CREWES Matlab® toolbox SEG-Y Input/Output update

Kevin W. Hall and Gary F. Margrave

ABSTRACT

The ability to read and write SEG-Y files into Matlab® using the CREWES toolbox has been evolving for a long time. Along the way, at least six separate attempts to code SEG-Y input/output (I/O) have been written by many people. None of this code was able to effectively deal with trace headers, custom trace headers, or with very large SEG-Y files. In addition, outputs were incompatible so that, for example, trace headers read from disk by one set of CREWES code could not be written to an output SEG-Y file using an unrelated set of CREWES code. This lead to user frustration. A major re-write has been undertaken that combines ideas from legacy code in an object-oriented way that can handle very large datasets, trace headers and custom trace headers. Coding has also begun to support SEG-Y revision 2 which was released in January of 2017. Legacy code in the CREWES toolbox is being removed from the toolbox or being re-written as wrappers that call the new code for backwards compatibility. This report provides an overview of the current state of the new code, with examples of how to use it.

INTRODUCTION

The ability to read and write SEG-Y files into Matlab® using the CREWES toolbox has been evolving for a long time. Along the way, at least six separate attempts to code SEG-Y input/output (I/O) have been written and modified by many people, some of whom documented their efforts (Hogan, 2004, Lloyd, 2010). All versions of SEG-Y code were stored concurrently in the CREWES toolbox, but none of it was compatible. This lead to, for example, a SEG-Y file read from disk by one set of CREWES code could not be written to an output SEG-Y file using an unrelated set of CREWES code.

SEG-Y revision 0 (Barry et al., 1975) and revision 1 (Norris and Faichney, 2002) files were fairly well supported by CREWES code with the exception of trace header values. New code developed in 2010 was able to read trace headers (Lloyd et al., 2010), but was never optimized for speed. These codes were able to read IBM floating-point data, but not able to write it. CREWES code up to this point typically tried to read entire SEG-Y files into RAM, which was all right for small files but no good for files that were larger than the available memory on a given computer.

SEG-Y revision 2 (Hagelund and Stewart, 2017) as released in January of 2017, which prompted another look at these issues. A major re-write has been undertaken that combines ideas from legacy code in an object-oriented way that can handle very large datasets, trace headers and custom trace headers. This code aims to be more efficient, more robust, more flexible, and easier to maintain than past code.

Legacy code in the CREWES toolbox is being removed from the toolbox or being re-written as wrappers that call the new code for backwards compatibility. When this project is complete it will no longer be possible to read data with CREWES SEG-Y code only to be unable to write that data using a different set of CREWES SEG-Y code.

This report provides an overview of the current state of the new code. Rather than attempt to exhaustively document all features of code that is still under development, we show code examples that have been tested in Matlab® 2016b and 2017a. These examples can be copied from this document and pasted onto the Matlab® command line. All examples shown in this report are available in the file crewes/segy/examplecode.m

The example code that follows is organized by topic and numbered sequentially by the order in which the item appears in a SEG-Y file. For example, the textual file header examples in Example sections number 1.# and the binary file header examples are in sections 2.#, where # refers to 0) header definition information, 1) new header definition cell array, 2) new header value struct, 3) convert between struct and double, 4) file read and 5), file write.

The educational release of the CREWES Matlab® toolbox can be downloaded from https://www.crewes.org/ResearchLinks/FreeSoftware. The CREWES sponsor release can be downloaded from https://www.crewes.org/ForOurSponsors/Software/matlab/crmatlab.

CLASSES AVAILABLE IN EDUCATIONAL RELEASE

SegyTextHeader

Usage is*: thdr = SegyTextHeader(filename,permission,byteorder,segyrevision,gui)*. See the Appendix for more details.

SegyTextHeader is a class for creating, reading and writing SEG-Y textual file headers. When a SegyTextHeader object is created for an existing SEG-Y file, the code attempts to guess if the text header is ASCII or EBCDIC format. The guess can be manually overridden setting the SegyRevision property after creating a SegyTextHeader object.

```
%TextHeader
%% Example 1.2: Create new textual file header
disp('*** Example 1.2: Create new textual file header ***')
thdr = SegyTextHeader; %Create a new SegyTextHeader object
txthdr=thdr.new; %Create a new SegyRevision 1 (default) textual file
header
thdr.SegyRevision = 0; %Update SegyRevision
txthdr=thdr.new; %Create a new SegyRevision 0 textual file header
thdr.SegyRevision = 2; %Update SegyRevision
txthdr=thdr.new; %Create a new SegyRevision 2 textual file header

%% Example 1.4: Read textual file header from disk
disp('*** Example 1.4: Read textual file header ***')
thdr = SegyTextHeader(insgyfile); %Create a new SegyTextHeader object,
TextFormat is guessed from file
txthdr = thdr.read; %Read textual file header from disk
thdr.TextFormat = 'ascii'; %Update TextFormat
txthdr = thdr.read; %Read textual file header from disk
thdr.TextFormat = 'ebcdic'; %Update TextFormat
txthdr = thdr.read; %Read textual file header from disk

%% Example 1.5: Write textual file header
disp('*** Example 1.5: Write textual file header ***')
```

```
thdr = SegyTextHeader(outsgyfile,'w'); %Create a new SegyTextHeader
object with write permission
thdr.write(thdr.new); %Create and write a new textual file header to
disk
thdr.Permission = 'r'; %Update Permission
txthdr = thdr.read; %Read textual file header from disk
```

SegyBinaryHeader

Usage is: *bhdr = SegyBinaryHeader(filename,permission,byteorder,segyrevision,gui)*. See the Appendix for more details.

SegyBinaryHeader is a class for creating, reading and writing SEG-Y binary file headers. When a SegyBinaryHeader object is created for an existing SEG-Y file, it guesses if the disk file is big ('b') or little ('l') endian byte-order by examining the data sample format code, which should be in the range 1-16. If the byte-order is incorrect, the format code will be read as a number greater than 255. This guess can be manually overridden by setting the ByteOrder property after creating a SegyBinaryHeader object.

Reading and writing a binary header word structure is governed by a binary header definition cell array with four columns, where the first column contains character strings that will be turned into struct fieldnames. The second column contains character strings specifying the data type for the SEG-Y header word. The data type can be any of the following: 'uint8', 'uint16', 'uint24', 'uint32', 'uint64', 'int8', 'int16', 'int24', 'int32', int64', 'ibm32', 'ieee32', and 'ieee64' where 'u' means unsigned, 'int' means integer, ibm means IBM floating point, ieee means IEEE floating point and the trailing number is the number of bits (8 bits in 1 byte) used to store the value. The third column contains the byte location (numeric) of the SEG-Y header word relative to the beginning of the binary file header. Note that the SEG-Y standard starts counting at byte 1, so the first header word, 'Job Identification Number', is at byte 1 in the standard, but this code starts counting at byte zero, so the start byte will be 0 in this case. The fourth column contains a long description of the header word as a character string

```
%% Binary Header
%% Example 2.0: Get binary header definition information
disp('*** Example 2.0: Get binary header definition information ***')
bhdr = SegyBinaryHeader; %Create a new SegyBinaryHeader object
bhdr.SegyRevision = 2; %Update SeGYRevision
bhdr.HdrDef; %Display current header definition cell array
[hwname, byteloc, idx] = bhdr.byte2word(20); %Return information about
header word closest to byte 20
bhdr.HdrDef(idx,:); %Display header word definition for row number idx

%% Example 2.1: Create a new binary header definition
disp('*** Example 2.1: Create new binary header definition ***')
bhdr = SegyBinaryHeader; %Create a new SegyBinaryHeader object
bindef = bhdr.newDefinition; %Create a new SegyRevision=1 (default)
header definition
bindef = bhdr.newDefinition(0); %Create a new SegyRevision 0 header
definition
bindef{50,1} = 'NewField'; %Update header word name for row 50 of the
definition
bhdr.HdrDef = bindef; %Update object's header definition
```

```
bhdr.HdrDef(50,:); %Display row 50 of object's definition

%% Example 2.2: Create a new binary header
disp('*** Example 2.2: Create new binary header ***')
bhdr = SegyBinaryHeader; %Create a new SegyBinaryHeader object
binhdr = bhdr.new; %Create a new binary header struct, using object
defaults
bhdr.SamplesPerTrace = 500; %Override number of samples per trace
bhdr.SampleInterval = 1000; %Override sample interval
[binhdr,bindef] = bhdr.new; %Create a new binary file header struct and
header definition

%% Example 2.4: Read binary file header
disp('*** Example 2.4: Read binary file header ***')
bhdr = SegyBinaryHeader(insgyfile); %Create a new SegyBinaryHeader
object, ByteOrder is guessed from file
binhdr = bhdr.read; %Read binary file header from disk
bhdr.ByteOrder = 'l'; %Update ByteOrder: 'l' is little-endian
binhdr = bhdr.read; %Read binary file header from disk
bhdr.ByteOrder = 'b'; %Update ByteOrder: 'b' is big-endian (SEG_Y
revision 0 and 1 standard)
binhdr = bhdr.read; %Read binary file header from disk

%% Example 2.5: Write binary file header to disk
disp('*** Example 2.5: Write binary file header to disk ***')
thdr = SegyTextHeader(outsgyfile,'w'); %Create a new SegyTextHeader
object with write permission
thdr.write(thdr.new); %Create a new text header and write it to disk
bhdr = SegyBinaryHeader(outsgyfile,'a'); %Create a new SegyBinaryHeader
object with append permission
nsamp = 10; %Samples per trace
sampint = 1000; %Sample interval in microseconds
bhdr.write(bhdr.new(nsamp,sampint)); %Write a new binary file header to
disk, override bhdr.SamplesPerTrace and bhdr.SampleInterval
bhdr.Permission = 'r'; %Update Permission
bhdr.read; %Read binary file header from disk
```

SegyExtendedTextHeader

This is a stub. No code has been written or tested for creating, reading or writing extended textual file headers (if any) for SEG-Y revision 1 and 2 files.

SegyTrace

Usage is: *trc = SegyTrace(filename,permission,fmtcode,byteorder,segyrevision,gui)*. See the Appendix for more details.

SegyTrace is a class for creating, reading and writing SEG-Y trace headers and trace data. When a SegyTrace object is created it uses a temporary SegyBinaryHeader object to get basic information from the binary file header which enables calculation of the number of traces in the file. Some of this information, for example, the number of samples per trace may be incorrect in the binary header. These properties be overridden before conducting read operations. If the files Segy Revision number is 0 and the binary header data sample format code is 1 (4-byte floating point), the class constructor tests the trace data to guess if

it is IBM or IEEE floating point. If the latter, the objects FormatCode is updated to 5 (IEEE 4-byte floating point; see Hall, 2017) At this time, this class does not handle variable length traces, although they can still be read by manually overriding properties such as OFFSET and SamplesPerTrace.

Reading and writing trace headers words is governed by a cell array, similar to the SegyBinaryHeader class, however, a trace header definition cell array has five columns. The first column contains header word name character strings that will be turned into struct fieldnames. The second column contains character strings specifying the data type for the SEG-Y header word. Data types can be any of the following: ‘uint8’, ‘uint16’, ’uint24’, ‘uint32’, ’uint64’, ’int8’, ’int16’, ’int24’, ‘int32’, int64’, ’ibm32’, ’ieee32’, and ’ieee64’ where ‘u’ means unsigned, ‘int’ means integer, ibm means IBM floating point, ieee means IEEE floating point and the trailing number is the number of bits (8 bits in 1 byte) used to store the value. The third column contains the byte location (numeric) of the SEG-Y header word relative to the beginning of the trace header. Note that the SEG-Y standard starts counting at byte 1 so the first header word, ‘Trace Number in Line’, is at byte 1 in the standard, but this code starts counting from zero so the start byte will be 0 in this example. The fourth column contains the name of the scalar header word that will be applied to the current header word if the ApplyCoordScalars property is set to 1 (default). The scalar header word name must also exist in column 1. The fifth column contains a long description of the header word as a character string.

```
%% Trace
%% Example 4.0: Get trace header definition information
disp('*** Example 4.0: Get trace header definition information ***')
trc = SegyTrace; %Create a new SegyTrace object
trc.HdrDef; %Display the current trace header definition cell array
[hwname, byteloc, idx] = trc.byte2word(20); %Get info about header word
closest to byte 20
trc.word2byte('EnsembleNum'); %Get byte number for a given header word
name
trc.HdrDef(idx,:); %Display header word definition for row number idx

%% Example 4.1: Create new trace header definition cell array
disp('*** Example 4.1: Create new trace header definition ***')
trc = SegyTrace; %Create a new SegyTrace object
trc.HdrDef{80,1} = 'NewField'; %Update field name for row 80 of the
definition
trc.HdrDef(80,:) %Display row 80 of HdrDef
trcdef = trc.newDefinition(0); %Create a new SegyRevision 0 header
definition
trcdef = trc.newDefinition(2); %Create a new SegyRrevision 2 header
definition
%Modify trcdef manually or by using uiSegyDefinition (not shown)
trc.HdrDef = trcdef; %Override object's header definition

%% Example 4.2: Create new trace header struct, data array and
definition
disp('*** Example 4.2: Create new trace header struct, data array and
definition ***')
trc = SegyTrace; %Create a new SegyTrace object
trcdef = trc.newDefinition(); %Return a new trace header definition
```

```
[trchdr, trcdat, trcdef] = trc.new(ntrace, nsamp, sampint); %Return new
trace header, trace data and trace definition
trc.SegyRevision = 0; %Update SegyRevision
trcdef = trc.newDefinition(); %Create a new SegyRevision 0 trace header
definition
[trchdr, trcdat, trcdef] = trc.new (ntrace, nsamp, sampint); %Return
new trace header, trace data and trace definition
trc.SegyRevision = 2; %Update SegyRevision
trcdef = trc.newDefinition; %Create a new SegyRevision 2 trace header
definition
[trchdr, trcdat, trcdef] = trc.new (ntrace, nsamp, sampint); %Return
new trace header, trace data and trace definition

%% Example 4.3: Convert between trace header struct and array of
doubles
disp('*** Example 4.3: Convert between trace header struct and array of
doubles ***')
[hdrvals,fieldnames,datatypes] = trc.struct2double(trchdr); %Convert
trace header struct to array of doubles
trchdr = trc.double2struct(hdrvals,fieldnames,datatypes); %Convert
array of doubles to a trace header struct

%% Example 4.4: Read trace headers and data
disp('*** Example 4.4: Read trace headers and data ***')
trc = SegyTrace(insgyfile); %Create a new SegyTrace object
[trchdr,trcdat,trcdef] = trc.read; %Read all trace headers and trace
data and return trace definition used
[trchdr,trchdr,trcdef] = trc.read(10:20); %Read traces 10-20
[trchdr,trchdr,trcdef] = trc.read(20:-1:10); %Read traces 10-20 in
reverse order
[trchdr,trchdr,trcdef] = trc.read([1,5,8]); %Read traces 1, 5 and 8
trchdr = trc.read([1,5,8],'headers'); %Read trace headers only
trcdat = trc.read([1,5,8],'data'); %Read trace data only
tnl = trc.read(1:100,'TrcNumLine'); %Read header word TrcNumLine from
traces 1-100
trc.SegyRevision = 1; %Update SegyRevision
trc.ByteOrder = 'b'; %Update ByteOrder
trc.SamplesPerTrace = 1001; %Update SamplesPerTrace
[trchdr,trcdat] = trc.read; %Read all trace headers and trace data and
return trace definition used
%NOTE, SegyRevision, ByteOrder and SamplesPerTrace are incorrect,
trchdr and trcdat will be garbage!

%% Example 4.5: Write trace headers and data to disk
disp('*** Example 4.5: Write trace headers and data to disk ***')
thdr = SegyTextHeader(outsgyfile,'w'); %Create a new SegyTrace object
with write permission
thdr.write(thdr.new); %Create a new text header and write to disk
bhdr = SegyBinaryHeader(outsgyfile,'a'); %Create a new SegyBinaryHeader
object with append permission
bhdr.write(bhdr.new(nsamp,sampint)); %Create and write a new binary
header to disk
trc = SegyTrace(outsgyfile,'a'); %Create a new SegyTrace object with
append permission
[trchdr,trcdat] = trc.new(ntrace, nsamp, sampint); %Create new trace
header and trace data
trc.write(trchdr, trcdat); %Write trace header and trace data to disk
```

SegyDataTrailer

This is a stub. No code has been written or tested for creating, reading or writing data trailers for SEG-Y revision 2 files.

SegyFile

Usage is *sf = SegyFile(filename,permission,segyrevision,sampint,nsamps,segfmt,... txtfmt,bytorder,bindef,trcdef,gui)*

The SegyFile class provides a convenient interface for SegyTextHeader, SegyBinaryHeader, SegyExtendedTextHeader, SegyTrace and SegyDataTrailer. If the SegyRevision or ByteOrder properties are changed at the top level in a SegyFile object, the change is propagated to all sub-objects.

```
%% SegyFile
%% Example 6.2: Create new file headers and trace data array
disp('*** Example 6.2: Create new file headers, trace data array, and
header definitions ***')
sf = SegyFile; %Create a new SegyFile object
[txthdr, binhdr, exthdr, trchdr, trcdat, bindef, trcdef] = sf.new;
%Create new file and trace headers, trace data and header defintions
[txthdr, binhdr, exthdr, trchdr, trcdat] = sf.new(ntrace, nsamp,
sampint); %Create new file and trace headers, trace data and header
defintions over-riding nsamp and sampint
%NOTE that exthdr will be empty ([])

%% Example 6.4: Read an existing SEG-Y file
disp('*** Example 6.4: Read SEG-Y file ***')
%In the following examples, exthdr will always be empty (exthdr = []).
This
%is a placeholder for when the ExtendedText Header class has been fully
written and tested.u
if exist('uiSegyFile.m','file')
    sf = uiSegyFile(insgyfile); %Create a new SegyFile object and
inspect SEG-Y file with GUI (sponsors toolbox release only)
end
sf = SegyFile(insgyfile); %Create a new SegyFile object
[txthdr, binhdr, exthdr, trchdr, trcdat, bindef, trcdef] = sf.read;
%Read entire file
[txthdr, binhdr, exthdr, trchdr, trcdat, bindef, trcdef] =
sf.read(1:2); %Read traces 1 and 2
txthdr = sf.TextHeader.read; %Read just the textual file header
binhdr = sf.BinaryHeader.read; %Read just the binary header
exthdr = sf.ExtendedTextHeader.read; %Read just the extended text file
header(s)
trchdr = sf.Trace.read(1,'headers'); %Read trace header 1
trchdr = sf.Trace.read([],'headers'); %Read all trace headers
trcdat = sf.Trace.read(1:2:10,'data'); %Read data from traces 1-10 by
two's
trcdat = sf.Trace.read([],'data'); %Read all trace data
[trchdr,trcdat] = sf.Trace.read; %Read all trace headers and trace data

%% Example 6.5: Write a new SEG-Y file: PROBLEMS
disp('*** Example 6.5: Write new SEG-Y file ***')
```

```
sf = SegyFile(insgyfile); %Create a new SegyFile object
[txthdr, binhdr, exthdr, trchdr, trcdat, bindef, trcdef] = sf.read;
%Read entire file
gui=1; %Command line prompts
% gui=[]; %GUI prompts
sf2 = SegyFile(outsgyfile,'w',sf.SegyRevision,sf.SampleInterval,...
    sf.SamplesPerTrace,sf.FormatCode,sf.TextFormat,sf.ByteOrder,...
    bindef,trcdef,gui); %Create a new SegyFile object with write
permissions
sf2.write(txthdr, binhdr, exthdr, trchdr, trcdat, ...
    bindef, trcdef); %Write SEG-Y file to disk
```

Wrappers

A number of wrapper functions are provided for convenience and for backwards compatability:

```
%% Wrappers
%% Example 7.0: Get trace header information
disp('*** Example 7.0: Get trace header information ***')
tracebyte2word(232); %Get header name closest to byte 20, revision 1
trace header definition (default)
tracebyte2word(232,0); %Get header name closest to byte 20, revision 0
trace header definition
tracebyte2word(232,2); %Get header name closest to byte 20, revision 2
trace header definition
traceword2byte('Unassigned01'); %Get byte location for header name,
revision 1 trace header definition (default)
traceword2byte('Unassigned14',0); %Get byte location for header name,
revision 0 trace header definition
traceword2byte('TrcHdrName',2); %Get byte location for header name,
revision 2 trace header definition
traceheaderdump(trchdr); %List all header names in trchdr struct that
contain non-zero values
[dump,words,inotempty]=traceheaderdump(trchdr); %Return all header
values, names and indices in trchdr struct for non-zero values
traceheaderdump_g(trchdr); %Display GUI that can plot values for up to
three separate header words

%% Example 7.4: Read an SEG-Y file
disp('*** Example 7.4: Read SEG-Y file ***')
%Read SEG-Y file with no overrides, display uiSegyFile() GUI if it
exists
[trcdat,segyrev,sampint,fmtcode,txtfmt,bytord,txthdr,...
        binhdr,exthdr,trchdr,bindef,trcdef] = ...
        readsegy(insgyfile);
%Read SEG-Y file using all available overrides
%Update input paramters:
trcrange = []; %Empty => all traces
gui = 1; %Command line prompts, no GUI
nsamps = []; %Empty => Determine number of samples per trace from file
on disk
%Read SEG-Y file using all overrides, display uiSegyFile() GUI if it
exists
[trcdat,segyrev,sampint,fmtcode,txtfmt,bytord,txthdr,binhdr,exthdr,trch
dr,...
```

```
    bindef,trcdef] =
readsegy(insgyfile,trcrange,segyrev,sampint,nsamps,...
    fmtcode,txtfmt,bytord,bindef,trcdef,gui);

%% Example 7.5: Write a new SEG-Y file
disp('*** Example 7.5: Write SEG-Y file ***')
writesegy(outsgyfile,trcdat); %Write Seg-Y revision 1 file using just
trcdat and defaults
writesegy(outsgyfile,trcdat,segyrev,sampint,fmtcode,txtfmt, ...
    bytord,txthdr,binhdr,exthdr,trchdr,bindef,trcdef); %Write SEG-Y
file using all overrides
```

ADDITIONAL CODE AVAILABLE IN SPONSOR RELEASE

Two graphical user interfaces (GUI) have been written to aid setting properties for SegyFile() objects. These are called uiSegyFile() and uiSegyDefinition().

uiSegyDefinition

Usage is *hdrdef = uiSegyDefinition() or hdrdef = uiSegyDefinition(hdrdef).*

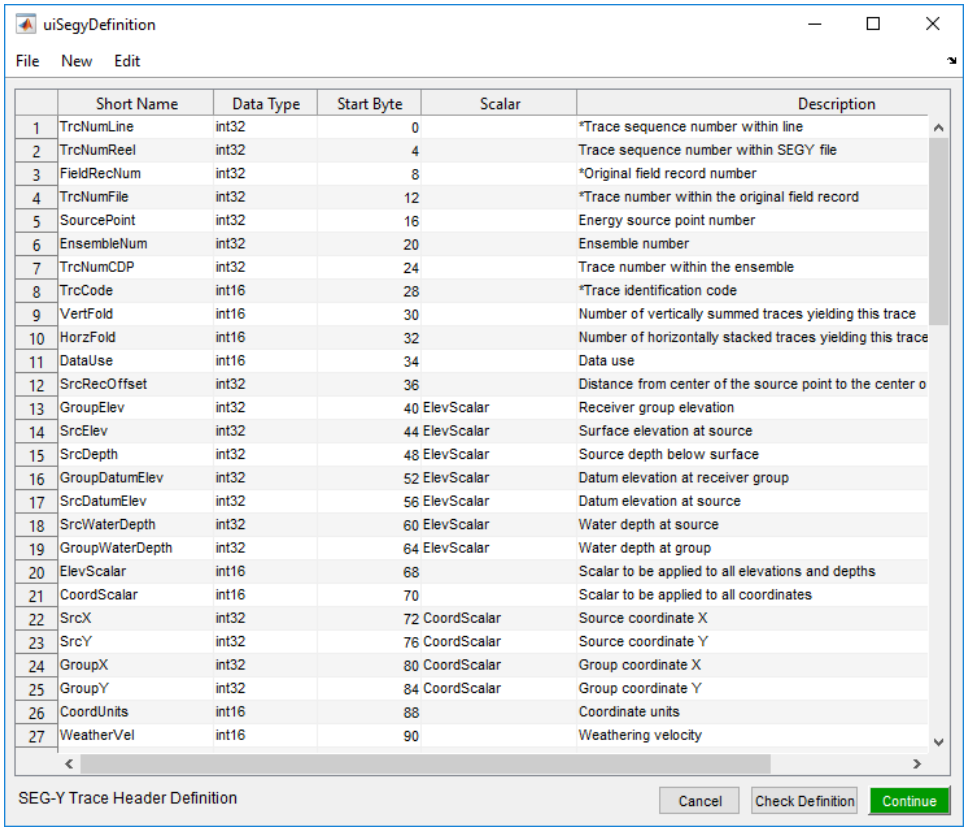
The uiSegyDefinition GUI displays a binary file header or trace header definition cell array in an editable table (Figure 1). Users may check the validity of the definition at any time by clicking the 'Check Definition' button. If the definition is not valid, the 'Continue' button will be disabled and changed from green to a red background color. Three pull-down menus are provided: 1) 'File' allows you to save the current displayed definition to a .mat file on disk, and to read it back into the table, 2) 'New' replaces the current definition with a new one, and 3) 'Edit' allows insertion, deletion and sorting (based on the valued of Start Byte) of rows in the table. The GUI returns the edited table as a header definition cell array when the 'Continue' button is clicked.

uiSegyFile

Usage is *sf = uiSegyFile(filename) or uiSegyFile()*

The uiSegyFile GUI (Figure 2) allows interactive inspection of an existing SEG-Y disk file and returns a SegyFile object when the 'Continue' button is clicked or if 'Close' is selected from the 'File' pull-down menu. Three pull-down menus are available, 1) 'File' allows you to open a different SEG-Y file, 2) 'Edit' calls uiSegyDefinition for either the current binary header definition or the current trace header definition, and 3) 'Plot' plots either a single trace header word or trace data values for up to 100 traces on either side of the current trace, using plotimage(). 'Plot' takes into account the radio buttons for 'Time', 'Frequency', 'Gain Correction', and 'Bandpass Filter' on the 'Trace Display' panel. 'Gain Correction' applies an automatic envelope correction using the aec() function in the CREWES toolbox. 'Bandpass Filter' applies a 10-15-55-60 Hz Ormsby filter using the filtorm() function in the CREWEs toolbox. Bandpass frequency limits are not customizable at this time.

Figures 2 through 7 show some examples of using uiSegyfile() for a correlated Vibroseis source gather file.



	Short Name	Data Type	Start Byte	Scalar	Description
1	TrcNumLine	int32	0		*Trace sequence number within line
2	TrcNumReel	int32	4		Trace sequence number within SEGY file
3	FieldRecNum	int32	8		*Original field record number
4	TrcNumFile	int32	12		*Trace number within the original field record
5	SourcePoint	int32	16		Energy source point number
6	EnsembleNum	int32	20		Ensemble number
7	TrcNumCDP	int32	24		Trace number within the ensemble
8	TrcCode	int16	28		*Trace identification code
9	VertFold	int16	30		Number of vertically summed traces yielding this trace
10	HorzFold	int16	32		Number of horizontally stacked traces yielding this trace
11	DataUse	int16	34		Data use
12	SrcRecOffset	int32	36		Distance from center of the source point to the center o
13	GroupElev	int32	40	ElevScalar	Receiver group elevation
14	SrcElev	int32	44	ElevScalar	Surface elevation at source
15	SrcDepth	int32	48	ElevScalar	Source depth below surface
16	GroupDatumElev	int32	52	ElevScalar	Datum elevation at receiver group
17	SrcDatumElev	int32	56	ElevScalar	Datum elevation at source
18	SrcWaterDepth	int32	60	ElevScalar	Water depth at source
19	GroupWaterDepth	int32	64	ElevScalar	Water depth at group
20	ElevScalar	int16	68		Scalar to be applied to all elevations and depths
21	CoordScalar	int16	70		Scalar to be applied to all coordinates
22	SrcX	int32	72	CoordScalar	Source coordinate X
23	SrcY	int32	76	CoordScalar	Source coordinate Y
24	GroupX	int32	80	CoordScalar	Group coordinate X
25	GroupY	int32	84	CoordScalar	Group coordinate Y
26	CoordUnits	int16	88		Coordinate units
27	WeatherVel	int16	90		Weathering velocity

FIG. 1. GUI displayed for >> trcdef = uiSegyDefinition(sf.Trace.HdrDef)

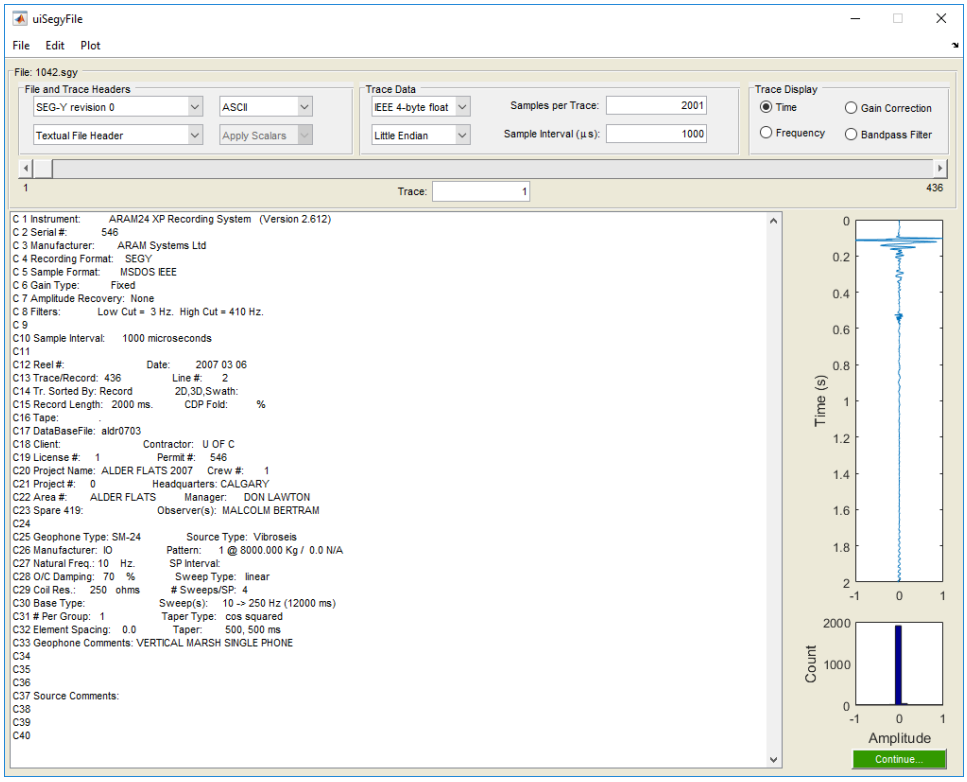


FIG. 2. Textual file header (table) and trace one data (right) are displayed.

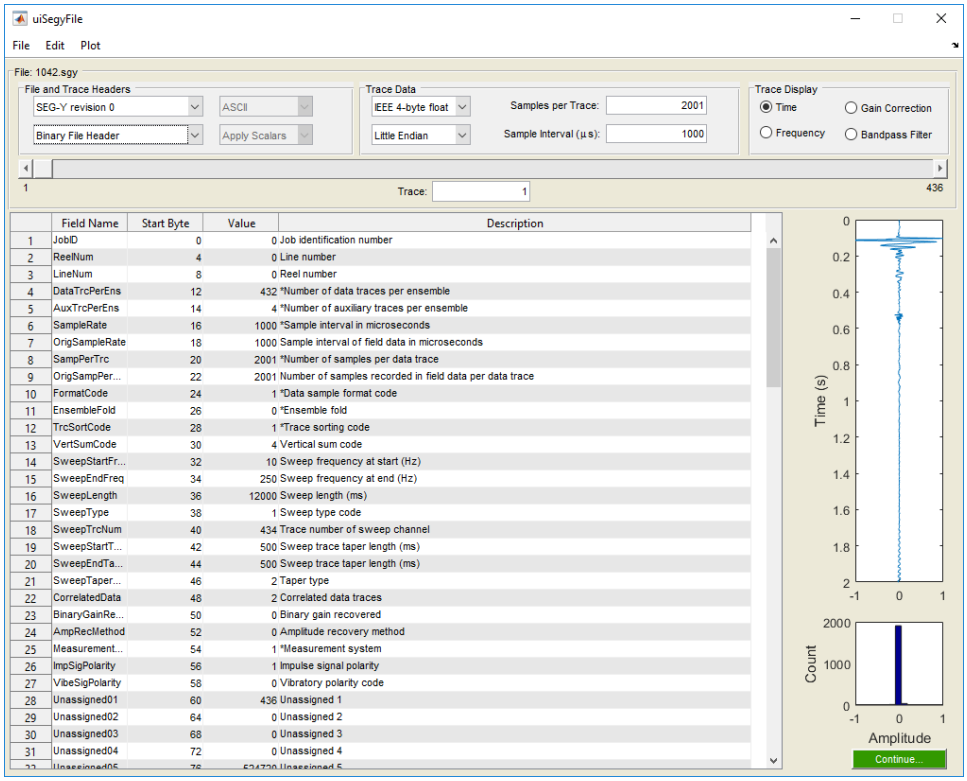


FIG. 3. Binary file header (table) and trace one data read correctly using little-endian byte order.

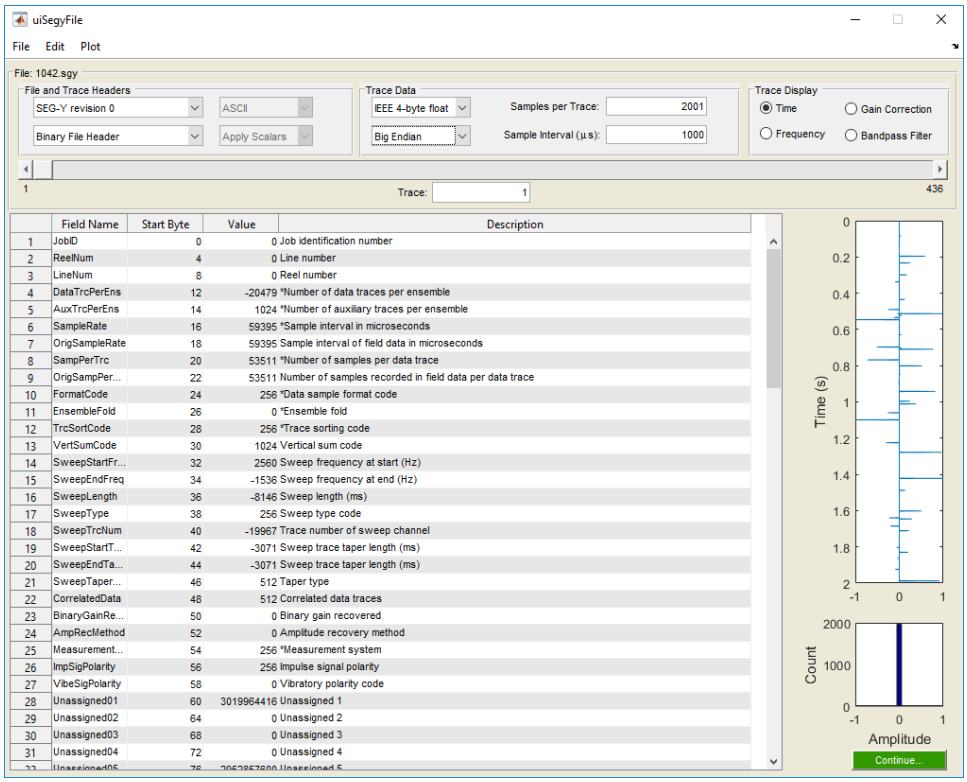


FIG. 4. Binary file header (table) and trace one data read incorrectly using big-endian byte order.

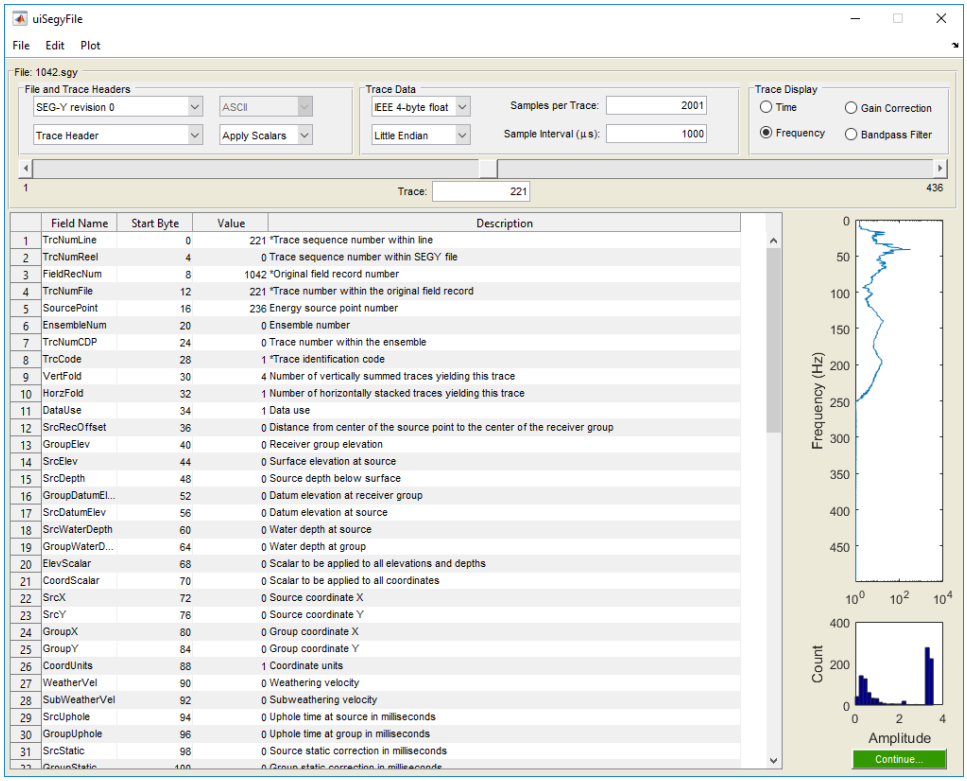


FIG. 5. Trace 221 header (table) and data read correctly using little-endian byte order, trace data displayed in frequency rather than time.

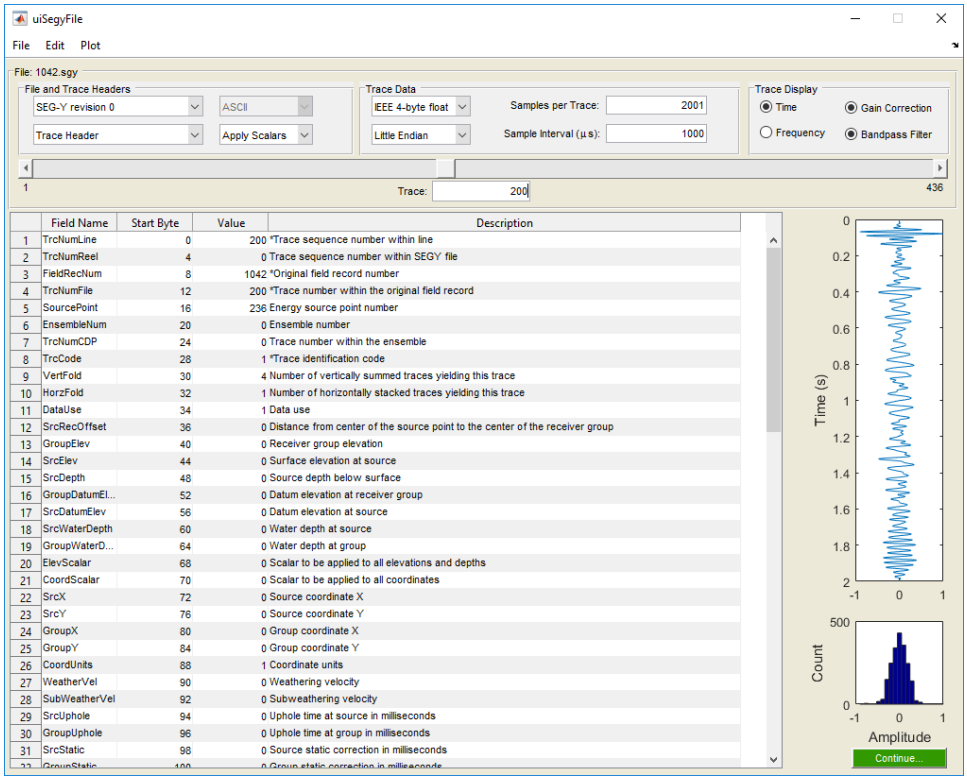


FIG. 6. Trace 200 trace header and trace data displayed with gain correction and bandpass filter.

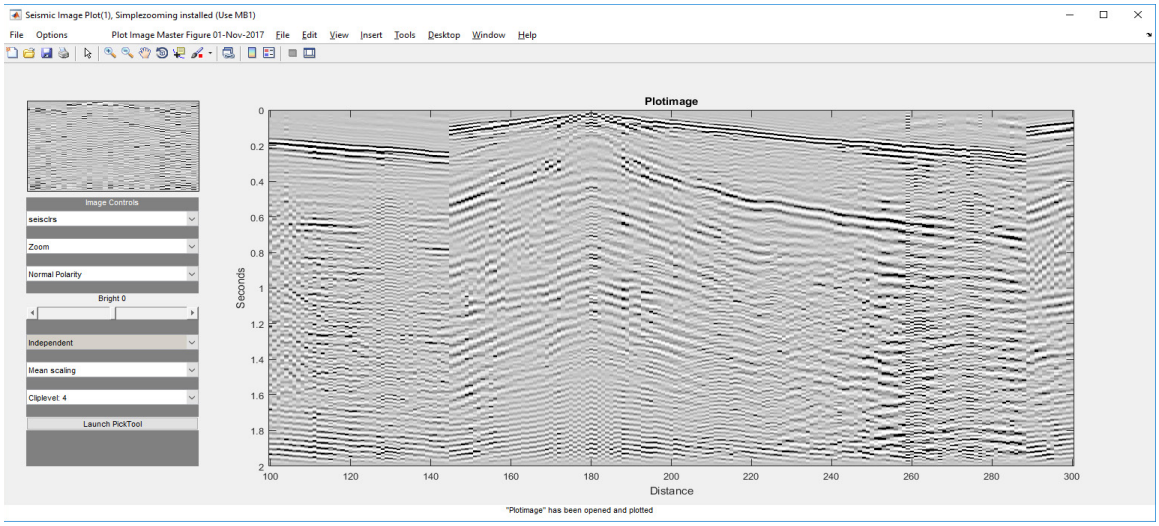


FIG. 7. Traces 100-300 plotted with gain correction and bandpass filter using the Plot->Traces pull-down menu,

CONCLUSIONS AND FUTURE WORK

New object-oriented code to read and write SEG-Y files has been written and tested, but is not yet complete. However, it is more internally consistent, robust, useable and maintainable than any previous code released via the CREWES Matlab® toolbox. Development is on-going. All SEG-Y revision 0, 1 and 2 trace data formats are supported for reading and writing except for format code 4 (Fixed point with gain). Variable trace lengths in a file are not yet supported. SEG-Y revision 1 and 2 extended textual file headers are not yet supported. SEG-Y revision 2 extended trace headers and data trailers are not yet supported.

Future work includes writing code to address the issues listed above, removing incompatible code from the CREWES toolbox and writing wrappers for some older code for backwards compatibility (eg. altreadsey() and altwritesegy()).

ACKNOWLEDGEMENTS

The authors would like to thank everyone who has previously worked on SEG-Y I/O for the CREWES toolbox, including, Henry Bland, Chad Hogan, Heather Lloyd and Carla Osborne. We thank the sponsors of CREWES for continued support. This work was funded by CREWES industrial sponsors and NSERC (Natural Science and Engineering Research Council of Canada) through the grant CRDPJ 461179-13.

REFERENCES

Barry, K. M., Cavers, D. A. and Kneale, C. W., 1975, Report on recommended standards for digital tape formats: Geophysics, 40, no. 02, 344-352.

Hagelund, R., and Stewart, A.L., Eds., SEG Technical Standards Committee, 2017, SEG-Y _r2.0: SEG-Y revision 2.0 Data Exchange Format: SEG, Tulsa, OK, www.seg.org.

Hall, K.W., 2017, Everything you never wanted to know about IBM and IEEE floating point numbers, CREWES Research Report, this volume.

Hogan, C., 2004, SEGY Matlab Tool Documentation, CREWES internal documentation.

Lloyd, H., Hall, K., Margrave, M., 2010, New Matlab® functions for reading, writing and modifying SEG-Y files, CREWES research report, **22**, 58.

Norris, M.W. and Faichney, A.K., Eds., SEG Technical Standards Committee, 2002, SEG-Y rev 1 Data Exchange Format: SEG, Tulsa, OK, www.seg.org.

APPENDIX

The following are class summaries that were auto-generated for the SegyTextHeader, SegyBinaryHeader SegyTrace and SegyFile classes by Matlab® based on help and comment lines contained within the code.

MATLAB File Help: SegyTextHeader	View code for SegyTextHeader	Go to online doc for SegyTextHeader	Default Topics

SegyTextHeader

```
classdef SegyTextHeader

 SEG-Y textual file header class

 Usage: thdr = SegyTextHeader(filename,permission,byteorder,segyrevision)

 Where:
   filename     = string containing full file name or a file id from fopen
   permission   = string containing file permissions to use with fopen
                      (optional), default is 'r' (see help fopen)
   byteorder    = string containing 'b' or 'l' for big- or little-endian
                      (optional), default is 'n' (see help fopen)
   segyrevision = 0, 1, or 2
   gui          = 0: no prompts,
                  1: text prompts
                  figure handle or empty: gui prompts

 NOTE: This SOFTWARE may be used by any individual or corporation for any purpose
 with the exception of re-selling or re-distributing the SOFTWARE.
 By using this software, you are agreeing to the terms detailed in this software's
 Matlab source file.

 Authors: Kevin Hall 2009, 2017

 NOTE: This SOFTWARE may be used by any individual or corporation for any purpose
 with the exception of re-selling or re-distributing the SOFTWARE.
 By using this software, you are agreeing to the terms detailed in this software's
 Matlab source file.
```

Class Details

Superclasses	File
Sealed	false
Construct on load	false

Constructor Summary

SegyTextHeader	classdef SegyTextHeader

Property Summary

ByteOrder	Byte Order, 'l' for little endian, 'b' for big endian
FileID	File handle output from fopen
FileName	FileName corresponding to FileID
GUI	GUI flag: 0: no prompts, 1: text prompts (default), figure handle: gui prompts
OFFSET	Header offset from beginning of file in bytes
Permission	File Permission, eg. 'r', 'w', 'a'
SIZE	Header size in bytes
SegyRevision	SEG-Y revision number: 0, 1 (default), or 2

TextFormat	Text format: 'ascii' or 'ebcdic'

Method Summary

	addlistener	Add listener for event.
	delete	Delete a handle object.
	eq	== (EQ) Test handle equality.
	fclose	'fclose()' with error checking
	findobj	Find objects matching specified conditions.
	findprop	Find property of MATLAB handle object.
	fopen	'fopen()' with error checking
	fprintf	'fprintf()'
	fread	'fread()' with error checking and additional data types
	freopen	'freopen()' with error checking
	fseek	'fseek()' with error checking
	fsize	File size in bytes
	ftell	'ftell()' with error checking
	fwrite	'fwrite()' with error checking and addition datatypes
	ge	>= (GE) Greater than or equal relation for handles.
	gt	> (GT) Greater than relation for handles.
	guessTextFormat	Guess if SEG-Y file textual file header is ASCII or EBCDIC
Sealed	isvalid	Test handle validity.
	le	<= (LE) Less than or equal relation for handles.
	listenByteOrder	ByteOrder has changed, freopen file for file operations
	listenFileName	FileName has changed, freopen file for file operations
	listenPermission	Permission has changed, freopen file for file operations
	lt	< (LT) Less than relation for handles.
	ne	~= (NE) Not equal relation for handles.
	new	Create and populate a new textual file header based on the SEG-Y revision number
	notify	Notify listeners of event.
	read	`Read a textual file header from a SEG-Y file
	write	`Write a textual file header to a SEG-Y file

Event Summary

ByteOrderChanged	Notifies listeners that the ByteOrder has changed
FileNameChanged	Notifies listeners that the FileName has changed
ObjectBeingDestroyed	Notifies listeners that a particular object has been destroyed.
PermissionChanged	Notifies listeners that the Permission has changed

MATLAB File Help: SegyBinaryHeader	View code for SegyBinaryHeader	Go to online doc for SegyBinaryHeader	Default Topics

SegyBinaryHeader

```
classdef SegyBinaryHeader

 SEG-Y binary file header class

 Usage:
   bh = SegyBinaryHeader(filename,permission,byteorder,segyrevision,gui)

 Where:
   filename     = string containing full file name or a file id from fopen
   permission   = string containing file permissions to use with fopen
                  (optional), default is 'r' (see help fopen)
   byteorder    = string containing 'b' or 'l' for big- or little-endian
                  (optional), default is 'n' (see help fopen)
   segyrevision = 0, 1, or 2
   gui          = 0: no prompts,
                  1: text prompts
                  figure handle or empty: gui prompts

 Authors: Kevin Hall, 2009, 2017

 NOTE: This SOFTWARE may be used by any individual or corporation for any purpose
 with the exception of re-selling or re-distributing the SOFTWARE.
 By using this software, you are agreeing to the terms detailed in this software's
 Matlab source file.
```

Class Details

Superclasses	File
Sealed	false
Construct on load	false

Constructor Summary

SegyBinaryHeader	classdef SegyBinaryHeader

Property Summary

ByteOrder	Byte Order, 'l' for little endian, 'b' for big endian
FileID	File handle output from fopen
FileName	FileName corresponding to FileID
GUI	GUI flag: 0: no prompts, 1: text prompts (default), figure handle: gui prompts
HdrDataTypes	HdrDef col 2 (Dependent)
HdrDef	Cell array containing the binary file header definition
HdrFieldNames	HdrDef col 1 (Dependent)
HdrLongNames	HdrDef col 4 (Dependent)
HdrStartBytes	HdrDef col 3 (Dependent)
OFFSET	Header offset from beginning of file in bytes
Permission	File Permission, eg. 'r', 'w', 'a'

SIZE	Header size in bytes
SegyRevision	SEG-Y revision number: 0, 1 (default), or 2

Method Summary

	addlistener	Add listener for event.
	byte2word	Return the header word nearest a byte location in the binary header
	check	Determine if input is struct or cell array and pass control to checkDefinition or checkStruct
	checkDefinition	Check a binary header definition cell array for validity
	checkStruct	Compare a binary header struct to the current binary header definition
	delete	Delete a handle object.
	eq	== (EQ) Test handle equality.
	fclose	'fclose()' with error checking
	findobj	Find objects matching specified conditions.
	findprop	Find property of MATLAB handle object.
	fopen	'fopen()' with error checking
	fprintf	'fprintf()'
	fread	'fread()' with error checking and additional data types
	freopen	'freopen()' with error checking
	fseek	'fseek()' with error checking
	fsize	File size in bytes
	ftell	'ftell()' with error checking
	fwrite	'fwrite()' with error checking and addition datatypes
	ge	>= (GE) Greater than or equal relation for handles.
	gt	> (GT) Greater than relation for handles.
	guessByteOrder	Use format code to guess if file is big or little-endian
Sealed	isvalid	Test handle validity.
	le	<= (LE) Less than or equal relation for handles.
	listenByteOrder	ByteOrder has changed, freopen file for file operations
	listenFileName	FileName has changed, freopen file for file operations
	listenPermission	Permission has changed, freopen file for file operations
	listenSegyRevision	SegyRevision has changed, reset the header definition
	lt	< (LT) Less than relation for handles.
	ne	~= (NE) Not equal relation for handles.
	new	Create and populate a new binary header struct for use with write()
	newDefinition	Create a new binary header definition based on the SEG-Y revision number
	notify	Notify listeners of event.
	read	'Read a binary header struct from a SEG-Y file using the current binary header definition
	readExtSampleInterval	Read extended sample interval [rev 2] (Does not use the Header Definition)
	readExtSamplesPerTrace	Read extended samples per trace [rev 2] (Does not use the Header Definition)
	readExtTracesPerRec	Read extended traces per record [rev 2] (Does not use the Header

		Definition)
	readFileInfo	Read basic information from a SEG-Y file (Does not use header defintion)
	readFixedTrcLength	Read fixed trace length flag [rev 1+] (Does not use the Header Definition)
	readFormatCode	Read data sample format code (Does not use the Header Definition)
	readHeaderWord	Read a single header word from a given byte position and data type
	readIntegerConstant	Read integer constant [rev 2] (Does not use the Header Definition)
	readNumDataTrailers	Read number of data trailers [rev 2] (Does not use the Header Definition)
	readNumExtHeaders	Read number of extended textual file headers [rev 1+] (Does not use the Header Definition)
	readNumExtTrcHeaders	Read number of extended trace headers [rev 2] (Does not use the Header Definition)
	readSampleInterval	Read sample interval (Does not use the Header Definition)
	readSamplesPerTrace	Read samp es per trace (Does not use the Header Definition)
	readSegyRevision	Read Segy Revision [rev 1+] (Does not use the Header Definition)
	readTraceOneOffset	Read trace one offset [rev 2] (Does not use the Header Definition)
	readTracesPerRec	Read traces per record (Does not use the Header Definition)
	write	'Write a binary file header structure to a SEG-Y file

Event Summary

ByteOrderChanged	Notifies listeners that the ByteOrder has changed
FileNameChanged	Notifies listeners that the FileName has changed
ObjectBeingDestroyed	Notifies listeners that a particular object has been destroyed.
PermissionChanged	Notifies listeners that the Permission has changed
SegyRevisionChanged	Notifies listeners that the SegyRevision has changed

MATLAB File Help: SegyTrace View code for SegyTrace Go to online doc for SegyTrace Default Topics

SegyTrace

```
classdef SegyTrace

 Class for SEGY Traces (Headers and Data)

 Usage:
   trc = SegyTrace(filename,permission,fmtcode,byteorder,segyrevision,gui)

 Where:
   filename     = string containing full file name or a file id from fopen
   permission   = string containing file permissions to use with fopen
                  (optional), default is 'r' (see help fopen)
   byteorder    = string containing 'b' or 'l' for big- or little-endian
                  (optional), default is 'n' (see help fopen)
   segyrevision = 0, 1, or 2
   gui          = 0: no prompts,
                  1: text prompts
                  figure handle or empty: gui prompts

 Authors: Kevin Hall, 2009, 2017

 NOTE: This SOFTWARE may be used by any individual or corporation for any purpose
 with the exception of re-selling or re-distributing the SOFTWARE.
 By using this software, you are agreeing to the terms detailed in this software's
 Matlab source file.
```

Class Details

Superclasses	File
Sealed	false
Construct on load	false

Constructor Summary

SegyTrace	classdef SegyTrace

Property Summary

ApplyCoordScalars	Apply Coordinate Scalars, default=true
ByteOrder	Byte Order, 'l' for little endian, 'b' for big endian
BytesPerSample	Bytes Per Sample, eg. uint8 is 1, uint32 is 4,...
ExtSampleInterval	Extended Sample interval [rev 2], default = 0
ExtSamplesPerTrace	Extended Samples Per Trace [rev2], default = 0
ExtTracesPerRec	Extended Traces Per Record [rev 2], default = 0
FileID	File handle output from fopen
FileName	FileName corresponding to FileID
FixedTrcLength	Fixed Trace Length flag [rev 1+], default = 1
FormatCode	Format Code, default = 5, IEEE 4-byte float
FormatCodeType	Text equivalent to the numeric Format Code
GUI	GUI flag: 0: no prompts, 1: text prompts (default), figure handle: gui prompts

HdrDataTypes	Header Data Types, HdrDef col 2
HdrDef	Trace Header Definition (cell array)
HdrFieldNames	Trace Header Field Names, HdrDef col 1
HdrLongName	Header Descriptive (Long) name, HdrDef col 5
HdrScalars	Header Scalars, HdrDef col 4
HdrStartBytes	Header Word Start Bytes, HdrDef col 3
IntegerConstant	Integer constant [rev 2], default = 0
NumExtTrcHeaders	Number of Extended Trace Headers [rev 2], default = 0
NumExtTxtHeaders	Number of Extended Textual File headers [rev 1+], default = 0
OFFSET	Bytes from beginning of file to start of trace one
Permission	File Permission, eg. 'r', 'w', 'a'
SIZE	Size of trace header in bytes
SampleInterval	Sample Interval in microseconds, default = 1000
SamplesPerTrace	Samples Per Trace
SegyRevision	SEG-Y revision number, 0, 1 (default), 2
TraceSize	Size of one trace in bytes
TracesInFile	Total number of traces in file
TracesPerRec	Traces Per Record, includes data and aux traces

Method Summary

	addlistener	Add listener for event.
	applyCoordinateScalars	Apply coordinate scalars to trace header values
	byte2word	Return the header word name nearest a byte location in the trace header definition
	check	Call checkDefinition(), checkStruct() or checkArray() as appropriate
	checkArray	Check trace data values to see if they can be stored using the current FormatCode
	checkDefinition	Check a trace header definition cell array for validity
	checkStruct	Compare a trace header struct to the current trace header definition for validity
	delete	Delete a handle object.
Static	double2struct	Converts an array of doubles to a trace header struct
	eq	== (EQ) Test handle equality.
	fclose	'fclose()' with error checking
	findobj	Find objects matching specified conditions.
	findprop	Find property of MATLAB handle object.
	fopen	'fopen()' with error checking
	fprintf	'fprintf()'
	fread	'fread()' with error checking and additional data types
	freopen	'freopen()' with error checking
	fseek	'fseek()' with error checking
	fsize	File size in bytes
	ftell	'ftell()' with error checking
	fwrite	'fwrite()' with error checking and addition datatypes

	ge	>= (GE) Greater than or equal relation for handles.
	gt	> (GT) Greater than relation for handles.
	guessFormatCode	Guess if trace data are IBM32 or IEEE32 for SegyRevision=0 and FormatCode=1
Sealed	isvalid	Test handle validity.
	le	<= (LE) Less than or equal relation for handles.
	listenByteOrder	ByteOrder has changed, freopen file for file operations
	listenFileName	FileName has changed, freopen file for file operations
	listenPermission	Permission has changed, freopen file for file operations
	lt	< (LT) Less than relation for handles.
	ne	~= (NE) Not equal relation for handles.
	new	Returns a new trace data matrix (all zeros) and a new trace header struct
	newDefinition	Returns a new trace header definition cell array based on SegyRevision
Static	newHeaderWord	Returns a new zero-filled vector of the appropriate data type
	notify	Notify listeners of event.
	read	Reads and returns trace data and/or a trace header struct from a SEG-Y file
	removeCoordinateScalars	Remove coordinate scalars from trace header values
Static	struct2double	Converts a trace header struct to an array of doubles
	word2byte	Return the byte location for a given header word name
	word2idx	Returns the index number in HdrDef for a given header word name
	write	Writes a trace data array and a trace header structure to a SEG-Y file

Event Summary

ByteOrderChanged	Notifies listeners that the ByteOrder has changed
FileNameChanged	Notifies listeners that the FileName has changed
ObjectBeingDestroyed	Notifies listeners that a particular object has been destroyed.
PermissionChanged	Notifies listeners that the Permission has changed

MATLAB File Help: SegyFile View code for SegyFile Go to online doc for SegyFile Default Topics

SegyFile

```
function sf = SegyFile(filename,permission,segyrevision,sampint,nsamp,...
                       fmtcode,txtfmt,byteorder,bindef,trcdef,gui)
 Optional Inputs ([] is accepted):
 filename    - SEG-Y disk file name
 permission  - 'r' (default), 'w', or 'a' (see help fopen)
 segyrevision - segy revision (0,1,2); Overrides SEG-Y Binary File Header on
                disk.
 sampint     - Sample interval in (s); Overrides SEG-Y Binary File Header on
                disk.
 nsamps      - Samples per trace; Overrides SEG-Y Binary File Header on
                disk.
 fmtcode     - SEG-Y trace data format code. eg. 1 = IBM float, 5 = IEEE
                float
 txtfmt      - Text format, 'ascii' or 'ebcdic'
 byteorder   - byte order of disk file 'l'=little-endian, 'b'=big-endian
 bindef      - 4 column binary header definition cell array such as provided by
                @BinaryHeader/new; See uiSegyDefinition().
                NOTE! writesegy will require the same bindef unless you
                modify binhdr!
 trcdef      - 5 column trace header definition cell array such as provided by
                @BinaryHeader/new; See uiSegyDefinition()
                NOTE! writesegy will require the same trcdef unless you
                modify trchdr!
 gui         - 0 (no progress bar), 1 (text progress bar),
                [] (default; gui progress bar and warnings), figure handle
                (same as [], but an attempt is made to center GUI popups on
                the figure represented by the figure handle)
 Outputs:
 sf          - A SEG-Y file object

 Example:
   s=SegyFile('file.sgy','r')
   s.TextHeader.read
   s.BinaryHeader.read
   s.Trace.read(1:2:100)

 Authors: Kevin Hall, 2017

 NOTE: This SOFTWARE may be used by any individual or corporation for any purpose
 with the exception of re-selling or re-distributing the SOFTWARE.
 By using this software, you are agreeing to the terms detailed in this software's
 Matlab source file.
```

Class Details

Superclasses	File
Sealed	false
Construct on load	false

Constructor Summary

SegyFile	function sf = SegyFile(filename,permission,segyrevision,sampint,nsamp,...

Property Summary

BinaryHeader	Contains BinaryHeader object
ByteOrder	Byte Order, 'l' for little endian, 'b' for big endian
ExtendedTextHeader	Contains SegyExtendedTextHeader object
FileID	File handle output from fopen
FileName	FileName corresponding to FileID
FormatCode	Data sample format code: 1,2,3,5,6,7,8,9,10,11,12,15,16
GUI	GUI flag: 0: no prompts, 1: text prompts (default), figure handle: gui prompts
Permission	File Permission, eg. 'r', 'w', 'a'
SampleInterval	Sample interval in microseconds
SamplesPerTrace	Number of data samples per trace
SegyRevision	Segy revision number: 0, 1 (default), 2
TextFormat	Text format: 'ascii' or 'ebcdic'
TextHeader	Contains SegyTextHeader object
Trace	Contains SegyTrace object

Method Summary

	addlistener	Add listener for event.
	delete	Delete a handle object.
	eq	== (EQ) Test handle equality.
	fclose	'fclose()' with error checking
	findobj	Find objects matching specified conditions.
	findprop	Find property of MATLAB handle object.
	fopen	'fopen()' with error checking
	fprintf	'fprintf()'
	fread	'fread()' with error checking and additional data types
	freopen	'freopen()' with error checking
	fseek	'fseek()' with error checking
	fsize	File size in bytes
	ftell	'ftell()' with error checking
	fwrite	'fwrite()' with error checking and addition datatypes
	ge	>= (GE) Greater than or equal relation for handles.
	gt	> (GT) Greater than relation for handles.
Sealed	isvalid	Test handle validity.
	le	<= (LE) Less than or equal relation for handles.
	listenByteOrder	ByteOrder has changed, freopen file for file operations
	listenFileName	FileName has changed, fopen file for file operations
	listenFormatCode	FormatCode has changed, update binary header and trace objects
	listenGUI	GUI has changed, update header and trace objects
	listenPermission	Permission has changed, freopen file for file operations
	listenSampleInterval	Sample Interval has changed, update binary header and trace objects
	listenSamplesPerTrace	SamplesPerTrace has changed, update binary header and trace objects
	listenSegyRevision	SegyRevision has changed, update header and trace objects

	listenTextFormat	TextFormat has changed, update text header objects
	lt	< (LT) Less than relation for handles.
	ne	~= (NE) Not equal relation for handles.
	new	Returns new text header (char), binary header (struct), trace header (struct) and trace data (double)
	notify	Notify listeners of event.
	read	Reads a SEG-Y file
	write	Writes a new SEG-Y file

Event Summary

ByteOrderChanged	Notifies listeners that the ByteOrder has changed
FileNameChanged	Notifies listeners that the FileName has changed
FormatCodeChanged	Notifies listeners that the FormatCode has changed
GUIchanged	Notifies listeners that the GUI has changed
ObjectBeingDestroyed	Notifies listeners that a particular object has been destroyed.
PermissionChanged	Notifies listeners that the Permission has changed
SampleIntervalChanged	Notifies listeners that the SamplesInterval has changed
SamplesPerTraceChanged	Notifies listeners that the SamplesPerTrace has changed
SegyRevisionChanged	Notifies listeners that the SegyRevision has changed
TextFormatChanged	Notifies listeners that the TextFormat has changed