Introduction to Madagascar

Prof. Daniel Leal Macedo

Faculdade de Geofísica - UFPa

31 de julho a 2 de agosto de 2018

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 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.
```

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)
```

Madagascar programs

Self-documetation

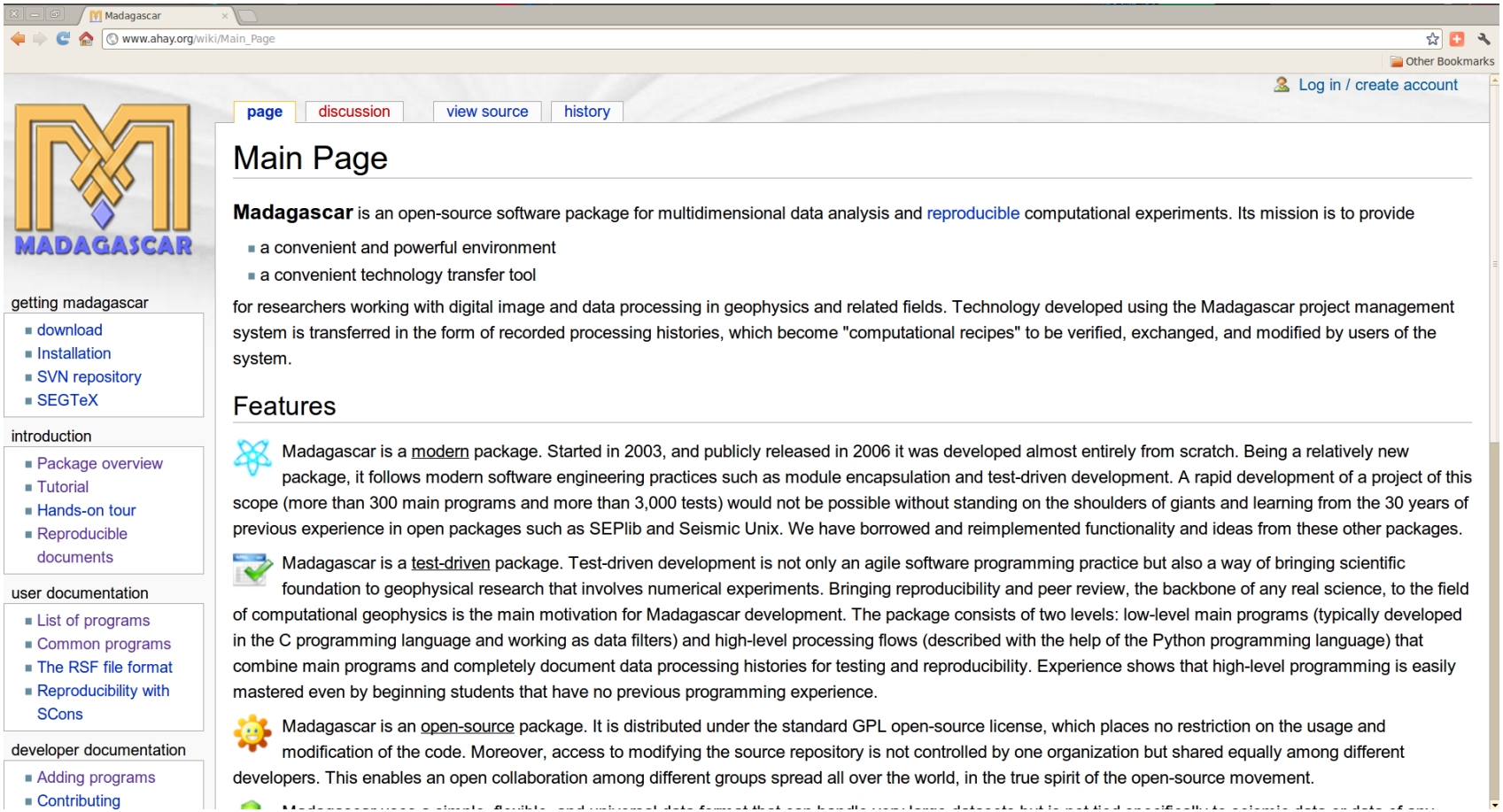
sfprog no arguments

```
...
        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`



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$ cd /path/to/madagascar/school/directory/1-Intro
```

Command-line usage - Example

```
bash$ cd /path/to/madagascar/school/directory/1-Intro
bash$ ls
```

Command-line usage - Example

```
bash$ cd /path/to/madagascar/school/directory/1-Intro
bash$ ls
```

```
g.asc plot
```

Command-line usage - Example

```
bash$ cd /path/to/madagascar/school/directory/1-Intro
bash$ ls
```

g.asc plot

```
bash$ sfspike
```

Command-line usage - Example

```
bash$ cd /path/to/madagascar/school/directory/1-Intro
bash$ ls
```

```
g.asc plot
```

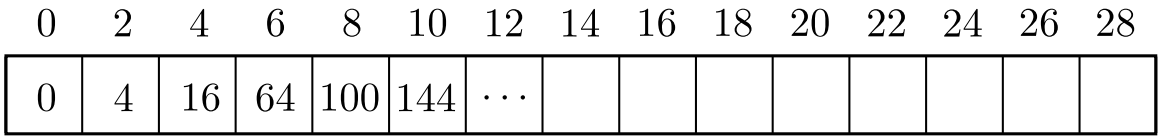
```
bash$ sfspike
```

```
NAME
        sfspike
DESCRIPTION
        Generate simple data: spikes, boxes, planes, constants.
SYNOPSIS
        sfspike < in.rsf > spike.rsf mag= nsp=1 k#=[0,...] l#=[k1,k2,...] p#=[0
COMMENTS

        Spike positioning is given in samples and starts with 1.

PARAMETERS
        float   d#=[0.004,0.1,0.1,...]  sampling on #-th axis
        ints    k#=[0,...]      spike starting position  [nsp]
        ints    l#=[k1,k2,...]  spike ending position  [nsp]
```

RSF data structure



- n1 = 15;
- o1 = 0;
- d1 = 2;

$$f(x) = x^2$$

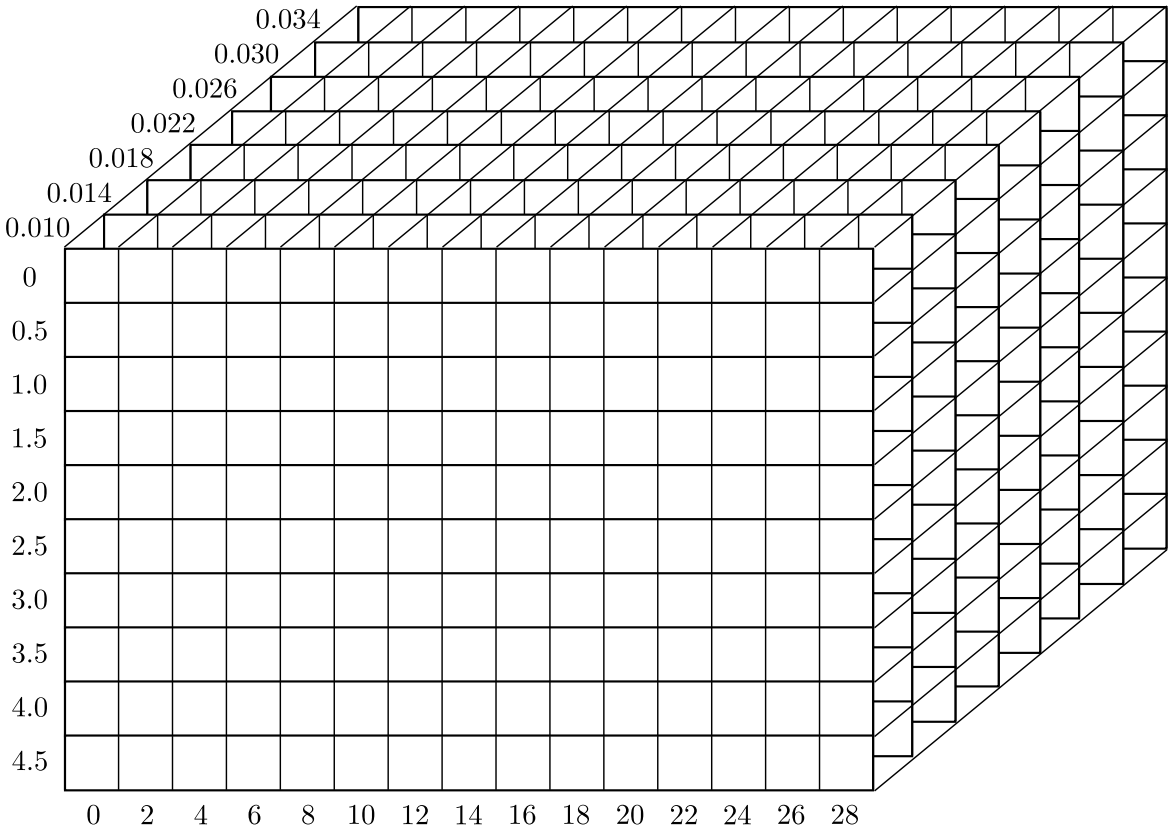
RSF data structure

	0	2	4	6	8	10	12	14	16	18	20	22	24	26	28
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0.5	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1.0	0	2	4	6	8	10	12	14	16	18	20	22	24	26	28
1.5	$\ddots$	$\cdots$													
2.0	$\vdots$	$\ddots$													
2.5															
3.0															
3.5															
4.0															
4.5															

- n1 = 15; n2 = 10;
- o1 = 0; o2 = 0;
- d1 = 2; d2 = 0.5;

$$f(x, z) = xz$$

RSF data structure



- n1 = 15;
  n2 = 10;
  n3 = 7;
- o1 = 0;
  o2 = 0;
  o3 = 0.010;
- d1 = 2;
  d2 = 0.5;
  d3 = 0.004;

Command-line usage - Example

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

Command-line usage - Example

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

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

Command-line usage - Example

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

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

```
bash$ ls
```

Command-line usage - Example

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

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

```
bash$ ls
```

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

Command-line usage - Example

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

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

```
bash$ ls
```

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

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

Command-line usage - Example

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

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

```
bash$ ls
```

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

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

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

Command-line usage - Example

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

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

```
bash$ ls
```

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

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

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

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

Command-line usage - Example

```
bash$ sfmath
```

Command-line usage - Example

```
bash$ sfmath
```

```
NAME
        sfmath
DESCRIPTION
        Mathematical operations on data files.
SYNOPSIS
        sfmath > out.rsf nostdin=n n#= d#=(1,1,...) o#=(0,0,...) label#= unit#=
COMMENTS

        Known functions:
        cos,  sin,  tan,  acos,  asin,  atan,
        cosh, sinh, tanh, acosh, asinh, atanh,
        exp,  log,  sqrt, abs,
        erf,  erfc, sign (for float data),
        arg,  conj, real, imag (for complex data).

        sfmath will work on float or complex data, but all the input and output
        files must be of the same data type.

        An alternative to sfmath is sfadd, which may be more efficient, but is
        less versatile
```

Command-line usage - Example

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

Command-line usage - Example

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

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

Command-line usage - Example

```
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
```

Command-line usage - Example

```
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$ < sfadd
```

Command-line usage - Example

```
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$ < sfadd
```

```
NAME
        sfadd
DESCRIPTION
        Add, multiply, or divide  RSF datasets.
SYNOPSIS
        sfadd > out.rsf scale= add= sqrt= abs= log= exp= mode= [< file0.rsf]
file1.rsf file2.rsf ...
COMMENTS
        The various operations, if selected, occur in the following order:

        (1) Take absolute value, abs=
        (2) Add a scalar, add=
```

Command-line usage - Example

```
bash$ < a.rsf sfadd scale=1,-2 b.rsf > c.rsf
```

Command-line usage - Example

```
bash$ < a.rsf sfadd scale=1,-2 b.rsf > c.rsf
bash$ sfdisfil < a.rsf
```

Command-line usage - Example

```
bash$ < a.rsf sfadd scale=1,-2 b.rsf > c.rsf
bash$ sfdisfil < a.rsf
```

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

Command-line usage - Example

```
bash$ < a.rsf sfadd scale=1,-2 b.rsf > c.rsf
bash$ sfdisfil < a.rsf
```

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

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

Command-line usage - Example

```
bash$ < a.rsf sfadd scale=1,-2 b.rsf > c.rsf
bash$ sfdisfil < a.rsf
```

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

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

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

Command-line usage - Example

```
bash$ < a.rsf sfadd scale=1,-2 b.rsf > c.rsf
bash$ sfdisfil < a.rsf
```

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

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

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

```
bash$ sfdisfil < c.rsf
```

Command-line usage - Example

```
bash$ < a.rsf sfadd scale=1,-2 b.rsf > c.rsf
bash$ sfdisfil < a.rsf
```

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

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

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

```
bash$ sfdisfil < c.rsf
```

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

Command-line usage - auxiliary/extra input/output

Command-line usage - auxiliary/extra input/output

1. file enumeration: `sfadd`

Command-line usage - auxiliary/extra input/output

1. file enumeration: `sfadd`

```
bash$ sfadd
```

Command-line usage - auxiliary/extra input/output

1. file enumeration: `sfadd`

```
bash$ sfadd
```

```
NAME
        sfadd
DESCRIPTION
        Add, multiply, or divide  RSF datasets.
SYNOPSIS
        sfadd > out.rsf scale= add= sqrt= abs= log= exp= mode= [< file
0.rsf] file1.rsf file2.rsf ...
COMMENTS
        The various operations, if selected, occur in the following order:

        (1) Take absolute value, abs=
        (2) Add a scalar, add=
```

Command-line usage - auxiliary/extra input/output

1. file enumeration: `sfadd`
2. via parameters: `sfawefd2d`

Command-line usage - auxiliary/extra input/output

1. file enumeration: `sfadd`
2. via parameters: `sfawefd2d`

```
bash$ sfawefd2d
```

Command-line usage - auxiliary/extra input/output

1. file enumeration: `sfadd`
2. via parameters: `sfawefd2d`

```
bash$ sfawefd2d
```

```
NAME
        sfawefd2d
DESCRIPTION
        2D acoustic time-domain FD modeling
SYNOPSIS
        sfawefd2d < file_wav.rsf vel=file_vel.rsf sou=file_src.rsf rec
=file_rec.rsf > file_dat.rsf wfl=file_wfl.rsf den=file_den.rsf verb=n
snap=n expl=n dabc=n cden=n adj=n fsrf=n optfd=n fdorder=4 hybridbc=n
sinc=n jsnap=nt jdata=1 nqz=sf_n(az) nqx=sf_n(ax) oqz=sf_o(az) oqx=sf_
o(ax) dqz=sf_d(az) dqx=sf_d(ax)
    .
    .
    .
```

Command-line usage - auxiliary/extra input/output

1. file enumeration: `sfadd`
2. via parameters: `sfawefd2d`

```
bash$ sfawefd2d
```

```
PARAMETERS
        bool    adj=n [y/n]     adjoint flag
        bool    cden=n [y/n]    Constant density
        bool    dabc=n [y/n]    Absorbing BC
        string  den=    auxiliary input file name
        float   dqx=sf_d(ax)    Saved wfld window dx
        float   dqz=sf_d(az)    Saved wfld window dz
        bool    expl=n [y/n]    Multiple sources, one wvlt
        int     fdorder=4       spatial FD order
        bool    fsrf=n [y/n]    Free surface flag
        bool    hybridbc=n [y/n]        hybrid Absorbing BC
        int     jdata=1         # of t steps at which to save receiver data
        int     jsnap=nt        # of t steps at which to save wavefield
```

Command-line usage - auxiliary/extra input/output

1. file enumeration: `sfadd`
2. via parameters: `sfawefd2d`

```
bash$ sfawefd2d
```

```
int     nqx=sf_n(ax)    Saved wfld window nx
int     nqz=sf_n(az)    Saved wfld window nz
bool    optfd=n [y/n]   optimized FD coefficients flag
float   oqx=sf_o(ax)    Saved wfld window ox
float   oqz=sf_o(az)    Saved wfld window oz
file    rec=    auxiliary input file name
bool    sinc=n [y/n]    sinc source injection
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
```

Command-line usage - auxiliary/extra input/output

1. file enumeration: `sfadd`
2. via parameters: `sfawefd2d`
3. variable definition: `sfmath`

Command-line usage - auxiliary/extra input/output

1. file enumeration: `sfadd`
2. via parameters: `sfawefd2d`
3. variable definition: `sfmath`

```
bash$ sfmath
```

Command-line usage - auxiliary/extra input/output

1. file enumeration: `sfadd`
2. via parameters: `sfawefd2d`
3. variable definition: `sfmath`

```
bash$ sfmath
```

```
NAME
        sfmath
DESCRIPTION
        Mathematical operations on data files.
SYNOPSIS
        sfmath > out.rsf nostdin=n n#= d#=(1,1,...) o#=(0,0,...) label
#= unit#= type= label= unit= output=
COMMENTS

        Known functions:
        cos,  sin,  tan,  acos,  asin,  atan,
    .
    .
```

Command-line usage - auxiliary/extra input/output

1. file enumeration: `sfadd`
2. via parameters: `sfawefd2d`
3. variable definition: `sfmath`

```
bash$ sfmath
```

```
      Examples:

      sfmath x=file1.rsf y=file2.rsf power=file3.rsf
        output='sin((x+2*y)^power)' > out.rsf
      sfmath < file1.rsf tau=file2.rsf output='exp(tau*input)' > out.rsf
      sfmath n1=100 type=complex output="exp(I*x1)" > out.rsf
```

.
.
.

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` ....

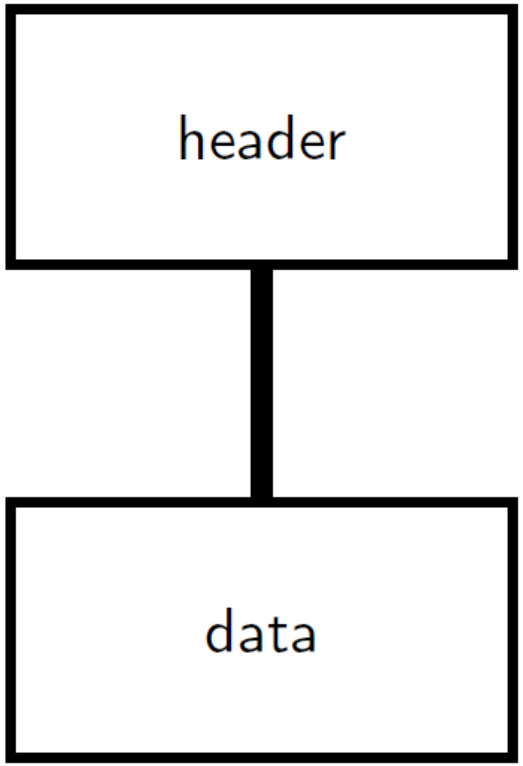
RSF "file" componets

Header file

- Text
- Small
- Portable

Data file

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



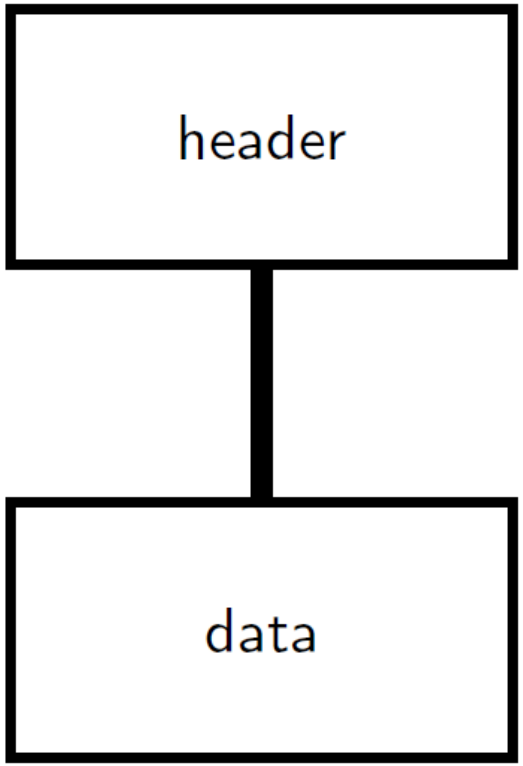
RSF "file" componets

Header file

- Text
- Small
- Portable

Data file

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




```
bash$ echo $DATAPATH
```

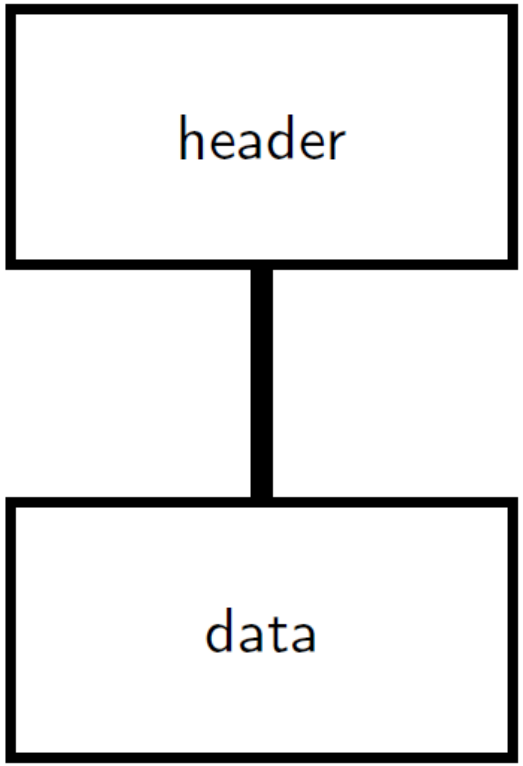
RSF "file" componets

Header file

- Text
- Small
- Portable

Data file

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




```
bash$ echo $DATAPATH
```

```
/home/daniel/rsfdata/
```

Header information

Example: construct Matrix

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

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
```

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
```

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
5:          6
```

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
5:          6
```

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

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
5:              6
```

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

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/daniel/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

Header information

Print out header

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

```
bash$ cd $DATAPATH
bash$ ls -l */*/*
```

.
.
.

RSF dataset attributes

Print out attributes

```
sfattr <  file.rsf
```

RSF dataset attributes

Print out attributes

```
sfattr <  file.rsf
```

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

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

(re)write header

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

Modify header

(re)write header

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

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

Modify header

(re)write header

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

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

```
d.rsf:
    in="/home/daniel/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
```

Modify header

(re)write header

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

```
bash$ < d.rsf sfput n1=6 n2=1 label1=''time'' > d2.rsf
```

Modify header

(re)write header

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

```
bash$ < d.rsf sfput n1=6 n2=1 label1=''time'' > d2.rsf
bash$ sfin d2.rsf
```

Modify header

(re)write header

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

```
bash$ < d.rsf sfput n1=6 n2=1 label1="time" > d2.rsf
bash$ sfin d2.rsf
```

```
d2.rsf:
    in="/home/daniel/rsfdata/d2.rsf@"
    esize=4 type=float form=native
    n1=6            d1=1            o1=1            label1="time"
    n2=1            d2=1            o2=1
   6 elements 24 bytes
```

Memory allocation

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

Memory allocation

```
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
```

Memory alocation

```
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
```

```
  0:          111          211          311          121          221
  5:          321          112          212          312          122
 10:          222          322
```

Memory alocation

```
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
```

```
  0:          111          211          311          121          221
  5:          321          112          212          312          122
 10:          222          322
```

```
bash$ sfdisfil < e.rsf col=3
```

Memory alocation

```
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
```

```
  0:          111          211          311          121          221
  5:          321          112          212          312          122
 10:          222          322
```

```
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

Moving RSF dataset

`mv` moves header ONLY

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

Moving RSF dataset

`mv` moves header ONLY

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

Moving RSF dataset

`mv` moves header ONLY

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

```
f.rsf:
    in="/home/daniel/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
```

Moving RSF dataset

`mv` moves header ONLY

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

```
f.rsf:
    in="/home/daniel/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
```

Moving RSF dataset

`mv` moves header ONLY

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

```
f.rsf:
    in="/home/daniel/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
```

Moving RSF dataset

Move header and data

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

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

Moving RSF dataset

Move header and data

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

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

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
```

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/daniel/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
```

Copying and deleting RSF

Copy header and data

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

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

Copying and deleting RSF

Copy header and data

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

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

Copying and deleting RSF

Copy header and data

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

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

```
bash$ sfin b.rsf
```

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

Copying and deleting RSF

Copy header and data

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

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

```
bash$ sfin b.rsf
```

```
b.rsf:
    in="/home/daniel/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
```

Copying and deleting RSF

Copy header and data

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

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

```
bash$ sfin b.rsf
```

```
b.rsf:
    in="/home/daniel/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 [...]
```

Copying and deleting RSF

Delete header and data

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

```
bash$ rm a.rsf
```

Copying and deleting RSF

Delete header and data

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

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

Copying and deleting RSF

Delete header and data

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

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

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

Copying and deleting RSF

Delete header and data

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

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

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

```
bash$ sfrm b.rsf
```

Copying and deleting RSF

Delete header and data

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

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

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

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

Copying and deleting RSF

Delete header and data

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

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

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

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

```
ls: cannot access '/home/daniel/rsfdata/b.rsf@' : No such file or directory
```

Copying and deleting RSF

Delete header and data

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

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

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

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

```
ls: cannot access '/home/daniel/rsfdata/b.rsf@' : No such file or directory
```

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

Copying and deleting RSF

Delete header and data

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

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

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

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

```
ls: /home/daniel/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
```

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
```

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
```

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
```

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

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
```

Exercício 1

Construa um vetor com 20 amostras que contenha um período completo de $\sin t$. Visualize na tela seu resultado. Em seguida concatene dois destes períodos para formar um vetor de 40 amostras que contenha dois períodos. Visualize na tela seu resutado.

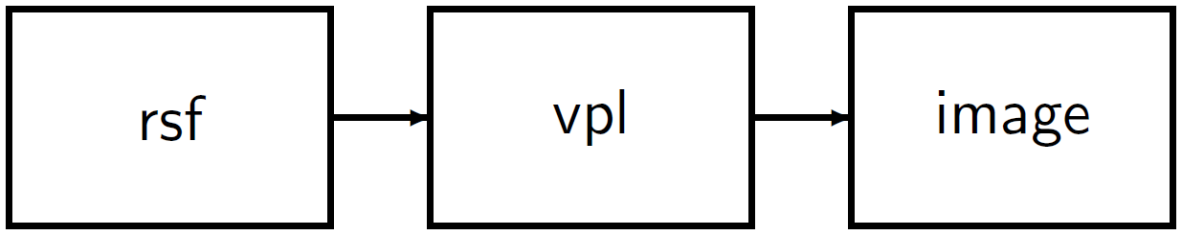
Exercício 2

Construa uma matriz que contenha os valores referentes a um parabolóide circular centrado na origem. A origem deve estar no centro da matriz. Primeiramente faça a matriz quadrada e visualize o resultado na tela. Em seguida faça uma matriz retangular e visualize na tela.

VPLOT

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

VPLOT



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$ cd plot
```

sfgraph

```
bash$ cd plot
bash$ mkdir Fig/
```

sfgraph

```
bash$ cd plot
bash$ mkdir Fig/
bash$ ls
```

sfgraph

```
bash$ cd plot
bash$ mkdir Fig/
bash$ ls
```

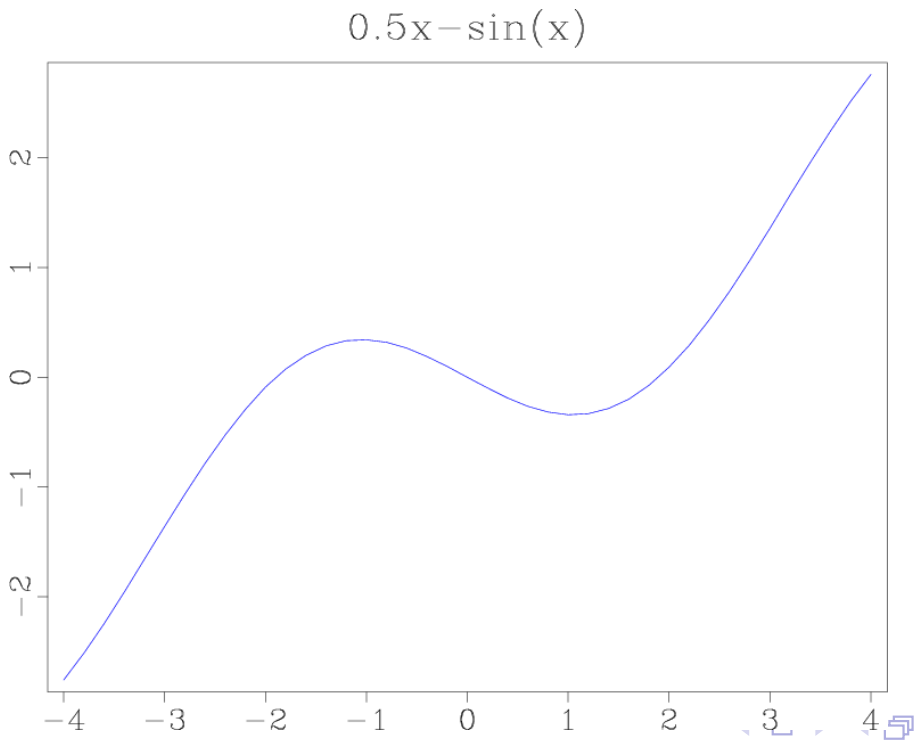
```
data.rsf  Fig  plot.sh  SConstruct  shots.rsf
```

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$ 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
```

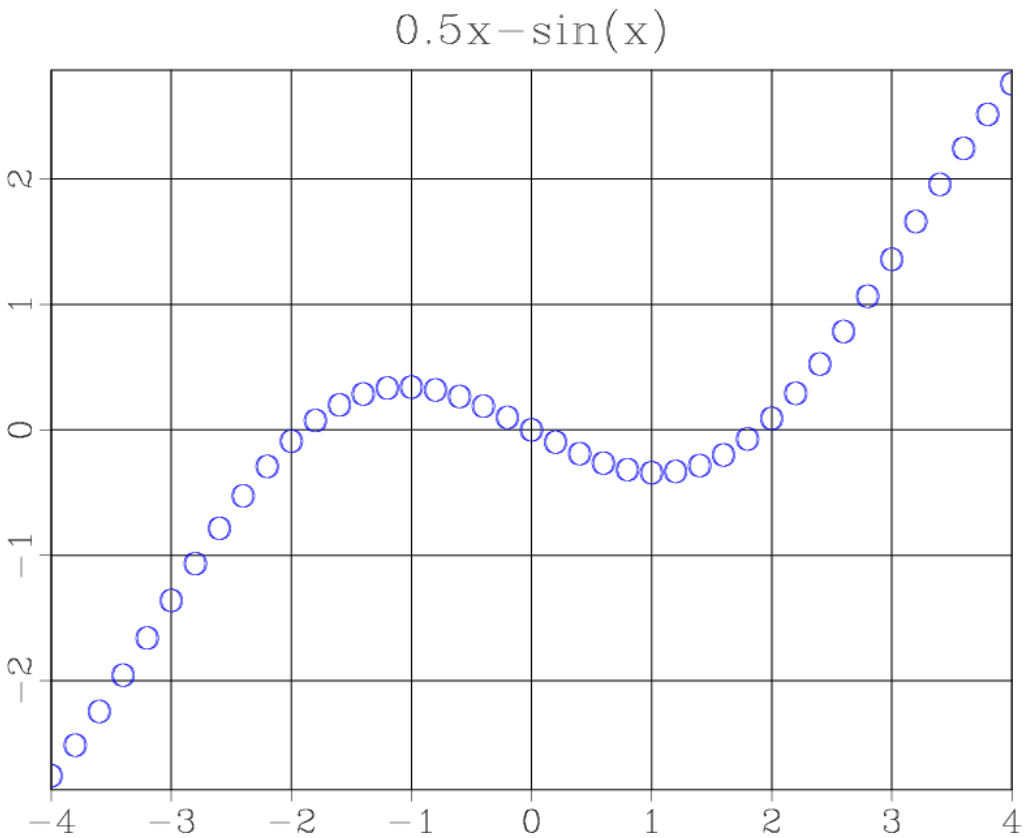


sfgraph

```
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$ < 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
```

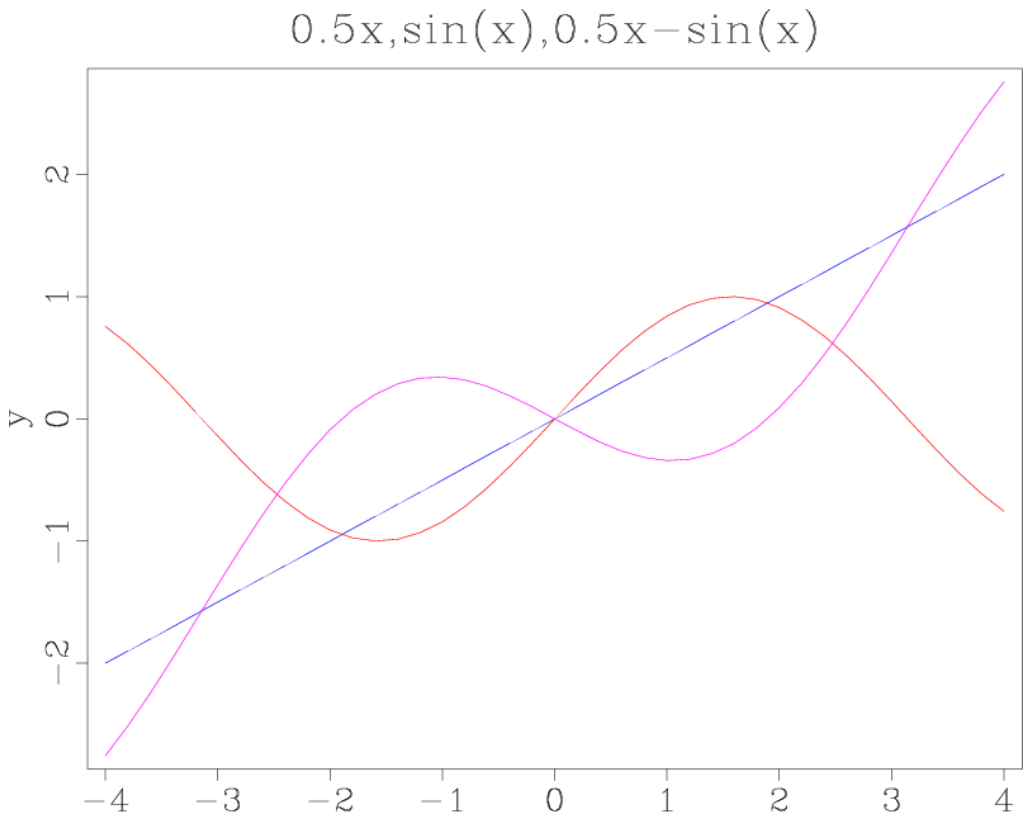


sfgraph

```
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
```

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

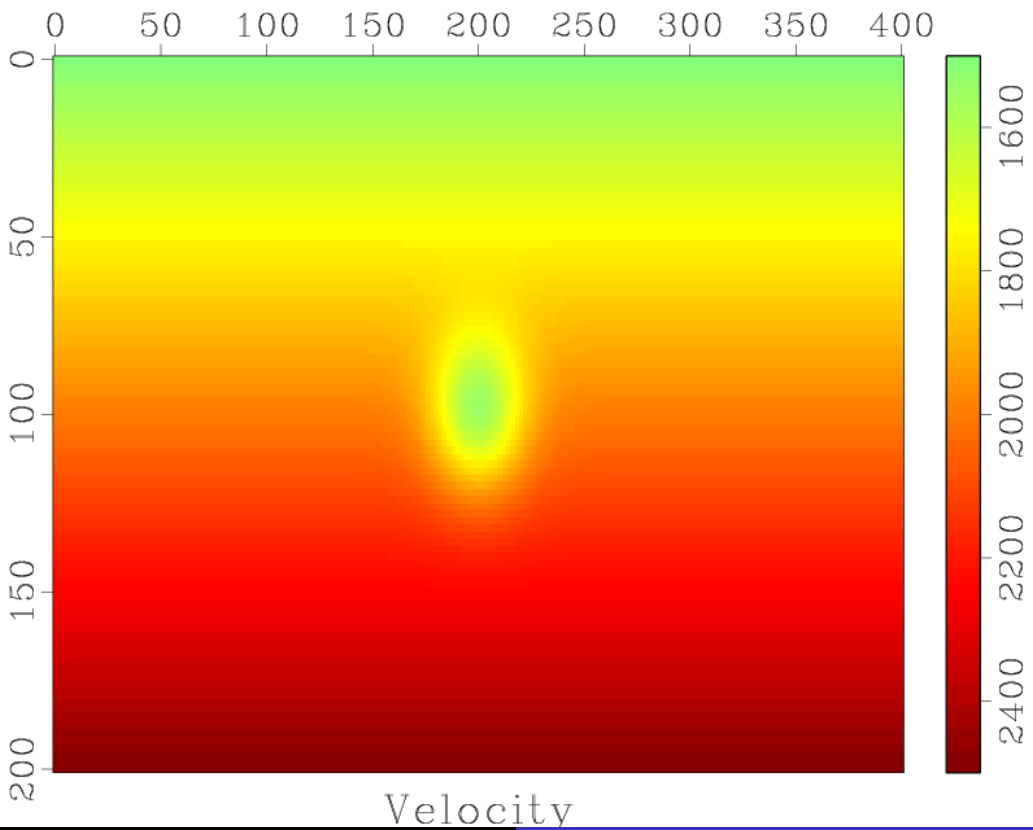
```
bash$ sfmath n1=101 d1=2 n2=201 d2=2 output="1500+5*x1" >
vb.rsf
bash$ < vb.rsf sfmath output="-exp(-.002*((x1-100)*(x1-100)+
(x2-200)*(x2-200)))*450" > v1.rsf
```

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
```

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
```

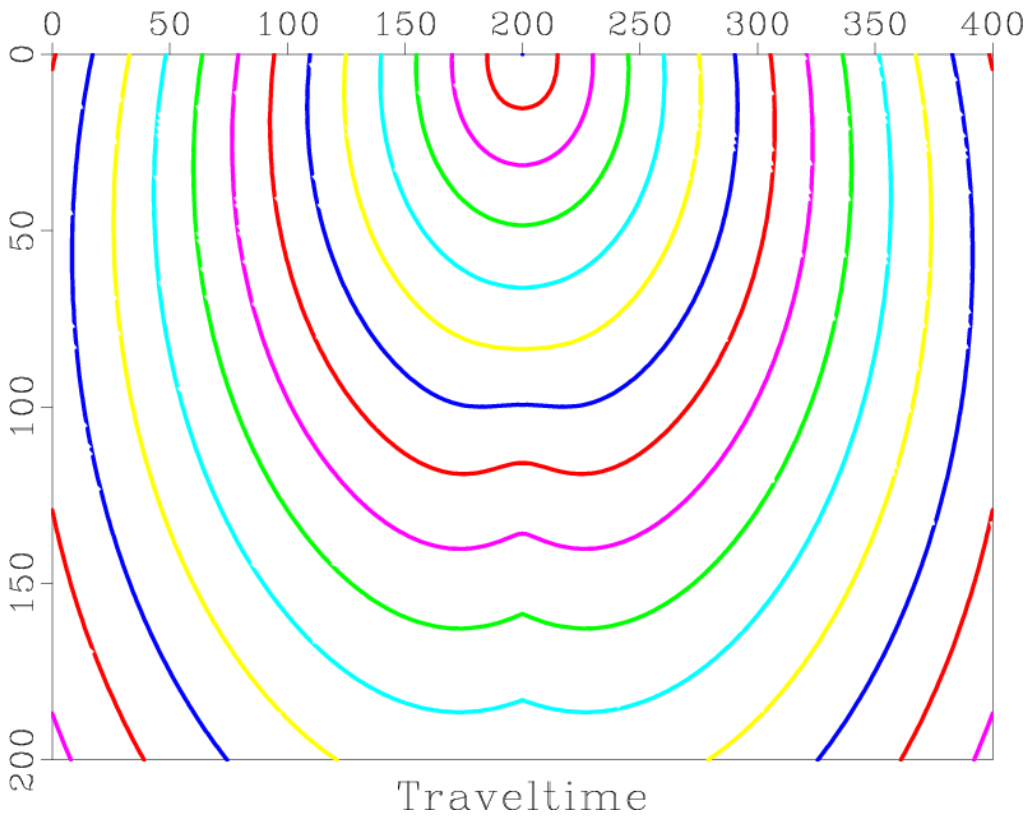


sfcontour

```
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
```

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
```

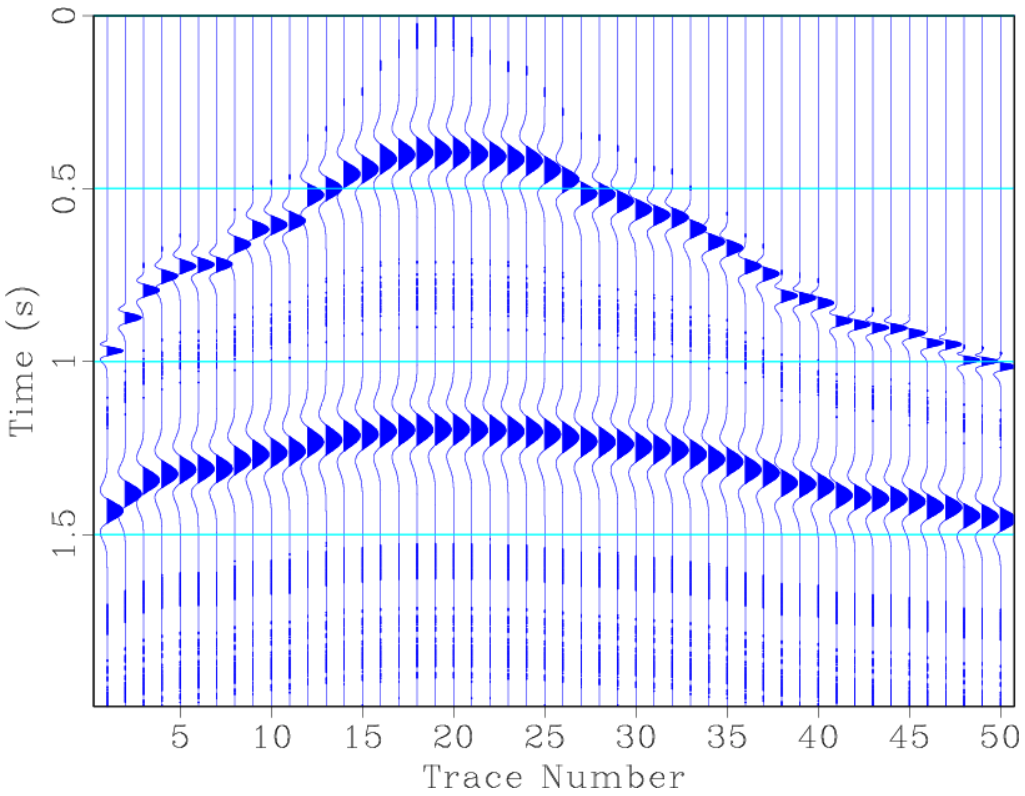


sfwiggle

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

sfwiggle

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

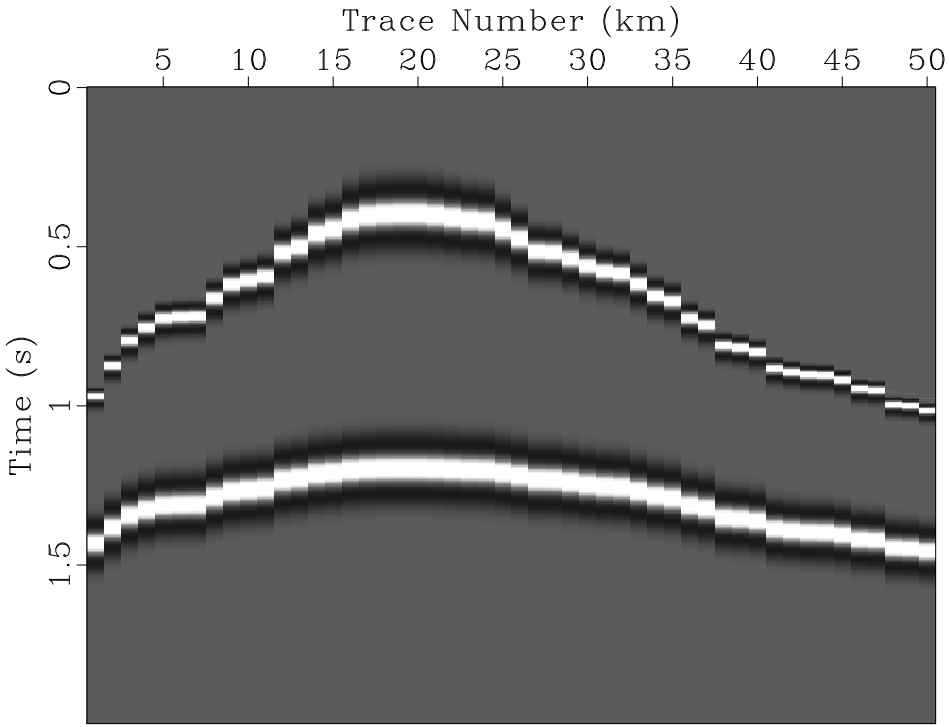


sfwiggle × sfgrey

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

sfwiggle × sfgrey

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

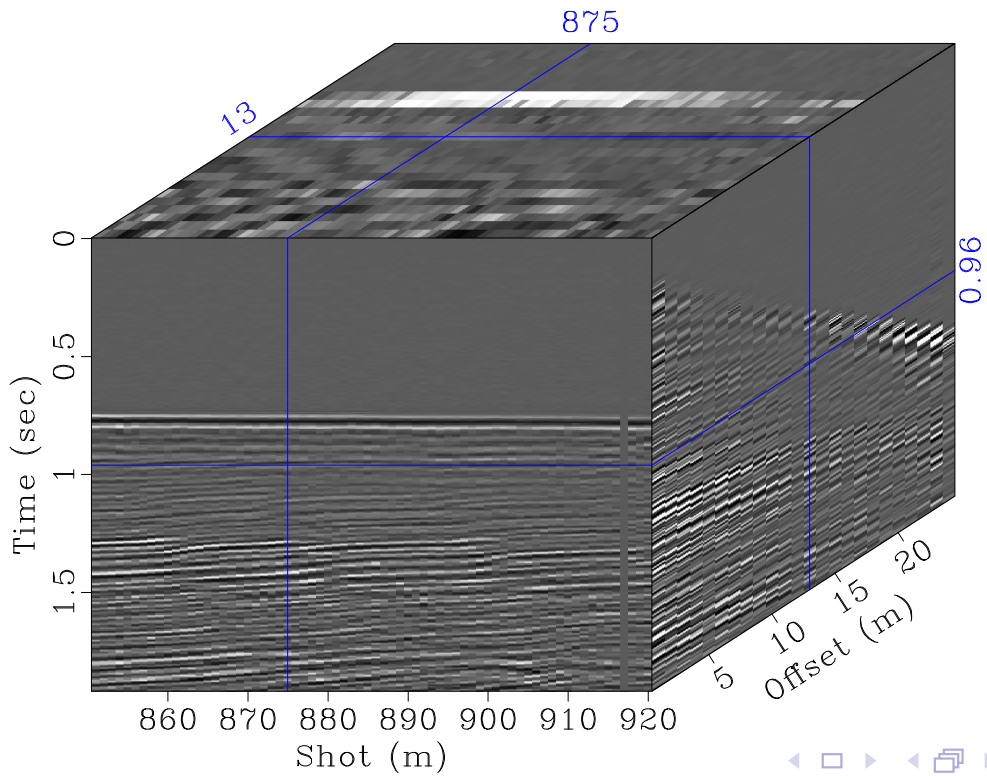


sfgrey3

```
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
```

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 3

Plote os resultados obtidos no exercício 2 usando `sfcontour` e `sfgrey`.

Exercício 4

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 5

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.