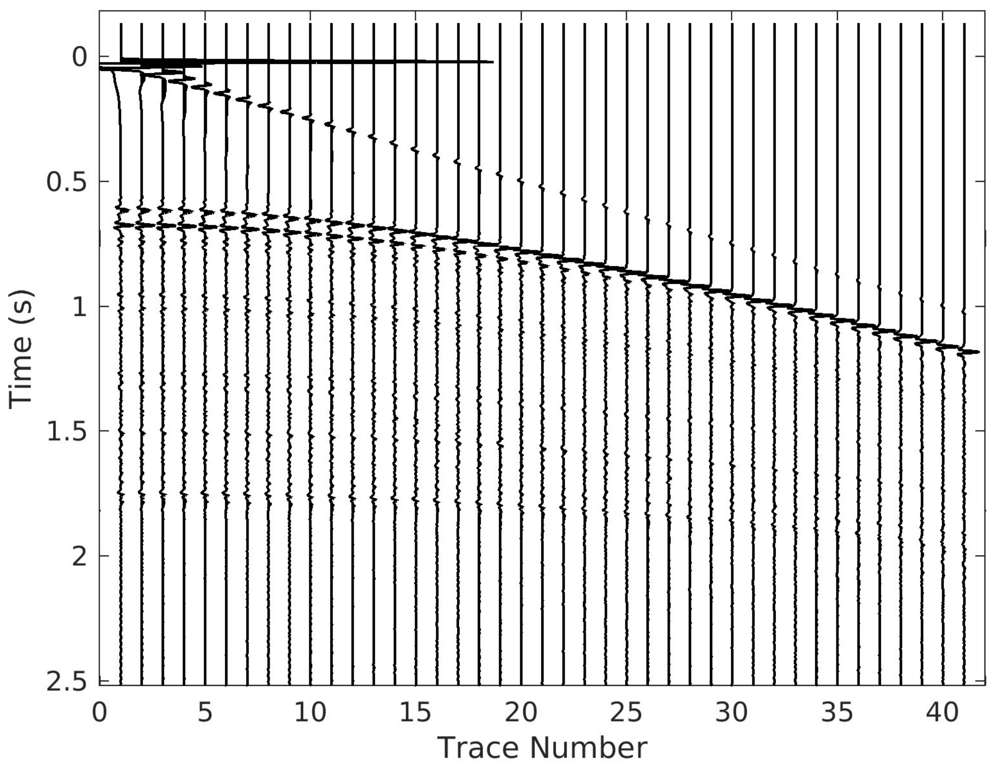
SegyMAT

A Matlab/Octave toolbox for reading and writing SEG-Y formatted files



Thomas Mejer Hansen

Table of Contents

SegyMAT

SegyMAT is a set of m-files for reading and writing SEG-Y files from Matlab and Octave, that aims to

- completely support SEG-Y revision 0 and 1;
- be easy to use in other projects;
- be a Swiss Army knife dealing with the SEGY-Y format in Matlab/Octave.

SegyMAT is not lightning fast. SegyMAT makes heavy use of 'structures'. Unfortunately structures are not very effective in terms of speed in Matlab. (Or they have not been implemented very effectively in SegyMAT). However structures make the implementation and maintenance easier, and the code (hopefully) easy to read. That said, some effort has been made to optimize SegyMAT for speed.

The latest **stable** version of SegyMAT is available from Sourceforge.

The current **development** version of SegyMAT is available from Github.

Quickstart (The wiggle plot on the cover):

```
[Data,SegyTraceHeaders,SegyHeader]=ReadSegy('f11_02673_45Hz.segy');
wiggle([SegyTraceHeaders.TraceNumber],SegyHeader.time,Data,'VA',.006);
xlabel('Trace Number');ylabel('Time (s)')
```

License (LGPL)

Installation and requirements

SegyMAT has been developed and tested using Matlab (R2016a) running on Linux and Windows 10. Any other Matlab supported platform should work.

As of version 1.02 Octave (version >2.1.64) is supported as well.

No Matlab toolboxes are required.

Local Installation

Running Matlab with Java extensions (the default), the path can be set using the `pathtool` command, by selecting the install directory (and subfolder GUI), and save the path and you are done:

```
>> pathtool
```

To install without using the commandline, one can can manually add the install folder to Matlabs search path. If the install directory of SegyMAT is `/usr/share/matlab/SegyMAT` simply use:

```
>> addpath /usr/share/matlab/SegyMAT -begin
>> addpath /usr/share/matlab/SegyMAT/GUI -begin
```

Global Installation

For a system wide installation add the following line (substituting the location of the directory)

```
>> addpath /usr/share/matlab/SegyMAT -begin
>> addpath /usr/share/matlab/SegyMAT/GUI -begin
```

to `pathdef.m` , usually located in `$MATLAB_INSTALL/toolbox/local/pathdef.m`

The SEGY-Y format

SegyMAT has been implemented using the SEG-Y revision 0 and revision 1 standards as defined by SEG[1].

SegyMAT also has support for reading and writing the format used by CWP's Seismic Unix package (the SU format), which is merely a simplified version the SEG-Y format.

A short description of the formats follows here.

Structure of a file

A SEG-Y file consists of a 3600 byte header; a number of extended textual headers; a number trace headers+data.

- A 3200 byte Textual File Header, ASCII or EBCDIC formated.
- A 400 byte Binary File Header
- A (optional) number of 'Extended Textual File Headers', 3200 bytes long, ASCII or EBCDIC formatted.
- A number of traces, separated into a 240 bytes long binary Trace Header, followed by the Trace Data, that can be formatted in a number of ways : IEEE, IBM Floating Point, 1,2 and 4 byte two's complement integers.

Structure of a SU file

A SU formatted file is just a simple version of a file, containing only trace information :

- No 3200 byte textual header and no extended textual headers.
- No binary header.
- The data must be formatted as IEEE.

- Data can be both little and big endian formatted.

What is supported in SegyMAT ?

The following parts of the SEG-Y format, revision 0 and 1, are supported

Textual file headers

The Textual 400 byte file header can be both ASCII and EBCDIC formatted, using revision 1.

Extended Textual Headers

In revision 1 a number of extended textual file headers are allowed.

Data Sample Format / Revision

The following data formats are supported :

REVISION 0 (1975):

Type	DataSampleFormat	Supported
1	4 Byte IBM Floating Point	Yes
2	4 Byte Fixed Point	No
3	2 Byte Fixed Point	No
4	4 Byte Fixed Point with Gain	No

REVISION 1 (2002)

Type	DataSampleFormat	Supported
1	4 Byte IBM Floating Point	Yes
2	4 Byte two's complement integer	Yes
3	2 Byte two's complement integer	Yes
4	4 Byte Fixed Point with Gain	No
5	4 Byte IEEE FLoating Pint	Yes
6	Not Specified	
7	Not Specified	
8	1 Byte Fixed Point with Gain	Yes

The type number is the number that should be used as '`dsf`' (Data Sample Format), for functions like ReadSegy, WriteSegy, WriteSegyStructure.

Segy Trace Header name definition

The definition of trace header names, location in the Trace Header and precision can be listed by running

```
TraceHeaderDef;
```

which provides the folloing output:

```
 POS   PREC Traece Header Name
   0  int32 TraceSequenceLine
   4  int32 TraceSequenceFile
   8  int32 FieldRecord
  12  int32 TraceNumber
  16  int32 EnergySourcePoint
  20  int32 cdp
  24  int32 cdpTrace
  28  int16 TraceIdenitifactionCode
  30  int16 NSummedTraces
  32  int16 NStackedTraces
  34  int16 DataUse
  36  int32 offset
  40  int32 ReceiverGroupElevation
  44  int32 SourceSurfaceElevation
  48  int32 SourceDepth
  52  int32 ReceiverDatumElevation
  56  int32 SourceDatumElevation
  60  int32 SourceWaterDepth
  64  int32 GroupWaterDepth
  68  int16 ElevationScalar
  70  int16 SourceGroupScalar
  72  int32 SourceX
  76  int32 SourceY
  80  int32 GroupX
  84  int32 GroupY
  88  int16 CoordinateUnits
  90  int16 WeatheringVelocity
  92  int16 SubWeatheringVelocity
  94  int16 SourceUpholeTime
  96  int16 GroupUpholeTime
  98  int16 SourceStaticCorrection
 100  int16 GroupStaticCorrection
 102  int16 TotalStaticApplied
 104  int16 LagTimeA
 106  int16 LagTimeB
 108  int16 DelayRecordingTime
 110  int16 MuteTimeStart
 112  int16 MuteTimeEND
 114 uint16 ns
 116 uint16 dt
 118  int16 GainType
 120  int16 InstrumentGainConstant
 122  int16 InstrumentInitialGain
 124  int16 Correlated
 126  int16 SweepFrequenceStart
 128  int16 SweepFrequenceEnd
 130  int16 SweepLength
 132  int16 SweepType
 134  int16 SweepTraceTaperLengthStart
 136  int16 SweepTraceTaperLengthEnd
 138  int16 TaperType
 140  int16 AliasFilterFrequency
```

```
  142  int16 AliasFilterSlope
  144  int16 NotchFilterFrequency
  146  int16 NotchFilterSlope
  148  int16 LowCutFrequency
  150  int16 HighCutFrequency
  152  int16 LowCutSlope
  154  int16 HighCutSlope
  156  int16 YearDataRecorded
  158  int16 DayOfYear
  160  int16 HourOfDay
  162  int16 MinuteOfHour
  164  int16 SecondOfMinute
  166  int16 TimeBaseCode
  168  int16 TraceWeightningFactor
  170  int16 GeophoneGroupNumberRoll1
  172  int16 GeophoneGroupNumberFirstTraceOrigField
  174  int16 GeophoneGroupNumberLastTraceOrigField
  176  int16 GapSize
  178  int16 OverTravel
  180  int32 cdpX
  184  int32 cdpY
  188  int32 Inline3D
  192  int32 Crossline3D
  196  int32 ShotPoint
  200  int16 ShotPointScalar
  202  int16 TraceValueMeasurementUnit
  204  int32 TransductionConstantMantissa
  208  int16 TransductionConstantPower
  210  int16 TransductionUnit
  212  int16 TraceIdentifier
  214  int16 ScalarTraceHeader
  216  int16 SourceType
  218  int32 SourceEnergyDirectionMantissa
  222  int16 SourceEnergyDirectionExponent
  224   in32 SourceMeasurementMantissa
  228  int16 SourceMeasurementExponent
  230  int16 SourceMeasurementUnit
  232  int32 UnassignedInt1
  236  int32 UnassignedInt2
```

1

> [1]. The Society of Exploration Geophysicists ↩

Reading SEG-Y files

This section documents how SEG-Y files are read using SegyMAT.

ReadSegy

`ReadSegy.m` can be used to read SEG-Y formatted files :

```
[Data,SegyTraceHeaders,SegyHeader]=ReadSegy('data.segy');
wiggle(Data,[],SegyHeader.time,[SegyTraceHeaders.cdp],'VA')
imagesc([SegyTraceHeaders.cdp],[SegyHeader.time],Data)
```

This will read `data.segy` using the revision and data sample format specified in the binary header (`SegyHeader`), and plot the data using the `wiggle` plotting function.

`Data` is a 2D variable containing the seismic data of size `[Nsamples x Ntraces]` .

`SegyTraceHeaders` is a structure of size `[1,Ntraces]' structure containing all the header values from the traces. Type` SegyTraceHeaders `to see a list of header information.` SegyTraceHeaders(9)`, list all header names and values of trace number 9.

```
>> SegyTraceHeaders(9)

   ans =

                        SegyMAT_TraceStart: 91952
                         TraceSequenceLine: 0
                         TraceSequenceFile: 9
                               FieldRecord: 0
                               TraceNumber: 9
                         EnergySourcePoint: 0
                                       cdp: 0
                                  cdpTrace: 0
                   TraceIdenitifactionCode: 0
                             NSummedTraces: 0
                            NStackedTraces: 0
                                   DataUse: 0
                                    offset: 400
                                      ...
             SourceEnergyDirectionMantissa: 0
             SourceEnergyDirectionExponent: 0
                 SourceMeasurementMantissa: 0
                 SourceMeasurementExponent: 0
                     SourceMeasurementUnit: 0
                            UnassignedInt1: 0
                            UnassignedInt2: 0
                    SegyMAT_TraceDataStart: 92192
```

To access an array of trace header values simply use square brackets as :

```
    cdp=[SegyTraceHeaders.cdp];
    offset=[SegyTraceHeaders.offset];
    ...
```

`SegyHeader` is a structure containing all the Segyheader values. Typing `SegyHeader` will list the names and values of all header values.

```
>> SegyHeader

    SegyHeader =

                          Rev: [1x2 struct]
            TextualFileHeader: [3200x1 double]
                          Job: 0
                         Line: 0
                         Reel: 0
         DataTracePerEnsemble: 0
    AuxiliaryTracePerEnsemble: 0
                           dt: 1000
                       dtOrig: 0
                           ns: 2701
                       nsOrig: 0
             DataSampleFormat: 5
                 EnsembleFold: 0
                 TraceSorting: 0
              VerticalSumCode: 0
          SweepFrequencyStart: 0
            SweepFrequencyEnd: 0
                  SweepLength: 0
                    SweepType: 0
                 SweepChannel: 0
        SweepTaperlengthStart: 0
          SweepTaperLengthEnd: 0
                    TaperType: 0
         CorrelatedDataTraces: 0
                   BinaryGain: 0
      AmplitudeRecoveryMethod: 0
            MeasurementSystem: 1
        ImpulseSignalPolarity: 0
        VibratoryPolarityCode: 0
                  Unassigned1: [120x1 double]
     SegyFormatRevisionNumber: 100
         FixedLengthTraceFlag: 1
    NumberOfExtTextualHeaders: 0
                  Unassigned2: [47x1 double]
                         time: [1x2701 double]
```

A number of arguments can be given to `ReadSegy`, controlling what type of and which part of the data to read.

Read specific trace numbers

To read traces 100, 201 and 320 use e.g.

```
>> [Data,SegyTraceHeaders,SegyHeader]=ReadSegy(filename,'traces'
,[100 201 320]);
```

Use for example ReadSegyTraceHeadervalue.m and 'find' to find a list of trace ids (this is equivalent to using the 'minmax' option)

```
>> cdp=ReadSegyTraceHeaderValue(filename,'key','cdp');
>> traces = find(cdp>100 & cdp<200)
>> [Data,SegyTraceHeaders,SegyHeader]=ReadSegy(filename,'traces'
,[100 201 320]);
```

Read only every 5th trace

```
>> [Data,SegyTraceHeaders,SegyHeader]=ReadSegy(filename,'jump',5
);
```

To read time slice 0.5 < t < 5

```
      >> [Data,SegyTraceHeaders,SegyHeader]=ReadSegy(filename,'t
range',.5,3);
```

Read data in a CDP header range : 5000<cdp<5800 (the 'minmax' option)

`cdp` can be changed to any other valid header name

```
>> [Data,SegyTraceHeaders,SegyHeader]=
      ReadSegy(filename,'minmax','cdp',5000,5800);
```

Read only header values

In some cases it can be desirable only to read the header information (the SegyHeader and SegyTraceHeaders). This will return an empty `Data` variable.

```
>> [Data,SegyTraceHeaders,SegyHeader]=
      ReadSegy(filename,'SkipData',1);
```

SEG-Y format revision

SEG-Y format revision number can be '0' (1975) or '1' (2002). By default the SEG-Y format revision number is read in the binary header, but this can be overruled using :

```
>> [Data,SegyTraceHeaders,SegyHeader]=
      ReadSegy(filename,'revision',0);
```

A specific Data Sample Format

One can overrule the Data Sample Format listed in the binary header, using the `dsf` argument. See Data Sample Formats for a list of valid and supported values.

```
>> % Rev 0, IBM FLOATING POINT
>>  [Data,SegyTraceHeaders,SegyHeader]=
       ReadSegy(filename,'revision',0,'dsf',1);
>> % Rev 1, IEEE FLOATING POINT
>>  [Data,SegyTraceHeaders,SegyHeader]=
       ReadSegy(filename,'revision',1,'dsf',5);
```

If `dsf` is set to 5 and `revision` to 0, a warning message will occur, since data sample format 5 is only defined in revision 1. The revision is then automatically set to 1.

Force the use of a specific SegyHeader

```
>> [Data,SegyTraceHeaders,SegyHeader]=
      ReadSegy(filename,'SegyHeader',SegyHeader);
```

ReadSegyFast

`ReadSegyFast.m` is a faster implementation of ReadSegy.m since no trace header values are read. Thus this function will just return the seismic data and the SegyHeader. e.g. :

```
>> [Data,SegyHeader]=ReadSegyFast('data.segy');
>> imagesc(Data)
```

If `ReadSegy` is called with only one output argument, `ReadSegyFast` will be used instead of `ReadSegy` .

ReadSegyFast options

Most of the same options that works for ReadSegy.m will also work for ReadSegyFast.m. The data sample format can be chosen using the 'revision' and 'dsf' tags. Also a 'SegyHeader' can be specified.

ReadSegyFast.m is currently optimized only for reading the whole SEGY-Y file, but the options 'jump' and 'trange' can be used (but will currently not result in faster read times).

Since the trace header values are not read, the 'minmax' option is not supported.

ReadSegyHeader

`ReadSegyHeader.m` reads the Binary Segy Header only. It can be called with the same options as ReadSegy.m

Force using little endian :

```
>> SegyHeader=ReadSegyHeader(filename,'endian','l');
```

ReadSegyTraceHeaderValue

`ReadSegyTraceHeaderValue.m` reads one trace header value into an array. This approach is much faster than to read the whole file

using a keyword

To read a trace header value by its trace header key. See the definition of all the Trace Header names to use the correct key:

```
cdp=ReadSegyTraceHeaderValue(filename,'key','cdp');
SourceX=ReadSegyTraceHeaderValue(filename,'key','SourceX');
SourceY=ReadSegyTraceHeaderValue(filename,'key','SourceY');
plot(SourceX,SourceY)
```

using location+type

To read a trace header by its position in the trace header using a specific data sample format, use:

```
SourceX=ReadSegyTraceHeaderValue(filename,'pos',70,'precision','
int32');
```

ReadSegyConstantTraceLength

Assuming a constant trace length (which is much more common than not) allows much faster reading of parts of large file. For example to read trace number 2030, the whole SEG-Y file must be sequentially read, assuming variable trace length. Assuming constant trace length the trace can be directly (and fast) located in the data cube.

To read trace 2030 use

```
[Data,SegyTraceHeader,SegyHeader]=ReadSegyConstantTraceLength(fi
lename,'trace',2030);
```

To read traces 1-2000 and 2020-2040 use

```
[Data,SegyTraceHeader,SegyHeader]=ReadSegyConstantTraceLength(fi
lename,'trace',[1:2000,2020:2040]);
```

using keywords

Several keywords can be used to efficiently read parts of larger files.

To read only the part of a file with SourceX between 1000-2000 and SourceY between 4000-5000 use :

```
[Data,SegyTraceHeader,SegyHeader]=ReadSegyConstantTraceLength(fi
lename,'minmax','SourceX',1000,2000,'minmax','SourceY',4000,5000
]);
```

ReadSu

`ReadSu.m` works similar to ReadSegy.m and the same input parameters can be used. A `SuHeader` can optionally be returned, but as there is no (SEG-Y)-Header information in a SU file it is mostly empty.

```
>> [Data,SuTraceHeaders,SuHeader]=ReadSu(filename);
```

Writing SEG-Y files

WriteSegy

`WriteSegy` can be used to save a matrix of data as a SEG-Y formatted file.

Specify values for the SGY Header

Here `dt` is a scalar and `Inline` , `Crossline` , `X` and `Y` are arrays of values of size size(data,2)

```
>> WriteSegy('datacube.segy',data,
    'dt',.004,'Inline3D',Inline,'Crossline3D',Crossline,
    'cdpX',X,'cdpY',Y);
```

Specify revision

```
>> WriteSegy('test.segy',seisdata,'revision',0); % SEG-Y Revision 0
>> WriteSegy('test.segy',seisdata,'revision',1); % SEG-Y Revision 1
```

Specify data sample format

See Data Sample Formats for a list of valid and supported values for the datasample format `dsf` .

```
>> % Force Revision 1 and IEEE Floating point :
>> WriteSegy('test.segy',seisdata,'dsf',5,'revision',1);
>>
>> % Force Revision 0 and IBM Floating point :
>> WriteSegy('test.segy',seisdata,'dsf',1,'revision',0);
```

WriteSegyStructure

`WriteSegyStructure` can be used to write a seismic data to disk given that both `SegyHeader` , `SegyTraceheaders` and the data `Data` are known. They can be obtained using ReadSegy like ;

```
>> [Data,TraceHeaderInfo,SegyTraceHeaders,SegyHeader]=ReadSegy('
data.segy');
```

To write the data using `WriteSegyStructure` simply do

```
>> WriteSegyStructure('datacube.segy',SegyHeader,SegyTraceHeader
s,Data);
```

Force revision

```
>> % Revision 0
>> WriteSegyStructure('datacube.segy',SegyHeader,
                      SegyTraceHeaders,Data,'revision',0);
>> % Revision 1
>> WriteSegyStructure('datacube.segy',SegyHeader,
                      SegyTraceHeaders,Data,'revision',1);
```

Force Data Sample Format

See Data Sample Formats for a list of valid and supported values for the datasample format `dsf` .

```
>> % To force the use of SEG Y revision 0 and data sampling form
at IEEE :
>> WriteSegyStructure('datacube.segy',SegyHeader,
                       SegyTraceHeaders,Data,'revision',1,'dsf',
5);
```

WrityeSegyTraceHeaderValue

`WriteSegyTraceHeaderValue.m` writes one trace header from an array into the Trace Hader of a SGY file.

using keyword

To read a read, edit and write the 'cdp' header values (see Trace Header Definitions for a list of defined keys) use for example:

```
cdp=ReadSegyTraceHeaderValue(file,'key','cdp');  % READ CDP
cdp=cdp+10;                                      % change CDP
WriteSegyTraceHeaderValue(file,cdp,'key','cdp'); % UPDATE CDP
```

using location+precision

To manually update a trace header at a specific location, using a specific data type (precision) use for for example:

```
% Update all trace header values starting at position 72, in integer32
% format, to the values in array 'data'
ntraces=311;
data=[1:1:311]*10;
WriteSegyTraceHeaderValue(filename,data,'pos',72,'precision','int32');
d_header=ReadSegyTraceHeaderValue(filename,'pos',72,'precision','int32');
```

Take a look at Trace eEader Definictions to find the position of all trace header values.

Misc

Visualization

wiggle

`wiggle.m` is used to plot seismic data using using wiggle or variable area type plotting, optionally on top of an image plot of the data

wiggle type:

```
[Data,STH,SH]=ReadSegy('841_m.sgy','jump',10);
                      % get from
                      % http://gdr.nrcan.gc.ca/seismtlitho/archi
ve/le/stacks_fgp_e.php
wiggle([STH.TraceNumber],SH.time,Data,'wiggle',700);
```

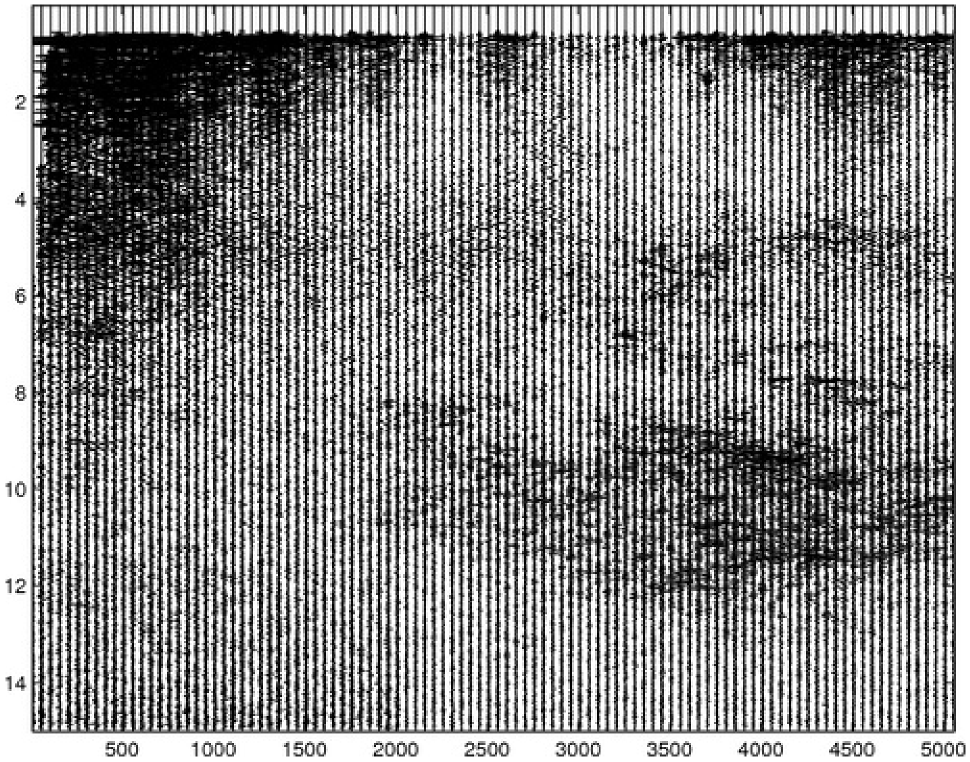


Figure: Wiggle plot

Variable area:

```
[Data,STH,SH]=ReadSegy('841_m.sgy','jump',10,'minmax','TraceNumb
er',3500,4000,'trange',8,10);
wiggle([STH.TraceNumber],SH.time,Data,'VA',700);
```

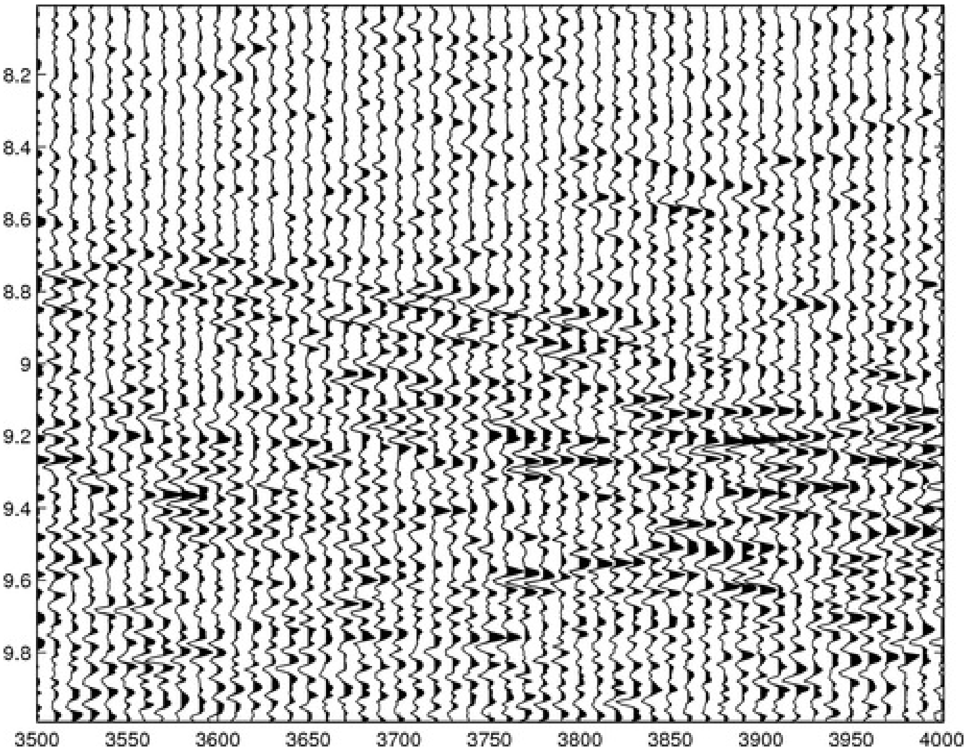


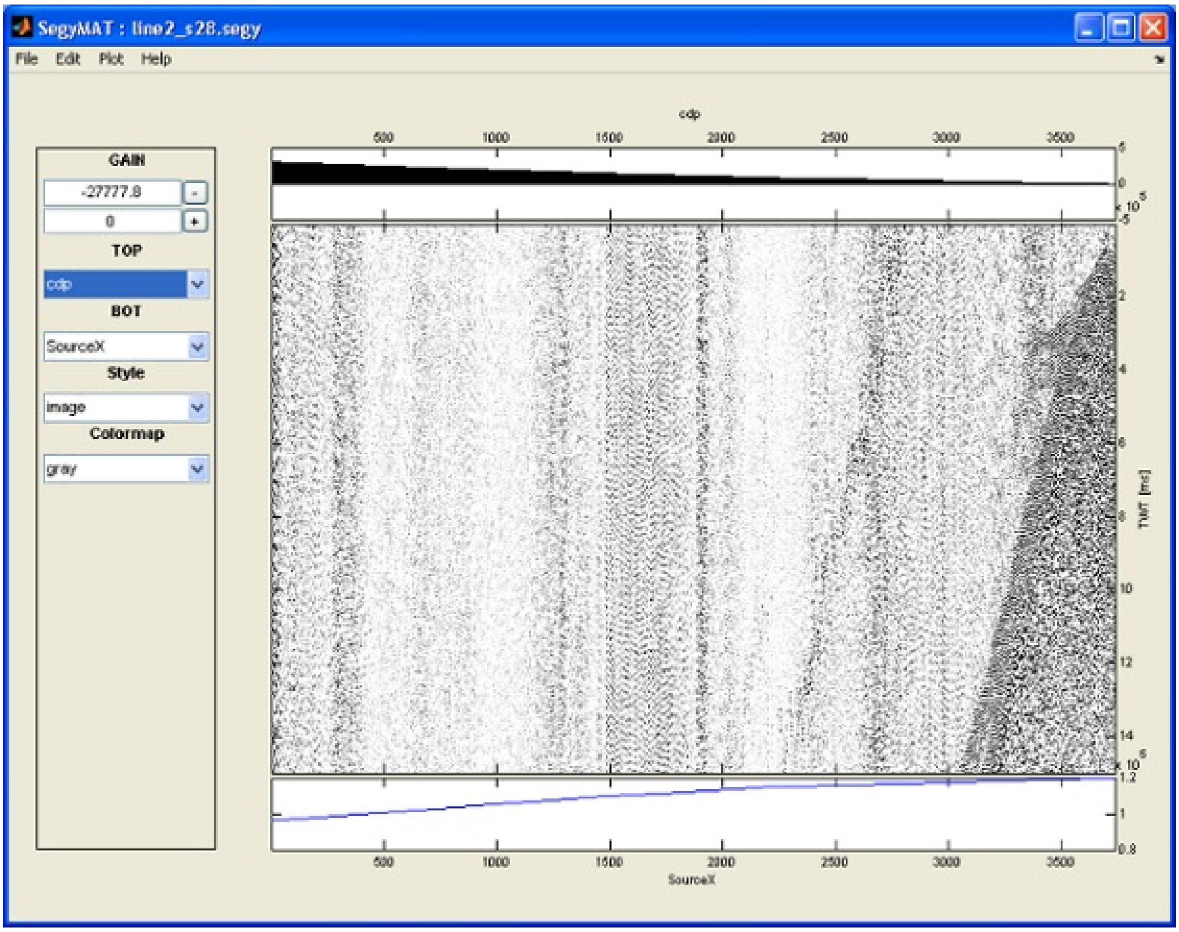
Figure: Wiggle plot

Graphical User Interface utilities

A simple graphical user interface has been implemented in Matlab (Note : this section is unsupported in Octave).

SEGYMAT GUI

Calling `segymat` opens a graphical user interface for viewing and editing SGY formatted files:



SegyMAT : line2_s28.segy

GAIN

-27777.8

0

TOP

cdp

BOT

SourceX

Style

image

Colormap

gray

Figure: Editing the SGY binary header - segymat GUI

Keyboard shortcuts:

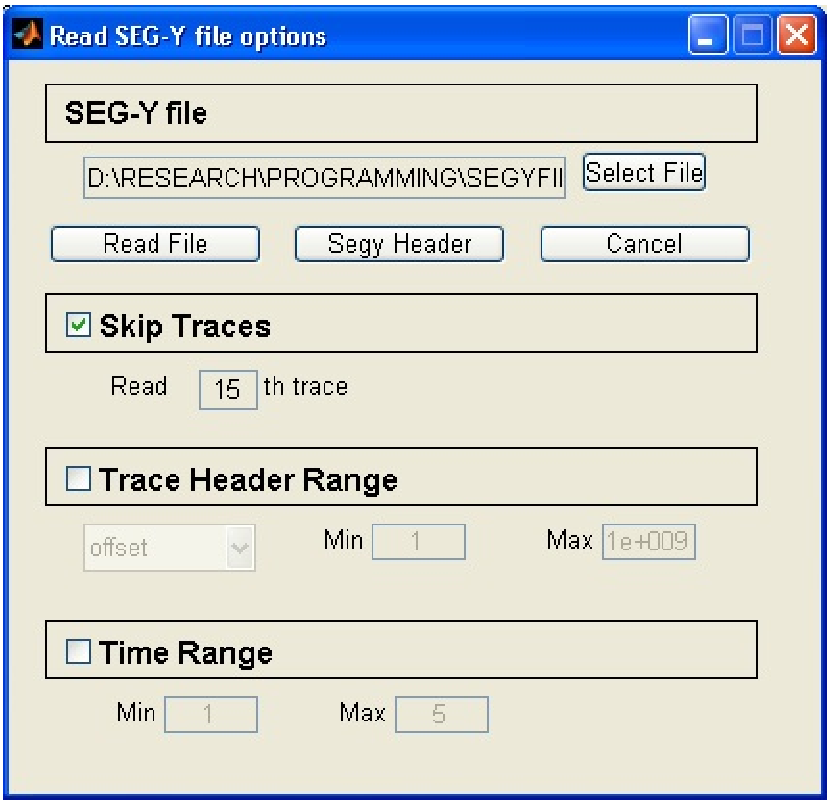
Shortcut	Action
+	Increase gain
-	Decrease gain
4	Pan left
6	Pan right
2	Pan down
8	Pan up
1	Pan down/left
3	Pan down/right
7	Pan up/left
9	Pan up/right
5	Center
a / arrow left	Zoom in
z / arrow right3	Zoom out
h	toggle hiding plotting preferences

simple reading SEG-Y files

Select File->Open to select a SEG-Y file, which will be read using the original SEG-Y header information.

expert reading SEG-Y files

Select File->Open(expert) to handle SEG-Y header values prior to reading the file, and to read in only part of the SEG-Y file.



SEG-Y file

D:\RESEARCH\PROGRAMMING\SEGYFII

☑ Skip Traces

Read 15 th trace

□ Trace Header Range

offset Min 1 Max 1e+009

□ Time Range

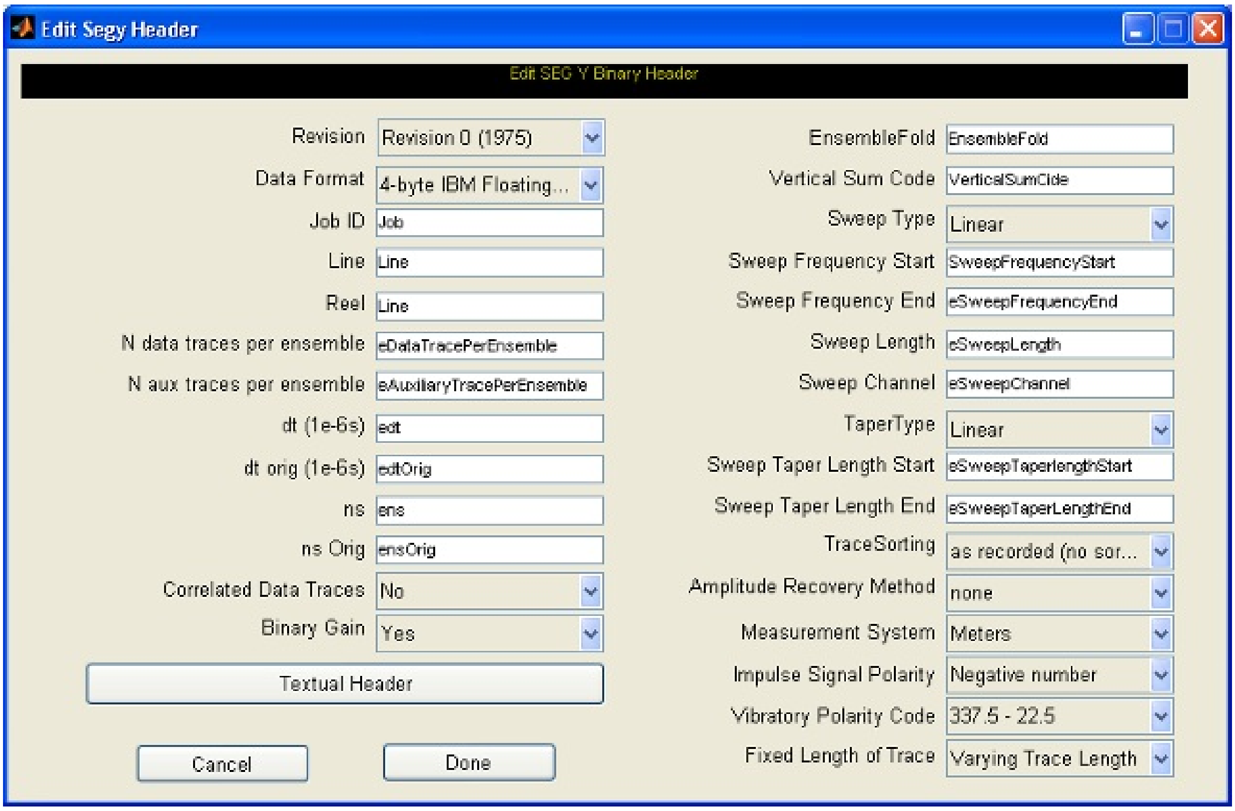
Min 1 Max 5

Figure: Editing the SGY binary header - segymat GUI

Editing the SGY header

`GUIEditSegyHeader` is a GUI for editing the SGY header.

```
[Data,STH,SH]=ReadSegy('841_m.sgy');
SH=GUIEditSegyHeader(SH);
```



Revision	Revision 0 (1975)
Data Format	4-byte IBM Floating...
Job ID	Job
Line	Line
Reel	Line
N data traces per ensemble	eDataTracePerEnsemble
N aux traces per ensemble	eAuxiliaryTracePerEnsemble
dt (1e-6s)	edt
dt orig (1e-6s)	edtOrig
ns	ens
ns Orig	ensOrig
Correlated Data Traces	No
Binary Gain	Yes
EnsembleFold	EnsembleFold
Vertical Sum Code	VerticalSumCide
Sweep Type	Linear
Sweep Frequency Start	SweepFrequencyStart
Sweep Frequency End	eSweepFrequencyEnd
Sweep Length	eSweepLength
Sweep Channel	eSweepChannel
TaperType	Linear
Sweep Taper Length Start	eSweepTaperlengthStart
Sweep Taper Length End	eSweepTaperLengthEnd
TraceSorting	as recorded (no sor...
Amplitude Recovery Method	none
Measurement System	Meters
Impulse Signal Polarity	Negative number
Vibratory Polarity Code	337.5 - 22.5
Fixed Length of Trace	Varying Trace Length

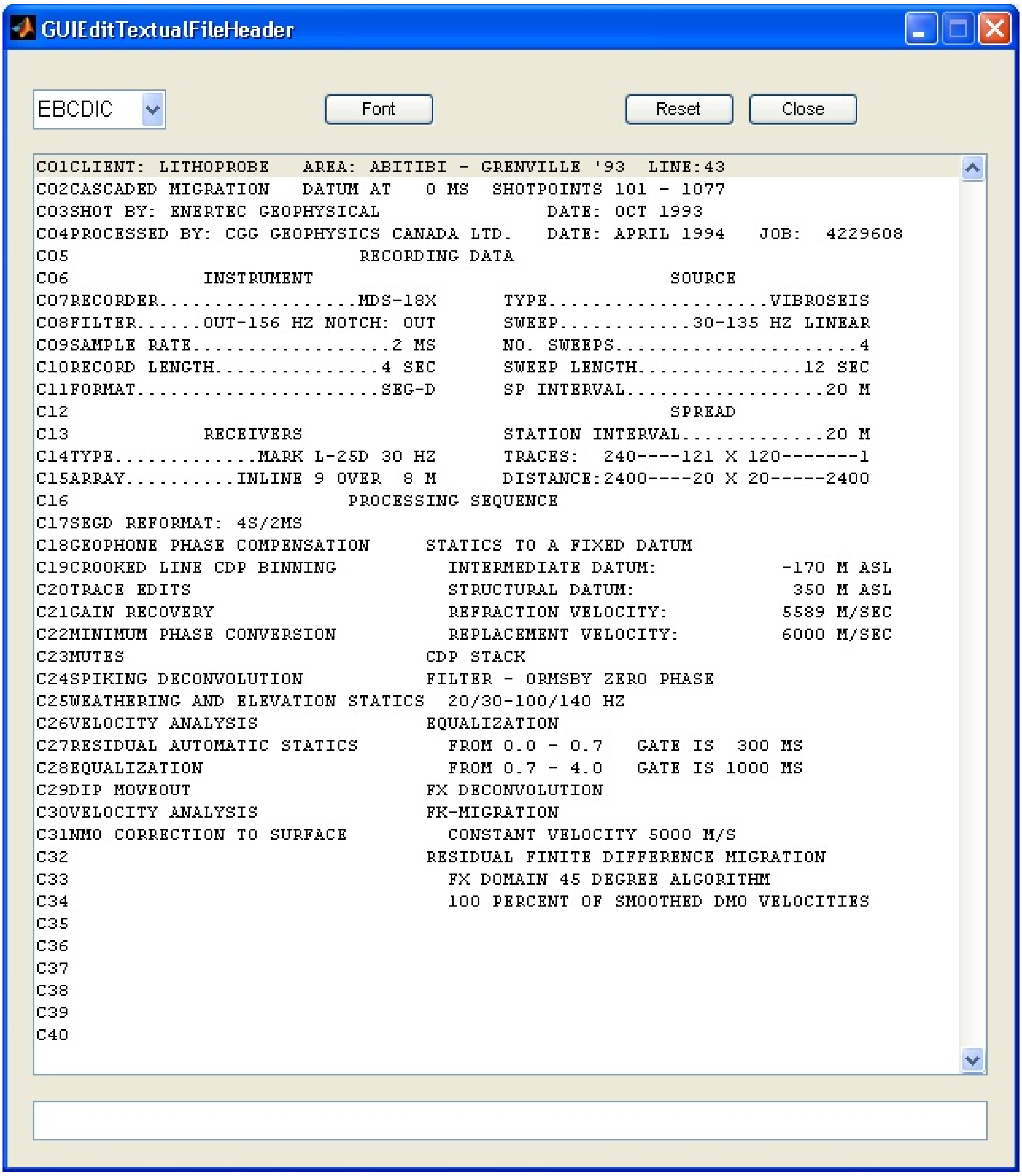
Figure: Editing the SGY binary header - segymat GUI

From this GUI it is possible to view and edit the Textual File Header (Editing the SGY header)

Viewing the textual file header

`GUIEditTextualFileHeader` is a GUI for viewing the textual file header (either in ASCII of EBCDIC format) [editing is not yet implemented].

```
[Data,STH,SH]=ReadSegy('841_m.sgy');
SH=GUIEditTextualFileHeader(SH);
```



```
C01CLIENT: LITHOPROBE   AREA: ABITIBI - GRENVILLE '93  LINE:43
C02CASCADED MIGRATION   DATUM AT   0 MS  SHOTPOINTS 101 - 1077
C03SHOT BY: ENERTEC GEOPHYSICAL               DATE: OCT 1993
C04PROCESSED BY: CGG GEOPHYSICS CANADA LTD.   DATE: APRIL 1994   JOB:  4229608
C05                          RECORDING DATA
C06            INSTRUMENT                                SOURCE
C07RECORDER..................MDS-18X      TYPE....................VIBROSEIS
C08FILTER......OUT-156 HZ NOTCH: OUT      SWEEP............30-135 HZ LINEAR
C09SAMPLE RATE..................2 MS      NO. SWEEPS......................4
C10RECORD LENGTH...............4 SEC      SWEEP LENGTH...............12 SEC
C11FORMAT......................SEG-D      SP INTERVAL..................20 M
C12                                                      SPREAD
C13            RECEIVERS                  STATION INTERVAL.............20 M
C14TYPE.............MARK L-25D 30 HZ      TRACES:  240----121 X 120-------1
C15ARRAY..........INLINE 9 OVER  8 M      DISTANCE:2400----20 X 20-----2400
C16                         PROCESSING SEQUENCE
C17SEGD REFORMAT: 4S/2MS
C18GEOPHONE PHASE COMPENSATION     STATICS TO A FIXED DATUM
C19CROOKED LINE CDP BINNING          INTERMEDIATE DATUM:           -170 M ASL
C20TRACE EDITS                       STRUCTURAL DATUM:              350 M ASL
C21GAIN RECOVERY                     REFRACTION VELOCITY:          5589 M/SEC
C22MINIMUM PHASE CONVERSION          REPLACEMENT VELOCITY:         6000 M/SEC
C23MUTES                           CDP STACK
C24SPIKING DECONVOLUTION           FILTER - ORMSBY ZERO PHASE
C25WEATHERING AND ELEVATION STATICS  20/30-100/140 HZ
C26VELOCITY ANALYSIS               EQUALIZATION
C27RESIDUAL AUTOMATIC STATICS        FROM 0.0 - 0.7   GATE IS  300 MS
C28EQUALIZATION                      FROM 0.7 - 4.0   GATE IS 1000 MS
C29DIP MOVEOUT                     FX DECONVOLUTION
C30VELOCITY ANALYSIS               FK-MIGRATION
C31NMO CORRECTION TO SURFACE         CONSTANT VELOCITY 5000 M/S
C32                                RESIDUAL FINITE DIFFERENCE MIGRATION
C33                                  FX DOMAIN 45 DEGREE ALGORITHM
C34                                  100 PERCENT OF SMOOTHED DMO VELOCITIES
C35
C36
C37
C38
C39
C40
```

Figure: Editing the SGY binary header - segymat GUI

This GUI is integrated into `GUIEditSegyHeader` (Editing the SGY header).

Acknowledgment

Thanks to Brian Farrelly, Norsk Hydro Research Centre, Bergen, Norway, for supplying functions (`num2ibm.m` and `ibm2num.m`)to convert between IBM Floating Point format and doubles. (June, 2002. ver 0.35 ->)

Thanks to Urs Boeringer for adding a patch to WriteSegy, to enable use of an arbitrary set of TraceHeader values. (March 2007. ver 1.08 ->)

`sacsun2mat` was written by F Tilmann whos based his work on sac_sun2pc_mat by C. D. Saragiotis. from Matlab Central.

Thanks to Sourceforge and githubfor hosting the project.

Revisions

Version	Date	Changes
1.6	2016-10-10	Updated manual (switched to gitbook and small bug-fixes'.
1.5	2011-10-28	ReadSegy.m : Added option 'traces' that allow fast reading of specific traces. When the 'minmax' option is used, the corresponding traces are first located through header files, and then data are read using the 'traces' options. For larger files this cause the reading time to decrease significantly when using the 'minmax' option.
1.4	2011-04-05	ReadSegyHeader.m : Fixed 'SkipData' conflict with Robust Control Toolbox. Disabled Waitbar. wiggle.m : Allowed specification of line color. Allow overlaying wiggle plots. Allow NaN value in 'VA' style plotting. ReadSu : Fixed typo in line 221. MergeSegy.m : Added mfile to merge Segy Files.
1.3	2011-01-20	Added 'ReadSegyTraceHeaderValue' and 'WriteSegyTraceHeaderValue' that can be used to read and write the TraceHeaderValues one by one. Much faster that reading the whole dataset.
1.2	2009-01-01	Updated GUI to work for Matlab R2008a. Enabled loading of partial segyfile (using time and header ranges) from GUI (ctrl X). Enabled editing of the textual file header (both ASCII and EBCDIC)
1.11	2003-08-01	Kristian Stormark contributed a change to GetSegyTraceHeader that reduce the number of disc operations causing a significant speed up.
1.08	2007-03-01	Urs Boeniger contributed a patch that allows arbitrary SegyTraceHeaders to be specified for WriteSegy.m.
1.06		Fixed a bug that casue a fixed length of 5011 samples in ReadSu.
1.02		Cleaning up code to work with Octave 2.1.57.
1.01		'jump' related fixes.
1.00	2004-11-15	Cleaning up some Matlab 7.0 specific bugs.

M-file Reference

CheckSegyTraceHeader

```
  SegyTraceHeader=CheckSegyTraceHeader(SegyTraceHeader);

  Checks that all fields of the SegyTraceHeader is set.
  If not, they are initialized.
```

Contents

```
  SegyMAT : A toolbox to read, write and manipulating SEG Y form
atted files
  Version 1.00

  New Features.
    README

  Main
    ReadSegy          - Reads Segy File
    ReadSegyHeader    - Reads SegyHeader from Segy File
    ReadSegyFast      - Reads Segy File in fast mode. No header
values will be read.
    WriteSegy         - Write Segy formatted data
    WriteSegyStructure- Write Segy formatted data using SegyMAT
data structures

    ReadSu            - Reads a SU formatted file.
    ReadSuFast        - Reads a SU formatted file in fast mode.
No header values will be read.
    WriteSu           - Write SU formatted data
    WriteSuStructure  - Write Su formatted data using SegyMAT da
ta structures

  Lower Level IO
```

```
    GetSegyHeader          - Reads the segyheader of a SEGY Y fo
rmatted file
    GetSegyHeaderBasics    - Default Segy header settings

    GetSegyTrace.m         - Read Segy Trace Header and Data fro
m filehandle
    GetSegyTraceHeader     - Read Segy Trace Header from filehan
dle
    GetSegyTraceData       - Read Segy Trace Data from filehandl
e

    PutSegyHeader          - Write Segy Header to filehandle
    PutSegyTrace           - Write Segy Trace Header and Data to
 filehandle

    InitSegyTraceHeader    - Initalize all fields in the SegyTra
ceheader
    CheckSegyTraceHeader.m - Check a SegyTraceHeader for all req
uired fields


  SU<-> SEG-Y conversion
    Su2Segy - Convert SU formatted files to SEG Y
    Segy2Su - Convert SEG Y formatted files to SU

  Plotting
    wiggle  - wiggle/variable area/image plotting of seismic dat
a

  Misc
    ibm2num - Convert IBM 32 bit floatto double
    num2ibm - Convert IEEE 754 doubles to IBM 32 bit floating po
int format
    ebcdic2ascii - convert ebcdic to ascii format
    SegymatVerbose - controls amount of info written to screen
    SegymatVersion - Return the current SegyMAT version


  Seismic Processing :
    SegyMAT_GAIN : 'agc' and 'power' gain.
  (C) 2001-2004 Thomas Mejer Hansen, tmh@gfy.ku.dk/thomas@cultpe
nguin.com



    Overloaded methods:
       serial/Contents
       mmreader/Contents
       VideoReader/Contents
       instrument/Contents
       dioline/Contents
       digitalio/Contents
       daqdevice/Contents
       daqchild/Contents
       aochannel/Contents
       analogoutput/Contents
       analoginput/Contents
       aichannel/Contents
       rsmd/Contents
       resultset/Contents
       drivermanager/Contents
       driver/Contents
       dmd/Contents
       dbtbx/Contents
       database/Contents
       cursor/Contents
       videosource/Contents
       videoinput/Contents
       imaqdevice/Contents
       imaqchild/Contents
       rfmodel.Contents
       rfdata.Contents
       rfckt.Contents
       rfchart.Contents
```

GetSegyHeader

```
  GetSegyHeader : Reads the segyheader of a SEGY Y formatted file

  Call :
   [SegyHeader]=GetSegyHeader(segyid);

   segyid can be a filehandle or a filename


  (C) 2001-2004 Thomas Mejer Hansen, tmh@gfy.ku.dk/thomas@cultpenguin.com



     This program is free software; you can redistribute it and/or modify
     it under the terms of the GNU General Public License as published by
     the Free Software Foundation; either version 2 of the License, or
     (at your option) any later version.

     This program is distributed in the hope that it will be useful,
     but WITHOUT ANY WARRANTY; without even the implied warranty of
     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
     GNU General Public License for more details.

     You should have received a copy of the GNU General Public License
     along with this program; if not, write to the Free Software
     Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
```

GetSegyHeaderBasics

```
  GetSegyHeaderBasics : Default Segy Header Header settings

  Call :
  Rev=GetSegyHeaderBasics
```

GetSegyTrace

```
  GetSegyTrace : Reads a seg y trace, data and header

  [SegyTraceHeader,SegyData]=GetSegyTrace(segyid,TraceStart,Data
Format,ns);
```

GetSegyTraceData

```
  GetSegyTraceData : Get Segy trace data if filehandle

  Call :

    tracedata=GetSegyTraceData(segyid,ns,SegyHeader)
```

GetSegyTraceHeader

```
  GetSegyTraceHeader : Reads a seg y trace, data and header

  [SegyTraceHeader]=GetSegyTraceHeader(segyid,TraceStart);


  (C) 2001-2012 Thomas Mejer Hansen, thomas.mejer.hansen@gmail.com

  Revisions:
  07/2008 Kristian Stormark (<kristian.stormark@gmail.com>) : Reduce the
          number of disc operations causing a significant speed up

  03/2012 Cleaned up after suggestion from Kristian Stormark
```

GetSegyTraceHeaderInfo

```
  GetSegyTraceHeaderInfo : Returns a array of a SEGY Y TraceHeader value

  Call :
  [value]=GetSegyHeaderInfo(SegyTraceHeaders,header)

  header is a header value like 'cdp','dt','TraceNumber'
```

InitSegyTraceHeader

```
  InitSegyTraceHeaders : returns an empty SegyTraceHeader structure

  EX:
  SegyTraceHeader=InitSegyTraceHeader(ns,dt);
```

MakeXmlRef

MergeSegy

```
  MergeSegy : Merge multiple SEGY files

  Example :
     MergeSegy('*.sgy','merge.sgy')

     f{1}='file1.sgy';
     f{2}='file2.sgy';
     f{3}='file3.sgy';
     MergeSegy(f,'merge.sgy')


  Note: All imput segy files must have the same constant trace l
ength
        The SEGY header of the merged SEGY file will be the SEGY
 header
        form the first input SEGY file.
```

PutSegyHeader

```
  PutSegyHeader : Writes SEG-Y header to disk.
  PutSegyHeader(segyid,SegyHeader)

  (C) 2001-2004, Thomas Mejer Hansen, tmh@gfy.ku.dk/thomas@cultp
enguin.com

     This program is free software; you can redistribute it and/
or modify
     it under the terms of the GNU General Public License as pub
lished by
     the Free Software Foundation; either version 2 of the Licen
se, or
     (at your option) any later version.

     This program is distributed in the hope that it will be use
ful,
     but WITHOUT ANY WARRANTY; without even the implied warranty
 of
     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See t
he
     GNU General Public License for more details.

     You should have received a copy of the GNU General Public L
icense
     along with this program; if not, write to the Free Software
     Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  0
2111-1307  USA
```

PutSegyTrace

```
  PutSegyTrace(segyid,tracedata,SegyTraceHeader,SegyHeader)
  Write a SegyTrace to a filehandle 'segyid'

  (C) 2001-2004, Thomas Mejer Hansen, tmh@gfy.ku.dk/thomas@cultp
enguin.com
```

ReadSegy

```
  ReadSegy : Reads a SEG Y rev 1 formatted file

  Call :
  [Data,SegyTraceHeaders,SegyHeader]=ReadSegy(filename);

  To read time slice 0.5<t<5 :
  [Data,SegyTraceHeaders,SegyHeader]=ReadSegy(filename,'trange',
.5,3);
  To read time trace number 100,110 and 150 :
  [Data,SegyTraceHeaders,SegyHeader]=ReadSegy(filename,'traces',
[100 110 150]);
  Skip every 5th trace :
  [Data,SegyTraceHeaders,SegyHeader]=ReadSegy(filename,'jump',5)
;
  Read data in a CDP header range : 5000<cdp<5800 :
  (change cdp to any other valid TraceHeader value)
  [Data,SegyTraceHeaders,SegyHeader]=ReadSegy(filename,'minmax',
'cdp',5000,5800);
  Read only the header values (Data will return empty)
  [Data,SegyTraceHeaders,SegyHeader]=ReadSegy(filename,'SkipData
',1);

  SEG-Y format revision number can be '0' (1975) or
  '100' (similar to '1') (2002).
  By default the SEG-Y format revision number is read in the
  binary header, but this can be overruled using :
  [Data,SegyTraceHeaders,SegyHeader]=ReadSegy(filename,'revision
',0);

  Read using a specific Data Sample Format :
  Rev 0, IBM FLOATING POINT
  [Data,SegyTraceHeaders,SegyHeader]=ReadSegy(filename,'revision
',0,'dsf',1);
  Rev 1, IEEE FLOATING POINT
  [Data,SegyTraceHeaders,SegyHeader]=ReadSegy(filename,'revision
',1,'dsf',5);
```

```
  A SegyHeader can be forced on the SEG-Y file using :
  [Data,SegyTraceHeaders,SegyHeader]=ReadSegy(filename,'SegyHead
er',SegyHeader);
  The SegyHeader can be obtain by GetSegyHeader(segyfilename), a
nd
  then edited.

  To read using little endian :
  [Data,SegyTraceHeaders,SegyHeader]=ReadSegy(filename,'endian',
'l');

  Combine any combination of the above
  [Data,SegyTraceHeaders,SegyHeader]=ReadSegy(filename,'jump',1,
'minmax','cdp',5300,5400);



  Plot the data using e.g.
  imagesc([SegyTraceHeaders.cdp],SegyHeader.time,Data);
  wiggle([SegyTraceHeaders.TraceNumber],SegyHeader.time,Data);

  (C) 2003-2012, Thomas Mejer Hansen, thomas.mejer.hansen@gmail.
com
```

ReadSegyConstantTraceLength

```
  ReadSegy : Reads a SEG Y rev 1 formatted file, and forces Cons
tant Trcae Length

  Call :
  [Data,SegyTraceHeaders,SegyHeader]=ReadSegyConstantTraceLength
(filename);

  See ReadSegy for optional arguments

  See also : ReadSegy
```

ReadSegyFast

```
  ReadSegyFast : Reads a SEG Y rev 1 formatted file, without header values (faster than ReadSegy)

  Call :
  [Data]=ReadSegyFast(filename);
  and equivalent to :
  [Data]=ReadSegy(filename);


  Read only the data of a SegFile - NOT Their headers.
  Much faster than ReadSegy

  'minmax', 'skip'
```

ReadSegyHeader

```
  ReadSegyHeader : Reads a SEG Y Binary Header

  Call :
  [SegyHeader]=ReadSegyHeader(filename);

  To read using little endian :
  [SegyHeader]=ReadSegyHeader(filename,'endian','l');
```

ReadSegyTrace

```
  ReadSegyTrace
```

ReadSegyTraceHeaderValue

```
  ReadSegyTraceHeaderValue : Read a spedicifc trace header value

  Call:
     % By Name
     cdp=ReadSegyTraceHeaderValue(filename,'key','cdp');
     SourceX=ReadSegyTraceHeaderValue(filename,'key','SourceX');
     SourceY=ReadSegyTraceHeaderValue(filename,'key','SourceY');

     % By location in Trace Header
     SourceX=ReadSegyTraceHeaderValue(filename,'pos',72,'precisi
on','int32');

     % Call 'TraceHeaderDef(1)' to see a list of TraceHeader 'ke
y' names

  See also WriteSegyTraceHeaderValue, TraceHeaderDef
```

ReadSu

```
  ReadSu : Reads a SU formatted file (Seismic Unix)

  Call :
  [Data,SuTraceHeaders,SuHeader]=ReadSu(filename);

  To read in big endian format (default):
  [Data,SuTraceHeaders,SuHeader]=ReadSu(filename,'endian','b');
  To read in little endian format :
  [Data,SuTraceHeaders,SuHeader]=ReadSu(filename,'endian','l');



  To read in trace data as 'int32' :
  [Data,SuTraceHeaders,SuHeader]=ReadSu(filename,'DataFormat','int32');
  To read time slice 0.5<t<5 :
  [Data,SuTraceHeaders,SuHeader]=ReadSu(filename,'trange',.5,3);
  Skip every 5th trace :
  [Data,SuTraceHeaders,SuHeader]=ReadSu(filename,'jump',5);
  Read data in a CDP header range : 5000<cdp<5800
  (change cdp to any other valid TraceHeader value)
  [Data,SuTraceHeaders,SuHeader]=ReadSu(filename,'minmax','cdp'5000,5800);

  Combine any combination of the above
  [Data,SuTraceHeaders,SuHeader]=ReadSu(filename,'jump',1,'minmax','cdp',5300,5400);
```

ReadSuFast

```
  ReadSuFast

  PURPOSE : reads a SEISMIC section i  SU format in big endian f
ormat,
            strips the headers and returns the field in the matr
ix seis.
            If nx==0 and nt<>0, nx will be computed
            If nt==0 and nx<>0, nt will be computed

  Call : function seis=ReadSuFast(fileid,nt,nx,'byteorder');
            byteorder : 'l' for little or 'b' for big endian (De
fault : Native )

  BY : TMH 1/8 1997
  Updated by Thomas Mejer Hansen : 22-03-1999
```

Sac2Segy

```
  Sac2Segy : Reads SAC formatted data into a SegyMAT (SGY) struc
ture

  CALL :
    [Data,SegyTraceHeader,SegyHeader]=Sac2Segy(files_in,segyfile
_out,varargin)

    files_in : Either a single filename or a strcture of filenam
es
             files_in='d1.SAC';
             or
             files_in{1}='d1.SAC';
             files_in{2}='d2.SAC';

  Examples :
    [D,STH,SH]=Sac2Segy('','test.segy','FixedLengthTraceFlag',1)
;
               converts all SAC files into one SEGY file (test.s
egy), using
               a FixedLengthTraceFlag of 1. This is compatible w
```

```
ith mosty
               any SEGY reader.

    [D,STH,SH]=Sac2Segy('','test.segy','FixedLengthTraceFlag',0)
;
               converts all SAC files into one SEGY file (test.s
egy), using
               a FixedLengthTraceFlag of 0, allowing varying tra
ce length of SEGY files
               This is only compatible with revision 1 of the SE
GY format.

    [D,STH,SH]=Sac2Segy('file.sac');
               convert file.sac to file.segy

    [D,STH,SH]=Sac2Segy('file.sac','another_file.segy');
               convert file.sac to another_file.segy


    Force little endian byte format for SAC file:
    Sac2Segy('file.sac','test.sgy','endian','l');

  Relies on sac2mat.m

  Download SAC files from : http://www.iris.edu/hq/ssn/events
```

Segy2Su

```
  Segy2Su : Converts SEGY file to SU format

  Call : Segy2Su(filename,ReadSegyOption)
     Replaces the filename suffix to '.su';
     'ReadSegyOptions' are the same as to 'ReadSegy'

   See also : ReadSegy
```

SegyMAT_GAIN

```
  SegyMAT_GAIN : Gain plugin for SegyMAT

  [Data,SegyTraceHeaders,SegyHeader]=SegyMAT_GAIN(Data,SegyTraceHeaders,SegyHeader,varargin);

  ex. AGC using AGC window of 100 ms :
  [Data]=SegyMAT_GAIN(Data,SegyTraceHeaders,SegyHeader,'agc',.1);
  ex. apply t^(pow), pow=2
  [Data]=SegyMAT_GAIN(Data,SegyTraceHeaders,SegyHeader,'pow',2);



  (C) Thomas Mejer Hansen (thomas@cultpenguin.com), 2002
```

SegyMATdemo1

```
  SegyMATdemo1 : Creates, Reads and plots a Segy File;
```

SegymatHelp

SegymatRevision

```
  SegymatRevision - Returns the revision history

  Call : [Revision]=SegymatRevision
```

SegymatVerbose

```
  SegymatVerbose : Writes out verbose information to the screen



  Call :
    SegymatVerbose(text,verboselevel)
    prints out 'text' to screen if verboselevel is higher than threshold
    set in m-file.
```

SegymatVersion

```
  SegymatVersion - Returns the version and release date

  [ver,d]=SegymatVersion;
```

Su2Segy

```
  SU2Segy : Converts SEGY file to SU format
```

TraceHeaderDef

```
  TraceHeaderDef : Defines names, position, and precision for Trace Headers

  % To get a Matlab structure with trace header definitions call:
  STH==TraceHeaderDef;
  % To get a list fo trace header definision listed on the screen call:
  STH==TraceHeaderDef(1)

  See also: ReadSegyTraceHeaderValue, WriteSegyTraceHeaderValue
```

WriteSegy

```
  WriteSegy : writes data to disk using SEGY REV 1 standard.

  EX
  WriteSegy('datacube.segy',data,'dt',.004,'Inline3D',Inline,'Crossline3D',Crossline,'cdpX',X,'cdpY',Y);

  to use a specific SEG revision use :
  WriteSegy('test.segy',seisdata,'revision',0); % SEG-Y Revision 0
  WriteSegy('test.segy',seisdata,'revision',1); % SEG-Y Revision 1

  to use a specific Data Sampling Format use :
  WriteSegy('test.segy',seisdata,'dsf',1); % IBM FLAOTING POINT

  Forice Revision 1 and IEEE Floating point :
  WriteSegy('test.segy',seisdata,'dsf',5,'revision',1);

  See also : WriteSegyStructure, WriteSu, WriteSuStructure
```

WriteSegyStructure

```
  WriteSegyStructure : writes data to disk using SEGY REV 0 and
1 standards.

  EX
  WriteSegyStructure('datacube.segy',SegyHeader,SegyTraceHeaders
,Data);

  To force the use of SEG Y revision 0
  WriteSegyStructure('datacube.segy',SegyHeader,SegyTraceHeaders
,Data,'revision',0);
  To force the use of SEG Y revision 1
  WriteSegyStructure('datacube.segy',SegyHeader,SegyTraceHeaders
,Data,'revision',1);
  To force the data sampling format to be IBM Floating Point
  WriteSegyStructure('datacube.segy',SegyHeader,SegyTraceHeaders
,Data,'dsf',1);

  To force the use of SEG Y revision 0 and data sampling format
IEEE :
  WriteSegyStructure('datacube.segy',SegyHeader,SegyTraceHeaders
,Data,'revision',1,'dsf',5);

  See the dokumentation for for proper values of 'dsf'
```

WriteSegyTrace

```
  WriteSegyTrace

  Call :
    [Data,SegyTraceHeader,SegyHeader]=WriteSegyTrace(filename,tr
aces,Data,SegyTraceHeader,SegyHeader);


    Example :
    %% EXAMPLE : Change polarity of trace 10 and 12
    itrace=[10,12];
    [D,STH,SegyHeader]=ReadSegy(filename,'traces',itrace);
    WriteSegyTrace(filename_copy,itrace,D,STH,SegyHeader)
```

WriteSegyTraceHeaderValue

```
  WriteSegyTraceHeaderValue : Write trace header valaue at specific location

  Call:

     % Update all trace header values starting at position 72, in integer32
     % format, to the value 30
     data=30;
     WriteSegyTraceHeaderValue(filename,data,'pos',72,'precision','int32',);

     % Update all trace header values starting at position 72, in integer32
     % format, to the values in array 'data'
     ntraces=311;
     data=[1:1:311]*10;
     WriteSegyTraceHeaderValue(filename,data,'pos',72,'precision','int32');
     d_header=ReadSegyTraceHeaderValue(filename,'pos',72,'precision','int32');

     % Update the 'cdp' TraceHeader value:
     cdp=ReadSegyTraceHeaderValue(file,'key','cdp');  % READ CDP
     cdp=cdp+10;                                      % change CDP
     WriteSegyTraceHeaderValue(file,cdp,'key','cdp'); % UPDATE CDP

     Call 'TraceHeaderDef(1)' to see a list of TraceHeader 'key' names

  See also ReadSegyTraceHeaderValue, PutSegyTraceHeader, TraceHeaderDef
```

WriteSu

```
  WriteSu : writes data to disk using SEGY REV 2 standard.

  EX
  WriteSu('datacube.su',data,'dt',.004,'Inline3D',Inline,'Crossl
ine3D',Crossline,'cdpX',X,'cdpY',Y);

  to use a specific SEG revision use :
  WriteSu('test.su',seisdata,'revision',0); % SEG-Y Revision 0
  WriteSu('test.su',seisdata,'revision',1); % SEG-Y Revision 1

  to use a specific Data Sampling Format use :
  WriteSu('test.su',seisdata,'dsf',1); % IBM FLAOTING POINT

  Forice Revision 1 and IEEE Floating point :
  WriteSu('test.su',seisdata,'dsf',5,'revision',1);
```

WriteSuStructure

```
  WriteSuStructure : writes data to disk using SU-CWP format

  EX
  WriteSuStructure('datacube.segy',SegyHeader,SegyTraceHeaders,D
ata);
```

ascii2ebcdic

```
  ascii2ebcdic : Converts ASCII formatted text to EBCDIC formatt
ed text

  CALL : ebcdic=ascii2ebcdic(ascii);

  ascii  : Array on unsigned integers
  ebcdic : Array on unsigned integers

  (C) 2002-2009, Thomas Mejer Hansen, tmh@gfy.ku.dk/thomas.mejer
.hansen@gmail.com
```

cmap_rwb

ebcdic2ascii

```
  ebcdic2ascii : Converts EBCDIC formatted text to ASCII formatt
ed text

  CALL : ascii=ebcdic2ascii(ebcdic);

  ebcdic : Array on unsigned integers
  ascii  : Array on unsigned integers

  (C) 2002-2004, Thomas Mejer Hansen, tmh@gfy.ku.dk/thomas@cultp
enguin.com
```

gse2segy

ibm2num

```
  ibm2num : convert IBM 32 bit floating point format to doubles
     x=num2ibm(b)
  b is a matrix of uint32
  x is a corresponding matrix of doubles


  See also num2ibm
```

isoctave

```
  isoctave : checks of octave
```

num2ibm

```
  num2ibm : convert IEEE 754 doubles to IBM 32 bit floating poin
t format
     b=num2ibm(x)
  x is a matrix of doubles
  b is a corresponding matrix of uint32

  The representations for NaN and inf are arbitrary

  See also ibm2num
```

pick_line

```
  pick_line : pick a line from a figure;


  Based on doc(ginput);
```

progress_txt

```
 progress_txt : console based progress bar

 Ex1 :
  for i=1:10000;
   progress_txt(i,10000,'Ciao');
  end

 Ex1 :

  for i=1:10;
  for j=1:10;
  for k=1:10;
   progress_txt([i j k],[10 100 1000],'i','j','k');
  end
  end
  end

 TMH/2005, thomas@cultpenguin.com
```

read_gse_int

sac2mat

```
 [SACdata,SeisData,filenames] = SAC2MAT('file1','file2',..., 'f
ilen',endian )

 reads n SAC files file1, file2, filen
 and converts them to matlab
 format. The filenames can contain globbing characters (e.g. *
and ?).
 These are expanded and all matching files loaded.

 files are assumed big endian formatted (e.g. SUN), little endi
an can be
 forced using endian='l': sac2mat('file1.sac','l');
```

```
  SACSUN2MAT( cellarray ) where cellarray={'file1','file2',...,'filen'}
  is equivalent to the standard form.

  SACdata is an n x 1 struct array containing the header variables
          in the same format as is obtained by using MAT function
          of SAC2000.
          SACdata(i).trcLen contains the number of samples.

  SeisData is an m x n array (where m=max(npts1, npts2, ...) )
          containing the actual data.

  filenames is a n x 1 string cell array with the filenames actually read.

  Note that writing

   [SACdata,SeisData] = sac2mat('file1','file2',..., 'filen' ,endian)

  is equivalent to the following sequence

  sac2000
  READ file1 file2 .. filen
  MAT

  (in fact the failure of above sequence to work properly on my
  system motivated this script).


  SAC2MAT was written by F Tilmann (tilmann@esc.cam.ac.uk)
  based on sac_sun2pc_mat  by C. D. Saragiotis (I copied the
  routines doing the actual work from this code but
  used a different header structure and made the routine
  flexible).
  It was tested on MATLAB5 on a PC but
  should work on newer versions, too.
  (C) 2004

  Update 10/2008 by Thomas Mejer Hansen: Merged sac2sun2mat and
sacpc2mat
  into sac2mat.m
```

sacpc2mat

```
  [SACdata,SeisData,filenames] = SACPCMAT('file1','file2',..., '
filen' )

  reads n SAC files file1, file2, filen (SAC files are assumed t
o have
  PC byte order) and converts them to matlab
  format. The filenames can contain globbing characters (e.g. *
and ?).
  These are expanded and all matching files loaded.

  SACPCMAT( cellarray ) where cellarray={'file1','file2',...,'fi
len'}
  is equivalent to the standard form.

  SACdata is an n x 1 struct array containing the header variabl
es
          in the same format as is obtained by using MAT functio
n
          of SAC2000.
          SACdata(i).trcLen contains the number of samples.

  SeisData is an m x n array (where m=max(npts1, npts2, ...) )
          containing the actual data.

  filenames is a n x 1 string cell array with the filenames actu
ally read.

  Note that writing

   [SACdata,SeisData] = sacsun2mat('file1','file2',..., 'filen'
)

  is equivalent to the following sequence

  sac2000
  READ file1 file2 .. filen
  MAT

  (in fact the failure of above sequence to work properly on my
  system motivated this script).


  SACPC2MAT was written by F Tilmann (tilmann@esc.cam.ac.uk)
  based on sac_sun2pc_mat  by C. D. Saragiotis (I copied the
  routines doing the actual work from this code but
  used a different header structure and made the routine
  flexible).
  It was tested on MATLAB5 on a PC but
  should work on newer versions, too.

  (C) 2004
```

sacsun2mat

```
  [SACdata,SeisData,filenames] = SACSUN2MAT('file1','file2',...,
 'filen' )

  reads n SAC files file1, file2, filen (SAC files are assumed t
o have
  SUN byte order) and converts them to matlab
  format. The filenames can contain globbing characters (e.g. *
and ?).
  These are expanded and all matching files loaded.

  SACSUN2MAT( cellarray ) where cellarray={'file1','file2',...,'
filen'}
  is equivalent to the standard form.
  SACdata is an n x 1 struct array containing the header variabl
es
          in the same format as is obtained by using MAT functio
n
          of SAC2000.
          SACdata(i).trcLen contains the number of samples.

  SeisData is an m x n array (where m=max(npts1, npts2, ...) )
          containing the actual data.

  filenames is a n x 1 string cell array with the filenames actu
ally read.

  Note that writing

   [SACdata,SeisData] = sacsun2mat('file1','file2',..., 'filen'
)

  is equivalent to the following sequence

  sac2000
  READ file1 file2 .. filen
  MAT

  (in fact the failure of above sequence to work properly on my
  system motivated this script).


  SACSUN2MAT was written by F Tilmann (tilmann@esc.cam.ac.uk)
  based on sac_sun2pc_mat  by C. D. Saragiotis (I copied the
  routines doing the actual work from this code but
  used a different header structure and made the routine
  flexible).
  It was tested on MATLAB5 on a PC but
  should work on newer versions, too.

  (C) 2004
```

segymat_release_test

testWriteSegy

```
  testWriteSegy : Script to test WriteSegy and WriteSegyStructure
```

wiggle

```
  wiggle : plot wiggle/VA/image plot

  Call
     wiggle(Data); % wiggle plot
     wiggle(Data,scale); % scaled wiggle plot
     wiggle(x,t,Data); % wiggle plt
     wiggle(x,t,Data,'VA') % variable Area (pos->black;neg->tran
sp)
     wiggle(x,t,Data,'VA2') % variable Area (pos->black;neg->red
)
     wiggle(x,t,Data,'wiggle',scale); % Scaled wiggle
     wiggle(x,t,Data,'wiggle',scale,showmax); % Scaled wiggle an
d max
                                                showmax traces.
     wiggle(x,t,Data,'wiggle',scale,showmax,plimage); % wiggle +
 image
     wiggle(x,t,Data,'wiggle',scale,showmax,plimage,caxis); % wi
ggle +
                                                              sc
aled image

  Data : [nt,ntraces]
  x : [1:ntraces] X axis (ex [SegyTraceheaders.offset])
  t : [1:nt] Y axis
  style : ['VA'] : Variable Area
          ['wiggle'] : Wiggle plot
  scale : scaling factor, can be left empty as []
  showmax [scalar] : max number of traces to show on display [de
f=100]
  plimage [0/1] : Show image beneath wiggles [def=0];
  caxis [min max]/[scalar] : amplitude range for colorscale


  MAKE IT WORK FOR ANY X-AXIS !!!
```