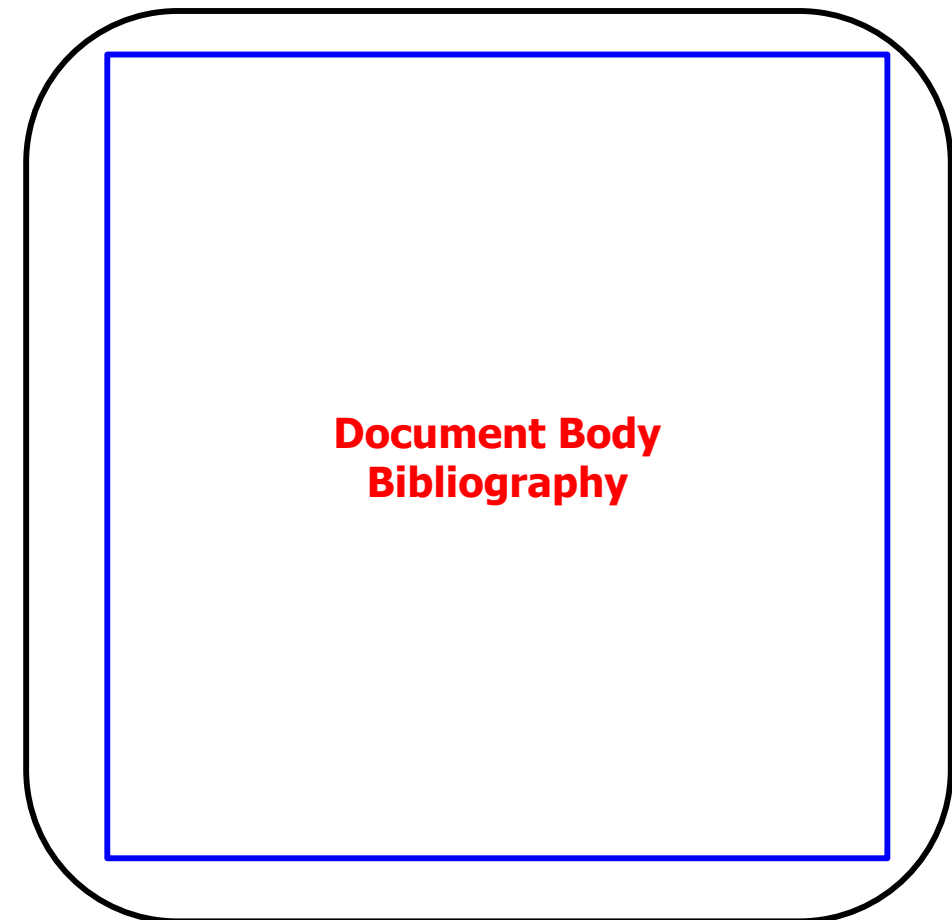


Similar to common $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ format, but with a few differences

$\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ tex file



Madagascar tex file



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='')
```

Diagram illustrating the configuration of the `Paper` function:

- `texname` (points to `'texname'`)
- `Document class` (points to `'geophysics'`)
- `options` (points to `'manuscript, revised'`)
- `Used packages \usepackage` (points to the list of packages: `amsmath empheq graphicx float threeparttable multirow algorithm [caption=false] subfig`)
- `Custom definitions` (points to `include=''`)

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

`\usepackage[caption=false]{subfig}`



SEG Abstract:

```
lclass='segabs'
```

GEOPHYSICS:

```
lclass='geophysics'
```

options='manuscript'

EAGE Abstract:

```
lclass='name'
```

Two-column article:

```
lclass='geophysics'
```

```
options='twoside,twocolumn'
```

SEP Report:

no need

class and options

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

SEGTeX

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

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('wiggles', 'Txx.HH', 'wiggles')
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 ('wiggles', '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}
```

2) Sections, citations and equations

`\section{section name, e.g., Introduction}` (Refname1, year)

The paper starts from old idea `\cite[]{}{refname1}`.

`\cite{}{refname2}` also improve it. ...

Refname2 (year)

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

`\subsection{subsection name}`

The theory is shown as follows ...

`\begin{equation}`

Write equation label here

`\label{eq:eq1}`

`\mathbf{C} = \frac{\mathbf{A}}{b} \ ;,`

`\end{equation}`

Refer to the equation by its label

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

3) Figures and their references

```
\section{section name, e.g., Synthetic examples}
```

The paper has several synthetic tests.

```
\inputdir{example#1 folder name}
```

Declare figure folder first

```
\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 ...
```

Figure 1a

```
\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.}
```

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}
```

- **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 .

- 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}}}
```



- 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}  
}
```

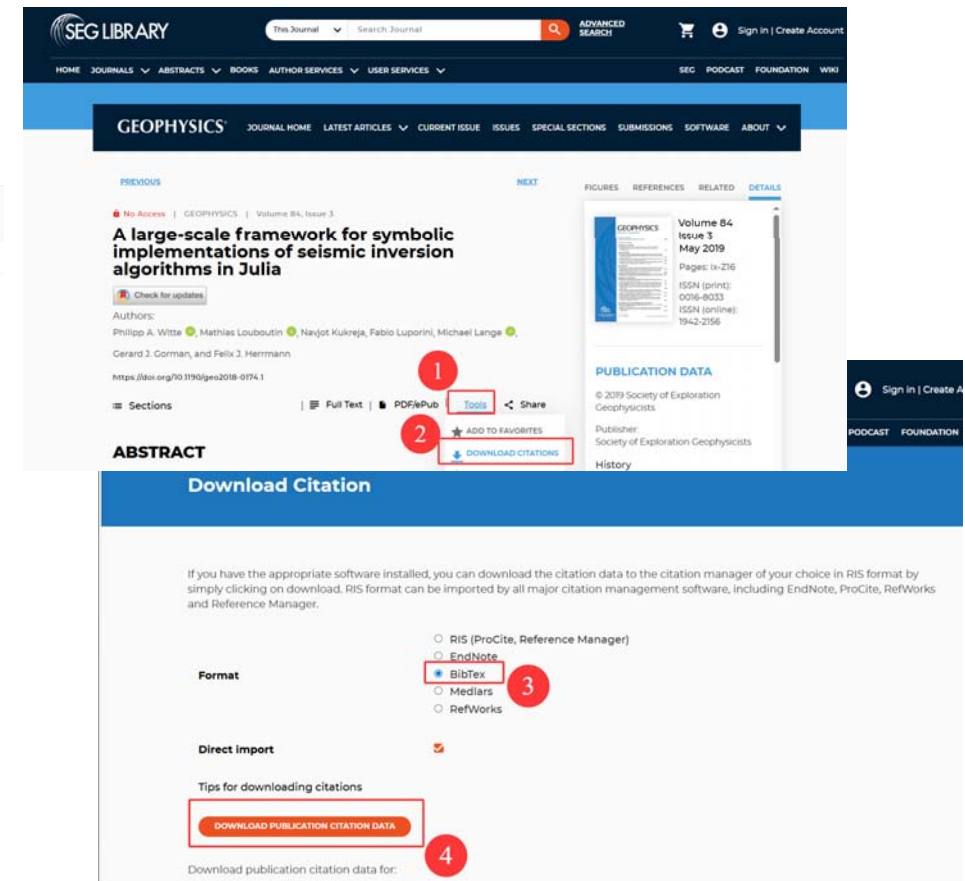
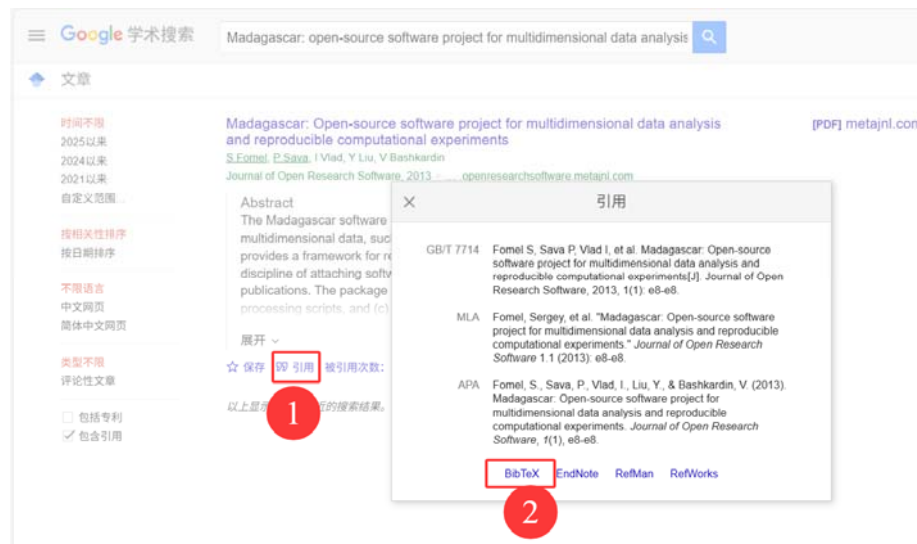
```
@inproceedings{Curry04,  
  author    = {W[illiam] Curry},  
  title     = {Interpolation with multi-shifted-scale  
prediction-error filters},  
  booktitle = {74th Annual International Meeting},  
  year      = {2004},  
  publisher = {SEG, Expanded Abstracts},  
  pages     = {2005-2008}  
}
```

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')
```

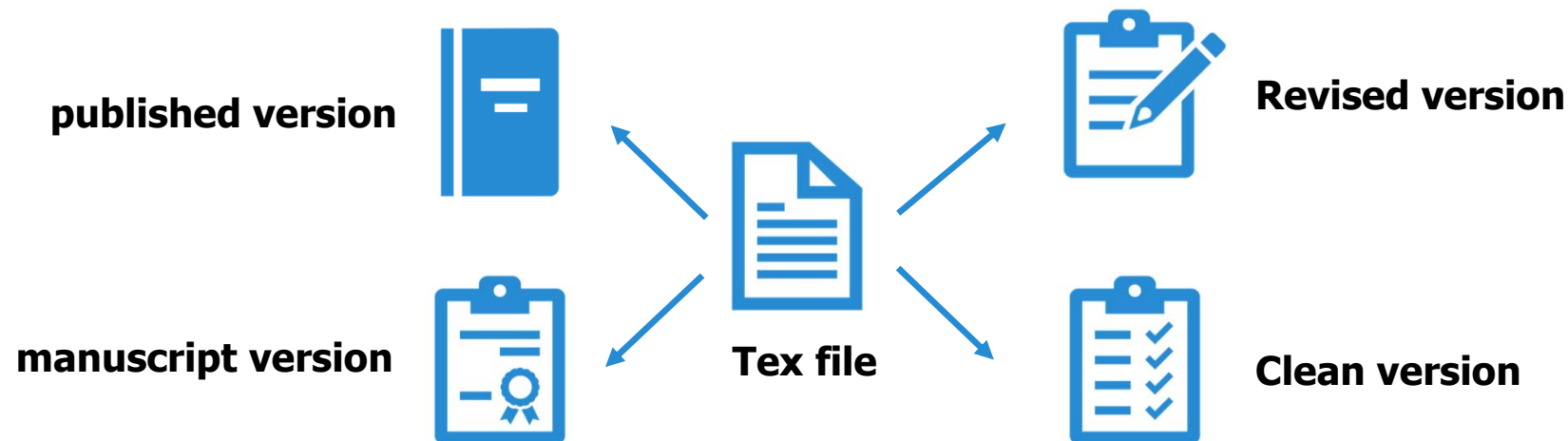
4) Insert only grey figures in LaTeX (SConstruct [1]):

```
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 ('wiggles', 'contour', 'grey'):  
    Result(plot, 'windowed', plot + plotpar)
```

```
f'{plot} {plotpar}'
```

```
'%s %s'%(plot, plotpar)
```



- Motivations
- Prerequisites
- Figure preparation
 - ▣ Plotting with M8R
 - ▣ Plotting with Python
 - ▣ Plotting with Matlab/Octave
 - ▣ Plotting with GMT
- SEGTeX
- **Alternative $\text{\LaTeX}$ template**
- Simple Example

How to use alternative templates

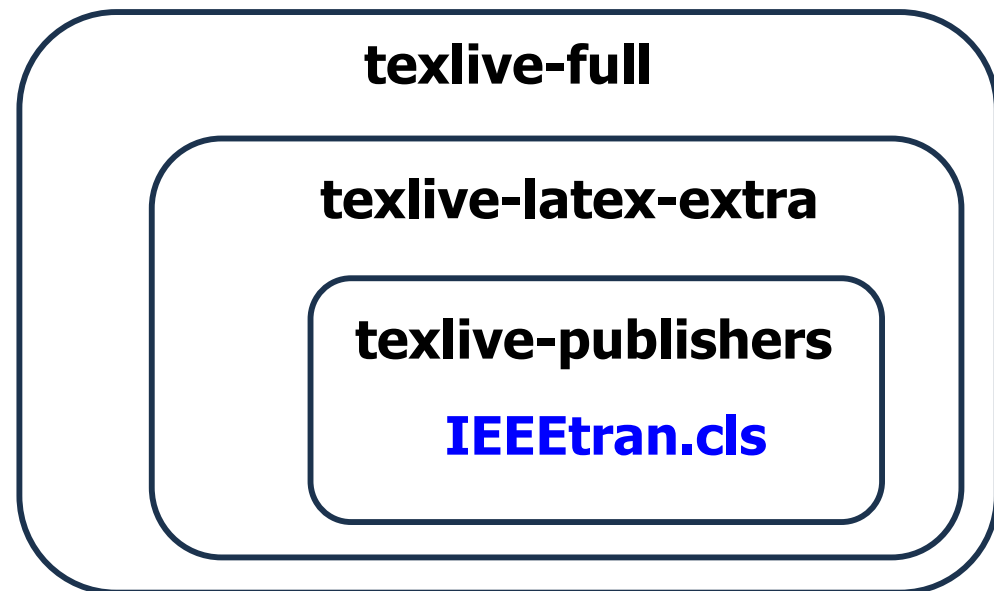


- 1) Download a LaTeX template file and corresponding class file.
- 2) Open the template file. According to the preamble contents in template .tex file, configure `Paper()` in SConstruct [1] following its **rules**.
- 3) Remove all preamble contents (before `\begin{document}`) including `\begin{document}` 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 L^AT_EX 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}}''')
```

```
1 \documentclass[journal,twoside]{IEEEtran}  
2 \usepackage[amsmath,amsfonts]  
3 \usepackage[algorithmic]  
4 \usepackage[array]  
5 \usepackage[caption=false,font=normalsize,labelfont=sf,textfont=sf]  
6 \subfig  
7 \usepackage{textcomp}  
8 \usepackage{stfloats}  
9 \usepackage{url}  
10 \usepackage{verbatim}  
11 \usepackage{graphicx}  
12 \hyphenation{op-tical net-works semi-conduc-tor IEEE-Xplore}  
13 \def\BibTeX{{\rm B\kern-.05em{\sc i\kern-.025em b}\kern-.08em  
14 | T\kern-.1667em\lower.7ex\hbox{E}\kern-.125emX}}  
15 \usepackage{balance}  
16 \begin{document}  
17 \title{How to Use the IEEEtran LATEX Templates}  
18 \author{IEEE Publication Technology Department  
19 \thanks{Manuscript created October, 2020; This work was developed  
20 by the IEEE Publication Technology Department. This work is  
distributed under the LATEX Project Public License (LPPL) (  
http://www.latex-project.org/) version 1.3. A copy of the LPPL,  
version 1.3, is included in the base LATEX documentation of all  
distributions of LATEX released 2003/12/01 or later. The  
opinions expressed here are entirely that of the author. No  
warranty is expressed or implied. User assumes all risk.}}
```

2

Configure Paper()

3

Remove preamble

4

scons school2025.read

JOURNAL OF L^AT_EX CLASS FILES, VOL. 18, NO. 9, SEP 2020

How to Use the IEEEtran L^AT_EX Templates

IEEE Publication Technology Department

Abstract—This document describes the most common article elements and how to use the IEEEtran class with L^AT_EX to produce files that are suitable for submission to the Institute of Electrical and Electronics Engineers (IEEE). IEEEtran can produce conference, journal and technical note (correspondence) papers with a suitable choice of class options.

Keywords: IEEEtran, L^AT_EX, paper, style, template,

I. INTRODUCTION

WELCOME to the updated and simplified documentation to using the IEEEtran L^AT_EX class file. The IEEE has examined hundreds of author submissions using this package to help formulate this easy to follow guide. We will cover the most commonly used elements of a journal article. For less common elements we will refer back to the "IEEE-

III. L^AT_EX DISTRIBUTIONS: WHERE TO GET THEM

IEEE recommends using the distribution from the T_EX User Group at <http://www.tug.org>. You can join TUG and obtain a DVD distribution or download for free from the links provided on their website: <http://www.tug.org/texlive/>. The DVD includes distributions for Windows, Mac OS X and Linux operating systems.

IV. WHERE TO GET THE IEEETRAN TEMPLATES

The **IEEE Template Selector** will always have the most up-to-date versions of the L^AT_EX and MSWord templates. Please see: <https://template-selector.ieee.org/> and follow the steps to find the correct template for your intended publication. Many publications use the IEEEtran L^AT_EX templates, however,



- Motivations
- Prerequisites
- Figure preparation
 - ▣ Plotting with M8R
 - ▣ Plotting with Python
 - ▣ Plotting with Matlab/Octave
 - ▣ Plotting with GMT
- SEGTeX
- Alternative $\text{\LaTeX}$ template
- 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



Madagascar L^AT_EX and Web tools

Sergey Fomel

Bureau of Economic Geology
Jackson School of Geosciences
University of Texas at Austin

August 30, 2006

(Fomel, School, 2006)

S. Fomel (UT Austin)

L^AT_EX and Web

Vancouver School & Workshop

1 / 17

Madagascar School for Advanced Users in Qingdao 2015



Reproducible Paper:

Writing and publishing using L^AT_EX and Madagascar



Yang Liu

College of Geo-exploration Sciences and Technology

Jilin University

yangliu1979@jlu.edu.cn

Aug 8, 2015

(Liu, School, 2015)

1

Writing a paper using Madagascar

Tariq Alkhalifah

Seismic Analysis Group (SWAG)
KAUST
swag.kaust.edu.sa

(Alkhalifah, School, 2011)



Madagascar School
of Reproducible Computational
Geophysics Shanghai 2017



同济大学
TONGJI UNIVERSITY

Writing a Reproducible Paper using LaTeX and Madagascar

Jiubing Cheng & Yang Liu



(Cheng and Liu, School, 2017)

References



下载
Installation
GitHub repository
SEGTex

Introduction
Package overview
Tutorial
Hands-on tour
Reproducible
documents
Hall of Fame

User Documentation
List of programs
Common programs
Popular programs
The RSE file format
Reproducibility with
SCons

Developer
documentation
Adding programs
Contributing programs
API demo: clipping
data
API demo: explicit
finite differences

Community
Conferences
User mailing list
Developer mailing list
GitHub organization
LinkedIn group
Development blog
Twitter
Slack

工具
输入页面
相关更改
特殊页面
打印版本
固定链接
页面信息

首页 讨论

Main Page

Madagascar is an open-source software package for multidimensional data analysis and reproducible computational experiments. Its mission is to provide

- a convenient and powerful environment
- a convenient technology transfer tool

for researchers working with digital image and data processing recipes" to be verified, exchanged, and modified by the users.

Features [编辑]

Madagascar is a **modern** package. It started in 2003 and driven development. The rapid growth of a project of the packages such as SEPIlib and Seismic Unix. We have borrowed

Madagascar is a **test-driven** package. Test-driven development peer review, the backbone of any real science, to computing as data filters) and high-level processing flows (describing programming is easily mastered even by beginning students v

Madagascar is an **open-source** package. It is distributed but shared equally among developers. Sharing the responsibility

Madagascar uses a **simple, flexible, and universal** data format allows us to share general-purpose data processing

Latest News [编辑]

Effects of lateral heterogeneity on time-domain processing
An old paper is added to the collection of reproducible documents
2024-12-05 20:54:15

Well log interpolation guided by geologic distance
An old paper is added to the collection of reproducible documents
2024-12-05 20:52:35

Estimation of timeshifts in time-lapse seismic images using
An old paper is added to the collection of reproducible documents
2024-12-05 20:51:13

Least-squares non-stationary triangle smoothing
An old paper is added to the collection of reproducible documents
2024-12-05 20:49:26

Investigating the possibility of locating microseismic sources
An old paper is added to the collection of reproducible documents
2024-12-05 20:45:56

Using well-seismic mistle to update the velocity model
An old paper is added to the collection of reproducible documents
2024-12-05 20:44:26

Full waveform inversion of passive seismic data for source
An old paper is added to the collection of reproducible documents
2024-12-05 20:42:44



下载
Installation
GitHub repository
SEGTex

Introduction
Package overview
Tutorial
Hands-on tour
Reproducible
documents
Hall of Fame

User Documentation
List of programs
Common programs
Popular programs
The RSE file format
Reproducibility with
SCons

Developer
documentation
Adding programs
Contributing programs
API demo: clipping
data
API demo: explicit
finite differences

Community
Conferences
User mailing list
Developer mailing list
GitHub organization
LinkedIn group
Development blog
Twitter
Slack

工具
输入页面
相关更改
特殊页面
打印版本
固定链接
页面信息

页面 讨论

Conferences

In reverse chronological order.

目录 [编辑]

- 1 Schools, Workshops, and Working Workshops
- 2 Conference presentations
 - 2.1 Online 2021 (SPE)
 - 2.2 Lausanne 2016 (PASC)
 - 2.3 San Francisco 2015 (AGU)
 - 2.4 Stanford 2015 (SIAM Geosciences)
 - 2.5 New York 2013 (NYU Poly)
 - 2.6 Providence 2012 (ICERM)
 - 2.7 Austin 2012 (SciPy)
 - 2.8 Cary 2011 (Interface)
 - 2.9 Long Beach 2011 (SIAM Geosciences)
 - 2.10 Reno 2011 (SIAM CS&E)
 - 2.11 Austin 2010 (SciPy)
 - 2.12 Salt Lake City 2010 (NSF Archive Workshop)
 - 2.13 Düsseldorf 2008 (Berlin 6)
 - 2.14 Austin 2008 (Texas Python Unconference)
 - 2.15 Austin 2008 (Scientific Software Days)
 - 2.16 Rio de Janeiro 2007 (SBGI)
 - 2.17 Vancouver 2007 (AIP)
 - 2.18 Honolulu 2007 (ICASSP)
 - 2.19 Austin 2007 (Scientific Software Day)
 - 2.20 New Orleans 2006 (SEG)
 - 2.21 Vienna 2006 (EAGE)

Schools, Workshops, and Working Workshops [编辑]

Madagascar Workshop at RAUGM, Puerto Vallarta 2023
Madagascar School on Open-Source Geophysics in Hefei 2020
SEG Working Workshop - Python and Julia for Geophysics, Houston August 8-11, 2018
SEG Working Workshop in Houston 2017
Madagascar School of Reproducible Computational Geophysics Shanghai 2017
Madagascar School at the University of Houston 2017
SEG Land 3D Seismic Processing Working Workshop Houston 2016
Madagascar School on Reproducible Computational Geophysics in Zürich 2016
Workshop - Open-source Software in Applied Geosciences Vienna 2016
Madagascar School for Advanced Users in Qingdao 2015
SEG 3D Land Seismic Processing Working Workshop Houston 2015
School in Harbin 2015
Second Madagascar Working Workshop Houston 2014
Workshop at EAGE/St. Petersburg 2014



中文 (中国大陆) 未登录 讨论 贡献 创建账号 登录

阅读 编辑 查看历史 搜索Madagascar内容

<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>