Proceedings of the
7th Graz Brain-Computer Interface Conference 2017 DOI: 10.3217/978-3-85125-533-1-10

SIGVIEWER – CURRENT STATUS AND RECENT DEVELOPMENTS

C. Brunner[1], Y. Lin[2], P. Sajda[3], J. Faller[3]

[1]Institute of Psychology, University of Graz, Austria
[2]Manhattan School of Music, NY, USA
[3]Laboratory for Intelligent Imaging and Neural Computing, Columbia University, NY, USA

E-mail: clemens.brunner@uni-graz.at

ABSTRACT: SigViewer is an open source cross-platform biosignal viewer designed to visualize and annotate biomedical data streams. It supports a wide variety of file formats, including BDF, EDF, GDF, CNT, BrainVision, and BCI2000. Recently, support for loading multi-stream XDF data has been added. Besides visualizing raw data, SigViewer supports loading, displaying, creating, and editing events that can be used to annotate specific segments within a signal. Other useful tools include offset removal, computation of event-related potentials, and calculation of power spectral densities. To our knowledge, SigViewer is the only open source cross-platform multi-format biosignal viewer currently available that supports XDF files. Furthermore, SigViewer is completely free in that it does not depend on any proprietary software such as e. g. MATLAB. SigViewer is actively maintained and widely used across the globe (as measured by the monthly downloads). Filtering data in the frequency domain before visualization to e. g. remove line noise or excessive drift is one of the next planned features for a future release.

INTRODUCTION

Inspecting and visualizing raw biophysiological data such as EEG (electroencephalography), EOG (electrooculography) or ECG (electrocardiography) remains one of the first steps in any processing pipeline. Visualization can help in assessing general data quality, which includes detecting segments contaminated with artifacts, identifying noisy or completely bad channels, and inspecting events co-registered with the data.

Most available visualization tools are either tied to specific hardware, restricted to a small number of file formats, limited to a specific operating system, depend on proprietary programming environments, do not feature a fully featured GUI, and/or need to be purchased (i. e. are neither free nor open source). Relevant open source tools that focus on EEG/MEG analysis and ship with a visualization component include the MATLAB-based toolboxes EEGLAB [1], FieldTrip [2], Brainstorm [3], Biosig [4] (does not have a GUI[1]), and the Python package MNE [5] (does not have a GUI). Commercial tools include the MATLAB-based g.BSanalyze[2] as well as BrainVision Analyzer[3].

In contrast, SigViewer [6,7] is free, open source, cross-platform, supports many different file formats, and is written in standard-compliant C++ using the GUI toolkit Qt[4]. Development of SigViewer started some 10 years ago as a software project at Graz University of Technology, Austria. A first public version was uploaded to a dedicated SourceForge.net repository[5] in May 2010. The initial design goals of SigViewer can be summarized as follows (note that they are still valid today for the most recent release):

1. SigViewer should be cross-platform (i. e. it needs to run on Windows, Mac OS X, and Linux) with a native look-and-feel.

2. Visualization and interaction should be fast and responsive.

3. SigViewer should be completely free and open source (that is, all components required to build and run SigViewer should be available as open source). This specifically excludes MATLAB as a development environment, which many alternative viewers are based on.

An important use case for SigViewer is post-hoc manual signal inspection with the aim to identify and mark continuous artifact segments within the data. In a typical workflow, once artifact segments have been selected with SigViewer, they can be exported and thus integrated in any subsequent data analysis pipeline. Figure 1 illustrates this use case with SigViewer's main window displaying three EEG channels and one ECG channel, together with different events shown as rectangular areas in varying colors. Conveniently, these segments can be exported to a file for reuse in other stages of the processing pipeline.

[1] Graphical User Interface
[2] http://www.gtec.at/Products/Software/g.BSanalyze-Specs-Features
[3] http://www.brainproducts.com/productdetails.php?id=17
[4] https://www.qt.io/
[5] http://sigviewer.sourceforge.net/

FEATURES

SigViewer uses *libbiosig* from the BioSig project[6] to support various file formats. Specifically, SigViewer provides read access for the following file types: BDF (Biosemi), EDF (European Data Format), GDF (General Data Format), CNT (Neuroscan), EEG/VHDR/VMRK (BrainVision), and DAT (BCI2000). Recently, support for XDF (Extensible Data Format) has been added. Other file formats supported by *libbiosig*[7] may work but are not fully tested yet; official support for these formats may be added at a later stage.

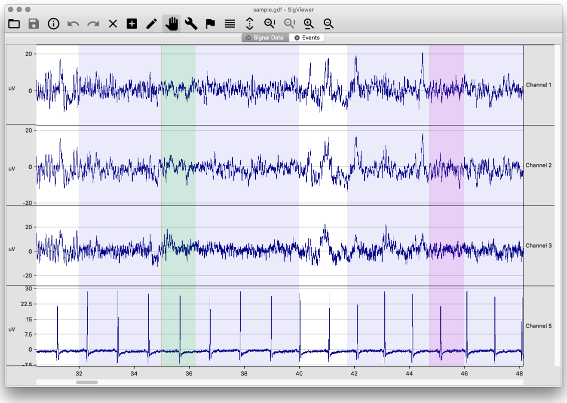


Figure 1: SigViewer running on Mac OS X.

After an initial search for global minimum and maximum values on a per-channel basis (required for auto-scaling the data), SigViewer renders the signal traces and events contained in the loaded file. Smooth and rapid scrolling through the file is supported, in contrast to many viewers which feature only page-wise (and thus relatively slow) navigation through a file.

In addition to visualizing the signals contained in a file, a list of events is automatically generated in a separate tab. Selected events can be deleted from this list.

Events can also be visualized, edited, and created within the main signal view. Events are marked with colored areas ranging from event onset to event offset. Multiple events occurring at the same time can be discerned by either assigning custom colors and/or alpha (transparency) levels. Events with duration zero are drawn as vertical lines. Events can be selected and edited graphically, which includes deleting and changing both the onset and offset as well as the event type. New events can be created by dragging the mouse pointer over the desired time range. Events can also be exported for use in other applications.

Basic meta information about the data can be displayed in a dedicated dialog window, which depends on the file type. Possible displayed fields include file type, recording time, patient ID, number of events, sample rate, channel labels, data types, physical dimensions of the recorded signals, and so on.

In addition to these visualization features, SigViewer also supports simple signal processing operations. Currently, power spectral densities as well as event-related potentials can be computed for selected channels. Both tools operate on (and thus require) selected event types in order to compute averages.

RECENT DEVELOPMENTS

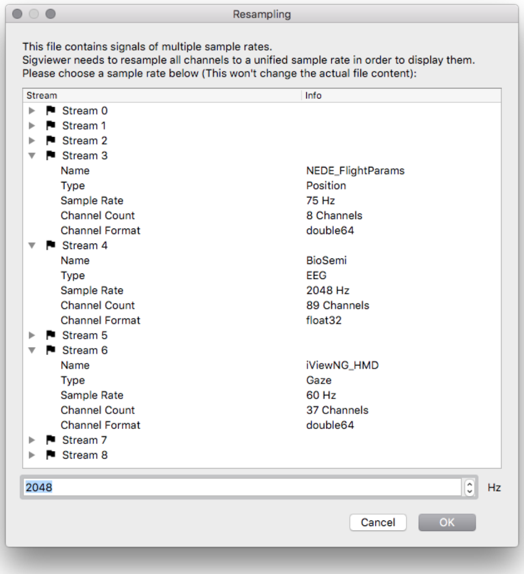
The latest stable release is 0.5.2, which has been available for almost four years. It can be downloaded from the old SourceForge project website, but the project has since moved to a new GitHub repository[8]. The latest source code, issues, pull requests, and new releases will only be available on the new GitHub website. The old SourceForge page will remain online until further notice.

Since this version, development has focused on adding support for XDF files. Other minor changes include switching to the latest version of the Qt library (from version 4 to version 5) and replacing all icons with a more modern monochromatic icon set.

XDF files are flexible XML-based containers that store multiple data streams with different sample rates and data types[9]. This makes XDF an ideal format for multimodal data, which combines different modalities such as EEG, eye tracking, motion capturing, and joystick and mouse tracking into multiple synchronized data streams. To import XDF files, SigViewer uses the dedicated library *libxdf*, which handles all necessary conversions required to reshape the data contained in an XDF file to a format that SigViewer can process. Specifically, *libxdf* resamples all data streams to a common sample rate, which can be set by the user (see Figure 2). For this purpose, SigViewer presents a dialog window when opening an XDF file, which shows all streams contained in the file with their sample rates and a suggested common sample rate. Irregular streams (that is, streams without a constant sample rate) are linearly interpolated for visualization. Streams containing strings are treated as events (and are therefore listed in the event table as well as plotted over the signals).

Another new feature is detrending of signals, which is often required for EEG signals that were recorded without a highpass filter. This works for many recordings, but a more general solution using highpass frequency filters is necessary for signals exhibiting a significant amount of non-constant low-frequency activity.

[6] `http://biosig.sourceforge.net/`
[7] `http://pub.ist.ac.at/~schloegl/biosig/TESTED`
[8] `https://github.com/cbrnr/sigviewer`
[9] `https://github.com/sccn/xdf`



Resampling

This file contains signals of multiple sample rates.
Sigviewer needs to resample all channels to a unified sample rate in order to display them.
Please choose a sample rate below (This won't change the actual file content):

Stream	Info
▶ Stream 0	
▶ Stream 1	
▶ Stream 2	
▼ Stream 3	
Name	NEDE_FlightParams
Type	Position
Sample Rate	75 Hz
Channel Count	8 Channels
Channel Format	double64
▼ Stream 4	
Name	BioSemi
Type	EEG
Sample Rate	2048 Hz
Channel Count	89 Channels
Channel Format	float32
▶ Stream 5	
▼ Stream 6	
Name	iViewNG_HMD
Type	Gaze
Sample Rate	60 Hz
Channel Count	37 Channels
Channel Format	double64
▶ Stream 7	
▶ Stream 8	

2048 Hz

Figure 2: Resampling dialog window for XDF files.

There are two ways to save events for XDF files; events are either stored in a new XDF file or appended to an existing XDF file. Another recent change regarding events is that SigViewer now exports event to plain-text CSV (comma separated values) files for all supported file formats. Previously, it was only possible to export to binary EVT files (which are essentially GDF files without signals). Such binary files are not straightforward to open, whereas CSV files can be opened with any plain-text editor (or imported and analyzed with more specialized software like Microsoft Excel, LibreOffice Calc, R, and Python/Pandas). Table 1 shows how such an exported CSV file might look like (note that -1 in the channel column means that an event is associated with all channels).

Table 1: Exported events example (names abbreviated).

position	duration	channel	type	name
767	2048	-1	768	Start
767	2048	-1	786	Cross
1535	0	-1	785	Beep
1535	320	-1	769	class1
1791	768	-1	781	Feedback
...	...	...	...	...

BUILDING SIGVIEWER

Building SigViewer is straightforward and completely relies on open source tools. However, the exact steps vary depending on which operating system is used. SigViewer runs on Windows, Mac OS X, and Linux. Official builds are available for Windows 10, Mac OS X (macOS) 10.9–10.12, and recent Debian/Ubuntu/Arch Linux distributions. Older versions of these three platforms may work, but are neither fully tested nor officially supported.

The source code of SigViewer is cross-platform, that is, it works on all operating systems mentioned before without further modifications. A prerequisite for building SigViewer is therefore a folder with the source code, which can be downloaded from the GitHub project site (e. g. via downloading a zipped file or cloning the repository).

On Windows, Qt 5.8 needs to be downloaded and installed together with the included MinGW 5.3 toolchain. External dependencies must be copied to the *external* folder within SigViewer's source folder. At the time of writing, *libbiosig* is a required dependency, but building it on Windows is somewhat involved. Therefore, pre-built binaries of this library are available on the old SourceForge repository[10]. The contents of this zipped file needs to be extracted into the SigViewer source folder, which automatically creates the required *external* folder structure. Next, SigViewer is ready to be compiled. The easiest way is to open the source tree with Qt Creator (the file *sigviewer.pro* needs to be opened). After selecting *Qt 5.8 MinGW 5.3* as the toolkit, the project can be built as a release by selecting *Build – Build Project "sigviewer"*. This creates an executable in the *bin/release* folder.

On macOS, XCode (available from the App Store) and the Command Line Tools (these can be installed by running `xcode-select --install` in a terminal) are required. The source of the external dependency *libbiosig* can be downloaded from the project website (this requires at least version 1.8.4b). In the *Makefile*, the following lines need to be adapted: lines 199 and 207 need to be commented out, and 10.7 needs to be changed to 10.9 in lines 148 and 151. Furthermore, the program *gawk* is required, which can be installed via Homebrew[11] or by downloading a pre-built binary such as the one available from Rudix[12]. Then, `make libbiosig.a` can be executed in a terminal. This creates the file *libbiosig.a*, which needs to be copied to SigViewer's source into *external/lib*. Similarly, the files *biosig.h* and *gdftime.h* need to be copied to *external/include*. After that, `qmake` can be run in a terminal within the SigViewer source tree, followed by `make`. This creates the app in the *bin/release* folder.

The procedure on Linux is almost identical to the one described for macOS. The only difference concerns the way required dependencies are installed. We recommend to use the native package manager to install a GNU toolchain with *g++*, *gawk*, and Qt 5.

Since details in this process can change rapidly with ongoing development, the most recent instructions on how to build SigViewer can be found on the project website. Note that we provide pre-built binary packages of SigViewer for Windows, macOS, and Linux for users who wish to skip the build process.

[10] `https://sourceforge.net/projects/sigviewer/files/0.5.2/external-0.5.2-win32.zip/download`

[11] `http://brew.sh/`

[12] `http://rudix.org/packages/gawk.html`

DISCUSSION

SigViewer is an actively maintained open source viewer for biosignals. The number of monthly downloads is around 250 on average and around 20,000 in total according to the download statistics available from SourceForge (see Figure 3 for the monthly downloads over the past six and a half years). Given that these downloads are for a version that was released almost four years ago, we expect these numbers to increase when the new version with support for XDF is released.

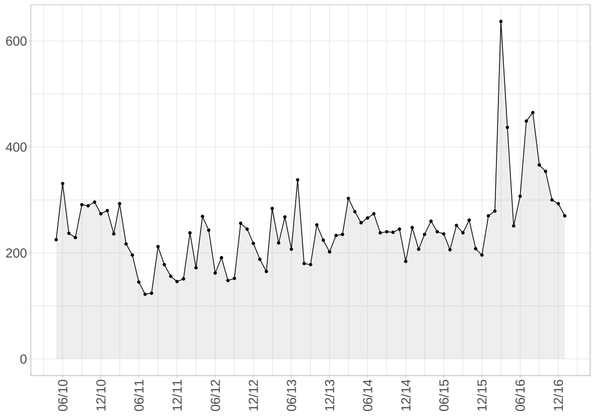


Figure 3: Monthly downloads from SourceForge. The total number of downloads since project upload until 2017-01-31 is 20079, and the average number of monthly downloads is 248.

Planned enhancements for the near future include adding support for spectral filters (highpass, lowpass, notch) to improve visualization of signals with known noise characteristics such as large low-frequency drifts or 50/60 Hz power line noise. Other future efforts will be directed to refactoring parts of the code base, for example to create a more suitable header structure for meta information (which facilitates displaying multi-rate streams without the need for resampling), to implement a hierarchical buffer which stores the signals in different resolutions for rapid visualization in different zoom levels, and to implement a more modular file reader interface.

CONCLUSION

We have added several useful features to SigViewer, most notably support for visualizing multi-stream XDF files. Continued development ensures that SigViewer will remain a valuable tool for inspecting raw biomedical data, which is an important stage in any signal processing pipeline including brain-computer interface (BCI) research.

ACKNOWLEDGMENTS

The authors declare that there is no conflict of interest. We would like to thank Thomas Brunner for starting development on SigViewer, for continued discussions on current issues, and for providing many helpful comments for this manuscript. In addition, we would like to thank Christoph Eibel for his contributions to SigViewer as part of his master's thesis.

REFERENCES

[1] Delorme A, Makeig S. EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics. J. Neurosci. Meth. 2004;134:9–21.
[2] Oostenveld R, Fries P, Maris E, Schoffelen J-M. FieldTrip: Open Source Software for Advanced Analysis of MEG, EEG, and Invasive Electrophysiological Data. Comp. Intell. Neurosci. 2011;2011:156869.
[3] Tadel F, Baillet S, Mosher JC, Pantazis D, Leahy RM. Brainstorm: A User-Friendly Application for MEG/EEG Analysis. Comp. Intell. Neurosci. 2011;2011:879716
[4] Schlögl A, Brunner C. BioSig: A Free and Open Source Software Library for BCI Research. IEEE Comput. 2008;41(10):44–50.
[5] Gramfort A, Luessi M, Larson E, Engemann D, Strohmeier D, Brodbeck C, Parkkonen L, Hämäläinen M. MNE software for processing MEG and EEG data. Neuroimage 2014;86:446–460.
[6] Brunner C, Schlögl A, Pfurtscheller G. SigViewer – an open source viewing and scoring program for biomedical signals, in Proc. 4th International Brain-Computer Interface Workshop and Training Course 2008, 396.
[7] Brunner C, Breitwieser C, Müller-Putz GR. SigViewer and SignalServer – open source software projects for biosignal analysis. Biomed. Eng./Biomed. Tech. 2013;58(S1).