Package ‘gemlog’

June 15, 2020

Type Package

Title File Conversion for 'Gem Infrasound Logger'

Date 2020-06-15

Version 0.41

Author Jake Anderson

Maintainer Jake Anderson <ajakef@gmail.com>

Description Reads data files from the 'Gem infrasound logger' for analysis and converts to segy format (which is convenient for reading with traditional seismic analysis software). The Gem infrasound logger is a low-cost, lightweight, low-power instrument for recording infrasound in field campaigns; email the maintainer for more information.

Depends signal

License GPL

LazyLoad yes

Repository CRAN

NeedsCompilation no

Date/Publication 2020-06-15 20:40:02 UTC

R topics documented:

gemlog-package *File Conversion for 'Gem Infrasound Logger'*

Description

Reads data files from the 'Gem infrasound logger' for analysis and converts to segy format (which is convenient for reading with traditional seismic analysis software). The Gem infrasound logger is a low-cost, lightweight, low-power instrument for recording infrasound in field campaigns; email the maintainer for more information.

Details

The DESCRIPTION file:

Package:	gemlog
Type:	Package
Title:	File Conversion for 'Gem Infrasound Logger'
Date:	2020-06-15
Version:	0.41
Author:	Jake Anderson
Maintainer:	Jake Anderson <ajakef@gmail.com>
Description:	Reads data files from the 'Gem infrasound logger' for analysis and converts to segy format (which is convenier
Depends:	signal
License:	GPL
LazyLoad:	yes
Repository:	CRAN

Index of help topics:

~~ An overview of how to use the package, including the most important ~~ ~~ functions ~~

Author(s)

Jake Anderson

Maintainer: Jake Anderson <ajakef@gmail.com>

References

Anderson, JF, JB Johnson, DC Bowman, and TJ Ronan (2018). The Gem Infrasound Logger and Custom-Built Instrumentation. Seismological Research Letters 89 (1), 153-164. https://doi.org/10.1785/0220170067

Examples

```
## Not run:
# define bitweight for 0.5 inch sensor with Rg = 2.2k
sensitivity = 22.014e-6 # 22.014 uV/Pa
Rg = 2.2 # gain-setting resistor value in kilo-ohms
gain = 1 + 49.4/2.2 # amplifier gain
A2D = 0.256/2^15 # volts per count in analog-digital converter
bitweight = A2D / (gain * sensitivity) # conversion from counts to Pa (Pa/count)

# convert files from two Gems (SNs 000 and 001)
Convert('raw/000', bitweight = bitweight)
Convert('raw/001', bitweight = bitweight)

## End(Not run)

## Not run:
ReadGem(0:1, 'raw/000') # read files raw/000/FILE0000.TXT and raw/000/FILE0001.TXT

## End(Not run)
```

`Convert` *Convert raw Gem data to segy*

Description

Convert takes a directory of raw Gem data files and converts them to PASSCAL segy files, including interpolating time with GPS strings and converting from counts to pressure units.

Usage

```
Convert(rawpath = ".", convertedpath = "converted", metadatapath = "metadata",
metadatafile = NA, gpspath = "gps", gpsfile = NA, t1 = -Inf, t2 = Inf, nums = NaN,
SN = character(), bitweight = NaN, units = 'Pa', time_adjustment = 0, blockdays = 1)
```

Arguments

`rawpath`	Directory containing raw data to be converted.
`convertedpath`	Directory (to be created, if necessary) where output segy files will be saved.
`metadatapath`	Directory (to be created, if necessary) where output metadata file will be saved.
`metadatafile`	Filename for output metadata. If set, overrides metadatapath. If unset, Convert creates the next logical filename in metadatapath.
`gpspath`	Directory (to be created, if necessary) where output gps file will be saved.
`gpsfile`	Filename for output gps data. If set, overrides gpspath. If unset, Convert creates the next logical filename in gpspath.

`t1`	Time at which conversion should start (class POSIXct).
`t2`	Time at which conversion should end (class POSIXct).
`nums`	File numbers to convert.
`SN`	Serial number of Gem data files to convert (to be safe, when data from multiple Gems could be mixed).
`bitweight`	Conversion factor between counts and output units (e.g., Pa per count). Leave as NaN to use default value (for logger version, configuration, and 'units').
`units`	Units for output: can be 'Pa', 'V', or 'counts'. Only used when bitweight is left unset and default bitweight is calculated.
`time_adjustment`	Offset to add to output times (usually unnecessary, sometimes +/- 1 s).
`blockdays`	Amount of data (measured in days) to process at a time. This can be decreased to 0.5 or 0.25 in case of memory problems.

Details

This is the usual function to use when converting data to segy files. To read data directly into R, use ReadGem.

Value

None; writes files only.

Note

A good directory structure might be something like

projectname

—-raw

———010: Directory containing data from Gem SN 010 (e.g.)

———011: Directory containing data from Gem SN 011 (e.g.)

—-converted

———segy files

—-gps

———010gps_000.txt: GPS file for Gem 010

———011gps_000.txt: GPS file for Gem 011

—-metadata

———010metadata_000.txt: Metadata file for Gem 010

———011metadata_000.txt: Metadata file for Gem 011

—-projectname_notes.txt

Author(s)

Jake Anderson

See Also

ReadGem

Examples

```
## Not run:
# define bitweight for 0.5 inch sensor with Rg = 2.2k
sensitivity = 22.014e-6 # 22.014 uV/Pa
Rg = 2.2 # gain-setting resistor value in kilo-ohms
gain = 1 + 49.4/2.2 # amplifier gain
A2D = 0.256/2^15 # volts per count in analog-digital converter
bitweight = A2D / (gain * sensitivity) # conversion from counts to Pa (Pa/count)

# convert files from two Gems (SNs 000 and 001)
Convert('raw/000', bitweight = bitweight)
Convert('raw/001', bitweight = bitweight)

## End(Not run)
```

`GemSpecs` *Gem specs table*

Description

Specs table for various Gem versions.

Usage

```
data(GemSpecs)
```

Format

Data frame with following elements:

version	Version number corresponding to following specs
bitweight_Pa	Pressure resolution for default (high) gain (Pa/count)
bitweight_V	Voltage resolution for default (high) gain (V/count)
min_SN	Lowest serial number made for this version
max_SN	Highest serial number made for this version

Examples

```
data(GemSpecs)

## determine pressure bitweight for logger 014
i = which(14 >= GemSpecs$min_SN & 14 <= GemSpecs$max_SN)
GemSpecs[i,]$bitweight_Pa
```

`gem_metadata` *Example Metadata from Gem Logger*

Description

Metadata (temperature, battery, etc.) from a Gem data logger. Provided as an example for running PlotMetadata.

Usage

```
data(gem_metadata)
```

Format

List containing vectors (time series) of different metadata types.

Source

Recording made by author.

See Also

PlotMetadata

`PlotMetadata` *Read and plot Gem metadata*

Description

ScanMetadata reads a Gem metadata file produced by Convert. PlotMetadata plots battery, temperature, GPS metadata.

Usage

```
ScanMetadata(fn, plot = TRUE)
PlotMetadata(M, xlim = range(M$t, na.rm = TRUE))
```

Arguments

`fn`	Filename to read
`plot`	If true, plots metadata after reading the file
`M`	Metadata, such as output of ScanMetadata
`xlim`	Time limits to plot, in fractional days of year

Value

ScanMetadata: list including metadata from file:

`millis`	millis count of metadata sample
`batt`	battery voltage
`temp`	temperature in (deg C)
`maxWriteTime`	maximum time required to write a sample
`minFifoFree`	minimum number of free sampes in FIFO buffer
`maxFifoUsed`	maximum number of used samples in FIFO buffer
`maxOverruns`	maximum number of sample overruns
`gpsOnFlag`	1 if gps is turned on, 0 otherwise
`unusedStack1`	free memory in stack 1
`unusedStackIdle`	free memory in idle stack

PlotMetadata: None.

Author(s)

Jake Anderson

Examples

```
## Not run:
M = ScanMetadata('metadata/001metadata_000.txt') # scan the first metadata file from Gem SN 001

## End(Not run)

data(gem_metadata)
PlotMetadata(gem_metadata)
```

`ReadGem` *Read raw Gem data*

Description

Reads raw Gem data into R. To write segy files, use Convert.

Usage

```
ReadGem(nums = 0:9999, path = './', SN = character(), units = 'Pa',
bitweight = NaN, bitweight_V = NaN, bitweight_Pa = NaN, alloutput =
FALSE, verbose = TRUE, requireGPS = FALSE)
```

Arguments

`nums`	File numbers to convert.
`path`	Directory in which raw data files are contained.
`SN`	If set, only read files of this serial number.
`units`	For unit conversions: should the output be in V, Pa, or counts?
`bitweight`	If set, override the default bitweight for this logger in the current configuration.
`bitweight_V`	If set, override the default V bitweight for this logger in the current configuration.
`bitweight_Pa`	If set, override the default Pa bitweight for this logger in the current configuration.
`alloutput`	Include raw data in the output, in addition to the processed data.
`verbose`	Provide verbose output.
`requireGPS`	Require GPS strings to perform the conversion.

Value

`t`	sample times (POSIXct)
`p`	samples (in whatever units were set by user)

- gps$yryear of gps samples
- gps$dategps sample time, as fractional day of year
- gps$latlatitude
- gps$lonlongitude

- metadata$millismillis count of metadata sample
- metadata$battbattery voltage
- metadata$temptemperature in (deg C)
- metadata$maxWriteTimemaximum time required to write a sample
- metadata$minFifoFreeminimum number of free sampes in FIFO buffer
- metadata$maxFifoUsedmaximum number of used samples in FIFO buffer
- metadata$maxOverrunsmaximum number of sample overruns
- metadata$gpsOnFlag1 if gps is turned on, 0 otherwise
- metadata$unusedStack1free memory in stack 1
- metadata$unusedStackIdlefree memory in idle stack

- header$filevector of raw file names
- header$SNvector of Gem serial numbers
- header$latmean latitude
- header$lonmean longitude
- header$t1start time
- header$t2end time
- header$alloutputif alloutput == TRUE, list including raw data
- configconfiguration settings set using Gem configuration file

Author(s)

Jake Anderson

See Also

Convert

Examples

```
## Not run:
ReadGem(nums = 0:1, path = 'raw/', SN = '000') # read files raw/FILE0000.000 and raw/FILE0001.000

## End(Not run)
```

Index