Introduction to Madagascar

Prof. Daniel Leal Macedo

Segundo Curso de Inverno do Observatório Sismológico - UnB

6 e 7 de Junho de 2017

MADAGASCAR

Open-source software package for geophysical data processing and reproducible numerical experiments.

MADAGASCAR

Open-source software package for geophysical data processing and reproducible numerical experiments.

- Standalone programs (data analysis, processing and imaging);
- A development kit (C, Fortran, Python, Matlab, Octave...);
- A framework for reproducible numerical experiments (SCons);
- A framework for scientific publications (SCons and LaTeX);
- A collection of reproducible scientific articles;
- A collection of datasets.

MADAGASCAR

Open-source software package for geophysical data processing and reproducible numerical experiments.

- Standalone programs (data analysis, processing and imaging);
- A development kit (C, Fortran, Python, Matlab, Octave...);
- A framework for reproducible numerical experiments (SCons);
- A framework for scientific publications (SCons and LaTeX);
- A collection of reproducible scientific articles;
- A collection of datasets.

MADAGASCAR

Open-source software package for geophysical data processing and reproducible numerical experiments.

- Standalone programs (data analysis, processing and imaging);
- A development kit (C, Fortran, Python, Matlab, Octave...);
- A framework for reproducible numerical experiments (SCons);
- A framework for scientific publications (SCons and LaTeX);
- A collection of reproducible scientific articles;
- A collection of datasets.

MADAGASCAR

Open-source software package for geophysical data processing and reproducible numerical experiments.

- Standalone programs (data analysis, processing and imaging);
- A development kit (C, Fortran, Python, Matlab, Octave...);
- A framework for reproducible numerical experiments (SCons);
- A framework for scientific publications (SCons and LaTeX);
- A collection of reproducible scientific articles;
- A collection of datasets.

MADAGASCAR

Open-source software package for geophysical data processing and reproducible numerical experiments.

- Standalone programs (data analysis, processing and imaging);
- A development kit (C, Fortran, Python, Matlab, Octave...);
- A framework for reproducible numerical experiments (SCons);
- A framework for scientific publications (SCons and LaTeX);
- A collection of reproducible scientific articles;
- A collection of datasets.

MADAGASCAR

Open-source software package for geophysical data processing and reproducible numerical experiments.

- Standalone programs (data analysis, processing and imaging);
- A development kit (C, Fortran, Python, Matlab, Octave...);
- A framework for reproducible numerical experiments (SCons);
- A framework for scientific publications (SCons and LaTeX);
- A collection of reproducible scientific articles;
- A collection of datasets.

MADAGASCAR

Open-source software package for geophysical data processing and reproducible numerical experiments.

- Standalone programs (data analysis, processing and imaging);
- A development kit (C, Fortran, Python, Matlab, Octave...);
- A framework for reproducible numerical experiments (SCons);
- A framework for scientific publications (SCons and LaTeX);
- A collection of reproducible scientific articles;
- A collection of datasets.

MADAGASCAR

Open-source software package for geophysical data processing and reproducible numerical experiments.

- Standalone programs (data analysis, processing and imaging);
- A development kit (C, Fortran, Python, Matlab, Octave...);
- A framework for reproducible numerical experiments (SCons);
- A framework for scientific publications (SCons and LaTeX);
- A collection of reproducible scientific articles;
- A collection of datasets.

Madagascar Community

1. Home site (www.ahay.org).
2. Mailing list (https://lists.sourceforge.net/lists/listinfo/rsf-user).
3. Development blog (http://ahay.org/blog/).

Schedule

1. Introduction to Madagascar; command lines.
2. Ploting; scripting with Scons.
3. Modeling with Madagascar.
4. Reading / Writing SEG-Y. Reading / Writing to ASCII.
5. Simple processing workflow with Madagascar.

Introduction to Madagascar

1. command line usage
    - programs
    - file format
2. plotting

Madagascar programs

- 'sf' prefix
- $>$ 1000 programs
- Developed using C, C++, Fortran, Python...
- Applications
  - general data analysis
  - Seismic modeling, processing and imaging
  - visualization
- Documentation based on examples
  - self-documentation
  - on-line documentation

Madagascar programs

List of all programs

sfdoc -k .

Madagascar programs

List of all programs

sfdoc -k .

```
bash$ sfdoc -k .
sfwave: Rice HPCSS seismic modeling and migration.
sferf: Bandpass filtering using erf function.
sfinfill: Shot interpolation.
sfslice: Extract a slice using picked surface (usually from a stack or
a semblance).
sfin: Display basic information about RSF files.
sfdmo: Kirchhoff DMO with antialiasing by reparameterization.
sfic: Imaging condition
sfradstretch: Stretch of the time axis.
sflpef: Find PEF on aliased traces.
sfmul: Add, multiply, or divide RSF datasets.
sfrefer: Subtract a reference from a grid.
sfvplotdiff: Vplot diff - see if 2 vplot files represent "identical"plots.
sflevint: Leveler inverse interpolation in 1-D.
sflorenz: Generate Lorenz attractor.
sfnoise: Add random noise to the data.
sfScanCoef: Coeffecients of the eta expansion eikonal solver (3-D).
```

Madagascar programs

List of all programs

sfdoc -k .

List of specific programs

sfdoc -k keyword

Madagascar programs

List of all programs

sfdoc -k .

List of specific programs

sfdoc -k keyword

```
bash$ sfdoc -k inversion
sfvelinvww: Inverse velocity spectrum with interpolation by modeling from
inversion result
sfcgscan: Hyperbolic Radon transform with conjugate-directions inversion
sfdeblur: Non-stationary debluring by inversion
sfconjgrad: Generic conjugate-gradient solver for linear inversion
sfcconjgrad: Generic conjugate-gradient solver for linear inversion with
complex data
sfimospray: Inversion of constant-velocity nearest-neighbor inverse NMO.
```

Madagascar programs

Self-documetation

sfprog no arguments

```
bash$ sfawefd2d
NAME
        sfawefd2d
DESCRIPTION
        2D acoustic time-domain FD modeling.
SYNOPSIS
        sfawefd2d < Fwav.rsf vel=Fvel.rsf sou=Fsou.rsf rec=Frec.rsf wfl=Fwfl.rsf
> Fdat.rsf den=Fden.rsf
verb=n snap=n free=n expl=n dabc=n jdata=1 jsnap=nt nqz=sf_n(az) nqx=sf_n(ax)
oqz=sf_o(az) oqx=sf_o(ax)

COMMENTS
        4th order in space, 2nd order in time. Absorbing boundary conditions

PARAMETERS
        bool    dabc=n [y/n]    absorbing BC
        file    den=    auxiliary input file name
        bool    expl=n [y/n]    "exploding reflector"
        bool    free=n [y/n]    free surface flag
        int     jdata=1
        int     jsnap=nt
        int     nqx=sf_n(ax)
        int     nqz=sf_n(az)
        float   oqx=sf_o(ax)
```

Madagascar programs

Self-documetation

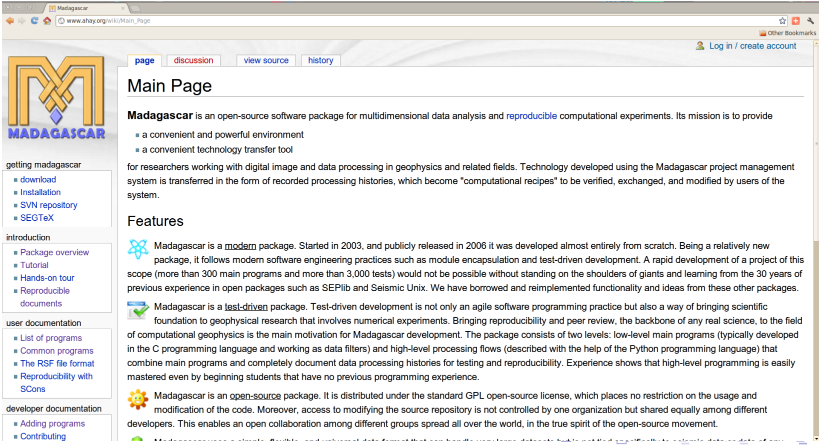
sfprog no arguments

```
...
        file    rec=    auxiliary input file name
        bool    snap=n [y/n]    wavefield snapshots flag
        file    sou=    auxiliary input file name
        file    vel=    auxiliary input file name
        bool    verb=n [y/n]    verbosity flag
        file    wfl=    auxiliary output file name
USED IN
        cwp/geo2007StereographicImagingCondition/flat4
        cwp/geo2007StereographicImagingCondition/gaus1
        cwp/geo2008InterferometricImagingCondition/circle
        cwp/geo2008InterferometricImagingCondition/sact1
        cwp/geo2008IsotropicAngleDomainElasticRTM/marm2oneA
        cwp/geo2011WideAzimuthAngleDecomposition/flatEICangle
        cwp/jse2006RWEImagingOverturningReflections/sigsbee
        cwp/pept2011MicroearthquakeMonitoring/saf1
        cwp/pept2011MicroearthquakeMonitoring/saf2
        cwp/pept2011MicroearthquakeMonitoring/saf3
        data/amoco/fdmod
        data/marmousi/fdmod
        data/marmousi2/fdMod
        data/pluto/fdmod
        data/sigsbee/fdmod2A
...
```

Madagascar programs

on-line documetation

`http://www.ahay.org`



Madagascar
www.ahay.org/wiki/Main_Page
Other Bookmarks
Log in / create account

MADAGASCAR

getting madagascar
- download
- Installation
- SVN 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

page | discussion | view source | history

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 in geophysics and related fields. Technology developed using the Madagascar project management system is transferred in the form of recorded processing histories, which become "computational recipes" to be verified, exchanged, and modified by users of the system.

Features

Madagascar is a modern package. Started in 2003, and publicly released in 2006 it was developed almost entirely from scratch. Being a relatively new package, it follows modern software engineering practices such as module encapsulation and test-driven development. A rapid development of a project of this scope (more than 300 main programs and more than 3,000 tests) would not be possible without standing on the shoulders of giants and learning from the 30 years of previous experience in open packages such as SEPlib and Seismic Unix. We have borrowed and reimplemented functionality and ideas from these other packages.

Madagascar is a test-driven package. Test-driven development is not only an agile software programming practice but also a way of bringing scientific foundation to geophysical research that involves numerical experiments. Bringing reproducibility and peer review, the backbone of any real science, to the field of computational geophysics is the main motivation for Madagascar development. The package consists of two levels: low-level main programs (typically developed in the C programming language and working as data filters) and high-level processing flows (described with the help of the Python programming language) that combine main programs and completely document data processing histories for testing and reproducibility. Experience shows that high-level programming is easily mastered even by beginning students that have no previous programming experience.

Madagascar is an open-source package. It is distributed under the standard GPL open-source license, which places no restriction on the usage and modification of the code. Moreover, access to modifying the source repository is not controlled by one organization but shared equally among different developers. This enables an open collaboration among different groups spread all over the world, in the true spirit of the open-source movement.

Command-line usage

Single Program

```
[< in.rsf] sfprog [par1=] [par2=] [...] [> out.rsf]
```

- Single input: `<in.rsf`
- Single output: `>out.rsf`
- Multiple parameters: `par=val`

multiple Programs

```
[< in.rsf] sfprog1 [par=] | ... | sfprogn [par=] [> out.rsf]
```

- ONE task per program
- Data passed through pipes

Command-line usage - Example

```
bash$ sfspike
bash$ sfspike n1=5 k1=2 > a.rsf
```

- standard in: none
- standard out: a.rsf

```
bash$ ls
```

```
a.rsf g.asc
```

```
bash$ sfdisfil < a.rsf
```

```
0:         0       1       0       0       0
```

- standard in: a.rsf
- standard out: screen

Command-line usage - Example

```
bash$ sfmath
bash$ sfspike n1=5 k1=2 | sfmath output="1-input" | sfdisfil
```

```
0:        1       0       1       1       1
```

```
bash$ sfspike n1=5 k1=4 > b.rsf
bash$ < a.rsf sfadd scale=1,-2 b.rsf > c.rsf
bash$ sfdisfil < c.rsf
```

```
0:        0       1       0      -2       0
```

Regularly Sampled Format

To design a perfect anti-Unix, make all file formats binary and opaque, and require heavyweight tools to read and edit them.

If you feel an urge to design a complex binary file format, or a complex binary application protocol, it is generally wise to lie down until the feeling passes.

Regularly Sampled Format

1. Discrete representation of n-d functions
2. Uniform sampling
3. RSF dataset is n-d matrices with physical dimensions
4. Data type `int`, `float`, `double`, `complex` ....

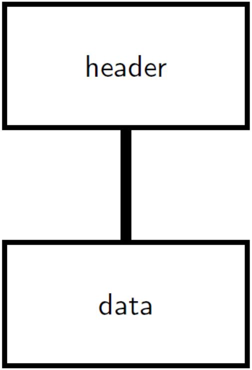
RSF componets

Header file

- Text
- Small
- Portable

Data file

- ASCII or binary (native or XDR)
- Large (Huge)
- Path under `$DATAPATH`



Header information

Example: construct Matrix

$$D = \begin{bmatrix} 1 & 2 \\ 2 & 4 \\ 3 & 6 \end{bmatrix}$$

```
bash$ sfmath n1=3 o1=1 n2=2 o2=1 output="x1*x2" > d.rsf
bash$ < d.rsf sfdisfil
```

```
  0:         1         2         3         2         4
  3:         6
```

```
bash$ sfmath n1=3 o1=1 n2=2 o2=1 output="x1*x2" > d.rsf
bash$ < d.rsf sfdisfil col=3
```

```
  0:         1         2         3
  3:         2         4         6
```

Header information

Print out header

```
sfin file0.rsf [file1.rsf] [file2.rsf] ...
```

```
bash$ sfin d.rsf
```

```
d.rsf:
    in="/home/dlmacbr/rsfdata/d.rsf@"
    esize=4 type=float form=native
    n1=3            d1=1            o1=1
    n2=2            d2=1            o2=1
6 elements 24 bytes
```

- n: number of samples
- o: origin of samples
- d: sampling interval
- label: axis label
- unit: axis unit

```
bash$ ls -l /home/dlmacbr/rsfdata/d.rsf*
```

```
-rwxrwx--- 1 root daniel 24 2012-11-14 02:05 /home/dlmacbr/rsfdata/d.rsf@
```

```
bash$ echo $DATAPATH
```

RSF dataset attributes

Print out attributes

```
sfattr <  file.rsf
```

```
bash$ sfattr < d.rsf
```

```
*******************************************
     rms =        3.41565
    mean =              3
  2-norm =         8.3666
variance =            3.2
 std dev =        1.78885
     max =              6 at 3 2
     min =              1 at 1 1
nonzero samples = 6
  total samples = 6
*******************************************
```

Modify header

write header

```
sfput < in.rsf key1=val1 [...] > out.rsf
```

```
bash$ sfin d.rsf
```

```
d.rsf:
    in="/home/dlmacbr/rsfdata/d.rsf@"
    esize=4 type=float form=native
    n1=3           d1=1           o1=1
    n2=2           d2=1           o2=1
6 elements 24 bytes
```

```
bash$ < d.rsf sfput n1=6 n2=1 > d2.rsf

bash$ < sfin d2.rsf
```

```
d2.rsf:
    in="/home/dlmacbr/rsfdata/d2.rsf@"
    esize=4 type=float form=native
    n1=6           d1=1           o1=1
    n2=1           d2=1           o2=1
6 elements 24 bytes
```

Modify header

```
bash$ sfmath n1=3 o1=1 n2=2 o2=1 n3=2 o3=1 output="x1*100 +
x2*10 + x3" > e.rsf
bash$ sfdisfil < e.rsf col=3
```

```
0:        111        211        311
3:        121        221        321
6:        112        212        312
9:        122        222        322
```

Moving RSF dataset

mv moves header ONLY

```
bash$ mv d.rsf f.rsf
bash$ sfin f.rsf
```

```
f.rsf:
    in="/home/dlmacbr/rsfdata/d.rsf@"
    esize=4 type=float form=native
    n1=3           d1=1           o1=1
    n2=2           d2=1           o2=1
6 elements 24 bytes
```

```
bash$ ls d.rsf
```

```
ls: cannot access d.rsf: No such file or directory
```

Moving RSF dataset

Move header and data

```
sfmv in.rsf out.rsf
```

```
bash$ mv f.rsf d.rsf
bash$ sfmv d.rsf f.rsf
bash$ sfin f.rsf
```

```
f.rsf:
    in="/home/dlmacbr/rsfdata/f.rsf@"
    esize=4 type=float form=native
    n1=3           d1=1           o1=1
    n2=2           d2=1           o2=1
6 elements 24 bytes
```

Copying and deleting RSF

Copy header and data

```
sfcp in.rsf out.rsf
```

```
bash$ sfcp a.rsf b.rsf
```

```
b.rsf:
    in="/home/dlmacbr/rsfdata/b.rsf@"
    esize=4 type=float form=native
    n1=5           d1=0.004       o1=0          label1="Time" unit1="s"
5 elements 20 bytes
```

```
bash$ sfdisfil < b.rsf
```

```
   0:          0          1          0          0          0
```

Copying and deleting RSF

Delete header and data

```
sfrm file1.rsf file2.rsf [...]
```

```
bash$ rm a.rsf
bash$ ls /home/dlmacbr/rsfdata/a.rsf@
```

```
/home/dlmacbr/rsfdata/a.rsf@
```

```
bash$ sfrm b.rsf
bash$ ls /home/dlmacbr/rsfdata/b.rsf@
```

```
ls: /home/dlmacbr/rsfdata/b.rsf@ : No such file or directory
```

```
bash$ sfrm a.rsf
```

```
sfrm: build/api/c/files.c: Cannot open file a.rsf: No such file or directory
```

RSF dataset in a single file

Packing header and data

```
[< in.rsf] sfprog [> out.rsf] out=stdout
```

```
bash$ sfmath n1=3 o1=1 n2=2 o2=1 output="x1*x2" out=stdout >
a.rsf
bash$ sfin a.rsf
```

```
a.rsf:
    in="stdin"
    esize=4 type=float form=native
    n1=3           d1=1           o1=1
    n2=2           d2=1           o2=1
6 elements 24 bytes
```

in="stdin" indicates standalone RSF dataset

Exchange dataset between systems

```
< in.rsf sfdd form=xdr out=stdout > out.rsf
```

VPLOT

- ".vpl" suffix
- Vector image can be scaled without affecting quality
- Displayed by pen programs
- Compact

VPLOT

rsf → vpl → image

Madagascar plotting programs: `sfprog < in.rsf par= > out.vpl`

- sfgraph
- sfgrey
- sfgrey3
- sfcontour
- sfdots
- ...

pen progrms convert `.vpl` to images (`.eps`, `.gif`, `.png`, ...)

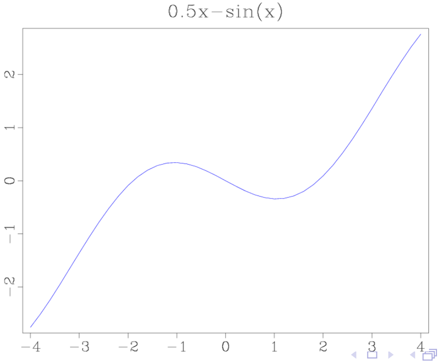
- vppen
- xtpen
- pspen
- ...

sfgraph

```
bash$ mkdir Fig/
```

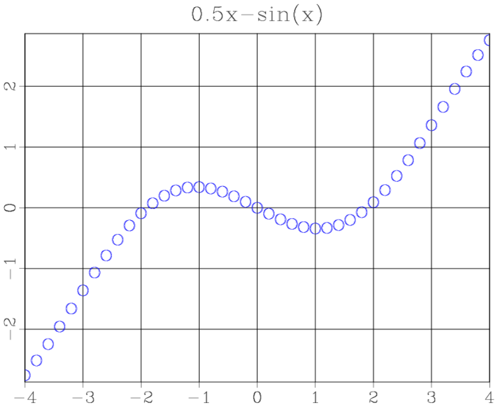
sfgraph

```
bash$ sfmath n1=41 o1=-4 d1=.2 output=".5*x1" > y1.rsf
bash$ < y1.rsf sfmath output="sin(x1)" > y2.rsf
bash$ < y1.rsf sfmath sin=y2.rsf output="input-sin" > y3.rsf
bash$ < y3.rsf sfgraph title="0.5x-sin(x)" > Fig/fig1.vpl
bash$ sfpen < Fig/fig1.vpl
```



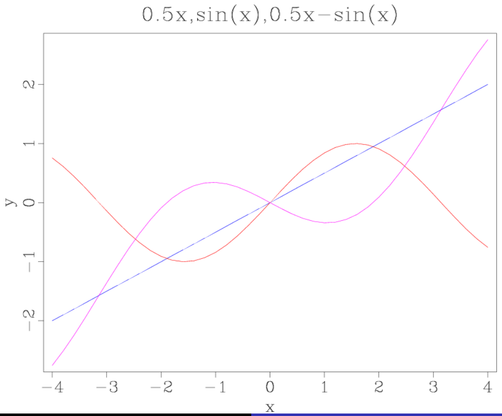
sfgraph

```
bash$ < y3.rsf sfgraph title="0.5x-sin(x)" symbol=o
symbolsz=12 grid=y min1=-4 max1=4 > Fig/fig2.vpl
bash$ sfpen < Fig/fig2.vpl
```



sfgraph

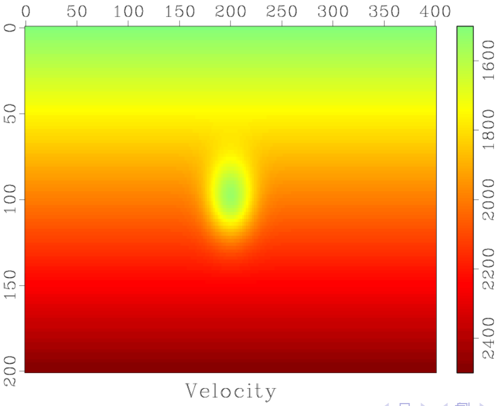
```
bash$ < y1.rsf sfcat y2.rsf y3.rsf axis=2 > y4.rsf
bash$ < y4.rsf sfgraph title="0.5x,sin(x),0.5x-sin(x)"
label1=x label2=y > Fig/fig3.vpl
bash$ sfpen < Fig/fig3.vpl
```



sfgrey

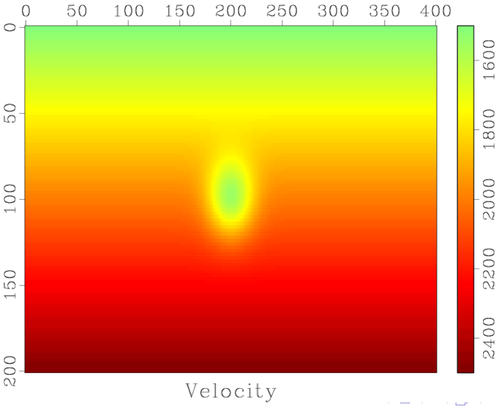
```
bash$ sfmath n1=101 d1=2 n2=201 d2=2 output="1500+5*x1" >
vb.rsf

bash$ < vb.rsf sfmath output= ‘‘-exp(-.002*((x1-100)*(x1-100)+
(x2-200)*(x2-200)))*45’’ > v1.rsf
```



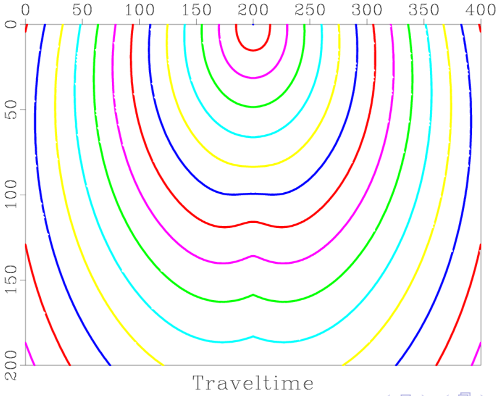
sfgrey

```
bash$ sfadd < vb.rsf v1.rsf scale=1,1 > v.rsf
bash$ < v.rsf sfgrey title=Velocity color=j bias=1500
scalebar=y barreverse=y > Fig/fig4.vpl
bash$ sfpen < Fig/fig4.vpl
```



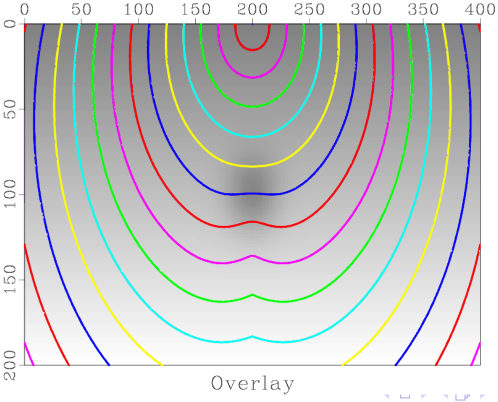
sfcontour

```
bash$ < v.rsf sfeikonal yshot=200 > eik.rsf
bash$ < eik.rsf sfcontour nc=45 title=Traveltime plotfat=5 >
Fig/fig5.vpl
bash$ sfpen < Fig/fig5.vpl
```



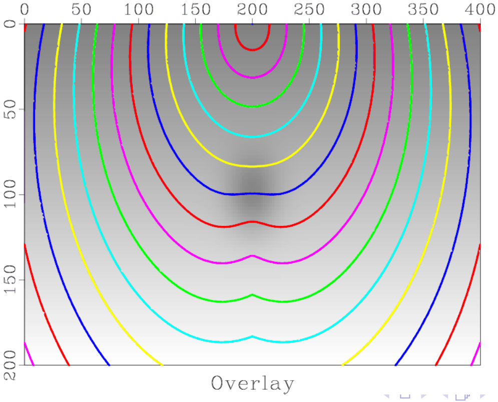
Overlay

```
bash$ < v.rsf sfgrey bias=1500 min1=0 max1=200 min2=0 max2=400
wanttitle=n wantaxis=n > v.vpl
bash$ < eik.rsf sfcontour title=Overlay nc=45 plotfat=5 min1=0
max1=200 min2=0 max2=400 > eik.vpl
```



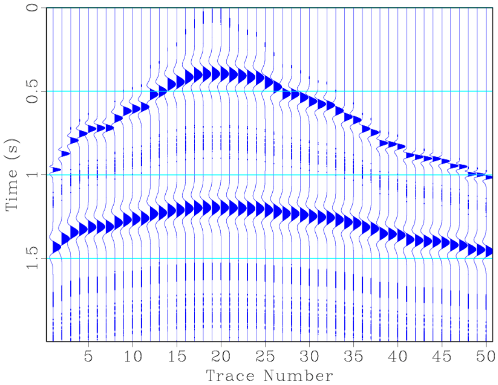
Overlay

```
bash$ vppen erase=o vpstyle=n v.vpl eik.vpl > Fig/fig6.vpl
bash$ sfpen < Fig/fig6.vpl
```



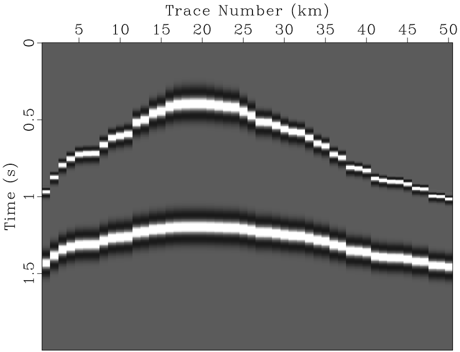
sfwiggle

```
bash$ < data.rsf sfwiggle title= label2="Trace Number"
yreverse=y transp=y poly=y > Fig/fig7.vpl
bash$ sfpen < Fig/fig7.vpl
```



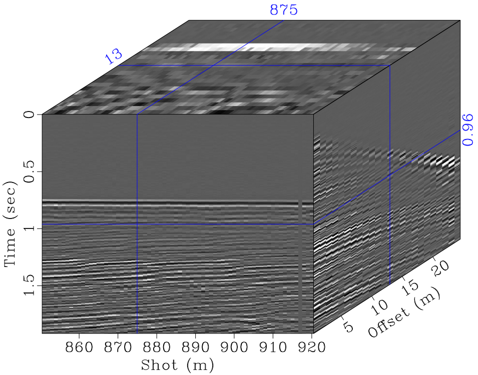
sfwiggle × sfgrey

```
bash$ < data.rsf sfgrey title= label2="Trace Number»
Fig/fig8.vpl
bash$ sfpen < Fig/fig8.vpl
```



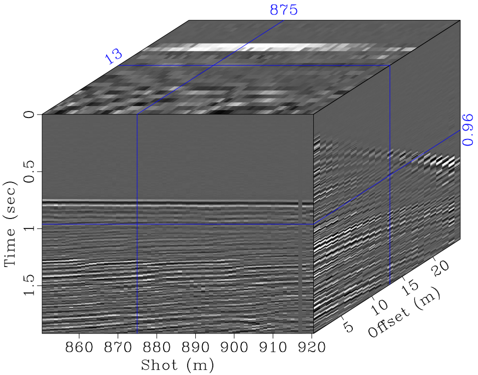
sfgrey3

```
bash$ retrieve(["shots.hh" ], [])
bash$ < shots.hh sfdd form=native type=float | sfwindow
n1=480 n2=24 | sftransp plane=23 | sfput label1=Time unit1=sec
label3=Offset unit3=m label2=Shot unit2=m > shots.rsf
```



sfgrey3

```
bash$ < shots.rsf sfbyte | sfgrey3 frame1=240 frame2=24
frame3=12 point1=0.7 point2=0.65 wanttitle=n flat=n
title="Data" > Fig/fig9.vpl
bash$ sfpen < Fig/fig9.vpl
```



Exercício 1

Montar um modelo de velocidade de 2000 m inline por 1000 m de profundidade, com amostras espaçadas de 10 m tanto na vertical como na horizontal. A velocidade de fundo é constante igual a 1500 m/s e com uma perturbação retangular com intensidade, posição e tamanho que você quiser.

Exercício 2

Montar um modelo de velocidade com as mesmas dimensões anteriores mas com velocidade de fundo com um gradiente de 1.5 $1/s$ e velocidade inicial de 1500 m/s. Além disso incluir uma perturbação circular com velocidade constante de 3000 m/s com raio e centro aonde você desejar.