processEEG

Generated by Doxygen 1.8.10

Sat Dec 31 2016 10:58:52

Contents

Chapter 1

Research

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 4

Class Documentation

4.1 COLOUR Struct Reference

Public Attributes

- double **r**
- double **g**
- double **b**

The documentation for this struct was generated from the following file:

- /Users/vinay/Google Drive/Science/Research/Morlets/Plot.cc

4.2 edf_annotation_struct Struct Reference

Public Attributes

- long long **onset**
- char **duration** [16]
- char **annotation** [EDFLIB_MAX_ANNOTATION_LEN+1]

The documentation for this struct was generated from the following file:

- /Users/vinay/Google Drive/Science/Research/include/edflib.h

4.3 edf_hdr_struct Struct Reference

Public Attributes

- int **handle**
- int **filetype**
- int **edfsignals**
- long long **file_duration**
- int **startdate_day**
- int **startdate_month**
- int **startdate_year**
- long long **starttime_subsecond**

- int **starttime_second**
- int **starttime_minute**
- int **starttime_hour**
- char **patient** [81]
- char **recording** [81]
- char **patientcode** [81]
- char **gender** [16]
- char **birthdate** [16]
- char **patient_name** [81]
- char **patient_additional** [81]
- char **admincode** [81]
- char **technician** [81]
- char **equipment** [81]
- char **recording_additional** [81]
- long long **datarecord_duration**
- long long **datarecords_in_file**
- long long **annotations_in_file**
- struct edf_param_struct **signalparam** [EDFLIB_MAXSIGNALS]

The documentation for this struct was generated from the following file:

- /Users/vinay/Google Drive/Science/Research/include/edflib.h

4.4 edf_param_struct Struct Reference

Public Attributes

- char **label** [17]
- long long **smp_in_file**
- double **phys_max**
- double **phys_min**
- int **dig_max**
- int **dig_min**
- int **smp_in_datarecord**
- char **physdimension** [9]
- char **prefilter** [81]
- char **transducer** [81]

The documentation for this struct was generated from the following file:

- /Users/vinay/Google Drive/Science/Research/include/edflib.h

4.5 edfhdrblock Struct Reference

Public Attributes

- FILE * **file_hdl**
- char **path** [1024]
- int **writemode**
- char **version** [32]
- char **patient** [81]
- char **recording** [81]

- char **plus_patientcode** [81]
- char **plus_gender** [16]
- char **plus_birthdate** [16]
- char **plus_patient_name** [81]
- char **plus_patient_additional** [81]
- char **plus_startdate** [16]
- char **plus_admincode** [81]
- char **plus_technician** [81]
- char **plus_equipment** [81]
- char **plus_recording_additional** [81]
- long long **l_starttime**
- int **startdate_day**
- int **startdate_month**
- int **startdate_year**
- int **starttime_second**
- int **starttime_minute**
- int **starttime_hour**
- char **reserved** [45]
- int **hdrsize**
- int **edfsignals**
- long long **datarecords**
- int **recordsize**
- int **annot_ch** [EDFLIB_MAXSIGNALS]
- int **nr_annot_chns**
- int **mapped_signals** [EDFLIB_MAXSIGNALS]
- int **edf**
- int **edfplus**
- int **bdf**
- int **bdfplus**
- int **discontinuous**
- int **signal_write_sequence_pos**
- long long **starttime_offset**
- double **data_record_duration**
- long long **long_data_record_duration**
- int **annots_in_file**
- int **annotlist_sz**
- int **total_annot_bytes**
- int **eq_sf**
- struct edfparamblock * **edfparam**

The documentation for this struct was generated from the following file:

- /Users/vinay/Google Drive/Science/Research/src/edflib.c

4.6 edfparamblock Struct Reference

Public Attributes

- char **label** [17]
- char **transducer** [81]
- char **physdimension** [9]
- double **phys_min**
- double **phys_max**

- int **dig_min**
- int **dig_max**
- char **prefilter** [81]
- int **smp_per_record**
- char **reserved** [33]
- double **offset**
- int **buf_offset**
- double **bitvalue**
- int **annotation**
- long long **sample_pntr**

The documentation for this struct was generated from the following file:

- /Users/vinay/Google Drive/Science/Research/src/edflib.c

4.7 pngwriter Class Reference

Public Member Functions

- **pngwriter** (const pngwriter &rhs)
- **pngwriter** (int width, int height, int backgroundcolour, char ∗filename)
- **pngwriter** (int width, int height, double backgroundcolour, char ∗filename)
- **pngwriter** (int width, int height, int backgroundcolour, const char ∗filename)
- **pngwriter** (int width, int height, double backgroundcolour, const char ∗filename)
- pngwriter & **operator=** (const pngwriter &rhs)
- void **plot** (int x, int y, int red, int green, int blue)
- void **plot** (int x, int y, double red, double green, double blue)
- void **plotHSV** (int x, int y, double hue, double saturation, double value)
- void **plotHSV** (int x, int y, int hue, int saturation, int value)
- int **read** (int x, int y, int colour)
- int **read** (int x, int y)
- double **dread** (int x, int y, int colour)
- double **dread** (int x, int y)
- int **readHSV** (int x, int y, int colour)
- double **dreadHSV** (int x, int y, int colour)
- void **clear** (void)
- void **close** (void)
- void **pngwriter_rename** (char ∗newname)
- void **pngwriter_rename** (const char ∗newname)
- void **pngwriter_rename** (long unsigned int index)
- void **line** (int xfrom, int yfrom, int xto, int yto, int red, int green, int blue)
- void **line** (int xfrom, int yfrom, int xto, int yto, double red, double green, double blue)
- void **triangle** (int x1, int y1, int x2, int y2, int x3, int y3, int red, int green, int blue)
- void **triangle** (int x1, int y1, int x2, int y2, int x3, int y3, double red, double green, double blue)
- void **square** (int xfrom, int yfrom, int xto, int yto, int red, int green, int blue)
- void **square** (int xfrom, int yfrom, int xto, int yto, double red, double green, double blue)
- void **filledsquare** (int xfrom, int yfrom, int xto, int yto, int red, int green, int blue)
- void **filledsquare** (int xfrom, int yfrom, int xto, int yto, double red, double green, double blue)
- void **circle** (int xcentre, int ycentre, int radius, int red, int green, int blue)
- void **circle** (int xcentre, int ycentre, int radius, double red, double green, double blue)
- void **filledcircle** (int xcentre, int ycentre, int radius, int red, int green, int blue)
- void **filledcircle** (int xcentre, int ycentre, int radius, double red, double green, double blue)
- void **readfromfile** (char ∗name)

- void **readfromfile** (const char ∗name)
- int **getheight** (void)
- int **getwidth** (void)
- void **setcompressionlevel** (int level)
- int **getbitdepth** (void)
- int **getcolortype** (void)
- void **setgamma** (double gamma)
- double **getgamma** (void)
- void **bezier** (int startPtX, int startPtY, int startControlX, int startControlY, int endPtX, int endPtY, int end↩ ControlX, int endControlY, double red, double green, double blue)
- void **bezier** (int startPtX, int startPtY, int startControlX, int startControlY, int endPtX, int endPtY, int end↩ ControlX, int endControlY, int red, int green, int blue)
- void **settext** (char ∗title, char ∗author, char ∗description, char ∗software)
- void **settext** (const char ∗title, const char ∗author, const char ∗description, const char ∗software)
- void **write_png** (void)
- void **plot_text** (char ∗face_path, int fontsize, int x_start, int y_start, double angle, char ∗text, double red, double green, double blue)
- void **plot_text** (char ∗face_path, int fontsize, int x_start, int y_start, double angle, char ∗text, int red, int green, int blue)
- void **plot_text_utf8** (char ∗face_path, int fontsize, int x_start, int y_start, double angle, char ∗text, double red, double green, double blue)
- void **plot_text_utf8** (char ∗face_path, int fontsize, int x_start, int y_start, double angle, char ∗text, int red, int green, int blue)
- int **bilinear_interpolation_read** (double x, double y, int colour)
- double **bilinear_interpolation_dread** (double x, double y, int colour)
- void **plot_blend** (int x, int y, double opacity, int red, int green, int blue)
- void **plot_blend** (int x, int y, double opacity, double red, double green, double blue)
- void **invert** (void)
- void **resize** (int width, int height)
- void **boundary_fill** (int xstart, int ystart, double boundary_red, double boundary_green, double boundary_↩ blue, double fill_red, double fill_green, double fill_blue)
- void **boundary_fill** (int xstart, int ystart, int boundary_red, int boundary_green, int boundary_blue, int fill_red, int fill_green, int fill_blue)
- void **flood_fill** (int xstart, int ystart, double fill_red, double fill_green, double fill_blue)
- void **flood_fill** (int xstart, int ystart, int fill_red, int fill_green, int fill_blue)
- void **polygon** (int ∗points, int number_of_points, double red, double green, double blue)
- void **polygon** (int ∗points, int number_of_points, int red, int green, int blue)
- void **plotCMYK** (int x, int y, double cyan, double magenta, double yellow, double black)
- void **plotCMYK** (int x, int y, int cyan, int magenta, int yellow, int black)
- double **dreadCMYK** (int x, int y, int colour)
- int **readCMYK** (int x, int y, int colour)
- void **scale_k** (double k)
- void **scale_kxky** (double kx, double ky)
- void **scale_wh** (int finalwidth, int finalheight)
- void **plotHSV_blend** (int x, int y, double opacity, double hue, double saturation, double value)
- void **plotHSV_blend** (int x, int y, double opacity, int hue, int saturation, int value)
- void **line_blend** (int xfrom, int yfrom, int xto, int yto, double opacity, int red, int green, int blue)
- void **line_blend** (int xfrom, int yfrom, int xto, int yto, double opacity, double red, double green, double blue)
- void **square_blend** (int xfrom, int yfrom, int xto, int yto, double opacity, int red, int green, int blue)
- void **square_blend** (int xfrom, int yfrom, int xto, int yto, double opacity, double red, double green, double blue)
- void **filledsquare_blend** (int xfrom, int yfrom, int xto, int yto, double opacity, int red, int green, int blue)
- void **filledsquare_blend** (int xfrom, int yfrom, int xto, int yto, double opacity, double red, double green, double blue)
- void **circle_blend** (int xcentre, int ycentre, int radius, double opacity, int red, int green, int blue)
- void **circle_blend** (int xcentre, int ycentre, int radius, double opacity, double red, double green, double blue)

- void **filledcircle_blend** (int xcentre, int ycentre, int radius, double opacity, int red, int green, int blue)
- void **filledcircle_blend** (int xcentre, int ycentre, int radius, double opacity, double red, double green, double blue)
- void **bezier_blend** (int startPtX, int startPtY, int startControlX, int startControlY, int endPtX, int endPtY, int endControlX, int endControlY, double opacity, double red, double green, double blue)
- void **bezier_blend** (int startPtX, int startPtY, int startControlX, int startControlY, int endPtX, int endPtY, int endControlX, int endControlY, double opacity, int red, int green, int blue)
- void **plot_text_blend** (char *face_path, int fontsize, int x_start, int y_start, double angle, char *text, double opacity, double red, double green, double blue)
- void **plot_text_blend** (char *face_path, int fontsize, int x_start, int y_start, double angle, char *text, double opacity, int red, int green, int blue)
- void **plot_text_utf8_blend** (char *face_path, int fontsize, int x_start, int y_start, double angle, char *text, double opacity, double red, double green, double blue)
- void **plot_text_utf8_blend** (char *face_path, int fontsize, int x_start, int y_start, double angle, char *text, double opacity, int red, int green, int blue)
- void **boundary_fill_blend** (int xstart, int ystart, double opacity, double boundary_red, double boundary_↩ green, double boundary_blue, double fill_red, double fill_green, double fill_blue)
- void **boundary_fill_blend** (int xstart, int ystart, double opacity, int boundary_red, int boundary_green, int boundary_blue, int fill_red, int fill_green, int fill_blue)
- void **flood_fill_blend** (int xstart, int ystart, double opacity, double fill_red, double fill_green, double fill_blue)
- void **flood_fill_blend** (int xstart, int ystart, double opacity, int fill_red, int fill_green, int fill_blue)
- void **polygon_blend** (int *points, int number_of_points, double opacity, double red, double green, double blue)
- void **polygon_blend** (int *points, int number_of_points, double opacity, int red, int green, int blue)
- void **plotCMYK_blend** (int x, int y, double opacity, double cyan, double magenta, double yellow, double black)
- void **plotCMYK_blend** (int x, int y, double opacity, int cyan, int magenta, int yellow, int black)
- void **laplacian** (double k, double offset)
- void **filledtriangle** (int x1, int y1, int x2, int y2, int x3, int y3, int red, int green, int blue)
- void **filledtriangle** (int x1, int y1, int x2, int y2, int x3, int y3, double red, double green, double blue)
- void **filledtriangle_blend** (int x1, int y1, int x2, int y2, int x3, int y3, double opacity, int red, int green, int blue)
- void **filledtriangle_blend** (int x1, int y1, int x2, int y2, int x3, int y3, double opacity, double red, double green, double blue)
- void **arrow** (int x1, int y1, int x2, int y2, int size, double head_angle, double red, double green, double blue)
- void **arrow** (int x1, int y1, int x2, int y2, int size, double head_angle, int red, int green, int blue)
- void **filledarrow** (int x1, int y1, int x2, int y2, int size, double head_angle, double red, double green, double blue)
- void **filledarrow** (int x1, int y1, int x2, int y2, int size, double head_angle, int red, int green, int blue)
- void **cross** (int x, int y, int xwidth, int yheight, double red, double green, double blue)
- void **cross** (int x, int y, int xwidth, int yheight, int red, int green, int blue)
- void **maltesecross** (int x, int y, int xwidth, int yheight, int x_bar_height, int y_bar_width, double red, double green, double blue)
- void **maltesecross** (int x, int y, int xwidth, int yheight, int x_bar_height, int y_bar_width, int red, int green, int blue)
- void **filleddiamond** (int x, int y, int width, int height, int red, int green, int blue)
- void **diamond** (int x, int y, int width, int height, int red, int green, int blue)
- void **filleddiamond** (int x, int y, int width, int height, double red, double green, double blue)
- void **diamond** (int x, int y, int width, int height, double red, double green, double blue)
- int **get_text_width** (char *face_path, int fontsize, char *text)
- int **get_text_width_utf8** (char *face_path, int fontsize, char *text)

Static Public Member Functions

- static double **version** (void)

The documentation for this class was generated from the following file:

- /Users/vinay/Google Drive/Science/Research/include/pngwriter.h

4.8 RANGE Struct Reference

Public Attributes

- double **minimum**
- double **maximum**

The documentation for this struct was generated from the following file:

- /Users/vinay/Google Drive/Science/Research/Morlets/Plot.cc

Chapter 5

File Documentation

5.1 /Users/vinay/Google Drive/Science/Research/include/wavelet.h File Reference

The supporting header file for generating the Continuous Wavelet Transform.

```
#include <stdio.h>
#include <stdlib.h>
#include <assert.h>
#include <string.h>
#include <math.h>
#include <fftw3.h>
#include <omp.h>
```

Macros

- #define QUAD_ROOT_PI 0.7511255444649425

  the quad root of pi (pi^ -0.25) computed to machine precision
- #define **C_SIGMA** 1.0000000000018794 /∗! \var C_SIGMA pow((1.0 + exp(-W_0_2) - 2.0 ∗ exp(-0.75 ∗ W_0_2)), -0.5) ∗/
- #define **K_SIGMA** 1.5229979744712628e-08
- #define **W_0** 6.0
- #define **W_0_2** 36.0
- #define **D_J** 0.125
- #define **FS** 2048
- #define **DT** 1.0/FS
- #define **S0** 2.0 ∗ DT
- #define **FREQ** 16.0
- #define **DATA_SIZE** 6144
- #define **MAX_FREQUENCY** 512.0
- #define **MIN_FREQUENCY** 0.5
- #define **MAX_DATA_SIZE** 10000000
- #define **MIN_I** FREQ_TO_SCALE(MAX_FREQUENCY)
- #define **MAX_I** FREQ_TO_SCALE(MIN_FREQUENCY)
- #define FREQ_TO_SCALE(x) floor((log2((W_0) / (S0 ∗ 2 ∗ M_PI ∗ x)))/D_J)

  Converts a given frequency x to a scale, handy for debugging. Note the scale is divided into sub octaves.
- #define SCALE_TO_FREQ(x) (W_0)/(x ∗ 2 ∗ M_PI)

  Converts a given scale x to its corrosponding frequency.
- #define MAGNITUDE(x, y) (x ∗ x) + (y ∗ y)

  Computes the 2- norm or the x ^ 2 + y ^ 2, of x and y.

Functions

- void FillData (double ∗data)

  Populates the input data array with a 3 sparse sine waves.
- void **TestCases** (double ∗data, int flag)
- int **ReadFile** (double data[ ], char ∗filename)
- int **WriteFile** (double ∗data, double ∗frequency, int x, int y, const char ∗filename)
- int **WriteDebug** (double ∗data, int length, const char ∗filename)
- int **ERSP** (double ∗data, double ∗scales, int sampling_frequency, int n, int J, int trials, double ∗output)
- void **Plot** (double ∗data, double ∗periods, int num_x, int num_y)
- double **CompleteFourierMorlet** (double w, double scale)
- int **Wavelet** (double ∗raw_data, double ∗scales, double sampling_frequency, int n, int J, double ∗result)
- void **CleanData** (double ∗data, double n)
- double ∗ **GenerateScales** (double minimum_frequency, double maximum_frequency)
- double ∗ **IdentifyFrequencies** (double ∗scales, int count)
- void **Convolute** (double ∗data, double ∗conWindow, double ∗complexWindow, double conSize, double ∗result, double ∗complexResult)
- int **CalculatePaddingSize** (int array_size, int FLAG)

5.1.1 Detailed Description

The supporting header file for generating the Continuous Wavelet Transform.

5.1.2 Function Documentation

5.1.2.1 void FillData (double ∗ *data*)

Populates the input data array with a 3 sparse sine waves.

Parameters

data	A 1 - dimentional block of memory that will be overwritten.

Sine Wave Sample

Sine Wave Sample

Sine Wave Sample

Index