

Australian/New Zealand Standard™

**Information technology—Multimedia  
content description interface**

**Part 4: Audio**



**Standards Australia**



Document Sharing Hub

## **AS/NZS 15938.4:2003**

This Joint Australian/New Zealand Standard was prepared by Joint Technical Committee IT-001, Information Systems—Interconnection. It was approved on behalf of the Council of Standards Australia on 22 April 2003 and on behalf of the Council of Standards New Zealand on 22 April 2003. It was published on 2 June 2003.

---

The following are represented on Committee IT-001:

Australian Bureau of Statistics  
Australian Communications Industry Forum  
Australian Information Industry Association  
Australian Telecommunications Users Group  
Australian Vice-Chancellors Committee  
Electrical Compliance Testing Association  
Information Technology Association of New Zealand

---

### **Keeping Standards up-to-date**

Standards are living documents which reflect progress in science, technology and systems. To maintain their currency, all Standards are periodically reviewed, and new editions are published. Between editions, amendments may be issued. Standards may also be withdrawn. It is important that readers assure themselves they are using a current Standard, which should include any amendments which may have been published since the Standard was purchased.

Detailed information about joint Australian/New Zealand Standards can be found by visiting the Standards Australia web site at [www.standards.com.au](http://www.standards.com.au) or Standards New Zealand web site at [www.standards.co.nz](http://www.standards.co.nz) and looking up the relevant Standard in the on-line catalogue.

Alternatively, both organizations publish an annual printed Catalogue with full details of all current Standards. For more frequent listings or notification of revisions, amendments and withdrawals, Standards Australia and Standards New Zealand offer a number of update options. For information about these services, users should contact their respective national Standards organization.

We also welcome suggestions for improvement in our Standards, and especially encourage readers to notify us immediately of any apparent inaccuracies or ambiguities. Please address your comments to the Chief Executive of either Standards Australia International or Standards New Zealand at the address shown on the back cover.

---

*This Standard was issued in draft form for comment as DR 03037.*

---

# Australian/New Zealand Standard™

## Information technology—Multimedia content description interface

### Part 4: Audio

First published as AS/NZS 15938.4:2003.

#### **COPYRIGHT**

© Standards Australia/Standards New Zealand

All rights are reserved. No part of this work may be reproduced or copied in any form or by any means, electronic or mechanical, including photocopying, without the written permission of the publisher.

Jointly published by Standards Australia International Ltd, GPO Box 5420, Sydney, NSW 2001  
and Standards New Zealand, Private Bag 2439, Wellington 6020

ISBN 0 7337 5252 7

## PREFACE

This Standard was prepared by the Joint Standards Australia/Standards New Zealand Committee IT-001, Information Systems—Interconnection. This Standard is identical with and has been reproduced from ISO/IEC 15938-4:2002, *Information technology—Multimedia content description interface*, Part 4: *Audio*.

The objective of this Standard is to provide a standardized set of technologies for describing multimedia content. It addresses a broad spectrum of multimedia applications and requirements by providing a metadata system for describing the features of multimedia content.

This Standard is Part 4 of AS/NZS 15938, *Information technology—Multimedia content description interface*, which, when complete, will consist of the following:

Part 1: Systems

Part 2: Description definition language

Part 3: Visual

Part 4: Audio (this Standard)

Part 5: Multimedia description schemes

Part 6: Reference software

Part 7: Conformance testing

Part 8: Extraction and use of MPEG-7 descriptions

The term ‘informative’ has been used in this Standard to define the application of the annex to which it applies. An ‘informative’ annex is only for information and guidance.

As this Standard is reproduced from an International Standard, the following applies:

- (a) Its number does not appear on each page of text and its identity is shown only on the cover and title page.
- (b) In the source text ‘this part of ISO/IEC 15938’ should read ‘this part of AS/NZS 15938’.
- (c) A full point should be substituted for a comma when referring to a decimal marker.

## CONTENTS

|          | <i>Page</i>                                           |
|----------|-------------------------------------------------------|
| <b>1</b> | <b>Scope ..... 1</b>                                  |
| 1.1      | Definition of Scope ..... 1                           |
| 1.2      | Fields of application ..... 1                         |
| <b>2</b> | <b>Terms and definitions ..... 2</b>                  |
| <b>3</b> | <b>Symbols and abbreviated terms ..... 2</b>          |
| <b>4</b> | <b>Conventions ..... 3</b>                            |
| 4.1      | Description Definition Language ..... 3               |
| 4.2      | Audio representation ..... 3                          |
| <b>5</b> | <b>Audio Framework ..... 4</b>                        |
| 5.1      | Introduction ..... 4                                  |
| 5.2      | Scalable Series ..... 4                               |
| 5.2.1    | Introduction ..... 4                                  |
| 5.2.2    | ScalableSeriesType ..... 5                            |
| 5.2.3    | SeriesOfScalarType ..... 6                            |
| 5.2.4    | SeriesOfScalarBinaryType ..... 9                      |
| 5.2.5    | SeriesOfVectorType ..... 10                           |
| 5.2.6    | SeriesOfVectorBinaryType ..... 13                     |
| 5.3      | Low level Audio Descriptors ..... 13                  |
| 5.3.1    | Introduction ..... 13                                 |
| 5.3.2    | AudioLLDScalarType ..... 14                           |
| 5.3.3    | AudioLLDVectorType ..... 15                           |
| 5.3.4    | AudioWaveformType ..... 16                            |
| 5.3.5    | AudioPowerType ..... 17                               |
| 5.3.6    | Audio Spectrum Descriptors ..... 17                   |
| 5.3.7    | AudioSpectrumEnvelopeType ..... 18                    |
| 5.3.8    | AudioSpectrumCentroidType ..... 21                    |
| 5.3.9    | AudioSpectrumSpreadType ..... 23                      |
| 5.3.10   | AudioSpectrumFlatnessType ..... 24                    |
| 5.3.11   | AudioSpectrumBasisType ..... 26                       |
| 5.3.12   | AudioSpectrumProjectionType ..... 29                  |
| 5.3.13   | AudioHarmonicityType ..... 33                         |
| 5.3.14   | Timbre Descriptors ..... 36                           |
| 5.3.15   | LogAttackTimeType ..... 38                            |
| 5.3.16   | HarmonicSpectralCentroidType ..... 39                 |
| 5.3.17   | HarmonicSpectralDeviationType ..... 41                |
| 5.3.18   | HarmonicSpectralSpreadType ..... 42                   |
| 5.3.19   | HarmonicSpectralVariationType ..... 44                |
| 5.3.20   | SpectralCentroidType ..... 45                         |
| 5.3.21   | TemporalCentroidType ..... 46                         |
| 5.4      | Silence ..... 46                                      |
| 5.4.1    | Introduction ..... 46                                 |
| 5.4.2    | SilenceHeaderType ..... 47                            |
| 5.4.3    | SilenceType ..... 47                                  |
| 5.4.4    | Usage, examples and extraction (informative) ..... 48 |
| <b>6</b> | <b>High Level Tools ..... 49</b>                      |
| 6.1      | Introduction ..... 49                                 |
| 6.2      | Audio Signature ..... 49                              |

|                              |                                                                            |            |
|------------------------------|----------------------------------------------------------------------------|------------|
| 6.2.1                        | Introduction .....                                                         | 49         |
| 6.2.2                        | AudioSignatureType .....                                                   | 50         |
| 6.2.3                        | Instantiation requirements .....                                           | 50         |
| 6.2.4                        | Usage and examples (informative) .....                                     | 50         |
| 6.3                          | Timbre .....                                                               | 51         |
| 6.3.1                        | Introduction .....                                                         | 51         |
| 6.3.2                        | InstrumentTimbreType .....                                                 | 52         |
| 6.3.3                        | HarmonicInstrumentTimbreType .....                                         | 53         |
| 6.3.4                        | PercussiveInstrumentTimbreType .....                                       | 54         |
| 6.3.5                        | Usage, extraction and examples (informative) .....                         | 55         |
| 6.4                          | General Sound Recognition and Indexing .....                               | 56         |
| 6.4.1                        | Introduction .....                                                         | 56         |
| 6.4.2                        | SoundModelType .....                                                       | 57         |
| 6.4.3                        | SoundClassificationModelType .....                                         | 59         |
| 6.4.4                        | SoundModelStatePathType .....                                              | 61         |
| 6.4.5                        | SoundModelStateHistogramType .....                                         | 62         |
| 6.4.6                        | General Sound Classification and Indexing Applications (informative) ..... | 64         |
| 6.5                          | Spoken Content .....                                                       | 66         |
| 6.5.1                        | Introduction .....                                                         | 66         |
| 6.5.2                        | SpokenContentHeaderType .....                                              | 67         |
| 6.5.3                        | SpeakerInfoType .....                                                      | 68         |
| 6.5.4                        | SpokenContentIndexEntryType .....                                          | 71         |
| 6.5.5                        | ConfusionCountType .....                                                   | 71         |
| 6.5.6                        | WordType, PhoneType, WordLexiconIndexType and PhoneLexiconIndexType .....  | 73         |
| 6.5.7                        | LexiconType .....                                                          | 74         |
| 6.5.8                        | WordLexiconType .....                                                      | 74         |
| 6.5.9                        | phoneticAlphabetType .....                                                 | 75         |
| 6.5.10                       | PhoneLexiconType .....                                                     | 75         |
| 6.5.11                       | SpokenContentLatticeType .....                                             | 76         |
| 6.5.12                       | SpokenContentLinkType .....                                                | 78         |
| 6.5.13                       | Usage, extraction and examples (informative) .....                         | 79         |
| 6.6                          | Melody .....                                                               | 84         |
| 6.6.1                        | Introduction .....                                                         | 84         |
| 6.6.2                        | MelodyType .....                                                           | 84         |
| 6.6.3                        | Meter .....                                                                | 85         |
| 6.6.4                        | scaleType .....                                                            | 86         |
| 6.6.5                        | MelodyKey .....                                                            | 86         |
| 6.6.6                        | MelodyContourType .....                                                    | 88         |
| 6.6.7                        | contourType .....                                                          | 88         |
| 6.6.8                        | beatType .....                                                             | 89         |
| 6.6.9                        | MelodySequence .....                                                       | 90         |
| 6.6.10                       | Usage of MelodyContour (informative) .....                                 | 92         |
| 6.6.11                       | Usage of MelodySequence (informative) .....                                | 94         |
| 6.6.12                       | Examples (informative) .....                                               | 94         |
| <b>Annex A (informative)</b> | <b>Usage, extraction and examples of Scalable Series .....</b>             | <b>96</b>  |
| <b>Annex B (informative)</b> | <b>Patent statements .....</b>                                             | <b>105</b> |

## AUSTRALIAN/NEW ZEALAND STANDARD

**Information technology — Multimedia content description interface —****Part 4:  
Audio****1 Scope****1.1 Definition of Scope**

This International Standard defines a Multimedia Content Description Interface, specifying a series of interfaces from system to application level to allow disparate systems to interchange information about multimedia content. It describes the architecture for systems, a language for extensions and specific applications, description tools in the audio and visual domains, as well as tools that are not specific to audio-visual domains. As a whole, this International Standard encompassing all of the aforementioned components is known as “MPEG-7.” MPEG-7 is divided into eight parts (as defined in the Foreword).

This part of the MPEG-7 Standard (Part 4: Audio) specifies description tools that pertain to multimedia in the audio domain. See below for further details of application.

This part of the MPEG-7 Standard is intended to be implemented in conjunction with other parts of the standard. In particular, MPEG-7 Part 4: Audio assumes knowledge of Part 2: Description Definition Language (DDL) in its normative syntactic definitions of Descriptors and Description Schemes. This part of the standard also has dependencies upon clauses in Part 5: Multimedia Description Schemes, namely many of the fundamental Description Schemes that extend the basic type capabilities of the DDL.

MPEG-7 is an extensible standard. The method to extend the standard beyond the Description Schemes provided in the standard is to define new ones in the DDL, and to make those DSs available with the instantiated descriptions. Further details are available in Part 2. To avoid duplicate functionality with other parts of the standard, the DDL is the only extension facility provided.

**1.2 Fields of application**

MPEG-7 Part 4: Audio is applicable to all forms of audio content. The encoding format or medium of the said audio is not limited in any way, and may include audio held in an analogue medium such as magnetic tape or optical film. The content of the audio is not limited within or without music, speech, sound effects, soundtracks, or any mixtures thereof.

The tools listed in this part of the International Standard are applicable to both audio in isolation and to audio associated with video.

The specific tools provided within the Audio portion of the standard are designed to work in conjunction with the Multimedia Description Schemes that apply to both audio and video. Because of the “toolbox” nature of the standard, the most appropriate tools from the different parts of the standard may be mixed, within the constraints of the DDL.

The MPEG-7 Audio tools are applicable to two general areas: low-level audio description, in the case of the Audio Framework (clause 5), and application-driven description, in the case of the High Level Tools (clause 6).

The Audio Framework tools are applicable to general audio, without regard to the specific content carried by the encoded signal. The Scalable Series provides general capabilities for multi-level sampled data. The Audio Description Framework defines specific descriptors for use with the Scalable Series or with Audio Segments, which has properties inherited from the general Segment described in the Multimedia Description Schemes part of the standard. The Silence Descriptor works with the Segment descriptor, and is applicable across all possible audio signals.

The high level description tools are applicable to specific types of content within audio. The specific domains are well documented within the introduction to each sub-clause. The audio domains encompassed by the various MPEG-7 Audio tools are speech, sound effects, musical instruments, melodies within music and general audio recognition. These specialised tools may be employed in conjunction with the other tools within the standard.

## 2 Terms and definitions

For the purposes of this part of ISO/IEC 15938, the following terms and definitions apply.

### 2.1

#### Frame

A Frame is defined as a short period of time of the signal on which the instantaneous analysis is performed. For a signal, noted  $s(t)$  (in continuous time noted  $t$ ), and for an analysis window of type hamming, noted  $h(t)$  and of temporal length  $L$ , the  $f^{\text{th}}$  signal frame is defined as

$$x(f, t) = s(t) \times h(t - f \times S)$$

where  $S$  is the hop size

### 2.2

#### Hop size

The hop size defines the temporal distance between two successive analyses

### 2.3

#### Running window analysis

A running window analysis is an analysis obtained by multiplying the signal by a window function which is shifted along time by integer multiple of a parameter called the hop size. For a window function  $h(t)$ , and a hop size  $S$ , the  $f^{\text{th}}$  shifting of the window is equal to  $h(t - fS)$ .

### 2.4

#### Instantaneous values

The instantaneous value of a (Timbre) descriptor based peak estimation is defined to be the result of analysis on a frame level. The global value of a (Timbre) descriptor based on peak estimation is defined to be the average over all frames of the segment of the instantaneous value.

## 3 Symbols and abbreviated terms

|       |                                 |
|-------|---------------------------------|
| — ASR | Automatic Speech Recognition    |
| — CPU | Central Processing Unit         |
| — D   | Descriptor                      |
| — DC  | Direct Current (0 Hz)           |
| — DDL | Description Definition Language |
| — DFT | Discrete Fourier Transform      |
| — DS  | Description Scheme              |

|        |                                                                                                      |
|--------|------------------------------------------------------------------------------------------------------|
| — FFT  | Fast Fourier Transform                                                                               |
| — HMM  | Hidden Markov Model                                                                                  |
| — Hz   | Hertz, frequency in cycles per second                                                                |
| — LLD  | Low Level Descriptor                                                                                 |
| — log  | Logarithm (unspecified base)                                                                         |
| — LPC  | Linear Predictive Coding                                                                             |
| — MSD  | Maximum Squared Distance (from the mean)                                                             |
| — OOV  | Out of Vocabulary, describing a word that is not in the vocabulary of an automatic speech recogniser |
| — RMS  | Root Mean Square                                                                                     |
| — SR   | Sample Rate                                                                                          |
| — STFT | Short Time Fourier Transform                                                                         |
| — XML  | Extensible Markup Language                                                                           |

## 4 Conventions

### 4.1 Description Definition Language

All DDL in this document is defined in a single namespace. The schema wrapper is assumed to begin

```
<schema targetNamespace="urn:mpeg:mpeg7:schema:2001"
  xmlns:mpeg7="urn:mpeg:mpeg7:schema:2001"
  xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:xml="http://www.w3.org/XML/1998/namespace"
  elementFormDefault="qualified" attributeFormDefault="unqualified">
```

and end

```
</schema>
```

Under this definition, the default namespace in a schema definition document is specified as XML Schema and thus a prefix `xsd:` is not needed. Instead, references to the element and types defined in the MPEG-7 schema must be qualified with `mpeg7:` prefix. For example,

```
<complexType name="MyElementType">
  <sequence>
    <element name="MyVector" type="mpeg7:MyVectorType"/>
  </sequence>
  <attribute name="myAttribute" type="mpeg7:unsigned8"/>
</complexType>
```

### 4.2 Audio representation

Within the scope of this standard, the samples of the described audio signals are interpreted as two's complement fractional numbers (i.e. numbers between  $-1$ , inclusive, and  $+1$ , exclusive), where the Most Significant Bit (MSB) represents the value  $-1$ .

## 5 Audio Framework

### 5.1 Introduction

The Audio Framework contains low level tools designed to provide a basis for construction of higher level audio applications.

There are essentially two ways of describing low-level audio features. One may sample values at regular intervals or one may use `AudioSegments` to demark regions of similarity and dissimilarity within the sound. Both of these possibilities are embodied in the low-level descriptor types, `AudioLLDScalarType` and `AudioLLDVectorType`. A descriptor of either of these types may be instantiated as sampled values in a `ScalableSeries`, or as a summary descriptor within an `AudioSegment`. `AudioSegment`, which is a concept that permeates the MPEG-7 Audio standard, is specified in ISO/IEC 15938 Part 5, Multimedia Description Schemes, but we also give a brief overview here.

An `AudioSegment` is a temporal interval of audio material, which may range from arbitrarily short intervals to the entire audio portion of a media document. A required element of an `AudioSegment` is a `MediaTime` descriptor that denotes the beginning and end of the segment. The `TemporalMask` DS is a construct that allows one to specify a temporally non-contiguous `AudioSegment`. An `AudioSegment` (as with any `SegmentType`) may be decomposed hierarchically to describe a tree of `Segments`.

Another key concept is in the abstract datatypes: `AudioDType` and `AudioDSType`. In order for an audio descriptor or description scheme to be attached to a segment, it must inherit from one of these two types. They are defined in ISO/IEC 15938 part 5. The relationship between these types is shown in Figure 1.

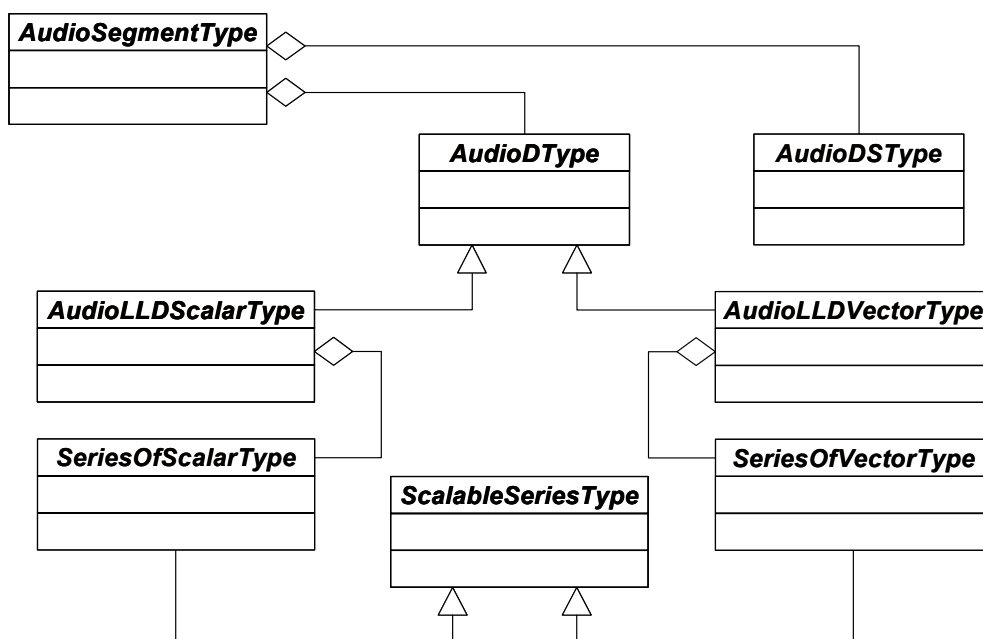


Figure 1 — Illustration of the various structural types in the Audio Framework

### 5.2 Scalable Series

#### 5.2.1 Introduction

Scalable series are datatypes for series of values (scalars or vectors). They allow the series to be scaled (downsampled) in a well-defined fashion. Two types are available: `SeriesOfScalarType` and `SeriesOfVectorType`. They are useful in particular to build descriptors that contain *time series* of values.

### 5.2.2 ScalableSeriesType

This is an abstract type inherited by `SeriesOfScalarType` and `SeriesOfVectorType`. Its attributes define the dimensions and scaling ratio of the series.

#### 5.2.2.1 Syntax

```
<!-- ##### -->
<!-- Definition of ScalableSeries datatype -->
<!-- ##### -->
<complexType name="ScalableSeriesType" abstract="true">
  <sequence>
    <element name="Scaling" minOccurs="0" maxOccurs="unbounded">
      <complexType>
        <attribute name="ratio" type="positiveInteger" use="required"/>
        <attribute name="numOfElements" type="positiveInteger"
          use="required"/>
      </complexType>
    </element>
  </sequence>
  <attribute name="totalNumOfSamples" type="positiveInteger" use="required"/>
</complexType>
```

#### 5.2.2.2 Semantics

| Name                            | Definition                                                                                                                                                                                                                                                                       |
|---------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>ScalableSeriesType</code> | An abstract type representing series of values, at full resolution or after scaling (downsampling) by a scaling operation. In the latter case the series contains sequences that have been concatenated together. Within each sequence, the elements share the same scale ratio. |
| <code>Scaling</code>            | To specify how the original samples are scaled. If absent, the original samples are described without scaling.                                                                                                                                                                   |
| <code>ratio</code>              | Scale ratio (number of original samples represented by each scaled sample) common to all elements in a sequence. The value to be used when <code>Scaling</code> is absent is 1.                                                                                                  |
| <code>numOfElements</code>      | Number of scaled elements in a sequence. The value to be used when <code>Scaling</code> is absent is equal to the value of <code>totalNumOfSamples</code> .                                                                                                                      |
| <code>totalNumOfSamples</code>  | Total number of samples of the original series (before scaling).                                                                                                                                                                                                                 |

Note that the last sample of the series may summarize fewer than `ratio` samples. This happens if `totalNumOfSamples` is smaller than the sum over runs of the product of `numOfElements` by `ratio`. An illustration of the Scalable Series is shown in Figure 2, where 'k' is an index in the scaled series. In this figure, the 31 samples of the original series (filled circles) are summarized by 13 samples of the scaled series (open circles). The first three scaled samples each summarizes two original samples, the next two six, the next two one, etc. The last scaled sample has nominally a ratio of two, but actually summarizes only one original sample. This situation is legal, and detected by comparing the sum of `ratio` times `numOfElements` products to `totalNumOfSamples`.



| Name     | Definition                                                                                                                                                                                                                                                                       |
|----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Max      | Series of maxima of groups of samples. The value of <code>numOfElements</code> shall equal the length of the vector. This element shall be absent or empty if the <code>Raw</code> element is present.                                                                           |
| Mean     | Series of means of groups of samples. The value of <code>numOfElements</code> shall equal the length of the vector. This element shall be absent or empty if the <code>Raw</code> element is present.                                                                            |
| Random   | Downsampled series (one sample selected at random from each group of samples). The value of <code>numOfElements</code> shall equal the length of the vector. This element shall be absent or empty if the <code>Raw</code> element is present.                                   |
| First    | Downsampled series (first sample selected from each group of samples). The value of <code>numOfElements</code> shall equal the length of the vector. This element shall be absent or empty if the <code>Raw</code> element is present.                                           |
| Last     | Downsampled series (last sample selected from each group of samples). The value of <code>numOfElements</code> shall equal the length of the vector. This element shall be absent or empty if the <code>Raw</code> element is present.                                            |
| Variance | Series of variances of groups of samples. The value of <code>numOfElements</code> shall equal the length of the vector. This element shall be absent or empty if the <code>Raw</code> element is present. Mean must be present in order for <code>Variance</code> to be present. |
| Weight   | Optional series of weights. Contrary to other fields, these do not represent values of the descriptor itself, but rather auxiliary weights to control scaling (see below). The value of <code>numOfElements</code> shall equal the length of the vector.                         |

Note: Data of a full resolution series (`ratio = 1`) are stored in the `Raw` field. Accompanying zero-sized fields (such as `Mean`) indicate how the series may be scaled, if the need for scaling arises. The data are then stored in the scaled field(s) and the `Raw` field disappears.

Scalable Series allow data to be stored at reduced resolution, according to a number of possible scaling operations. The allowable operations are those that are *scalable* in the following sense. Suppose the original series is scaled by a scale ratio of  $P$ , and this scaled series is then rescaled by a factor of  $Q$ . The result is the same as if the original series had been scaled by a scale ratio of  $N = PQ$ .

Figure 3 illustrates the scalability property. This scaled series can be derived indifferently from the original series by applying the scaling operation with the `ratios` shown, or from the scaled Series of Figure 2 by applying the appropriate rescaling operation. The result is identical. Scaling operations are chosen among those for which this property can be enforced.

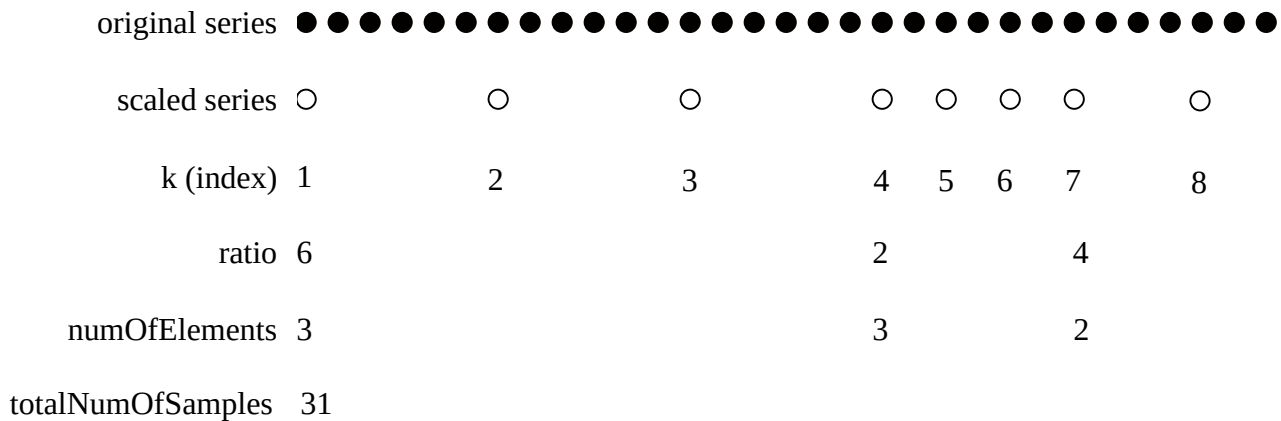


**Figure 3 — An illustration of the scalability property**

If the scaling operations are used, they shall be computed as follows.

| Name     | Definition                                                                                                    | Definition if Weight present                                                                                                                                |
|----------|---------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Min      | $m_k = \min_{i=1+(k-1)N}^{kN} x_i$                                                                            | Ignore samples with zero weight. If all have zero weight, set to zero by convention.                                                                        |
| Max      | $M_k = \max_{i=1+(k-1)N}^{kN} x_i$                                                                            | Ignore samples with zero weight. If all have zero weight, set to zero by convention.                                                                        |
| Mean     | $\bar{x}_k = (1/N) \sum_{i=1+(k-1)N}^{kN} x_i$                                                                | $\bar{x}_k = \sum_{i=1+(k-1)N}^{kN} w_i x_i \bigg/ \sum_{i=1+(k-1)N}^{kN} w_i$ <p>If all samples have zero weight, set to zero by convention.</p>           |
| Random   | choose at random among N samples                                                                              | Choose at random with probabilities proportional to weights. If all samples have zero weight, set to zero by convention.                                    |
| First    | choose the first of N samples                                                                                 | Choose first non-zero-weight sample. If all samples have zero weight, set to zero by convention.                                                            |
| Last     | choose the last of N samples                                                                                  | Choose last non-zero-weight sample. If all samples have zero weight, set to zero by convention.                                                             |
| Variance | $z_k = (1/N) \sum_{i=1+(k-1)N}^{kN} (x_i - \bar{x}_k)^2$ $= (1/N) \sum_{i=1+(k-1)N}^{kN} x_i^2 - \bar{x}_k^2$ | $z_k = \sum_{i=1+(k-1)N}^{kN} w_i (x_i - \bar{x}_k)^2 \bigg/ \sum_{i=1+(k-1)N}^{kN} w_i$ <p>If all samples have zero weight, set to zero by convention.</p> |
| Weight   | $\bar{w}_k = (1/N) \sum_{i=1+(k-1)N}^{kN} w_i$                                                                |                                                                                                                                                             |

In these formulae,  $k$  is an index in the scaled series, and  $i$  an index in the original series.  $N$  is the number of samples summarized by each scaled sample. The formula for `Variance` differs from the standard formula for unbiased variance by the presence of  $N$  rather than  $N-1$ . Unbiased variance is easy to derive from it. If the `Weight` field is present, the terms of all sums are weighted.

#### 5.2.4 SeriesOfScalarBinaryType

Use this type to instantiate a series of scalars with a uniform power-of-two ratio. The restriction to a power-of-two ratio eases the comparison of series with different `ratios` as the decimation required for the comparison will also be a power of 2. Such decimation allows perfect comparison. It also allows an additional scaling operation to be defined (scalewise variance). Considering these computational properties of power-of-two scale ratios, the `SeriesOfScalarBinaryType` is the most useful of the Scalable Series family.

Note that the types `SeriesOfScalarBinaryType` and `SeriesOfVectorBinaryType` inherit from the appropriate non-binary type. This means that although they are not used directly in this document, they can be used in place of the non-binary type at any time.

Figure 4 shows an illustration of `SeriesOfScalarBinaryType`. In this subtype of `SeriesOfScalarType`, all `ratios` must all be powers of two, and all `numOfElements` values must be such that the series may be rescaled to the largest `ratio` that it contains. In this example, the largest `ratio` is 8. The first four samples (`ratio` 2) can be rescaled to a single sample with `ratio` 8, and the last four samples can be rescaled to two samples with `ratio` 8. Note, as in previous illustrations, that the last scaled sample summarizes fewer original samples than its nominal `ratio`.

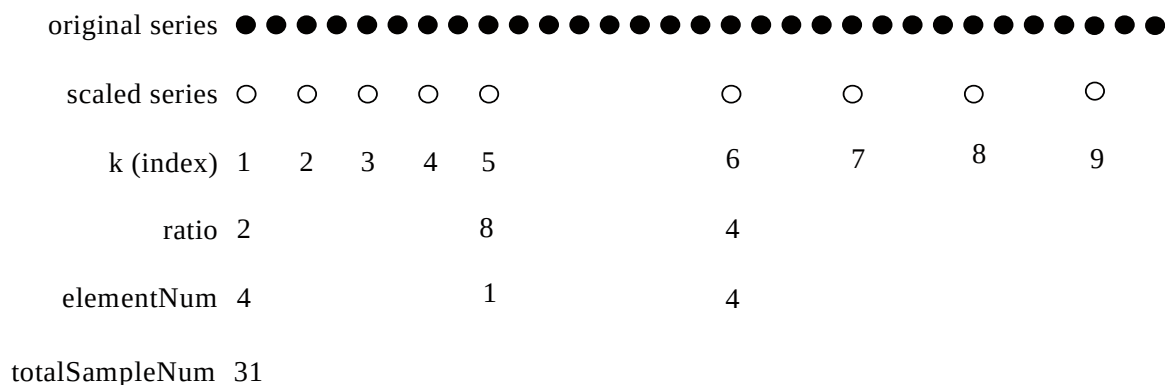


Figure 4 — An illustration of `SeriesOfScalarBinaryType`

##### 5.2.4.1 Syntax

```
<!-- ##### -->
<!-- Definition of SeriesOfScalarBinary datatype -->
<!-- ##### -->
<complexType name="SeriesOfScalarBinaryType">
  <complexContent>
    <extension base="mpeg7:SeriesOfScalarType">
      <sequence>
        <element name="VarianceScalewise" type="mpeg7:FloatMatrixType"
          minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

### 5.2.4.2 Semantics

| Name                                  | Definition                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>SeriesOfScalarBinaryType</code> | A representation of a series of scalar values scaled by a power of two factor.                                                                                                                                                                                                                                                                                                                                                       |
| <code>VarianceScalewise</code>        | Optional array of arrays of scalewise variance coefficients. Scalewise variance is a decomposition of the variance into a series of coefficients, each of which describes the variability at a particular scale. There are $\log_2(\text{ratio})$ such coefficients. See definition below. Number of rows must equal <code>numOfElements</code> , number of columns must equal the number of coefficients of the scalewise variance. |

### 5.2.4.3 Usage, extraction and examples

#### 5.2.4.3.1 Scalewise Variance

Scalewise variance is a decomposition of the variance into a vector of coefficients that describe variability at different scales. The sum of these coefficients equals the variance. To calculate the scalewise variance of a set of  $N = 2^m$  samples, first recursively form a binary tree of means:

$$\bar{x}_k^1 = (x_{2k-1} + x_{2k}) / 2, \quad k = 1, \dots, N/2$$

$$\bar{x}_k^2 = (\bar{x}_{2k-1}^1 + \bar{x}_{2k}^1) / 2, \quad k = 1, \dots, N/4$$

...

$$\bar{x}_k^m = (\bar{x}_{2k-1}^{m-1} + \bar{x}_{2k}^{m-1}) / 2, \quad k = 1$$

where  $x$  is a sample. Then calculate the coefficients  $z$ :

$$z^1 = (2/N) \sum_{k=1}^{N/2} (x_{2k-1} - x_{2k})^2 / 2$$

$$z^2 = (4/N) \sum_{k=1}^{N/4} (\bar{x}_{2k-1}^1 - \bar{x}_{2k}^1)^2 / 2$$

...

$$z^m = (\bar{x}_{2k-1}^{m-1} - \bar{x}_{2k}^{m-1})^2 / 2$$

The vector formed by these coefficients is the scalewise variance for this group of samples. The `VarianceScalewise` field stores a series of such vectors.

### 5.2.5 SeriesOfVectorType

This descriptor represents a series of vectors.

### 5.2.5.1 Syntax

```

<!-- ##### -->
<!-- Definition of SeriesOfVector datatype -->
<!-- ##### -->
<complexType name="SeriesOfVectorType">
  <complexContent>
    <extension base="mpeg7:ScalableSeriesType">
      <sequence>
        <element name="Raw" type="mpeg7:FloatMatrixType" minOccurs="0"/>
        <element name="Min" type="mpeg7:FloatMatrixType" minOccurs="0"/>
        <element name="Max" type="mpeg7:FloatMatrixType" minOccurs="0"/>
        <element name="Mean" type="mpeg7:FloatMatrixType" minOccurs="0"/>
        <element name="Random" type="mpeg7:FloatMatrixType" minOccurs="0"/>
        <element name="First" type="mpeg7:FloatMatrixType" minOccurs="0"/>
        <element name="Last" type="mpeg7:FloatMatrixType" minOccurs="0"/>
        <element name="Variance" type="mpeg7:FloatMatrixType" minOccurs="0"/>
        <element name="Covariance" type="mpeg7:FloatMatrixType"
          minOccurs="0"/>
        <element name="VarianceSummed" type="mpeg7:floatVector"
          minOccurs="0"/>
        <element name="MaxSqDist" type="mpeg7:floatVector" minOccurs="0"/>
        <element name="Weight" type="mpeg7:floatVector" minOccurs="0"/>
      </sequence>
      <attribute name="vectorSize" type="positiveInteger" default="1"/>
    </extension>
  </complexContent>
</complexType>

```

### 5.2.5.2 Semantics

| Name               | Definition                                                                                                                                                                                                                                                                    |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| SeriesOfVectorType | A type for scaled series of vectors.                                                                                                                                                                                                                                          |
| Raw                | Series of unscaled samples (full resolution). Use only if <code>ratio=1</code> for the entire series.                                                                                                                                                                         |
| Min                | Series of minima of groups of samples. Number of rows must equal <code>numOfElements</code> , number of columns must equal <code>vectorSize</code> . This element must be absent or empty if the element <code>Raw</code> is present.                                         |
| Max                | Series of maxima of groups of samples. Number of rows must equal <code>numOfElements</code> , number of columns must equal <code>vectorSize</code> . This element must be absent or empty if the element <code>Raw</code> is present.                                         |
| Mean               | Series of means of groups of samples. Number of rows must equal <code>numOfElements</code> , number of columns must equal <code>vectorSize</code> . This element must be absent or empty if the element <code>Raw</code> is present.                                          |
| Random             | Downsampled series (one sample selected at random from each group of samples). Number of rows must equal <code>numOfElements</code> , number of columns must equal <code>vectorSize</code> . This element must be absent or empty if the element <code>Raw</code> is present. |
| First              | Downsampled series (first sample selected from each group of samples). Number of rows must equal <code>numOfElements</code> , number of columns must equal <code>vectorSize</code> . This element must be absent or empty if the element <code>Raw</code> is present.         |

| Name           | Definition                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Last           | Downsampled series (last sample selected from each group of samples). Number of rows must equal <code>numOfElements</code> , number of columns must equal <code>vectorSize</code> . This element must be absent or empty if the element <code>Raw</code> is present.                                                                                                                                         |
| Variance       | Series of variance vectors of groups of vector samples. Number of rows must equal <code>numOfElements</code> , number of columns must equal <code>vectorSize</code> . This element must be absent or empty if the element <code>Raw</code> is present. <code>Mean</code> must be present in order for <code>Variance</code> to be present.                                                                   |
| Covariance     | Series of covariance matrices of groups of vector samples. This is a three-dimensional matrix. Number of rows must equal <code>numOfElements</code> , number of columns and number of pages must both equal <code>vectorSize</code> . This element must be absent or empty if the element <code>Raw</code> is present. <code>Mean</code> must be present in order for <code>Covariance</code> to be present. |
| VarianceSummed | Series of summed variance coefficients of groups of samples. Size of the vector must equal <code>numOfElements</code> . This element must be absent or empty if the element <code>Raw</code> is present. <code>Mean</code> must be present in order for <code>VarianceSummed</code> to be present.                                                                                                           |
| MaxSqDist      | Maximum Squared Distance (MSD). Series of coefficients representing an upper bound of the distance between groups of samples and their mean. Size of array must equal <code>numOfElements</code> . This element must be absent or empty if the element <code>Raw</code> is present. If <code>MaxSqDist</code> is present, <code>Mean</code> must also be present.                                            |
| Weight         | Optional series of weights. Weights control downsampling of other fields (see explanation for <code>SeriesOfScalars</code> ). Size of array must equal <code>numOfElements</code> .                                                                                                                                                                                                                          |
| vectorSize     | The number of elements of each vector within the series.                                                                                                                                                                                                                                                                                                                                                     |

Most of the above operations are straightforward extensions of operations previously defined in section 5.2.3.2 for series of scalars, applied uniformly to each dimension of the vectors. Operations that are specific to vectors are defined here:

| Name           | Definition                                                                                         | Definition if <code>weight</code> present                                                                                                                                 |
|----------------|----------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Covariance     | $\sigma_k^{jj'} = \frac{1}{N} \sum_{i=1+(k-1)N}^{kN} (x_i^j - \bar{x}^j)(x_i^{j'} - \bar{x}^{j'})$ | $\sigma_k^{jj'} = \frac{\sum_{i=1+(k-1)N}^{kN} w_i (x_i^j - \bar{x}^j)(x_i^{j'} - \bar{x}^{j'})}{\sum_{i=1+(k-1)N}^{kN} w_i}$                                             |
| VarianceSummed | $z_k = (1/N) \sum_{j=1}^D \sum_{i=1+(k-1)N}^{kN} (x_i^j - \bar{x}_i^j)^2$                          | $z_k = \frac{\sum_{j=1}^D \sum_{i=1+(k-1)N}^{kN} w_i (x_i^j - \bar{x}_i^j)^2}{\sum_{i=1+(k-1)N}^{kN} w_i}$<br>If all samples have zero weight, set to zero by convention. |
| MaxSqDist      | $MSD_k = \max_{i=1+(k-1)N}^{kN} \ x_i - \bar{x}_k\ ^2$                                             | Ignore samples with zero weight. If all samples have zero weight, set to zero by convention                                                                               |

In these formulae,  $k$  is an index in the scaled series, and  $i$  an index in the original series.  $N$  is the number of vectors summarized by each scaled vector.  $D$  is the size of each vector and  $j$  is an index into each vector.  $\bar{x}_i^j$  is the mean of  $N$  samples.

The various variance/covariance options offer a choice of several cost/performance tradeoffs for the representation of variability.

## 5.2.6 SeriesOfVectorBinaryType

Use this type to instantiate a series of vectors with a uniform power-of-two `ratio`. The restriction to a power-of-two ratio eases the comparison of series with different `ratios` as the decimation necessary for the comparison is simply another power of 2. The use of power-of-two scale ratios is recommended.

### 5.2.6.1 Syntax

```

<!-- ##### -->
<!-- Definition of SeriesOfVectorBinary datatype -->
<!-- ##### -->
<complexType name="SeriesOfVectorBinaryType">
  <complexContent>
    <extension base="mpeg7:SeriesOfVectorType">
      <sequence>
        <element name="VarianceScalewise" type="mpeg7:FloatMatrixType"
          minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

### 5.2.6.2 Semantics

| Name                                  | Definition                                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>SeriesOfVectorBinaryType</code> | A representation of a reduced-resolution series of vector samples with a power-of-two <code>ratio</code> .                                                                                                                                                                                                                                                  |
| <code>VarianceScalewise</code>        | Array of arrays of scalewise summed-variance coefficients. Scalewise variance is a decomposition of the variance into a series of coefficients, each of which describes the variability at a particular scale. Number of rows must equal to <code>numOfElements</code> , number of columns must equal the number of coefficients of the scalewise variance. |

## 5.3 Low level Audio Descriptors

### 5.3.1 Introduction

Low-level Audio Descriptors (LLDs) consist of a collection of simple, low complexity descriptors that are designed to be used within the `AudioSegment` framework, see ISO/IEC 15938-5 (E). Whilst being useful in themselves, they also provide examples of a design framework for future extension of the audio descriptors and description schemes.

All low-level audio descriptors are defined as subtypes of either `AudioLLDScalarType` or `AudioLLDVectorType`, except the `AudioSpectrumBasis` D. There are two description strategies using these data types: single-valued summary and sampled series segment description. These two description strategies are made available for the two data types, `Scalar/SeriesOfScalarType` and `Vector/SeriesOfVectorType`, and are implemented as a choice in DDL.

When using summary descriptions (containing a single scalar or vector) there are no normative methods for calculating the single-valued summarization. However, when using series-based descriptions, the summarization

values shall be calculated using the scaling methods provided by the `SeriesOfScalarType` and `SeriesOfVectorType` descriptors, such as the `min`, `max` and `mean` operators.

### 5.3.2 AudioLLDScalarType

#### 5.3.2.1 Syntax

```
<!-- ##### -->
<!-- Definition of AudioLLDScalar D -->
<!-- ##### -->
<complexType name="AudioLLDScalarType" abstract="true">
  <complexContent>
    <extension base="mpeg7:AudioDType">
      <choice>
        <element name="Scalar" type="float"/>
        <element name="SeriesOfScalar" minOccurs="1" maxOccurs="unbounded">
          <complexType>
            <complexContent>
              <extension base="mpeg7:SeriesOfScalarType">
                <attribute name="hopSize" type="mpeg7:mediaDurationType"
                  default="PT10N1000F"/>
              </extension>
            </complexContent>
          </complexType>
        </element>
      </choice>
      <attribute name="confidence" type="mpeg7:zeroToOneType" use="optional"/>
    </extension>
  </complexContent>
</complexType>
```

#### 5.3.2.2 Semantics

| Name               | Definition                                                                                                                                                                                                                                                                                  |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| AudioLLDScalarType | Abstract definition inherited by all scalar datatype audio descriptors.                                                                                                                                                                                                                     |
| Scalar             | Value of the descriptor                                                                                                                                                                                                                                                                     |
| SeriesOfScalar     | Scalar values for sampled-series description of an audio segment. Use of this scalable series datatype promotes compatibility between sampled descriptions.                                                                                                                                 |
| hopSize            | Time interval between data samples for series description. The default value is PT10N1000F which is 10 milliseconds.<br><br>Values other than the default shall be integer multiples/divisors of 10 milliseconds. This will ensure compatibility of descriptors sampled at different rates. |

### 5.3.3 AudioLLDVectorType

#### 5.3.3.1 Syntax

```

<!-- ##### -->
<!-- Definition of AudioLLDVector D -->
<!-- ##### -->
<complexType name="AudioLLDVectorType" abstract="true">
  <complexContent>
    <extension base="mpeg7:AudioDType">
      <choice>
        <element name="Vector" type="mpeg7:floatVector"/>
        <element name="SeriesOfVector" minOccurs="1" maxOccurs="unbounded">
          <complexType>
            <complexContent>
              <extension base="mpeg7:SeriesOfVectorType">
                <attribute name="hopSize" type="mpeg7:mediaDurationType"
                  default="PT10N1000F"/>
              </extension>
            </complexContent>
          </complexType>
        </element>
      </choice>
    </extension>
  </complexContent>
</complexType>

```

#### 5.3.3.2 Semantics

| Name               | Definition                                                                                                                                                                                                                                                                                  |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| AudioLLDVectorType | Abstract definition inherited by all vector datatype audio descriptors.                                                                                                                                                                                                                     |
| Vector             | Vector value of descriptor                                                                                                                                                                                                                                                                  |
| SeriesOfVector     | Vector values for sampled-series description of an audio segment. Use of this scalable series datatype promotes compatibility between sampled descriptions.                                                                                                                                 |
| hopSize            | Time interval between data samples for series description. The default value is PT10N1000F which is 10 milliseconds.<br><br>Values other than the default shall be integer multiples/divisors of 10 milliseconds. This will ensure compatibility of descriptors sampled at different rates. |

#### 5.3.3.3 Usage and Extraction

Audio descriptors that calculated at regular intervals (sample period or frame period) shall use the `hopSize` field to specify the extraction period. In all cases, the `hopSize` shall be a positive integer multiple/divisor of the default 10 millisecond sampling period. Note that downsampling by means of the scalable series does not change the specified `hopSize` but instead specifies the downsampling scale ratio to be used together with the `hopSize`.

`AudioLLDScalarType` and `AudioLLDVectorType` are both abstract and therefore never instantiated.

### 5.3.4 AudioWaveformType

AudioWaveform D describes the audio waveform envelope, typically for display purposes.

#### 5.3.4.1 Syntax

```
<!-- ##### -->
<!-- Definition of AudioWaveform D -->
<!-- ##### -->
<complexType name="AudioWaveformType">
  <complexContent>
    <extension base="mpeg7:AudioLLDScalarType">
      <attribute name="minRange" type="float" use="optional"/>
      <attribute name="maxRange" type="float" use="optional"/>
    </extension>
  </complexContent>
</complexType>
```

#### 5.3.4.2 Semantics

| Name              | Definition                                       |
|-------------------|--------------------------------------------------|
| AudioWaveformType | Description of the waveform of the audio signal. |
| minRange          | Lower limit of audio signal amplitude            |
| maxRange          | Upper limit of audio signal amplitude            |

#### 5.3.4.3 Usage and extraction

##### 5.3.4.3.1 Purpose

AudioWaveform D allows economical display of an audio waveform. For example, a sound editing application program can display a summary of an entire audio file immediately without processing the audio data and data may be displayed and edited over a network, etc. Whatever the number of samples, the waveform may be displayed using a small set of values that represent extrema (min and max) of frames of samples. Min and max are stored as scalable time series within the AudioWaveform D. They may also be used for fast comparison between waveforms.

##### 5.3.4.3.2 To create a description

- Instantiate an AudioWaveformType with hopSize set to the required temporal resolution (default 10ms), or if the raw signal is desired set hopSize to 1/(sampling rate).
- Determine the equivalent number of samples, ns, in each frame (ns = sampling rate x hop size).
- If ns = 1 then instantiate the Raw and store the raw data. Instantiate empty Min and Max fields.
- Otherwise take the minimum and maximum value in each hopSize frame. Instantiate Min and Max fields and fill them with scaled samples.

##### 5.3.4.3.3 To use a description for display

- Read the arrays of min and max values from the AudioWaveformType.

- b) For each `min/max` pair, draw a vertical line from `min` to `max`.
- c) Label axes using the `hopSize` and scaling information.

### 5.3.5 AudioPowerType

AudioPower D describes the temporally-smoothed instantaneous power (square of waveform values).

#### 5.3.5.1 Syntax

```
<!-- ##### -->
<!-- Definition of AudioPower D -->
<!-- ##### -->
<complexType name="AudioPowerType">
  <complexContent>
    <extension base="mpeg7:AudioLLDScalarType"/>
  </complexContent>
</complexType>
```

#### 5.3.5.2 Semantics

| Name           | Definition                                    |
|----------------|-----------------------------------------------|
| AudioPowerType | Description of the power of the audio signal. |

#### 5.3.5.3 Usage and extraction

##### 5.3.5.3.1 Extraction

Instantaneous power is calculated by taking the square of waveform samples. These are averaged over time intervals of length corresponding to `hopSize` and stored in the `Mean` field of a `SeriesOfScalarType`.

##### 5.3.5.3.2 Purpose

Instantaneous power is a useful measure of the amplitude of a signal as a function of time,  $P(t)=|s(t)|^2$ . In association with `AudioSpectrumCentroid D` and `AudioSpectrumSpread D`, the `AudioPower D` provides an economical description of the power spectrum (spreading the power over the spectral range specified by the centroid and spread) that can be compared with a log-frequency spectrum. Another possibility is to store instantaneous power at high temporal resolution, in association with a high spectral resolution power spectrum at low temporal resolution, to obtain a cheap representation of the power spectrum that combines both spectral and temporal resolution.

##### 5.3.5.3.3 Motivation for the design

Instantaneous power is coherent with the power spectrum. A signal labeled with the former can meaningfully be compared to a signal labeled with the latter. Note however that temporal smoothing operations are not quite the same, so values may differ slightly for identical signals.

### 5.3.6 Audio Spectrum Descriptors

#### 5.3.6.1 Introduction

The following descriptors (5.3.7 to 5.3.12) are all descriptions of the audio spectrum. They have a central link that they all derive from a time-frequency analysis of the audio signal.

### 5.3.6.2 AudioSpectrumAttributeGrp

The AudioSpectrumAttributeGrp defines a common set of attributes applicable to many of the spectrum descriptions.

#### 5.3.6.2.1 Syntax

```
<!-- ##### -->
<!-- Definition of audioSpectrumAttributeGrp -->
<!-- ##### -->
<attributeGroup name="audioSpectrumAttributeGrp">
  <!-- Edge values are in Hertz -->
  <attribute name="loEdge" type="float" default="62.5"/>
  <attribute name="hiEdge" type="float" default="16000"/>
  <attribute name="octaveResolution" use="optional">
    <simpleType>
      <restriction base="string">
        <enumeration value="1/16"/>
        <enumeration value="1/8"/>
        <enumeration value="1/4"/>
        <enumeration value="1/2"/>
        <enumeration value="1"/>
        <enumeration value="2"/>
        <enumeration value="4"/>
        <enumeration value="8"/>
      </restriction>
    </simpleType>
  </attribute>
</attributeGroup>
```

#### 5.3.6.2.2 Semantics

| Name       | Definition                                                                                            |
|------------|-------------------------------------------------------------------------------------------------------|
| loEdge     | Lower edge of logarithmically-spaced frequency bands.                                                 |
| hiEdge     | Higher edge of logarithmically-spaced frequency bands.                                                |
| Resolution | Frequency resolution of logarithmic spectrum (width of each spectrum band between loEdge and hiEdge). |

### 5.3.7 AudioSpectrumEnvelopeType

AudioSpectrumEnvelope D describes the spectrum of the audio according to a logarithmic frequency scale.

#### 5.3.7.1 Syntax

```
<!-- ##### -->
<!-- Definition of AudioSpectrumEnvelope D -->
<!-- ##### -->
<complexType name="AudioSpectrumEnvelopeType">
  <complexContent>
    <extension base="mpeg7:AudioLLDVectorType">
      <attributeGroup ref="mpeg7:audioSpectrumAttributeGrp"/>
    </extension>
  </complexContent>
</complexType>
```

### 5.3.7.2 Semantics

| Name                      | Definition                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| AudioSpectrumEnvelopeType | Description of the power spectrum of the audio signal. The spectrum consists of one coefficient representing power between 0Hz and <code>loEdge</code> , a series of coefficients representing power in <code>resolution</code> sized bands, between <code>loEdge</code> and <code>hiEdge</code> , and a coefficient representing power beyond <code>hiEdge</code> , in this order. |

### 5.3.7.3 Usage and extraction

#### 5.3.7.3.1 Purpose

The `AudioSpectrumEnvelope` D describes the short-term power spectrum of the audio waveform as a time series of spectra with a logarithmic frequency axis. It may be used to display a spectrogram, to synthesize a crude "auralization" of the data, or as a general-purpose descriptor for search and comparison.

#### 5.3.7.3.2 Motivation for the design

A logarithmic frequency axis is used to reconcile requirements of concision and descriptive power. Peripheral frequency analysis in the ear roughly follows a logarithmic axis.

The power spectrum is used because of its scaling properties (the power spectrum over an interval is equal to the sum of power spectra over subintervals).

#### 5.3.7.3.3 Specification of the descriptor (normative)

These specifications are normative and ensure that descriptions produced by different programs are either directly comparable, or else can easily be converted to comparable descriptions. Importantly, a high-resolution description is readily convertible to a low-resolution description, in both the spectral or temporal dimension.

The `AudioSpectrumEnvelope` D describes the spectrum over the frequency range between `loEdge` and `hiEdge`. The range between `loEdge` and `hiEdge` is divided into multiple bands. The resolution, in octaves, of the bands is specified by `resolution`. Except for when the `octaveResolution` is 1/8, both `loEdge` and `hiEdge` must be related to 1kHz as described in the following equation:

$$edge = 2^m \times 1KHz \quad (5.3.7.1)$$

where  $r$  is the resolution in octaves,  $m \in \mathbb{Z}$  (i.e.,  $m$  an integer).

For the case when `resolution` is "8 octave" the spectrum delivers a single coefficient representing within-band power, and two extra coefficients for below-band and above-band power. In this case the default values for `loEdge` and `hiEdge` should be used.

If  $m_l$  and  $m_h$  are the integers corresponding to Equation 5.3.7.1 when `edge` equals `loEdge` and `hiEdge`, respectively, then the full set of band edges are given by  $edge = 2^m \times 1KHz, m_l \leq m \leq m_h$ .

In every case the spectrum contains two extra values, one representing the energy between 0Hz and `loEdge`, and the other energy between `hiEdge` and half the sampling rate (See Figure 5). If `hiEdge` equals half the sampling rate then the second extra value is set to zero. These values measure the "out-of-band" energy. Default `hiEdge` is 16000Hz (corresponding to the upper limit of hearing). Default `loEdge` is 62.5 Hz (8 octaves below `hiEdge`). The default analysis frame period is 10 ms, which is within the range of estimates for temporal acuity of the ear (8 to 13 ms) and is also the default analysis frame period for sampled audio descriptors.

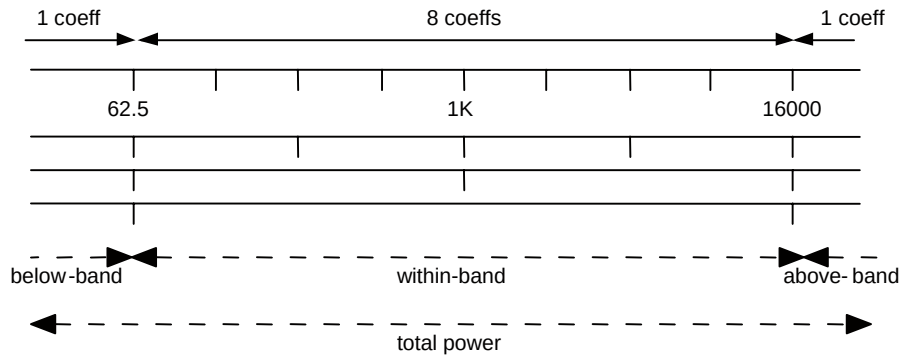


**Figure 5 — Illustration of Audio Spectrum Envelope Bands**

#### 5.3.7.3.4 Extraction (informative)

To extract the AudioSpectrumEnvelope the following method is recommended. The method involves a sliding window FFT analysis, with a resampling to logarithmic spaced bands.

- Determine the required hop length  $h$ , corresponding to the hopsize. If the sampling rate is  $sr$ , then  $h = sr \cdot \text{hopsize}$  (e.g.  $h = 16000 \cdot 0.01 = 160$  samples). If  $sr \cdot \text{hopsize}$  is not a whole number of samples then generate a vector  $h$  such that  $\text{mean}(h) = sr \cdot \text{hopsize}$  (e.g.  $sr \cdot \text{hopsize} = 22050 \cdot 0.01 = 220.5$ ,  $h = [220 \ 221]$ ). By cycling through the vector of hop lengths the analysis will not stray over time, but will give minor jitter from the defined hopsize. This enables reasonable comparison of data sampled at differing rates.
- Determine the analysis window length  $lw$ . The analysis window has been chosen to have a default value of 3 hopsizes, 30ms. This is to provide enough spectral resolution to roughly resolve the 62.5 Hz-wide first channel of a '1 octave' resolution spectrum.
- Determine the FFT size,  $NFFT$ .  $NFFT$  is the next-larger power-of-two number of samples from  $lw$ , e.g. If  $lw = 1323$  samples then  $NFFT$  would be 2048.
- Perform a STFT using a Hamming window of length  $lw$ , a shift of  $h$  samples (where  $h$  is a vector, rotate through the vector to prevent stray, and deliver minimal jitter), using a  $NFFT$  point FFT, with out-of-window samples set to 0. The descriptor only retains the square magnitude of the FFT coefficients,  $|X_w(k)|^2$ . The sum of the power spectrum coefficients is equal to the average power in the analysis window,  $P_w$ . By Parseval's theorem there is a further factor of  $1/NFFT$  to equate the sum of the squared magnitude of the FFT coefficients to the sum of the squared, zero-padded, windowed signal.

$$P_w = \frac{1}{lw} \sum_{n=0}^{lw-1} |x_w(n)|^2 = \frac{1}{lw * NFFT} \sum_{k=0}^{NFFT-1} |X_w(k)|^2$$

where  $x_w(n) = s(n) * w(n)$ ,  $0 \leq n < lw$  and  $w(n)$  is the Hamming window of length  $lw$ .

$$\text{Hence } P_x(k) = \frac{1}{lw * NFFT} |X_w(k)|^2$$

Since the audio signal is a real signal its Fourier transform has even symmetry. Hence only the spectral coefficients up to the Nyquist frequency need be retained.

- Resample to a logarithmic scale. Let  $DF$  be the frequency spacing of the FFT ( $DF = sr/NFFT$ ). An FFT coefficient more than  $DF/2$  from a band edge is assigned to the band. A coefficient less than  $DF/2$  from a band edge is proportionally shared between bands, as illustrated in Figure 6.

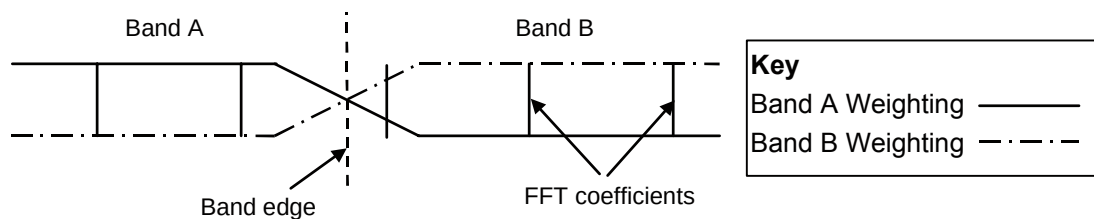


**Figure 6 — Weighting Method for linear - log conversion**

Important Note: Due to the weighting method illustrated in Figure 6 it is important to select an appropriate `loEdge` at fine frequency resolutions. To be able to resolve the logarithmic bands there needs to be at least one FFT coefficient in each band. In some cases this means that the default `loEdge` is unsuitable. Table 1 indicates the minimum value that `loEdge` should be set to for some popular sampling frequencies, assuming default `hopSize`.

**Table 1 — Minimum `loEdge` for particular resolutions (exceeding default 62.5Hz)**

| Resolution  | Minimum <code>loEdge</code><br>(DF = 31.25, FFT size 1024, SR = 32kHz, FFT size 512, SR = 16kHz) | Minimum <code>loEdge</code><br>(DF = 21.53, FFT size 2048, SR = 44.1kHz, FFT size 1024, SR = 22.05kHz) | Minimum <code>loEdge</code><br>(DF = 23.44, FFT size 2048, SR = 48kHz) |
|-------------|--------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------|
| 1/4 octave  | 88.388 ( $62.5 \cdot 2^{0.5}$ )                                                                  | 62.5                                                                                                   | 105.1 ( $62.5 \cdot 2^{3/4}$ )                                         |
| 1/8 octave  | 324.21 ( $62.5 \cdot 2^{19/8}$ )                                                                 | 176.8 ( $62.5 \cdot 2^{12/8}$ )                                                                        | 192.78 ( $62.5 \cdot 2^{13/8}$ )                                       |
| 1/16 octave | 545.25 ( $62.5 \cdot 2^{25/8}$ )                                                                 | 439.1 ( $62.5 \cdot 2^{45/16}$ )                                                                       | 478.8 ( $62.5 \cdot 2^{47/16}$ )                                       |

### 5.3.8 AudioSpectrumCentroidType

`AudioSpectrumCentroid D` describes the center of gravity of the log-frequency power spectrum. The `SpectrumCentroid` is defined as the power weighted log-frequency centroid.

#### 5.3.8.1 Syntax

```
<!-- ##### -->
<!-- Definition of AudioSpectrumCentroid D -->
<!-- ##### -->
<complexType name="AudioSpectrumCentroidType">
  <complexContent>
    <extension base="mpeg7:AudioLLDScalarType"/>
  </complexContent>
</complexType>
```

### 5.3.8.2 Semantics

| Name                      | Definition                                                                                                  |
|---------------------------|-------------------------------------------------------------------------------------------------------------|
| AudioSpectrumCentroidType | Description of the center of gravity of the log-frequency power spectrum.<br>Range: -5 to $\log_2(sr/2000)$ |

### 5.3.8.3 Usage and extraction

#### 5.3.8.3.1 Extraction (informative)

To be coherent with other descriptors, in particular `AudioSpectrumEnvelope D`, the spectrum centroid is defined as the center-of-gravity of a log-frequency power spectrum. This definition is adjusted in the extraction to take into account the fact that a non-zero DC component creates a singularity, and eventual very-low frequency components (possibly spurious) have a disproportionate weight.

To extract the spectrum centroid:

- a) Calculate the power spectrum coefficients, as described in `AudioSpectrumEnvelope D` extraction parts a-d.

$$P_x(k), k = 0, \dots, \frac{NFFT}{2}$$

- b) Power spectrum coefficients below 62.5 Hz are replaced by a single coefficient, with power equal to their sum and a nominal frequency of 31.25 Hz.

$$bound = floor\left(\frac{62.5 \times NFFT}{sr}\right)$$

$$P'_x(0) = \sum_{k=0}^{bound} P_x(k), f(0) = 31.25$$

$$P'_x(n) = P_x(n + bound), f(n) = (n + bound) \frac{sr}{NFFT} \text{ where } n = 1, \dots, \frac{NFFT}{2} - bound$$

- c) Frequencies of all coefficients are scaled to an octave scale anchored at 1 kHz. The spectrum centroid is calculated as:  $C = \sum_n \log_2(f(n)/1000) P'_x(n) / \sum_n P'_x(n)$  where  $P'_x(n)$  is the power associated with frequency  $f(n)$ .

#### 5.3.8.3.2 Purpose

Spectrum centroid is an economical description of the shape of the power spectrum. It indicates whether the power spectrum is dominated by low or high frequencies and, additionally, it is correlated with a major perceptual dimension of timbre; i.e. sharpness.

#### 5.3.8.3.3 Motivation for the design

There are many different ways to design a spectrum centroid, according to the scale used for the values (amplitude, power, log power, cubic root power, etc.) and frequencies (linear or logarithmic scale) of spectrum coefficients. Perceptual weighting and masking can also be taken into account in more sophisticated measures. This particular design of `AudioSpectrumCentroid D` was chosen to be coherent with other descriptors, in

particular `AudioSpectrumEnvelope D`, so that a signal labeled with the former can reasonably be compared to a signal labeled with the latter.

### 5.3.9 AudioSpectrumSpreadType

`AudioSpectrumSpread D` describes the second moment of the log-frequency power spectrum.

#### 5.3.9.1 Syntax

```
<!-- ##### -->
<!-- Definition of AudioSpectrumSpread D -->
<!-- ##### -->
<complexType name="AudioSpectrumSpreadType">
  <complexContent>
    <extension base="mpeg7:AudioLLDScalarType"/>
  </complexContent>
</complexType>
```

#### 5.3.9.2 Semantics

| Name                                 | Definition                                                     |
|--------------------------------------|----------------------------------------------------------------|
| <code>AudioSpectrumSpreadType</code> | Description of the spread of the log-frequency power spectrum. |

#### 5.3.9.3 Usage and extraction

##### 5.3.9.3.1 Extraction

To be coherent with other descriptors, in particular `AudioSpectrumEnvelope D`, the spectrum spread is defined as the RMS deviation of the log-frequency power spectrum with respect to its center of gravity. Details are similar to `AudioSpectrumCentroid D`.

To extract the spectrum spread:

- Calculate the power spectrum,  $P'_x(n)$ , and corresponding frequencies,  $f(n)$ , of the waveform as for `AudioSpectrumCentroid D` extraction, parts a-b.
- Calculate the spectrum centroid,  $C$ , as described in `AudioSpectrumCentroid D` extraction part d.
- Calculate the spectrum spread,  $S$ , as the RMS deviation with respect to the centroid, on an octave scale:

$$S = \sqrt{\sum_n ((\log_2(f(n)/1000) - C)^2 P'_x(n)) / \sum_n P'_x(n)}$$

##### 5.3.9.3.2 Purpose

Spectrum spread is an economical descriptor of the shape of the power spectrum that indicates whether it is concentrated in the vicinity of its centroid, or else spread out over the spectrum. It allows differentiating between tone-like and noise-like sounds.

### 5.3.9.3.3 Motivation for the design

As for the spectrum centroid, there are many different ways to design a spectrum spread measure. This definition follows the same criteria as `AudioSpectrumCentroid D`, with which it is coherent.

### 5.3.10 AudioSpectrumFlatnessType

`AudioSpectrumFlatness D` describes the flatness properties of the spectrum of an audio signal within a given number of frequency bands.

#### 5.3.10.1 Syntax

```
<!-- ##### -->
<!-- Definition of AudioSpectrumFlatness D -->
<!-- ##### -->
<complexType name="AudioSpectrumFlatnessType">
  <complexContent>
    <extension base="mpeg7:AudioLLDVectorType">
      <attribute name="loEdge" type="float" default="250"/>
      <attribute name="hiEdge" type="float" default="16000"/>
    </extension>
  </complexContent>
</complexType>
```

#### 5.3.10.2 Semantics

| Name                                   | Definition                                                      |
|----------------------------------------|-----------------------------------------------------------------|
| <code>AudioSpectrumFlatnessType</code> | Description of the audio spectral flatness of the audio signal. |
| <code>loEdge</code>                    | Lower edge frequency (a default value of 250 is assumed)        |
| <code>hiEdge</code>                    | Upper edge frequency (a default value of 16000 is assumed)      |

### 5.3.10.3 Usage, extraction and examples

#### 5.3.10.3.1 Purpose (informative)

The `AudioSpectrumFlatnessType` describes the flatness properties of the short-term power spectrum of an audio signal. This descriptor expresses the deviation of the signal's power spectrum over frequency from a flat shape (corresponding to a noise-like or an impulse-like signal). A high deviation from a flat shape may indicate the presence of tonal components. The spectral flatness analysis is calculated for a number of frequency bands. It may be used as a feature vector for robust matching between pairs of audio signals.

#### 5.3.10.3.2 Extraction (normative)

The extraction of the `AudioSpectrumFlatnessType` can be efficiently combined with the extraction of the `AudioSpectrumEnvelopeType` and is done in several steps:

- A spectral analysis (windowing, DFT) of the input signal is performed using the same procedure and parameters specified for the extraction of the `AudioSpectrumEnvelopeType` part a-d, but with the window length, `lw`, corresponding to hop size (i.e. no overlap between subsequent calculations). Hence `hopSize = 30ms` is recommended for this descriptor.

- b) A frequency range from `loEdge` to `hiEdge` is covered. Both limits must be chosen in quarter octave relation to 1kHz as described in the following equation, i.e.

$$edge = 2^{0.25m} \times 1\text{kHz}$$

where  $m \in \mathbb{Z}$  (i.e.,  $m$  an integer).

In view of the limitations in available frequency resolution, use of `AudioSpectrumEnvelopeType` below 250 Hz is not recommended. A logarithmic frequency resolution of a 1/4 octave is used for all bands. Thus, all `AudioSpectrumFlatnessType` bands are commensurate with the frequency bands employed by `AudioSpectrumEnvelopeType`. In order to reduce the sensitivity against deviations in sampling frequency, the bands are defined in an overlapping fashion: For the calculation of the actual edge frequencies, the nominal lower edge and higher edge frequencies of each band are multiplied by the factors 0.95 and 1.05, respectively. Consequently, each band overlaps with its neighbor band by 10%. This results in band edges  $f_b$  as described in Table 2 (assuming the default `loEdge` value of 250 Hz).

The band edge frequencies are transformed to indices of power spectrum coefficients as follows: If  $DF$  is the frequency spacing of the DFT ( $DF = \text{sampling rate} / \text{DFT size}$ ), the lower and higher edge of band  $b$  are defined by their power spectrum coefficient indices,  $il(b)$  and  $ih(b)$ , respectively, which are derived from the edge frequencies by  $nint(f_b / DF)$ , where  $nint()$  denotes rounding to the nearest integer.

For each frequency band, the flatness measure is defined as the ratio of the geometric and the arithmetic mean of the power spectrum coefficients (i.e. squared absolute DFT value, incl. grouping if required)  $c(i)$  within the band  $b$  (i.e. from coefficient index  $il$  to coefficient index  $ih$ , inclusive).

$$SFM_b = \frac{\sqrt[il(b)-il(b)+1]{\prod_{i=il(b)}^{ih(b)} c(i)}}{\frac{1}{ih(b)-il(b)+1} \sum_{i=il(b)}^{ih(b)} c(i)}$$

If no audio signal is present (i.e. the mean power is zero), a flatness measure value of 1 is returned.

In order to reduce the computational effort and adapt the frequency resolution to “log” bands, all power spectrum coefficients in bands above the edge frequency of 1kHz are grouped, i.e. the above calculation is carried out using the average values over a group of power spectral coefficients rather than the single coefficients themselves. The grouping is defined in the following way:

- For all bands between nominal 1kHz and 2kHz, a grouping of 2 consecutive power spectrum coefficients is used. For all bands between nominal 2kHz and 4kHz, a grouping of 4 consecutive power spectrum coefficients is used. For all bands between nominal 4kHz and 8kHz, a grouping of 8 consecutive power spectrum coefficients is used and so on.
- For the last group of coefficients in each band, the following rule is applied: If at least 50% of the required coefficients for the group are available in that band, this last group is included using the necessary amount of additional coefficients from the successive band. Otherwise this group is not included, and the number of coefficients used from the particular band is reduced accordingly.

If the signal available to the extraction process does not supply proper signal content beyond a certain frequency limit (e.g. due to the signal sampling rate or other bandwidth limitations), no flatness values should be extracted for bands extending beyond this frequency limit. Instead, `hiEdge` should be reduced accordingly to signal the number of bands available with proper flatness data.

**Table 2 — Band overlaps**

|                    |                     |                     |                     |                     |     |
|--------------------|---------------------|---------------------|---------------------|---------------------|-----|
| Band edges $f_b$ : | Nominal             | 250.0 Hz – 297.3 Hz | 297.3 Hz – 353.6 Hz | 353.6 Hz – 420.4 Hz | ... |
|                    | Actual (overlapped) | 237.5 Hz – 312.2 Hz | 282.4 Hz – 371.2 Hz | 335.9 Hz – 441.5 Hz | ... |

### 5.3.10.3.3 Example (informative)

The following is an example instantiation of `AudioSpectrumFlatnessType`. Consider a series of calculated flatness measurements of 128 consecutive analysis frames. The following instantiation shows how to summarize these values by using the `Min`, `Max`, `Mean` and `Variance` fields of the `SeriesOfVector` datatype for a 4-band flatness descriptor:

```
<AudioDescriptor xsi:type="AudioSpectrumFlatnessType" loEdge="500"
                hiEdge="1000">
  <SeriesOfVector vectorSize="4" totalSampleNum="128">
    <Scaling ratio="64" elementNum="2"/>
    <Min dim="2 4"> 0.3 0.1 0.4 0.3 0.2 0.3 0.3 0.2 </Min>
    <Max dim="2 4"> 0.8 0.5 0.7 0.6 0.5 0.6 0.6 0.8 </Max>
    <Mean dim="2 4"> 0.6 0.4 0.5 0.4 0.4 0.5 0.4 0.4 </Mean>
    <Variance dim="2 4"> 0.1 0.11 0.06 0.08 0.07 0.1 0.09 0.07 </Variance>
  </SeriesOfVector>
</AudioDescriptor>
```

### 5.3.11 AudioSpectrumBasisType

The `AudioSpectrumBasis` D contains basis functions that are used to project high-dimensional spectrum descriptions into a low-dimensional representation. Spectrum dimensionality reduction plays a substantial role in automatic classification applications by compactly representing salient statistical information about audio segments. These features have been shown to perform well for automatic classification and retrieval applications. Applications to spectrogram summarization are also discussed below.

#### 5.3.11.1 Syntax

```
<!-- ##### -->
<!-- Definition of AudioSpectrumBasis D -->
<!-- ##### -->
<complexType name="AudioSpectrumBasisType">
  <complexContent>
    <extension base="mpeg7:AudioLLDVectorType">
      <attributeGroup ref="mpeg7:audioSpectrumAttributeGrp"/>
    </extension>
  </complexContent>
</complexType>
```

### 5.3.11.2 Semantics

| Name                   | Definition                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| AudioSpectrumBasisType | <p>Statistical basis functions of a spectrogram used for dimension reduction and summarization. Basis functions are stored in the <code>Raw</code> field of a <code>SeriesOfVector</code>, the dimensions of the series depend upon the usage model:</p> <p>For stationary basis components the dimension attribute is set to <code>dim="N K"</code> where <code>N</code> is the spectrum length and <code>K</code> is the number of basis functions.</p> <p>For time-varying basis components <code>dim="M N K"</code> where <code>M</code> is the number of blocks within the segment, <code>N</code> is the spectrum length and <code>K</code> is the number of basis functions per block. Block lengths must be at least <code>K</code> frames for <code>K</code> basis functions; default <code>hopSize</code> is <code>PT500N1000F</code>.</p> |
| AudioSpectrumAttrGroup | Spectrum parameters for the basis vectors; see <code>AudioSpectrumEnvelope</code> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

### 5.3.11.3 Example (informative)

The following example shows an instance of five stationary basis functions spanning an audio segment derived from an `AudioSpectrumEnvelope` `D` with  $\frac{1}{4}$ -octave resolution.

```
<AudioDescriptor xsi:type="AudioSpectrumBasisType" loEdge="62.5" hiEdge="8000"
    octaveResolution="1/4">
  <SeriesOfVector totalNumOfSamples="1" vectorSize="31 9">
    <Raw dim="31 9">
      0.082 -0.026 0.024 -0.093 0.010 -0.021 0.063 -0.103 0.057
      0.291 0.073 0.025 -0.039 0.026 -0.086 0.185 0.241 0.107
      0.267 0.062 0.030 -0.026 0.054 -0.115 0.171 0.266 0.240
      0.267 0.062 0.030 -0.026 0.054 -0.115 0.171 0.266 0.240
      0.271 -0.008 0.039 0.007 0.119 -0.067 0.033 0.165 0.175
      0.271 -0.008 0.039 0.007 0.119 -0.067 0.033 0.165 0.175
      0.269 -0.159 0.062 0.074 0.182 0.071 -0.194 0.054 -0.009
      <!-- more values here . . . -->
    </Raw>
  </SeriesOfVector>
</AudioDescriptor>
```

### 5.3.11.4 Extraction (normative)

The following section defines the method for extracting basis functions from a spectrum. Some details of extraction are indicated to be informative with no loss of compatibility between implementations. To extract a reduced-dimension basis from an `AudioSpectrumEnvelope` spectrum the following steps shall be executed:

- Power spectrum:** instantiate an `AudioSpectrumEnvelope` descriptor using the extraction method defined in the `AudioSpectrumEnvelope` `D`. The resulting data will be a `SeriesOfVectors` with `M` frames and `N` frequency bins.
- Log-scale norming:** for each spectral vector,  $\mathbf{x}$ , in `AudioSpectrumEnvelope`, convert the power spectrum to a decibel scale:

$$\chi = 10 \log_{10}(\mathbf{x}_t)$$

and compute the *L2-norm* of the resulting vector:

$$r = \sqrt{\sum_{k=1}^N \chi_k^2}$$

the new unit-norm spectral vector is given by:

$$\tilde{\mathbf{x}} = \frac{\boldsymbol{\chi}}{r}$$

- c) **Observation matrix:** place each normalized spectral frame *row-wise* into a matrix. The size of the resulting matrix is  $M \times N$  where  $M$  is the number of time frames and  $N$  is the number of frequency bins. The matrix will have the following structure:

$$\tilde{\mathbf{X}} = \begin{bmatrix} \tilde{\mathbf{x}}_1^T \\ \tilde{\mathbf{x}}_2^T \\ \vdots \\ \vdots \\ \tilde{\mathbf{x}}_M^T \end{bmatrix}$$

- d) **Basis extraction:** Extract a basis using a singular value decomposition (SVD), commonly implemented as a built-in function in many mathematical software packages using the command  $[\mathbf{U}, \mathbf{S}, \mathbf{V}] = \text{SVD}(\mathbf{X}, 0)$ . Use the *economy* SVD when available since the row-basis functions are not required and this will increase extraction efficiency. The SVD factors the matrix from step (c) in the following way:

$$\tilde{\mathbf{X}} = \mathbf{U} \mathbf{S} \mathbf{V}^T$$

where  $\mathbf{X}$  is factored into the matrix product of three matrices; the row basis  $\mathbf{U}$ , the diagonal singular value matrix  $\mathbf{S}$  and the transposed column basis functions  $\mathbf{V}$ . Reduce the spectral (column) basis by retaining only the first  $k$  basis functions, i.e. the first  $k$  columns of  $\mathbf{V}$ :

$$\mathbf{V}_K = [\mathbf{v}_1 \quad \mathbf{v}_2 \quad \cdots \quad \mathbf{v}_k]$$

$k$  is typically in the range of 3-10 basis functions for sound classification and spectrum summarization applications.

To calculate the proportion of information retained for  $k$  basis functions use the singular values contained in matrix  $\mathbf{S}$ :

$$I(k) = \frac{\sum_{i=1}^k S_{ii}}{\sum_{j=1}^N S_{jj}}$$

where  $I(k)$  is the proportion of information retained for  $k$  basis functions and  $N$  is the total number of basis functions which is also equal to the number of spectral bins. The SVD basis functions are stored using a `SeriesOfVector` `D` in the `AudioSpectrumBasis` `D`.

- e) **Statistically independent basis (Optional):** after extracting the reduced SVD basis,  $\mathbf{V}$ , a further step consisting of basis rotation to directions of maximal statistical independence is required for some applications. This is necessary for any application requiring maximum separation of features; for example, separation of source components of a spectrogram. A statistically independent basis is derived using an additional step of independent component analysis (ICA) after SVD extraction. The ICA basis is the same size as the SVD basis and is placed in the same `SeriesOfVector` field as the SVD basis.

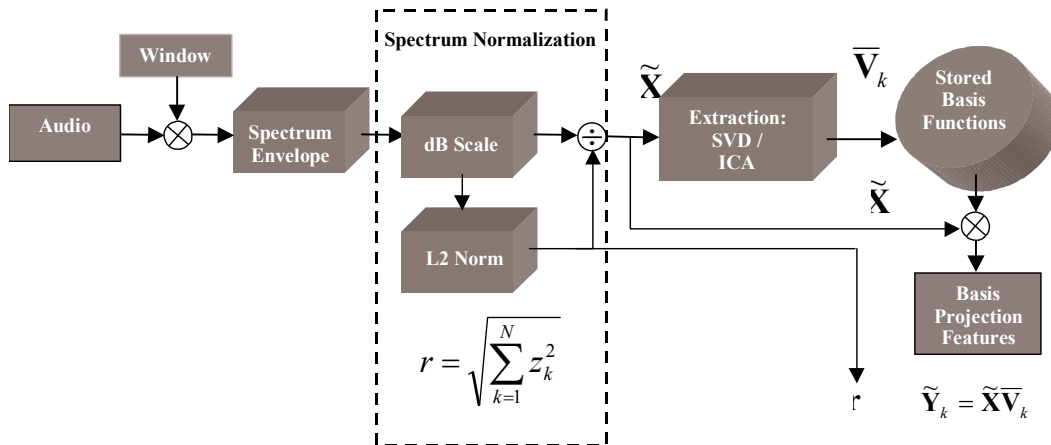


Figure 7 — Extraction method for `AudioSpectrumBasisType` and `AudioSpectrumProjectionType`

- f) *Time varying components (Optional)*: the extraction process (a)-(e) outlined above can be segmented into blocks over an `AudioSegment` thus providing a time-varying basis. To do this, the basis is sampled at regular intervals, default 500ms (`hopSize = PT500N1000F`), and a three-dimensional `SeriesOfVector` matrix results. The first dimension is the block index, the second is the spectral dimension and the third gives the number of basis vectors. This representation can track basis functions belonging to sources in an auditory scene.

### 5.3.12 AudioSpectrumProjectionType

The `AudioSpectrumProjection` D is the complement to the `AudioSpectrumBasis` D and is used to represent low-dimensional features of a spectrum after projection against a reduced rank basis. These two types are always used together. The low-dimensional features of the `AudioSpectrumProjection` D consist of a `SeriesOfVectors`, one vector for each frame,  $t$ , of the normalized input spectrogram,  $\tilde{\mathbf{X}}_t$ . Each spectral frame from steps (a)-(c) above yields a corresponding projected vector,  $\mathbf{y}_t$ , that is stored in the `SeriesOfVector` D.

#### 5.3.12.1 Syntax

```
<!-- ##### -->
<!-- Definition of AudioSpectrumProjection D -->
<!-- ##### -->
<complexType name="AudioSpectrumProjectionType">
  <complexContent>
    <extension base="mpeg7:AudioLLDVectorType"/>
  </complexContent>
</complexType>
```