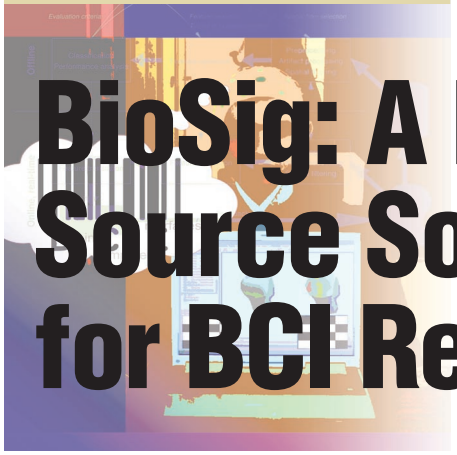

BioSig: A Free and Open Source Software Library for BCI Research

Alois Schlögl and Clemens Brunner
Graz University of Technology

The lack of common tools and standards has made software development a key issue in brain-computer interface research. With the BioSig project's tool repository, brain-computer interface researchers can avoid reinventing the wheel on every project.

The purpose of a brain-computer interface (BCI) is to identify the user's intention by observing and analyzing brain activity without relying on signals from muscles or peripheral nerves.[1] Researchers typically rely on electroencephalography (EEG)[1-3] to characterize brain activity, but they also use electrocorticography (EcoG), near-infrared spectroscopy (NIRS), or functional resonance imaging (fMRI). EEG is the most common because it is noninvasive, portable, can be used in almost any environment, and has excellent time resolution. Identifying the user's intention involves a complex chain of data-processing steps that is only as strong as the weakest link.

BCI researchers spend considerable time developing software to analyze data and evaluate different models of brain activity. The lack of a comprehensive repository of tools has made it difficult to reconcile data formats or demonstrate compatibility between certain neuroscientific concepts.

BCI COMPONENTS

Figure 1 illustrates a typical BCI. An online data-processing system controls devices in real time and provides feedback to the user. Providing feedback as fast and accurately as possible is critical. Any unnecessary noise or delay is adverse to the quality of the feedback and hinders users' abilities to train their brain patterns.

To generate the control signal, the BCI must extract and classify EEG features. The feature extraction method is based on the type of neurophysiological activation, while the classifier is typically obtained by offline analyses of previous data records from the same subject.

Every feature extraction method has its own hyperparameters. For example, model-based methods require selection of the model order, whereas fast Fourier transform (FFT)-based methods typically apply a smoothing window. Most classifiers also have hyperparameters—for example, it is necessary to select the number of hidden units for a multilayer perceptron or the number of supporting vectors for vector quantization. Optimizing such hyperparameters can be computationally demanding.

A BCI primarily uses spectral analysis—for example, of frequency band power or autoregressive spectra—to characterize spontaneous oscillatory EEG activity. It can also use autoregressive parameters directly to describe the entire spectral density function. In addition, a BCI can analyze the user's response to visual or acoustic stimuli, which can be presented one-by-one or in a steady-state (repetitive) mode. Because the stimulus rate determines the time resolution, a steady-state visual stimulus such as flickering is often used.

Data preprocessing is important to remove the influence of technical artifacts and nonbrain activity such as the electrical signals caused by eye movements or facial muscles. In the case of EEG recordings, spatial filters can also focus on a specific brain area or identify particular signal components.

A BCI uses offline analysis not only to estimate a reliable classifier, but to tune the processing steps' hyperparameters—for example, the adaptation speed, the feature extraction method's window length or frequency range, or some of the classifiers' regularization parameters. It can also compare and optimize different

Published by the IEEE Computer Society
0018-9162/08/$25.00 © 2008 IEEE

features or groups of features, as well as various classifiers and spatial filters, offline. To avoid overfitting data, researchers apply special cross-validation or resampling procedures. Several BCI competitions have provided data for testing different methods.[4,5]

A variant of the basic BCI computes and adapts the classifier online, which obviates the need for an additional data record without feedback and makes it possible to track the input signals' long-term nonstationarity.[6] Another version substitutes a simple threshold detector for the classifier. In this case, the investigator must select a single feature and, instead of training a classifier, the BCI selects the corresponding detection threshold.

BCI experiments typically use a cue-paced paradigm with a fixed trial length. To avoid such strict timing, some researchers have suggested self-paced or asynchronous experiments.[7] In some studies, a BCI uses class information for preprocessing as well as to compute the classifier. The most popular approach is calculating common spatial patterns from the class-specific covariance matrices of the data.[3,8]

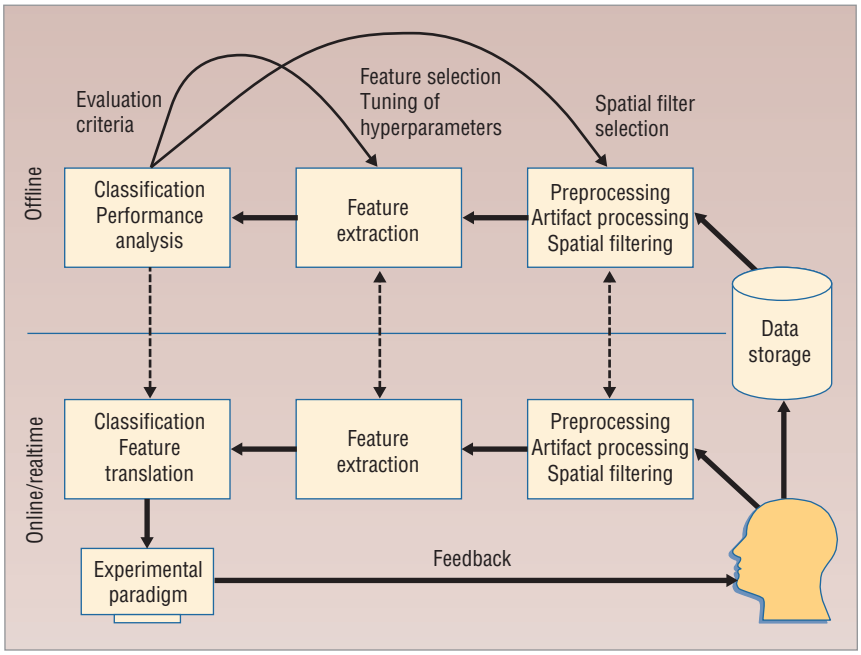

Figure 1. Typical brain-computer interface. A BCI uses an online data-processing system to control devices in real time and provide feedback to the user; it uses offline analysis to train the EEG feature classifier and optimize the various data-processing steps.

Table 1. Comparison of Matlab, Octave, and FreeMat.

Feature	Matlab	Octave	FreeMat
License	– (proprietary)	+ (GPL)	+ (GPL)
Speed	+	–	N/A
On-the-fly compression/decompression	–	+	?
Matlab executable interface	+	+	–
Graphical user interface	+	–	?
Integrated development environment	+	–	+

BCI RESEARCH PLATFORMS

Many BCI research labs use MathWorks' Matlab (www.mathworks.com), a popular tool for numerical data analysis, as their primary programming platform. GNU Octave (www.gnu.org/software/octave) and FreeMat (http://freemat.sourceforge.net) are free and open source alternatives to Matlab. Table 1 summarizes the strengths and weaknesses of all three BCI research platforms.

Although Matlab's integrated development environment and GUI are more sophisticated than those in Octave, these features are not critical for BCI research. Octave's principal advantage over Matlab is that it allows on-the-fly compression, is free software (with an unlimited number of licenses), and typically solves bugs faster. However, a BCI data analysis task takes four to five times longer in Octave than in Matlab (http://arxiv.org/abs/cs/0603001). Thus, Octave currently cannot fully replace Matlab for offline analysis or optimization of signal processing and classification methods. As for FreeMat, it lacks several crucial features, including output arguments for basic file I/O such as fseek and fread.

BIOSIG

Initiated in 2003, BioSig (http://biosig.sf.net) is a free and open source software library of biomedical signal-processing tools. The first releases were based solely on M-code for Matlab and Octave; this part of the library is now called BioSig for Octave and Matlab (biosig4octmat). BioSig for C/C++ (biosig4c++) was added later. Table 2 lists the main library components, most of which are implemented in M.

Although the biosig4octmat tools are very successful, there is a huge speed advantage in using functions implemented in C/C++ rather than in M. In addition, data acquisition software not based on M cannot use Matlab scripts for storing the data. Moreover, the US Federal Drug Administration and Health Level Seven, a healthcare standards group, have proposed an XML-

Table 2. BioSig software library components.

Component	Key features
Common interface	sload loads whole data files
	sopen, sread, and sclose load segments of data
	Common header structure
Data preprocessing	Triggering, partitioning of data
	Artifact processing
	Quality check of data through histogram analysis
	Spatial filters
	Detection of EMG artifacts
	Common spatial patterns
Feature extraction	Band power
	Adaptive autoregressive parameters
	Adaptive multivariate autoregressive parameters
	(Adaptive) Hjorth
	(Adaptive) Barlow
	(Adaptive) Wackermann
	(Adaptive) time-domain parameters
	Adaptive brain rate, spectral edge frequency
Feature classification	Linear discriminant analysis (LDA)
	Quadratic discriminant analysis (QDA)
	Support vector machines
	Naïve Bayesian classifier (NBC)
	Augmented NBC
	Sparse LDA
	Generalized discriminant analysis
Evaluation criteria	Classification accuracy
	Cohen's kappa coefficient
	Receiver operating characteristics (ROC)
	Area under the ROC curve
	Mutual information, information transfer rate
	Correlation coefficient
Metafunctions	findclassifier, cross-validation, standardized analysis
	demo2 (example of a standardized offline analysis)
Real-time data processing	Based on Matlab/Simulink and the Real-Time Workshop
	Provides standardized online BCI experiment
SigViewer	Stand-alone viewing and scoring software
	C++ and Qt

based data format for annotated electrocardiogram data (HL7aECG), but Matlab's and Octave's common XML tools are insufficient.

To address these limitations, biosig4c++ offers a group of tools for reading and writing biomedical-signal data formats implemented in C. Most of the data formats are little-endian but, like Matlab and Octave, biosig4c++ supports endian conversion for both big- and little-endian platforms. The tinyXML toolbox (http://tinyxml.sf.net) reads and writes the XML-based HL7aECG data format. The MEX interface and TCP/IP toolbox enable biosig4c++ users to load data into Matlab and Octave. The biosig4c++ tools can also be used in Python—for example via the simplified wrapper and interface generator (SWIG).

COMMON INTERFACE

BCI research labs typically use different data formats, which prevents the application of feature extraction and classification methods from one system to data recorded by some other system. The main issue is not the format of the sampled raw data, which can be converted, but the way researchers store the markers and state vectors. These are often user defined and must be decoded differently for each study.

BioSig currently provides a common interface to about 50 data formats supported by biosig4octmat and 25 supported by biosig4c++ (http://hci.tugraz.at/schloegl/biosig/TESTED). The most common data formats in BCI research are BCI2000 (www.bci2000.org), the BioSemi Data Format (BDF), the General Data Format (GDF, http://arxiv.org/abs/cs.DB/0608052), BrainVision, and various flavors of Matlab files.

GDF was designed so that all biosignal data formats can be converted into it without any loss of information. It is memory efficient, provides random data access, and, to minimize ambiguities, uses predefined codes for storing events and markers. GDF is the native data format of rtsBCI, the real-time system within BioSig. The BCI2000 recording system also supports GDF.

To facilitate support of different data formats, BioSig defines a common data structure for storing the file header information and defines standardized codes for events. Figure 2 shows the common header structure's fields, the details of which are described in the files biosig.h and header.txt. The description of the event codes is available in the file eventcodes.txt.

DATA PREPROCESSING

To address the common problem of artifacts in EEG recordings, BioSig provides automated overflow and saturation detection tools. If the data format itself does not provide information about the dynamic range, a method based on histogram analysis[9] can identify the upper and lower saturation values.

Because the high-frequency properties of facial-muscle activity violate the Nyquist theorem at typical sampling rates, detecting and rejecting these electromyography (EMG) artifacts is preferable to simple low-pass

GENERAL INFORMATION—FIXED HEADER

TYPE	File format
VERSION	Version number
FileName	Name of data file
FILE.OPEN	0: closed, 1: read mode, 2: write mode
FILE.POS	Position of file pointer in data segments or samples
T0	Date and time of start of recording
NS	Number of signals (channels)
SampleRate	Sampling rate
NRec	Number of records (data blocks)
SPR	Samples per record
Dur	Duration of each record

SUBJECT-SPECIFIC INFORMATION

Identification	Unique identification of subject or patient
VERSION	Version number
FileName	Name of data file
FILE.OPEN	0: closed, 1: read mode, 2: write mode
FILE.POS	Position of file pointer in data segments or samples
T0	Date and time of start of recording
NS	Number of signals (channels)
SampleRate	Sampling rate
NRec	Number of records (data blocks)
SPR	Samples per record
Dur	Duration of each record
Sex	Gender
Birthday	Subject's birthday
Handedness	0: unknown, 1: right, 2: left, 3: equal
Medication	Name of medication
Height	Height in centimeters
Weight	Weight in kilograms
...	Other patient-specific information

CHANNEL-SPECIFIC HEADER

Cal	Scaling, gain, calibration factor
Off	Offset
DigMin, DigMax	Measurement range (digital domain)
PhysMin, PhysMax	Measurement range (physical domain) PhysMin = DigMin*Cal + Off PhysMax = DigMax*Cal + Off
Label	Channel description
LeadIdCode	Standardized codes for channel description (ISO11073:10101/IEEE 1073)
PhysDim	Physical dimension (text)
PhysDimCode	Encoded physical dimension (ISO11073:10101/IEEE 1073)
Lowpass	Upper edge frequency
Highpass	Lower edge frequency
Notch	Notch filter Off/50Hz/60Hz
GDFTYP	Data type (3: int16, 5: int32, 16: float32, 17: float64)
SPR	Number of samples per record

EVENT TABLE

POS	(Starting) position of an event, marker, or annotation
TYP	Type of event according to predefined code table (1-255 is for user-defined events)
DUR	Duration of the event (optional)
CHN	0: not channel-specific, otherwise event related to specific channel
CodeDesc	Description of list user-specified event codes

Figure 2. Fields of the common header structure in BioSig. The common header structure facilitates support of different data formats.

filtering.[10] BioSig offers several tools to detect EMG artifacts. Detected artifacts are encoded as not-a-numbers, indicating missing values, and subsequent analysis methods are able to handle these missing values. A fully automated method based on regression analysis[11] is also available to reduce ocular artifacts.

Common average reference, local average reference, Hjorth's Laplace, or bipolar spatial filters depend on

the channel configuration. Because this does not always adhere to a fixed standard or is not available from the header information, no automated solution is available for these filters. Instead, researchers must define them for each specific electrode montage. Although spatial filters that use blind source separation methods do not depend on the channel configuration, the components have to be manually selected as well. Automated filters based on common spatial patterns are effective and widely used, but they require input from many EEG channels and can be unreliable in the long term.[12]

FEATURE EXTRACTION

Preprocessing and feature extraction are closely related, and a clear distinction between them is not always possible. In general, preprocessing methods do not introduce any time delay, while feature extraction methods use a sliding window approach to take into account the data's nonstationarity. The window size determines the time delay: A large window will allow only slow changes, while a small window is fast but comes at the expense of less accurate features. The challenge is obtaining an optimal tradeoff between speed and reliability.

The typical computational cost of a sliding window approach with a window size N is $O(N)$ per update step. An exception is adaptive estimation, which updates the past estimate with the current sample value only—this keeps the computational cost at $O(1)$ per update step. Segmentation methods also reduce computational effort but estimate the features at a lower data rate, thereby reducing time resolution. Adaptive methods are therefore preferable to segmentation and sliding window approaches. They also distribute computational needs more equally, which is advantageous in real-time and online BCI applications.

A key feature of EEG signals is the power of specific frequency bands. Traditionally, a BCI computes band power via FFT,[13] band-pass filtering and squaring,[14] or autoregressive spectral analysis. However, these techniques require preselecting the best discriminating frequency bands for each subject. To avoid this difficulty, researchers have developed adaptive autoregressive (AAR) parameters[2,15-17] that represent the whole spectrum. Whereas band power typically requires at least two hyperparameters, upper and lower edge frequency, the AAR model order is a single hyperparameter. BioSig includes several AAR estimators including least mean squares, recursive least squares, and Kalman filtering.

Traditional EEG processing uses several other time-domain parameters like Hjorth and Barlow parameters, global field descriptors, and spectral composite parameters such as brain rate and spectral edge frequency. These algorithms have been modified for adaptive use, making them suitable for online and real-time processing, as well as for use on data with missing values. Their availability through BioSig makes it much easier to compare different feature sets.[15] Future tools will address nonlinear properties and time-delayed correlation between channels.

FEATURE CLASSIFICATION

Typically, a BCI computes the feature classifier independently for each subject to take into account subject-specific properties—ideally during the short interval between experiments. This constraint often prevents the optimization of several hyperparameters. In practice, only one or at most two parameters can be optimized within each recording; all others must be fixed a priori.

Typically, a BCI computes the feature classifier independently for each subject to take into account subject-specific properties.

The first BCI classifiers relied on artificial neural networks.[2,13,18] These were succeeded by statistical classifiers that used linear discriminant analysis[2,14,17] and Mahalanobis discriminant analysis as well as support vector machines. BioSig contains several variants of LDA and MDA, linear and radial basis function SVMs, naïve Bayesian classifiers, sparse LDA, and generalized singular-value decomposition. GSVD techniques address the small sample-size problem—when the number of features exceeds the number of samples—and are thus promising candidates for investigating numerous features.

The logarithmic transformation of features with a non-normal distribution, such as band power and Hjorth or Barlow parameters, makes these features also suitable for use with linear classifiers. However, BCI researchers continue to pursue better techniques, and the common interface provided by BioSig facilitates comparison of different classifiers.

EVALUATION CRITERIA

Traditionally, BCI performance is quantified by classification accuracy or error rate. However, in the case of an unequal distribution of the classes due to different a priori probabilities, these metrics are unsuitable, and researchers have proposed Cohen's kappa coefficient, derived from the confusion matrix, as an alternative. In some cases, it is desirable to quantify BCI performance in terms of the information transfer rate. Other metrics are the correlation coefficient, mean square error, and area under the receiver operating characteristic curve.[11]

BioSig implements all of these evaluation criteria. Researchers are currently discussing the applicability of these metrics on asynchronous BCI systems.

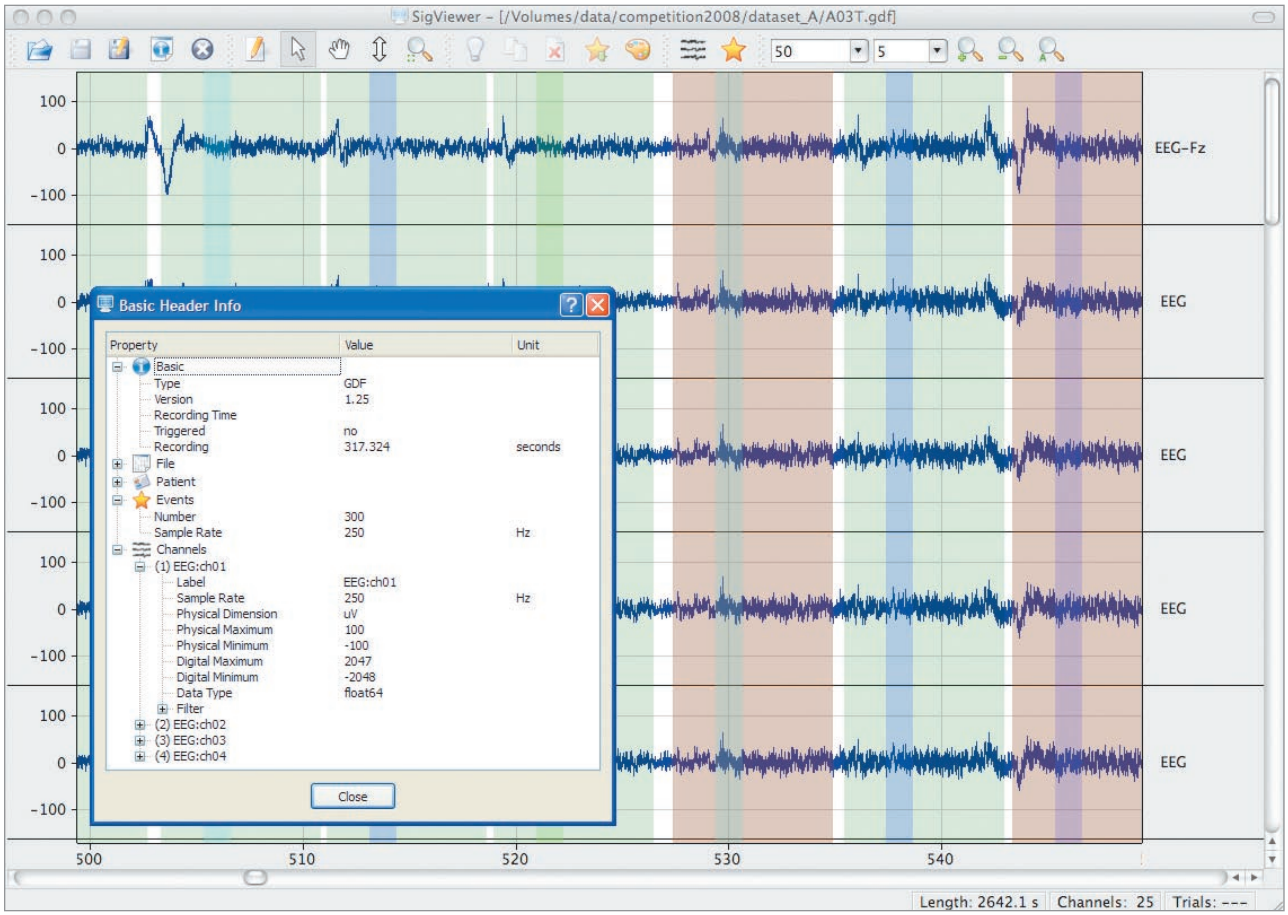

Figure 3. SigViewer screenshot. In addition to viewing biomedical signals, with BioSig users can create annotations to mark specific data segments. The window displays the open file's header information.

METAFUNCTIONS

A specific BCI design must combine components in a suitable way. In a typical motor imagery experiment, in which a subject is cued to perform a certain mental task out of numerous possibilities for several seconds, it is important to identify the optimal time window relative to some trigger point—typically presentation of the cue—to estimate the classifier. This optimal time window can also be used in the subsequent online feedback experiment, in which the subject is prompted to move a cursor toward the cue. Cross-validation techniques can assess the classifier's performance and average time course.

BioSig implements such a procedure in the wrapper function findclassifier.m. In combination with preprocessing and feature extraction methods (such as demo2), this provides a complete offline analysis of a motor imagery experiment. The same analysis can be applied to the feedback experiment.

BCI researchers are investigating the use of metafunctions to evaluate asynchronous experiments and experiments based on visual or steady-state evoked potential.

REAL-TIME DATA PROCESSING

Within the BioSig framework, the rtsBCI module is available to implement a real-time BCI system. This module is based on MathWorks' Simulink and Real-Time Workshop products and is well suited for rapid prototyping.[14] It features a ready-to-use data acquisition unit, easy exchange of signal processing methods, and numerous predefined techniques.

The rtsBCI module integrates with Matlab to efficiently compute and update new classifiers. It contains functions to correct for ocular and facial-muscle artifacts, estimate band power and AAR parameters, and control a virtual environment.[19,20] It also includes a ready-to-use two-class BCI experiment that tests a user's ability to guide a falling ball into a basket at the left or right corner of a computer screen.

SIGVIEWER

BioSig's viewing program for biomedical signals, SigViewer, is written in standards-compliant ISO C++ and uses only open source software. It features a convenient GUI implemented using the cross-platform library Qt (http://trolltech.com/products/qt) developed by Trolltech, now part of Nokia. SigViewer runs on many operating systems, including Linux X11, Microsoft Windows, and Mac OS X 10.5. SigViewer uses biosig4c++

for loading the EEG data; accordingly, SigViewer also supports the same data formats (currently 25).

The SigViewer screenshot in Figure 3 shows several EEG signals from a data file and a window with the open file's header information. In addition to viewing biomedical signals, users can create annotations or events, a process often referred to as *scoring*, to mark specific data segments continuously in time—for example, to tag a particular part of an EEG channel as an artifact.

Software development is a key issue in BCI research. Software can show the similarities and differences of different data processing methods. It can also make clear which hyperparameters must be determined for particular algorithms. And it can demonstrate whether certain concepts are compatible or not. With BioSig's comprehensive library of free and open source tools, combined with existing EEG databases—like those from BCI competitions—BCI researchers can avoid having to reinvent the wheel on every project.

References

1. J.R. Wolpaw et al., "Brain-Computer Interfaces for Communication and Control," *Clinical Neurophysiology,* June 2002, pp. 767-791.
2. G. Pfurtscheller et al., "Separability of EEG Signals Recorded during Right and Left Motor Imagery Using Adaptive Autoregressive Parameters," *IEEE Trans. Rehabilitation Eng.,* Sept. 1998, pp. 316-325.
3. B. Blankertz et al., "Optimizing Spatial Filters for Robust EEG Single-Trial Analysis," *IEEE Signal Processing Magazine,* Jan. 2008, pp. 41-56.
4. B. Blankertz et al., "The BCI Competition 2003: Progress and Perspectives in Detection and Discrimination of EEG Single Trials," *IEEE Trans. Biomedical Eng.,* June 2004, pp. 1044-1051.
5. B. Blankertz et al., "The BCI Competition III: Validating Alternative Approaches to Actual BCI Problems," *IEEE Trans. Neural Systems and Rehabilitation Eng.,* June 2006, pp. 153-159.
6. C. Vidaurre et al., "Study of On-line Adaptive Discriminant Analysis for EEG-Based Brain Computer Interfaces," *IEEE Trans. Biomedical Eng.,* Mar. 2007, pp. 550-556.
7. S.G. Mason and G.E. Birch, "A Brain-Controlled Switch for Asynchronous Control Applications," *IEEE Trans. Biomedical Eng.,* Oct. 2000, pp. 1297-1307.
8. H. Ramoser, J. Müller-Gerking, and G. Pfurtscheller, "Optimal Spatial Filtering of Single Trial EEG during Imagined Hand Movement," *IEEE Trans. Rehabilitation Eng.,* Dec. 2000, pp. 441-446.
9. A. Schlögl et al., "Quality Control of Polysomnographic Sleep Data by Histogram and Entropy Analysis," *Clinical Neurophysiology,* Dec. 1999, pp. 2165-2170.
10. B.S. Oken, "Filtering and Aliasing of Muscle Activity in EEG Frequency Analysis," *Electroencephalography and Clinical Neurophysiology,* July 1986, pp. 77-80.
11. A. Schlögl et al., "A Fully Automated Correction Method of EOG Artifacts in EEG Recordings," *Clinical Neurophysiology,* Jan. 2007, pp. 98-104.
12. M. Krauledat et al., "Reducing Calibration Time for Brain-Computer Interfaces: A Clustering Approach," *Proc. 20th Ann. Conf. Advances in Neural Information Processing Systems* (NIPS 06), MIT Press, 2007, pp. 753-760.
13. G. Pfurtscheller et al., "On-line EEG Classification during Externally-Paced Hand Movements Using a Neural Network-Based Classifier," *Electroencephalography and Clinical Neurophysiology,* Oct. 1996, pp. 416-425.
14. C. Guger et al., "Rapid Prototyping of an EEG-Based Brain-Computer Interface (BCI)," *IEEE Trans. Neural Systems and Rehabilitation Eng.*, Mar. 2001, pp. 49-58.
15. C. Vidaurre and A. Schlögl, "Comparison of Adaptive Features with Linear Discriminant Classifier for Brain Computer Interfaces," to appear in *Proc. 30th Ann. Int'l Conf. IEEE Eng. in Medicine and Biology Society* (EMBS 07), IEEE Press, 2008.
16. A. Schlögl, *The Electroencephalogram and the Adaptive Autoregressive Model: Theory and Applications*, Shaker Verlag, 2000.
17. A. Schlögl, C. Neuper, and G. Pfurtscheller, "Subject-Specific EEG Pattern during Motor Imagery," *Proc. 19th Ann. Int'l Conf. IEEE Eng. Medicine and Biology Society* (EMBS 97), IEEE Press, 1997, pp. 1530-1532.
18. D. Flotzinger, J. Kalcher, and G. Pfurtscheller, "EEG Classification by Learning Vector Quantization," *Biomedizinische Technik,* Dec. 1992, pp. 303-309.
19. R. Scherer et al., "The Self-Paced Graz Brain-Computer Interface: Methods and Applications," *Computational Intelligence and Neuroscience*, Jan. 2007; www.aksioma.org/brainloop/txt/computational_intelligence_neuroscience.pdf.
20. R. Scherer et al., "Towards Self-Paced Brain-Computer Communication: Navigation through Virtual Worlds," *IEEE Trans. Biomedical Eng.,* Feb. 2008, pp. 675-682.

Alois Schlögl *is a postdoctoral researcher and lecturer at the Institute for Human-Computer Interfaces, Graz University of Technology. His research interests include neuroscience, medical informatics, and biomedical signal processing. Schlögl received the habilitation in medical informatics from Graz University of Technology. He is a member of the IEEE Engineering in Biology and Medicine Society. Contact him at a.schloegl@ieee.org.*

Clemens Brunner *is a postdoctoral researcher at the Institute for Knowledge Discovery, Graz University of Technology. His research interests include brain-computer interfaces, biomedical signal processing, and software development. Brunner received a PhD in computer science from Graz University of Technology. Contact him at clemens.brunner@tugraz.at.*