Wearable EEG via lossless compression

Guillermo Dufort, Federico Favaro, Federico Lecumberry,
Álvaro Martín, Juan P. Oliver, Julián Oreggioni,
Ignacio Ramírez, Gadiel Seroussi and Leonardo Steinfeld*†

Abstract

This work presents a wearable multi-channel EEG recording system featuring a lossless compression algorithm. The algorithm, based in a previously reported algorithm by the authors, exploits the existing temporal correlation between samples at different sampling times, and the spatial correlation between different electrodes across the scalp. The low-power platform is able to compress, by a factor between 2.3 and 3.6, up to 300sps from 64 channels with a power consumption of 176μW/ch. The performance of the algorithm compares favorably with the best compression rates reported up to date in the literature.

1 Introduction

Monitoring brain activity can play an important role for understanding the functioning of human brain, as well as in preventing mental disorders or improving our quality of life. Nowadays the electroencephalogram (EEG) is the main method used for studying brain dynamics. However, current standard EEG systems are wired and uncomfortable, mainly used in clinical practice. In order to enable EEG recordings in daily-life activities, EEG technology needs to become wearable (wireless, low weight and small size), which means low-power operation and energy-efficient wireless data transmission [1].

The IEC standard specifies a minimum bandwidth of 0.5Hz to 50Hz for EEG equipment. This covers the most common diagnostic purposes, but higher bandwidths (from DC to 500Hz) may be required in other cases [1, 2]. In addition, the current miniaturization of analog front-ends (AFE) for acquiring EEG signals allows to record hundreds of channels. Thus, it becomes necessary to efficiently handle high data rates. For example, 64 channels, 12-bit per sample, at 1ksps, implies a data rate of 768kbps, which is an attainable throughput by Bluetooth; other low-power transmission protocols are unable to handle it (e.g.: IEEE 802.15.4). Moreover, currently, there are no low-power transmission protocols that support 256 channels (data rate of 3Mbps).

*This work was partially funded by CSIC-UDELAR (Comisión Sectorial de Investigación Científica, Universidad de la República, Uruguay), ANII (Agencia Nacional de Investigación e Innovación, Uruguay) and CAP-UDELAR (Comisión Académica de Posgrado, Universidad de la República, Uruguay).

†Guillermo Dufort, Federico Favaro, Federico Lecumberry, Álvaro Martín, Juan P. Oliver, Julián Oreggioni, Ignacio Ramírez, Gadiel Seroussi and Leonardo Steinfeld are with the Facultad de Ingeniería, Universidad de la República. Montevideo, Uruguay.

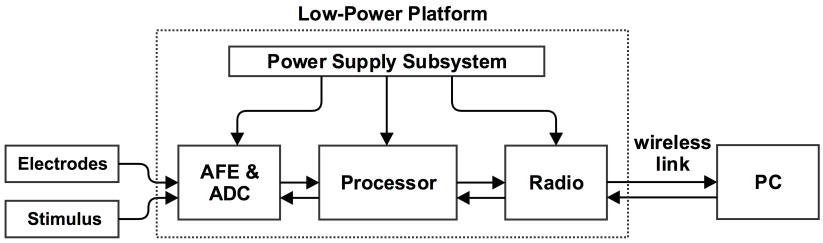

Figure 1: Weareable EEG recording system block diagram.

For the above reasons, compression is a key factor in a low-power platform, not only for reducing power consumption (in these platforms the power consumption is usually dictated by the transmission), but also to achieve higher data rates with the same power budget.

EEG data acquired for clinical purposes is often required to be procesed without distortion. This in turn leads to a need for lossless compression algorithms [3].

At the time of this writing, a wearable EEG recording platform for neuroscientific applications (see Fig. 1) is being constructed. This low-power platform comprises an analog front-end of 16 or 64 channels, an analog to digital converter (ADC), a processor block and a radio transceiver (some details of our platform will be presented in Section 2). By means of a Bluetooth link, the collected data is wirelessly transmitted to a PC, where a physician and/or a neuro-scientific can analyze it.

This work is focused in increasing the quality of the EEG data (increase the number of channels and/or the sample frequency) without jeopardizing the power consumption, by means of a lossless compression algorithm, firstly described by the authors in [3]. This algorithm and the modifications performed to run it in our low-power platform are discussed in Section 3. Results of compression level, memory usage, processing time and power consumption are disclosed in Section 4. Finally, conclusions are presented in Section 5.

2 Low-power platform

The core of the low-power platform is the processor (shown in Fig. 1), that is mainly dedicated to run the compression algorithm. A processor suitable for implementing a wearable EEG must satisfy very challenging and usually conflicting requirements:

- 16-bit or higher data width, in order to efficiently handle scalar 12-bit samples at minimum.
- Low power consumption, needed for an extended life-time battery. The total platform consumption should be below 8.3mW for 30 days of continuous use, powered by two AA batteries. This means 130μW per channel for 64 channels, in accordance to the power requirement estimation made in [2].

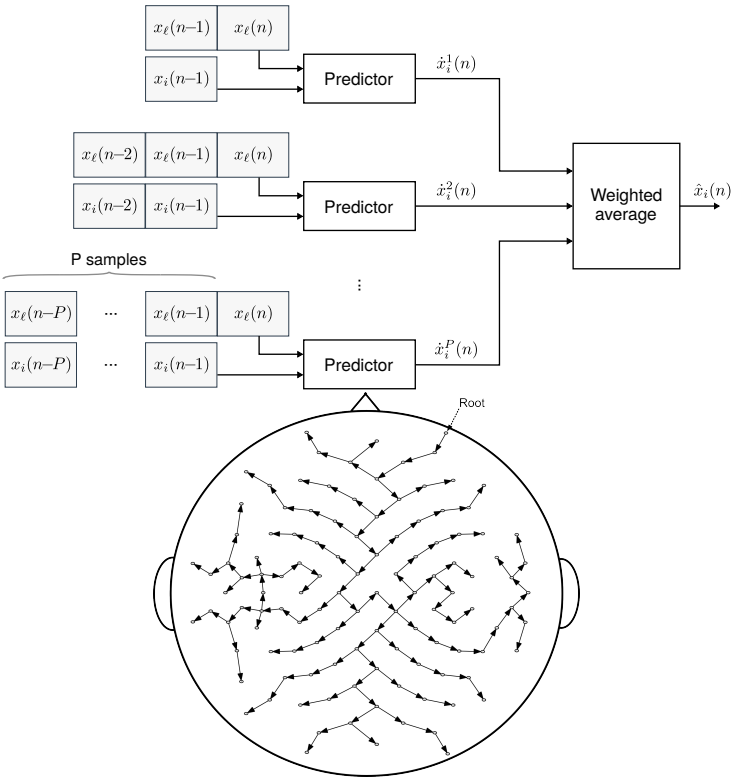

Figure 2: Compression algorithm. LEFT: block diagram of the prediction scheme; here $x_i(n)$ refers to the value of channel i at discrete time n, x_ℓ is the "helper" (parent) channel of x_i, P is the order of the filters, $\dot{x}_i^p$ is the p-th order prediction of x_i and $\hat{x}_i$ is the final prediction for that channel. RIGHT: sample tree used when deciding which channel helps which; the root channel is encoded with no help.

- Enough performance to ensure a computation time per sample low enough to process the total number of channels at the desired sample rate. For instance, the processing time to handle 64 channels at 1ksps needs to be less than 31.25μs per channel.

Depending on the compression algorithm, a hardware floating-point unit (FPU) can be necessary (this requirement can be eliminated by implementing floating point operations by software, at a higher performance requirement).

Nowadays, there are increasingly available commercial off-the-shelf microprocessors targeting low-power applications, and at the same time, relative high-performance processing capabilities. Still, higher clock frequencies demand higher supply voltages and thus higher power consumption. In addition to a core processor, common microcontrollers include programmable I/O peripherals (such as serial ports needed to receive the sampled data from the AFE) and memory (to store code and data). However, low-power microcontrollers have memory of relatively low size, since system memory is responsible for a large portion of the total energy budget [4]. Thus, an algorithm with a reduced

memory footprint is mandatory to achieve low power consumption.

We selected a MSP432P401R microcontroller, a 32-bit ARM Cortex-M4F running up to 48 MHz, manufactured by Texas Instruments. This microcontroller has 256kB of Flash and 64kB of RAM memory, and features a FPU with single cycle multiply and accumulate instructions and hardware support for single-precision square root, and so on. For the experiments, we used the evaluation kit *MSP432P401R LaunchPad*, which includes the *EnergyTrace* tool to measure the total system energy consumption.

The integrated development environment (IDE) used was Code Composer Studio (CCS) from Texas Instruments, which supports *EnergyTrace* and allows to configure various toolchains. The different versions of the algorithms were built using the GNU v4.8.4 (Linaro) compiler.

3 EEG Compression algorithm

Table 1: Platform performance depending on the compression algorithm version.

Algorithm	Number of channels	Proc. time per sample (ms)	Max sampling rate (sps)	Memory usage FLASH (kB)	Memory usage RAM (kB)
FLO-SW	16	40.18	25	44.9	26.7
FLO-SW	64	–	–	45.2	103.1
FLO	16	3.11	321	42.5	26.7
FLO	64	–	–	42.8	103.1
FIX	16	2.81	355	46.7	16.6
FIX	64	11.55	87	47.0	60.5
FIX2	16	0.83	1201	42.1	10.2
FIX2	64	3.83	296	42.4	33.8

As mentioned in the previous sections, the lossless, real time and low power requirements of our platform impose severe restrictions on the latency and computational resources of its embedded software. In order to meet such requirements, we adopt the algorithm described in [3] as our starting point, which is a low-latency, low-complexity algorithm with controllable per-sample distortion offering the best compression rates reported up to date in the literature.

As most EEG compression algorithms, the method in [3] exploits the existing temporal (between samples at different sampling times) and spatial (between different electrodes across the scalp) correlations. These are induced by natural properties of the target signal such as temporal continuity, natural correlation of neural activity across regions, and spatial smoothing of the latter signals due to the different layers of tissue that separate the source signals (the neurons) to the point where they are measured (the electrodes).

The essence of the algorithm (see Fig. 2), referred to as FLO hereafter, can be summarized as follows (we refer the interested reader to [3] for further details):

- The coding stage is predictive: both encoder and decoder predict the value of each sample from previously encoded samples; the actual value is described to the decoder by encoding the difference with respect to the prediction using the *Golomb-Rice code* (see, e.g., [5]).

- Channel samples are encoded in a pre-specified order following a tree; the root channel is predicted using past samples only, whereas all other channels have a *parent channel* (corresponding to their parent in the tree) which "helps" them, meaning that the past (and present) information about the parent channel is used for predicting the present sample of the child channel.

- Each sample prediction is a weighed average of a set of adaptive linear predictions of different orders, which are combined using an *exponential weighting scheme* [6] to form a final prediction.

- All adaptive linear predictions are computed using an efficient *lattice* implementation of the *multi-channel Recursive Least Squares algorithm* [7], which computes all (least-squares optimal) predictions of up to a fixed order p efficiently.

The FLO requires on average $1\mu s$ to encode (or decode) a single scalar sample, that is, about $C\ \mu s$ for encoding (decoding) an EEG sample composed of C scalar channel samples, on a common desktop computer (Intel i7 at 3.4GHz, on a single thread). Although this much more than meets the requirements on the desktop side of the platform (it would allow us to go beyond 10kHz with 64 channels), the FLO algorithm as originally proposed falls short of meeting them when ported to a low-power microcontroller running at a few tens of MHz.

In order to do this, we have introduced a number of significant changes to the original algorithm, resulting in a new algorithm which we refer to as FIX. More specifically:

- the lattice RLS predictor is replaced by a multichannel predictor that is adapted heuristically and requires only integer additions and multiplications,

- the remaining floating-point arithmetic is removed by approximating the exponential weighting scheme by one requiring only register shifts,

- memory is greatly reduced by approximating moving averages by a simple first order low-pass filter.

The above FIX algorithm reduces all computational requirements significantly, most importantly memory requirements, while maintaining the compression ratio essentially unchanged. However, simplifications are required in order to meet the goals set forth in this work. We will therefore refer to the algorithm that was finally implemented as FIX2; the main difference with FIX is that we have empirically selected a reduced set of predictors that are combined to form the final prediction for each sample. In FIX2 we used four predictors, which yields a very significant performance improvement at the cost of a slight compression rate degradation.

4 Results

Three versions of the algorithm described in Section 3 will be presented: FLO, FIX and FIX2. Each algorithm was initially developed on a desktop computer and later the source code was migrated to run on the selected platform.

The input-output interfaces were modified to avoid the utilization of a file system. Two versions of these interfaces were developed, the first one allows to read the samples from flash memory and the second allows read and write samples through a serial port using the internal UART of the microcontroller.

Time measurements were performed with the CCS Event Count tool, counting machine cycles between two breakpoints. The clock frequency of the MSP432 was set at 48MHz in all cases. The three versions of the algorithm were verified comparing the output data with the desktop version.

In order to asses the impact of using a FPU in the processing time, the same source code of the starting point algorithm was compiled using the FPU, version named FLO, and using emulated floating point operations from the *newlib* library with the FPU turned off, named FLO-SW.

In Table 1 the platform performance is presented in terms of processing time and memory usage, for the different compression algorithms. The third column shows the measured average time spent to process all channels, and the fourth column indicates the computed maximum sampling rate (calculated assuming that the microcontroller is always in active mode). In both cases, each sample is formed by 16 or 64 values as indicated in column two.

FLO-SW can handle up to 16 channels, because the RAM memory needed to process 64 channels ($103.1kB$) exceeds the available memory of the selected platform ($64kB$). Moreover, FLO-SW performs very poorly, and at 16 channels, it can only process up to $25sps$.

The use of the FPU (FLO) reduces the processing time, achieving a reasonable sample rate ($321sps$), but the limitation on the maximum number of channels remains (due to memory restrictions). The release of FIX implied an important effort in reducing computing complexity for suppressing floating point data and operations. This effort made it possible to run the compression algorithm for 64 channels in our low-power platform. Although the 16-channel processing time was acceptable (enabling a maximum sample rate of $355sps$), the 64-channel processing time exceeded our design goal (maximum sample rate of $87sps$). This issue was subsequently addressed by the FIX2 algorithm.

FIX2 presents a reduced memory footprint, $10.2kB$ and $32.8kB$ for 16 and 64 channels respectively, freeing RAM memory for other important functions (i.e. I/O buffers for acquisition and transmission). Moreover, much more importantly, FIX2 presents an outstanding improvement in terms of processing time with respect to FIX and FLO, achieving $1201sps$ and $296sps$ for 16 and 64 channels respectively. Finally, it is interesting to note that the number of channels and the processing time are linear.

A sample rate of $296sps$ for each channel, exceeds the minimum requirement for standard EEG recording. At the same time, processing 64-channels at $296sps$ in our platform, implies that the microcontroller draws a current of $3.75mA$ powered from $3V$, which represents a power consumption of $176\mu W/ch$ for the processor block. This current was measured with a Fluke 45 amperimeter while the processor was running the compression algorithm continuously. The same measure was performed with *EnergyTrace* and the current was $3.82mA$. These measurements are in accordance with the typical value $4.6mA$ reported in the microcontroller datasheet, presenting a very low relative error.

We evaluated the performance of FLO and FIX2 in terms of compression ratio using the following publicly available databases:

- DB1a and DB1b [8, 9]: 64-channel, 160Hz, 12bps EEG of 109 subjects

using the BCI2000 system. Recordings are divided in 2-minute motor imagery task (DB1a) and 1-minute calibration (DB1b).

- DB2a and DB2b [10] (BCI Competition III): 118-channel, 1000Hz, 16bps EEG of 6 subjects performing motor imagery tasks (DB2a). DB2b is a 100Hz downsampled version of DB2a.
- DB3 [11] (BCI Competition IV): 59-channel, 1kHz, 16bps EEG of 7 subjects performing motor imagery tasks.
- DB4 [12]: 31-channel, 1kHz, 16bps EEG of 15 subjects performing image classification and recognition tasks.

For each database, we compressed each data file separately and calculated the overall *Compression Rate (CR)* in bits per sample, $R = L/N$, where N is the the sum of the number of scalar samples over all files of the database, and L is the sum of the number of bits over all compressed files of the database. Alternatively, the results can be expressed in terms of the *Compression Ratio (CR%)*, $CR = 100 \times L/S$, where S is the total number of bits required to store all the uncompressed files in the database. Note that, in both cases, *smaller means better*.

Table 2: Compression Rate (bits per sample) of FLO and FIX2 for different databases.

Algorithm	DB1a	DB1b	DB2a	DB2b	DB3	DB4
FLO	$4,77$	$4,86$	$5,44$	$7,07$	$5,57$	$3,72$
FIX2	4.87	4.98	5.44	7.07	5.57	3.84

Table 2 shows that the compression rate of FIX2 is marginally higher than that of FLO. This means that the memory and time requirements of the compression algorithm were drastically reduced with almost no degradation in compression performance. Therefore, according to [3], FIX2 is well in line with the best compression rates reported up to date in the literature. In terms of compression ratio, the largest value 43.8% was obtained for DB2b, whereas the smallest value was 27.9% for DB4. This is implies that the transmitted data can be reduced by a factor between 2.3 and 3.6 respectively.

5 Conclusions

Firstly, we have presented a successful implementation, in a low-power platform, of a lossless multichannel EEG compression algorithm for 16 channels (FLO), with a compression ratio well in line with the best ones reported in the current literature.

Our multidisciplinary team, including researchers from information theory, computer science, digital signal processing, embedded systems, digital electronics, and analog microelectronics, was then faced with the challenge of changing the algorithm in order to handle 64 channels in a low-power platform while preserving the compression rate; the result, algorithm FIX2, achieves those goals.

Extending the use of EEG systems beyond clinical practice requires a technology capable of wireless, long-lasting low-power operation. In this work, we

have approached this goal with a platform able to compress, by a factor between 2.3 and 3.6, almost 300*sps* from 64-channel with a power consumption of 176μW/ch.

Future work includes assesing the low-power platform with other physiological signals. For example, results of the compression algorithm for electrocardiogram signals are very promissory [13].

References

[1] V. Mihajlovi, B. Grundlehner, R. Vullers, and J. Penders, "Wearable, wireless EEG solutions in daily life applications: What are we missing?," *IEEE Journal of Biomedical and Health Informatics*, vol. 19, no. 1, pp. 6–21, Jan 2015.

[2] A. J. Casson, D. C. Yates, S. J. M. Smith, J. S. Duncan, and E. Rodriguez-Villegas, "Wearable electroencephalography," *IEEE Engineering in Medicine and Biology Magazine*, vol. 29, no. 3, pp. 44–56, May 2010.

[3] I. Capurro, F. Lecumberry, A. Martin, I. Ramirez, E. Rovira, and G. Seroussi, "Low-complexity, multi-channel, lossless and near-lossless EEG compression," in *2014 Proceedings of the 22nd European Signal Processing Conference (EUSIPCO), 1- 5 sep 2014, Lisbon, Portugal*. 2014, pp. 2040–2044, IEEE.

[4] G. Chen, F. Li, M. Kandemir, O. Ozturk, and I. Demirkiran, "Compiler-Directed Management of Leakage Power in Software-Managed Memories," in *IEEE Computer Society Annual Symposium on Emerging VLSI Technologies and Architectures (ISVLSI'06)*. 2006, vol. 00, pp. 450–451, IEEE.

[5] B. Carpentieri, M. J. Weinberger, and G. Seroussi, "Lossless compression of continuous-tone images," *Proceedings of the IEEE*, vol. 88, no. 11, pp. 1797–1809, Nov 2000.

[6] A.C. Singer and M. Feder, "Universal linear prediction by model order weighting," *IEEE Trans. Sig. Processing*, vol. 47, no. 10, pp. 2685–2699, Oct 1999.

[7] G.-O. Glentis and N. Kalouptsidis, "A highly modular adaptive lattice algorithm for multichannel least squares filtering," *Signal Processing*, vol. 46, no. 1, pp. 47–55, Sept. 1995.

[8] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. Ch. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, "PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals," *Circulation*, vol. 101, no. 23, 2000 (June 13).

[9] G. Schalk, D.J. McFarland, T. Hinterberger, N. Birbaumer, and J.R. Wolpaw, "BCI2000: a general-purpose brain-computer interface (BCI) system," *IEEE Trans. Biomedical Engineering*, vol. 51, no. 6, pp. 1034–1043, June 2004.

[10] G. Dornhege, B. Blankertz, G. Curio, and K. Muller, "Boosting bit rates in noninvasive EEG single-trial classifications by feature combination and multiclass paradigms," *IEEE Trans. Biomedical Engineering*, vol. 51, no. 6, pp. 993–1002, June 2004.

[11] B. Blankertz, G. Dornhege, M. Krauledat, K.-R. Mller, and G. Curio, "The non-invasive Berlin braincomputer interface: Fast acquisition of effective performance in untrained subjects," *NeuroImage*, vol. 37, no. 2, pp. 539 – 550, 2007.

[12] A. Delorme, G. A. Rousselet, M. J.-M Mac, and M. Fabre-Thorpe, "Interaction of top-down and bottom-up processing in the fast visual analysis of natural scenes," *Cognitive Brain Research*, vol. 19, no. 2, pp. 103 – 113, 2004.

[13] Ignacio Capurro, Federico Lecumberry, Álvaro Martín, Ignacio Ramírez, Eugenio Rovira, and Gadiel Seroussi, "Efficient sequential compression of multi-channel biomedical signals," p. 13, may 2016, http://arxiv.org/abs/1605.04418.