ELSEVIER
Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computer Speech and Language 18 (2004) 1–24
COMPUTER SPEECH AND LANGUAGE
www.elsevier.com/locate/csl

Evaluation of BIC-based algorithms for audio segmentation

Mauro Cettolo [a,*], Michele Vescovi [b], Romeo Rizzi [b]

[a] *ITC-irst, Centro per la Ricerca Scientifica e Tecnologica, Via Sommarive, 18 Povo do Trento, I-38050 Povo, Trento, Italy*
[b] *Università degli Studi di Trento, Facoltà di Scienze MM.FF.NN. I-38050 Povo, Trento, Italy*

Received 24 June 2003; received in revised form 24 May 2004; accepted 26 May 2004

Abstract

The Bayesian Information Criterion (BIC) is a widely adopted method for audio segmentation, and has inspired a number of dominant algorithms for this application. At present, however, literature lacks in analytical and experimental studies on these algorithms. This paper tries to partially cover this gap.

Typically, BIC is applied within a sliding variable-size analysis window where single changes in the nature of the audio are locally searched. Three different implementations of the algorithm are described and compared: (i) the first keeps updated a pair of sums, that of input vectors and that of square input vectors, in order to save computations in estimating covariance matrices on partially shared data; (ii) the second implementation, recently proposed in literature, is based on the encoding of the input signal with cumulative statistics for an efficient estimation of covariance matrices; (iii) the third implementation consists of a novel approach, and is characterized by the encoding of the input stream with the cumulative pair of sums of the first approach.

Furthermore, a dynamic programming algorithm is presented that, within the BIC model, finds a globally optimal segmentation of the input audio stream.

All algorithms are analyzed in detail from the viewpoint of the computational cost, experimentally evaluated on proper tasks, and compared.

[*] Corresponding author. Tel.: +39-461-214-551; fax: +39-461-314-591.
E-mail addresses: cettolo@itc.it (M. Cettolo), vescovi@kirk.science.unitn.it (M. Vescovi), romeo@science.unitn.it (R. Rizzi).

0885-2308/$ - see front matter © 2004 Published by Elsevier Ltd.
doi:10.1016/j.csl.2004.05.008

1. Introduction

In the last years, efforts have been devoted amongst the research community to the problem of audio segmentation. The number of application of this procedure is considerable: from the extraction of information from audio data (e.g., broadcast news, recording of meetings) to the automatic indexing of multimedia data, or the improvement of accuracy for recognition systems. Typically, these tasks are performed by complex systems, consisting of a number of modules, some of them computationally expensive. Although in these systems the audio segmentation does not represent a computational bottleneck, the response time of the segmentation module can become an important issue under specific requirements. For example, ITC-irst delivered to RAI (the national Italian broadcasting company) an off-line system for the automatic transcription of broadcast news programs. Part of the requirement was also to supply a real-time version of the transcription station, able to guarantee adequate performance. Given these constraints, the speed of each component needs to be maximized without affecting its accuracy.

The segmentation problem has been handled in different ways that can be roughly grouped in three classes:

`energy` based methods: each silence occurring in the input audio stream is detected either by using an explicit model for the silence or by thresholding the signal energy. Segment boundaries are then located in correspondence of detected silences.

`metrics` based methods: the segmentation of the input stream is achieved by evaluating its "distance" from different segmentation models. Distances can be measured by the Hotelling's T^2 test (Wegmann et al., 1999), the Kullback Leibler distance (Kemp et al., 2000; Siegler et al., 1997), the generalized likelihood ratio (Gish et al., 1991), the entropy loss (Kemp et al., 2000), and the Bayesian Information Criterion (BIC) (Schwarz, 1978).

`explicit models` based methods: models are built for a given set of pre-determined acoustic classes – e.g., female and male speakers, music, noise, etc. Typically, the input data stream is classified from the maximum likelihood principle, through a dynamic programming decoding. The time indexes where the classification changes from one class to another are assumed to be the segment boundaries (Hain et al., 1998). Alternatively, in (Lu et al., 2001) Support Vector Machines are employed to learn class boundaries (Scheirer and Slaney, 1997); compares the Gaussian Maximum A-Posteriori estimator, the Gaussian Mixture Model, the Nearest-Neighbor classifier and a spatial partitioning scheme based on K-d trees.

The main limitations of the first and the third approach are evident. Regarding the energy based methods, there is only a partial correlation between changes in the nature of the audio and silences. On the other hand, these methods are simple to implement and can perform their (limited) task in linear time and with sufficient precision – provided the absence of significant variations in the background acoustic conditions. With explicit models of acoustic classes, changes occurring within the same class are undetectable; for example, if only one model for female voices is employed, no change can be detected within a dialogue between female speakers. Moreover, it is required both to know in advance the classes of interest and the availability of suitable data for their training. On the other hand, these methods can reach very high accuracy rates with linear time cost complexity.

This paper concentrates on a specific class of metric-based methods. Metric-based methods do not require any prior knowledge, nor a training stage. They are efficient (linear in time if some approximations are introduced – as shown later), simple to implement and are able to give good results. This explains why in a lot of laboratories much attention has been devoted to this type of methods, and in particular to the BIC (Cettolo, 2000; Cettolo and Vescovi, 2003; Chen and Gopalakrishnan, 1998; Delacourt et al., 1999; Harris et al., 1999; Sivakumaran et al., 2001; Tritschler and Gopinath, 1999; Vescovi et al., 2003; Wellekens, 2001). Typically, the BIC "distance" is used locally, within a shift variable-size window, where single changes in the nature of audio are searched. In this paper, we deal with metric-based audio segmentation algorithms based on the BIC.

In the research community it is customary to combine different (almost orthogonal) solutions to exploit advantages and circumvent limitations. This is also the case of audio segmentation. Usually, real systems include a partitioning module that segments, classifies and clusters the audio stream via a combination of different algorithms. Among the most successful systems we find indeed highly integrated partitioning modules, as for example in Gauvain et al. (1998) and in Hain et al. (1998). Even if this trend has allowed the development of very competitive systems, this has led to the situation where it is difficult to have a clear insight on how to improve the state of the art.

As a matter of fact, literature regarding analytical and experimental studies of segmentation algorithms remains – somewhat – limited. The only known exception is Sivakumaran et al. (2001), where an efficient approach to the local BIC-based segmentation algorithm is proposed and its computational cost briefly analyzed. However, only a very limited comparison with other algorithms is given. In that work, the input audio stream is progressively encoded by cumulative statistics, and the encoding is exploited to avoid redundant operations in the computation of BIC values.

In this paper, we try to cover part of the studies on segmentation algorithms by proposing: (i) an innovative method for the implementation of the BIC-based local algorithm – this guarantees (to the best of our knowledge) the lowest computational cost; (ii) a global algorithm capable of finding the optimal BIC input segmentation – this represents the performance upper bound for the class of the BIC-based segmentation algorithms. The algorithms are described and compared, both analytically and experimentally, with the method given in Sivakumaran et al. (2001).

The paper is organized as follows. In Section 2, a general definition of the audio segmentation problem is given.

Section 3 presents the corpora used for experiments, together with the evaluation measures; a brief description of the signal processing front-end is also given.

In Section 4, the BIC-based local algorithm as proposed in Delacourt et al. (1999) is described, building on the general framework provided in Section 2. The computational cost of the local algorithm presented in Sivakumaran et al. (2001) is analyzed in detail (Section 4.1.2) and compared, both in theory and experimentally, with the cost of two other possible approaches: one more direct but more expensive, which represents a reference (Section 4.1.1); and a novel method that combines the good ideas of the other two (Section 4.1.3). On a test set of broadcast news programs, the new approach is 35% faster than the one proposed in Sivakumaran et al. (2001) (Section 4.2).

Section 5 presents, within the general framework of Section 2, a Dynamic Programming (DP) algorithm which uses the BIC method to find the globally optimal segmentation of the input audio. The algorithm with its computational costs are described in detail. It is also compared to the most efficient implementation of the local algorithm. On the 2000 NIST Speaker Recognition Evaluation test set, the global algorithm outperforms the local one by 2.4% (relative) F-score in the detection of changes, but is 38 times slower.

A summary ends the paper.

2. The segmentation problem and the BIC

Segmenting an audio stream consists of detecting the time indexes corresponding to changes in the nature of the audio, this to isolate segments that are acoustically homogeneous. This can be seen as a particular instance of the more general problem of partitioning data into distinct homogeneous regions (Baxter, 1996). The data partitioning problem arises in all applications that require partitioning data into chunks, e.g., image processing, data mining, text processing, etc.

The problem can be formulated as follows. Let $\mathcal{O} = o_1, o_2, \ldots, o_N$ be an ordered sequence of observations, called the *sample*, in which each o_i is a vector in $\mathbb{R}^d$. We assume that the sample is generated by a Gaussian process with a certain number of transitions. The problem of segmentation is that of detecting all the transition points in the data set. In the ambit of acoustic segmentation, transitions correspond to changes in the nature of the audio, represented by the sample $\mathcal{O}$.

A particular model of the input data is characterized by a specific number c of changes and a specific set $\{1 \leqslant t_1, t_2, \ldots, t_c < N\}$ of time indexes corresponding to these changes. In such a way, the input data are modeled by a set of $c+1$ Gaussian distributions, each one generating an observation subsequence bounded by two consecutive time indexes out of $\{1, t_1, \ldots, t_c, N\}$.

Among all possible models of the sample, the "best" one has to be selected, and its time indexes will define the segmentation of the input stream. The best model is the one that better fits the observations. The application of the maximum likelihood principle would however invariably lead to choosing the model with the maximum number of changes ($c_{\max} = N - 1$), as this model has the highest number of free parameters. In order to take into account the notion of "dimension" of the model, the following extension to the maximum likelihood principle was first proposed by Akaike (1977). The AIC (Akaike's Information Criterion) suggests to maximize the likelihood for each model M_i separately, obtaining say $L_{M_i} = L_{M_i}(\mathcal{O})$, and then choose the model for which $(\log L_{M_i} - F_{M_i})$ is largest, where F_{M_i} is the number of free parameters of the model M_i.

Several model selection criteria that can be applied to Akaike's framework of model selection have been proposed in the literature (see Cettolo and Federico, 2000 for a review). In general, each criterion proposes the introduction of a penalty function P that takes into account the dimension of the model.

The BIC is the penalty function proposed in Schwarz (1978), and is defined as

$$P_M = \frac{F_M}{2} \log N. \tag{1}$$

As an example, let us consider a sample of three observations $\mathcal{O} = o_1, o_2, o_3$. The possible models of $\mathcal{O}$ to be compared are:

- a 1-Gaussian process (let it be M_1):

$$o_1, o_2, o_3 \sim_{iid} N_d(\mu, \Sigma),$$

- two 2-Gaussians processes (M_2 and M_3):

$$o_1, o_2 \sim_{iid} N_d(\mu_{\mathrm{A}}, \Sigma_{\mathrm{A}}),$$

$$o_3 \sim_{iid} N_d(\mu_{\mathrm{B}}, \Sigma_{\mathrm{B}})$$

and:

$$o_1 \sim_{iid} N_d(\mu_{\mathrm{A}}, \Sigma_{\mathrm{A}}),$$

$$o_2, o_3 \sim_{iid} N_d(\mu_{\mathrm{B}}, \Sigma_{\mathrm{B}}),$$

- a 3-Gaussians process (M_4):

$$o_1 \sim_{iid} N_d(\mu_{\mathrm{A}}, \Sigma_{\mathrm{A}}),$$

$$o_2 \sim_{iid} N_d(\mu_{\mathrm{B}}, \Sigma_{\mathrm{B}}),$$

$$o_3 \sim_{iid} N_d(\mu_{\mathrm{C}}, \Sigma_{\mathrm{C}}).$$

Once $(\log L_{M_i}(\mathcal{O}) - P_{M_i})$ has been computed for each $i = 1, 2, 3, 4$, the highest value gives the winner model. If it is, for example, M_2, the best segmentation of $\mathcal{O}$ under the BIC consists of the following two segments: $(o_1 o_2)$ and (o_3).

2.1. Search

Given the set $\{M_i\}$ of possible models of the input sequence, the segmentation problem now reduces to solve the following maximization:

$$i_{\max} = \arg\max_i \log L_{M_i} - P_{M_i}. \qquad (2)$$

Instead of explicitly listing the set of all possible models, whose size is $\mathrm{O}\{2^N\}$, polynomial search algorithms can be employed. A widely adopted search algorithm uses a sliding variable-size analysis window, where the maximization given above is used to detect single changes. That is, models compared in Eq. (2) have only one transition or none at all. The algorithm and three implementations with different computational costs are described in Section 4. The algorithm main limitation is that the search is done locally – hence the objective function that determines the most plausible interpretations of the input audio stream is not globally maximized. In principle, this could affect the effectiveness of the method, especially if the audio stream includes many changes close to each other – as in the case of human–human conversations. In Section 5, a global algorithm able to find the solution of Eq. (2) in the BIC framework, is presented.

3. Data sets and evaluation measures

Two corpora were selected for experiments, namely the Italian Broadcast News Corpus (IBNC) and the 2000 NIST Speaker Recognition Evaluation corpus. The two collections are substantially different in the nature of audio recordings, allowing the evaluation of algorithms under very different working conditions.

3.1. The IBNC corpus

The IBNC corpus (Federico et al., 2000), collected by ITC-irst and available through ELDA, the Evaluations and Language resources Distribution Agency (IBNC, 2000), is a speech corpus of radio broadcast news in Italian. It consists of 150 recordings, for a total of about 30 h, covering radio news of several years. Data were provided by RAI. The corpus presents variations of topics, speakers, channel band (i.e., studio versus telephone), speaking mode (i.e., spontaneous versus planned), etc. It has been manually transcribed, segmented and labeled. Speaker gender and, when possible, identity are also annotated.

For testing purposes, six programs (about 75 min of audio) were selected, where 212 changes occur, between either different speakers or different acoustic classes (music, speech, noise, etc.).

In broadcast news, acoustically homogeneous segments are typically long. This makes IBNC data suitable for assessing the ability of segmentation algorithms in the detection of changes around which much homogeneous data is available.

3.2. The 2000 NIST speaker recognition evaluation

The 2000 NIST Speaker Recognition Evaluation (NIST, 2000) included a segmentation task, where systems were required to identify speech segments corresponding to each of two unknown speakers. The test set consists of 1000 telephone conversations, lasting about 1 min each, included in the Disc `r65_6_1`. 777 Speakers, of both genders, pronounced a total of 46K turns/segments. Fig. 1 shows the distribution of segment length. The longest segment is 26.5 s; the mean length is 1.3 s, while the median is less than 1 s; it is worth noticing that 81% of segments has length lower

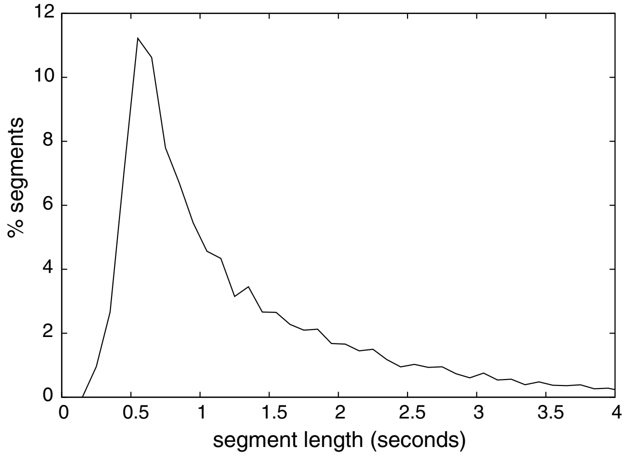

Fig. 1. Distribution of length of NIST test set segments.

than 2 s. This means that the NIST data are suitable to make evident if segmentation algorithms are able to detect changes bounding very short segments.

The NIST test set was also employed in this work. For evaluation, we relied on a reference segmentation obtained by merging the two reference segmentations provided by NIST for each speaker. It was observed that often the end of a sentence of a speaker does not exactly correspond to the beginning of the successive sentence of the other speaker: this is due either to small imprecision of the manual annotation, or to a short overlap of the sentences, or to a brief silence occurring between the two sentences. In such cases, short segments created when merging were removed by an automatic procedure.

3.3. Evaluation measures

Performance of automatic change detection is calculated with respect to a set of target changes. Tolerances in the detection can be introduced, e.g., ±0.5 s: in such a way the target is an interval rather than a single time index.

For comparing target and hypothesized changes, we adopt the *precision P* and the *recall R* measures:

$$P = \frac{c}{c+i} \times 100,$$

$$R = \frac{c}{c+d} \times 100,$$

where c is the number of target intervals which contain at least one hypothesized change, i is the number of hypothesized changes that do not fall inside any target interval (insertions), and d is the number of target intervals which contain no hypothesized change (deletions). In other words, precision is related to the number of false alarms, i.e., the number of wrongly split segments, while recall measures the deletion rate of correct changes, i.e., the number of wrongly merged segments.

The evaluation of the segmentation quality is made in terms of *F-score*, a combined measure of P and R of change detection. F-score is a measure proposed by van Rijsbergen (Frakes and Baeza-Yates, 1992) and is defined as

$$F = \frac{(1+\beta^2)PR}{\beta^2 P + R},$$

where β is a measure of the relative importance, to a user, of precision and recall. For example, β levels of 0.5, indicating that a user was twice as interested in precision as recall, and 2, indicating that a user was twice as interested in recall as precision, might be used. The most popular value corresponds to $\beta = 1$, for which F reduces to

$$F = \frac{2PR}{P+R}.$$

In this paper β was set to 1.

Concerning the NIST data set, after the official evaluation, NIST made available both the reference segmentations of the test files and a scoring script. The scoring script computes the segmentation error rate, i.e., the percentage of speech from one speaker wrongly assigned to the other speaker. The measure is suitable for the NIST task which, by definition, is a classification task. On the contrary, the algorithm presented here detects speaker/spectral changes, and no attempt is done to classify segments in terms of speakers. The NIST scoring script does not therefore fit with our evaluation requirements.

In fact, precision and recall (or the corresponding false alarm and miss detection rates) are metrics that better assess the performance of the change detection algorithm. For this reason, the two metrics and their F-score are also used in the evaluation of the algorithms with the NIST data set.

3.4. Audio processing

Multivariate observations derive from a short time spectral analysis, performed over 20 ms Hamming windows at a rate of 10 ms. For every window, 12 Mel scaled Cepstral coefficients and the log-energy are evaluated.

4. Local search: an approximated algorithm

First of all, let us recall some basic results. Given a sample $\mathcal{O} = o_1, o_2, \ldots, o_N$ of observations $o_i \in \mathbb{R}^d$, the likelihood function $L_{N_d(\mu,\Sigma)}(\mathcal{O})$ achieves its maximum value (Seber, 1984) in $\mu = \bar{o}$, the sample mean, and

$$\Sigma = \hat{\Sigma} = \frac{1}{N}\sum_{i=1}^{N}(o_i - \bar{o})(o_i - \bar{o})^{\mathrm{tr}}, \tag{3}$$

the maximum-likelihood estimate of the covariance matrix. Moreover

$$L_{N_d(\bar{o},\hat{\Sigma})}(\mathcal{O}) = (2\pi)^{-Nd/2}|\hat{\Sigma}|^{-N/2}e^{-Nd/2}. \tag{4}$$

From here on, sample means and estimates of covariance matrices will be denoted simply by μ and Σ: it should be clear from the context if symbols refer to variables or evaluated statistical estimators.

The number of free parameters in a multivariate normal distribution is equal to the dimension of the mean plus the number of variances and covariances to be estimated. For the full covariance matrix case, it is

$$F_{N_d(\mu,\Sigma)} = d + d\frac{(d+1)}{2}. \tag{5}$$

Assuming that the sequence $o_1, \ldots, o_N$ contains at most one change, and that data are generated by a Gaussian process, the maximization of Eq. (2) regards the following N different statistical models:

- $N-1$ two-segments models M_i ($i = 1, \ldots, N-1$), where model M_i states:

$$o_1, \ldots, o_i \sim_{iid} N_d(\mu_1, \Sigma_1),$$

$$o_{i+1}, \ldots, o_N \sim_{iid} N_d(\mu_2, \Sigma_2),$$

- one single-segment model M_N which states

  $$o_1, o_2, \ldots, o_N \sim_{iid} N_d(\mu, \Sigma).$$

The selection of the best model can then rely on the following decision rule:

- Look for the best two-segments model $M_{i_{\max}}$ for the data

  $$i_{\max} = \arg\max_{i=1,\ldots,N-1} \log L_{M_i} - P_{M_i}.$$

- Take the one-segment model function

  $$\log L_{M_N} - P_{M_N}. \qquad (6)$$

- Choose to segment the data at point $i_{\max}$ if and only if

  $$(\log L_{M_{i_{\max}}} - \log L_{M_N}) - (P_{M_{i_{\max}}} - P_{M_N}) > 0. \qquad (7)$$

If the BIC penalty term (1) is adopted, from Eqs. (4) and (5) it is easy to show that the above described decision rule is equivalent to computing for each $i = 1, \ldots, N-1$ the quantity

$$\Delta\mathrm{BIC}_i = \frac{N}{2}\log|\Sigma| - \frac{i}{2}\log|\Sigma_1| - \frac{(N-i)}{2}\log|\Sigma_2| - \lambda P_{N_d(\mu,\Sigma)}, \qquad (8)$$

where Σ, Σ_1 and Σ_2 are the maximum likelihood covariance estimates on $o_1 \ldots o_N$, $o_1 \ldots o_i$ and $o_{i+1} \ldots o_N$, respectively, and to hypothesize a change in $i_{\max}$, if $i_{\max}$ is the index that maximizes

```
init_window(1, $N_{min}$)

while(not end stream)

  ($\Delta BIC_{i_{max}}$, $i_{max}$) ← compute_ΔBIC($\delta_1$)

  while($\Delta BIC_{i_{max}} \le 0$ and
        current_win_size < $N_{max}$ and
        not end stream)
    growth_win($\Delta N_{grow}$)
    ($\Delta BIC_{i_{max}}$, $i_{max}$) ← compute_ΔBIC($\delta_1$)

  while($\Delta BIC_{i_{max}} \le 0$ and not end stream)
    shift_win($\Delta N_{shift}$)
    ($\Delta BIC_{i_{max}}$, $i_{max}$) ← compute_ΔBIC($\delta_1$)

  if($\Delta BIC_{i_{max}} > 0$) then
    center_win($i_{max}$, min(current_win_size, $N_{second}$))
    ($\Delta BIC_{i_{change}}$, $i_{change}$) ← compute_ΔBIC($\delta_h$)
    if($\Delta BIC_{i_{change}} > 0$) then
      output($i_{change}$)
      init_window($i_{change} + 1$, $N_{min}$)
    else
      init_window($i_{max} - N_{margin} + 1$, $N_{min}$)
```

Fig. 2. Pseudocode of the local sliding window algorithm.

$\Delta\mathrm{BIC}_i$ and $\Delta\mathrm{BIC}_{i_{\max}} > 0$. $\lambda \in \mathbb{R}$ is a weight, which allows to tune the sensitivity of the method to the particular task under consideration.

In order to apply the search of single changes to an arbitrary large number of potential changes, we implemented the local algorithm depicted in Fig. 2, inspired by that proposed in Delacourt et al. (1999). The main idea is to have a shifting variable-size window for the computation of ΔBIC values. The algorithm dynamically adapts the window size in such a way that no more than one change falls inside the window, ensuring the computation of reliable statistics and bounding the computational cost. Moreover, to save computations, ΔBIC values are not computed on all observations, but at a lower resolution, namely once every δ observations. The resolution is successively increased if a potential change is detected, in order to validate it and to refine its time position.

The main steps of the algorithm are:

Search start. ΔBIC values are computed only for the first $N_{\min}$ observations. $N_{\min}$ is the minimum size of the window. Values are computed with low resolution, e.g., $\delta_l = 30$. In order to have enough observations for computing both Σ_1 and Σ_2, ΔBIC are not computed for the N_{margin} indexes close to the left and right boundaries of the window.

Window growth. The window is enlarged by including ΔN_{grow} input observations until a change is detected, or a maximum size $N_{\max}$ is reached.

Window shift. The $N_{\max}$-sized window is shifted forward by $\Delta N_{\mathrm{shift}}$ observations.

Change confirmation. If in one of the three previous steps a change is detected, ΔBIC values are re-computed with the high resolution, e.g., $\delta_h \approx \delta_l/5$, centering the window at the hypothesized

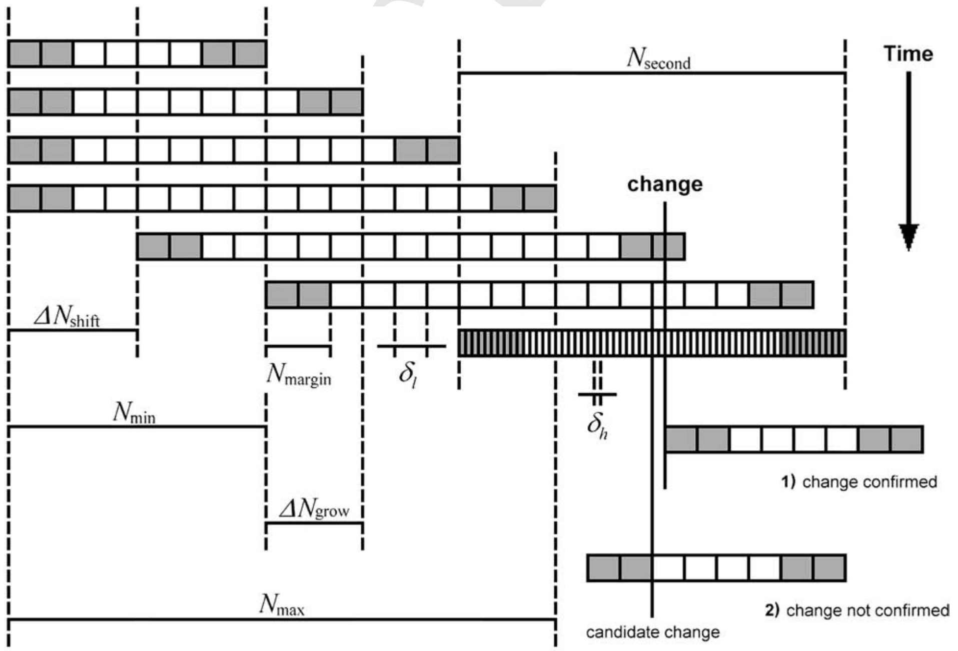

Fig. 3. Working scheme of the local sliding window algorithm.

change. The current size of the window is kept, unless it is larger than N_{second} observations, in which case it is narrowed to that value. If a change is detected again, it is output by the algorithm.

Window reset. After the change confirmation step, the algorithm has to go on resizing the analysis window to the minimum value $N_{\min}$ and locating it in a position dependent on the result of the confirmation step (see Fig. 3).

4.1. Computations

In the following subsections, three possible implementations of the algorithm described above, ordered according to decreasing computational cost, are presented in detail.

4.1.1. The sum approach (SA)

The evaluation of Eq. (8) determines the overall computational cost of the algorithm presented above, since a high number of ΔBIC values have to be computed for each window.

An efficient way to compute the determinant of the covariance matrix is based on the Cholesky decomposition which requires $\mathrm{O}\{d^3/6\}$ operations. The estimation of the mean vector μ and the covariance matrix Σ on N d-sized observations $o_a, \ldots, o_b$:

$$\mu_a^b = \frac{1}{N} \sum_{i=a}^{b} o_i, \tag{9}$$

$$\Sigma_a^b = \frac{1}{N} \left(\sum_{i=a}^{b} o_i \cdot o_i^{\mathrm{tr}} \right) - \mu_a^b \cdot \mu_a^{b^{\mathrm{tr}}} \tag{10}$$

requires, respectively, $d(N+1)$ and $d(d+1)(N+1.5)$ operations. Typically, the window size N is significantly larger than the vector dimension d, hence the computational cost for the evaluation of the covariance matrix determinant could be discarded.

In order to reduce the computational cost of estimating likelihoods of the normal distributions required for the computation of ΔBIC values, it is convenient to keep the sums of the input vectors (SV) and that of the square vectors (SQ):

$$\mathrm{SV}_a^b = \sum_{i=a}^{b} o_i, \quad \mathrm{SQ}_a^b = \sum_{i=a}^{b} o_i \cdot o_i^{\mathrm{tr}}.$$

In fact, besides the easy computation of the needed parameters:

$$\mu_a^b = \frac{1}{N} \cdot \mathrm{SV}_a^b, \quad \Sigma_a^b = \frac{1}{N} \cdot \mathrm{SQ}_a^b - \mu_a^b \cdot \mu_a^{b^{\mathrm{tr}}},$$

the use of SV and SQ avoids many redundant operations in the computation of ΔBIC values both within a given window and after a window growth/shift. With reference to the notation in Table 1, the following cases can happen:

- **growth** of the window by δ observations:
  - ○ $\mathrm{SV}_{\tilde{T}} = \mathrm{SV}_T + \sum_{j=n+N+1}^{n+N+\delta} o_j$,
  - ○ $\mathrm{SQ}_{\tilde{T}} = \mathrm{SQ}_T + \sum_{j=n+N+1}^{n+N+\delta} o_j \cdot o_j^{\mathrm{tr}}$.

Table 1
Notation

N	Current window size
n	Index of the vector that preceeds the first index of the window
T	Set of vectors inside the window $\{o_{n+1} \ldots o_{n+N}\}$
$\tilde{T}$	Set of vectors inside the window after a growth or a shift
A_k	Set of the first $k\delta$ vectors of the window $\{o_{n+1} \ldots o_{n+k\delta}\}$
B_k	Set of the last $N - k\delta$ vectors of the window $\{o_{n+k\delta+1} \ldots o_{n+N}\}$
SV_X	Sum of vectors of the set X
SQ_X	Sum of square vectors of the set X
Σ_X	Covariance matrix on the vectors of the set X
μ_X	Mean vector on the vectors of the set X

- **shift** of the window by δ observations:
  - ○ $\mathrm{SV}_{\tilde{T}} = \mathrm{SV}_T - \sum_{j=n+1}^{n+\delta} o_j + \sum_{j=n+N+1}^{n+N+\delta} o_j$,
  - ○ $\mathrm{SQ}_{\tilde{T}} = \mathrm{SQ}_T - \sum_{j=n+1}^{n+\delta} o_j \cdot o_j^{\mathrm{tr}} + \sum_{j=n+N+1}^{n+N+\delta} o_j \cdot o_j^{\mathrm{tr}}$.
- **computation** of $\Delta\mathrm{BIC}_i$ (at resolution δ):
  - ○ $\mathrm{SV}_{A_i} = \mathrm{SV}_{A_{i-1}} + \sum_{j=n+(i-1)\delta+1}^{n+i\delta} o_j$,
  - ○ $\mathrm{SQ}_{A_i} = \mathrm{SQ}_{A_{i-1}} + \sum_{j=n+(i-1)\delta+1}^{n+i\delta} o_j \cdot o_j^{\mathrm{tr}}$,
  - ○ $\mathrm{SV}_{B_i} = \mathrm{SV}_T - \mathrm{SV}_{A_i}$,
  - ○ $\mathrm{SQ}_{B_i} = \mathrm{SQ}_T - \mathrm{SQ}_{A_i}$.

With this approach, in each step of the algorithm the number of operations for computing Eq. (8) is:

- **growth** of the window by δ observations:

$$\underbrace{d(d+1)\cdot\delta}_{\mathrm{SQ}_{\tilde{T}}} + \underbrace{d\cdot\delta}_{\mathrm{SV}_{\tilde{T}}},$$

- **shift** of the window by δ observations:

$$\underbrace{d(d+1)\cdot 2\cdot\delta}_{\mathrm{SQ}_{\tilde{T}}} + \underbrace{d\cdot 2\cdot\delta}_{\mathrm{SV}_{\tilde{T}}},$$

- **computation** of the covariance matrix of the whole window:

$$\underbrace{d}_{\mu_T} + \underbrace{1.5\cdot d(d+1)}_{\Sigma_T},$$

- **computation** of Σ_{A_i} and Σ_{B_i} required for the evaluation of $\Delta\mathrm{BIC}_i$, with resolution δ $(\forall i,\ i = N_{\mathrm{margin}}/\delta + 1, \ldots, (N - N_{\mathrm{margin}})/\delta - 1)$:

$$\underbrace{d\cdot\delta}_{\mathrm{SV}_{A_i}} + \underbrace{d(d+1)\cdot\delta}_{\mathrm{SQ}_{A_i}} + \underbrace{d}_{\mathrm{SV}_{B_i}} + \underbrace{d(d+1)/2}_{\mathrm{SQ}_{B_i}} + \underbrace{2\cdot d}_{\mu_{A_i},\mu_{B_i}} + \underbrace{3\cdot d(d+1)}_{\Sigma_{A_i},\Sigma_{B_i}} = d(d+1)(\delta+3.5) + d(\delta+3).$$

4.1.2. The distribution approach (DA)

In order to further reduce the computational cost of the algorithm, it is possible to evaluate Eq. (8) through the approach proposed in Sivakumaran et al. (2001).

Let Σ_N and μ_N be the sample covariance matrix and the mean of a set of N d-dimensional observations. If a (sub)set of Δ observations with covariance matrix Σ_Δ and mean vector μ_Δ has to be added or subtracted to that set, the parameters of the updated set of vectors can be computed by:

$$\Sigma_{N\pm\Delta} = \frac{N}{N\pm\Delta}\Sigma_N \pm \frac{\Delta}{N\pm\Delta}\Sigma_\Delta \pm \frac{N\Delta}{(N\pm\Delta)^2}(\mu_N-\mu_\Delta)(\mu_N-\mu_\Delta)^{\mathrm{tr}}, \tag{11}$$

$$\mu_{N\pm\Delta} = \frac{N}{N\pm\Delta}\mu_N \pm \frac{\Delta}{N\pm\Delta}\mu_\Delta. \tag{12}$$

This formulation requires only $3\cdot d(d+1)+d$ and $3\cdot d$ operations for computing $\Sigma_{N\pm\Delta}$ and $\mu_{N\pm\Delta}$, respectively, instead of $d(d+1)(N\pm\Delta+1.5)$ and $d(N\pm\Delta+1)$ required by the plain definitions.

The alternative approach consists in computing from the input audio stream $o_1, o_2, \ldots, o_{N_{\mathrm{audio}}}$ the set of triples (Σ_1^n, μ_1^n, n), where $n=\delta_{\mathrm{h}}, 2\delta_{\mathrm{h}}, 3\delta_{\mathrm{h}}, \ldots, N_{\mathrm{audio}}$.

The key of this processing is Eqs. (11) and (12) which allow to obtain (Σ_1^n, μ_1^n, n) from $(\Sigma_1^{n-\delta_{\mathrm{h}}}, \mu_1^{n-\delta_{\mathrm{h}}}, n-\delta_{\mathrm{h}})$ and $(\Sigma_{n-\delta_{\mathrm{h}}+1}^n, \mu_{n-\delta_{\mathrm{h}}+1}^n, \delta_{\mathrm{h}})$, where $\Sigma_{n-\delta_{\mathrm{h}}+1}^n$ and $\mu_{n-\delta_{\mathrm{h}}+1}^n$ are computed directly from the vectors $o_{n-\delta_{\mathrm{h}}+1}, o_{n-\delta_{\mathrm{h}}+2}, \ldots, o_n$ through the definitions.

Since in this approach the estimation of a new distribution is based on already computed distributions, it will be referred with the name "distribution approach" (DA).

By constraining δ_{l} and N_{second} to be integers multiples of δ_{h} and by choosing $N_{\min}$, $N_{\max}$, ΔN_{grow}, $\Delta N_{\mathrm{shift}}$, N_{margin} to be divisible by δ_{l}, it is possible to use the cumulative distributions (Σ_1^n, μ_1^n, n) for the evaluation of ΔBIC values and to reduce the cost of the computation. In fact, whatever the step of the algorithm is, the covariance matrices required by Eq. (8) can be estimated from Eqs. (11) and (12):

$$\Sigma_{n+1}^{n+N} = \frac{n+N}{N}\Sigma_1^{n+N} - \frac{n}{N}\Sigma_1^n - \frac{(n+N)n}{N^2}\left(\mu_1^{n+N}\mu_1^n\right)\left(\mu_1^{n+N}-\mu_1^n\right)^{\mathrm{tr}}, \tag{13}$$

$$\Sigma_{n+1}^{n+i} = \frac{n+i}{i}\Sigma_1^{n+i} - \frac{n}{i}\Sigma_1^n - \frac{(n+i)n}{i^2}\left(\mu_1^{n+i}-\mu_1^n\right)\left(\mu_1^{n+i}-\mu_1^n\right)^{\mathrm{tr}}, \tag{14}$$

$$\Sigma_{n+i+1}^{n+N} = \frac{n+N}{N-i}\Sigma_1^{n+N} - \frac{n+i}{N-i}\Sigma_1^{n+i} - \frac{(n+N)(n+i)}{(N-i)^2}\left(\mu_1^{n+N}-\mu_1^{n+i}\right)\left(\mu_1^{n+N}-\mu_1^{n+i}\right)^{\mathrm{tr}}. \tag{15}$$

Clearly, Eq. (13) is evaluated only once for a given window, while Eqs. (14) and (15) have to be evaluated for each time index of interest (depending on the resolution). A scheme of the DA approach is given in Fig. 4.

The number of operations required by each step of the algorithm with the DA approach is:

- **growth** or **shift** of the window by δ observations:

$$\underbrace{d(\delta+1)}_{\mu_{n+N+1}^{n+N+\delta}} + \underbrace{d(d+1)(\delta+1.5)}_{\Sigma_{n+N+1}^{n+N+\delta}} + \underbrace{3\cdot d(d+1)+4\cdot d}_{(\Sigma_1^{n+N+\delta},\mu_1^{n+N+\delta},n+N+\delta)} = d(d+1)(\delta+4.5)+d(\delta+5).$$

Note that this cost is that of the input stream encoding.

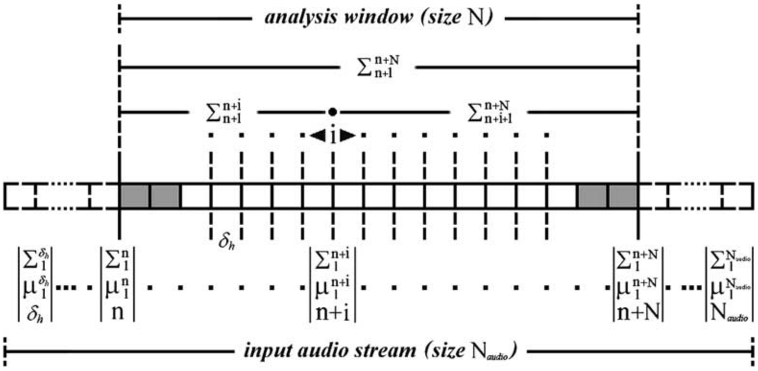
analysis window (size N)
Σ_{n+1}^{n+N}
Σ_{n+1}^{n+i}
Σ_{n+i+1}^{n+N}
i
δ_h
$\Sigma_1^{\delta_h}$ $\mu_1^{\delta_h}$ δ_h
Σ_1^{n} μ_1^{n} n
Σ_1^{n+i} μ_1^{n+i} n+i
Σ_1^{n+N} μ_1^{n+N} n+N
$\Sigma_1^{N_{audio}}$ $\mu_1^{N_{audio}}$ N_{audio}
input audio stream (size N_{audio})

Fig. 4. $\Delta\mathrm{BIC}_i$ computation in the DA approach.

- **computation** of the covariance matrix of the whole window:

$$\underbrace{3 \cdot d(d+1) + d}_{\Sigma_{n+1}^{n+N} = \Sigma_T},$$

- **computation** of Σ_{A_i} and Σ_{B_i} required for the evaluation of $\Delta\mathrm{BIC}_i$, with resolution δ $(\forall i,\ i = N_{\mathrm{margin}}/\delta + 1, \ldots, (N - N_{\mathrm{margin}})/\delta - 1)$:

$$\underbrace{6 \cdot d(d+1) + 2 \cdot d}_{\Sigma_{n+1}^{n+i\delta} = \Sigma_{A_i}, \Sigma_{n+i\delta+1}^{n+N} = \Sigma_{B_i}}.$$

4.1.3. The cumulative sum approach

In the previous methods, the estimation of the statistics required for the computation of the BIC are based either on the use of the sum and square sum of input vectors that fall inside the analysis window, or on the use of the set of statistics computed only once, as soon as the observations from the input stream are available. A combination of the two basic ideas gives the possibility to implement an even more efficient approach.

The idea is to encode the input stream, not through the distributions as in DA, but with the sums of the SA approach, that is with the sequence of triples $(\mathrm{SQ}_1^n, \mathrm{SV}_1^n, n)$ computed at resolution δ_h. In this new approach, the cost of each step of the algorithm, which can be called "cumulative sum approach" (CSA), is reported in the last column of Table 2. This table also includes, as a summary, the SA and DA costs.

Table 2
Cost of each step of the SA, DA, and CSA approaches

Step	SA	DA	CSA
Growth	$\delta d(d+1) + \delta d$	$(\delta + 4.5)d(d+1) + (\delta + 5)d$	$\delta d(d+1) + \delta d$
Shift	$2\delta d(d+1) + 2\delta d$	$(\delta + 4.5)d(d+1) + (\delta + 5)d$	$\delta d(d+1) + \delta d$
Σ_T	$1.5d(d+1) + d$	$3d(d+1) + d$	$2d(d+1) + 2d$
$\Sigma_{A_i}, \Sigma_{B_i}$	$(\delta + 3.5)d(d+1) + (\delta + 3)d$	$6d(d+1) + 2d$	$4d(d+1) + 4d$

The higher efficiency is given by: (i) the redundant computations in SA are avoided since each input vector is used only once, during the encoding of the input stream; (ii) the new encoding is cheaper than the DA encoding (cf. the grow/shift costs); (iii) the computation of covariance matrices from sums requires less operations than starting from other distributions.

4.2. Experimental evaluation

4.2.1. Database

The main goal of these experiments is to compare the running times of the three implementations of the local algorithm. Since the variable-size window algorithm performs more operations with long segments than with short ones, the IBNC corpus (see Section 3.1) suits the nature of this kind of comparison.

4.2.2. Costs comparison

Since the computation of $\Delta\mathrm{BIC}_i$ values is done approximately N/δ times in each window, the total cost of the algorithm mainly depends on the cost of that operation, and this is the reason for which the DA and CSA approaches are convenient with respect to the SA approach; for example, the number of operations with the DA approach does not depend on δ, whereas the SA does, and in our case ($d = 13$) it results convenient for $\delta \geqslant 3$.

In order to validate the theoretical comparison described in Sections 4.1.1, 4.1.2 and 4.1.3, and in particular the dependence of the overall computational cost from the resolution δ, the three implementations have been run with a simplified setup. We set $N_{\min} = N_{\max}$, in order to eliminate the window grow step, and the value of λ was set high enough that no candidate change was detected, constraining the computations to be done only at resolution $\delta = \delta_{\mathrm{l}}$.

Table 3
Theoretical and experimental costs comparison of SA, DA and CSA approaches

δ	# Operations			Execution time (s)		
	SA ($\times 10^6$)	DA/SA (%)	CSA/SA (%)	SA (s)	DA/SA (%)	CSA/SA (%)
1	374.0	123.9	92.1	830.0	103.8	68.0
5	124.4	80.6	61.5	272.0	68.3	46.5
10	93.0	59.0	46.3	200.6	50.4	35.5
25	73.6	37.6	31.2	157.5	32.0	24.5

Setup: $N_{\min} = N_{\max} = 1000,\ \Delta N_{\mathrm{shift}} = 200,\ N_{\mathrm{margin}} = 50,\ d = 13.$

Table 4
Performance comparison of SA, DA and CSA approaches in their best setup: $N_{\min} = 500, N_{\max} = 2000, N_{\mathrm{second}} = 1500, \Delta N_{\mathrm{grow}} = 100, \Delta N_{\mathrm{shift}} = 300, N_{\mathrm{margin}} = 100, \delta_{\mathrm{l}} = 25, \delta_{\mathrm{h}} = 5, \lambda = 2.175$

	F-score	Execution time	
		s	☆/SA (%)
SA	88.4	289.4	100.0
DA	89.4	101.0	34.9
CSA	89.4	65.9	22.8

Given the setup in the caption and setting $N_{\text{audio}} = 50,000$, the total number of operations required by the three approaches are given in the columns "#operations" of Table 3, for different values of δ. The costs include the computation of the covariance matrices determinant ($d^3/6$), since it can no longer be neglected. Again with the setup in the caption, the execution times were measured on a Pentium III 600 MHz on the 75-min IBNC test set.

Finally, Table 4 compares the three approaches on the algorithm setup detailed in the caption. The table shows results in terms of *F-score* on the IBNC test set, together with execution times. The slight difference in F-score for SA is due to some minor differences in the SA implementation. Concerning the execution times, since δ_1 was set to 25, the ratio between the costs of the three implementations expected from the results of the last row of Table 3 is confirmed.

5. Global search: a DP algorithm

In order to avoid the approximation inherent in the local search, we developed a DP algorithm able to find the globally optimal segmentation of the input audio stream according to the maximization (2) and the BIC penalty term (1). Between the $O\{2^N\}$ possible segmentations, it allows to select the best one in $O\{N^3\}$ steps, as described in the following.

The BIC value in Eq. (1) can be seen as the difference between the log-likelihood ($\log L_M(\mathcal{O})$) and the penalty term $P_M = \frac{F_M}{2}\log(N)$ that takes into account the complexity (size) of the model M. Assuming a Gaussian process, the models $\{M\}$ among which the algorithm has to select the best one, i.e., the one with the highest BIC value, differ in either the number of Gaussians or the way the input stream is partitioned by the Gaussians.

Let $\mathcal{G}_k$ be the set of models with k Gaussians. Since the number of free parameters of a d-dimensional Gaussian is $F = d + (d(d+1)/2)$, the penalty term, for each $M \in \mathcal{G}_k$, is

$$P_M = P(k,N) = \frac{F}{2}k\log(N), \quad M \in \mathcal{G}_k.$$

This means that all $M \in \mathcal{G}_k$ have the same penalty term. Then, the best way of segmenting the input stream in k homogeneous segments is given by the model M_k such that

$$M_k = \arg\max_{M\in\mathcal{G}_k}\text{BIC}(M \mid \mathcal{O}) = \arg\max_{M\in\mathcal{G}_k}\log L_M(\mathcal{O}).$$

The algorithm for searching the optimum BIC segmentation builds the set $\mathscr{M} = \{M_k : k = 1,\ldots,K\}$, with $K \leqslant N$, of the optimum ways of segmenting the input stream in k segments, for all possible ks.

Let $V_{k,t}$ be the following matrix for the DP

$$V_{k,t} := \max\{\log L_M(o_1,\ldots,o_t) : M \in \mathcal{G}_k\}, \qquad k = 1,\ldots,K, \quad t = 1,\ldots,N.$$

The maximum number K of segments for each t can be defined as $K = \lfloor t/S_{\min}\rfloor$, where $S_{\min}$ is the minimum allowed duration (size) of a segment whatever the approach we are using for the segmentation. The matrix $V_{k,t}$ is filled column by column by means of the following equations:

$$V_{1,t} = A(o_1,\ldots,o_t), \tag{16}$$

$$V_{k,t} = \max_{(k-1)S_{\min} \leqslant t' \leqslant t-S_{\min}} (V_{k-1,t'} + A(o_{t'+1},\ldots,o_t)), \quad k = 2,\ldots,K, \tag{17}$$

where $A(o_a,\dots,o_b)$ denotes the *auto-consistency* of $o_a,\dots,o_b$, that is the log-likelihood of G_a^b on $o_a,\dots,o_b$, where G_a^b is the Gaussian estimated on $o_a,\dots,o_b$. Note that the range of t' is a function of $S_{\min}$. The main role of the parameter $S_{\min}$ is to ensure smoothed differences between covariance matrices, and then auto-consistencies, of segments that differ for only few observations. Indeed, if we allowed segments of a single observation, the resulting segmentation would strongly depend on noise. However, in the context of our DP approach to segmentation, a careful choice of the parameter $S_{\min}$ can also help in reducing the computational requirements of the algorithm. This issue will be analyzed in Section 5.5.2.

The best segmentation with k segments of the input up to time t is obtained by searching, through Eq. (17), the best segmentation in $k-1$ segments up to $t' < t$, and adding to it the kth segment containing the observations from $t'+1$ to t.

In addition to $V_{k,t}$, another matrix $M_{k,t}$ is needed for recording the time indexes of change:

$$M_{1,t} = 0,$$
$$M_{k,t} = \arg\max_{(k-1)S_{\min} \leqslant t' \leqslant t-S_{\min}} (V_{k-1,t'} + A(o_{t'+1},\dots,o_t)), \quad k = 2,\dots,K.$$

The two matrices $V_{k,t}$ and $M_{k,t}$ have the same structure, and their entries store, respectively, the log-likelihood of the best segmentation in k segments of the input up to the index t, and the time index of the last spectral change.

When the end of the input is reached, each entry of the last column of $V_{k,t}$ contains the log-likelihood of the best segmentation of the whole input stream in k segments. By subtracting the penalty term, it is possible to obtain k_{opt}, the number of segments of the optimum segmentation

$$k_{\text{opt}} = \arg\max_{k=1,\dots,K} (V_{k,N} - \lambda P(k,N)) \quad \text{with } K = \lfloor N/S_{\min} \rfloor.$$

The optimum segmentation is finally obtained by backtracking over the matrix $M_{k,t}$, starting from the entry $M_{k_{\text{opt}},N}$ and going back to the previous rows, at the columns (time indexes of changes) specified in the entries.

5.1. Efficient computation of the auto-consistency

The algorithm requires the computation of the auto-consistency of $o_{t'+1},\dots,o_t$, which by definition can be computed by

$$A(o_{t'+1},\dots,o_t) = -\frac{t-t'}{2}\Big(\log\left|\Sigma_{t'+1}^t\right| + d\log 2\pi + d\Big). \tag{18}$$

An efficient way of computing the covariance matrix $\Sigma_{t'+1}^t$ for all possible indexes t and t' is the CSA method described in Section 4.1.3. This method allows the computation of $\Sigma_{t'+1}^t$ with a number of operations equals to $2d(d+1)+2d$ (see Table 2), by exploiting the encoding of the input signal with cumulative statistics.

5.2. Bounding the auto-consistency

Computing $A(o_a,\dots,o_b)$ is costly since it requires the estimation of a covariance matrix and the computation of its determinant (see Eq. (18)). However, in some cases it is possible to avoid those computations. Let $B(o_a,\dots,o_b)$ be an upper bound on $A(o_a,\dots,o_b)$, that is

$$A(o_a, \ldots, o_b) \leqslant B(o_a, \ldots, o_b) \quad \forall a, b;\ a \leqslant b, \tag{19}$$

where we assume that the computation of $B()$ is cheaper than that of $A()$. If during the computation of $V_{k,t}$, it happens that for a given t'

$$V_{k,t} \geqslant V_{k-1,t'} + B(o_{t'+1}, ..., o_t) \quad \forall k,$$

then, given Eq. (19), we have

$$V_{k,t} \geqslant V_{k-1,t'} + A(o_{t'+1}, \ldots, o_t) \quad \forall k.$$

This means that it is not convenient to hypothesize a change in t', whatever the number of segments k; therefore, for that t', the computation of $A(o_{t'+1}, \ldots, o_t)$ can be avoided.

The problem is now to define such a bound $B()$. Let μ_a^b and Σ_a^b be the parameters of the Gaussian G_a^b estimated on the d-dimensional observations $o_a, \ldots, o_b$ with $a \leqslant b$ and $n = b - a + 1$, and let assume that $A(o_a, \ldots, o_b)$ has already been computed. Let us suppose that our goal is to obtain the bound $B(o_c, \ldots, o_{a-1}, o_a, \ldots, o_b)$ for $A(o_c, \ldots, o_{a-1}, o_a, \ldots, o_b)$. It can be shown that $A(\mu_a^b, \ldots, \mu_a^b, o_a, \ldots, o_b) \leqslant A(o_c, \ldots, o_{a-1}, o_a, \ldots, o_b)$; then we can set the bound $B()$ equal to the auto-consistency of the sequence where the new observations $o_c, \ldots, o_{a-1}$ have been substituted with the mean of the old sub-sequence. Moreover, such a bound can be efficiently computed by

$$\begin{aligned} B(\underbrace{o_c, \ldots, o_{a-1}}_{m}, \underbrace{o_a, \ldots, o_b}_{n}) &= A(\mu_a^b, \ldots, \mu_a^b, o_a, \ldots, o_b) \\ &= \frac{n+m}{n} A(o_a, \ldots, o_b) - \frac{d(n+m)}{2} \log\left(\frac{n}{n+m}\right) \end{aligned}$$

based on the the value of the previous auto-consistency and the number $m = a - c$ of the new included observations.

5.3. Further reductions of the algorithm cost

The complexity of the DP algorithm can be further reduced by introducing some reasonable and quite obvious approximations listed below.

$K_{\max}$, the maximum number of searched segments. It bounds the number of rows of the DP matrices V and M, with a number that is independent from the input audio size N.

$S_{\max}$, the maximum size of a segment. Actually, this approximation allows to greatly reduce the number of operations of the algorithm, but it can be too dangerous, unless $S_{\max}$ is set to a very high value, thus losing the benefit of its use. Then, to keep the advantage without losing accuracy, the possibility of hypothesizing a segment larger than $S_{\max}$ is kept in few specific circumstances.

Resolution δ, for encoding the audio stream. Like in the local algorithm described in Section 4, δ establishes the resolution of the computation of triples $(\mathrm{SQ}_1^t, \mathrm{SV}_1^t, t)$ and of the entries of matrices $V_{k,t}, M_{k,t}$; that is, those values are computed only for $t = \delta, 2\delta, 3\delta, \ldots, N$. The resolution δ reduces the total cost of the algorithm of a factor which is a function of the square of δ.

5.4. Computational costs

The main stages of the DP algorithm without any approximation have the following costs, including the $d^3/6$ operations for the determinant evaluation of the covariance matrices:

- input stream encoding: $N(d(d+1)+d)$,
- matrices initialization: $\mathrm{O}\{N\cdot(d^3/6+2d(d+1)+2d)\}$,
- matrices filling: $\mathrm{O}\{N^2/2\cdot(N/S_{\min}+d^3/6+2d(d+1)+2d)\}$,
- selection of the winner model: $\mathrm{O}\{N/S_{\min}\}$,

where the most expensive step is obviously the filling of the DP matrices. The overall complexity of the algorithm is then $\mathrm{O}\{N^3/S_{\min}+N^2d^3\}$.

By introducing the approximations described in Section 5.3, the step costs become:

- input stream encoding: $N(d(d+1)+d)$,
- matrices initialization: $\mathrm{O}\{(N/\delta)\cdot(d^3/6+2d(d+1)+2d)\}$,
- matrices filling: $\mathrm{O}\{(NS_{\max}/\delta^2)\cdot(K_{\max}+d^3/6+2d(d+1)+2d)\}$,
- selection of the winner model: $\mathrm{O}\{K_{\max}\}$.

Also in this case, the DP step is the most expensive. The complexity of the algorithm is $\mathrm{O}\{(NS_{\max}/\delta^2)\cdot(K_{\max}+d^3)\}$.

The benefit given by the bounding $B()$ introduced in Section 5.2 has been experimentally quantified: depending on the value of $S_{\max}$, it allowed 25–36% reduction of the execution time.

5.5. Experimental evaluation

5.5.1. Database

The main goal of these experiments is to compare the segmentation accuracy of the global and local algorithms. Since it is known that short segments are not well handled by the local algorithm, the NIST data are suitable to make evident the gain, if any, given by the global algorithm over the local one. The evaluation has mainly been done on the NIST test set described in Section 3.2. Nevertheless, the global algorithm has also been tested on the IBNC test set (Section 3.1), and compared with the local algorithm performance given in Section 4.2.2.

5.5.2. Results

First of all, the global algorithm has been compared with the CSA implementation of the local one (see Section 4.1.3) on the NIST test set. The set-ups of the two algorithms are reported in Tables 5 and 6.

Table 7 shows both performance and execution times [1] measured for the two algorithms. The improvement in terms of F-score given by the global algorithm is 2.4% relative and the cost is 38 times higher than the local algorithm. It is worth noticing the benefit in running time (36%) given by the use of the bound $B()$ (Section 5.2).

We now present the results of the experiments carried out to measure the impact of the approximations introduced in the exact DP algorithm.

Figs. 5–8 plot F-score and execution time as functions of $S_{\min}$, $K_{\max}$, $S_{\max}$ and δ, respectively – each computed keeping fixed all the other parameters of the algorithm. In the following considerations, the size of the parameters are sometimes in seconds, and not in number of observations as usual: in these experiments 1 s of audio is represented by 100 observation vectors.

[1] Experiments were performed on a Pentium III 1 GHz, 1 Gb RAM.

Table 5
Set-up of the global algorithm for the segmentation of the NIST data

$S_{\min} = 75$	$S_{\max} = 1500$	$K_{\max} = 80$
$\delta = 5$	$\lambda = 0.60$	

Table 6
Set-up of the local (CSA) algorithm for the segmentation of the NIST data

$N_{\min} = 200$	$\Delta N_{\text{grow}} = 50$	$\delta_{\text{l}} = 25$
$N_{\max} = 500$	$\Delta N_{\text{shift}} = 100$	$\delta_{\text{h}} = 5$
$N_{\text{second}} = 400$	$N_{\text{margin}} = 50$	$\lambda = 0.85$

Table 7
Performance of the local and global algorithms on the NIST test set (0.3 s is the tolerance admitted in the change detection)

	Performance			Execution time (s)	
	Precision	Recall	*F*-score		Without *B*()
Local	54.4	75.7	63.3	367.7	–
Global	54.7	79.7	64.8	13,930.5	21,796.8

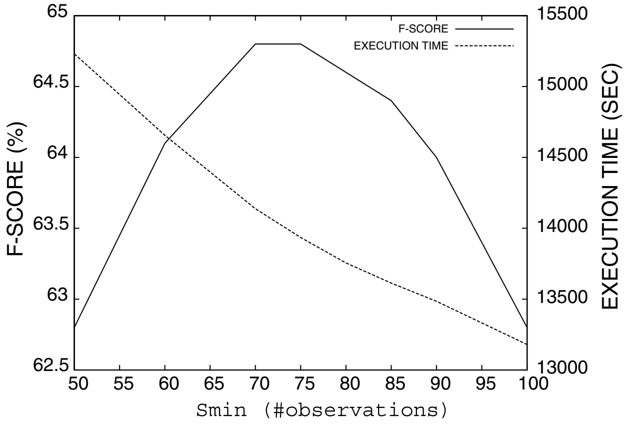

Fig. 5. *F*-score and execution time as functions of $S_{\min}$.

$S_{\min}$: It results that the value of the minimum window is critical, and the best value is not equal to the minimum size of test segments as expected (half a second, after the merging stage mentioned in Section 3.2), but is slightly larger (0.75 s). Perhaps, that value could be – somehow – related to the tolerance used in the automatic evaluation procedure (0.3 s).

$K_{\max}$: The limit on the number of searched segments also affects accuracy when its value is lower than a threshold (about 50). Since the average length of segments in the test set is 1.3 s, and the audio files lasted about 1 min, the expected number of segments in each file is about 45, which is

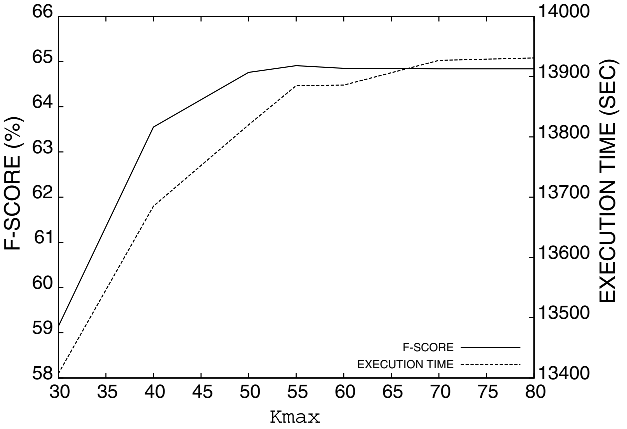

Fig. 6. F-score and execution time as functions of $K_{\max}$.

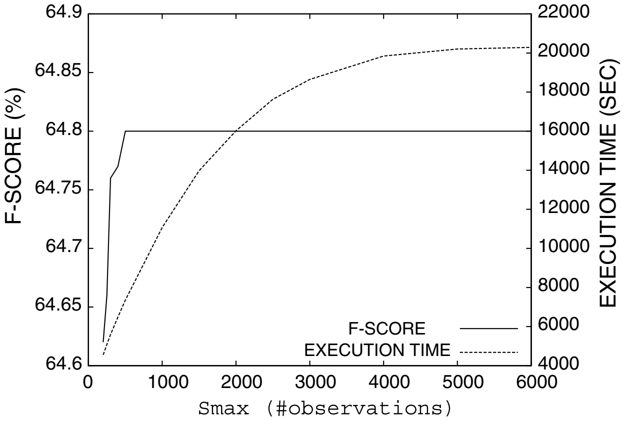

Fig. 7. F-score and execution time as functions of $S_{\max}$.

compatible with the threshold observed in the experiments. This means that $K_{\max}$ must be carefully chosen, taking into account the characteristics of audio under processing.

$S_{\max}$: It can be noted that accuracy does not change if $S_{\max}$ is reduced from 60 s down to 10 s, halving the execution time. Below 10 s, the number of insertions increases considerably, affecting the overall accuracy. The nature of the test set (only 0.1% of segments is longer than 10 s) allows to reduce $S_{\max}$ down to 10 s, but experiments not reported here on a different test set (news programs) show the importance of introducing some corrections in order to relax, at least in some circumstances, the $S_{\max}$ constraint.

Resolution δ: The use of a δ slightly higher than 1 for the processing allows to considerably reduce the amount of time required by the algorithm. Furthermore, $\delta = 5$ results in a better performance than the maximum resolution, which causes more false alarms.

It is worth pointing out that if $S_{\min}$, $K_{\max}$, $S_{\max}$ and δ are set to proper values, their use does not introduce any degradation of the approximated DP algorithm with respect to the exact one.

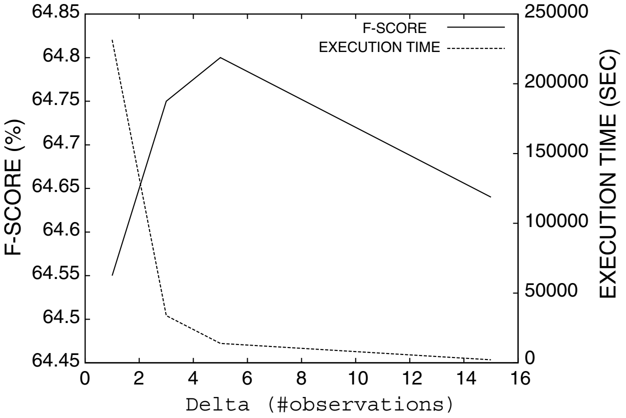

Fig. 8. *F*-score and execution time as functions of δ.

Table 8
Performance of the local (using both diagonal and full covariance matrices) and global algorithms on broadcast news programs (tolerance = 0.5 s)

	Performance			Execution time (s)	
	Precision	Recall	*F*-score		Without $B()$
Local (diagonal Σ)	82.5	84.4	83.5	16.3	–
Local	89.2	89.6	89.4	42.2	–
Global	92.9	86.8	89.8	4021.3	4771.7

For completeness, the global algorithm and the CSA implementation of the local one have also been compared on the IBNC test set. Results are shown in Table 8 (rows `local` and `global`). In this case, the benefit of the global search over the local one is negligible, as it was expected given the audio characteristics of broadcast news recordings. The first row of the table refers to the CSA implementation of the local search with the use of diagonal covariance matrices, instead of full matrices as in all the experiments reported so far. The rationale behind this trial is that it is generally recognized that Cepstral coefficients are reasonably uncorrelated and can be satisfactorily modeled with diagonal-covariance Gaussians, which dramatically reduce computational requirements of our algorithms. Unfortunately, the impressive reduction in time costs (16.3 s vs. 42.2) is paid with a 6.6% relative *F*-score decrease.

6. Summary

In this work, three different approaches to the implementation of the well-known local BIC-based audio segmentation algorithm have been beforehand analyzed: (i) a simple method that uses only a sum and a square sum of the input vectors, in order to save computations in estimating covariance matrices on partially shared data; (ii) the approach proposed in Sivakumaran et al. (2001), that encodes the input signal with cumulative distributions; (iii) an innovative approach that encodes the input signal in cumulative pair of sums. The two latter approaches provide a better use of the typical approximation made in the algorithm, which is the computation of ΔBIC values not on all observations, but at a lower resolution.

The three approaches have been compared both theoretically and experimentally: the proposed new approach has turned out to be the most efficient.

The local algorithm is heuristic, hence the quality of its solutions is only a lower bound on the quality of the solutions achievable by the BIC criterion. In order to discover if there is any room of improvement within or outside the heuristics, we have developed a DP algorithm that, within the BIC model, finds a globally optimal segmentation of the input audio stream. In the paper, it is described, analyzed, and experimentally compared with the local BIC-based algorithm.

The global algorithm yields a small but consistent improvement of performance, with respect to the local one, while its time cost is definitely higher. Its computational complexity was reduced without affecting accuracy, by introducing some reasonable approximations, yielding a less than real time cost.

Summarizing, results show that no much further improvement is possible under the BIC criterion, fact that is important to be aware of. On the other side, experiments make evident that the local algorithm is able to segment an audio stream almost as well as the global algorithm is. That is, the sliding window approach is effective as an heuristics towards the BIC criterion objective function. This encouraged us in proposing improvements to the implementation of the local search, providing a benchmark for possible future experimental work on segmentation.

Acknowledgements

The authors are grateful to Marcello Federico for his help in the mathematical formalization of the segmentation problem in terms of model selection. The authors also thank Helmer Strik, Alberto Pedrotti, and the anonymous referees for their suggestions on how to improve the original version of the manuscript.

References

Akaike, H., 1977. On entropy maximization principle. In: Krishnaiah, P.R. (Ed.), Applications of Statistics. North-Holland, Amsterdam, Netherlands, pp. 27–41.

Baxter, R.A., 1996. Minimum message length inference: theory and applications. Ph.D. Thesis, Department of Computer Science Monash University, Clayton, Victoria, Australia.

Cettolo, M., Federico, M., 2000. Model selection criteria for acoustic segmentation. In: Proceedings of the ISCA Automatic Speech Recognition Workshop, Paris, France.

Cettolo, M., Vescovi, M., 2003. Efficient audio segmentation algorithms based on the BIC. In: Proceedings of the ICASSP, vol. VI, Hong Kong, pp. 537–540.

Cettolo, M., 2000. Segmentation, classification and clustering of an Italian broadcast news corpus. In: Proceedings of the Sixth RIAO – Content-Based Multimedia Information Access – Conference, Paris, France.

Chen, S.S., Gopalakrishnan, P.S., 1998. Speaker, environment and channel change detection and clustering via the Bayesian Information Criterion. In: Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop, Lansdowne, VA.

Delacourt, P., Kryze, D., Wellekens, C., 1999. Speaker-based segmentation for audio data indexing. In: Proceedings of the ESCA ETRW Workshop Accessing Information in Spoken Audio, Cambridge, UK.

Federico, M., Giordani, D., Coletti, P., 2000. Development and evaluation of an Italian broadcast news corpus. In: Proceedings of the Second International Conference on Language Resources and Evaluation (LREC), Athens, Greece.

Frakes, W.B., Baeza-Yates, R., 1992. Information Retrieval: Data Structures and Algorithms. Prenctice-Hall, Englewood Cliffs, NJ.

Gauvain, J.-L., Lamel, L., Adda, G., 1998. Partitioning and transcription of broadcast news data. In: Proceedings of the ICSLP, Sidney, Australia, pp. 1335–1338.

Gish, H., Siu, M., Rohlicek, R., 1991. Segregation of speakers for speech recognition and speaker identification. In: Proceedings of the ICASSP, vol. II, Toronto, Canada, pp. 873–876.

Hain, T., Johnson, S.E., Tuerk, A., Woodland, P.C., Young, S.J., 1998. Segment generation and clustering in the HTK broadcast news transcription system. In: Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop, Lansdowne, VA.

Harris, M., Aubert, X., Haeb-Umbach, R., Beyerlein, P., 1999. A study of broadcast news audio stream segmentation and segment clustering. In: Proceedings of the EUROSPEECH, vol. III, Budapest, Hungary, pp. 1027–1030.

IBNC, 2000. Available from: <http://www.elda.fr/catalogue/en/speech/S0093.html>.

Kemp, T., Schmidt, M., Westphal, M., Waibel, A., 2000. Strategies for automatic segmentation of audio data. In: Proceedings of the ICASSP, vol. III, Istanbul, Turkey, pp. 1423–1426.

Lu, L., Li, S.Z., Zhang, H.-J., 2001. Content-based audio segmentation using support vector machines. In: Proceedings of the ICME, Tokyo, Japan, pp. 956–959.

NIST, 2000. Available from: <www.nist.gov/speech/tests/spk/2000/>.

Scheirer, E., Slaney, M., 1997. Construction and evaluation of a robust multifeature speech/music discriminator. In: Proceedings of the ICASSP, Munich, Germany, pp. 1331–1334.

Schwarz, G., 1978. Estimating the dimension of a model. The Annals of Statistics 6 (2), 461–464.

Seber, G.A.F., 1984. Multivariate Observations. John Wiley & Sons, New York, NY.

Siegler, M.A., Jain, U., Raj, B., Stern, R.M., 1997. Automatic segmentation, classification and clustering of broadcast news audio. In: Proceedings of the DARPA Speech Recognition Workshop, Chantilly, VA.

Sivakumaran, P., Fortuna, J., Ariyaeeinia, A.M., 2001. On the use of the Bayesian Information Criterion in multiple speaker detection. In: Proceedings of the EUROSPEECH, vol. II, Aalborg, Denmark, pp. 795–798.

Tritschler, A., Gopinath, R., 1999. Improved speaker segmentation and segments clustering using the Bayesian Information Criterion. In: Proceedings of the EUROSPEECH, vol. II, Budapest, Hungary, pp. 679–682.

Vescovi, M., Cettolo, M., Rizzi, R., 2003. A DP algorithm for speaker change detection. In: Proceedings of the EUROSPEECH, vol. IV, Geneva, Switzerland, pp. 2997–3000.

Wegmann, S., Zhan, P., Gillick, L., 1999. Progress in broadcast news transcription at Dragon systems. In: Proceedings of the ICASSP, Phoenix, AZ, pp. 33–36.

Wellekens, C., 2001. Seamless navigation in audio files. In: Proceedings of the ITRW on Speaker Recognition, Crete, Greece, pp. 9–12.