PART I

HUMAN BODY MONITORING

1

INTERFACING BIOLOGY AND CIRCUITS: QUANTIFICATION AND PERFORMANCE METRICS

Alexander J. Casson and Esther Rodriguez-Villegas

1.1 INTRODUCTION

A key aim of bioelectronics is to provide an interface between the biological world (blood pressure, electrocardiogram [ECG], and the like) and the electronics world (analog and digital hardware, software, and the like). This interface allows the characterization and quantification of the biological world, which can be used to gain further understanding of the fundamental biological processes being monitored. Alternatively, long-term monitoring of physiological parameters can lead to new and more effective diagnostic and treatment methods for particular medical conditions.

A typical interface between the biological and electronic worlds is shown in Figure 1.1. Here a suitable sensor or electrode is used to detect a biological parameter, and the resulting signal is then amplified and converted into the digital domain. Once this has been done, the signal can be transferred to a computer for long-term storage and processing. Depending on the application requirements, the data may be transfered over cables or via a wireless link.

The biological world interface system thus includes everything from the sensor to the wired or wireless link. In many applications this system must be as physically small as possible, and capable of operating autonomously over long periods of time. This may be because data is being collected from a lab animal that is physically small, or because a human is being monitored, and they are

CMOS Biomicrosystems: Where Electronics Meet Biology, First Edition. Edited by Krzysztof Iniewski.

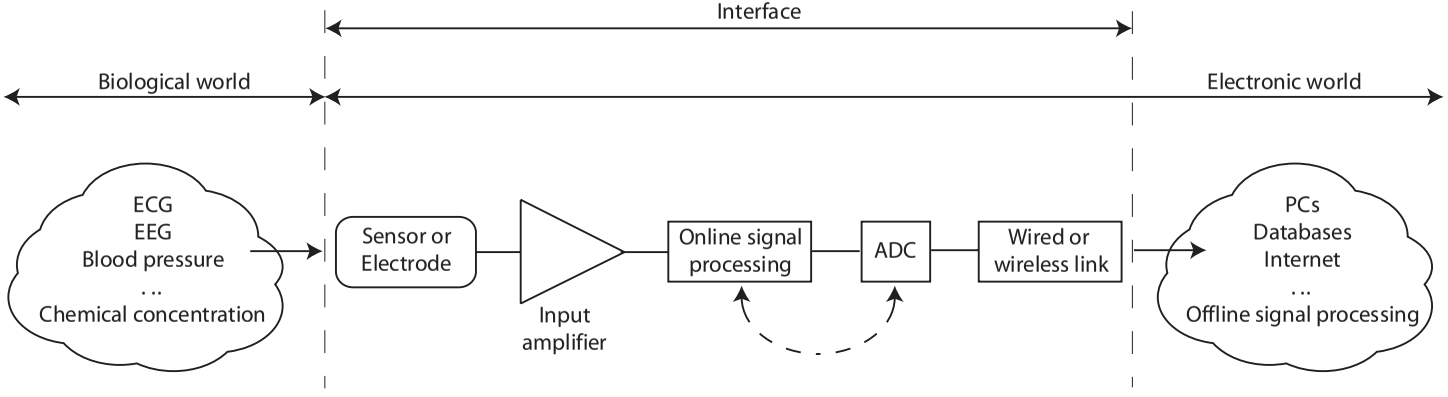


Figure 1.1. A typical interface system between the biological world and the electronic world. Physiological parameters are sensed, amplified, and converted to digital signals, which can be stored and processed on a computer. Online signal processing, implemented in either the analog or digital domain, can be of significant use in enabling device miniaturization.

expected to be going about their normal daily life. For this to be possible, the interface device must be unobtrusive, comfortable, socially acceptable, and long lasting.

Miniaturized, unobtrusive devices imply that only physically small batteries, which have limited energy storage and current sourcing capabilities, are available for use. Simultaneously, long-term monitoring implies that the limited energy capacity of the batteries has to exploited to the maximum by using very low-power electronics. Key for realizing these miniaturized interface systems is thus the optimization of the electronic design. For example, a system using only a 12-bit analog-to-digital converter (ADC) produces much less data to transmit and leads to much lower overall power consumptions than systems using a 24-bit ADC. However, performing such optimizations inevitably requires detailed knowledge of the biological requirements for the given application, and obtaining these requirements is by no means trivial. As an example, consider the electroencephalogram (EEG), which records electrical potentials from the scalp.

Recommendations from the International Federation of Clinical Neurophysiology call for a 12-bit (72 dB) sampling resolution once the direct current (DC) component of the signal has been removed [1]. Most commercial EEG units use 16 or more bits, exceeding this recommendation. Typical analysis of the EEG produced, however, is performed by a human using 16 EEG traces displayed on a screen with 1024 vertical pixels. This gives just 6 bits of resolution [2]. For comparison, traditional paper-based EEG systems had a dynamic range of around 7 bits [2]. Potential room for optimization is thus present, especially if only automated analyses of the EEG are to be performed.

Of course, such uncertainties in the performance requirements are not confined to the biological world alone. For low-power, low-dynamic-range signal processing, analog circuit implementations can potentially significantly outperform their digital counterparts [3]. However, it is then necessary to contend with an amount of *mismatch*: for example, when implemented on a microchip capacitor values may be no more than 20% accurate, and no two transistors will be exactly the same. This leads to a variance in the performance of the analog circuit, and the range of this must be quantified to ensure that such a variance is acceptable. Accurate quantification of both the biological and electronic worlds is thus essential for optimizing the electronic design for device miniaturization.

One further method used to enable device miniaturization is online signal processing, as shown in Figure 1.1. As an example, if the EEG is being monitored, rather than transmitting the entire EEG recording it is possible to detect potential *interesting* sections of data online, and transmit only these sections. This significantly reduces the amount of EEG data to be sent, mitigating the use of high-power transmitters. However, accurate quantification is again necessary. It is essential that the accuracy of this data reduction method is known and acceptable. How many of the interesting sections are missed and how many false detections are made?

Unfortunately, in general bioelectronics applications, the quantification of online signal processing aiming to reduce the system power consumption, and hence the system size, is a problem subject to many constraints that make the algorithm design, implementation, and performance testing far from trivial. On the one hand, the algorithm must achieve acceptable performance accuracy for the given application. On the other hand, this must be done while developing an algorithm that can be implemented in very low-power circuits. There is no benefit in designing an algorithm that, when implemented, requires more power to operate than any potential power savings it provides. In addition to this, the potential for nonidealities in the end implementation, for example, from analog mismatch, must be accounted for.

The procedure for tackling this kind of problem is thus not to optimize any one aspect of it in isolation, but rather to look for a global solution that meets the constraints imposed by both the engineering design (such as the power consumption) and the biological application (such as a clinically acceptable detection accuracy) simultaneously. The overall interface design problem is thus an interdisciplinary one in which the bioelectronics designers must know the aspects of the biology that are going to condition the specifications of the electronic blocks; identify the best metrics to quantify performance for the given application; and devise a rigorous and representative test methodology that characterizes the performance within a certain confidence level.

Accurate characterization of the online signal processing algorithm is an essential part of this. For optimal power performance, these algorithms are best implemented as dedicated circuits, as opposed to in software. The circuit design, however, likely requires man-years of effort. For this not to be wasted on unpromising algorithms, accurate and reliable performance characterization is necessary at the algorithm design stage.

The aim of this chapter is to present the reader with examples of how to design a rigorous test methodology to characterize the performance of online signal processing systems designed to reduce the system-level power consumption. For example, what test factors need to be known and what performance metrics are best used in order to elucidate the most information possible about the performance? What is the impact of using different performance metrics, and how is it possible to ensure that the results are accurate and reliable? To illustrate the results on an actual algorithm, an EEG data reduction algorithm is considered. Nevertheless, although this algorithm is application specific, the methods and characterization routine are similar across many bioelectronics situations.

Section 1.2 thus presents the first part of the problem: the biological application and algorithm aim, in this case data reduction during monitoring of electrical brain activity (EEG). This motivates the need for an online signal processing stage and sets its engineering requirements. Section 1.3 then considers the factors that make representative performance testing nontrivial for this kind of application. Section 1.4 derives different performance metrics that could be used to characterize performance, discussing the advantages and disadvantages of each. Finally, Section 1.5 discusses the statistical testing of the results.

1.2 THE SIGNAL PROCESSING AIM

1.2.1 Introduction

The first step in the design of a rigorous test methodology for an online signal processing algorithm is the precise definition of the algorithm objective. This sets the required specifications for the algorithm, and also sets the objectives of the required test methodology. This Section quantifies the use of online data reduction in bioelectronics interface systems to decrease the system power consumption, and in turn the device size. The analysis allows the required level of data reduction to be found, motivating the algorithm design. The form of the analysis here applies to any general bioelectronics application, although as an illustrative case, the specific numbers here are taken from EEG monitoring.

1.2.2 The Need for Online Data Reduction

EEG recording, where electrodes are placed on the scalp and detect the microvolt-sized signals that result outside the head due to the accumulated neuronal action within the brain [4], is a characteristic bioelectronics problem requiring long-term, miniaturized interface systems connecting the biological and electronic worlds. This is because, although long-term inpatient EEG recordings are ideal for applications such as epilepsy diagnosis [5], they are resource intensive and not universally available [6]. Instead, ambulatory EEG (AEEG) recordings are available, during which the patient has their EEG recorded on a portable unit while undertaking their normal daily life. Such recordings cost approximately 50% of their inpatient counterparts [6] and so are highly desirable, provided the AEEG recording unit is miniaturized.

In general, this miniaturization is limited by the size of the battery required. If the overall device is assumed to have a volume of $1\,cm^3$ (a common aim for long-term ubiquitous recording applications) and half of this space is reserved for a custom-made battery with an energy density of 200 Wh/L, 100 mWh of energy is stored. For operation over 30 days, the average power consumption must be less than 140 μW [7].

An input amplifier and ADC system with a measured 25 μW power consumption per EEG input channel is presented in Yazicioglu et al. [8], representing the current state-of-the-art performance. Assuming 200 Hz and 12-bit sampling, 300 bytes per second per EEG input channel of data are produced. For a good transmitter that consumes 50 nJ/bit transmitted, including all of the overheads of data buffering, channel selection, and the like, transmitting each channel consumes approximately 120 μW. With these figures, only EEG systems with one input channel are feasible. To overcome this, online data reduction must be used.

1.2.3 Optimizing the Power–Device Size Trade-Off

The aim of the online signal processing indicated in Figure 1.1 should thus be to reduce the amount of data that is passed through the transmitter stage. From the

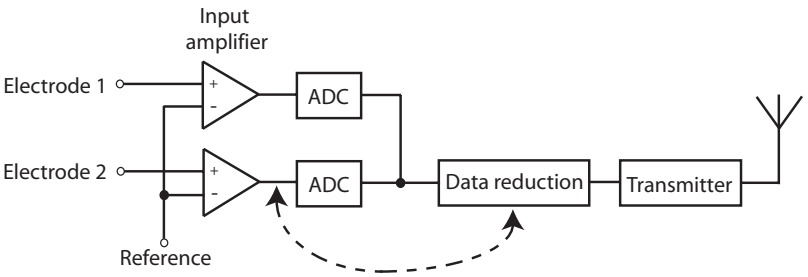


Figure 1.2. A simplified model of a two-input channel wireless interface system based on Yates and Rodriguez-Villegas [9]. The data reduction block can be in either the analog or digital domain.

example above, this transmission consumed 120 μW per channel, compared with only 25 μW for the front-end systems. Transmission thus dominates the system power consumption, and reducing the amount of data to transmit can significantly reduce the overall system power consumption even though the online data reduction will itself require some power. This trade-off is considered here using the framework from Yates and Rodriguez-Villegas [9].

To begin, consider the simplified interface system model in Figure 1.2. This basic architecture contains an input amplifier, an ADC, a data reduction block, and a transmitter. In principle, this architecture could be used to record many different physiological parameters; the form of analysis here is not unique to EEG acquisition.

As a first approximation, the power consumption of the entire system is given by

$$P_{\text{sys}} = \text{NP}_{\text{amp}} + \text{NP}_{\text{ADC}} + P_{\text{c}} + CP_{\text{t}}, \tag{1.1}$$

where n is the number of input channels, C is the percentage of data transmitted giving the ratio of the number of bits that are actually transmitted to the total number of bits if no compression was present, P_{t} is the power consumption of the transmitter, and the other three terms are the power consumptions of the amplifier, ADC and compression, respectively.

If the transmitter has a net power consumption, including overheads, of J joules per bit, P_{t} is given by

$$P_{\text{t}} = Jf_{\text{s}}RN, \tag{1.2}$$

where f_{s} is the sampling frequency and R is the resolution in bits of the ADC.

If the system is operated with no compression stage present, $P_{\text{c}} = 0$ and $C = 1$. Thus, if the inequality

$$P_{\text{c}} < JNf_{\text{s}}R(1-C) \tag{1.3}$$

is satisfied, the online data reduction can be used to decrease the total system power consumption.

Simultaneously, of course, the total system power consumption P_{sys} is governed by the size and capacity of the battery used. For a cell of volume V and energy density D operating over a device lifetime between battery changes T,

$$P_{\text{sys}} = \frac{V \times D}{T}. \tag{1.4}$$

For independence from any particular battery technology, the normalized operational lifetime can be defined as

$$T_{\text{n}} = \frac{T}{V \times D}, \tag{1.5}$$

and there is thus a direct three-way trade-off between the amount of data reduction achieved, the power budget available to implement this data reduction, and the normalized operational lifetime that is then possible. This trade-off is illustrated in Figure 1.3 using the EEG system figures from Section 1.2.2 and $n = 2$ for the two-channel system as illustrated in Figure 1.2.

A more comprehensive discussion of this idea can be found in Yates and Rodriguez-Villegas [9], but in brief, Figure 1.3 can be used to quantify the required engineering objectives of the online signal processing algorithm. For example, if from any given algorithm a data reduction of 25% is achieved, using the

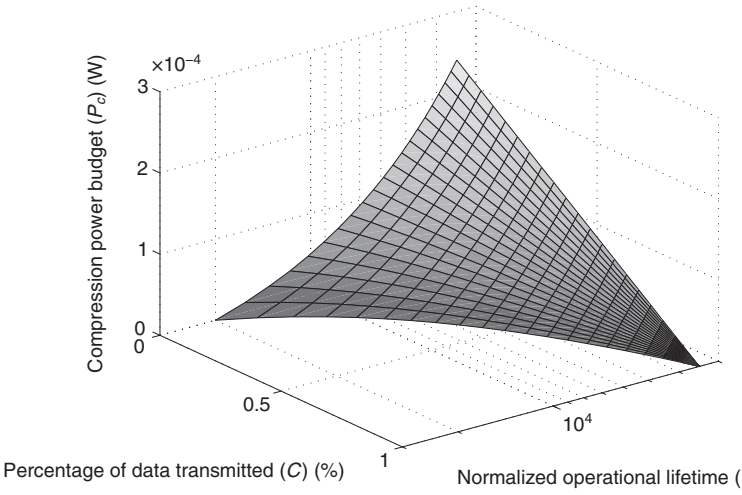


Figure 1.3. The three-way trade-off between the amount of data reduction, the normalized operational lifetime, and the available power budget to implement the compression algorithm.

hypothetical battery from Section 1.2.2, which stores 100 mWh of energy, for operation over 30 days, the online data reduction algorithm must operate using no more than 30 μW of power. Such power consumptions are challenging, but achievable. To avoid wasted effort on the required electronic design, however, the importance of fully verifying that the algorithm operates satisfactorily before attempting the design is clear.

1.3 REPRESENTATIVE TESTING

1.3.1 Introduction

The algorithm aim has now been defined, and it is assumed that a suitable algorithm has been developed. The next step is the design of the test methodology. The performance of any bioelectronics online signal processing algorithm is most suitably assessed prior to hardware implementation by carrying out a range of simulations using a software model of the algorithm. A set of input biological data is passed through the algorithm, and its operation observed. In general, the performance would then be compared with that of a human expert, and performance metrics comparing the two derived. Before considering the necessary performance metrics in detail, it must be ensured, however, that the simulation test methodology is rigorous, representative, repeatable, and accurate. Unfortunately, in real bioelectronics applications, there are a large number of factors that complicate the situation, and need to be identified and controlled if possible. These facts are discussed here for the special case of EEG recording to illustrate the typical factors that must be accounted for.

1.3.2 Data Recording Factors

For the software verification of the algorithm, an amount of biological test data must first be recorded from a subject in order to then be passed through the algorithm. However, one section of data is not necessarily representative of another data section, and is not necessarily representative of the type of data produced in the targeted application population. The data collection recording settings must thus be tightly controlled as they can affect the data traces produced, and hence the algorithm performance. Controlling these factors will also allow the situation under which the algorithm is characterized to be clearly stated.

For example, in the case of EEG collection, different EEG equipment can have different sampling rates, bandwidths, and electrode types, all of which should be specified. Also, the EEG typically records from multiple electrodes placed on the scalp and different *montages* are possible depending on how the channels are interconnected, and these affect the shape of the signals produced. The type of recording must also be controlled. Routine (20–30 minutes), long-term (1–3 days), or AEEG recordings are possible and may use different equipment, settings, channels, and montages. In the case of clinical applications, inpatient and outpatient tests may have very different artifacts present in the EEG recording

and such artefacts can be affected by the testing procedure used, for example, whether the eyes are open or closed. In addition, different subject states during testing, such as being awake or drowsy, can affect the EEG traces.

Overall, it is essential that the algorithm is tested using subjects who reflect the anticipated end user population. Examples of this include whether the subject is on medication, and the potential presence of multiple diseases (comorbidity). Factors such as the age of the subject can also be significant in determining the signal conditions, such as the amplitude and the amount of activity. For clinical uses, factors such as the sex, handedness, and many others are also be taken into account and recorded.

1.3.3 The Amount of Data to Test

Once the setup of the data recording has been controlled, a suitable amount of test data must then be collected. For example, to test an EEG spike detection algorithm, Wilson and Emerson [10] recommend that for comprehensive testing 100 subjects, 10,000 spikes and 800 hours of EEG should be used. This seems to be a very high level, especially given the effort required by an expert to mark the events in an EEG trace (see Section 1.3.4). The level, however, is perhaps correct if all of the above recording factors are to be made insignificant purely by the amount of testing done. To our knowledge at this date, only Persyst [11], which uses 18,503 events in 266 hours of data, and Liu et al. [12], who use 145,230 events from 81 patients in 800 hours of data, test anywhere near this amount.

When determining the amount of data to use, note should be made that it is often easy to get good algorithm performance when testing very short (where a human interpreter is essentially perfect) or artifact-free data. Generally, however, neither of these reflect situations where online signal processing algorithms would actually be of use. Long-term data, which is not preselected for the inclusion or exclusion of artifacts, should be used. Of course, to get long-duration recordings, it is often necessary to include multiple recording periods from probably multiple subjects. Due to variations between subjects, testing in multiple different subjects is potentially much more comprehensive than testing the same amount of data in just one subject.

Unfortunately, in most testing situations, it is unlikely that such large data sets are available for use. Even if they are, unless the algorithm being developed operates substantially quicker than real time, it may be impractical to experiment with very many different algorithm setups. To overcome this, it is possible to mathematically gain an insight into how much data should be tested through the idea of confidence intervals. These are considered in Section 1.5.3, once the necessary performance metrics have been defined.

1.3.4 Marker Reliability

Having collected the test data, an expert marker would generally then be used to identify the features of interest. Again, considering the example of an EEG spike detection algorithm, the time locations of the spikes are marked. This then

sets the baseline against which the algorithm will be compared. This process can represent a major limitation of the testing procedure as different markers do not always mark the same events, and one marker will sometimes mark differently when reviewing a record for the second time [13]. Marker agreement can be anywhere between 0 and 90%. Any results produced can thus be at most as accurate as this marking procedure. Also, the time markings themselves may only be accurate to the nearest second, minute, or hour, depending on the timescale of the signal being analyzed. This fact should be taken into account when determining how close an expert-marked event and an algorithm-detected event need to be for the detection to be successful.

Ideally, more than one expert marker should be used, and the method for combining the markings of each expert should also be clearly stated. Unfortunately, of course, this significantly increases the amount of time and resources required to prepare the algorithm testing.

1.3.5 Practicability and Ethics

Having collected and marked the biological data accounting for the above issues, suitable testing simulations can now be carried out. In practice, however, it is unlikely that all of the factors identified here will be known, or controllable. This is especially the case when the data is historical, rather than especially collected for the current study, or when the algorithm is developed by engineers, but the data are collected by clinicians, potentially working quite separately. This does not mean that it is impossible to perform good and informative studies without ideal test data. However, the potential limitations of such studies should be appreciated.

It may be ideal if algorithms were tested on a standardized database, and this would allow much more direct comparison between studies. Often, however, ethical approval is not in place to allow individual researchers to share their databases, and this is understandable. Nevertheless, standardized online databases are becoming ever more available.

The very large number of factors, and the potential difficulties in controlling them, can thus be seen. This makes it clear why the generation of accurate performance metrics for a given situation is a nontrivial task and deserves significant attention. Ideally, all of the factors above should be controlled and reported in any algorithm-testing publications. Again, however, there are some ethical considerations that may prevent this from being done. When working with a high-incidence disorder such as epilepsy, reporting that subject 1 has epilepsy and was 29 at the time of the EEG recording (which could have been several years before the first publication based on that test) does not devolve any significant information about subject 1.

It may be, however, that when working with rare diseases, or specific subsets of more prevalent ones, there may only be tens or hundreds of sufferers worldwide. In this case, reporting that subject 2 is male, 29 at the time of test, and is left-handed could provide significant clues to the subject's identity, which could

be compromised by a dedicated and resourceful person. Thus, although such information may be relevant, it should not be reported.

1.4 PERFORMANCE METRICS

1.4.1 Introduction

With suitable test data collected, having controlled for factors impacting the test methodology, it is thus now possible to consider which performance metrics should be used, and how these affect the algorithm performance that is reported. Again, a specific EEG data reduction algorithm is considered here as an illustrative case for the similar analysis that should be carried out for each bioelectronics algorithm prior to testing.

1.4.2 Illustration Data Reduction Algorithm

The data reduction strategy investigated here is concisely illustrated in Figure 1.4. Rather than continuously recording the EEG signal, an attempt is made to detect the features of interest, in this case spikes that occur between seizures in epilepsy patients,[1] and to record only a window of data around these automated detections. It can be seen how this can lead to a significant data reduction, even with a number of false detections present [14]. The algorithm aim should be to record all possible events for later interpretation by a human, while still cutting out some background data. Having a relatively high number of false detections does not necessarily compromise this process.

The core performance quantification problem is thus essentially one of signal detection: How many of the spike events are correctly recorded and how many false detections are made? The better the detection performance, the fewer false detections, and therefore more data reduction should be achieved. Reviews of similar EEG spike detection algorithms, although not necessarily for low-power implementation in bioelectronics interface systems, have been given by Frost [15] and Gotman [16] in 1985, and more recently by Wilson and Emerson [10] in 2002. Despite the level of interest illustrated by these, however, a definitive spike detection solution has not been found. It is clear that the task of finding a clinically acceptable trade-off between the number of events correctly detected and the number of false detections is nontrivial, highlighting the importance of accurate performance metrics.

The particular algorithm considered here is a developed version of the one proposed in Casson et al. [17] and is shown at a high level in Figure 1.5. The core of the processing is the extraction of frequency content in two bands by wavelet-based, band-pass filtering (see Casson et al. [18] for details on the filters). The C_5

[1] Here all between-seizure (interictal) events such as spikes, sharp waves, and spike-and-waves are considered under the umbrella term spikes.

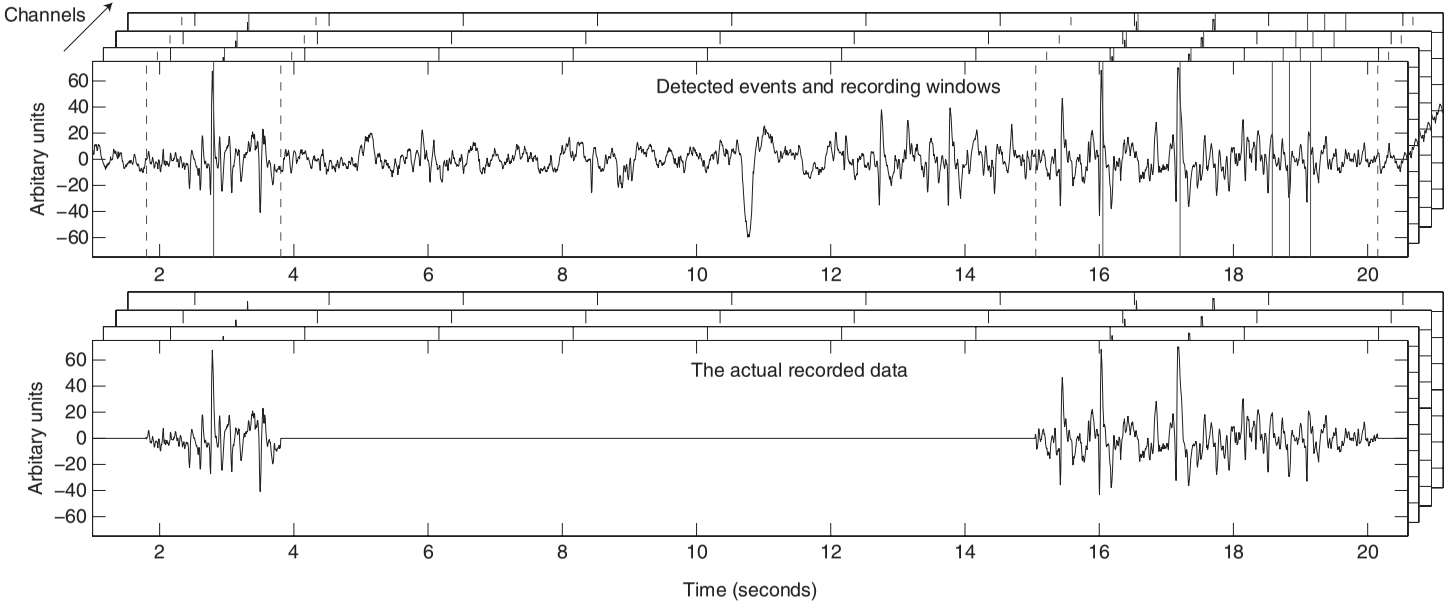


Figure 1.4. A data reduction strategy based on discontinuous recording. Only EEG data for a brief period (dashed vertical lines) on either side of an automated detection of a *candidate spike event* (solid vertical lines) is selected to be recorded. Other data sections are discarded online, significantly reducing the amount of data to be transmitted.

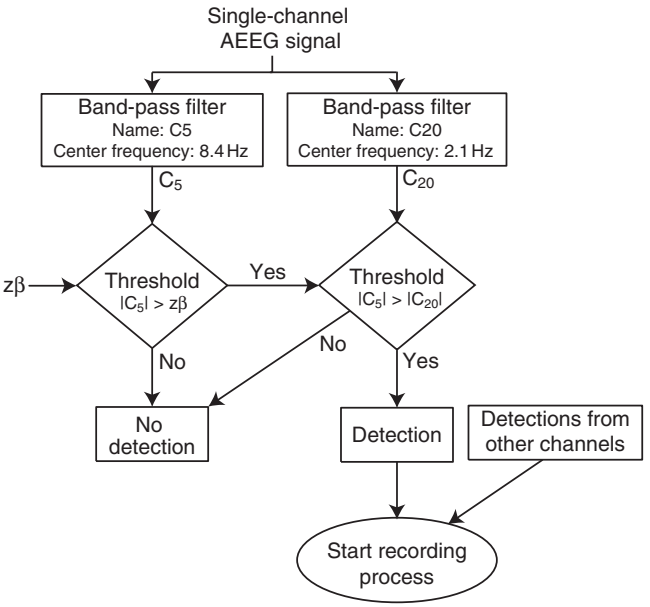


Figure 1.5. A high-level overview of the algorithm to be investigated. Detections in any monitored EEG input channel cause the algorithm to start recording. In the recording process (not shown), a memory buffer must be present to allow sections of EEG data immediately preceding a detection to also be recorded.

information produced is then compared with a threshold value of $z\beta$. z is an automatically generated normalizing parameter to correct for broad level amplitude differences in different EEG traces given approximately by the root mean square (RMS) of the EEG signal. β is a user-set detection threshold. The user is free to sweep β to obtain a range of performances from the algorithm. If all of the comparisons are satisfied, a detection is made, causing a section of EEG data from before (stored in a buffer) and after the detection point to be recorded.

1.4.3 Metrics

In principle, two metrics are required to illustrate the operation of any algorithm. One is the *performance* indicating how well the method operates, and the other is the *cost*, which indicates what undesirable factors are also present. Inevitably, there is some form of trade-off between the two. For event detection algorithms such as the EEG interictal spike case considered here, there are four common measures associated with characterizing the performance compared with that of an expert marker. These all use the following terminology:

- True positives (TP): the number of correct detections of a spike as a spike
- False positives (FP): the number of incorrect detections of a nonspike event as a spike

- True negatives (TN):the number of correct detections of a nonspike as a nonspike
- False negatives (FN): the number of incorrect detections of a spike as a nonspike

and the metrics are given in the following list:

1. *Sensitivity*: the fraction of spikes that are correctly detected:

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}} \times 100\% \tag{1.6}$$

2. *Specificity*: the fraction of nonspikes that are correctly rejected:

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \times 100\% \tag{1.7}$$

3. *Selectivity*: the fraction of correct detections:

$$\text{Selectivity} = \frac{\text{TP}}{\text{TP} + \text{FP}} \times 100\% \tag{1.8}$$

4. *FP rate*: the average number of FP per minute or hour.

Not all papers generate any of these measures explicitly, but it is essentially universal to have some quantification of the *performance* and the *cost*, allowing such measures to be derived if wanted. For example, some articles, such as that by Indiradevi et al. [19], use an *accuracy* metric, although what is meant by this is not always defined, and it is unlikely to be used consistently. Similarly, for the algorithm considered here and, in general, for any algorithm intended to be mapped into hardware, where the aim is data reduction rather than correctly counting the number of spike events, the FP rate is not a suitable metric for use. Instead the percentage of data transmitted (C) introduced in Section 1.2.3, indicating what fraction of the full EEG recording is selected to be transmitted, is a more suitable *cost* metric. However, this is somewhat related to the false detection rate, as the more false detections occur, the larger the amount of data to be transmitted will be.

The sensitivity is the core *performance* metric, and it illustrates how many of the wanted features are correctly found—a high sensitivity is always wanted (although not all papers give a measure of this performance, classically Gotman and Gloor [20]). As noted previously, it is usually calculated by comparing the algorithm detections with those made by an expert marker. When doing this, it is important to specify the detection window, indicating how closely the two must match for a TP to have been found.

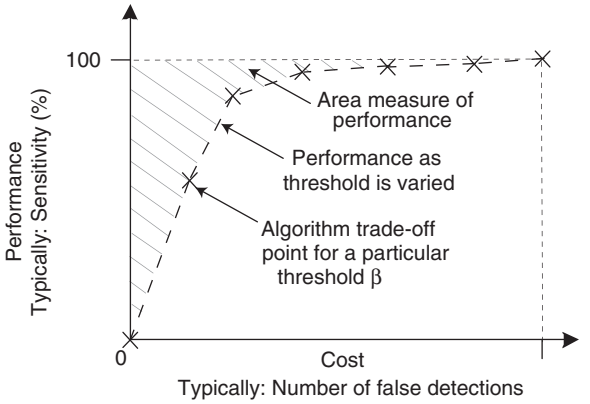


Figure 1.6. A schematic example illustrating how algorithm detection performance can be shown on a trade-off graph.

The specificity, selectivity, and FP rate provide measures of the *cost*. The specificity and selectivity should be high, or the FP rate low, for good results. In the context of spike detection, although the specificity is a well-defined concept, it is not clear what a true negative (TN) EEG event is. It is presumably a measure of how much background activity is present, but it is not easily reduced to an integer as the TP, FP, and FN measures are, and so the specificity is perhaps not an optimal measure to use, although again some articles, such as that by Exarchos et al. [21], attempt it. One method of calculating it is to use a time domain approach. Say for review by a human, 10 seconds is recorded in response to each false detection. A false detection rate of one per hour, which is quite poor if accurate counting of the number of events is wanted, results in a specificity of 99.7%. A specificity value over 99% thus does not necessarily correspond to a particularly good performance, and again other metrics such as the FP rate may be more illustrative.

Given these metrics, it is generally found that there is a trade-off between the sensitivity of an algorithm and the number of false detections that it makes. It is possible to achieve a high sensitivity (correctly detecting lots of events) if lots of false detections are tolerated. Most algorithms take this aim, although some, such as Ramabhadran et al. [22], aim for no false detections at the cost of a reduced sensitivity.

To display the results, as Wilson and Emerson [10] note, if an algorithm detection parameter, called say β, can be easily varied, the most natural way to display this trade-off is on a graph, as illustrated in Figure 1.6. For each value of β, a particular *performance* (sensitivity) is obtained at a certain *cost* (e.g., number of false detections). As β is varied, this pair of points defines a curve, which is essentially a receiver–operator curve (ROC) and the area to the top left of the line can be used as a high-level, dimensionless measure of the algorithm performance.

1.4.4 Illustrating the Average Performance

When using large numbers of subjects or tests to characterize the algorithm performance, it is not feasible to present the above metrics for each individual test, and it is desirable to have an overall headline performance figure. Methods for displaying the average performance of the algorithm over different tests are thus required. Different methods for calculating the average are possible, for example, by weighting the sensitivity values found to correct for nonideal test cases, and these are investigated below.

One example of a nonideal test case is that some data records may contain hundreds of expert-marked events, while others will contain only one. Thus, in different records, the detection or nondetection of one event can have a very different effect on the sensitivity found for the record, and this affects the average sensitivity found. A quantitative treatment showing the effect of the averaging method is very insightful for showing how results can be inadvertently weighted.

Here, four different methods of calculating the average sensitivity between different records are investigated, although other methods are undoubtedly possible, and will likely provide different properties to the methods considered here. The calculation procedures are detailed below using the terminology that there are M records and the ith record has a duration T_i, with N_i marked events and D_i correctly detected events. The sensitivity for any one record is given by $100\% \times D_i/N_i$. Sensitivities for different records can then be combined in the following ways:

1. Arithmetic mean:

$$\text{Sensitivity} = \frac{1}{M}\sum_{i=1}^{M}\frac{D_i}{N_i}\times 100\% \tag{1.9}$$

2. Time-weighted average:

$$\text{Sensitivity} = \frac{1}{\sum_{i=1}^{M} T_i}\sum_{i=1}^{M}\frac{D_i}{N_i}T_i \times 100\% \tag{1.10}$$

3. Total sensitivity average:

$$\text{Sensitivity} = \frac{1}{\sum_{i=1}^{M} N_i}\sum_{i=1}^{M} D_i \times 100\% \tag{1.11}$$

4. Time/event-weighted average:

$$\text{Sensitivity} = \frac{1}{\sum_{i=1}^{M} T_i / N_i}\sum_{i=1}^{M}\frac{D_i}{N_i}\frac{T_i}{N_i}\times 100\% \tag{1.12}$$

TABLE 1.1. A Synthetic Algorithm Results Set with Tests of Variable Lengths and Differing Numbers of Events and Correct Detections

Data Set (i)	Duration (T_i) (minutes)	Events (N_i)	Detections (D_i)	Duration/ Events (T_i/N_i)	Record Sensitivity (D_i/N_i) (%)
1	20	2	1	10	50.0
2	20	400	385	0.05	96.3
3	30	6	6	5	100
4	30	25	15	1.2	60.0
5	60	28	28	2.1	100
6	60	500	463	0.12	92.6
7	60	5	4	12	80.0
8	1440	40	19	36	47.5
9	1440	16	15	90	93.8
10*a*	60	3	1	20	33.3
10*b*	60	3	2	20	66.7

TABLE 1.2. Results of the Different Calculation Methods on Data Sets ***a*** and ***b***

	Sensitivity (%)	
Averaging method	1–9 and 10*a*	1–9 and 10*b*
Arithmetic mean	75.3	78.7
Time weighted	71.3	71.9
Total sensitivity	91.4	91.5
Time/event weighted	74.1	77.9

The arithmetic mean method treats all of the different test cases equally, and so any one record that has, say a low sensitivity, can significantly affect the overall value found. As a result, it is potentially weighted by records that are very short or contain very few events. In contrast, the total sensitivity measure treats all of the records as if they were one long record concatenated together. It is, thus, potentially weighted by records with large numbers of spikes or ones where the detection rate is particularly good. The time-weighted average and time/event-weighted average weight the individual sensitivities to make longer records more significant and long records with few events significant, respectively, in an attempt to overcome the limitations noted above.

The effect of the different averaging methods is illustrated, using purely numerical arguments, in Tables 1.1 and 1.2. Table 1.1 illustrates a typical set of 10 data records available for algorithm testing, with variable test lengths and numbers of events present in each recording. In addition, an assumed example level of algorithm performance is shown, with the number of correct detections made being arbitrarily chosen to illustrate a range of performance cases. For each data set, given the number of detections and the number of events, the sensitivity within each record is calculated and shown.

For record 10, however, two different cases, *a* and *b*, are considered. In case *a*, only one of the three events in the recording is correctly detected, while in case *b*, two events are correctly detected. The effect of this slight change is illustrated in Table 1.2. In this table, the sensitivity is calculated using $M = 10$, the four different sensitivity methods discussed previously, and the figures for N_i and D_i from Table 1.1. In case *a* (the middle column in Table 1.2), data sets 1–9 and 10*a* are analyzed, while in case *b* (the right-hand column), data sets 1–9 and 10*b* are analyzed.

It can be seen how the detection or nondetection of just one event from over 1000 events in 32 hours of data appreciably affects the averages found. Furthermore, the different averages change by noticeably different amounts. For example, the total sensitivity hardly changes (one event in 1033 is a very small percentage), while the arithmetic mean changes noticeably (one event in the three in record 10 is a large percentage). It is also interesting to note the spread of average values that are present in Table 1.2: more than 20% in case *a*. When devising an algorithm testing methodology, it is essential to take these potential weightings into account.

To illustrate how the ROC-like curves proposed in Section 1.4.3 are affected, an example testing situation using the algorithm from Section 1.4.2 is considered. For testing, three different data sets are analyzed: *A*, *B*, and *C*, detailed in Table 1.3. Data set *A* contains nine tests ($M = 9$), each of which is approximately an hour or more in length, with a total duration of 16 hours. There are 120 expert-marked events present. Data set *B* is the same as *A*, but with an extra 37-minute data set, making little difference to the total time analyzed ($M = 10$). This record, however, contains 644 events. Data set *C* is again the same as *A*, but with an extra 5-minute record, which contains seven events ($M = 10$). The algorithm is run on these three data sets, and the results produced using the four different metrics are shown in Figure 1.7. Results for both axes are weighted according to the averaging method used.

For the analysis, 19 values for the threshold β between 1 and 0.32 are used to produce the trade-off curve, and 5 seconds of EEG are recorded in response to each detection. (A typical spike has a duration of 140 msec [23].) For a detection to be considered correct, it must occur no more than 2 seconds away from the expert-marked position. In addition to the calculated result points, the *known* end points that if no data is sent the sensitivity must be 0% and, similarly, that if all of the data is sent the sensitivity must be 100% are also included. Finally, the *chance performance* line is also drawn. If, as a first approximation, spike events are assumed to occur at random times, randomly transmitting 10% of the raw EEG trace should result in a 10% sensitivity. The $y = x$ line thus corresponds to the performance of a chance detection scheme, and any algorithm performance should always be above this.

Considering Figure 1.7, data set *A* illustrates the baseline level of performance. Although the performance of the algorithm will always depend on the data analyzed to some extent, in principle the sensitivity is a property of the algorithm and should not be weighted by less representative test cases. This is

TABLE 1.3. Data Available for Analysis with the Algorithm Detailed in Section 1.4.2

Record (i)	Data Set	Duration (T_i) (HH:MM:SS)	Events (N_i)
1	A, B, and C	00:59:08	4
2	A, B, and C	00:58:56	4
3	A, B, and C	02:00:11	41
4	A, B, and C	02:00:11	7
5	A, B, and C	02:00:11	3
6	A, B and C	02:00:11	21
7	A, B and C	02:00:11	28
8	A, B and C	02:00:11	9
9	A, B and C	02:00:11	3
10	B	00:36:55	644
11	C	00:05:00	7

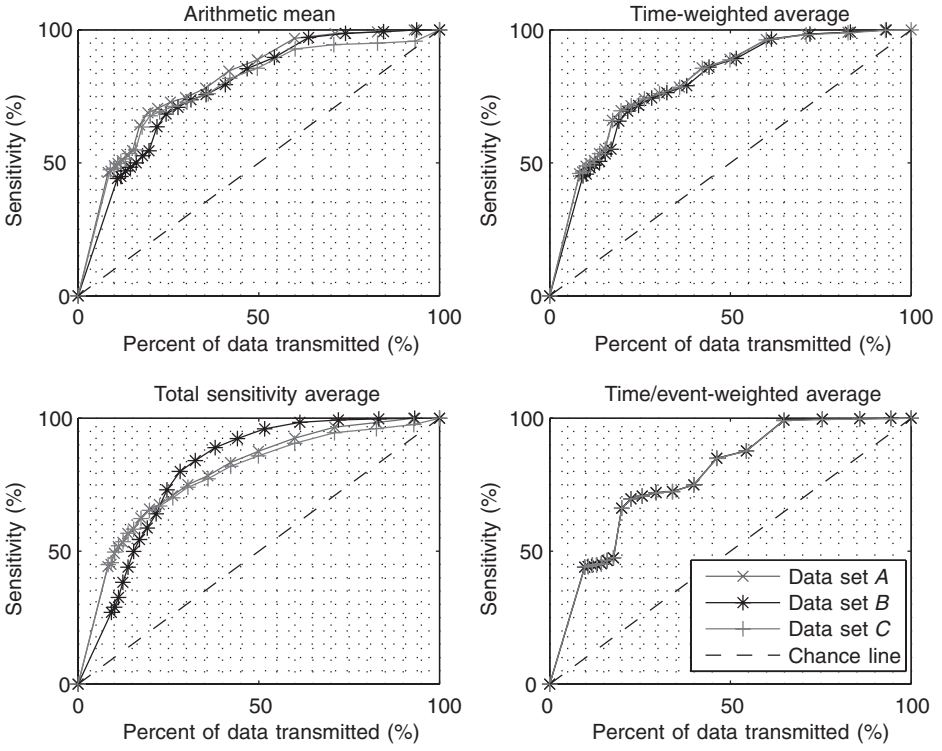


Figure 1.7. Performance results for the data reduction algorithm when different data sets are analyzed using the four different averaging methods.

clearly not the case, however, for the arithmetic mean or total sensitivity averaging methods where the results are visually different depending on which data is analyzed. The time-weighted average and time/event-weighted average have less sensitivity to the data analyzed, although less skew is present in the time/event averaging case, particularly when the high event count record is analyzed (data set *B*).

Given these results, it is thus essential to appreciate how the testing procedure can potentially be weighted. Assuming that the same test setup is used in all cases (see Section 1.3), the two core parameters that can vary between tests from a signal processing point of view are the test duration and the number of events present. The test duration can be controlled to a certain extent, although longer tests require more resources to carry out. The number of events present cannot be controlled. (Although only a set number can be considered if all records contain this minimum number.)

Thus, it does not seem unreasonable to normalize for both these parameters, giving the time/event-weighting method. At the same time, however, weighting the results can distort the sensitivity figure that is seen. For example, for the data set *B* case from Figure 1.7, at one threshold level 611 of the 764 events are correctly detected, giving a total sensitivity of 80%, but the time/event-weighted sensitivity is only 72%. In some cases, the time/event-weighted figure thus does not well represent the number of correct detections made. It is thus likely that reporting both the total sensitivity and time/event-weighted sensitivity is advisable when attempting to draw as much information from the algorithm performance as possible.

Finally, when interpreting the average performance lines, consideration must be given to the fact that although an algorithm may achieve a certain average performance level, there is not necessarily an equal probability of different events being detected. Wilson et al. [13] puts it as

> a particular spike has a probability of detection, which may be high or low depending on its size, morphology, background activity and other attributes.

As a result, the average performance is not necessarily easily related back to the performance of any one test: It can be used only as a general guideline of overall performance.

1.4.5 Illustrating the Variance in Performance

Methods for presenting the algorithm's average performance have been discussed in detail above. It is equally important, however, to illustrate how the performance results obtained vary from test to test, and person to person. Given the differences between people, it is not necessarily unexpected if an algorithm performs very well on one person, but only moderately well on another person. This would indicate the some subject-dependent tuning is required. It may even be found that different recordings in the same person produce noticeably differ-

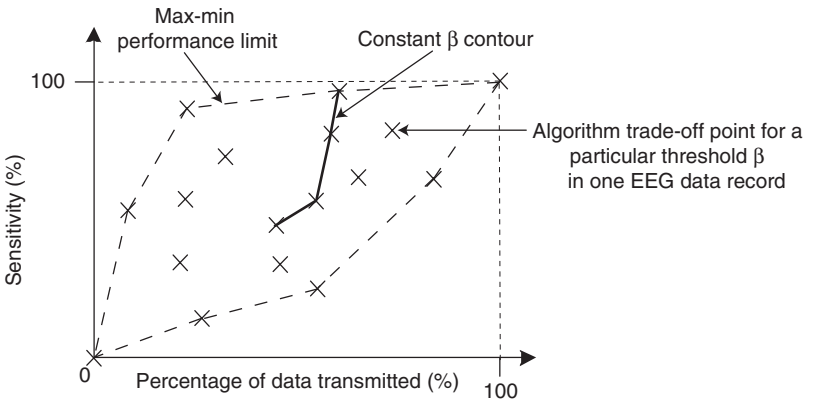


Figure 1.8. An example method for illustrating the performance variance by showing the max-min performance limit and the performance at the same threshold β in each record.

ent performances, for example, due to different equipment setups or awareness states during the data recording. Some variance in the performance is thus entirely reasonable, but for a successful algorithm it must be at an acceptable level, and so needs quantifying.

Of course, for any arbitrary algorithm, it is unlikely that the variance follows a normal distribution about the average. Simply plotting the mean result and the standard deviation is thus not a viable option. Instead, one potential method is illustrated in Figure 1.8 and discussed here, although as with the averaging methods considered above, other methods are undoubtedly possible.

First, on the same fundamental ROC-like curve, the *individual results* are shown. If the data set *i* is being analyzed and contains one or more marked events, each time the algorithm is run at a particular threshold β, individual values for the sensitivity ($S_i = 100\% \times D_i/N_i$) and percentage of data transmitted (C_i) are found. This pair of values (C_i, S_i) can then be plotted, giving a large number of data points if multiple tests and thresholds are used.

Then, the convex hull joining the outer most individual results can be drawn. This thus illustrates the max-min performance limit of the algorithm. Given the test data used, the algorithm performance can always be expected to lie in this region, and ideally, this region should also lie above the chance performance line. The generation of this line though does assume that only *sensible* values for the threshold β are used. Otherwise, the region can be artificially expanded by using thresholds that would never be considered in practice.

Finally, a constant threshold line is drawn. This is a line connecting all of the individual results that are calculated at the same β value. This reflects the system performance when used in practice where β would be set in advance and the performance will vary along this line. In the graphs here, this is drawn from the highest sensitivity point to the lowest sensitivity point, and so cannot *double back* in this direction.

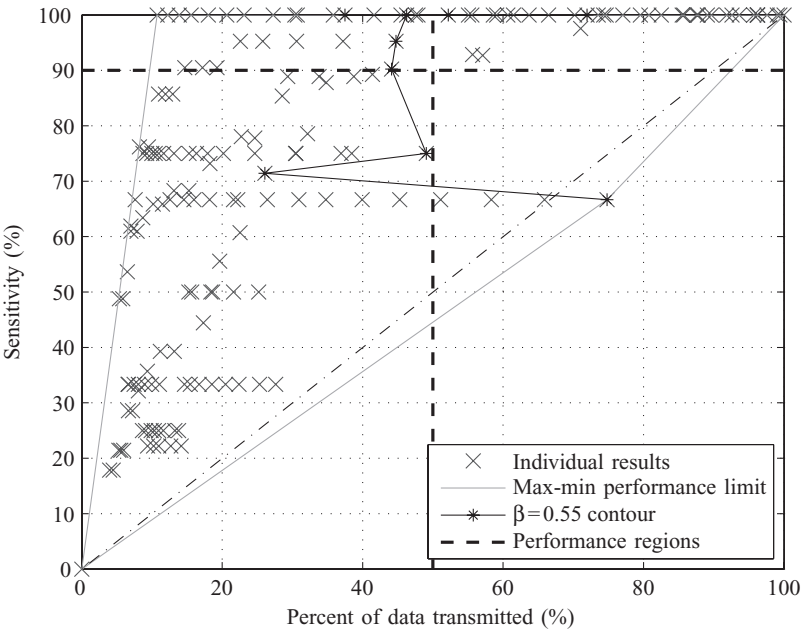


Figure 1.9. Results showing the variance in performance for the algorithm from Section 1.4.2 using data set *A* and a 5-second recording window.

Figure 1.9 thus shows such a variance graph using the data set *A* detailed in Table 1.3. The value of β for the contour line is arbitrarily chosen to show a particular operating point. In addition to the lines identified above, the graph has also been split into four regions. These come from the assumed specifications that 90% sensitivity with more than 50% data reduction lead to acceptable performance. These are arbitrarily selected in this case, and in reality, must be carefully chosen based on the particular application requirements.

It can thus be seen that for four tests (44%), the sensitivity is over 90% and less than 50% of the raw data is transmitted. The algorithm is thus operating well for these cases. For two cases (22%), high sensitivities are achieved, but a lot of data has to be transmitted. This could correspond either to tests that have a very large number of events relative to their length or simply to poor algorithm performance. It is straightforward to investigate how much data would be sent by an ideal algorithm for these tests to differentiate between these two possibilities.

For the other three cases present (33%), the sensitivity is too low and it is likely that further algorithm development is necessary. For the two cases where the amount of data transmitted is below 50%, the performance is not necessarily disastrous. From the system-level point of view (Section 1.2), provided that the algorithm implementation consumes very little power, turning off the high-power transmitter stage can lead to significant overall power reductions. This will

increase the operational lifetime and it may be possible to *recover* some of the lost sensitivity by recording potentially more events in this longer lifetime.

The single point in the lower right-hand quadrant, however, clearly represents poor algorithm performance: the performance is below that of the chance line. Events that should be recorded are missed, and still large amounts of data are selected for transmitting. Whether such a situation is tolerable at all, and for what fraction of cases if it is, is an open question.

The presented method thus clearly demonstrates all aspects of the algorithm variance in one graph. It is still up to the user, however, to determine whether the level of variance is acceptable, and whether there is any pattern between the observed results. For example, subject 3 may always obtain a low sensitivity, indicating that some subject-specific modifications may be necessary, or the algorithm may perform much better in young subjects, but not old ones. Overall, it can be seen how the illustration of the algorithm variance is nontrivial, and leads to the generation of very complicated plots.

1.5 STATISTICAL VALIDATION

1.5.1 Introduction

Ideally, the results for the average and variance in the performance of an algorithm need to answer the following question: Is the algorithm performance statistically *good enough*? Unfortunately, this is not easily done. It depends on the overall aim of the algorithm and the performance necessary for acceptable operation. For example, in EEG spike detection, Gotman [16] suggests that it is unlikely that all spikes in the EEG record need to be correctly recorded and identified in order to allow accurate diagnosis by a human. The aim of diagnosis is to pool all of the available information to enable a decision based on the balance of probabilities to be made. The presence or absence of a small number of spikes should not be a critical factor in this decision process. Thus, sensitivities of 90% or 80% may be readily acceptable. The true performance can only be made in terms of the *aid to diagnosis* that miniaturized and long-lasting systems offer. Of course, however, this can only be measured once systems are in place, and it does not help in determining whether a particular algorithm is of use, and suitable for detailed investigation, optimization, and hardware implementation. In reality, such work may take several years, so we do not want to waste effort on unpromising algorithms.

1.5.2 Statistical Significance Testing

Instead, for most algorithms, it is likely that the closest possible test is showing that the average algorithm performance is statistically better than chance performance. This would illustrate that the algorithm does have some skill and acts as a basis point for further improvement. To do this for the ROC-like result curves such as the ones considered here, the Mann–Whitney *U*-test can be used. This is

a nonparametric test and is frequently used for testing the areas under ROC curves [24].

The test works as follows. For each ROC-like average performance curve, y-axis points corresponding to sensitivity values are used to form the set s, and x-axis points corresponding to percentage of data transmitted values are used to form the set c. The following null hypothesis is then considered [25]:

"*There is no tendency for members of set* s *to exceed members of set* c."

That is, if the sensitivity values tend to be the same as the data transmited values, the algorithm performance would be along the chance line ($y = x$). If they are statistically different, and the ROC curve is seen to lie above the chance performance line, the performance must be statistically better than chance. Note that the Mann–Whitney U-test assumes that samples in s and c are independent. For the specific situation illustrated here, this seems reasonable provided that spikes are rare, short events: With any level of transmission, it is possible to get any sensitivity; it is not necessary to send large amounts of data to get a large sensitivity value.

As an example, the statistics for the 12 result lines (four different averages with three data sets) in Figure 1.7 are calculated. To do this, the x and y values from each result curve are extracted to form sets s and c, with the known end points at 0 and 100% excluded. Each set thus has 19 entries ($n_1 = n_2 = 19$) corresponding to the 19 thresholds used in the algorithm. U-values are then calculated using the formula [25]

$$U = \sum_{i=1}^{n_1} r_i - \frac{n_1(n_1+1)}{2}, \tag{1.13}$$

where r_i is the *rank*. The rank is calculated by concatenating sets s and c and then sorting into numerical order. The rank is then the end positions of the members of set s. Note that this formulation assumes that each value in s and c are unique. If repetitions are present, a slightly modified procedure should be used [24, 25].

The U-value is calculated for each result line and is shown in Table 1.4. To perform the statistical test, each value is then compared with the critical U-value, tabulated in Bland [25]. For a $p = 0.05$ two-tailed test, $U_{\text{crit}} = 113$. All of the U-values in Table 1.4 are below the critical U-value, and so the null hypothesis is rejected ($p = 0.05$ two-tailed test, $n_1 = n_2 = 19$, $U_{\text{crit}} = 113$, $U_{\max} = 61$). It is thus concluded that the average performance of the algorithm is statistically superior to that of a chance classifier. Again, this does not provide any insight into whether the algorithm performance is acceptable, but it is undoubtedly a good result.

1.5.3 Confidence Intervals to Determine Test Sample Size

Once the performance metrics of interest for the algorithm have been defined and investigated, it is then possible to go back and use them to help guide the design of the algorithm testing methodology. This can be done, as suggested in

TABLE 1.4. *U*-Values from the Mann–Whitney *U*-Test Applied to the 12 Result Curves from Figure 1.7

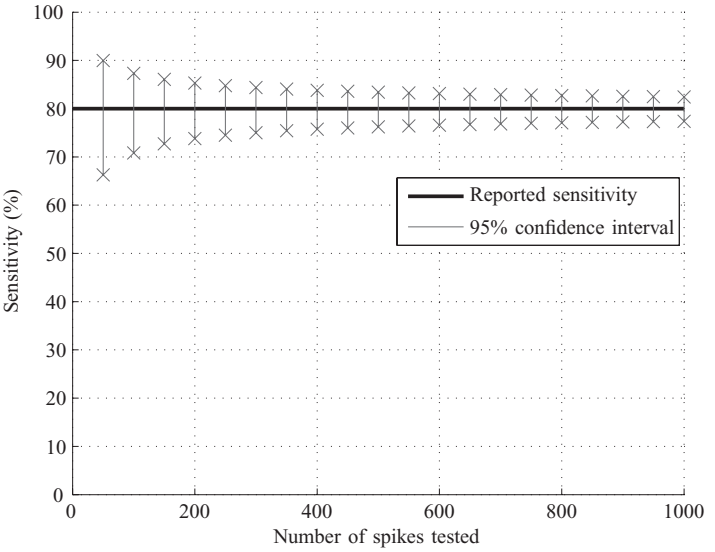
Data Set	Arithmetic Mean	Time-Weighted Average	Total Sensitivity	Time/Event-Weighted Average
A	48	48	50	61
B	55	50	59	61
C	49	48	51	61

Section 1.3, by using statistical information to guide the amount of data that should be tested. This procedure is based on statistical confidence intervals. Note that it has to be first assumed that the biological test data is representative, containing the features of interest in all of their likely morphologies from all of the potential test setups and user populations. Confidence intervals only illustrate how much of this representative data should be analyzed. Beyond this, the results below depend only on the performance metrics and amount of data used, not on the particular algorithm being investigated.

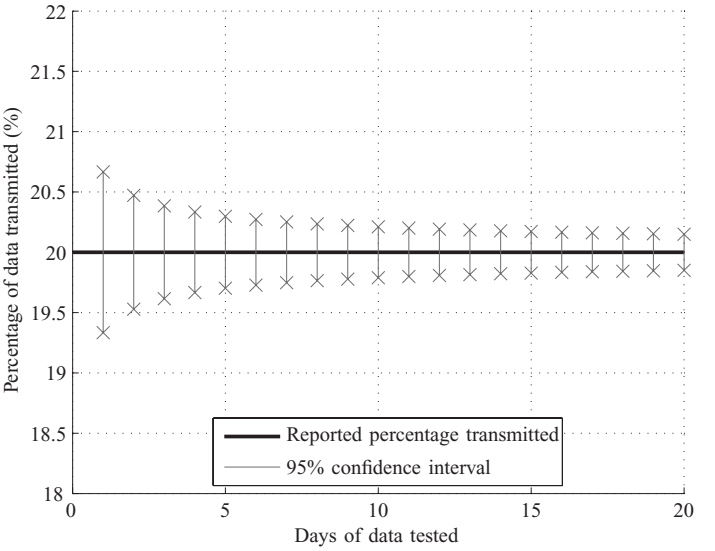
The method is based on the idea that at a particular operating point, the algorithm has a certain *true* performance. The algorithm testing attempts to find this by analyzing a set of data and obtaining an estimate, or *reported* performance. The aim, of course, is that the true and reported performances should match. Confidence intervals calculate the range of values that, given the reported result and amount of data tested, the true result could actually lie in and reasonably produce the reported result by chance alone. This is clarified in an illustration here. As in Sackellares et al. [26], confidence intervals for the *performance* (sensitivity) and *cost* (percentage of data transmitted) are calculated separately. In principle, these values could be plotted on the ROC-like result curves to show the uncertainties at each trade-off point, but it is likely that this will lead to overcomplicated graphs. Instead they are plotted in isolation in Figure 1.10.

The starting point for confidence interval generation is the assumption of a suitable probability distribution that describes the underlying performance metric. For the sensitivity metric, a binomial distribution, which quantifies the probability of detecting a certain fraction of events from a total number, is suitable [26]. If there are a total of n events and the reported sensitivity is S, a distribution $B(n, S/100)$ is used. The 95% two-tailed confidence intervals can then be either simply read from tables [27] or generated using the MATLAB binofit function.

The intervals are illustrated in Figure 1.10a for any algorithm with a reported sensitivity of 80% [26]. Given a reported sensitivity S, these intervals show the range in which the true value of S could reasonably occur. Note that the binomial distribution assumes that the detection of any one event is independent from the detection of any other. This seems reasonable when testing a large amount of data from multiple people. In general, without detailed modeling of the probability distribution of the algorithm detections, it is not possible to avoid such assumptions entirely. Instead, however, rough values can be generated, which, while approximate, are very informative.



(a) Sensitivity intervals for 80% performance



(b) Percentage of data reduction intervals for 20% performance

Figure 1.10. Estimated 95% two-tailed confidence intervals showing the range that, given the observed result, the true value could reasonably lie within. There is a 95% chance that the true sensitivity is within the interval shown. (a) Sensitivity intervals for 80% performance; (b) percentage of data reduction intervals for 20% performance. (a) Based on Reference 26.

The graph from Figure 1.10a can be replicated for different reported sensitivity values if so desired. The largest confidence intervals are found when the reported sensitivity is 50%. When testing 120 events as in data set *A*, it is found that the reported sensitivity may be overestimating the true sensitivity by up to 9.26%. Insufficient data is being analyzed to produce statistically confident results. For most sensitivity values, however, the confidence intervals will be smaller than this. Also, if 10,000 events, as recommended by Wilson and Emerson [10], are analyzed, the sensitivity should not be overestimated by more than 0.98% in the worst case.

If confidence intervals are wanted for the percentage of data transmitted, they can be calculated similarly. In this case, however, a binomial distribution is less suitable for use as there is no analog of the value *n*. Instead, in lack of a better probability distribution suitable for use, a procedure based on the false detection rate can be used to approximate the confidence intervals.

False detection rate confidence intervals are found from a Poisson distribution [26]. This distribution arises from a binomial distribution when the number of tests is very large, and the events (false detections) are rare and occur at a fixed average rate. It is thus equivalent to testing at each instantaneous time point to observe whether a false detection has occurred, and 95% two-tailed confidence intervals can thus be generated for an assumed false detection rate based on confidence interval tables [27]. For the EEG data reduction algorithm considered previously, assuming that false detections are rare, when recording the signal around one false detection, it will not *overlap* with the signal recorded from another false detection. There is thus a fixed ratio between the false detection rate and the amount of data transmitted, allowing the wanted confidence intervals to be calculated.

Results for a percentage of data transmitted value of 20% are shown in Figure 1.10b. For the approximately 16 hours of data in data set *A*, the confidence intervals indicate that the percentage of data transmitted should not be underestimated by more than 1%. For 800 hours, as suggested by Wilson and Emerson [10], the value should not be overestimated by more than 0.12%.

It is thus possible to use confidence interval figures to show that sufficient data is being tested in order to have reasonable confidence that the results produced are representative of the algorithm performance, and are unlikely to have occured by chance. Given the form of the intervals seen in Figure 1.10, large amounts of data must be tested in order to get significant improvements in the confidence intervals, but a guide to a reasonable amount of data to test can be made.

1.6 CONCLUSIONS

In order to optimize the interface between the biological world and the electronic world, it is essential that the performance of the interface system is accurately quantified. This will allow sensible design decisions to be taken so that effort is not wasted on the implementation of algorithmic methods that will not result in

satisfactory performance. Methodological and rigorous testing, however, is not a trivial task. It is often difficult to collect sufficient data for comprehensive testing of such algorithms, and when such data is available, it is often not completely controlled and all of the recording parameters known. It is still possible, and often necessary, to perform insightful and useful testing using such data, but the limitations should be taken into account.

Using a data reduction algorithm for EEG monitoring as an example, this chapter has shown how the algorithm results can potentially be weighted to compensate for nonideal test cases. When selecting a method to present the average performance of an algorithm, care must be taken to select the appropriate averaging scheme, given the overall application aims. Otherwise, it is possible to inadvertently weight the results produced, which could lead to incorrect conclusions and design decisions being made. A potential method for quantifying the variance in the performance has also been outlined.

Furthermore, given the limitations of algorithm testing, is it essential that appropriate statistical validation be used where possible. For ROC-like result cases, as illustrated here, the Mann–Whitney *U*-test can be used to determine whether the algorithm performance is statistically better than that of a chance classifier. Such algorithms can then act as a starting point for algorithm improvements, although they do not help answer the question of whether a particular performance is good enough. Confidence intervals can also be used to help determine how much data should be tested, and the benefit of spending large amounts of time in collecting more data for testing. The performance metrics considered here are inevitably application specific to some extent, but it is likely that similar metrics and weightings are suitable for many different applications.

Finally, for future low-power online signal processing algorithms implemented in the analog domain, the mismatch associated with analog circuits will lead to an amount of uncertainty about the exact performance of any one chip before it is explicitly tested. For statistical algorithm testing, there is also an amount of uncertainty as the performance will never be known in advance, and will depend on how representative the test data is. Thus, it may be possible to *couple* these uncertainties in order to get improved performance. For example, the mismatch of a filter may lead to slight differences in the performance of an algorithm. If this has a large effect on the performance, mismatch robust filter topologies must be used, and this may come at the cost of other factors, such as power consumption, linearity, and noise performance. Alternatively, if the filter mismatch makes no statistically significant change to the algorithm performance, alternative topologies can be sought, potentially easing the design constraints elsewhere. The investigation of such trade-offs is essential for future interface system research.

REFERENCES

[1] M. R. Nuwer, G. Comi, R. Emerson, A. Fuglsang-Frederiksen, J.-M. Guerit, H. Hinrichs, A. Ikeda, F. J. C. Luccas, and P. Rappelsberger, "IFCN standards for digital recording

of clinical EEG," in *Recommendations for the Practice of Clinical Neurophysiology: Guidelines of the International Federation of Clinical Physiology (Electroencephalography and Clinical Neurophysiology Supplement 52)*, 2nd ed., G. Deuschl, and A. Eisen, Eds. Amsterdam: Elsevier, 1999, pp. 11–14.

[2] G. L. Krauss and R. S. Fisher, *The Johns Hopkins Atlas of Digital EEG: An Interactive Training Guide*. Baltimore: Johns Hopkins University Press, 2006.

[3] E. A. Vittoz, "Future of analog in the VLSI environment," *IEEE ISCAS*, New Orleans, May 1990.

[4] C. D. Binnie, A. J. Rowan, and T. Gutter, *A Manual of Electroencephalographic Technology*. Cambridge, U.K.: Cambridge University Press, 1982.

[5] P. E. M. Smith and S. J. Wallace, *Clinicians' Guide to Epilepsy*. London: Arnold, 2001.

[6] E. Waterhouse, "New horizons in ambulatory electroencephalography," *IEEE Eng. Med. Biol. Mag.*, 22(3), pp. 74–80, 2003.

[7] B. Gyselinckx, C. Van Hoof, J. Ryckaert, R. Yazicioglu, P. Fiorini, and V. Leonov, "Human++: Autonomous wireless sensors for body area networks," *IEEE CICC*, San Jose, September 2005.

[8] R. F. Yazicioglu, P. Merken, R. Puers, and C. Van Hoof, "A 200 μW eight-channel EEG acquisition ASIC for ambulatory EEG systems," *IEEE J. Solid-State Circuits*, 43(12), pp. 3025–3038, 2008.

[9] D. C. Yates and E. Rodriguez-Villegas, "A key power trade-off in wireless EEG headset design," *IEEE EMBS NER*, Hawaii, May 2007.

[10] S. B. Wilson and R. Emerson, "Spike detection: A review and comparison of algorithms," *Clin. Neurophysiol.*, 113(12), pp. 1873–1881, 2002.

[11] Persyst Development Corporation, "Persyst application notes: Optimizing spike and seizure detection for long-term monitoring and event notifications II," Insights newsletter, Spring 2004.

[12] H. S. Liu, T. Zhang, and F. S. Yang, "A multistage, multimethod approach for automatic detection and classification of epileptiform EEG," *IEEE Trans. Biomed. Eng.*, 49(12), pp. 1557–1566, 2002.

[13] S. B. Wilson, R. N. Harner, F. H. Duffy, B. R. Tharp, M. R. Nuwer, and M. R. Sperling, "Spike detection. I. Correlation and reliability of human experts," *Electroencephalogr. Clin. Neurophysiol.*, 98(3), pp. 186–198, 1996.

[14] A. J. Casson and E. Rodriguez-Villegas, "Data reduction techniques to facilitate wireless and long term AEEG epilepsy monitoring," *IEEE EMBS NER*, Hawaii, May 2007.

[15] J. D. Frost, "Automatic recognition and characterization of epileptiform discharges in the human EEG," *J. Clin. Neurophysiol.*, 2(3), pp. 231–249, 1985.

[16] J. Gotman, "Automatic recognition of interictal spikes," in *Long-Term Monitoring in Epilepsy*, J. Gotman, J. R. Ives, and P. Gloor, Eds. Amsterdam: Elsevier, 1985, pp. 93–114.

[17] A. J. Casson, D. C. Yates, S. Patel, and E. Rodriguez-Villegas, "Algorithm for AEEG data selection leading to wireless and long term epilepsy monitoring," *IEEE EMBC*, Lyon, August 2007.

[18] A. J. Casson, D. C. Yates, S. Patel, and E. Rodriguez-Villegas, "An analogue bandpass filter realisation of the continuous wavelet transform," *IEEE EMBC*, Lyon, August 2007.

[19] K. P. Indiradevi, E. Elias, P. S. Sathidevi, S. Dinesh Nayak, and K. Radhakrishnan, "A multi-level wavelet approach for automatic detection of epileptic spikes in the electroencephalogram," *Com. Biol. Med.*, 38(7), pp. 805–816, 2008.

[20] J. Gotman and P. Gloor, "Automatic recognition and quantification of interictal epileptic activity in the human scalp EEG," *Electroencephalogr. Clin. Neurophysiol.*, 41(5), pp. 513–529, 1976.

[21] T. P. Exarchos, A. Tzallas, D. I. Fotiadis, S. Konitsiotis, and S. Giannopoulos, "EEG transient event detection and classification using association rules," *IEEE Trans. Inform. Technol. Biomed.*, 10(3), pp. 451–457, 2006.

[22] B. Ramabhadran, J. D. Frost, J. R. Glover, and P. Y. Ktonas, "An automated system for epileptogenic focus localization in the electroencephalogram," *J. Clin. Neurophysiol.*, 16(1), pp. 59–68, 1999.

[23] G. E. Chatrian, L. Bergamini, M. Dondey, D. W. Klass, M. Lennox-Buchthal, and I. Petersén, "A glossary of terms most commonly used by clinical electroencephalographers," *Electroencephalogr. Clin. Neurophysiol.*, 37(5), pp. 538–548, 1974.

[24] S. J. Mason and N. E. Graham, "Areas beneath the relative operating characteristics (ROC) and relative operating levels (ROL) curves: Statistical significance and interpretation," *Q. J. R. Meteorol. Soc.*, 128, pp. 2145–2166, 2002.

[25] M. Bland, *An Introduction to Medical Statistics*, 3rd ed. Oxford: Oxford University Press, 2000.

[26] J. C. Sackellares, D.-S. Shiau, K. M. Kelly, and S. P. Nair, "Testing a prediction algorithm: Assessment of performance," in *Seizure Prediction in Epilepsy*, B. Schelter, J. Timmer, and A. Schulze-Bonhage, Eds. Weinheim: Wiley, 2008, pp. 249–259.

[27] R. A. Fisher and F. Yates, *Statistical Tables for Biological, Agricultural and Medical Research*, 6th ed. Edinburgh: Oliver and Boyd, 1963.