An Examination and Interpretation of ITU-R BS.1387: Perceptual Evaluation of Audio Quality

P. Kabal

Department of Electrical & Computer Engineering

McGill University

Version 2: 2003-12-08

http://WWW.TSP.ECE.McGill.CA

Table of Contents

An Examination and Interpretation of ITU-R BS.1387: Perceptual Evaluation of Audio Quality

This report examines the standard which describes a method for the objective measure of perceived audio quality (ITU-R Recommendation BS.1387). This standard uses a number of psycho-acoustical measures which are combined to give a measure of the quality difference between two instances of a signal (a reference and a test signal). Many aspects of the standard are under-specified. This report examines alternate interpretations. It also looks at efficiency issues in the implementation of computationally intensive parts of the algorithm.

1 Introduction

This document examines the Perceptual Evaluation of Audio Quality (PEAQ) as described in ITU-R Recommendation BS.1387, *Method for Objective Measurements of Perceived Audio Quality* [1]. PEAQ can be used for rating the quality of, for instance, an audio coder. Additional background on PEAQ can be found in [2]. The description in BS.1387 is inadequate by itself to allow for a conforming implementation. Corrections and clarifications to BS.1387 are available [3], but they still fail to provide all of the necessary details. These have been incorporated into a Draft Revision of the standard [4].

A group of graduate students at the TSP Lab, Electrical & Computer Engineering, McGill University implemented parts of PEAQ as part of a course project. Different members of the team independently implemented a C-language and a Matlab version. The results and assumptions used for the implementations were compared and rationalized. However, in the resulting implementation still did not always give reasonable results. Part of the blame was put on ambiguous and poorly described parts of the standard. Only after the project was finished was it discovered that some of the tables in the standard had scrambled entries. (These are corrected in [4].)

At the same time, F. Baumgarte (from Bell-Labs, Lucent Technologies) and A. Lerch (now at zplane) were attempting separate implementations, but also were having trouble interpreting the intentions of the standard [5]. Lerch has implemented the code for the Basic version of PEAQ (available on-line at one time [6]).

Problems with BS.1387

The BS.1387 standard has two options: a Basic version and an Advanced version. The Basic version uses a FFT based ear model, while the Advanced version uses that model as well as a filter bank based ear model. In both cases, model output variables are combined using a trained neural network to give a single metric, the Objective Difference Grade (ODG) which measures the degradation of a test input relative to a reference input.

A standard is normally written so as to unambiguously specify the operations needed for a conforming implementation. Certain parts of BS.1387 are underspecified, so that it is not possible to choose between plausible alternatives. Some of the model output variables are poorly described. The EHS_B model output variable is a case in point. The description is ambiguous. The description of the operations for parts of the Advanced version has additional problems. Some of the specification is in the form of pseudo-code. There are errors in the pseudo-code, some of which were corrected in [4]. It remains an open question as to whether the reference implementation also contains these mistakes.

The standard gives a table of the output quality measure for a number of test cases. However, the standard does not give values for the model output variables that lead to these values. In the Basic version of PEAQ, there are 11 model output variables that are combined with a neural network. If these values had been given as part of the conformance tests, it might have been possible to disambiguate parts of the standard. *It is the opinion of this author that a conforming implementation is not possible without access to additional information, information which is not provided by the standard.*

Goals of this Document

This document attempts to rationalize the interpretation of ambiguous or poorly described sections of the standard. The goal of this present work is to use BS.1387 as a point of departure to understand the techniques employed in the standard to evaluate audio quality. The hope is that such an understanding will lead to use of parts of the PEAQ algorithm to guide design choices for audio coders, first in a conceptual sense and later perhaps even in the process of better dynamically allocating bits within the coder. One element of the examination is the efficiency of implementation of different parts of PEAQ with a view to incorporation in real-time audio coders.

2 Time to Frequency Domain (FFT-based Ear Model)

2.1 Input

The PEAQ model assumes that the input signals are sampled at 48 kHz. The test and reference signals are assumed to be time-aligned. Processing occurs for frames of 2048 samples (43 ms). The frame advance is 1024 samples, resulting in a 50% overlap of frames. The first steps in the process of converting from time samples to frequency-domain samples are the same for the test and reference signals and indeed for the channels in a stereo signal.

The sampling frequency will be denoted as F_s and the frame length as N_F. Let a frame of data contain samples $x[n]$ with n running from 0 to $N_F - 1$, inclusive.

2.2 Windowing

The frame of data is windowed with a Hann window. A basic discrete-time Hann window is given by

$$h[n, N_F] = \begin{cases} \frac{1}{2}[1 - \cos(\frac{2\pi n}{N_F - 1})], & 0 \le n \le N_F - 1, \\ 0, & \text{otherwise}, \end{cases} \tag{1}$$

where the sampling frequency has been denoted as F_s. This window is zero-valued at the endpoints $n = 0$, and $n = N_F - 1$, meaning that this window has only $N_F - 2$ non-zero terms.[1] The sum of squared window values is

$$\sum_{n=-\infty}^{\infty} h^2[n, N_F] = \frac{3}{8}(N_F - 1). \tag{2}$$

This value can be used to scale the window such that for DC or white noise, the energy per sample after windowing is the same as the energy per sample before windowing.

[1] There are three reasonable choices for the window length (in the notation of Eq. (1)), N_F, $N_F + 1$ or $N_F + 2$, each with N_F or fewer non-zero coefficients. If the window were a Hamming window (a Hann window on a pedestal), the first would be the natural choice (giving N_F non-zero values). For a Hann window, the number of non-zero samples is less by two. Consider a pure sine with a frequency which coincides with the centre of one of the DFT bins (frequency kF_s / N_F). Only the middle choice of Hann window length is free of spectral leakage for those sine frequencies. The last choice of window length gives the narrowest main lobe in the frequency response of the window.

The actual window used in the standard is a scaled version of Eq. (1),

$$h_w[n] = \sqrt{\frac{8}{3}} h[n, N_F]. \tag{3}$$

Clearly, the window scaling was meant to compensate for the energy loss due to the taper in the window. However, the scaling used is off by the factor $\sqrt{1-1/N_F}$.

2.3 Frequency Domain Coefficients

The windowed data is converted to the frequency domain using a Discrete-Fourier Transform,

$$X[k] = \frac{1}{N_F} \sum_{n=0}^{N_F-1} h_w[n]\, x[n] e^{-j2\pi nk/N_F}. \tag{4}$$

The scaling factor $1/N_F$ is unconventional (at least in the engineering signal processing literature). The DFT values are defined for $0 \le k \le N_F - 1$. However, only values for $0 \le k \le N/2$, corresponding to frequencies from 0 to $F_s/2$ (24 kHz) will be needed in the sequel.

2.4 Calibration of the Loudness Scaling Factor

Some of the perceptual quality factors depend on the actual sound pressure level of the test signal. A calibration step is needed to fix the mapping from input signal levels to loudness. The calculation of the scaling factor is discussed in Appendix A. The additional scaling factor, denoted here as G_L, (which takes into account the window scaling) is equal to

$$G_L = \frac{10^{L_p/20}}{\gamma(f_c)\frac{A_{\max}}{4}(N_F - 1)} \sqrt{\frac{3}{8}} N_F, \tag{5}$$

where L_p is the sound pressure level (SPL) in dB corresponding to a full-scale test sine. In the absence of other information, BS.1387 indicates that L_p should be set to 92 dB SPL. The parameter $A_{\max}$ is the maximum amplitude of the sine (for instance 32 768 for 16-bit data) and $\gamma(f_c)$ is a factor which varies from 0.84 to 1 depending on where the frequency of the test sine falls relative to the DFT bins. For a test frequency of 1019.5 Hz as suggested in BS.1387, $\gamma(f_c)$

is equal to 0.8497. Using these values, G_L is equal to 3.504. The last terms in the scaling factor can be omitted if one uses a standard Hann window and a standard DFT definition.

The standard states:

> "Where the normalization factor Norm is calculated by taking a sine wave of 1 109.5 Hz and 0 dB full scale as the input signal and calculating the maximum absolute value of the spectral coefficients over 10 frames."

This measurement is unnecessary, since the appropriate gain value can be calculated analytically.

The scale factors for the Hann window, the DFT and the loudness scaling factor can be lumped together if desired and applied once. Appendix F.1 gives Matlab code for the operations described above.

The calibration procedure involves setting the peak value of the DFT for a sinusoidal input to a given value. However, sound pressure level is an energy phenomenon. A more appropriate calibration would involve the total energy which by Parseval's relationship is preserved in the frequency domain. Such a calibration procedure is independent of the frequency of the test sine.

2.5 Outer and Middle Ear Modelling

The frequency response of the weighting filter as given in BS.1387 is restated here. The response of the outer and middle ear is modelled as

$$\begin{aligned} A_{\mathrm{dB}}(f_{\mathrm{kHz}}) &= -2.184(f/1000)^{-0.8} + 6.5e^{-0.6(f/1000-3.3)^2} - 0.001(f/1000)^{3.6}, \\ W(f) &= 10^{A_{\mathrm{dB}}(f)/20}. \end{aligned} \tag{6}$$

Note that at zero frequency, the response in dB is $-\infty$. This response is plotted in Fig. 1. The response at 1 kHz is -1.9 dB and the peak value of $+5.6$ dB occurs near 3.3 kHz.

The response in Eq. (6) is similar to that given by Terhardt [7]. The difference is in the first term which controls the response at low frequencies. Terhardt uses a factor 3.64 rather than 2.184 (which is given in the standard as 0.6×3.64) in the first term. Later in Section 2.16 of the standard, we meet the other part of the factor (0.4×3.64) as the contribution due to internal noise.

The vector of weights (linear scale) is given by

$$W[k] = W\left(\frac{kF_s}{N_F}\right), \qquad 0 \le k \le \frac{N_F}{2}. \tag{7}$$

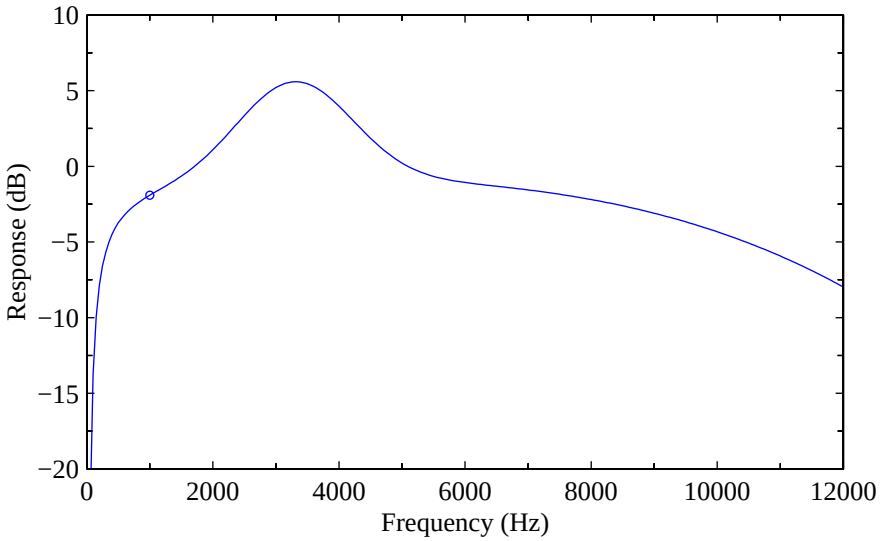

Fig. 1 Outer and middle ear response. A marker appears at 1 kHz.

Note that at zero frequency ($k = 0$), the weight is zero. Using these weights, the weighted DFT energy (including the loudness scale factor) is

$$\left|X_w[k]\right|^2 = G_L^2 W^2[k]\left|X[k]\right|^2, \qquad 0 \le k \le \frac{N_F}{2}. \tag{8}$$

2.6 Critical Band Decomposition

The grouping into critical bands uses a frequency to Bark scale conversion,

$$\begin{aligned} z &= B(f) \\ &= 7\,\mathrm{asinh}(f/650), \end{aligned} \tag{9}$$

where the units of z are Barks. The inverse mapping is

$$\begin{aligned} f &= B^{-1}(z) \\ &= 650\sinh(z/7). \end{aligned} \tag{10}$$

Frequency Bands

The processing uses frequency bands which differ in width for the Basic version and the Advanced version of PEAQ. For the Basic version, $\Delta z = \frac{1}{4}$, while for the Advanced version, $\Delta z = \frac{1}{2}$. The frequency bands start at f_L (80 Hz) and stop at f_U (18 kHz). All bands but the last have the same width in Barks.

The bands can be specified in terms of a lower frequency edge, a centre frequency and an upper frequency edge. The centre frequency is the frequency corresponding to the centre of the filter band on the Bark scale. The band values on the Bark scale are given as follows,

$$\begin{aligned} z_l[i] &= z_L + i\Delta z \\ z_u[i] &= \begin{cases} z_L + (i+1)\Delta z), & i+1 \le (z_U - z_L)/\Delta z \\ z_U, & \text{otherwise} \end{cases} \\ z_c[i] &= \frac{z_u[i] - z_l[i]}{2}, \end{aligned} \tag{11}$$

where $z_L = B(f_L)$ and $z_U = B(f_U)$. The corresponding band edges in frequency are given by using the inverse Bark mapping

$$\begin{aligned} f_l[i] &= B^{-1}(z_l[i]), \\ f_c[i] &= B^{-1}(z_c[i]), \\ f_u[i] &= B^{-1}(z_u[i]). \end{aligned} \tag{12}$$

For the Basic version, there are 109 filter bands; for the Advanced version there are 55 bands. The band edges in Hz are given to 3 decimal places in tables in BS.1387. The band edges calculated using the procedure described above agree with the tabulated values in BS.1387 to within 0.003 Hz. Appendix F.2 gives the Matlab code to calculate the critical band parameters.

Grouping of Frequency Bins

The next step of processing is to take a frame of frequency domain samples (based on DFT bins) and group them into the frequency bands defined above. The grouping is done as follows. DFT bin k corresponds to frequency $f[k] = kF_s / N_F$ and is considered to be distributed over the bin width, i.e., the bin extends $\pm F_s/(2N_F)$ from the centre frequency. BS.1387 contains pseudo-code which calculates the energy per frequency band given the energy distribution in DFT bins.

A more computationally efficient procedure is to precompute tables, obviating the need for comparisons. For the i'th frequency band, the contribution from the energy in DFT bin k is

$$U[i,k] = \frac{\max\left[0, \min(f_u[i], \frac{2k+1}{2}\frac{F_s}{N_F}) - \max(f_l[i], \frac{2k-1}{2}\frac{F_s}{N_F})\right]}{\frac{F_s}{N_F}}. \tag{13}$$

The energy in band i for a DFT-based signal is

$$E_a[i] = \sum_{k=k_l[i]}^{k_u[i]} U[i,k]\left|X_w[k]\right|^2, \tag{14}$$

where $U[i,k]$ is non-zero over the interval $k_l[i] \le k \le k_u[i]$. Note that $U[i,k]$ is equal to unity when k is strictly inside the interval, leading to the simplification[2]

$$E_a[i] = U_l[i]\left|X_w[k_l[i]]\right|^2 + \sum_{k=k_l[i]+1}^{k_u[i]-1} \left|X_w[k]\right|^2 + U_u[i]\left|X_w[k_u[i]]\right|^2, \tag{15}$$

where $U_l[i] = U[i,k_l[i]]$ and $U_u[i] = U[i,k_u[i]]$. Appendix F.3 includes Matlab code to carry out the grouping of the frequency bands.

A last step is to set the energy to 1×10^{-12} if it is less than this value,

$$E_b[i] = \max(E_a[i], E_{\min}), \tag{16}$$

where $E_{\min}$ is 1×10^{-12}.

2.7 Internal Noise

An offset is added to the band energies to compensate for internal noise generated in the ear,

$$E[i] = E_b[i] + E_{\mathrm{IN}}[i], \tag{17}$$

where the internal noise is modelled as

$$\begin{aligned} E_{\mathrm{INdB}}(f_{\mathrm{kHz}}) &= 1.456\left(f/1000\right)^{-0.8}, \\ E_{\mathrm{IN}}[i] &= 10^{E_{\mathrm{INdB}}(f_c[i])/10}. \end{aligned} \tag{18}$$

The factor 1.456 is given in the standard as 0.4×3.64, which is the missing part of the formula given by Terhardt, referred to earlier. The response is plotted in Fig. 2. At 1 kHz, the contribution is 1.46 dB.

The energies $E[i]$ are referred to as the *pitch patterns*.

[2] This formulation assumes $k_u[i] > k_l[i]$. If not then one of $U_l[i]$ or $U_u[i]$ should be set to zero.

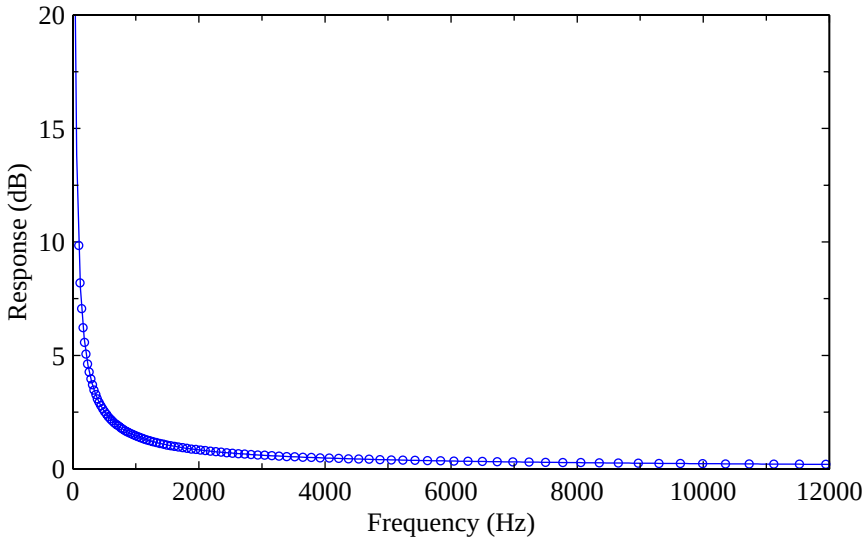

Fig. 2 Internal noise contribution. The markers indicate the centres of the frequency bands for Basic version of PEAQ.

2.8 Frequency Spreading

The spreading operation is described in Appendix B in terms of a continuous Bark spectrum. That description is converted here to the corresponding calculation on a discrete Bark scale used in BS.1387.

The spread Bark-domain energy response is

$$E_s[i] = \frac{1}{B_s[i]} \left(\sum_{l=0}^{N_c-1} \left(E[l] S(i,l,E[l]) \right)^{0.4} \right)^{\frac{1}{0.4}}, \tag{19}$$

where N_c is the number of filter bands in the fractional critical band representation (109 for the Basic version and 55 for the Advanced version). The normalizing factor is calculated for a reference level of 0 dB for each band,

$$B_s[i] = \left(\sum_{l=0}^{N_c-1} \left(S(i,l,E_0) \right)^{0.4} \right)^{\frac{1}{0.4}}, \tag{20}$$

where $E_0 = 1$ (0 dB). The normalizing factor can be pre-computed since it does not depend on the data.

The spreading function is

$$S(i,l,E) = \frac{1}{A(l,E)} 10^{S_{\mathrm{dB}}(i,l,E)/10}, \tag{21}$$

where the normalizing term is chosen to give a unit area for each centre frequency l. The spreading function in dB is triangular,[3]

$$S_{\mathrm{dB}}(i,l,E) = \begin{cases} 27(i-l)\Delta z, & i \le l, \\ [-24 - \frac{230}{f_c[l]} + 2\log_{10}(E)](i-l)\Delta z. & i \ge l, \end{cases} \tag{22}$$

The computations involved in the evaluation of the spread energy can represent a large fraction of the overall computations needed for PEAQ. However, the form of the spreading function leads to a regrouping which allows for some terms to be pre-computed and others to be computed recursively.

$$\begin{aligned} S(i,l,E) &= \begin{cases} \frac{1}{A(l,E)} (10^{2.7\Delta z})^{i-l}, & i \le l, \\ \frac{1}{A(l,E)} [(10^{-2.4\Delta z})(10^{-23\Delta z / f_c[l]})(E^{0.2\Delta z})]^{i-l}, & i \ge l, \end{cases} \\ &= \begin{cases} \frac{1}{A(l,E)} a_L^{i-l}, & i \le l, \\ \frac{1}{A(l,E)} [a_U a_C[l] a_E(E)]^{i-l}, & i \ge l, \end{cases} \end{aligned} \tag{23}$$

where the terms a_L, a_U, $a_C[l]$, and $a_E(E)$ are implicitly defined by the first part of the equation. The normalizing term $A(l,E)$ is the sum over i of $S(i,l,E)$ and can be expressed in closed form as

$$\begin{aligned} A^{-1}(l,E) &= \sum_{i=0}^{l} a_L{}^{l-i} + \sum_{i=l}^{N_c-1} (a_U a_C[l] a_E(E))^{l-i} - 1 \\ &= \frac{1 - a_L^{-(l+1)}}{1 - a_L^{-1}} + \frac{1 - (a_U a_C[l] a_E(E))^{N_c - l}}{1 - a_U a_C[l] a_E(E)} - 1 \end{aligned} \tag{24}$$

Consider splitting the computation of the spread energy response in Eq. (19) into two parts,

$$E_s[i] = \frac{1}{B_s[i]} \left(E_{sL}[i] + E_{sU}[i] \right)^{\frac{1}{0.4}}, \tag{25}$$

[3] For low frequencies, the frequency spreading due to the time windowing of the input signal is a significant fraction of the critical band width. The spreading functions can be narrowed to compensate [9].

where

$$E_{sL}[i] = \sum_{l=i}^{N_c-1} \left(\frac{E[l]}{A(l, E[l])} \right)^{0.4} \left(a_L^{0.4} \right)^{i-l}, \tag{26}$$

and

$$E_{sU}[i] = \sum_{l=0}^{i-1} \left(\frac{E[l]}{A(l, E[l])} \right)^{0.4} \left(\left(a_U a_C[l] a_E(E[l]) \right)^{0.4} \right)^{i-l}. \tag{27}$$

Further computational reorganization can be done for the lower part of the spreading function,

$$\begin{aligned} E_{sL}[N_c - 1] &= \left(\frac{E[N_c - 1]}{A(l, E[N_c - 1])} \right)^{0.4}, \\ E_{sL}[i] &= a_L^{-0.4} E_{sL}(i+1) + \left(\frac{E[i]}{A(l, E[i])} \right)^{0.4} \quad i = N_c - 2, \ldots, 0. \end{aligned} \tag{28}$$

The term which includes the upper part of the spreading function is not quite so amenable to simplification. Some computational savings accrue if we compute the power term recursively.

The computational procedure for creating the excitation patterns is shown in Appendix F.4. The normalization term $B_s[i]$ is plotted in Fig. 3. Note the change in value at the ends of the spectrum. The spreading functions themselves (normalized by both $B_s[i]$ and $A(l, E)$) are shown in Fig. 4. The plot shows every fourth spreading function (a spacing of 1 critical band for the Basic version of PEAQ) for a signal level of 60 dB SPL.

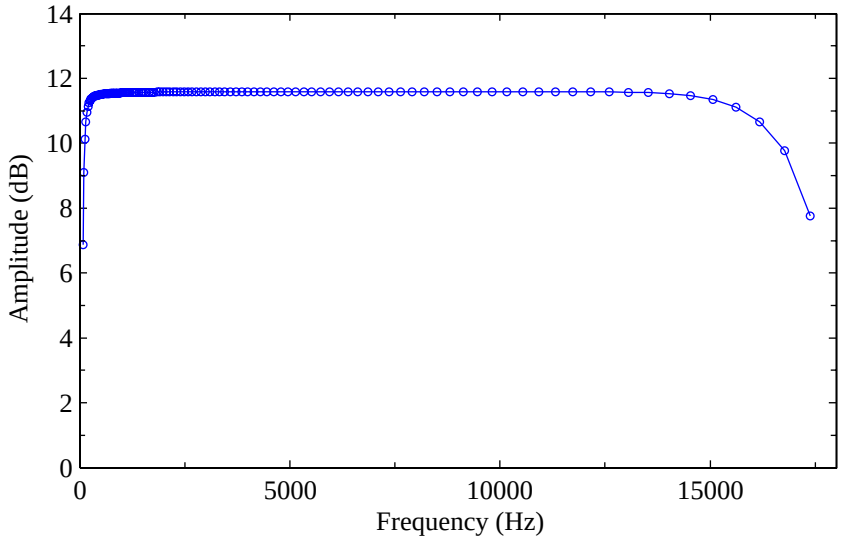

Fig. 3 Normalization factor $B_s[i]$ for the Basic version of PEAQ.

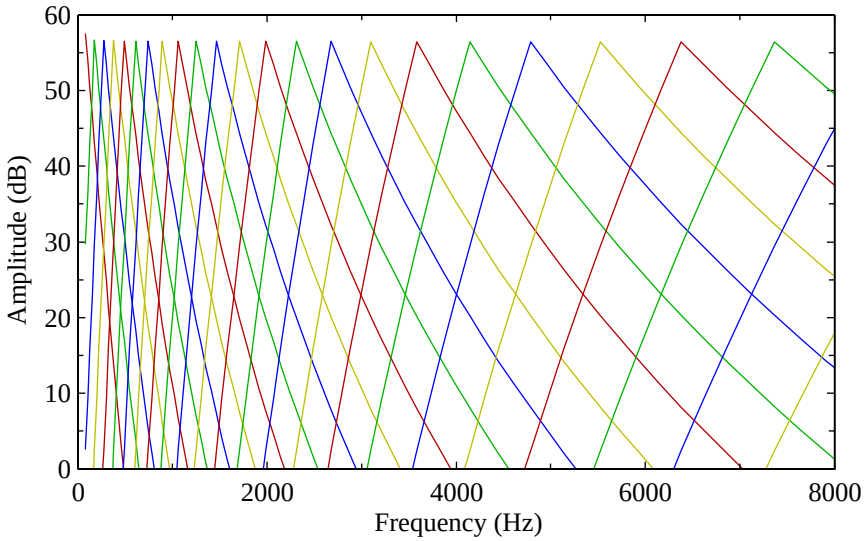

Fig. 4 Spreading functions (60 dB signal level). Every fourth spreading function is plotted for the Basic version of PEAQ.

The excitation patterns derived ($E_s[i]$) are referred to as *unsmeared excitation patterns* (unsmeared in time). They will be used in calculation of modulation patterns.

2.9 Time Domain Spreading

The formulation until now has been based entirely on processing a single frame. We now introduce a time spreading which depends on multiple frames. For this purpose we add a *frame* index n to the spread energy in the Bark domain, $E_s[i,n]$. Frames are updated every $N_F/2$ samples. The frame rate is

$$F_{ss} = \frac{F_s}{N_F/2}. \tag{29}$$

To model forward masking, a frequency dependent filtering (smearing) over time is implemented,

$$\begin{aligned} E_f[i,n] &= \alpha[i]E_f[i,n-1] + (1-\alpha[i])E_s[i,n], \\ \tilde{E}_s[i,n] &= \max(E_f[i,n], E_s[i,n]). \end{aligned} \tag{30}$$

The parameter $\alpha[i]$ controls the time constant of the averaging for decaying energies. The $\max(\cdot)$ function in the second line means that $E[i,n]$ follows increases in energy instantaneously.

The initial condition for the filtering is given as $E_f[i,0]=0$. We assume that frames are indexed from $n=0$ for consistency with the sequel. Then, the initial condition is actually $E_f[k,-1]=0$.

The outputs $\tilde{E}_s[i,n]$ are known as *excitation patterns*.

2.9.1 Time Constants

The time constant of the first order difference equation (in frames) is $-1/\log(\alpha[i])$. The time constant in seconds is

$$\tau[i] = -\frac{1}{F_{ss}\log(\alpha[i])}, \tag{31}$$

where F_{ss} is the frame rate. The time constant for filter band i is specified as,

$$\tau[i] = \tau_{\min} + \frac{100}{f_c[i]}(\tau_{100} - \tau_{\min}), \tag{32}$$

where $\tau_{100} = 0.030$ s and $\tau_{\min} = 0.008$ s. Then $\alpha[i]$ can be calculated from

$$\alpha[i] = \exp(-\frac{1}{F_{ss}\tau[i]}). \tag{33}$$

The time constant at 100 Hz is τ_{100}. The lowest centre frequency is only slightly below 100 Hz. The smallest time constant occurs at the highest centre frequency and is close to $\tau_{\min}$. The time constants are plotted in Fig. 5.

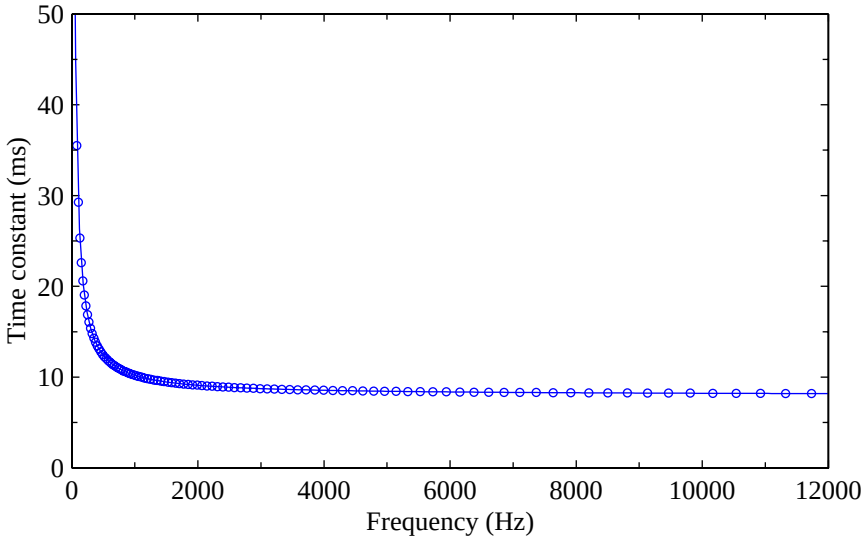

Fig. 5 Time constants as a function of frequency for $\tau_{\min} = 0.008$ s and $\tau_{100} = 0.030$ s. The markers indicate the centres of the frequency bands for the Basic version of PEAQ

3 The Filter Bank Ear Model

The Advanced version of PEAQ uses a Filter bank ear model as well as the FFT-based model. The scaling for the input to the filter bank is given assuming that full scale for the input is $A_{\max} = 32\,767$. More generally, the scaling applied to the input is,

$$g = 10^{L_p/20}\frac{A_{\max}}{32767}, \tag{34}$$

where in the absence of other information, L_p is assumed to be 92 dB SPL. If $A_{\max}$ is 32 767 (16-bit data), and $L_p = 92$ dB SPL, $g = 1.2150$.

3.1 DC Rejection Filter

The signal is then passed through a 4'th order DC rejection filter to remove subsonic signal components. This filter is a Butterworth highpass filter with a cutoff of 20 Hz and is realized as the cascade of two second order IIR sections. The filter response is derived in Appendix C. The magnitude of the frequency response is plotted in Fig. 6.

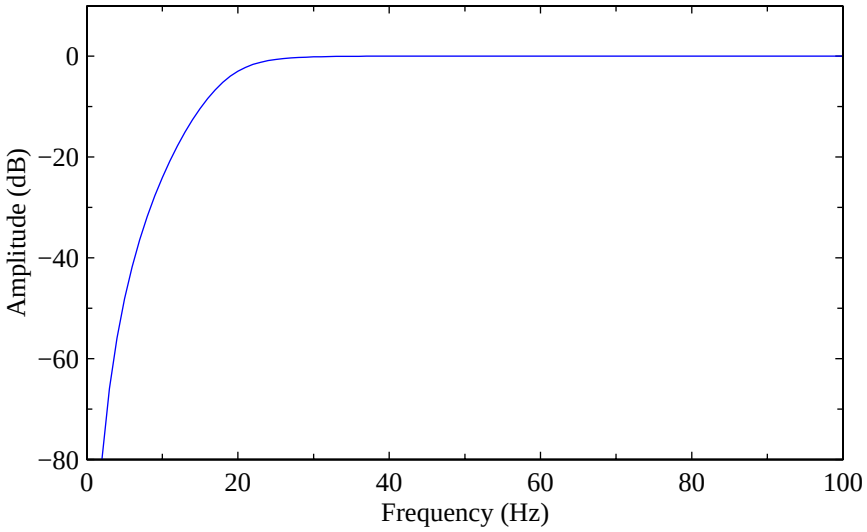

Fig. 6 Frequency response of a 4'th order Butterworth highpass filter with cutoff at 20 Hz.

Implementing the filter with second-order sections,

$$\begin{aligned} x_a[n] &= -a_{01}x_a[n-1] - a_{02}x_a[n-2] + x[n] - 2x[n-1] + x[n-2], \\ x_{hp}[n] &= -a_{11}x_{hp}[n-1] - a_{12}x_{hp}[n-2] + x_a[n] - 2x_a[n-1] + x_a[n-2]. \end{aligned} \tag{35}$$

Several memory saving approaches can be used for implementing the highpass filter. Each section uses the current and two previous inputs, and two previous outputs. Noting that the output of one section is the input to the next, some of this memory can be shared to give an efficient implementation.

3.2 Filter Bank

The filter bank uses bandpass filters at 40 centre frequencies ranging from $f_L = 50$ Hz to $f_U = 18000.02$ Hz. The centre frequencies are equally-spaced on the Bark scale with a spacing given by

$$\Delta z = \frac{B(f_U) - B(f_L)}{N_c - 1}. \tag{36}$$

The centre frequencies are[4]

$$\begin{aligned} z_c[k] &= B(f_L) + k\Delta z, && 0 \le k \le N_c - 1, \\ f_c[k] &= B^{-1}(z_c[k]) && 0 \le k \le N_c - 1. \end{aligned} \tag{37}$$

For each centre frequency, there is a pair of linear phase FIR filters, an in-phase filter and a quadrature filter,

$$\begin{aligned} h_I[k,n] &= h_p[k,n]\cos\left(2\pi \tfrac{f_c[k]}{F_s}(n - \tfrac{N_k}{2})\right), && 0 \le n \le N_k - 1, \\ h_Q[k,n] &= h_p[k,n]\sin\left(2\pi \tfrac{f_c[k]}{F_s}(n - \tfrac{N_k}{2})\right), && 0 \le n \le N_k - 1, \end{aligned} \tag{38}$$

where the lowpass prototype for filter k is

$$h_p[k,n] = \frac{4}{N_k}\sin^2(\pi \frac{n}{N_k}), \qquad 0 \le n \le N_k - 1. \tag{39}$$

The prototypes have lengths as tabulated in BS.1387. See Appendix D for a further discussion of the filter responses. The response of the filter bank is shown in Fig. 7. The figure shows the superimposed responses of the in-phase filters. The abscissa has been warped to the Bark scale, but is labelled with frequency in Hz. Note that the lowest frequency bandpass filter has a significant response down to DC. However, the subsonic components up to 20 Hz have already been removed by the DC rejection filter.

[4] The results when rounded to two decimal places differ by at most 0.01 Hz from the values tabulated in BS.1387.

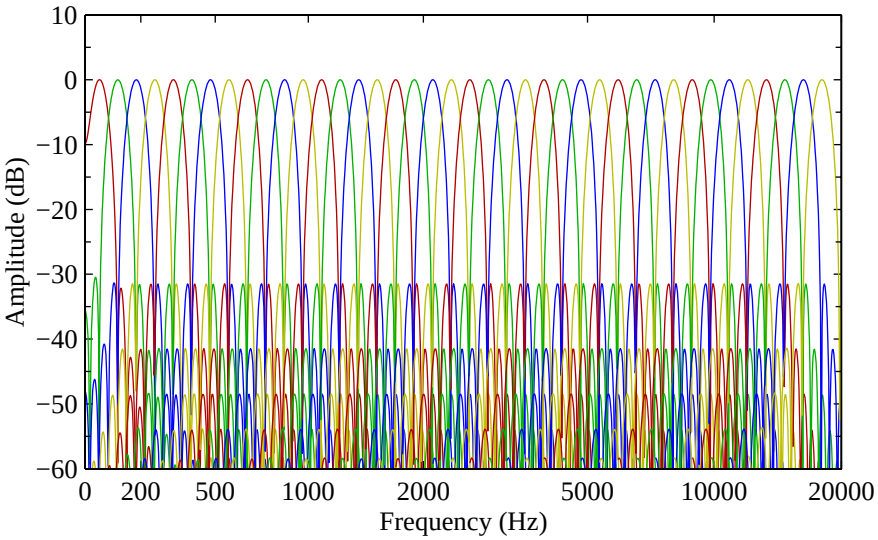

Fig. 7 Superimposed filter bank responses. The frequency axis is linear on a Bark scale.

Note that although the filters are specified to have an even number of coefficients, the first coefficient of each filter is zero. Thus they are effectively of length $N_k - 1$. There is a "middle" coefficient which is used as a time reference for the modulation terms in Eq. (38). The filters are aligned in time relative to the middle of each filter. This involves artificially adding delay to the shorter branches of the filter bank.

$$\begin{aligned} x_I[k,n] &= \sum_{m=0}^{N_k-2} \tilde{h}_I[k,m]\, x_{hp}[I_S n - m - D_k], \\ x_Q[k,n] &= \sum_{m=0}^{N_k-2} \tilde{h}_Q[k,m]\, x_{hp}[I_S n - m - D_k], \end{aligned} \tag{40}$$

where the filters re-indexed to omit the first (zero-valued) coefficient and scaled by g,

$$\begin{aligned} \tilde{h}_I[k,n] &= g\, h_I[k,n+1] \qquad 0 \le n \le N_k - 2, \\ \tilde{h}_Q[k,n] &= g\, h_Q[k,n+1] \qquad 0 \le n \le N_k - 2. \end{aligned} \tag{41}$$

The output is subsampled by the factor $I_S = 32$.

The centre coefficient in the re-indexed filters is at index $N_k / 2 - 1$. For the longest filter ($k = 0$), the delay is set to zero, $D_0 = 0$. The first coefficient of this filter aligns with $x_{hp}[n]$ and the middle of this filter aligns with $x_{hp}[n - N_0 / 2 + 1]$. The delays used to align the middle of the other filters with this same sample are

$$D_k = \frac{N_0 - N_k}{2}. \tag{42}$$

The standard indicates that the reference implementation adds one to the delay of each branch (including the branch with the longest filter).

The output of each of the filters has a sampling rate of 1500 Hz in each channel. As noted in the standard, this reduced sampling rate too small for the wider upper bands — some aliasing will be inevitable for those bands.

3.3 Outer and Middle Ear Modelling

The frequency response of the outer and ear model used earlier (Eq. (6)) evaluated at the center frequencies of the filters is used to weight the filter outputs,

$$\begin{aligned} x_{Iw}[k,n] &= W(f_c[k])\, x_I[k,n], \\ x_{Qw}[k,n] &= W(f_c[k])\, x_Q[k,n]. \end{aligned} \tag{43}$$

3.4 Frequency Domain Spreading

The filter bank outputs are spread in frequency. The spreading function is level and frequency dependent. The spreading function is similar to that encountered earlier for the FFT model. There are differences in the slopes of the function and the fact that it is applied in the magnitude domain instead of the power domain raised to the 0.4 exponent.

The instantaneous spreading function in dB is

$$S_{\mathrm{dB}}(z, z_c, E) = \begin{cases} 31(z - z_c), & z \le z_c, \\ \min[-4, -24 - \frac{230}{B^{-1}(z_c)} + 2\log_{10}(E),](z - z_c), & z_c \le z, \end{cases} \tag{44}$$

where E is the energy at the centre frequency. The instantaneous spreading function (in the linear domain) is

$$S(i,l,E) = 10^{S_{dB}(z_c[i], z_c[l], E)/20}. \tag{45}$$

Time Smoothing of the Spreading Function

The spreading function changes in response to changes in energy. The spreading function is smoothed to reduce these variations,

$$\tilde{S}[i,l,n] = \alpha\, \tilde{S}[i,l,n-1] + (1-\alpha)\, S(i,l,E[l,n]), \tag{46}$$

where $E[l,n]$ is the energy in band l at time n,

$$E[l,n] = X_{Iw}^2[l,n] + X_{Qw}^2[l,n]. \tag{47}$$

The smoothing parameter corresponds to a time constant of 100 ms ($\tau = 0.1$ s),

$$\alpha = \exp(-\frac{1}{F_{ss}\tau}), \tag{48}$$

where $F_{ss} = F_s / 32$ is the sampling rate at the output of the filter bank.

The smoothing of the spreading function with time is defined by pseudo-code in the standard. The pseudo-code is inconsistent with the text description. As implemented in the pseudo-code, the roles of α and $1-\alpha$ are interchanged, resulting in an extremely short time constant (58 µs).

The computations for frequency spreading can be simplified using an approach analogous to that developed earlier for the FFT model spreading. In fact the pseudo-code in the standard implements this type of recursion.

Spreading Applied to the Filter Outputs

The spreading function is applied to the in-phase and quadrature components in each band separately,

$$\begin{aligned} \tilde{x}_{Iw}[k,n] &= \sum_{l=0}^{N_c-1} x_{Iw}[l,n]\tilde{S}[k,l,n], \\ \tilde{x}_{Qw}[k,n] &= \sum_{l=0}^{N_c-1} x_{Qw}[l,n]\tilde{S}[k,l,n]. \end{aligned} \tag{49}$$

Finally, the energy of each channel is computed,

$$E_0[k,n] = \left(\tilde{x}_{Iw}[k,n]\right)^2 + \left(\tilde{x}_{Qw}[k,n]\right)^2. \tag{50}$$

3.5 Backward Masking

The frequency-spread energies are time-smeared with an FIR filter. The output of the filter is subsampled by a factor $I_D = 6$,

$$E_b[k,m] = c\sum_{i=0}^{N_B-1} h_B[i]E_0[k,mI_D - i], \tag{51}$$

where the constant c has the value 0.9761/6.[5] The filter $h_B[i]$ has $N_B = 12$ coefficients,

$$h_B[i] = \begin{cases} \cos^2\left(\pi \dfrac{i - (N_B/2 - 1)}{N_B}\right) & 0 \le n \le N_B - 1, \\ 0 & \text{otherwise.} \end{cases} \tag{52}$$

The filter pulse response is zero at $n = N_b - 1$, and so has only 11 non-zero coefficients.

The output sampling rate is $F_s/192$ for each channel.

3.6 Internal Noise

Internal noise is added to each band. The internal noise function is given by Eq. (18) evaluated at the centre frequencies of the filters. The result is

$$E_s[k,n] = E_b[k,n] + E_{\text{IN}}[k,n]. \tag{53}$$

These are the *unsmeared excitation patterns* for the filter bank model.

3.7 Forward Masking

Forward masking is implemented with a first order filter

$$\tilde{E}_s[k,n] = \alpha[k]\tilde{E}_s[k,n-1] + (1 - \alpha[k])E_c[k,n], \tag{54}$$

where the difference equation coefficients are calculated from the time constants $\tau_{100} = 0.020$ s and $\tau_{\min} = 0.004$ s ($F_{ss} = F_s/192$, see Section 2.9.1)[6]. The values after applying forward masking are the *excitation patterns* for the filter bank model.

---

[5] The origin of the value for the constant c is not clear. The filter acts on an energy signal. If the input is constant, then setting $c = 2/N_b$ preserves the energy level at the output.

[6] In [2], the time constant at 100 Hz is given as 50 ms.

4 Pattern Processing

The outputs of the FFT and filter bank blocks are further processed. For this discussion, there are two input signals and the corresponding outputs: the reference signal and the test signal. Subscripts R and T will refer to signals derived from the reference and test signals, respectively. The case of binaural signals will be discussed later in connection with the model output variables. For the purposes of this section, one can consider the two channels (left and right, for instance) to be separate signals for which there are reference and test instances.[7]

The processing the FFT and filter bank outputs can be considered together. The FFT outputs occur at a rate of $F_s/1024$, while the filter bank outputs occur at a rate of $F_s/192$. The number of centre frequencies for the FFT model is 109 for the Basic version and 55 for the Advanced version. For the filter bank (used in the Advanced version only), there are 40 centre frequencies. These parameters are summarized in Table 1.

Table 1 Processing parameters

PEAQ Version	Model	Sampling Rate F_{ss}	No. Centre Frequencies N_c
Basic	FFT	$F_s/1024$	109
Advanced	FFT	$F_s/1024$	55
	Filter Bank	$F_s/192$	40

For the Advanced version of PEAQ, processing has to occur at two rates. For every 3 outputs from the FFT model, there are 16 outputs from the filter bank.

The signals produced by the FFT model and the filter bank model are shown in Fig. 8. The figure also shows the sampling rate at different points of the processing.

[7] For the sequel, unless otherwise indicated, the index k represents frequency band and the index n represents a frame count.

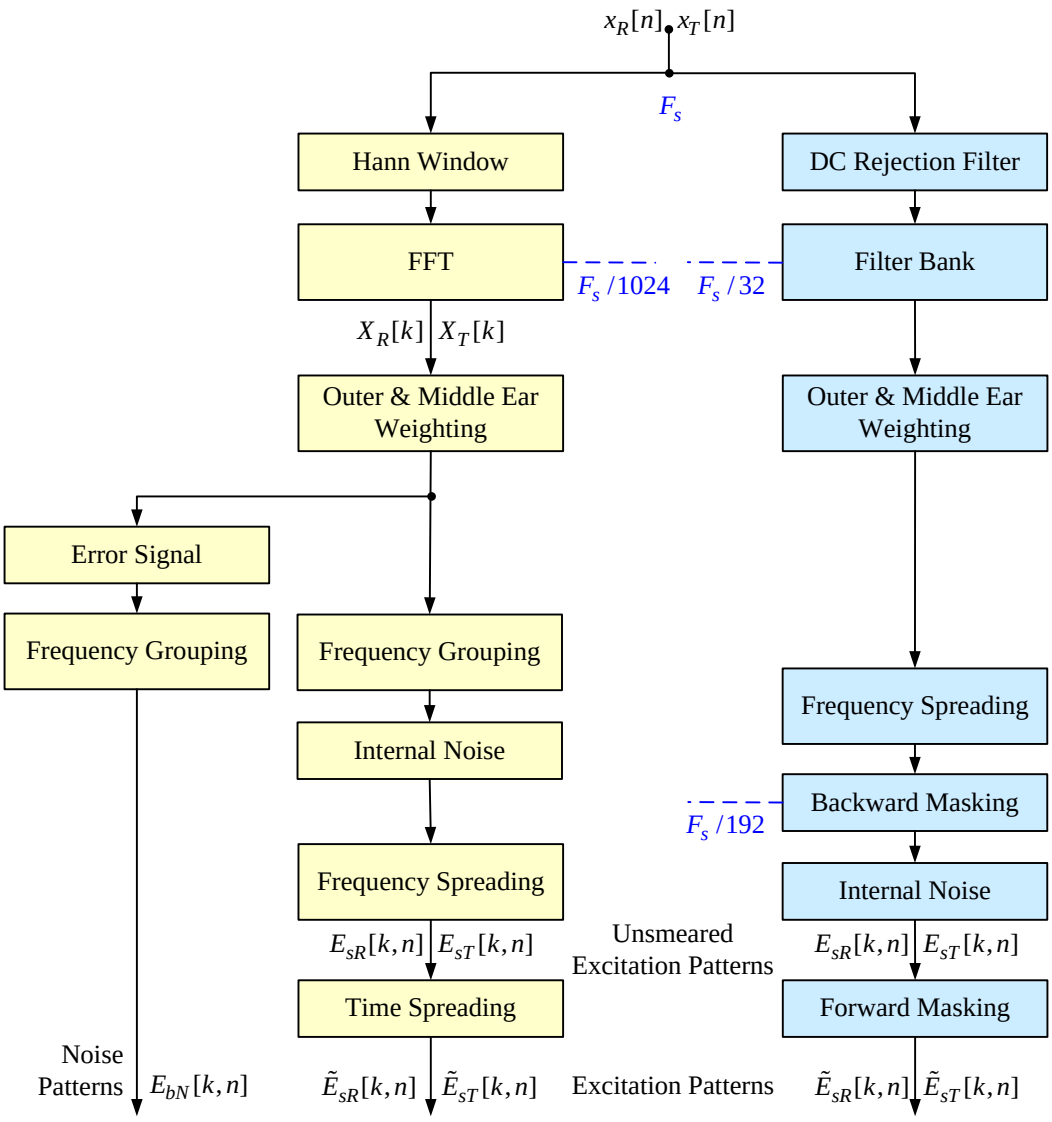

Fig. 8 Output signals from the FFT model and the filter bank model. Subscripts R and T will refer to signals derived from the reference and test signals, respectively.

4.1 Excitation Pattern Processing

The inputs are the excitation patterns (frequency spread and time smoothed): $\tilde{E}_{sR}[k,n]$ and $\tilde{E}_{sT}[k,n]$. These are functions of frequency and time. Interpretations of the following operations are given in [2,10].

Time Domain Spreading

The excitation patterns are time-spread again with a frequency-dependent time constant,

$$\tau[k] = \tau_{\min} + \frac{100}{f_c[k]}(\tau_{100} - \tau_{\min}),$$
$$\alpha[k] = \exp(-\frac{1}{F_{ss}\tau[k]}), \tag{55}$$

where the time constants are determined from $\tau_{100} = 0.050$ s and $\tau_{\min} = 0.008$ s, and F_{ss} is the sampling rate.[8] The preamble for the section on pattern processing in the draft revision to BS.1387 states:

> "If not given otherwise, all variables and recursive filters are initialized to zero."

The use of the phrase "if not given otherwise", would lead one to expect that some filtering operations are *not* initialized to zero. There is no example of the "otherwise" in this standard (but see the comments on initialization with respect to pattern adaptation).

Time-spreading occurs independently for each frequency band and separately for the reference and test signals,

$$P_R[k,n] = \alpha[k]P_R[k,n-1] + (1-\alpha[k])\tilde{E}_{sR}[k,n],$$
$$P_T[k,n] = \alpha[k]P_T[k,n-1] + (1-\alpha[k])\tilde{E}_{sT}[k,n]. \tag{56}$$

The initial conditions for this filtering are zero. These signals are used to adjust the levels of the reference and test signals.

The momentary correction factor averaged across frequency bands is

$$C_L[n] = \left(\frac{\sum_{k=0}^{N_c-1} \sqrt{P_T[k,n]P_R[k,n]}}{\sum_{k=0}^{N_c-1} P_T[k,n]} \right)^2. \tag{57}$$

Note that the denominator is guaranteed to be positive since an energy floor (FFT model) and an internal noise term was added. The excitation patterns are level corrected as follows,

[8] Note that τ_{100} is different from the value used earlier for time spreading in the FFT model.

$$E_{LR}[k,n]=\begin{cases}\tilde{E}_{sR}[k,n]/C_L[n] & C_L[n]>1,\\ \tilde{E}_{sR}[k,n] & C_L[n]\le 1,\end{cases}$$
$$E_{LT}[k,n]=\begin{cases}\tilde{E}_{sT}[k,n] & C_L[n]>1,\\ \tilde{E}_{sT}[k,n]C_L[n] & C_L[n]\le 1.\end{cases} \quad (58)$$

Pattern Adaptation

The outputs are further smoothed. First a time-smoothed correlation between the reference and test patterns for each frequency band is calculated. The numerator and denominator of the correlation term are (using the same time constants as earlier and zero initial conditions[9]),

$$R_n[k,n]=\alpha[k]R_n[k,n-1]+E_{LT}[k,n]E_{LR}[k,n],$$
$$R_d[k,n]=\alpha[k]R_d[k,n-1]+E_{LR}[k,n]E_{LR}[k,n]. \quad (59)$$

The ratio of these terms is used to form a pair of auxiliary signals which takes on values between 0 and 1,

$$R_R[k,n]=\begin{cases}1 & R_n[k,n]\ge R_d[k,n],\\ \frac{R_n[k,n]}{R_d[k,n]} & R_n[k,n]<R_d[k,n],\end{cases}$$
$$R_T[k,n]=\begin{cases}\frac{R_d[k,n]}{R_n[k,n]} & R_n[k,n]\ge R_d[k,n],\\ 1 & R_n[k,n]<R_d[k,n].\end{cases} \quad (60)$$

Special cases occur if the denominator term is zero.

- If $R_d[k,n]$ is zero and $R_n[k,n]$ is not zero, $R_T[k,n]$ is set to zero and $R_R[k,n]$ is set to one. This condition is automatically taken into account when the conditions are expressed as in Eq. (60).
- If both denominator and numerator are zero, the values are copied from the frequency below ($R_T[k,n]=R_T[k-1,n]$ and $R_R[k,n]=R_R[k-1,n]$).
- If there is no band below ($k=0$), then both $R_T[k,n]$ and $R_R[k,n]$ are set to one.

[9] This expression does not use the factor $(1-\alpha[k])$ in front of the second term in each line. Such a factor would modify the scaling of each term and would ultimately cancel when the ratio is taken.

The tests are unnecessary. The terms in the calculations are positive since an energy floor was imposed during the grouping into frequency bins (FFT model) and an internal noise term was added (FFT model and filter bank model).

The auxiliary signals are smoothed in time and frequency, to form *pattern correction factors* (same time constants; for initialization, see the comments below),

$$\begin{aligned} P_{CR}[k,n] &= \alpha[k]P_{CR}[k,n-1] + (1-\alpha[k])R_{aR}[i,n], \\ P_{CT}[k,n] &= \alpha[k]P_{CT}[k,n-1] + (1-\alpha[k])R_{aT}[i,n], \end{aligned} \tag{61}$$

where the frequency smoothed terms are

$$\begin{aligned} R_{aR}[k,n] &= \frac{1}{M_1[k]+M_2[k]+1}\sum_{i=k-M_1[k]}^{k+M_2[k]} R_R[i,n], \\ R_{aT}[k,n] &= \frac{1}{M_1[k]+M_2[k]+1}\sum_{i=k-M_1[k]}^{k+M_2[k]} R_T[i,n]. \end{aligned} \tag{62}$$

The frequency smoothing interval is nominally from $k-M_1$ to $k+M_2$, but is corrected at the lower and upper frequency bands,

$$M_1[k] = \min(M_1, k), \qquad M_2[k] = \min(M_2, N_c - 1 - k). \tag{63}$$

The parameters differ depending on the version and model used as shown in Table 2.

Table 2 Frequency smoothing parameters

PEAQ Version	Model	Sampling Rate F_{ss}	M_1	M_2	No. Centre Frequencies N_c
Basic	FFT	F_s /1024	3	4	109
Advanced	FFT	F_s /1024	1	2	55
	Filter Bank	F_s /192	1	1	40

Finally the pattern correction factors are applied to give the *spectrally adapted patterns*,

$$\begin{aligned} E_{PT}[k,n] &= E_{LT}[k,n]P_{CT}[k,n], \\ E_{PR}[k,n] &= E_{LR}[k,n]P_{CR}[k,n]. \end{aligned} \tag{64}$$

These spectrally adapted excitation patterns are the final outputs of this stage of processing.

Given the comments in the preamble to the description of the pre-processing of the excitation patterns in the draft revision to BS.1387, the initial conditions for the pattern correction fac-

tors are zero, since there is no information to the contrary. However, perhaps a more appropriate initialization for the pattern correction factors ($P_{CR}[k,n]$ and $P_{CT}[k,n]$) is unity. In fact, Lerch uses this initialization [6]. The spectrally adapted excitation patterns are used for the noise loudness calculations. In that computation, there is a 0.5 s delay in averaging the values. With this delay, the effect of the initial conditions will be minimal.

4.2 Modulation Pattern Processing

The unsmeared excitation patterns (spread in frequency, but not in time) are the inputs to this calculation. The goal is to compute averages and average differences in an approximate loudness domain (0.3 power domain). The average loudness is

$$\begin{aligned}\bar{E}_R[k,n] &= \alpha[k]\bar{E}_R[k,n-1] + (1-\alpha[k])\left(E_{sR}[k,n]\right)^{0.3},\\ \bar{E}_T[k,n] &= \alpha[k]\bar{E}_T[k,n-1] + (1-\alpha[k])\left(E_{sT}[k,n]\right)^{0.3}.\end{aligned} \tag{65}$$

The average loudness differences is

$$\begin{aligned}\bar{D}_R[k,n] &= \alpha[k]\bar{D}_R[k,n-1] + (1-\alpha[k])\,F_{ss}\left|\left(E_{sR}[k,n]\right)^{0.3} - \left(E_{sR}[k,n-1]\right)^{0.3}\right|,\\ \bar{D}_T[k,n] &= \alpha[k]\bar{D}_T[k,n-1] + (1-\alpha[k])\,F_{ss}\left|\left(E_{sT}[k,n]\right)^{0.3} - \left(E_{sT}[k,n-1]\right)^{0.3}\right|.\end{aligned} \tag{66}$$

The time constants are the same as used in the previous section. Zero initial conditions apply. These loudness estimates are combined to form a measure for the modulation of the envelope (at each frequency),

$$\begin{aligned}M_R[k,n] &= \frac{\bar{D}_R[k,n]}{1+\bar{E}_R[k,n]/0.3},\\ M_T[k,n] &= \frac{\bar{D}_T[k,n]}{1+\bar{E}_T[k,n]/0.3}.\end{aligned} \tag{67}$$

These modulation parameters, as well as the average loudness of the reference signal ($\bar{E}_R[k,n]$), will be used later in the calculation of the modulation differences.

4.3 Loudness Calculation

The loudness of the signals is used later to select frames to be included in the noise loudness model output variables. The specific loudness patterns are (from [11])

$$N_R[k,n] = c\left(\frac{E_t[k]}{s[k]E_0}\right)^{0.23}\left[\left(1 - s[k] + \frac{s[k]\tilde{E}_{sR}[k,n]}{E_t[k]}\right)^{0.23} - 1\right],$$
$$N_T[k,n] = c\left(\frac{E_t[k]}{s[k]E_0}\right)^{0.23}\left[\left(1 - s[k] + \frac{s[k]\tilde{E}_{sT}[k,n]}{E_t[k]}\right)^{0.23} - 1\right], \quad (68)$$

where the threshold index is

$$s_{\mathrm{dB}}(f) = -2 - 2.05\,\mathrm{atan}\left(\frac{f}{4000}\right) - 0.75\,\mathrm{atan}\left(\left(\frac{f}{1600}\right)^2\right),$$
$$s[k] = 10^{s_{\mathrm{dB}}(f_c[k])/10}, \quad (69)$$

and the excitation threshold is

$$E_{t\mathrm{dB}}(f) = 3.64\left(f/1000\right)^{-0.8},$$
$$E_t[k] = 10^{E_{t\mathrm{dB}}(f_c[k])/10}. \quad (70)$$

The threshold index is the ratio of the intensity of a just-audible test tone and the intensity of the internal noise within a critical band. The form used is attributed to Kapust, see [10]. The excitation threshold in quiet is the low frequency part of the outer and middle ear filtering and internal noise terms that appeared earlier. The excitation threshold and the threshold index are plotted in Fig. 9. At 1 kHz, the threshold index is about -3 dB.

The total loudness is the sum of the specific loudness patterns,

$$N_{\mathrm{tot}R}[n] = \frac{24}{N_c}\sum_{k=0}^{N_c-1}\max(N_R[k,n],0),$$
$$N_{\mathrm{tot}T}[n] = \frac{24}{N_c}\sum_{k=0}^{N_c-1}\max(N_T[k,n],0). \quad (71)$$

The factor 24 is the total number of barks in an audio signal. The term $24/N_c$ is approximately the width of each frequency band.

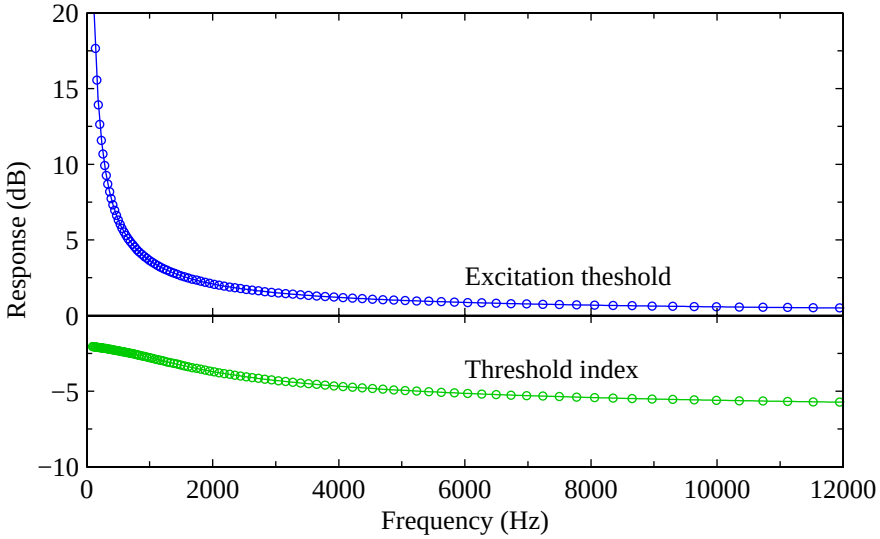

Fig. 9 Excitation threshold and threshold index. The markers indicate the centres of the frequency bands for the Basic version of PEAQ.

The total loudness is in units of sone. A 40 dB SPL sine at 1 kHz should give an output of 1 sone. To get this relationship, E_0 is set to 10^4 (40 dB relative to 0 dB SPL), and c is set equal to 1.07664 for the FFT-based ear model and equal to 1.26539 for the filter bank model.

The above setup does not predict the loudness of a sine. If we input a 1 kHz sine corresponding to 40 dB SPL, the total loudness is 0.584 sone. The sine wave amplitude for the 1 kHz sine was set to $A_{\max} 10^{L_p/20} / 10^{40/20}$, where L_p is the calibration sound pressure level (92 dB) and $A_{\max}$ is the peak amplitude of the 92 dB SPL calibration sine. Increasing the energy of the sine by a factor of 10 (sine amplitude multiplied by $\sqrt{10}$) should increase the loudness by 1 sone. In fact the total loudness increases by 0.770 sone. This is consistent with the fact that exponent (0.23) has been tuned for uniform exciting noise (noise with same energy in each critical band).

5 Calculation of the Model Output Variables

The outputs of the previous steps are generally functions of time and frequency for the reference signal and the test signal. Next these functions are distilled into functions of time. Finally, these functions of time are averaged to give a single value, the model output variable (MOV).

The processing parameters that differ between versions and models are given again in the table below.

Table 3 Processing parameters

PEAQ Version	Model	Sampling Rate F_{ss}	No. Centre Frequencies N_c
Basic	FFT	F_s /1024	109
Advanced	FFT	F_s /1024	55
	Filter Bank	F_s /192	40

All of the 11 model output variables used in the Basic version are derived from the FFT model. The model output variables are named in Table 4.

Table 4 Model Output Variables – PEAQ Basic

Model Output Variable	Model	Description
$BandwidthRef_B$	FFT	Bandwidth of the reference signal
$BandwidthTest_B$	FFT	Bandwidth of the test signal
$Total\ NMR_B$	FFT	Noise-to-mask ratio
$WinModDiff1_B$	FFT	Windowed modulation difference
ADB_B	FFT	Average block distortion
EHS_B	FFT	Harmonic structure of the error
$AvgModDiff1_B$	FFT	Average modulation difference
$AvgModDiff2_B$	FFT	Average modulation difference
$RmsNoiseLoud_B$	FFT	Distortion loudness
$MFPD_B$	FFT	Maximum filtered probability of detection
$RelDistFrames_B$	FFT	Relatively disturbed frames

In the Advanced version, there are 5 model output variables, some of which are derived from the FFT model and the rest come from the filter bank model. For the Advanced version the situation is as shown in Table 5.

Table 5 Model Output Variables – PEAQ Advanced

Model Output Variable	Model	Description
$RmsModDiff_A$	Filter Bank	Modulation changes
$RmsNoiseLoudAsym_A$	Filter Bank	Distortion loudness
Segmental NMR_B	FFT	Noise-to-mask ratio
EHS_B	FFT	Harmonic structure of the error
$AvgLinDist_A$	Filter Bank	Linear distortions

The Advanced version uses two MOV's from the FFT-based model, *Segmental* NMR_A which is used only in the Advanced version and EHS_B which is used in both versions.

For binaural signals, for most MOV's, the calculation is done separately for channel 1 and channel 2. The corresponding MOV's for the channels are then averaged. For two MOV's used in the Basic version, $MFPD_B$ and ADB_B, the channels are combined frequency by frequency before time averaging.

The figure below shows the inputs and outputs for the pattern processing of the previous section and which are used in this section.

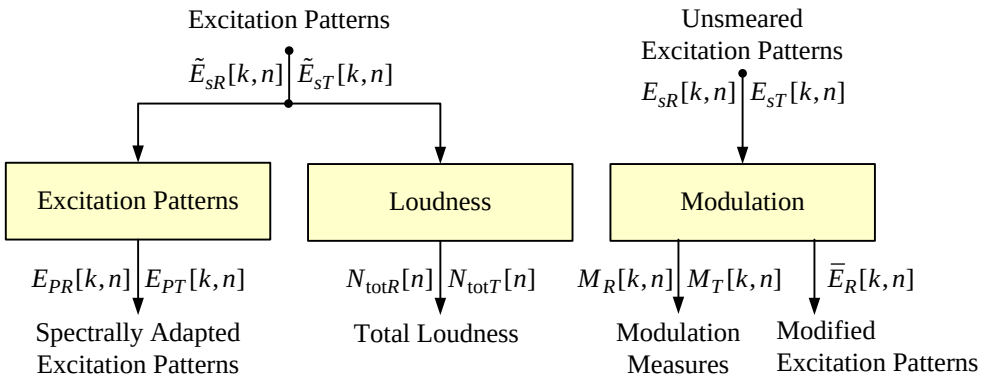

Fig. 10 Inputs and outputs from the pattern processing step.

The model output variables use the outputs of the pattern processing step and sometimes other signals as well. The relationships are shown in Fig. 11.

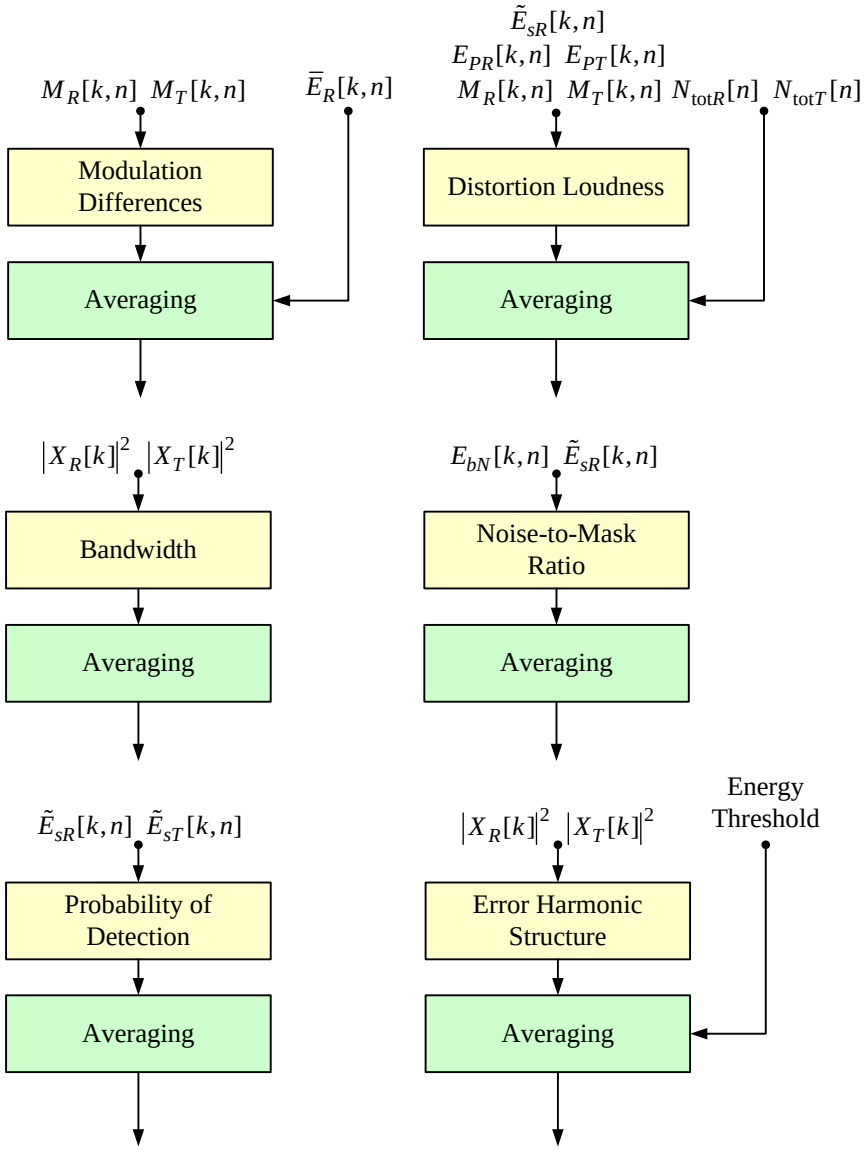

Fig. 11 Inputs to the model output variable calculations.

5.1 Data Boundary

The parameters that will be time averaged to become the model output variables are subject to a data boundary check. The draft revision to BS.1387 states that the data boundary criterion applies to *all* MOV's.

Low level frames at the beginning or end of the input sequence are identified. The test for the beginning or end of data is determined from the *reference* signal and is described as follows.

"The beginning or end of data is defined as the first location, scanning from the start or end of the file, where the sum of the absolute values over five succeeding samples exceeds 200, in one of the corresponding audio channels. Frames which are fully outside of this range are subsequently ignored."

One assumes that the threshold value is given for 16-bit signed integer data. The threshold for a signal with maximum amplitude $A_{\max}$ is

$$A_{\text{thr}} = 200\frac{A_{\max}}{32768}. \tag{72}$$

The implementation calls for a sliding window of $L = 5$ samples. The test is that the average magnitude of the samples in a window be greater than A_{thr} / L (equal to 40 for 16-bit signed integer data).

Interpretation: The data boundary start corresponds to the first sample of the first group of five samples to satisfy the criterion. Similarly for the end of the file, the data boundary end corresponds to the last sample of the last group of five samples which satisfies the criterion. In terms of frames, the start frame is the first frame containing the start-of-data sample and the end frame is the last frame containing the end-of-data sample.

It is an open question how to apply the data boundary condition for the Advanced version of PEAQ. For the Advanced version, in the filter bank processing, a definition of frame is somewhat more elusive than for the FFT processing.

In the subsequent descriptions, the frame index n will start counting from zero at the start frame and the number of frames N will count the frames up to the end frame.

5.2 Modulation Changes

The temporal envelopes for each frequency band are combined into several model output variables. The differences between the modulation patterns for the test and reference signals are first calculated for each frequency band and then averaged over frequency bands.

WinModDiff1$_B$ (Basic version, FFT model)

The instantaneous modulation difference for this MOV is given by

$$M_{\text{diff1}_\text{B}}[k,n] = \frac{|M_T[k,n] - M_R[k,n]|}{1 + M_R[k,n]}. \tag{73}$$

The scaled average over the bands is

$$\tilde{M}_{\mathrm{diff1_B}}[n] = \frac{100}{N_c} \sum_{k=0}^{N_c-1} M_{\mathrm{diff1_B}}[k,n]. \tag{74}$$

The final MOV is given by the sliding window average with $L = 4$ (85 ms),

$$M_{\mathrm{Wdiff1_B}} = \sqrt{\frac{1}{N-L+1} \sum_{n=L-1}^{N-1} \left(\frac{1}{L} \sum_{i=0}^{L-1} \sqrt{\tilde{M}_{\mathrm{diff1_B}}[n-i]} \right)^4}. \tag{75}$$

Delayed averaging is applied.

5.2.1 Delayed Averaging

For delayed averaging, the values calculated during the first 0.5 seconds are omitted. The number of frames skipped is

$$N_{\mathrm{del}} = \lceil \tau_{\mathrm{del}} F_{ss} \rceil, \tag{76}$$

where $\tau_{\mathrm{del}} = 0.5\,\mathrm{s}$. Specifically, the frame index n includes only the frames which occur after the initial delay and the total number of samples N that is used in the average, counts only those values.

Our interpretation is that the computation of the delay in frames should give a delay of *at least* 0.5 seconds.

AvgModDiff1$_B$ (Basic version, FFT model)

The instantaneous modulation difference is given by Eq. (73). The average over bands is given by Eq. (74). The difference for this MOV comes in the form of the averaging used. The final MOV is given by a temporally weighted time average,

$$M_{\mathrm{Adiff1_B}} = \frac{\sum_{n=0}^{N-1} W_{1_\mathrm{B}}[n] \tilde{M}_{\mathrm{diff1_B}}[n]}{\sum_{n=0}^{N-1} W_{1_\mathrm{B}}[n]}, \tag{77}$$

where the temporal weighting is determined by modulation pattern loudness for the reference signal (see Eq. (65)) and an internal noise term (see Eq. (18)),

$$W_{1_B}[n] = \sum_{k=0}^{N_c-1} \frac{\bar{E}_R[k,n]}{\bar{E}_R[k,n] + 100\left(E_{IN}[k]\right)^{0.3}}. \tag{78}$$

Again delayed averaging is used, see Section 5.2.1.

AvgModDiff2$_B$ (Basic version, FFT model)

The instantaneous modulation difference is given by

$$M_{diff2_B}[k,n] = \begin{cases} \dfrac{M_T[k,n] - M_R[k,n]}{0.01 + M_R[k,n]} & M_T[k,n] \geq M_R[k,n], \\ 0.1\dfrac{M_R[k,n] - M_T[k,n]}{0.01 + M_R[k,n]} & M_T[k,n] < M_R[k,n]. \end{cases} \tag{79}$$

The average over bands is

$$\tilde{M}_{diff2_B}[n] = \frac{100}{N_c} \sum_{k=0}^{N_c-1} M_{diff2_B}[k,n]. \tag{80}$$

The final MOV is given by a temporally weighted time average,

$$M_{Adiff2_B} = \frac{\sum_{n=0}^{N-1} W_{2_B}[n]\tilde{M}_{diff2_B}[n]}{\sum_{n=0}^{N-1} W_{2_B}[n]}, \tag{81}$$

where the temporal weighting is determined by modulation pattern loudness for the reference signal (see Eq. (65)) and an internal noise term (see Eq. (18)),

$$W_{2_B}[n] = \sum_{k=0}^{N_c-1} \frac{\bar{E}_R[k,n]}{\bar{E}_R[k,n] + 100\left(E_{IN}[k]\right)^{0.3}}. \tag{82}$$

Delayed averaging is used, see Section 5.2.1

RmsModDiff$_A$ (Advanced version, Filter bank model)

The instantaneous modulation difference is given by

$$M_{diff_A}[k,n] = \frac{|M_T[k,n] - M_R[k,n]|}{1 + M_R[k,n]}. \tag{83}$$

The average over bands is

$$\tilde{M}_{\mathrm{diff_A}}[n] = \frac{100}{N_c} \sum_{k=0}^{N_c-1} M_{\mathrm{diff_A}}[k,n]. \tag{84}$$

The final MOV is given by a temporally weighted time average of squared values,

$$M_{\mathrm{diff_A}} = \sqrt{N_c \frac{\sum_{n=0}^{N-1} \left(W_A[n]\tilde{M}_{\mathrm{diff_A}}[n]\right)^2}{\sum_{n=0}^{N-1} \left(W_A[n]\right)^2}}, \tag{85}$$

where the temporal weighting is determined by modulation pattern loudness for the reference signal (see Eq. (65)) and an internal noise term (see Eq. (18)),

$$W_A[n] = \sum_{k=0}^{N_c-1} \frac{\bar{E}_R[k,n]}{\bar{E}_R[k,n] + \left(E_{\mathrm{IN}}[k]\right)^{0.3}}. \tag{86}$$

Delayed averaging is used, see Section 5.2.1

5.3 Distortion Loudness

The goal is to quantify the partial loudness of distortions. The partial noise loudness is calculated as (see [10] for a derivation)

$$N_L[k,n] = \left(\frac{E_t[k]}{s_T[k,n]E_0}\right)^{0.23} \left[\left(1 + \frac{\max(s_T[k,n]E_{PT}[k,n] - s_R[k,n]E_{PR}[k,n], 0)}{E_t[k] + \beta[k,n]s_R[k,n]E_{PR}[k,n]}\right)^{0.23} - 1\right] \tag{87}$$

where $E_0 = 1$, $E_t[k]$ is the noise threshold (same as $E_{\mathrm{IN}}[k]$ in Eq. (18)) and the linear mapping from the modulation measures to the threshold factors is

$$\begin{aligned} s_R[k,n] &= T_0 M_R[k,n] + S_0, \\ s_T[k,n] &= T_0 M_T[k,n] + S_0. \end{aligned} \tag{88}$$

The parameter α determines the amount of partial loudness giving

$$\beta[k,n] = \exp\left(-\alpha \frac{E_{PT}[k,n] - E_{PR}[k,n]}{E_{PR}[k,n]}\right). \tag{89}$$

The parameter values (α, T_0 and S_0) differ depending on which MOV is being calculated.

RmsNoiseLoud$_B$ (Basic version, FFT model)

For this model output variable (FFT model), the parameters are $\alpha = 1.5$, $T_0 = 0.15$, and $S_0 = 0.5$. Spectral averaging is done as follows,

$$\tilde{N}_{iL}[n] = \frac{24}{N_c} \sum_{k=0}^{N_c-1} N_L[k,n]. \tag{90}$$

If the momentary loudness is less than zero, it is set to zero,

$$\tilde{N}_L[n] = \begin{cases} \tilde{N}_{iL}[n] & \tilde{N}_{iL}[n] \geq 0, \\ 0 & \tilde{N}_{iL}[n] < 0. \end{cases} \tag{91}$$

Temporal averaging of the squared values gives the final MOV,

$$N_{L\mathrm{rmsB}} = \sqrt{\frac{1}{N} \sum_{n=0}^{N-1} \left(\tilde{N}_L[n]\right)^2}. \tag{92}$$

For the computation of this MOV, delayed averaging is used (Section 5.2.1). In addition a loudness threshold is used to find the starting point for samples to be considered.

Our interpretation is that the total delay is the maximum of the 0.5 s delay due to delayed averaging and the delay specified by the loudness threshold.

5.3.1 Loudness Test

For the noise loudness model output variables, momentary values of the noise loudness are ignored at the beginning of the signal. The standard states:

"The values of the momentary noise loudness are not taken into account until 50 ms after the overall loudness for either the left or the right audio channel has once exceeded a value of $N_{Thres} = 0.1$ *sone* for both test and Reference Signal (see § 5.2.4.2)."

The test is then

$$\begin{aligned} &\left(N_{\mathrm{tot}T}[n] \geq L_t\right) \wedge \left(N_{\mathrm{tot}R}[n] \geq L_t\right) && \text{monaural} \\ &\left(\left(N_{\mathrm{tot}T1}[n] \geq L_t\right) \wedge \left(N_{\mathrm{tot}R1}[n] \geq L_t\right)\right) \vee \left(\left(N_{\mathrm{tot}T2}[n] \geq L_t\right) \wedge \left(N_{\mathrm{tot}R2}[n] \geq L_t\right)\right) && \text{binaural} \end{aligned} \tag{93}$$

where $L_t = 0.1\,\mathrm{sone}$ and the numerical subscripts indicate the channel number. The frame offset corresponding to a delay of at least 50 ms is

$$N_{\text{off}} = \lceil \tau_{\text{off}} F_{ss} \rceil, \tag{94}$$

where $\tau_{\text{off}} = 50$ ms and $\lceil \cdot \rceil$ indicates the ceiling function. For the Basic model the delay corresponding to the 50 ms delay is 3 frames and for the Advanced model it is 13 frames.

Our interpretation is that the loudness is given by Eq. (93) and Eq. (94), and the additional delay is calculated to give a delay of *at least* 50 ms.

RmsNoiseLoud$_A$ (Advanced version, Filter bank model)

For this model output variable (filter bank model), the *form* of the equations is the same as for the previous MOV. The parameters are $\alpha = 2.5$, $T_0 = 0.3$, and $S_0 = 1$. The corresponding momentary noise loudness is $\tilde{N}_{iL}[n]$. If the momentary loudness is less than 0.1, it is set to zero,

$$\tilde{N}_L[n] = \begin{cases} \tilde{N}_{iL}[n] & \tilde{N}_{iL}[n] \geq 0.1, \\ 0 & \tilde{N}_{iL}[n] < 0.1. \end{cases} \tag{95}$$

Temporal averaging of the squared values gives the final MOV,

$$N_{\text{LrmsA}} = \sqrt{\frac{1}{N} \sum_{n=0}^{N-1} \left(\tilde{N}_L[n] \right)^2}. \tag{96}$$

For the computation of this MOV, delayed averaging is used (Section 5.2.1). In addition a loudness threshold is used to find the starting point for samples to be considered.

This MOV will be combined with the *RmsMissingComponents* MOV to form the *RmsNoiseLoudAsym* MOV.

RmsMissingComponents (Advanced version, Filter bank model)

This MOV is computed using the same formulation as for the noise loudness MOV's, but with the excitation patterns for the reference and test signals interchanged. This gives the loudness for components that are missing in the test signal. The calculation with these changes is as follows.

$$N_M[k,n] = \left(\frac{E_t[k]}{s_R[k,n] E_0} \right)^{0.23} \left[\left(1 + \frac{\max(s_R[k,n] E_{PR}[k,n] - s_T[k,n] E_{PT}[k,n], 0)}{E_t[k] + \beta[k,n] s_T[k,n] E_{PT}[k,n]} \right)^{0.23} - 1 \right] \tag{97}$$

where $E_0 = 1$, $E_t[k]$ is the noise threshold (same as $E_{\text{IN}}[k]$ in Eq. (18)) and

$$\begin{aligned} s_R[k,n] &= T_0 M_R[k,n] + S_0, \\ s_T[k,n] &= T_0 M_T[k,n] + S_0. \end{aligned} \tag{98}$$

The values $\beta[k,n]$ are given by

$$\beta[k,n] = \exp(-\alpha \frac{E_{PR}[k,n] - E_{PT}[k,n]}{E_{PT}[k,n]}). \tag{99}$$

The parameter values are $\alpha = 1.5$, $T_0 = 0.15$, and $S_0 = 1$.

Spectral averaging is done as follows,

$$\tilde{N}_{iM}[n] = \frac{24}{N_c} \sum_{k=0}^{N_c - 1} N_M[k,n]. \tag{100}$$

If the momentary loudness is less than zero, it is set to zero,

$$\tilde{N}_M[n] = \begin{cases} \tilde{N}_{iM}[n] & \tilde{N}_{iM}[n] \geq 0, \\ 0 & \tilde{N}_{iM}[n] < 0. \end{cases} \tag{101}$$

Temporal averaging of the squared values gives the final MOV,

$$N_{M\,\mathrm{rms}} = \sqrt{\frac{1}{N} \sum_{n=0}^{N-1} \left(\tilde{N}_M[n] \right)^2}. \tag{102}$$

For the computation of this MOV, delayed averaging (Section 5.2.1) and a loudness threshold (Section 5.3.1) are used.

This MOV will be combined with the *RmsNoiseLoud*$_A$ MOV to form the *RmsNoise-LoudAsym* MOV.

RmsNoiseLoudAsym (Advanced version, Filter bank model)

This MOV is the weighted sum of the previous two MOV's,

$$N_{LM} = N_{L\mathrm{rmsA}} + 0.5 N_{M\,\mathrm{rms}}. \tag{103}$$

AvgLinDist$_A$ (Advanced version, Filter bank model)

This MOV measures the loudness of the test signal components that are lost during the spectral adaptation. The description in BS.1387 is as follows.

"It uses the spectrally adapted excitation of the Reference Signal as the reference and the unadapted excitation of the reference as the test signal."

This defines the signals, but leaves all else unstated.

An examination of the alternatives leads us to conclude that both loudness thresholds should be taken from the reference signal.

The formulation is as follows.

$$N_L[k,n]=\left(\frac{E_t[k]}{s_R[k,n]E_0}\right)^{0.23}\left[\left(1+\frac{\max(s_R[k,n]\tilde{E}_{sR}[k,n]-s_R[k,n]E_{PR}[k,n],0)}{E_t[k]+\beta[k,n]s_R[k,n]E_{PR}[k,n]}\right)^{0.23}-1\right] \quad (104)$$

where $E_0=1$, $E_t[k]$ is the noise threshold (same as $E_{\mathrm{IN}}[k]$ in Eq. (18)) and

$$s_R[k,n]=T_0M_R[k,n]+S_0. \quad (105)$$

The values $\beta[k,n]$ is given by

$$\beta[k,n]=\exp(-\alpha\frac{\tilde{E}_{sR}[k,n]-E_{PR}[k,n]}{E_{PR}[k,n]}). \quad (106)$$

The parameter values are $\alpha=1.5$, $T_0=0.15$, and $S_0=1$.

Spectral averaging is done as follows,

$$\tilde{N}_{iL}[n]=\frac{24}{N_c}\sum_{k=0}^{N_c-1}N_L[k,n]. \quad (107)$$

If the momentary loudness is less than zero, it is set to zero,

$$\tilde{N}_L[n]=\begin{cases}\tilde{N}_{iL}[n] & \tilde{N}_{iL}[n]\geq 0,\\ 0 & \tilde{N}_{iL}[n]<0.\end{cases} \quad (108)$$

Temporal averaging gives the final MOV,

$$N_{AL}=\frac{1}{N}\sum_{n=0}^{N-1}\tilde{N}_L[n]. \quad (109)$$

5.4 Bandwidth

The bandwidth is estimated for the reference and test signals. The operations for these calculations are described in terms of operations on the DFT outputs in dB. The operations are as follows.

- Test Signal: Find the largest component above 21.6 kHz. Call this value the threshold level.
- Reference Signal: Searching downward from 21.6 kHz, find the first value which exceeds the threshold level by 10 dB. Record the frequency as the bandwidth of the reference signal.
- Test Signal: Searching downward from the bandwidth of the reference signal, find the first value which exceeds the threshold level by 5 dB. Record the frequency as the bandwidth of the test signal.

If the frequency found for the *reference* signal is not above 8.1 kHz, the bandwidths for that frame are ignored.

The bandwidths are recorded as the corresponding DFT bin number. The operations described above can be carried out on the squared magnitude signals. The Matlab code for these operations appears in Appendix H.4.

BandwidthRef$_B$ (Basic version, FFT model)

Denote the bandwidth (DFT bin number) of the reference signal for frame n as $K_R[n]$. The MOV is the average of the instantaneous bandwidth of the reference signal,

$$W_R = \frac{1}{N}\sum_{n=0}^{N-1} K_R[n], \tag{110}$$

where the sum is over the set of frames for which the bandwidth of the reference signal exceeds 346 (8.1 kHz).

BandwidthTest$_B$ (Basic version, FFT model)

Denote the bandwidth of the test signal for frame n as $K_T[n]$. The MOV is the average of the instantaneous bandwidth of the reference signal,

$$W_T = \frac{1}{N}\sum_{n=0}^{N-1} K_T[n], \tag{111}$$

where the sum is over the set of frames for which the bandwidth of the *test* signal exceeds 346 (8.1 kHz).

5.5 Masking

The masking threshold is calculated from the excitation patterns. The parameter $m_{\mathrm{dB}}[k]$ (expressed in dB) gives the amount by which the masking threshold lies below the time-frequency spread Bark energy. The parameter $m_{\mathrm{dB}}[k]$ is given in BS.1387 as a piecewise linear function on the Bark scale,

$$m_{\mathrm{dB}}[k] = \begin{cases} 3, & k \le \frac{12}{\Delta z}, \\ 0.25k\Delta z, & k > \frac{12}{\Delta z}. \end{cases} \tag{112}$$

The breakpoint in the masking threshold offset occurs at $z_L + 12$ on the Bark scale (z_L is defined in Section 2.6), corresponding to a frequency of 1987 Hz. We can extend this discrete function to a continuous function on the bark scale,

$$m_{\mathrm{dB}}(z) = \begin{cases} 3 & z \le z_L + 12, \\ \frac{1}{4}(z - z_L) & z > z_L + 12. \end{cases} \tag{113}$$

The masking offset is plotted in Fig. 12 against frequency.

The masking threshold (in energy units) is then

$$\begin{aligned} M[k,n] &= \frac{\tilde{E}_{sR}[k,n]}{10^{m_{\mathrm{dB}}[k]/10}} \\ &= g_m[k]\tilde{E}_{sR}[k,n]. \end{aligned} \tag{114}$$

The weighting vector $g_m[k]$ can be precomputed.

The Noise-to-Mask ratio (NMR) is the ratio of the noise to the masking threshold. The noise in this case is the difference between the reference and test signal magnitudes (k is the DFT bin index),

$$X_{wN}^2[k] = \left|X_{wT}[k]\right|^2 - 2\sqrt{\left|X_{wT}[k]\right|^2 \left|X_{wR}[k]\right|^2} + \left|X_{wR}[k]\right|^2. \tag{115}$$

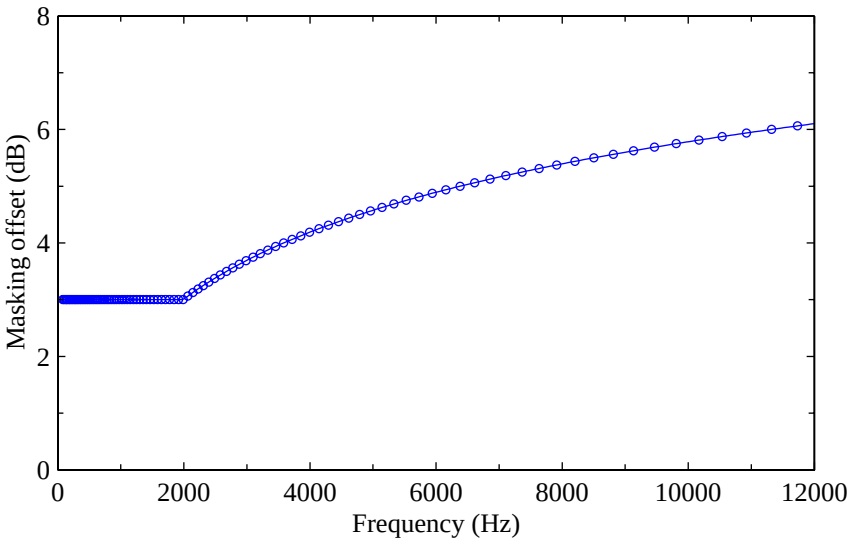

Fig. 12 Masking offset as a function of frequency. The markers indicate the centres of the frequency bands for the Basic version of PEAQ.

This noise signal is then grouped into frequency bins based on the critical band decomposition (see Section 2.6). The *Noise Patterns* in the bands are denoted as $E_{bN}[k,n]$ (k is the band index).

The Noise-to-Mask ratio in band k is

$$\begin{aligned} R_{NM}[k,n] &= \frac{E_{bN}[k,n]}{M[k,n]} \\ &= \frac{E_{bN}[k,n]}{g_m[k]\tilde{E}_{sR}[k,n]}. \end{aligned} \tag{116}$$

Total NMR$_B$ (Basic version, FFT model)

The total NMR MOV is calculated as the average (expressed in dB) of the average NMR in a frame,

$$R_{NM\text{tot}} = 10\log_{10}\left(\frac{1}{N}\sum_{n=0}^{N-1}\frac{1}{N_c}\sum_{k=0}^{N_c-1}R_{NM}[k,n]\right). \tag{117}$$

Relative Disturbed Frames$_B$ (Basic version, FFT model)

The maximum NMR in a frame is

$$R_{N\max}[n] = \max_{0\le k\le N_c-1}(R_{NM}[k,n]). \tag{118}$$

A disturbed frame is one in which the maximum NMR exceeds 1.5 dB. The test can be rearranged to test magnitudes, avoiding the need to convert the levels to decibels. The final MOV measures the fraction of frames for which the NMR exceeds 1.5 dB.

Segmental NMR$_B$ (Advanced version, FFT model)

The segmental NMR MOV is calculated as the average of the NMR in dB for each frame,

$$R_{NM\text{seg}} = \frac{1}{N}\sum_{n=0}^{N-1} 10\log_{10}\left(\frac{1}{N_c}\sum_{k=0}^{N_c-1} R_{NM}[k,n]\right). \tag{119}$$

5.6 Detection Probability

The model output variables in this category measure the probability of detecting differences between the reference and test signal. First we calculate the asymmetric excitation,

$$L[k,n] = \begin{cases} 0.3\tilde{E}_{sR\text{dB}}[k,n] + 0.7\tilde{E}_{sT\text{dB}}[k,n] & \tilde{E}_{sR}[k,n] > \tilde{E}_{sT}[k,n], \\ \tilde{E}_{sT\text{dB}}[k,n] & \tilde{E}_{sR}[k,n] \le \tilde{E}_{sT}[k,n], \end{cases} \tag{120}$$

where

$$\begin{aligned} \tilde{E}_{sR\text{dB}}[k,n] &= 10\log_{10}(\tilde{E}_{sR}[k,n]), \\ \tilde{E}_{sT\text{dB}}[k,n] &= 10\log_{10}(\tilde{E}_{sT}[k,n]). \end{aligned} \tag{121}$$

Next we calculate the effective detection step size (just noticeable difference) using a polynomial approximation (see [11]),

$$s[k,n] = \begin{cases} c_0 + c_1 L[k,n] + c_2 L^2[k,n] + c_3 L^3[k,n] + c_4 L^4[k,n] + d_1\left(\frac{d_2}{L[k,n]}\right)^{\gamma} & L[k,n] > 0, \\ 1\times 10^{30} & L[k,n] \le 0, \end{cases} \tag{122}$$

where the coefficients are

$$\begin{aligned} c_0 &= -0.198719 & c_1 &= 0.0550197 & c_2 &= -0.00102438 \\ c_3 &= 5.05622\times 10^{-6} & c_4 &= 9.01033\times 10^{-11} & d_1 &= 5.95072 \\ d_2 &= 6.39468 & \gamma &= 1.71332 \end{aligned}$$

The probability of detection is

$$p_c[k,n] = 1 - \left(\frac{1}{2}\right)^{\left(\frac{\tilde{E}_{sR\text{dB}}[k,n] - \tilde{E}_{sT\text{dB}}[k,n]}{s[k,n]}\right)^b} \tag{123}$$

where the steepness of the slope is

$$b = \begin{cases} 4 & \hat{E}_{sR}[k,n] > \tilde{E}_{sT}[k,n], \\ 6 & \tilde{E}_{sR}[k,n] \le \tilde{E}_{sT}[k,n]. \end{cases} \tag{124}$$

The differences in slope penalize the adding of audible components more than removing components. The form of Eq. (123) is such that if the difference in excitation is equal to the detection threshold, $p_c[k,n]$ becomes ½.

The number of steps above the threshold is given by

$$q_c[k,n] = \frac{|\,\mathrm{int}(\tilde{E}_{sR\mathrm{dB}}[k,n] - \tilde{E}_{sT\mathrm{dB}}[k,n])\,|}{s[k,n]}. \tag{125}$$

This function measures the distortion in such a way that small deviations across frequency will not add up.

The number of steps above the threshold is measured using an $\mathrm{int}(\bullet)$ function which rounds towards zero for both positive and negative values. A $\mathrm{floor}(\bullet)$ might have been more appropriate, since it is consistent in the direction of rounding.

5.6.1 Total Probability of Detection

The probability of detection and number of steps above threshold are calculated for each channel in a multi-channel signal. For each frequency and time, the total probability of detection and total steps above threshold are calculated from the larger of the channel values,

$$\begin{aligned} P_b[n] &= 1 - \prod_{k=0}^{N_c-1} \left(1 - \max(p_1[k,n], p_2[k,n])\right), \\ Q_b[n] &= \sum_{k=0}^{N_c-1} \max(q_1[k,n], q_2[k,n]), \end{aligned} \tag{126}$$

where the numerical subscripts indicate the channel. For monaural signals, the total values are calculated as follows,

$$\begin{aligned} P_b[n] &= 1 - \prod_{k=0}^{N_c-1} \left(1 - p_c[k,n]\right), \\ Q_b[n] &= \sum_{k=0}^{N_c-1} q_c[k,n]. \end{aligned} \tag{127}$$

MFPD$_B$ (Basic version, FFT model)

The maximum filtered probability of detection is calculated by filtering the binaural or monaural values calculated above,

$$\tilde{P}_b[n] = c_0 \tilde{P}_b[n-1] + (1-c_0) P_b[n], \tag{128}$$

where $c_0 = 0.9$ and the initial condition is zero. This corresponds to a time constant of 0.202 s. The maximum filtered probability of detection is

$$P_M[n] = \max(c_1 P_M[n-1], \tilde{P}_b[n]). \tag{129}$$

The forgetting factor c_1 is either 1 or 0.99 (corresponding to a time constant of 2.123 s). PEAQ has been calibrated for $c_1 = 1$. The other value can be used if one wants to model the effect that subjects tend to discount distortions early in the signal.

The final MOV is the last value calculated,

$$\text{MFPD}_\text{B} = \tilde{P}_M[N-1]. \tag{130}$$

ADB$_B$ (Basic version, FFT model)

The average distorted block MOV measures the total number of steps above the threshold,

$$Q_s = \sum_{n=0}^{N-1} Q_b[n], \tag{131}$$

where the sum is over the set of N values for which $P_b[n] > 0.5$. The distortion of the average distorted block is

$$\text{ADB}_\text{B} = \begin{cases} 0 & N = 0, \\ \log_{10}\left(\dfrac{Q_s}{N}\right) & N > 0 \text{ and } Q_s > 0, \\ -0.5 & N > 0 \text{ and } Q_s = 0. \end{cases} \tag{132}$$

5.7 Harmonic Structure of Error

BS.1387 is very brief in its description of the calculation of the model output variable *EHS*$_B$. In [2], the process is described as being a “cepstrum-like analysis”. Some relevant clarifications appear in the draft revision to BS.1387, but many ambiguities remain.

The DFT's of the windowed reference and test sequence for a particular frame will be denoted as $X_R[k]$ and $X_T[k]$, respectively. The difference in weighted log spectra is

$$\begin{aligned} D[k] &= \log\left(\left|W[k]X_T[k]\right|^2\right) - \log\left(\left|W[k]X_R[k]\right|^2\right) \\ &= \log\left(\frac{\left|X_T[k]\right|^2}{\left|X_R[k]\right|^2}\right) \qquad 0 \le k \le \frac{N_F}{2}. \end{aligned} \tag{133}$$

The outer ear weighting function $W[k]$ is defined in Section 2.5. Note that the exponent in the argument of the logarithm (here we use the square) and the base of the logarithm affect only the scaling of $D[k]$. In the subsequent normalization step, the scaling factor will cancel. Indeed, the order of the subtraction will also be irrelevant when the final output magnitude is calculated. However, scaling of one of the inputs or a misalignment in time between the reference and test signals will affect the result.

The draft revisions to BS.1387 specify that outer and inner ear weighting should be applied.

"The error is defined as the difference in the log spectra of the reference and processed signals, each weighted by the frequency response of the outer and inner ear (Sect 2.1.4, Eq. 7)."

The effect of the outer and inner ear frequency response cancels during the calculation. If the weights $W[k]$ are used, there will be an indeterminacy at zero frequency ($k = 0$), since the calculation will result in $\log(0) - \log(0)$ at zero frequency.

The log values in the calculation will be indeterminate if the response at any frequency is zero for either signal. The energy threshold used to eliminate low energy frames will make it unlikely in practice to have an exact zero valued DFT sample (see Section 5.7.2).

5.7.1 Calculation of the Correlation

Form a vector of length M from $D[k]$,

$$\mathbf{D}_i = \left[D[i], \ldots, D[i+M-1]\right]^T. \tag{134}$$

A normalized autocorrelation is calculated,

$$C(l,i) = \frac{\mathbf{D}_i \bullet \mathbf{D}_{i+l}}{\sqrt{\left|\mathbf{D}_i\right|^2 \left|\mathbf{D}_{i+l}\right|^2}}. \tag{135}$$

The maximum correlation lag (l) is specified as follows in BS.1387.

The maximum lag for obtaining the autocorrelation function is the largest power of two that is smaller than half the FFT frequency component number corresponding to 18 kHz.

We restate the condition here: The value $L_{\max}$ is the index of the frequency bin which is the largest power of two which corresponds to a frequency less than $F_{\max} = 9$ kHz,

$$L_{\max} = 2^{\lceil \log_2(N_F F_{\max}/F_s)\rceil - 1}, \tag{136}$$

where N_F is the FFT size (here 2048), and F_s is the sampling frequency (48 kHz). For these values, $L_{\max}$ is 256. The draft revisions to BS.1387 add the following statement.

"The length of the correlation is the same of the maximum lag (i.e., 256 in the example below)."

This statement is ambiguous. This can refer to the number of correlation terms or the size of the vectors used to calculate the correlation. Here we assume that both the number of correlation terms (N_L) and the size of the vectors (M) is equal to $L_{\max}$.

A strict reading of the text in the standard seems to indicate that the number of correlation lags and the maximum correlation lag are both equal to 256. The correlation lags then go from 1 to 256. Lerch in his implementation [6] used this lag range. A further discussion of the correlation lag range appears in Appendix E.

In terms of the notation developed above, the correlation calculated in BS.1387 is $C[l] = C(l,0)$. The calculation of this value requires the calculation of $|\mathbf{D}_l|^2$. This term can be computed recursively,

$$|\mathbf{D}_l|^2 = \begin{cases} |\mathbf{D}_0|^2, & l = 0, \\ |\mathbf{D}_{l-1}|^2 + D[l+M-1]^2 - D[l-1]^2, & 1 \le l \le L_{\max}. \end{cases} \tag{137}$$

There will be problems in normalizing the correlation values if the two signals are equal ($D[k] = 0$ for all k). If the reference and test signals are equal but non-zero, the correlation can be set to unity.

Direct evaluation of the correlation terms in the numerator of Eq. (135) is quite computationally intensive. An alternative is to use transform techniques to calculate the correlations. Consider the L-point transform of the numerator of $C[l]$. The transform length L is chosen to be at

least $L_{\max}+M$ to allow for the first $L_{\max}+1$ terms of its inverse transform to correspond to the correlation terms for lags 0 to $L_{\max}$. The transform is then

$$\begin{aligned} c[m] &= \sum_{u=0}^{M-1} D[u] W_L^{-um} \sum_{v=0}^{L_{\max}+M-1} D[v] W_L^{mv} \qquad L \geq L_{\max}+M \\ &= d_0^*[m] d[m], \end{aligned} \tag{138}$$

where $W_N = \exp(-j2\pi/N)$. The term $d_0[m]$ is the L-point DFT of the first M values of $D[i]$ and $d[m]$ is the L-point DFT of at least the first $L_{\max}+M$ values of $D[i]$. In both cases, the sequences are padded with zeros to a length L. The first $L_{\max}+1$ terms of the inverse transform of $c[m]$ are the correlation values,

$$C[l] = \frac{1}{\sqrt{|\mathbf{D}_o|^2 |\mathbf{D}_l|^2}} \frac{1}{L} \sum_{m=0}^{L-1} c[m] W_L^{-lm} \qquad 0 \leq l \leq L_{\max}. \tag{139}$$

Choosing $L = 512$, the indirect computation of the correlation using FFT's runs in about 62% of the time for the direct computation (measured with C-language code).

Lag Windowing

The computations involved in windowing the correlation are described in the draft revisions to the standard as:

> "The resulting vector of correlations is windowed with a normalized Hann window and, after removing the DC component by subtracting the average value, a power spectrum is computed with an FFT."

This refers to a "normalized Hann window". Earlier in Section 2.1.3 of the standard, a factor $\sqrt{8/3}$ was applied to normalize the window for (near) unit energy per sample. For the current application, with normalized correlation values, height normalization would seem to be more appropriate. However, Lerch in his implementation [6] does include the $\sqrt{8/3}$ factor.

> It will be assumed that the window includes the $\sqrt{8/3}$ factor.

The correlation lag values represent frequency difference values — lag l corresponding to a frequency difference lF_s/N_F (l times 23 Hz). The correlation is calculated for lags 1 through 256, corresponding to frequencies 23–6000 Hz. For typical audio material, harmonic spacings of,

say, 50–2000 Hz, are of most interest. The corresponding lag range is 2–85. The bottom end of the range occurs for voices; the top end of the range occurs for high-pitched instruments.

The discussion above would suggest that the widowing should emphasize the low-end of the lag range. Earlier in Section 2.1.3 of the standard, the Hann window was one-sided. Applying such a window would emphasize lags in the middle of the range (centered around 3 kHz).

The alignment of the window is not specified. In the absence of other information, it will be assumed that the window is one-sided. Here we assume that the window is of length $L_{\max}$. This gives a window with $L_{\max} - 2$ non-zero values.

The lag window to be used is

$$H[l] = \begin{cases} \sqrt{\frac{8}{3}} \frac{1}{2}[1 - \cos(\frac{2\pi l}{N_L - 1})], & 0 \le l \le L_{\max} - 1, \\ 0, & \text{otherwise.} \end{cases} \tag{140}$$

The standard calls for the removal of the DC component:

"The resulting vector of correlations is windowed with a normalized Hann window and, after removing the DC component by subtracting the average value, a power spectrum is computed with an FFT."

This seems to call for the removal of the mean after windowing. There is, of course, no need to do this since one can instead just ignore or set to zero the DC component in the transform domain. What seems more appropriate (and is noted by Baumgarte and Lerch [5]) is to remove the mean before windowing. With this change, the entire lobe centred at zero in the transform domain is attenuated. Appendix E compares different approaches to mean removal. The results there show there is a clear advantage to removing the mean before windowing.

The average value should be removed *before* windowing.

The computations to get the windowed correlation are as follows.

$$C_w[m] = H[m](C[m+1] - \bar{C}) \qquad 0 \le m \le L_{\max} - 1, \tag{141}$$

where

$$\bar{C} = \frac{1}{L_{\max}} \sum_{l=1}^{L_{\max}} C[l]. \tag{142}$$

Correlation Power Spectrum

The length of the DFT acting on the windowed correlation (referred to as an FFT in the standard) is not explicitly specified, but one assumes that the length will be $L_{\max}$, since N_L has been chosen to be a power of two.

It is assumed that the FFT length is N_L. The scaling of the FFT is not specified. Earlier in Section 2.1.3, scaling by the reciprocal of the length of the transform was used. It will be assumed that the FFT should be scaled by $1/N_L$.

The power spectral sequence is then

$$S[k] = \left| \frac{1}{N_L} \sum_{l=0}^{N_L-1} C_w[l] e^{j2\pi kl/N_L} \right|^2. \tag{143}$$

Fig. 13 plots a correlation power spectrum calculated as described. This input signal in this case is a 1 kHz triangular wave. The test signal (monaural) was coded using a low rate MP3 coder. The time resolution for the correlation power spectrum is $N_F/(N_L F_s)$, which works out to be $1/6$ ms. One can see the peak at 1 ms corresponding to the 1 kHz fundamental. The peak value, when multiplied by the 1000 factor used to scale the model output variable is less than the maximum given for the EHS_B MOV (see Section 6.4). The scaling by $1/N_L$ for the DFT is necessary to avoid exceeding that maximum.

Identification of the Harmonic Peak

The draft revision to BS.1387 states:

"The maximum peak in the spectrum after the first valley identifies the dominant frequency."

The correlation power spectrum is symmetric, so the search for a maximum should be stopped at the mid-point. There is also a question as to whether the middle point (point $N_L/2$) should be included in the search range.

The search for the first valley starts with index 1 and stops when $S[k] > S[k-1]$. The search for a maximum extends to index $N_L/2$ (inclusive).

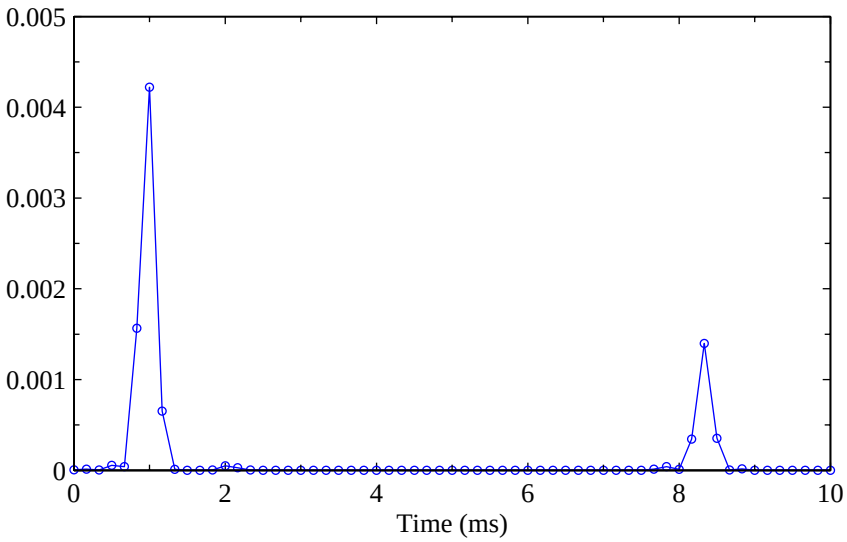

Fig. 13 Correlation power spectrum used to calculate the EHS_B model output variable. The reference signal is a 1 kHz triangular wave. The test signal is a low rate coded version of the reference signal.

For the example in the figure above, the value of the peak at 1 ms gives the value sought.

EHS_B (Basic version and Advanced version, FFT model)

Let the maximum peak in the correlation power spectrum sequence be $E_{H\max}[n]$. The average value of this peak value times 1000 gives the EHS_B model output variable,

$$\mathrm{E}_{H_\mathrm{B}} = \frac{1000}{N}\sum_{n=0}^{N-1} E_{H\max}(n). \qquad (144)$$

5.7.2 Energy Threshold

In the computation of the EHS_B MOV, low energy frames are not included. In BS.1387 the statement is as follows.

> "When the energy of the most recent half of a frame of 2048 samples is less than 8000, in either the mono channel, or both, the left and right channels of the reference and test data, the frame is ignored."

The threshold value is given for 16-bit signed integer data. The threshold for a signal with maximum amplitude $A_{\max}$ is

$$A_{\text{thr}}^2 = 8000\left(\frac{A_{\max}}{32768}\right)^2. \tag{145}$$

The energy computation for a particular frame is of the form (n is the sample index),

$$A^2 = \sum_{n=N_F/2}^{N_F-1} x^2[n]. \tag{146}$$

The comparison is such that the average energy of the half frame should be less than $A_{\text{thr}}^2/(N_F/2)$ (equal to 7.8 for 16-bit signed data, corresponding to a RMS value of 2.8). Our interpretation of the rather obfuscated statement in the standard is as follows.

For a monaural signal, the frame is ignored if the energy of the most recent half frame is less than 8000 for both the reference signal and the test signal. For a binaural signal, the frame is ignored if the energy of the most recent half frame is less than 8000 for both channels of the reference signal and for both channels of the test signal.

For the computation of EHS_B, a frame is ignored if,

$$\begin{aligned} &\left(A_T^2 < A_{\text{thr}}^2\right) \wedge \left(A_R^2 < A_{\text{thr}}^2\right) && \text{monaural} \\ &\left(A_{T1}^2 < A_{\text{thr}}^2\right) \wedge \left(A_{R1}^2 < A_{\text{thr}}^2\right) \wedge \left(A_{T2}^2 < A_{\text{thr}}^2\right) \wedge \left(A_{R2}^2 < A_{\text{thr}}^2\right) && \text{binaural} \end{aligned} \tag{147}$$

The energy threshold avoids problems in the calculation of the logarithms for zero or near-zero frames. For monaural signals, the instantaneous value need only be evaluated when it is guaranteed that at least one of the signals has non-zero samples. For binaural signals, only one of the four signals (2 signals and 2 channels) is guaranteed to be non-zero.

6 Calculation of the Objective Difference Grade

The previous sections have described the calculation of the model output variables. These MOV's will be combined using a neural network to give an objective difference grade which measures the degradation of the test signal with respect to the reference. The neural network has been trained to give good matches to the subjective impairment scale shown in the table below. The subjective impairment scale measures the difference between the grade given to the signal under test less the grade given to the reference signal [12][13].

Table 6 Model Output Variables – PEAQ Advanced

Difference Grade	Description of Impairments
0	Imperceptible
-1	Perceptible but not annoying
-2	Slightly annoying
-3	Annoying
-4	Very annoying

6.1 Model Output Variables – Basic Version

The model output variables for the Basic version are shown in the table below.

Table 7 Model Output Variables – PEAQ Basic

Index	Model Output Variable	Description
0	$BandwidthRef_B$	Bandwidth of the reference signal
1	$BandwidthTest_B$	Bandwidth of the test signal
2	$Total\ NMR_B$	Noise-to-mask ratio
3	$WinModDiff1_B$	Windowed modulation difference
4	ADB_B	Average block distortion
5	EHS_B	Harmonic structure of the error
6	$AvgModDiff1_B$	Average modulation difference
7	$AvgModDiff2_B$	Average modulation difference
8	$RmsNoiseLoud_B$	Distortion loudness
9	$MFPD_B$	Maximum filtered probability of detection
10	$RelDistFrames_B$	Relatively disturbed frames

6.2 Model Output Variables – Advanced Version

For the Advanced version, the model output variables are shown in the table below.

Table 8 Model Output Variables – PEAQ Advanced

Index	Model Output Variable	Description
0	$RmsModDiff_A$	Modulation changes
1	$RmsNoiseLoudAsym_A$	Distortion loudness
2	$Segmental\ NMR_B$	Noise-to-mask ratio
3	EHS_B	Harmonic structure of the error
4	$AvgLinDist_A$	Linear distortions

6.3 Binaural Signals

In the case of binaural signals, MOV's are calculated for each channel and then are averaged. For most MOV's, the calculations for the channels are independent of each other. For the model output variables ADB_B and $MFPD_B$, the calculation for binaural signals finds the maximum probability of detection across channels and frequencies, see Section 5.6.

6.4 Scaling the Model Output Variables – Basic Version

The first step in the processing is to shift and scale the model output variables.

$$M_v'[i] = \frac{M_v[i] - a_{\min}[i]}{a_{\max}[i] - a_{\min}[i]}. \tag{148}$$

The table below shows the scaling and shifting parameters for each of the MOV's. These parameters can be used to give estimates of the normal ranges of the model output variables.

Table 9 Model Output Variables: Scaling and Shifting Parameters – PEAQ Basic version

Index	Model Output Variable	Min. Value	Max. Value
0	$BandwidthRef_B$	393.916656	921.0
1	$BandwidthTest_B$	361.965332	881.131226
2	$Total\ NMR_B$	-24.045116	16.212030
3	$WinModDiff1_B$	1.110661	107.137772
4	ADB_B	-0.206623	2.886017
5	EHS_B	0.074318	13.933351
6	$AvgModDiff1_B$	1.113683	63.257874
7	$AvgModDiff2_B$	0.950345	1145.018555
8	$RmsNoiseLoud_B$	0.029985	14.819740
9	$MFPD_B$	0.000101	1.0
10	$RelDistFrames_B$	0.0	1.0

These parameters are described as scaling factors. Baumgarte and Lerch suggest the following.

> "The model output values (MOV) should be truncated to the range between a_min and a_max used in the neural network. Otherwise, the ODG can substantially increase while the subjective audio quality decreases." — Baumgarte and Lerch [5].

The scaling operation maps MOV's between the minimum and maximum values into the interval 0 to 1. The weights in the neural network rescale this interval. With these complications, it is not obvious that clipping of the MOV's is appropriate. Whether to apply clipping is an open question.

6.5 Scaling the Model Output Variables – Advanced Version

The table below shows the scaling and shifting parameters for the Advanced version (taken from the draft revision of BS.1387 [4]).

Table 10 Model Output Variables: Scaling and Shifting Parameters – PEAQ Advanced version

Index	Model Output Variable	Min. Value	Max. Value
0	*RmsModDiffA*	13.298751	2166.500
1	*RmsNoiseLoudAsymA*	0.041073	13.24326
2	*Segmental NMRB*	-25.018791	13.46708
3	EHS_B	0.061560	10.226771
4	*AvgLinDistA*	0.024523	14.224874

6.6 Neural Network – Basic Version

For the Basic version, the scaled and shifted MOV's are input to a neural network with 11 input nodes, 1 hidden layer with 3 nodes and a single output, the *distortion index*,

$$D_I = w_{yb} + \sum_{j=0}^{J-1}\left(w_y[j]\,\mathrm{sig}\left(w_{xb}[j] + \sum_{i=0}^{I-1} w_x[i,j]M_v'[i]\right)\right), \tag{149}$$

where I is the number of MOV's (11 for the Basic version) and J is the number of nodes in the hidden layer. The terms $w_{xb}[j]$ and w_{yb} are bias terms. The weights are shown in the tables below.

Table 11 Neural network input weights – PEAQ Basic version

Index i	**Weight** $w_x[i,0]$	**Weight** $w_x[i,1]$	**Weight** $w_x[i,2]$
0	-0.502657	0.436333	1.219602
1	4.307481	3.246017	1.123743
2	4.984241	-2.211189	-0.192096
3	0.051056	-1.762424	4.331315
4	2.321580	1.789971	-0.754560
5	-5.303901	-3.452257	-10.814982
6	2.730991	-6.111805	1.519223
7	0.624950	-1.331523	-5.955151
8	3.102889	0.871260	-5.922878
9	-1.051468	-0.939882	-0.142913
10	-1.804679	-0.503610	-0.620456

	Bias $w_{xb}[0]$	**Bias** $w_{xb}[1]$	**Bias** $w_x[2]$
bias	-2.518254	0.654841	-2.207228

Table 12 Neural network output weights – PEAQ Basic version

Index j	**Weight** $w_y[j]$
0	-3.817048
1	4.107138
2	4.629582

	Bias w_{yb}
bias	-0.307594

The non-linearity used is an asymmetric sigmoid,

$$\begin{aligned} \mathrm{sig}(x) &= \frac{1}{1+e^{-x}} \\ &= \tfrac{1}{2} + \tfrac{1}{2}\tanh(\tfrac{x}{2}). \end{aligned} \tag{150}$$

This function is plotted below.

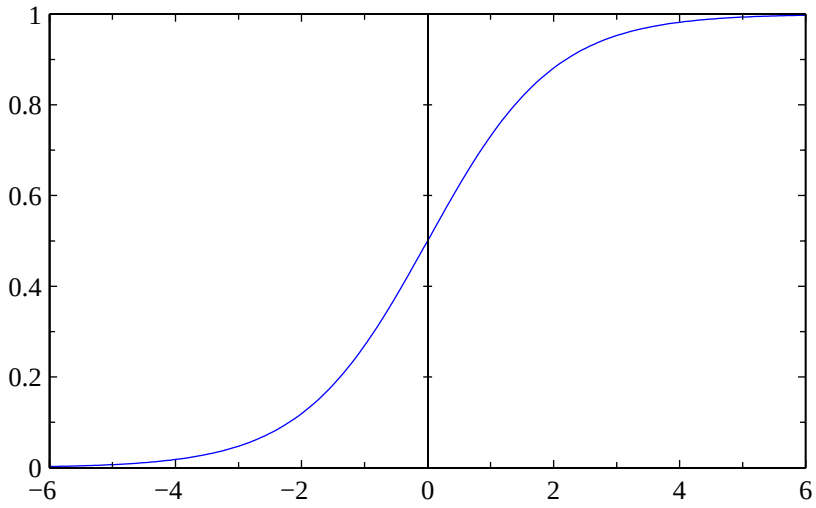

Fig. 14 Sigmoid function used in the neural network.

The final output is the objective grade difference, which is found from

$$\mathrm{ODG} = b_{\min} + (b_{\max} - b_{\min})\,\mathrm{sig}(D_I), \tag{151}$$

where $b_{\min} = -3.98$ and $b_{\max} = 0.22$.

6.7 Neural Network – Advanced Version

For the Advanced version, the scaled and shifted MOV's are input to a neural network with 5 input nodes, 1 hidden layer with 5 nodes and a single output, the *distortion index*,

$$D_I = w_{yb} + \sum_{j=0}^{J-1}\left(w_y[j]\,\mathrm{sig}\left(w_{xb}[j] + \sum_{i=0}^{I-1} w_x[i,j]M_v'[i]\right)\right), \tag{152}$$

where I is the number of MOV's (5 for the Advanced version) and J is the number of nodes in the hidden layer. The terms $w_{xb}[j]$ and w_{yb} are bias terms. The weights are shown in the tables below version (taken from the draft revision to the standard [4]).

Table 13 Neural network input weights – PEAQ Advanced

Index i	**Weight** $w_x[i,0]$	**Weight** $w_x[i,1]$	**Weight** $w_x[i,2]$	**Weight** $w_x[i,3]$	**Weight** $w_x[i,4]$
0	21.211773	-39.913052	-1.382553	-14.545348	-0.320899
1	-8.981803	19.956049	0.935389	-1.686586	-3.238586
2	1.633830	-2.877505	-7.442935	5.606502	-1.783120
3	6.103821	19.587435	-0.240284	1.088213	-0.511314
4	11.556344	3.892028	9.720441	-3.287205	-11.031250

	Bias $w_{xb}[0]$	**Bias** $w_{xb}[1]$	**Bias** $w_{xb}[2]$	**Bias** $w_{xb}[3]$	**Bias** $w_x[4]$
bias	1.330890	2.686103	2.096598	-1.327851	3.087055

Table 14 Neural network output weights – PEAQ Advanced version

Index j	**Weight** $w_y[j]$
0	-4.696996
1	-3.289959
2	7.004782
3	6.651897
4	4.009144

	Bias w_{yb}
bias	-1.360308

The final output is the objective grade difference, which is found from

$$\mathrm{ODG} = b_{\min} + (b_{\max} - b_{\min})\,\mathrm{sig}(D_I), \tag{153}$$

where $b_{\min} = -3.98$ and $b_{\max} = 0.22$.

6.8 Start and End Samples

There is uncertainty as how to handle the start and end of signals stored in files. The various choices can affect the final measure. The standard does not directly address these issues which can affect conformance. Frames at the beginning and end of the files which are very low level are automatically excluded using the data boundary criterion (Section 5.1).

Let the reference and test files start at sample zero and have lengths (in samples) of N_R and N_T, respectively. Further consider that samples before sample zero and after the end of the data are zero. Let the first sample in the first frame be aligned with sample n_{off}. Consider the following choices for n_{off}.

- The first frame starts at sample zero, $n_{\text{off}} = 0$. This however means that samples near the beginning of the signal appear with small weights due to the use of a Hann window in the DFT analysis stage.
- Offset the frame by half of a frame, i.e. $n_{\text{off}} = -N_F / 2$. The first frame then contains half of frame of zeros followed by half a frame of data. That same data will reappear in the second frame since frames are overlapped by half of their length.

For the end of data, similar considerations occur, but complicated by the fact that the signals need not be the same length. One choice is to have the length of signal to be processed to be the maximum of the lengths of reference and test signals.

$$N_s = \max(N_{sR}, N_{sT}). \tag{154}$$

We have to consider when to stop processing the signals.

- The last frame is the one that contains more than half a frame of data. The reasoning is that if one more frame were to be added, it would contain only samples that have already appeared in the previous frame.
- The last frame is the one that contains at least one sample of the signal.

The number of frames to be processed for these options is as follows.

$$N_p = \left\lceil \frac{N_s - n_{\text{off}} - n + 1}{N_F / 2} \right\rceil \qquad \text{At least } n \text{ samples in last frame.} \tag{155}$$

Option 1 for the start of data is consistent with option 1 for the end of data and option 2 for the start of data is consistent with option 2 for the end of data. Baumgarte and Lerch [5] suggest adding zeros to the beginning and end of the file. It is an open question as to the strategy used in the reference implementation.

6.9 Summary

This report has highlighted a large number of shortcomings in BS.1387, especially in the area of being underspecified. This can only be corrected with additional information from the authors of the reference implementation referred to in the standard. Some specific comments are as follows.

- Correct the pseudo-code for spreading in the filter bank model.
- Be more explicit about initialization of filtering operations, particularly for the pattern correction factors. Remove the offending phrase “if not given otherwise”.
- Explicitly address the issue of whether the MOV’s should be clipped to the given minimum and maximum values.
- Expand the description of the calculations of the error harmonic structure, addressing the ambiguities as to use of scaling, windowing and transformation. Provide pseudo-code to make explicit the algorithm to search for “the maximum peak in the spectrum after the first valley”. Clarify the energy threshold criterion for multiple channel inputs.
- Justify magic constants which may not give the intended results, e.g. the scaling factor in the loudness calculation and the scaling factor in the backward masking filter.
- Make explicit how to handle delayed averaging, e.g. should the delay be about 0.5 s or *at least* 0.5 s.
- Be more specific in the calculation of the average linear distortion MOV.
- Clarify the description of the data boundary condition. This should include information on how to handle the start and end of files. Should data before the starting data boundary be processed to establish the filter memories or not? Should the 0.5 s delays in processing certain MOV’s overlap the delay due to the starting data boundary?
- Provide test material for implementers. This need not be the large databases used for conformance testing, but only one or two inputs for test purposes. Results for individual model output variables, not only the final measure, should be made available.

7 References

1. ITU-R Recommendation BS.1387, *Method for Objective Measurements of Perceived Audio Quality*, Dec. 1998.
2. T. Thiede, W. C. Treurniet, R. Bitto, C. Schmidmer, T. Sporer, J. G. Beerends, C. Colomes, M, Keyhl, G. Stoll, K. Brandenburg, and B. Feiten, "PEAQ – The ITU Standard for Objective Measurement of Perceived Audio Quality", *J. Audio Eng. Soc.*, vol. 48, pp. 3–29, Jan.–Feb. 2000.
3. Opticom, *List of Corrections of the ITU-R Recommendation BS.1387*, web document: http://www.peaq.org/downloads/Corrections%20to%20BS1387.pdf, March 2001.
4. ITU-R Study Group 6, *Draft Revision to BS.1387, Method for Objective Measurements of Perceived Audio Quality*, Document 6/BL/30-E, July 2001.
5. F. Baumgarte and A. Lerch, *Implementation of Recommendation ITU-R BS.1387*, Delayed Contribution, Document 6Q/18-E, Feb. 2001.
6. A. Lerch, *EAQUAL – Evaluation of Audio QUALity*, Software repository: http://sourceforge.net/projects/eaqual, Jan. 2002.
7. E. Terhardt, "Calculating Virtual Pitch", *Hearing Research*, vol. 1, pp. 155–182, 1979.
8. J. D. Johnston, "Transform coding of audio signals using perceptual noise criteria", *IEEE. J. Selected Areas Commun.*, vol. 6, pp. 314–323, Feb. 1988.
9. G. A. Soulodre, *Adaptive Methods for Removing Camera Noise from Film Soundtracks*, Ph.D. Thesis, Dept. Electrical Eng., McGill University, Nov. 1998.
10. T. Thiede, *Perceptual Audio Quality Assessment Using a Non-Linear Filter Bank*, Ph.D. Thesis, Fachbereich Electrotechnik, Technical University of Berlin, 1999 (available on-line at http://www-ft.ee.tu-berlin.de/Publikationen/promotionen.htm).
11. E. Zwicker and H. Fastl, *Psychoacoustics: Facts and Models*, Second Ed., Springer Verlag, 1998.
12. ITU-R Recommendation BS.1116-1, *Methods for the Subjective Assessment of Small Impairments in Audio Systems Including Multichannel Sound Systems*, Oct, 1997.
13. ITU-R Recommendation BS.562-3, *Subjective Assessment of Sound Quality*, June 1990.

Appendix A Calibration of the Loudness Scaling Factor

Consider a sine wave test signal of amplitude A_c, frequency f_c and phase θ (relative to the start of the frame),

$$x(t) = A_c \cos(2\pi f_c t + \theta). \tag{156}$$

A Hann window of length W will be applied to this signal. A unit-height continuous-time Hann (raised-cosine) window of length W centered at $t = 0$ is

$$h_c(t,W) = \begin{cases} \frac{1}{2}\left[1 + \cos(\frac{2\pi t}{W})\right], & |t| \le \frac{W}{2}, \\ 0, & \text{elsewhere.} \end{cases} \tag{157}$$

The windowed data is

$$x_w(t) = h_c(t - \tfrac{W}{2}, W)\, x(t). \tag{158}$$

The Fourier Transform of the windowed sine data will be the transform of the sine wave (a pair of delta functions) convolved with the transform of the window.

The Fourier transform of the continuous-time Hann window (centered at zero) is

$$H_c(f) = \frac{W}{2} \frac{\sin(\pi fW)}{\pi fW} \frac{1}{1-(fW)^2}. \tag{159}$$

The Fourier transform of the windowed sine wave is

$$X_{wc}(f) = \frac{A_c}{2}\left[H_c(f - f_c) e^{-j\pi(f-f_c)W} e^{j\theta} + H_c(f + f_c) e^{-j\pi(f+f_c)W} e^{-j\theta} \right]. \tag{160}$$

The figure below plots the magnitude of $X_{wc}(f)$. The plot is for $f_c W = 10$. For the test frequency (1019.5 Hz) and window length (43 ms) used in BS.1387, $f_c W = 43.5$. This means that the two main lobes are even further apart than those shown in the figure. For the response centred at $f = -f_c$, the tail at $f = +f_c$ is attenuated by at least a factor $|\pi 2 f_c W(1-(2 f_c W)^2)|$ from the peak value. For $f_c W = 43.5$, the attenuation is at least 2×10^6 or 126 dB. For practical purposes, the effect of overlap is insignificant.

Discrete-Time Fourier Transform

The Discrete-Time Fourier Transform (DTFT) of the sampled signal is

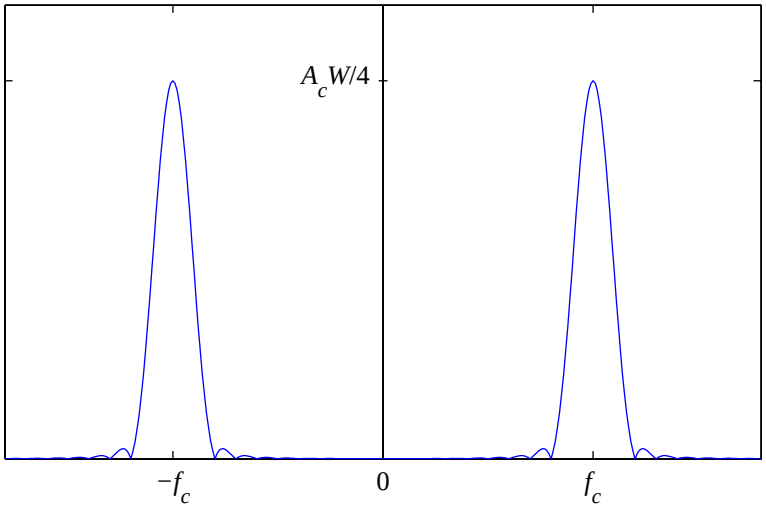

Fig. 15 Magnitude of $X_{wc}(f)$ for $f_c W = 10$.

$$X(\omega) = F_s \sum_{l=-\infty}^{\infty} X_{wc}((\frac{\omega}{2\pi} - l)F_s). \tag{161}$$

The response contains the sum of the window frequency responses centred at $lF_s \pm f_c$.

The Discrete Fourier Transform (DFT) is the DTFT of Eq. (161) evaluated at $\omega = 2\pi k / N_F$,

$$X[k] = F_s \sum_{l=-\infty}^{\infty} X_{wc}(\frac{k}{N_F} - l)F_s). \tag{162}$$

The figure below shows the magnitude of the DFT near $f = f_c$. The figure is for the case of a sine of frequency 1019.5 Hz and a window length of 2048 samples (48 kHz sampling rate). Note that the peak of the DTFT falls between the DFT bins. The maximum absolute value is

$$X_{\max}(A_c, f_c) = \gamma(f_c)\frac{A_c}{4} W F_s, \tag{163}$$

where the term $\gamma(f_c)$ varies from 0.84 to 1 depending on where the frequency of the sine wave falls relative to the DFT bins.[10]

Calibration Level

The discussion above is based on a unity height window and a standard definition of the DFT. In BS.1387, the window is of length $W = (N_F - 1)/F_s$. The additional scale factor G_L is

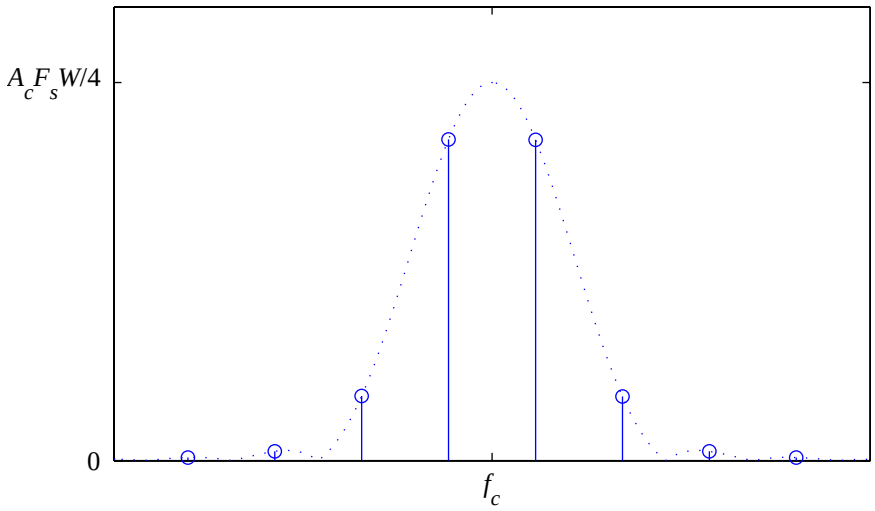

Fig. 16 DFT response to a sinusoid.

used to calibrate the absolute sound pressure level. Including G_L, the maximum absolute value of the DFT for a sine wave is

$$X(A_c, f_c) = \gamma(f_c) G_L \frac{A_c}{4}(N_F - 1). \tag{164}$$

Note that for the window length chosen, the zero crossings of the frequency response occur at regular intervals of $2\pi/(N_F - 1)$, slightly out of synchrony with the frequency spacing of the DFT bins (see Fig. 16).

In the BS.1387 standard, it is assumed that a full scale sine wave with a sound pressure level of L_p results in a peak magnitude of the scaled DFT values of $10^{L_p/20}$. Setting the sine wave amplitude to the maximum value, $X(A_{\max}, f_c) = 10^{L_p/20}$. Then using the standard Hann window formulation and the standard DFT scaling,

$$G_L = \frac{10^{L_p/20}}{\gamma(f_c)\frac{A_{\max}}{4}(N_F - 1)}. \tag{165}$$

[10] Let the distance from f_c to the nearest DFT bin be denoted as Δf. Then $\gamma(f_c)$ is given by $\sin(\pi \Delta f W)/[\pi \Delta f W(1-(\Delta f W)^2)]$.

For L_p equal to 92 dB SPL, $A_{\max}$ equal to 32 768 (for instance for 16-bit data), N_F equal to 2048, and f_c equal to 1019.5 (giving $\gamma(f_c) = 0.8497$), the scale factor G_L is equal to 2.794×10^{-3}.

Experiments verify the analysis. Using a sine wave frequency of 1019.5, the peak value of the magnitude of the DFT is essentially independent of the phase (and hence unchanged from frame-to-frame) and agrees with the predicted value to at least 6 decimal places. This constancy is further confirmation that the overlap between the lobes in the frequency response can be neglected.

BS.1387 suggests measuring the average maximum absolute DFT value over 10 frames. In fact, the combination of test frequency and frame length does not exercise many different phases. The phase advance for the sine between adjacent frames (1024 samples apart) is 1.499π. Neglecting sign reversals, the sine values nearly repeat every two frames.

Appendix B Spreading Function

The following description is given in terms of a continuous Bark spectrum. The energy calculated on the Bark scale is spread using a spreading function. The spreading function is level and frequency dependent. The spreading function for an energy component E at z_c will be denoted as $S(z, z_c, E)$. It will be assumed that the spreading functions add in the 0.4 power domain. After integrating the spreading function applied to each frequency, the result is brought back to the energy domain,

$$E_s(z) = \frac{1}{B_s(z)} \left(\int_{z_L}^{z_U} \left[E(z_c) S(z, z_c, E(z_c)) \right]^{0.4} dz_c \right)^{\frac{1}{0.4}}, \tag{166}$$

where $B(z)$ is a normalizing factor calculated by setting $E_s(z)$ to unity for $E(z_c)$ equal to unity,

$$B_s(z) = \left(\int_{z_L}^{z_U} \left[S(z, z_c, 1 \right]^{0.4} dz_c \right)^{\frac{1}{0.4}}. \tag{167}$$

The normalizing factor ensures that a signal with a constant energy distribution will result in a constant unit energy distribution after spreading [8].

The spreading function is defined in terms of the spreading function expressed in dB,

$$S(z, z_c, E) = \frac{1}{A(z_c, E)} 10^{S_{\mathrm{dB}}(z, z_c, E)/10}. \tag{168}$$

The factor $A(z_c, E)$ is a normalizing factor chosen to give a unit area to the spreading function in the power domain,

$$\int_{z_L}^{z_U} S(z, z_c, E)\, dz = 1. \tag{169}$$

On a dB scale, the spreading function is triangular, with the peak of the triangle at $z = z_c$. The slope for Bark values less than z_c is fixed. The slope for z larger than z_c depends on z_c and the level of the signal [7],

$$S_{\mathrm{dB}}(z, z_c, E) = \begin{cases} 27(z - z_c), & z \le z_c, \\ \left[-24 - \frac{230}{B^{-1}(z_c)} + 0.2 \cdot 10 \log_{10}(E) \right](z - z_c), & z_c \le z, \end{cases} \tag{170}$$

where the term $B^{-1}(z_c)$ is the frequency corresponding to Bark value z_c.

Appendix C Butterworth DC Rejection Filter

The digital highpass Butterworth filter will be derived from a prototype analogue lowpass filter. An analogue Butterworth filter of order N with cutoff Ω_c will have its poles uniformly spaced on the left half of a circle of radius Ω_c centered at $s = 0$ in the s-plane,

$$s_k = \Omega_c \exp(j\pi(\tfrac{2k+1}{2N} + \tfrac{1}{2}) \qquad 0 \le k \le N-1. \tag{171}$$

The zeros lie at infinity.

The digital filter will be derived using a bilinear transformation from the s-plane to the z-plane. This mapping will convert a lowpass analogue filter to a highpass digital filter,

$$z = \frac{as+1}{as-1}. \tag{172}$$

This transformation maps $s = 0$ to $z = -1$ and $s = j\infty$ to $z = 1$. It also maps the $j\Omega$ axis in the s-plane to the unit circle in the z-plane. The frequency warping function is

$$a\Omega = \frac{1}{\tan(\omega/2)} \tag{173}$$

The parameter a is chosen such that the analog cutoff Ω_c maps to the digital cutoff f_c,

$$a = \frac{1}{\Omega_c \tan(\pi f_c / F_s)}, \tag{174}$$

where F_s is the sampling frequency.

The bilinear transform maps the poles on the circle in the s-plane (radius Ω_c, centered at $s = 0$) to poles on a circle in the z-plane (radius $2\Omega_c a/(\Omega_c^2 a^2 - 1)$, centered at $(\Omega_c^2 a^2 + 1)/(\Omega_c^2 a^2 - 1)$). The zeros of the digital filter will appear at $z = 1$.

For the highpass Butterworth filter in BS.1387, the order is $N = 4$, the cutoff is $f_c = 20\,\text{Hz}$ and the sampling frequency is $F_s = 48\,\text{kHz}$. The poles of the prototype analogue filter can be calculated for, say, $\Omega_c = 1$. The mapping parameter a is then computed from Eq. (174). The poles are transformed to the z-plane using Eq. (172). Given these poles and the zeros (at $z = 1$), the filter response is

$$H(z) = \frac{(z-1)^2}{z^2 + a_{01}z + a_{02}} \frac{(z-1)^2}{z^2 + a_{11}z + a_{12}}, \tag{175}$$

where the coefficients (to the same precision as given in BS.1387) are

$$\begin{aligned} a_{01} &= -1.99517, & a_{02} &= 0.995174, \\ a_{11} &= -1.99799, & a_{12} &= 0.997998. \end{aligned} \tag{176}$$

Appendix D Filter Bank

The filter bank uses a set of increasingly wider bandpass filters at 40 centre frequencies ranging from $f_L = 50$ Hz to $f_U = 18000.02$ Hz. For each centre frequency $f_c[k]$, there is a pair of linear phase FIR filters, one with zero phase and the other filter with phase 90°. The filters are

$$\begin{aligned} h_I[n,k] &= h_p[k,n]\cos\left(2\pi \frac{f_c[k]}{F_s}(n-\frac{N_k}{2})\right), \quad 0 \le n \le N_k - 1, \\ h_Q[n,k] &= h_p[k,n]\sin\left(2\pi \frac{f_c[k]}{F_s}(n-\frac{N_k}{2})\right), \quad 0 \le n \le N_k - 1, \end{aligned} \tag{177}$$

where the lowpass prototype for filter k is

$$h_p[n,k] = \frac{4}{N_k}\sin^2(\pi\frac{n}{N_k}), \qquad 0 \le n \le N_k - 1. \tag{178}$$

The prototype filters are scaled Hann windows. The frequency response for the prototype is

$$H_p(\omega,k) = \tfrac{1}{2}H_r(\omega,N_k) - \tfrac{1}{4}H_r(\omega-\tfrac{2\pi}{N_k},N_k) - \tfrac{1}{4}H_r(\omega+\tfrac{2\pi}{N_k},N_k), \tag{179}$$

where $H_r(\omega,N)$ is the frequency response of a rectangular window,

$$H_r(\omega,N) = e^{j\omega\frac{N-1}{2}}\frac{\sin(\frac{\omega N}{2})}{\sin(\frac{\omega}{2})}. \tag{180}$$

This response of the lowpass prototype is the superposition of a main response at $\omega = 0$ and two half height responses at $\omega = \pm 2\pi / N_k$. This prototype has regular frequency domain zero crossings every $2\pi / N_k$ in the tail of the response. The main lobe width can be measured in a number of ways. The distance between zero crossings around the main lobe is $8\pi / N_k$, the half value (6 dB down) bandwidth is $4\pi / N_k$, and the half power bandwidth is $2.88\pi / N_k$.

The frequency responses of the modulated filters are given by

$$\begin{aligned} H_I(\omega,k) &= \tfrac{1}{2}\sum_{l=-\infty}^{\infty}[H_p(\omega-2\pi(\frac{f_c[k]}{F_s}+l),k) + H_p(\omega+2\pi(\frac{f_c[k]}{F_s}+l),k)], \\ H_Q(\omega,k) &= \tfrac{j}{2}\sum_{l=-\infty}^{\infty}[H_p(\omega-2\pi(\frac{f_c[k]}{F_s}+l),k) - H_p(\omega+2\pi(\frac{f_c[k]}{F_s}+l),k)]. \end{aligned} \tag{181}$$

The spacing between centre frequencies on the Bark scale is

$$\Delta z = \frac{B(f_U) - B(f_L)}{N_c - 1}. \tag{182}$$

For the parameters given, Δz is about 0.707 Bark.[11] The bandwidths of the filters should then be about this amount. An approximate expression for the filter lengths can be derived,

$$\hat{N}[k] = \frac{aF_s}{B^{-1}(z_c[k] + \Delta z/2) - B^{-1}(z_c[k] - \Delta z/2)}, \tag{183}$$

where $z_c[k] = B(f_c[k])$ is the centre frequency on the Bark scale. Choosing parameter a to be equal to 2 gives filter lengths close to the values tabulated in BS.1387. This value of a corresponds to overlapping the bandpass filters at their -6 dB points.

[11] Reference [2] gives the bandwidth as 0.6 Bark.

Appendix E Error Harmonic Structure

This appendix examines two aspects of the calculation of the EHS_B model output variable. First the effect of a shift in the correlation values is analyzed. In the second subsection, the alternative approaches to removing the mean of the correlations are examined.

E.1 Correlation Lag Values

The standard seems to indicate that the correlation values should be calculated for lags 1 through $L_{\max}$. An alternative is to calculate the correlation values for lags 0 through $L_{\max}-1$.

First, consider using lags 0 through $L_{\max}-1$. The windowed correlation (including mean removal before windowing) is

$$C_{w0}[l] = H[l](C[l] - \bar{C}_0) \qquad 0 \le l \le L_{\max} - 1. \tag{184}$$

The term $\bar{C}_0$ is the mean of the correlation values $C[0]$ through $C[L_{\max}-1]$. The correlation power spectrum is the squared magnitude of the DFT of the windowed correlation sequence.

A circular shift of the windowed correlation adds a phase term to the DFT. This phase term will disappear on calculating the squared magnitude. Then for the purpose of calculating the correlation power spectrum, we can use the circularly shifted windowed correlation,

$$\tilde{C}_{w0}[l] = \begin{cases} H[l+1](C[l+1] - \bar{C}_0) & 0 \le l \le L_{\max} - 2, \\ H[0](C[0] - \bar{C}_0) & l = L_{\max} - 1. \end{cases} \tag{185}$$

Since for the Hann window $H[0]$ is zero, this is really just a shift of the windowed correlation sequence. This shifted correlation sequence gives the same correlation power spectrum as the original correlation sequence.

Now consider using lags 1 through $L_{\max}$. The windowed correlation is then

$$C_{w1}[l] = H[l](C[l+1] - \bar{C}_1) \qquad 0 \le l \le L_{\max} - 1. \tag{186}$$

The term $\bar{C}_1$ is the mean of the correlation values $C[1]$ through $C[L_{\max}]$. For the Hann window, $H[L_{\max}-1]$ is zero.

The difference between the two windowed correlations is

$$C_{w1}[l]-\tilde{C}_{w0}[l]=\begin{cases}(H[l]-H[l+1])(C[l+1]-\bar{C}_0)+H[l]\dfrac{C[0]-C[L_{\max}]}{N_L} & 0\le l\le L_{\max}-2,\\ 0 & l=L_{\max}-1.\end{cases} \quad (187)$$

This second line of the expression has used the fact that the end points of the Hann window are zero. The difference in windowed correlations has two parts. The first is due to a shift by one sample of the window. The second is a correction for the differences in mean values.

Numerical results comparing these approaches are presented at the end of the next section.

E.2 Correlation Mean Removal

This section examines alternative methods to remove the mean of the correlation function used in the calculation of the EHS_B model output variable. For simplicity of notation, we deal with the correlation vector indexed from 0 to N_L-1. The correlation will be windowed with a Hann window ($H[l]$). The DFT of the windowed correlation sequence will be used to calculate a correlation power spectrum. The DFT under consideration is

$$S[k]=\left|\frac{1}{N_L}\sum_{l=0}^{N_L-1}\left(H[l]\left(C[l]-\bar{C}_a\right)-\bar{C}_b\right)W_{N_L}^{kl}\right|^2, \quad (188)$$

where $W_N=\exp(-j2\pi/N)$.

Mean Removed After Windowing

Consider removing the mean after windowing. Then $\bar{C}_a=0$ and

$$\bar{C}_b=\frac{1}{N_L}\sum_{l=0}^{N_L-1}H[l]C[l]. \quad (189)$$

With this choice of $\bar{C}_b$, the term $S[0]$ becomes zero without affecting $S[k]$ for other values of k.

Mean Removed Before Windowing

Consider removing the mean before windowing. Then $\bar{C}_b=0$ and the power spectrum becomes

$$S[k] = \left| \frac{1}{N_L} \sum_{l=0}^{N_L-1} H[l]C[l]W_{N_L}^{kl} - \frac{\bar{C}_a}{N_L} \sum_{l=0}^{N_L-1} H[l]W_{N_L}^{kl} \right|^2 . \quad (190)$$

The DFT of the window has a main lobe centred at $k = 0$. In the expression above, a scaled version of this main lobe is subtracted from the spectrum of the windowed correlation.

Choose $\bar{C}_a$ to be the value which sets $S[0] = 0$,

$$\bar{C}_a = \frac{\sum_{l=0}^{N_L-1} H[l]C[l]}{\sum_{l=0}^{N_L-1} H[l]}. \quad (191)$$

An alternate is to set $\bar{C}_a$ equal to the simple mean of $C[l]$,

$$\bar{C}_a = \frac{1}{N_L} \sum_{l=0}^{N_L-1} C[l]. \quad (192)$$

This choice does not necessarily set the modified spectrum at $k = 0$ equal to zero.

Tests with Alternate Methods for Mean Removal

The alternate methods for mean removal were compared. The reference input was a monaural 1 kHz triangular wave. The test input was the reference signal after being coded with a low rate MP3 audio coder. The test signal was time aligned and gain aligned with the reference. The figure below shows the correlation power spectra calculated as part of the EHS_B model output variable computation (N_L is 256).

The plot shows 8 different conditions. One set of four is for correlation lags 0 to 255. The second set of four is for correlation lags 1 to 256. These sets of curves are not identical, but at the resolution of the plot are indistinguishable.[12] Within a set of 4 curves, different mean removal approaches are compared.

The correlation power spectrum with the non-zero value at time zero (dashed line) was generated with the original correlation (no mean removal). The half-width of the main lobe of the Hann window is two samples. Removing the mean after windowing gives the curve which is zero

[12] An expanded view correlation power spectrum for the same input data for the case of correlation lags 1 to 256 and mean removal before windowing is shown in Fig. 13.

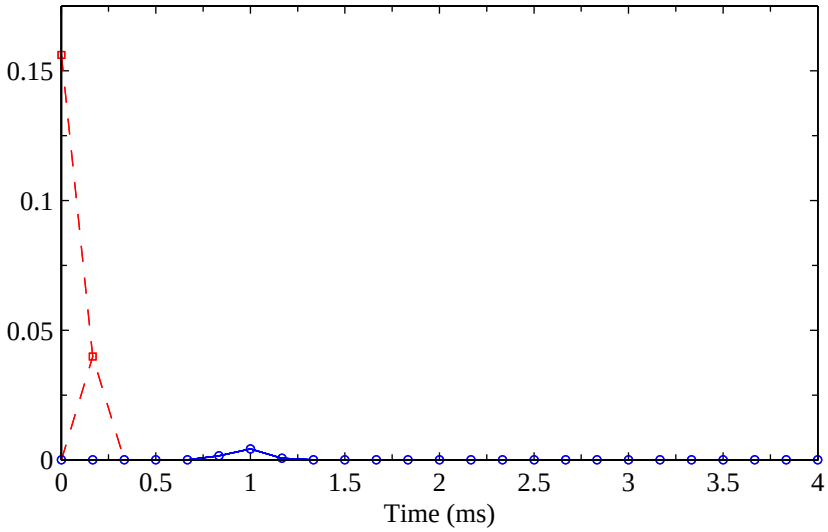

Fig. 17 Correlation power spectrum for different methods of mean removal. Upper dashed line near time zero: no mean removal. Dashed line starting at zero: mean removal after windowing. Other curves: mean removal before windowing. The reference signal is a 1 kHz triangular wave. The test signal is a low rate coded version of the reference signal.

at time zero and then joins the upper curve. The two alternate means to remove the mean before windowing have essentially the same curves which hug zero for the first two points. All curves coalesce at the third sample.

Removing the mean before windowing is effective in removing the “spectral leakage” from the main lobe of the window at time zero. The two alternatives for removal of the mean before windowing are both effective. Since the goal of the Error Harmonic Spectrum model output variable is to measure the height of the largest peak away from zero, removing the mean before windowing is useful in making that peak stand out.

For this example, using correlation lags 0 through 255 or 1 through 256 makes little difference. In other examples, the differences in the peak value used in the EHS_B MOV calculation were found to be less than 2%.

Appendix F FFT-Based Ear Model – Matlab Code

This appendix contains Matlab code for parts of BS.1387. These routines were used to verify the correctness of code that was restructured for efficiency.

F.1 FFT Processing

```
function X2 = PQDFTFrame (x)
% Calculate the DFT of a frame of data (NF values), returning the
% squared-magnitude DFT vector (NF/2 + 1 values)

% P. Kabal $Revision: 1.1 $  $Date: 2002/03/30 05:07:59 $

persistent hw

NF = 2048;      % Frame size (samples)

if (isempty (hw))
    Amax = 32768;
    fc = 1019.5;
    Fs = 48000;
    Lp = 92;
    % Set up the window (including all gains)
    GL = PQ_GL (NF, Amax, fc/Fs, Lp);
    hw = GL * PQHannWin (NF);
end

% Window the data
xw = hw .* x;

% DFT (output is real followed by imaginary)
X = PQRFFT (xw, NF, 1);

% Squared magnitude
X2 = PQRFFTMSq (X, NF);

%---------------------------------------
function GL = PQ_GL (NF, Amax, fcN, Lp)
% Scaled Hann window, including loudness scaling

% Calculate the gain for the Hann Window
%  - level Lp (SPL) corresponds to a sine with normalized frequency
%    fcN and a peak value of Amax

W = NF - 1;
gp = PQ_gp (fcN, NF, W);
GL = 10^(Lp / 20) / (gp * Amax/4 * W);

%----------
function gp = PQ_gp (fcN, NF, W)
% Calculate the peak factor. The signal is a sinusoid windowed with
% a Hann window. The sinusoid frequency falls between DFT bins. The
% peak of the frequency response (on a continuous frequency scale) falls
% between DFT bins. The largest DFT bin value is the peak factor times
% the peak of the continuous response.
```

```
%  fcN - Normalized sinusoid frequency (0-1)
%  NF  - Frame (DFT) length samples
%  NW  - Window length samples

% Distance to the nearest DFT bin
df = 1 / NF;
k = floor (fcN / df);
dfN = min ((k+1) * df - fcN, fcN - k * df);

dfW = dfN * W;
gp = sin(pi * dfW) / (pi * dfW * (1 - dfW^2));
```

```
function X = PQRFFT (x, N, ifn)
% Calculate the DFT of a real N-point sequence or the inverse
% DFT corresponding to a real N-point sequence.
% ifn > 0, forward transform
%          input x(n)  - N real values
%          output X(k) - The first N/2+1 points are the real
%            parts of the transform, the next N/2-1 points
%            are the imaginary parts of the transform. However
%            the imaginary part for the first point and the
%            middle point which are known to be zero are not
%            stored.
% ifn < 0, inverse transform
%          input X(k) - The first N/2+1 points are the real
%            parts of the transform, the next N/2-1 points
%            are the imaginary parts of the transform. However
%            the imaginary part for the first point and the
%            middle point which are known to be zero are not
%            stored.
%          output x(n) - N real values

% P. Kabal $Revision: 1.1 $  $Date: 2002/03/30 05:14:40 $

if (ifn > 0)
    X = fft (x, N);
    XR = real(X(0+1:N/2+1));
    XI = imag(X(1+1:N/2-1+1));
    X = [XR XI];
else
    xR = [x(0+1:N/2+1)];
    xI = [0 x(N/2+1+1:N-1+1) 0];
    x = complex ([xR xR(N/2-1+1:-1:1+1)], [xI -xI(N/2-1+1:-1:1+1)]);
    X = real (ifft (x, N));
end
```

```
function X2 = PQRFFTMSq (X, N)
% Calculate the magnitude squared frequency response from the
% DFT values corresponding to a real signal (assumes N is even)

% P. Kabal $Revision: 1.1 $  $Date: 2002/03/30 05:15:39 $

X2 = zeros (1, N/2+1);
```

```
X2(0+1) = X(0+1)^2;

for (k = 1:N/2-1)

    X2(k+1) = X(k+1)^2 + X(N/2+k+1)^2;
end
X2(N/2+1) = X(N/2+1)^2;
```

F.2 Critical Band Parameters

```
function [Nc, fc, fl, fu, dz] = PQCB (Version)
% Critical band parameters for the FFT model

% P. Kabal $Revision: 1.1 $  $Date: 2002/03/30 05:07:16 $

B = inline ('7 * asinh (f / 650)');
BI = inline ('650 * sinh (z / 7)');

fL = 80;
fU = 18000;

% Critical bands - set up the tables
if (strcmp (Version, 'Basic'))
    dz = 1/4;
elseif (strcmp (Version, 'Advanced'))
    dz = 1/2;
else
    error ('PQCB: Invalid version');
end

zL = B(fL);
zU = B(fU);
Nc = ceil((zU - zL) / dz);
zl = zL + (0:Nc-1) * dz;
zu = min (zL + (1:Nc) * dz, zU);
zc = 0.5 * (zl + zu);

fl = BI (zl);
fc = BI (zc);
fu = BI (zu);
```

F.3 Critical Band Grouping

```
function Eb = PQgroupCB (X2, Version)
% Group a DFT energy vector into critical bands
% X2 - Squared-magnitude vector (DFT bins)
% Eb - Excitation vector (fractional critical bands)

% P. Kabal $Revision: 1.2 $  $Date: 2002/04/18 18:21:26 $

persistent Nc kl ku Ul Uu Ver

Emin = 1e-12;
```

```
if (Ver ~= Version)
    Ver = Version;
    % Set up the DFT bin to critical band mapping
    NF = 2048;

    Fs = 48000;

    [Nc, kl, ku, Ul, Uu] = PQ_CBMapping (NF, Fs, Version);
end

% Allocate storage
Eb = zeros (1, Nc);

% Compute the excitation in each band
for (i = 0:Nc-1)
    Ea = Ul(i+1) * X2(kl(i+1)+1);       % First bin
    for (k = (kl(i+1)+1):(ku(i+1)-1))
        Ea = Ea + X2(k+1);              % Middle bins
    end
    Ea = Ea + Uu(i+1) * X2(ku(i+1)+1);  % Last bin
    Eb(i+1) = max(Ea, Emin);
end

%---------------------------------------
function [Nc, kl, ku, Ul, Uu] = PQ_CBMapping (NF, Fs, Version)

[Nc, fc, fl, fu] = PQCB (Version);

% Fill in the DFT bin to critical band mappings
df = Fs / NF;
for (i = 0:Nc-1)
    fli = fl(i+1);
    fui = fu(i+1);
    for (k = 0:NF/2)
        if ((k+0.5)*df > fli)
            kl(i+1) = k;        % First bin in band i
            Ul(i+1) = (min(fui, (k+0.5)*df) ...
                       - max(fli, (k-0.5)*df)) / df;
            break;
        end
    end
    for (k = NF/2:-1:0)
        if ((k-0.5)*df < fui)
            ku(i+1) = k;        % Last bin in band i
            if (kl(i+1) == ku(i+1))
                Uu(i+1) = 0;       % Single bin in band
            else
                Uu(i+1) = (min(fui, (k+0.5)*df) ...
                           - max(fli, (k-0.5)*df)) / df;
            end
            break;
        end
    end
end
```

F.4 Spreading (DFT-Based Model)

```
function Es = PQspreadCB (E, Version)
% Spread an excitation vector (pitch pattern) - FFT model

% P. Kabal $Revision: 1.2 $  $Date: 2002/04/18 18:25:06 $

persistent Bs Ver

if (Ver ~= Version)
    Ver = Version;
    Nc = length (E);
    Bs = PQ_SpreadCB (ones(1,Nc), ones(1,Nc), Version);
end

Es = PQ_SpreadCB (E, Bs, Version);

%-------------------------
function Es = PQ_SpreadCB (E, Bs, Version);

persistent Nc dz fc aL aUC Ver

% Power law for addition of spreading
e = 0.4;

if (Ver ~= Version)
    Ver  = Version;
    [Nc, fc, fl, fu, dz] = PQCB (Version);
end

% Allocate storage
aUCEe = zeros (1, Nc);
Ene = zeros (1, Nc);
Es = zeros (1, Nc);

% Calculate energy dependent terms
aL = 10^(-2.7 * dz);
for (m = 0:Nc-1)
    aUC = 10^((-2.4 - 23 / fc(m+1)) * dz);
    aUCE = aUC * E(m+1)^(0.2 * dz);
    gIL = (1 - aL^(m+1)) / (1 - aL);
    gIU = (1 - aUCE^(Nc-m)) / (1 - aUCE);
    En = E(m+1) / (gIL + gIU - 1);
    aUCEe(m+1) = aUCE^e;
    Ene(m+1) = En^e;
end

% Lower spreading
Es(Nc-1+1) = Ene(Nc-1+1);
aLe = aL^e;
for (m = Nc-2:-1:0)
    Es(m+1) = aLe * Es(m+1+1) + Ene(m+1);
end

% Upper spreading i > m
for (m = 0:Nc-2)
    r = Ene(m+1);
```

```
    a = aUCEe(m+1);
    for (i = m+1:Nc-1)
       r = r * a;
       Es(i+1) = Es(i+1) + r;
   end
end

for (i = 0:Nc-1)
    Es(i+1) = (Es(i+1))^(1/e) / Bs(i+1);
end
```

Appendix G Pattern Processing – Matlab Code

G.1 Level and Pattern Adaptation

```
function [EP, Fmem] = PQadapt (Ehs, Fmem, Ver, Mod)
% Level and pattern adaptation

% P. Kabal $Revision: 1.2 $  $Date: 2002/04/03 12:40:46 $

persistent a b Nc M1 M2 Version Model

if (~strcmp (Ver, Version) | ~strcmp (Mod, Model))
    Version = Ver;
    Model = Mod;
    if (strcmp (Model, 'FFT'))
        [Nc, fc] = PQCB (Version);
        NF = 2048;
        Nadv = NF / 2;
    else
        [Nc, fc] = PQFB;
        Nadv = 192;
    end
    Version = Ver;
    Model = Mod;
    Fs = 48000;
    Fss = Fs / Nadv;
    t100 = 0.050;
    tmin = 0.008;
    [a b] = PQtConst (t100, tmin, fc, Fss);
    [M1, M2] = PQ_M1M2 (Version, Model);
end

% Allocate memory
EP = zeros (2, Nc);
R = zeros (2, Nc);

% Smooth the excitation patterns
% Calculate the correlation terms
sn = 0;
sd = 0;
for (m = 0:Nc-1)
    Fmem.P(1,m+1) = a(m+1) * Fmem.P(1,m+1) + b(m+1) * Ehs(1,m+1);
    Fmem.P(2,m+1) = a(m+1) * Fmem.P(2,m+1) + b(m+1) * Ehs(2,m+1);
    sn = sn + sqrt (Fmem.P(2,m+1) * Fmem.P(1,m+1));
    sd = sd + Fmem.P(2,m+1);
end

% Level correlation
CL = (sn / sd)^2;

for (m = 0:Nc-1)

% Scale one of the signals to match levels
    if (CL > 1)
        EP(1,m+1) = Ehs(1,m+1) / CL;
        EP(2,m+1) = Ehs(2,m+1);
```

```
    else
        EP(1,m+1) = Ehs(1,m+1);

        EP(2,m+1) = Ehs(2,m+1) * CL;
    end

% Calculate a pattern match correction factor
    Fmem.Rn(m+1) = a(m+1) * Fmem.Rn(m+1) + EP(2,m+1) * EP(1,m+1);
    Fmem.Rd(m+1) = a(m+1) * Fmem.Rd(m+1) + EP(1,m+1) * EP(1,m+1);
    if (Fmem.Rd(m+1) <= 0 | Fmem.Rn(m+1) <= 0)
        error ('>>> PQadap: Rd or Rn is zero');
    end
    if (Fmem.Rn(m+1) >= Fmem.Rd(m+1))
        R(1,m+1) = 1;
        R(2,m+1) = Fmem.Rd(m+1) / Fmem.Rn(m+1);
    else
        R(1,m+1) = Fmem.Rn(m+1) / Fmem.Rd(m+1);
        R(2,m+1) = 1;
    end
end

% Average the correction factors over M channels and smooth with time
% Create spectrally adapted patterns
for (m = 0:Nc-1)
    iL = max (m - M1, 0);
    iU = min (m + M2, Nc-1);
    s1 = 0;
    s2 = 0;
    for (i = iL:iU)
        s1 = s1 + R(1,i+1);
        s2 = s2 + R(2,i+1);
    end
    Fmem.PC(1,m+1) = a(m+1) * Fmem.PC(1,m+1) + b(m+1) * s1 / (iU-iL+1);
    Fmem.PC(2,m+1) = a(m+1) * Fmem.PC(2,m+1) + b(m+1) * s2 / (iU-iL+1);

    % Final correction factor => spectrally adapted patterns
    EP(1,m+1) = EP(1,m+1) * Fmem.PC(1,m+1);
    EP(2,m+1) = EP(2,m+1) * Fmem.PC(2,m+1);
end

%--------------------------------------
function [M1, M2] = PQ_M1M2 (Version, Model)
% Return band averaging parameters

if (strcmp (Version, 'Basic'))
    M1 = 3;
    M2 = 4;
elseif (strcmp (Version, 'Advanced'))
    if (strcmp (Model, 'FFT'))
        M1 = 1;
        M2 = 2;
    else
        M1 = 1;
        M2 = 1;
    end
end
```

G.2 Modulation Patterns

```
function [M, ERavg, Fmem] = PQmodPatt (Es, Fmem)

% Modulation pattern processing

% P. Kabal $Revision: 1.2 $  $Date: 2002/04/03 03:24:59 $

persistent Nc a b Fss

if (isempty (Nc))
    Fs = 48000;
    NF = 2048;
    Fss = Fs / (NF/2);
    [Nc, fc] = PQCB ('Basic');
    t100 = 0.050;
    t0 = 0.008;
    [a, b] = PQtConst (t100, t0, fc, Fss);
end

% Allocate memory
M = zeros (2, Nc);

e = 0.3;
for (i = 1:2)
    for (m = 0:Nc-1)
        Ee = Es(i,m+1)^e;
        Fmem.DE(i,m+1) = a(m+1) * Fmem.DE(i,m+1) ...
                       + b(m+1) * Fss * abs (Ee - Fmem.Ese(i,m+1));
        Fmem.Eavg(i,m+1) = a(m+1) * Fmem.Eavg(i,m+1) + b(m+1) * Ee;
        Fmem.Ese(i,m+1) = Ee;

        M(i,m+1) = Fmem.DE(i,m+1) / (1 + Fmem.Eavg(i,m+1)/0.3);
    end
end

ERavg = Fmem.Eavg(1,:);
```

G.3 Loudness Calculation

```
function Ntot = PQloud (Ehs, Ver, Mod)
% Calculate the loudness

% P. Kabal $Revision: 1.1 $  $Date: 2002/03/30 04:56:27 $

e = 0.23;

persistent Nc s Et Ets Version Model

if (~strcmp (Ver, Version) | ~strcmp (Mod, Model))
    Version = Ver;
    Model = Mod;
    if (strcmp (Model, 'FFT'))
        [Nc, fc] = PQCB (Version);
        c = 1.07664;
    else
        [Nc, fc] = PQFB;
```

```
        c = 1.26539;
    end
    E0 = 1e4;
    Et = PQ_enThresh (fc);
    s = PQ_exIndex (fc);

    for (m = 0:Nc-1)

        Ets(m+1) = c * (Et(m+1) / (s(m+1) * E0))^e;
    end
end

sN = 0;
for (m = 0:Nc-1)
    Nm = Ets(m+1) * ((1 - s(m+1) + s(m+1) * Ehs(m+1) / Et(m+1))^e - 1);
    sN = sN + max(Nm, 0);
end
Ntot = (24 / Nc) * sN;

%====================
function s = PQ_exIndex (f)
% Excitation index

N = length (f);
for (m = 0:N-1)
    sdB = -2 - 2.05 * atan(f(m+1) / 4000) - 0.75 * atan((f(m+1) / 1600)^2);
    s(m+1) = 10^(sdB / 10);
end

%--------------------
function Et = PQ_enThresh (f)
% Excitation threshold

N = length (f);
for (m = 0:N-1)
    EtdB = 3.64 * (f(m+1) / 1000)^(-0.8);
    Et(m+1) = 10^(EtdB / 10);
end
```

Appendix H Model Output Parameters – Matlab Code

H.1 Modulation Differences

```
function MDiff = PQmovModDiffB (M, ERavg)
Modulation difference related MOV precursors (Basic version)

% P. Kabal $Revision: 1.2 $  $Date: 2002/04/03 03:29:04 $

persistent Nc Ete

if (isempty (Nc))
    e = 0.3;
    [Nc, fc] = PQCB ('Basic');
    Et = PQIntNoise (fc);
    for (m = 0:Nc-1)
        Ete(m+1) = Et(m+1)^e;
    end
end

% Parameters
negWt2B = 0.1;
offset1B = 1.0;
offset2B = 0.01;
levWt = 100;

s1B = 0;
s2B = 0;
Wt = 0;
for (m = 0:Nc-1)
    if (M(1,m+1) > M(2,m+1))
        num1B = M(1,m+1) - M(2,m+1);
        num2B = negWt2B * num1B;
    else
        num1B = M(2,m+1) - M(1,m+1);
        num2B = num1B;
    end
    MD1B = num1B / (offset1B + M(1,m+1));
    MD2B = num2B / (offset2B + M(1,m+1));
    s1B = s1B + MD1B;
    s2B = s2B + MD2B;
    Wt = Wt + ERavg(m+1) / (ERavg(m+1) + levWt * Ete(m+1));
end

MDiff.Mt1B = (100 / Nc) * s1B;
MDiff.Mt2B = (100 / Nc) * s2B;
MDiff.Wt = Wt;
```

H.2 Noise Loudness

```
function NL = PQmovNLoudB (M, EP)
% Noise Loudness

% P. Kabal $Revision: 1.1 $  $Date: 2002/04/03 03:28:29 $
```

```
persistent Nc Et
```

```
if (isempty (Nc))
    [Nc, fc] = PQCB ('Basic');
    Et = PQIntNoise (fc);
end

% Parameters
alpha = 1.5;
TF0 = 0.15;
S0 = 0.5;
NLmin = 0;
e = 0.23;

s = 0;
for (m = 0:Nc-1)
    sref  = TF0 * M(1,m+1) + S0;
    stest = TF0 * M(2,m+1) + S0;
    beta = exp (-alpha * (EP(2,m+1) - EP(1,m+1)) / EP(1,m+1));
    a = max (stest * EP(2,m+1) - sref * EP(1,m+1), 0);
    b = Et(m+1) + sref * EP(1,m+1) * beta;
    s = s + (Et(m+1) / stest)^e * ((1 + a / b)^e - 1);
end

NL = (24 / Nc) * s;
if (NL < NLmin)
    NL = 0;
end
```

H.3 Noise-to-Mask Ratio

```
function NMR = PQmovNMRB (EbN, Ehs)
% Noise-to-mask ratio - Basic version
% NMR.NMRavg average NMR
% NMR.NMRmax max NMR

% P. Kabal $Revision: 1.2 $  $Date: 2002/04/18 18:27:20 $

persistent Nc gm

if (isempty (Nc))
    [Nc, fc, fl, fu, dz] = PQCB ('Basic');
    gm = PQ_MaskOffset (dz, Nc);
end

NMR.NMRmax = 0;
s = 0;
for (m = 0:Nc-1)
    NMRm = EbN(m+1) / (gm(m+1) * Ehs(m+1));
    s = s + NMRm;
    if (NMRm > NMR.NMRmax)
        NMR.NMRmax = NMRm;
    end
end
NMR.NMRavg = s / Nc;
```

```
%---------------------------------------
function gm = PQ_MaskOffset (dz, Nc)


for (m = 0:Nc-1)
    if (m <= 12 / dz)
        mdB = 3;
    else
        mdB = 0.25 * m * dz;
    end
    gm(m+1) = 10^(-mdB / 10);
end
```

H.4 Bandwidth

```
function BW = PQmovBW (X2)
% Bandwidth tests

% P. Kabal $Revision: 1.1 $  $Date: 2002/03/30 05:00:31 $

persistent kx kl FR FT N

if (isempty (kx))
    NF = 2048;
    Fs = 48000;
    fx = 21586;
    kx = round (fx / Fs * NF);    % 921
    fl = 8109;
    kl = round (fl / Fs * NF);    % 346
    FRdB = 10;
    FR = 10^(FRdB / 10);
    FTdB = 5;
    FT = 10^(FTdB / 10);
    N = NF / 2;     % Limit from pseudo-code
end

Xth = X2(2,kx+1);
for (k = kx+1:N-1)
    Xth = max (Xth, X2(2,k+1));
end

% BWRef and BWTest remain negative if the BW of the test signal
% does not exceed FR * Xth for kx-1 <= k <= kl+1
BW.BWRef = -1;
XthR = FR * Xth;
for (k = kx-1:-1:kl+1)
    if (X2(1,k+1) >= XthR)
        BW.BWRef = k + 1;
        break;
    end
end

BW.BWTest = -1;
XthT = FT * Xth;
for (k = BW.BWRef-1:-1:0)
    if (X2(2,k+1) >= XthT)
        BW.BWTest = k + 1;
```

```
        break;
    end
end
```

H.5 Probability of Detection

```
function PD = PQmovPD (Ehs)
% Probability of detection

% P. Kabal $Revision: 1.3 $  $Date: 2002/04/08 13:59:08 $

Nc = length (Ehs);

% Allocate storage
PD.p = zeros (1, Nc);
PD.q = zeros (1, Nc);

persistent c g d1 d2 bP bM

if (isempty (c))
    c = [-0.198719 0.0550197 -0.00102438 5.05622e-6 9.01033e-11];
    d1 = 5.95072;
    d2 = 6.39468;
    g = 1.71332;
    bP = 4;
    bM = 6;
end

for (m = 0:Nc-1)
    EdBR = 10 * log10 (Ehs(1,m+1));
    EdBT = 10 * log10 (Ehs(2,m+1));
    edB = EdBR - EdBT;
    if (edB > 0)
        L = 0.3 * EdBR + 0.7 * EdBT;
        b = bP;
    else
        L = EdBT;
        b = bM;
    end
    if (L > 0)
        s = d1 * (d2 / L)^g ...
            + c(1) + L * (c(2) + L * (c(3) + L * (c(4) + L * c(5))));
    else
        s = 1e30;
    end
    PD.p(m+1) = 1 - 0.5^((edB / s)^b);        % Detection probability
    PD.q(m+1) = abs (fix(edB)) / s;           % Steps above threshold
end
```

H.6 Error Harmonic Structure

```
function EHS = PQmovEHS (xR, xT, X2)
% Calculate the EHS MOV values

% P. Kabal $Revision: 1.4 $  $Date: 2002/04/18 18:41:14 $

persistent NF Nadv NL M Hw kst

if (isempty (NL))
    NF = 2048;
    Nadv = NF / 2;
    Fs = 48000;

    Fmax = 9000;

    NL = 2^(PQ_log2(NF * Fmax / Fs));
    M = NL;
    Hw = (1 / M) * sqrt(8 / 3) * PQHannWin (M);
    kst = 1;
end

EnThr = 8000;
kmax = kst + NL + M - 1;

EnRef  = xR(Nadv+1:NF-1+1) * xR(Nadv+1:NF-1+1)';
EnTest = xT(Nadv+1:NF-1+1) * xT(Nadv+1:NF-1+1)';

% Set the return value to be negative for small energy frames
if (EnRef < EnThr & EnTest < EnThr)
    EHS = -1;
    return;
end

% Allocate storage
D = zeros (1, kmax);

% Differences of log values
for (k = 0:kmax-1)
    D(k+1) = log (X2(2,k+1) / X2(1,k+1));
end
X2(1,1:10)

% Correlation computation
C = PQ_Corr (D, D, kst+NL, M);

% Normalize the correlations
Cn = PQ_NCorr (C, D, kst+NL, M);
Cm = Cn(kst+1:kst+NL) - (1 / NL) * sum (Cn(kst+1:kst+NL));

% Window the correlation
Cm = Hw .* Cm;

% DFT
Cm = PQRFFT (Cm, NL, 1);

% Squared magnitude
c2 = PQRFFTMSq (Cm, NL);
1000 * c2(1:20)
```

```
% Search for a peak after a valley
EHS = PQ_FindPeak (c2, NL/2+1);

%----------------------------------------
function log2 = PQ_log2 (a)

log2 = 0;
m = 1;
while (m < a)
    log2 = log2 + 1;
    m = 2 * m;
end
log2 = log2 - 1;

%----------
function C = PQ_Corr (D0, D1, NL, M)
% Correlation calculation

% Calculate the correlation using DFT's
NFFT = 512;
D0x = [D0(1:M) zeros(1,NFFT-M)];
D1x = [D1(1:M+NL-1) zeros(1,NFFT-(M+NL-1))];

% DFTs of the zero-padded sequences
D0x = PQRFFT (D0x, NFFT, 1);
D1x = PQRFFT (D1x, NFFT, 1);

% Multiply the (complex) DFT sequences (D0x is conjugated)
dx(0+1) = D0x(0+1) * D1x(0+1);
for (n = 1:NFFT/2-1)
    m = NFFT/2 + n;         % n => real part, m => imaginary part
    dx(n+1) = D0x(n+1) * D1x(n+1) + D0x(m+1) * D1x(m+1);
    dx(m+1) = D0x(n+1) * D1x(m+1) - D0x(m+1) * D1x(n+1);
end
dx(NFFT/2+1) = D0x(NFFT/2+1) * D1x(NFFT/2+1);

% Inverse DFT
cx = PQRFFT (dx, NFFT, -1);
C = cx(1:NL);

%----------
function Cn = PQ_NCorr (C, D, NL, M)
% Normalize the correlation

Cn = zeros (1, NL);

s0 = C(0+1);
sj = s0;
Cn(0+1) = 1;
for (i = 1:NL-1)
    sj = sj + (D(i+M-1+1)^2 - D(i-1+1)^2);
    d = s0 * sj;
    if (d <= 0)
        Cn(i+1) = 1;
    else
        Cn(i+1) = C(i+1) / sqrt (d);
    end
end
```

```
%----------
function EHS = PQ_FindPeak (c2, N)
% Search for a peak after a valley

cprev = c2(0+1);
cmax = 0;
for (n = 1:N-1)
    if (c2(n+1) > cprev)    % Rising from a valley
        if (c2(n+1) > cmax)
            cmax = c2(n+1);
        end
    end
end
EHS = cmax;
```