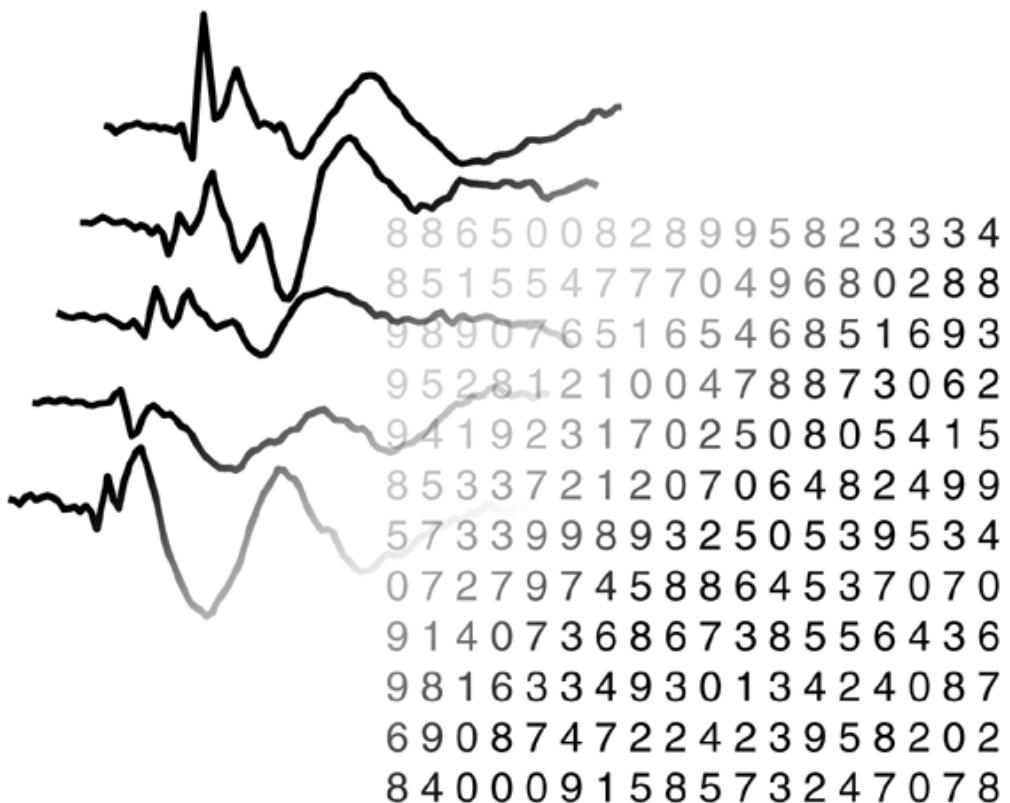


Fiff Access

Version 1.2
by Kimmo Uutela



Contents

1	Release notes	4
1.1	Copyright and warranty	4
1.2	Changes in FiffAccess 1.2	4
2	Installation	5
2.1	Installing files on HPUX 10.20	5
2.2	Installing files on Linux RedHat 6.2 or compatible	5
2.3	Setting Matlab path	5
2.4	Installing files to different directories in HPUX	5
2.5	Installing files to different directories in Linux	6
3	Using the toolbox	7
3.1	Transformation matrices	7
3.2	Reading MEG data	8
3.3	Writing MEG data	9
4	Function reference	11
4.1	Function badchans	11
4.2	Function chaninfo	12
4.3	Function channames	13
4.4	Function loadb dip	14
4.5	Function loadfif	15
4.6	Function loadmri	16
4.7	Function loadtrans	17
4.8	Function loadtri	18
4.9	Function megmodel	19
4.10	Function projmat	20
4.11	Function rawchannels	21
4.12	Function rawdata	23
4.13	Function savefif	25
4.14	Function savemri	26
4.15	Function subjno	27
4.16	Function usemodel	28
4.17	Function writeraw	29

1 Release notes

1.1 Copyright and warranty

© 2001 Kimmo Uutela. All rights reserved.

The author makes no warranty of any kind with regard to this material or the software.

This program was developed at the Brain Research Unit of the Low Temperature Laboratory in Helsinki University of Technology. The public distribution is sponsored by Neuromag Ltd. Matlab is a registered trademark by The Mathworks, Inc.

1.2 Changes in FiffAccess 1.2

Release 4: November 12, 2001

- Added manual
- Fixed error on rawdata without MEG channels

Release 3: February 19, 2001

- Added support for Matlab 6 on Linux

Release 2: January 18, 2001

- Added "device"-option to megmodel

Release 1: November 30, 2000

- Initial release

2 Installation

2.1 Installing files on HPUX 10.20

Log in the workstation as the root user. Unpack file `meg-pd-1.2-x.hpxtar.gz` into the directory `/opt/neuromag` by issuing the following commands.

```
# cd /
# gunzip DIRECTORY/meg-pd-1.2-1.hpxtar.gz
# tar -xf DIRECTORY/meg-pd-1.2-1.hpxtar
```

2.2 Installing files on Linux RedHat 6.2 or compatible

Log in the workstation as the root user. Install the package issuing the following command:

```
# rpm -i meg-pd-1.2-1.i386.rpm
```

2.3 Setting Matlab path

Add the following line to the definition of the path in the file

```
$MATLAB/toolbox/local/pathdef.m:
' /opt/neuromag/meg-pd-1.2/: ', ...
```

Alternatively, you can issue the Matlab command

```
addpath /opt/neuromag/meg-pd-1.2
```

in the beginning of a Matlab session.

As is the case with [reading MEG files](#), writing [evoked](#) data files and [continuous](#) data files has to be differently. However, the coil information for both methods has to be set up first using the [MEGMODEL](#) command and a model file.

2.4 Installing files to different directories in HPUX

If you can not use the supported installation methods, the toolbox may also work when installed in a different location. Unpack the file `meg-pd-1.2-x.hpxtar.gz` into some directory. Before starting Matlab, set the environmental variable `SHLIB_PATH` correctly. For example, using `/bin/sh` as a shell, you should give the commands

```
SHLIB_PATH=DIRECTORY/opt/neuromag/lib/hppa1.1-hp-hpx10.20;
export SHLIB_PATH.
```

2.5 Installing files to different directories in Linux

If you can not use the supported installation methods, the toolbox may also work when installed in a different location. Unpack the file `meg-pd-1.2-x.i386.tar.gz` into some directory. Before starting Matlab, set the environmental variable `LD_LIBRARY_PATH` correctly. For example using `/bin/bash` as a shell, you should give the command `export LD_LIBRARY_PATH=DIRECTORY/i386` before starting Matlab.

3 Using the toolbox

3.1 Transformation matrices

Neuromag software uses different kinds of coordinate systems. The transformations between different coordinate systems can be described using an 4x4 transformation matrix:

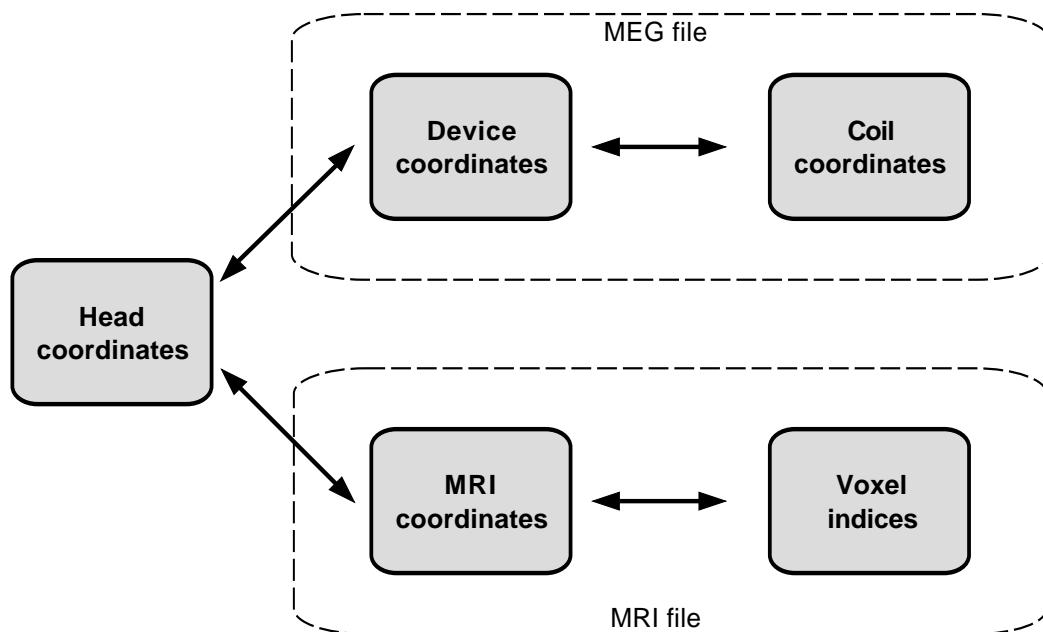
ex1	ey1	ez1	tr1
ex2	ey2	ez2	tr2
ex3	ey3	ez3	tr3
0	0	0	1

To transform a coordinate from one system to each other, load the transformation matrix T , augment the original coordinate with one $[r_orig; 1]$, and calculate the transformation as

```
r_new = T * [r_orig; 1].
```

The 3×3 matrix $[ex\dots ez]$ is an orthonormal rotation, and tr is the transformation vector in meters. To rotate an orientation, augment the original orientation with zero:
 $q_{new} = T * [q_{orig}; 0]$.

Coordinate system used with Neuromag software



LOADTRANS Explanation**argument**

HEAD	Head coordinate system
DEVICE	The coordinate system of the magnetometer.
	Close to head coordinate system but origin in the center of the sensor array, usually lower than the head coordinate system origin.
MRI	Coordinate system of the MRI device. Usually DICOM coordinate system.
COIL	Coordinate system for a single sensor coil. The center is at the center of the sensor, z axis is the orthogonal orientation, gradiometers measure the gradient along x axis.
TORSO	Coordinate system for magnetocardiography

See also

[LOADTRANS](#), [CHANINFO](#), [LOADMRI](#)

3.2 Reading MEG data

Neuromag fif-files can have MEG data in two different formats. [Evoked](#) responses are saved in a file where all the samples of certain channel are saved at one time. On the other hand, the [continuous, unaveraged data](#) is usually so large that it can not be read at one time to the memory. Therefore, it is saved as buffers having data from all channels but only short data range.

Reading evoked responses

The evoked response can be read using function [LOADFIF](#). The evoked file may contain several sets, for example, different averaged categories. The first data set of file "filename.fif" can be read with the command

[B,SF,T0]=loadfif('filename.fif'); . A specific set can be given as an argument to LOADFIF .

A vector of time points can be generated as

T=T0+(0:SIZE(B,1)-1)/SF;

and the results can be plotted as

plot(T,B'); If the file includes also non-MEG channels, all the channels can be read as

[B,SF,T0]=loadfif('filename.fif','any');

The channel names can be obtained with the command [CHANNAMES](#).

Reading continuous data

To process raw data, you should read it one buffer at a time using the function [RAWDATA](#). Continuous data file "fileraw.fif" can be processed using a script like

```
status='ok';
rawdata('any','fileraw.fif'); while strcmp(status,'ok'),
    B=rawdata('next');
    process_data(B);
end;
```

```
rawdata('close');
```

Only one raw data file can be open at one time.

See also

[LOADFIF](#), [RAWDATA](#), [RAWCHANNELS](#)

3.3 Writing MEG data

As is the case with [reading MEG files](#), writing [evoked](#) data files and [continuous](#) data files has to be differently. However, the coil information for both methods has to be set up first using the [MEGMODEL](#) command and a model file.

Writing evoked responses

To specify the coil information and create an evoked data fif-file, use commands like

```
B=data;SF=600;T0=-0.100;
megmodel('head','example.fif');
savefif('newfile.fif',B,SF,T0,'Test file');
```

Writing continuous data

To process raw data, you should write one buffer at a time using the function [WRITERAW](#).

Continuous data file "rawfile.fif" can be processed using a script like

```
megmodel('head','example.fif');
SF=600;T0=-0.100;
writeraw('rawfile.fif',SF,T0,'Test file');
for n=1:10;
    data=...;
    writeraw(data);
end;
```

```
writeraw('close');
```

Normally writeraw creates the file in floating-point format. In this way, the file can have an arbitrary scale. To save disk space the 16-bit format can be used by defining the range and calibration with WRITERAW. For example, when processing existing raw data files, the scaling can be read with the command [RAWDATA](#).

See also

[MEGMODEL](#), [SAVEFIF](#), [WRITERAW](#)

4 Function reference

4.1 Function badchans

Get or set bad channels

Function BADCHANS is used to read the bad channels from a fif-file or mark the bad channels for saved fif-files.

Synopsis

```
[V, NAMES] = BADCHANS
    Returns the channels currently set bad
    BADCHANS(V)
    Sets the bad channels
    BADCHANS(NAMES)
    Set the bad channels
    BADCHANS('reset')
    Reload bad channels from the model file
```

Arguments

V	nx1 double	Selection of bad channels in the channel order. 1 = bad channel, 0 = good channel. Length should match the number of channels in model file
NAMES	cell array	A string array of strings like {'MEG 0111', 'MEG 0112'}. Adds the named channels to the list of bad channels

Returns

V	n x 1 double	Selection in the channel order.
NAMES	n x 1 double	A string array of names of the bad channels

See also

[MEGMODEL](#)

4.2 Function chaninfo

Get information about channels in used model

CHANINFO returns information of the currently selected MEGMODEL.

Synopsis

```
[ NUMBER , NAME , T ]= CHANINFO
    Return channel numbers, names & places
    [ GRAD , TYPE ]= CHANINFO ( 'type' )
    Return channel types
        NOISE = CHANINFO ( 'noise' )
    Return channel noise levels
        CHANINFO ( CHANNELS , ... )
    Return info of specific channels
```

Arguments

CHANNAMES n x 1 double *Specifies the index of channels for the query. Starts from 1.*

Returns

NUMBER	n x 1 double	<i>Channel ID numbers</i>
NAME	n x m char	<i>Channel name matrix.</i>
T	n x 1 cell array	<i>The transformation between 'COIL' and either 'HEAD' or 'DEVICE' coordinates. See section Coordinates.</i>
GRAD	n x 1 double	<i>0 for magnetometers, 1 for planar gradiometers.</i>
TYPE	n x 1 cell array	<i>Names of the coil types</i>
NOISE	n x 1 double	<i>Weighting factor for the coil type</i>

See also

[CHANNAMES](#), [MEGMODEL](#)

4.3 Function channames

Get information about all channels in fif-file

Returns information from a fif-file (not using [MEGMODEL](#)).

Synopsis

```
[ NAME, TYPE, NUMBER ]= CHANNAMES ( FILENAME )
```

Arguments

FILENAME string *The fif-file name*

Returns

NAME n x 1 cell array *The channel names*

TYPE n x 1 double *Numbers of coil types*

NUMBER n x 1 double *Channel logical numbers*

See also

[CHANINFO](#)

4.4 Function loadbdip

Load binary dipole file

Reads the locations and orientations of dipoles from a binary file.

Synopsis

```
[R, Q] = LOADBDIP(FILENAME)
```

Arguments

FILENAME string *input filename file.bdip*

Returns

R 3 x n double *The dipole locations [X ; Y ; Z]*

Units: m

Q 3 x n double *The dipole amplitude [QX ; QY ; QZ].*

Units: Am

See also

[LOADFIF](#)

4.5 Function loadfif

Load an evoked response

Reads an evoked response matrix B, sampling frequency and starting time from a fif-file.

Synopsis

```
[B, SF, T0] = LOADFIF(FILENAME)
Load MEG channels from the first set
[B, SF, T0] = LOADFIF(FILENAME, 'any')
Load also non-MEG channels
[B, SF, T0] = LOADFIF(FILENAME, SET)
Load the specified set
[NSETS, COMMENTS] = LOADFIF(FILENAME, 'sets')
Load the comments of filesets
```

Arguments

FILENAME	string	<i>The fif-file name</i>
SET		<i>The number of filesets to load. First fileset has index 0</i>

Returns

B	n x m double	<i>The data of one channel in each row, a single time sample in each column.</i> <i>Units: SI units. fT/m for gradiometers, fT for magnetometers, V for electric channels.</i>
SF	double	<i>Sampling frequency</i> <i>Units: Hz</i>
T0	double	<i>Start time of the epoch</i> <i>Units: s</i>
NSETS	double	<i>The number of filesets in the file</i>
COMMENTS	cell array	<i>Comments of each file set</i>

See also

[CHANINFO](#), [CHANNAMES](#), [RAWDATA](#), [SAVEFIF](#), [Reading evoked responses](#)

4.6 Function loadmri

Load MR images

Read an MR image from fif-file. For use of coordinate transformations, see section [Coordinates](#).

Synopsis

```
[DATA, T] = LOADMRI(FILENAME)
```

Arguments

FILENAME string *The name of the MRI fif-file*

Returns

DATA 3D uint8 *The gray levels of the MR image*

T 4 x 4 double *The transformation matrix from voxel indices to MRI coordinates. See section [Coordinates](#) for using the coordinate systems.*

Units: m

4.7 Function loadtrans

Load coordinate transformation

Reads from a fif-file a coordinate transformation from coordinate system COORDFROM to coordinate transform COORDTO. See section [Coordinates](#) for using the transformations.

Synopsis

`T = LOADTRANS (FILENAME, CFROM, CTO)`

Load the transformation matrix from coordinate type CFROM to coordinate type CTO

`T = LOADTRANS (FILENAME)`

Load the transformation matrix form HEAD to DEVICE coordinates

`T = LOADTRANS`

The transformation of current meg model

Arguments

FILENAME string *The fif-file name*

CFROM string *The original coordinate system*

CTO string *The destination, transformed coordinate system*

Returns

T 4 x 4 double *The transformation matrix*

See also

[Coordinates](#), [MEGMODEL](#)

4.8 Function loadtri

Load triangulated surface from fif-file

Loads the first triangel net from a BEM fif-file. The results can be used, for example, with the trimesh-function using the TRIMESH commans:

```
TRIMESH( TRIANGLES , NODES( :, 1 ) , NODES( :, 2 ) , NODES( :, 3 ) );
```

Synopsis

```
[NODES, TRIANGLES, NORMALS] = LOADTRI(FILENAME)
```

Arguments

FILENAME string *The fif-file name*

Returns

NODES nn x 3 double *Coordinates of the nodes (vertices) of the net. Number of rows (nn) is the number of nodes.*

Units: m

TRIANGLES nt x 3 double *The indices of vertices belonging to each triangle, between 1 and nn. . Number of rows is the number of triangles (nt). NORMALS nnx3 double Surface normal vectors for each node. Points outwards.*

4.9 Function megmodel

Set coil information for saving

Function MEGMODEL sets the coil positions and conductivity model that will be later used for creating fif-files with [savefif](#).

Synopsis

```
MEGMODEL ('head', COILFILE)
  Use head coordinate transformation in COILFILE
MEGMODEL ('device', COILFILE)
  Discard head coordinate transformation
MEGMODEL (TRANSFORM, COILFILE)
  Use specified head coordinate transformation
```

Arguments

COILFILE	string	<i>The name of the fif-file with coil information</i>
TRANSFORM	cell array	<i>{'trans',T} to specify head coordinate transformation or {'trans',TRANSFORMFILE} to load head coordinate from a different fif-file.</i>

See also

[USEMODEL](#), [SAVEFIF](#), [WRITERAW](#), [LOADTRANS](#), [Coordinates](#)

4.10 Function projmat

Load SSP projection

Returns the transformation matrix representing the SSP. Use as

`B_ssp = T * B`

Synopsis

`T = PROJMAT (FILENAME)`

`T = PROJMAT (FILENAME, BAD_CHANNELS)`

`T = PROJMAT (PROJFILE, DATAFILE, BAD_CHANNELS)`

`T = PROJMAT (PROJFILE, DATAFILE, BAD_CHANNELS, PROJCHAN)`

Arguments

<code>FILENAME</code>	string	<i>The name of fif-file with projection and coil information</i>
<code>PROJFILE</code>	string	<i>The name of fif-file with projection</i>
<code>DATAFILE</code>	string	<i>The name of fif-file with coil information</i>
<code>BAD_CHANNELS</code>	cell array	<i>The names of the bad channels.</i>
<code>PROJCHAN</code>	cell array	<i>The names of the channels in the result</i>

Returns

`T` n x n double *The transformation matrix*

See also

[CHANNAMES](#), [BADCHANS](#)

4.11 Function rawchannels

Load some channels from raw data file

An utility function for reading one or a few channels from a raw data fif-file. If the file contains skips, each is represented with 100 samples of zeros (compatible with Graph).

Synopsis

```
[B, SF] = RAWCHANNELS(FILENAME, CHANNELNAMES)
Reads named channel from file FILENAME
[B, SF] = RAWCHANNELS(FILENAME, CHANNELNUMS)
Reads channels by numbers
[B, SF, T0] = RAWCHANNELS(FILENAME, CHANNELNAMES, RANGE)
Returns data within specified time range
[B, SF, T0] = RAWCHANNELS(FILENAME, CHANNELNAMES, 'noskips')
Returns data without skips. Timing will not match the times used in Graph.
```

Arguments

FILENAME	string	<i>The fif-file name</i>
CHANNELNAMES	cell array	<i>Names of the channel names to be read.</i>
		<i>Example:</i> { 'MEG 0111', 'MEG 0112', 'STI 001'}
CHANNELNUMS	1 x n double	<i>Order numbers of the channels, starting from 1.</i>
		<i>Example:</i> [1 2 3 10]
RANGE	1 x 2 double	<i>The time range to be read.</i> <i>Format [START END].</i> <i>Units:</i> s

Returns

B	nc x nt double	<i>The data.</i>
		<i>Units:</i> SI units. fT/m for gradiometers, fT for magnetometers, V for electric channels.
SF	double	<i>Sampling frequency</i>
		<i>Units:</i> Hz
T0	double	<i>Start time of the epoch</i>
		<i>Units:</i> s

See also

[RAWDATA](#), [LOADFIF](#)

4.12 Function rawdata

Read continuous data fif-files

Handles reading raw data from fif-files. See section [Reading continuous data](#).

Synopsis

RAWDATA (FILENAME)

Opens the fif raw or evoked data files for reading MEG channels.

RAWDATA ('any', FILENAME)

Opens the fif raw or evoked data files for reading MEG channels.

[B, STATUS] = **RAWDATA** ('next')

Gets the next data buffer

T = **RAWDATA** ('goto', T)

Goes to buffer including defined time point

RAWDATA ('close')

Closes the raw data file

SF = **RAWDATA** ('sf')

Returns the sampling frequency

T = **RAWDATA** ('t')

Returns the current time

SAMPLE = **RAWDATA** ('samples')

Gets number of samples read this far

[RANGE, CAL] = **RAWDATA** ('range')

Gets the range and calibration

Arguments

FILENAME string *The fif-file name*

T double *The time to go to. To go to beginning, use -inf
Units: s*

Returns

B	nc x nt double	<i>The data.</i> Units: SI units. fT/m for gradiometers, fT for magnetometers, V for electric channels.
STATUS	string	<i>Status of the reading. Either 'ok' (buffer read normally), 'skip' (a skip in the data), 'eof' (no more buffers), or 'error' (file error reading data)</i>
SF	double	<i>Sampling frequency</i> Units: Hz
T	double	<i>Start time of the epoch (start time of the next buffer to be read)</i> Units: s
SAMPLE	string	<i>Number of samples since the beginning of the file</i>

See also

[CHANNAMES](#), [LOADFIF](#), [RAWCHANNELS](#), [Reading continuous data](#), [WRITERAW](#)

4.13 Function savefif

Save simulated response

Saves an evoked response matrix to a fif-file. The channel information is copied from current MEGMODEL.

Synopsis

SAVEFIF (FILENAME, B, SF, T0, COMMENT)

Create file FILENAME with specified parameters

SAVEFIF (FILENAME, B)

Create file with SF=100 Hz and T0=0.

SAVEFIF (FILENAME, B, CHANNELS, SF, T0, COMMENT)

Save specified channels

Arguments

FILENAME	string	<i>The fif-file name</i>
B	n x m double	<i>The data of one channel in each row, a single time sample in each column.</i> Units: SI units. fT/m for gradiometers, fT for magnetometers, V for electric channels.
SF	double	<i>Sampling frequency</i> Units: Hz
T0	double	<i>Start time of the epoch</i> Units: s
COMMENT	string	<i>A comment to the created fif-file</i>
CHANNELS	n x 1 cell array	<i>The channel names to be saved</i>

See also

[LOADFIF](#), [MEGMODEL](#), [WRITERAW](#), Writing evoked responses

4.14 Function savemri

Save MR images to fif-files

Creates an MRI fif-file with the given data

Synopsis

SAVEMRI (FILENAME, DATA, VOXELSIZE, THEAD, ID)

Saves the mri data to FILENAME

SAVEMRI (FILENAME, DATA, VOXELSIZE)

Assumes MRI in head coordinates and sets ID to 0

Arguments

FILENAME string *The name of the MRI fif-file*

DATA 3D UINT8 *The MR images to be saved. Size width x height x depth. VOXELSIZE 3 x 1 double Size of the voxel; [width; height; depth] Units: m THEAD 4 x 4 double coordinate transformation from 'MRI' to 'HEAD' coordinates ID double subject ID*

See also

[LOADMRI](#), [SUBJNO](#), [LOADTRANS](#), [Coordinates](#)

4.15 Function subjno

Read subject number from fif-file

Synopsis

```
ID = SUBJNO (FILENAME)
```

Arguments

FILENAME *The fif-file name*

Returns

ID double *The subject ID*

4.16 Function usemodel

Switch between models

By calling USEMODEL before MEGMODEL several MEG models can be used without MEGMODEL commands.

Example:

```
oldmodel=USEMODEL;  
USEMODEL('new');  
MEGMODEL(...);  
...  
USEMODEL('del');  
USEMODEL(oldmodel);
```

Synopsis

N = USEMODEL ('new')
Makes room for a new MEG model

USEMODEL (N)
Selects the given model

USEMODEL ('del', N)
Frees an obsolete model N

N = USEMODEL
Return the number of current model

Arguments

N double *The model ID*

Returns

N double *The model ID*

See also

[MEGMODEL](#)

4.17 Function writeraw

Save processed data to continuous data file

Saves data buffers to a fif-file. The channel information is copied from current MEG model file.

Synopsis

```
WRITERAW (FILENAME, SF, T0, COMMENT)
  Open file in floating point file format
WRITERAW (FILENAME)
  Open file with SF=100 Hz and T0=0
WRITERAW (FILENAME, CHANNELS, . . .)
  Open file for saving specified channels
WRITERAW(FILENAME,RANGECAL,...);
  Opens file in 16 bit file
WRITERAW(B);
  Write a buffer to the file
WRITERAW('skip',SKIPLength);
  Write a skip buffer to the file
WRITERAW('close');
  Close the file
```

Arguments

FILENAME	string	<i>The fif-file name</i>
SF	double	<i>Sampling frequency</i> Units: Hz
T0	double	<i>Start time of the epoch</i> Units: s
COMMENT	string	<i>A comment to the created fif-file</i>
CHANNELS	n x 1 cell array	<i>The channel names to be saved</i>
RANGECAL	1 x 2 double	<i>The range and calibration; see RAWDATA</i>
B	n x m double	<i>The data of one channel in each row, a single time sample in each column.</i> Units: SI units. fT/m for gradiometers, fT for magnetometers, V for electric channels.
SKIPLength		<i>The length of the skip in samples.</i>

See also

[SAVEFIF](#), [RAWDATA](#), [MEGMODEL](#), Writing continuous data