Digital Audio Processing
6. Audio Coding

Nicolas Obin
nicolas.obin@sorbonne-universite.fr
2021
Sorbonne Université

On the Origins of Audio Coding

The origins of audio coding

The Vocoder, abbreviation for “VOice enCODER”, is first system proposed for audio coding, originally applied to speech.

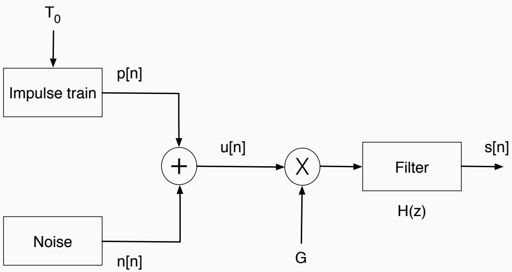
- Invented and patented by Homer Dudley at Bell Labs in 1935
- Objective: reducing the bandwidth necessary for transmitting speech through a communication network
- At analysis: coding a speech signal by measuring the frequency response a_k of a filterbank composed of K filters representing isolated frequency bands
- At synthesis: the speech signal is synthesized by passing an excitation signal through the previous filterbank

Source/Filter Model

The source/filter model is a simplified signal representation of sound production by a physical system, which is commonly used in audio coding (speech, music).

The source/filter model assumes that the audio signal $s[n]$ results from the excitation of a filter $H(z)$ by a source signal $u[n]$:

$$s[n] = (Gu * h)[n] \tag{1}$$



Schematic illustration of the source/filter model.

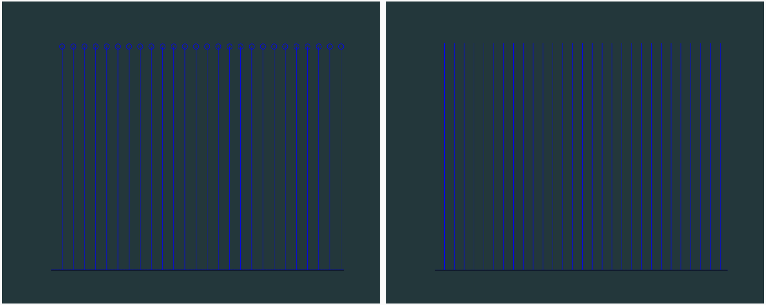
The source signal $u[n]$ is a mix of a periodic signal $p[n]$ and a noise signal $n[n]$:

$$u[n] = G(p[n] + n[n]) \tag{2}$$

- The periodic signal $p[n]$ is an impulse train with fundamental period T_0 (respectively, fundamental frequency F_0):

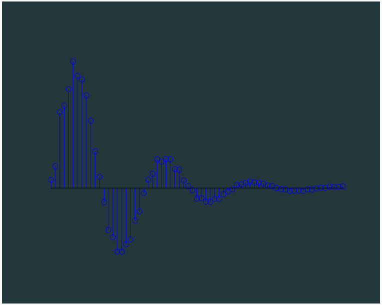
$$p[n] = \text{Ш}_{T_0}[n] \tag{3}$$

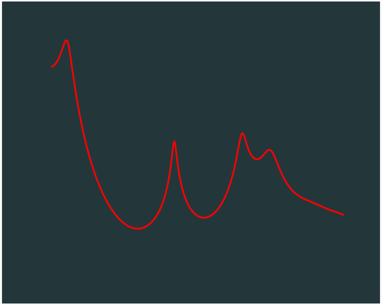
- The noise signal $n[n]$ is a random signal.



Impulse train $p[n]$ at T_0 (left), and corresponding TFD $P[k]$ at F_0 (right)

The filter is defined by its transfer function $H(z)$:





Filter: impulse response $h[n]$ (left), and frequency response $H(z)$(right).

The convolution in the time domain is a multiplication in the frequency domain:

$$S(z) = GU(z) \times H(z) \tag{4}$$

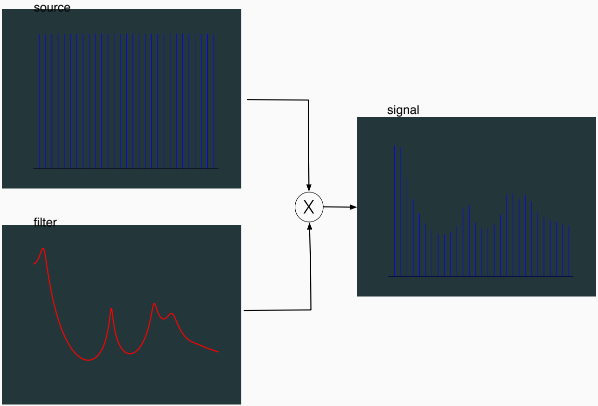


Illustration of the source/filter model in the frequency domain.

Audio Coding

Introduction

Linear Predictive Coding (LPC) is historically the most widespread method for the estimation of the source/filter parameters in audio coding, recognition, and synthesis.

The basic assumption of LPC is that the current audio sample can be approximated as the linear combination of past P audio samples:

$$\hat{s}[n] = \sum_{p=1}^{P} \alpha_p s[n-p] \tag{5}$$

The corresponding TZ is:

$$\hat{S}[z] = \sum_{p=1}^{P} \alpha_p S[z] z^{-p} \tag{6}$$

J. Makhoul. Linear Prediction : A Tutorial Review. In Proceedings of the IEEE, volume 63, pages 561–580, Cambridge, Massachusetts, 1975.

LPC and the source/filter production model of speech

Firstly, we consider the source/filter model

Hypothesis: the transfer function of the vocal-tract $H(z)$ is a all-pole filter

$$H(z) = \frac{S(z)}{GU(z)} = \frac{1}{1 - \sum_{p=1}^{P} a_p z^{-p}} \tag{7}$$

This can be written in the time domain as:

$$s[n] = \sum_{p=1}^{P} a_p s[n-p] + Gu[n] \tag{8}$$

Prediction Model

Secondly, we consider the LPC model.

The current sample $\hat{s}[n]$ as predicted by LPC is:

$$\hat{s}[n] = \sum_{p=1}^{P} \alpha_p s[n-p] \tag{9}$$

The transfer function of the system defined by input $s[n]$ and by output $\hat{s}[n]$ is:

$$P[z] = \frac{\hat{S}(z)}{S(z)} = \sum_{p=1}^{P} \alpha_p z^{-p} \tag{10}$$

Prediction Error

The prediction error $e[n]$ is defined as the difference between the current sample $s[n]$ and the current sample $\hat{s}[n]$ as predicted by LPC:

$$e[n] = s[n] - \hat{s}[n] \tag{11}$$

By definition:

$$e[n] = s[n] - \sum_{p=1}^{P} \alpha_p s[n-p] \tag{12}$$

The error signal $e[n]$ is also know as the residual signal.

The transfer function of the system defined by input $s[n]$ and by output $e[n]$ is:

$$\boxed{\mathrm{A}[\mathrm{z}] = \frac{E(z)}{S(z)} = 1 - P(z) = 1 - \sum_{p=1}^{P} \alpha_p z^{-p}} \tag{13}$$

LPC and the source/filter production model of speech

Finally, by equating the source/filter model and the LPC model:

$$s[n] = \sum_{p=1}^{P} a_p s[n-p] + Gu[n] = \sum_{p=1}^{P} \alpha_p s[n-p] + e[n] \tag{14}$$

By identification:

$$Gu[n] = e[n] \tag{15}$$

$$\alpha[n] = a[n] \tag{16}$$

Then:

$$\boxed{\mathrm{H(z)} = \frac{1}{A(z)}} \tag{17}$$

Which means that $A(z)$ is the inverse filter of $H(z)$.

Estimation Issues

For stationary signals:

- Determining the source/filter parameters from the Discrete Fourier Transform (DFT).
- Estimating α_p from the Discrete Fourier Transform (DFT).
- The estimation is performed so as to minimize the mean-squared prediction error.
- Idealistically, the LPC is the actual filter.

For non-stationary signals:

- Determining the time-varying filter from the Short-Term Fourier Transform (STFT).

Estimation

The mean squared error is:

$$
\begin{aligned}
E &= \sum_{n=1}^{N} e^2[n] && (18)\\
&= \sum_{n=1}^{N} (s[n] - \hat{s}_n[n])^2 && (19)\\
&= \sum_{n=1}^{N} (s[n] - \sum_{p=1}^{P} \alpha_p s[n-p])^2 && (20)
\end{aligned}
$$

The solution $\{\hat{\alpha}_p\}_{p=1}^{P}$ is determined so as to minimize the short-term mean squared error:

$$
\hat{\alpha}_p = \arg\min_{\alpha_p} E \qquad (21)
$$

Estimation

The solution $\{\hat{\alpha}_p\}_{p=1}^{P}$ is determined by using numerical optimization.

Set partial derivatives to 0 with respect to α_i:

$$\frac{\partial E}{\partial \alpha_i} = 0, \qquad \forall i \in [1, P] \tag{22}$$

This leads to a set of equations:

$$\sum_{n=1}^{N}(s[n-i](s[n] - \sum_{p=1}^{P}\hat{\alpha}_p s[n-p])) = 0, \qquad \forall i \in [1, P] \tag{23}$$

Estimation

After expansion:

$$\sum_{p=1}^{P} \alpha_p \left(\sum_{n=1}^{N} s[n-i]s[n-p] \right) = \sum_{n=1}^{N} s[n]s[n-p] \qquad \forall i \in [1, P] \tag{24}$$

This is a set of P equations with P unknown parameters $\{\alpha_p\}_{p=1}^{P}$

Let consider $R(p)$ the auto-correlation function defined as:

$$R(p) = \sum_{n=1}^{N} s[n]s[n+p] \tag{25}$$

Finally, this can be written as a set of Yule-Walker equations:

$$\sum_{p=1}^{P} \alpha_p R(|i-p|) = R(i) \qquad \forall i \in [1, P] \tag{26}$$

Estimation

This can be expressed in a matrix form:

$$\mathbf{R}\alpha = \mathbf{r} \tag{27}$$

i.e:

$$\begin{bmatrix} R(0) & R(1) & \cdots & R(P-1) \\ R(1) & R(0) & \cdots & R(P-2) \\ \vdots & \vdots & \vdots & \vdots \\ R(P-1) & R(P-2) & \cdots & R(0) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_P \end{bmatrix} = \begin{bmatrix} R(1) \\ R(2) \\ \vdots \\ R(P) \end{bmatrix} \tag{28}$$

This is a Toeplitz matrix (symmetric with all diagonal elements equal)

The solution is obtained by matrix inversion:

$$\alpha = \mathbf{R}^{-1}\mathbf{r} \tag{29}$$

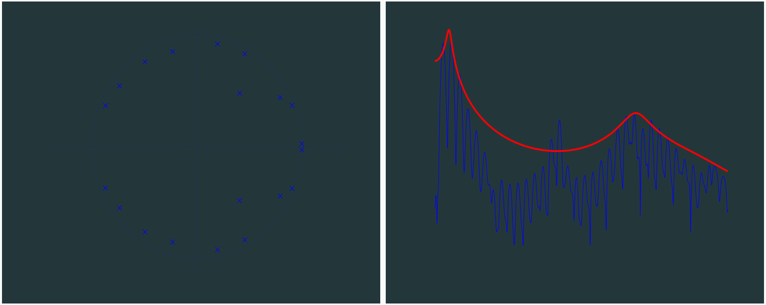
The matrix inversion can be efficiently solved with the Levinson-Durbin algorithm.

Interpretation in the Frequency Domain

LPC approximates the filter response $H(z)$ with an all-pole filter:

$$H(z) = \frac{1}{1 - \sum_{p=1}^{P} \alpha_p z^{-p}} \tag{30}$$

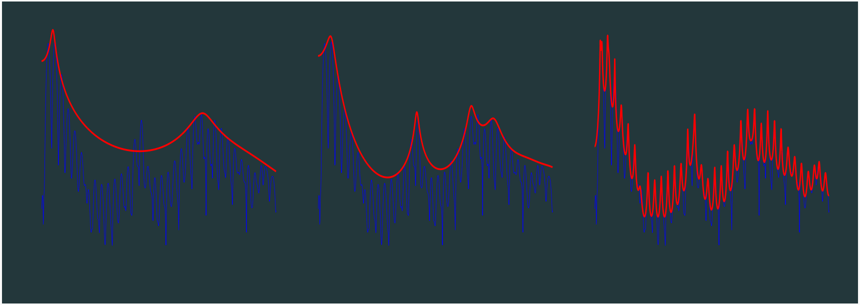
Each pair of poles represents a resonance in the magnitude response of the filter.



Pole positions and transfer function of a LPC filter ($P = 20$)

LPC order

The choice of the LPC order P is crucial for source/filter deconvolution.



From left to right: under-estimated (P = 20), correctly-estimated (P = 50), over-estimated (P = 500) transfer function.

Main issue: no optimal solution for the choice of the order.

Introduction

The source/filter model assumes that the audio signal is the convolution of a source by a filter:

$$s[n] = (e * h)[n] \tag{31}$$

The frequency response of this system is then a point-wise multiplication:

$$S[z] = E(z).H(z) \tag{32}$$

One solution for the source/filter deconvolution is to determine some homomorphic transform $\blacksquare$ for which the convolution is replaced by a sum:

$$s[n] = (e * h)[n] \rightarrow \blacksquare \rightarrow \hat{s}[k] = \hat{e}[k] + \hat{h}[k] \tag{33}$$

If so,

$$\hat{e}[k] \rightarrow \blacksquare^{-1} \rightarrow e[n] \tag{34}$$

$$\hat{h}[k] \rightarrow \blacksquare^{-1} \rightarrow h[n] \tag{35}$$

Complex Cepstrum

The TZ is an homorphic transform which replaces a convolution by a product:

$$s[n] = (e * h)[n] \rightarrow S(z) = E(z).H(z) \tag{36}$$

Then, a simple log transform replaces a product by an sum:

$$S(z) = E(z).H(z) \rightarrow \log \rightarrow \log(S(z)) = \log(E(z)) + \log(H(z)) \tag{37}$$

The inverse TZ preserves the sum, so that:

$$\log(S(z)) = \log(E(z)) + \log(H(z)) \rightarrow TZ^{-1} \rightarrow \hat{s}[k] = \hat{e}[k] + \hat{h}[k] \tag{38}$$

Finally, one homomorphic transform can be defined as:

$$\blacksquare = TZ^{-1}(.) \circ log(.) \circ TZ(.) \tag{39}$$

$$\blacksquare^{-1} = TZ^{-1}(.) \circ exp(.) \circ TZ(.) \tag{40}$$

The DFT version of this transform is: the complex cepstrum.

Real Cepstrum

The main issue of the complex cepstrum is the phase: after the log transform, the phase is defined $\mod (2\pi)$, so that the inverse transform is difficult.

The solution to this issue is to ignore the phase during transforms.

This is the real cepstrum:

$$\blacksquare = DFT^{-1}(.) \circ log(|.|) \circ DFT(.) \tag{41}$$

$$\blacksquare^{-1} = DFT^{-1}(.) \circ exp(|.|) \circ DFT(.) \tag{42}$$

The cepstrum is defined as the DFT of a DFT (!)

Interpretation

For a signal, the DFT is the representation of the signal as a sum of frequency contributions.

Thus, the cepstrum is the representation of the spectrum as a sum of frequency contributions:

- low cepstral bins represent slow variations (low frequencies) in the spectrum
- high cepstral bins represent fast variations (high frequencies) in the spectrum

The cepstrum terminology is just constructed from back slang:

$$\begin{aligned} \textit{spec} - \textit{trum} &= \textit{ceps} - \textit{trum} \\ \textit{freque} - \textit{ncy} &= \textit{quefre} - \textit{ncy} \\ \textit{filt} - \textit{ering} &= \textit{lift} - \textit{ering} \end{aligned}$$

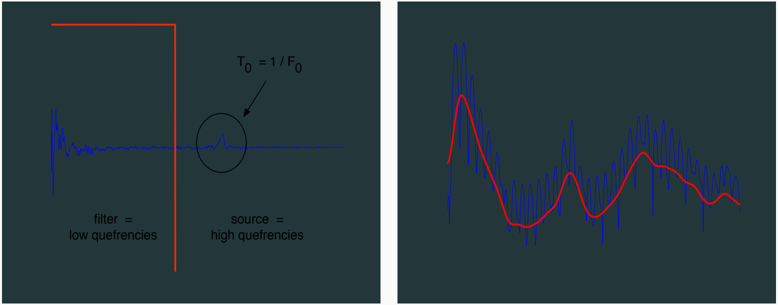
B. P. Bogert, M. J. R. Healy, and J. W. Tukey.
The Quefrency Alanysis of Time Series for Echoes: Cepstrum, Pseudo Autocovariance, Cross-Cepstrum and Saphe Cracking. Proceedings of the Symposium on Time Series Analysis, Chapter 15, 209-243. New York: Wiley, 1963.

Interpretation

Beyond cepstrum, one assumes that source and filter can be linearly separated in the cepstrum domain:

- source: the source has fast frequency variations, with an upper bound defined by fundamental frequency F_0 of the the signal (if periodic).
- filter: the filter has slow frequency variations.

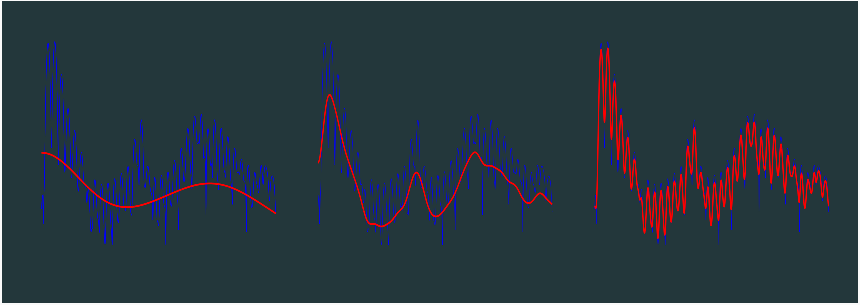
The source/filter deconvolution is obtained through low-quefrency liftering.



Cepstrum Order

Advantage: the optimal solution to the cepstrum order can be directly determined by:

$$P_{opt} = \frac{F_s}{2F_0} \tag{43}$$



From left to right: under-estimated ($P = 25$), correctly-estimated (P_{opt}), over-estimated ($P = 1024$) filter response.

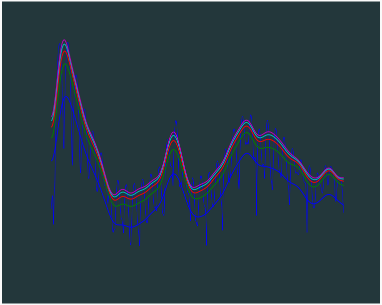
This requires the a priori estimation of the fundamental frequency (F_0)

Limitations

LPC over-estimates the frequency response of the filter.

Limitation: because of averaging, the cepstrum under-estimates the frequency response of the filter.

Solution: the True-Enveloppe method provides iterative estimation of the filter response.

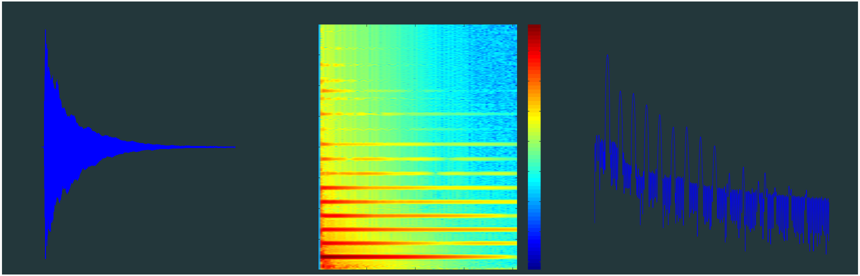


S. Imai et Y. Abe. Spectral Envelope Extraction by Improved Cepstral Method.

Example: Sound Analysis by Synthesis

For a piano, the source/filter model can be interpreted as:

- source
  a noise produced by the strike of the touch on the keyboard, and the strike of the hammer on the chord.
  a periodic impulse train at $F_0 = 440Hz$ produced by dump oscillations of a fixed-length chord after hammer strike on the chord.
- filter
  a filter which is due to the resonance table of the piano



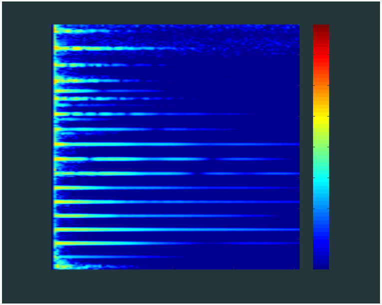
Example: Sound Analysis by Synthesis

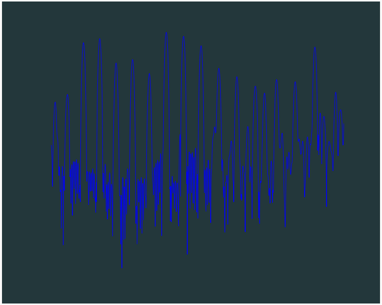
Once the filter $H(z)$ is determined, inverse filtering can be processed to obtain the residual signal $e[n]$.

When the filter $H(z)$ is correctly estimated, the residual equals the source signal:

$$e[n] = u[n] = p[n] + n[n] \tag{44}$$

The residual contains: “white” harmonic comb at F_0, and noise.





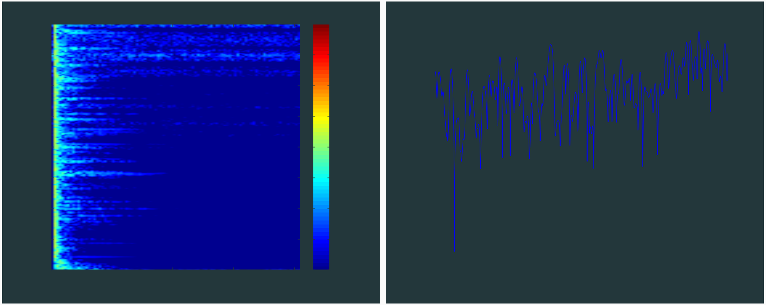
Example: Sound Analysis by Synthesis

When the filter $H(z)$ is over-estimated, then the filter contains the periodic part $p[n]$ of the source signal $u[n]$.

Then, the residual contains the noise part of the source signal:

$$e[n] = n[n] \tag{45}$$

One can clearly hear the attack of the piano (strike of the touch on the keyboard)!



Residual after LPC inverse filtering (P = 500): spectrogram (left), and DFT of a frame (right).

Audio Compression

Why compression?

Compression is important for stocking, transferring, and streaming data

Example for digital audio: Fs = 44,100 Hz, 16 bits, 2 channels

- Data rate: sampling rate * quantification bits * channels
- Without compression: 1.4 Mb / second, about 5 Gb / hour
- With compression: transparent at 128 or 256 Kb / seconds!!!

Existing audio formats

- Without compression: wav, aiff, flac, etc...
- With compression: mp3, aac, etc...

Lossless vs. lossy audio compression

Lossless compression

- Principle: Do not encode what is redundant.
- Example : Entropic enconding, Huffmann algorithm.
- Details: Minimise the entropy of the encoded data, using observation probability distribution of symbols. Less bits to frequent symbols, more bits to rare symbols. Variable-length encoding (vs. fixed-length encoding)

The compression is lossless in the sense that the compressed data is strictly equivalent to the raw data

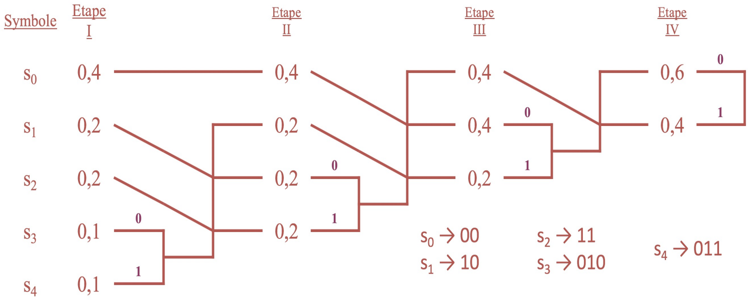
Lossy compression

- Principle: Do not encode what cannot be perceived.
- Example : MPEG encoding.
- Details: Requires a psycho-acoustics model of human audition.

The compression is lossy in the sense that there is a loss of information, but this loss is assumed to be imperceptible by a human

Lossless coding

Le code de Huffman :



$$\bar{n} = 0{,}4\times 2 + 0{,}2\times 2 + 0{,}2\times 2 + 0{,}1\times 3 + 0{,}1\times 3 = 2{,}2$$

$$\boxed{H(S) \le \bar{n} < H(S)+1}$$

$$H(S) = -0{,}4.\log_2(0{,}4) - 2\times 0{,}2.\log_2(0{,}2) - 2\times 0{,}1.\log_2(0{,}1) = 2{,}12\,\text{bits}$$

The MPEG audio format

Moving Pictures Experts Group (MPEG, created in 1988): a group of experts in information technology who collaborate on the definition of international **ISO** standards for the coding of audio and video contents.

- First psycho-acoustic masking code, by Murray Hill (AT&T, Bell Labs., 1978)

- MPEG-1 audio: lossy compression of audio
- MPEG-1 audio (1992), MPEG-2 audio (1994), MPEG-4, MPEG-7
- MPEG-1 audio, layer I, II, III (aka MP3, 1993)
- AAC (Advanced Audio Coding): included in MPEG-4 (aka, MP4), allows streaming over internet

- Napster (1999): MP3 peer-to-peer sharing

Optimal coding

Objective: Distribute optimally R bits within K channels

This is formally equivalent to minimize the distortion D defined as:

$$D(R_1, \ldots, R_K) =< e^2 >= \frac{1}{K}\sum_{k=1}^{K} < e_k^2 > \tag{46}$$

under the constraint:

$$R = \frac{1}{K}\sum_{k=1}^{K} R_k \tag{47}$$

The quantification error caused by allocating R_k bits to the k-th channel is:

$$< e_k^2 > \approx < x_k^2 > /(3 \times 2^{R_k}) \tag{48}$$

This is strictly equivalent to the fact that the error is reduced by 3 dB for each bit allocated.

Optimal coding

Objective: Distribute optimally R bits within K channels

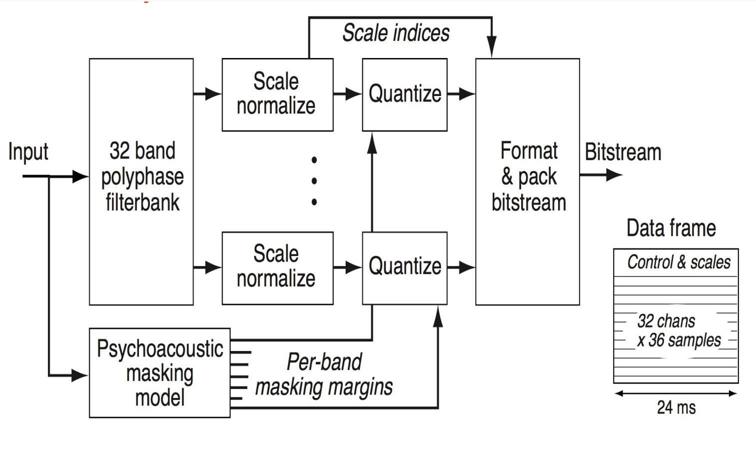
This is an **optimization under constraint** which can be solved by using Lagrange multipliers.

The quantification error caused by allocating R_k bits to the k-th channel is:

$$R_k = R + \frac{1}{2}\log_2 < x_k^2 > - \frac{1}{2}\log_2 \left(\prod_{j=1}^{K} \right)^{1/K} \tag{49}$$

This is equivalent to the fact that the error is reduced by 3 dB for each bit allocated.

MPEG-1, Layer-1 : Architecture



MPEG-1, Layer-1 : Perceptual coding

- Based on psychoacoustics, as long as the quantization noise is lower than the masking threshold at any frequency, the noise will be inaudible.

- In order to make the reconstruction error least detectable by the human ear, **the NMR in each channel must be minimised**

$$NMR(k) = SMR(k) - SNR(k) \tag{50}$$

where:

NMR is the Noise-to-Mask Ratio (in dB)

SMR is the Signal-to-Mask Ratio (in dB), as computed by the psycho-acoustics model
SNR is the Signal-to-Noise Ratio (in dB), corresponding to the quantification error due the the number of bits allocated

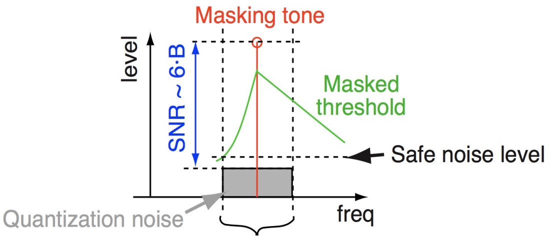
- After bit allocation, the scale factor of the channel is encoded with 3 bits (similar to the gain envelope of the amplitude spectrum)
- Each normalized channel is quantized accordingly to the bit allocation

MPEG-1, Layer-1 : Optimizing the Mask-to-Noise Ratio

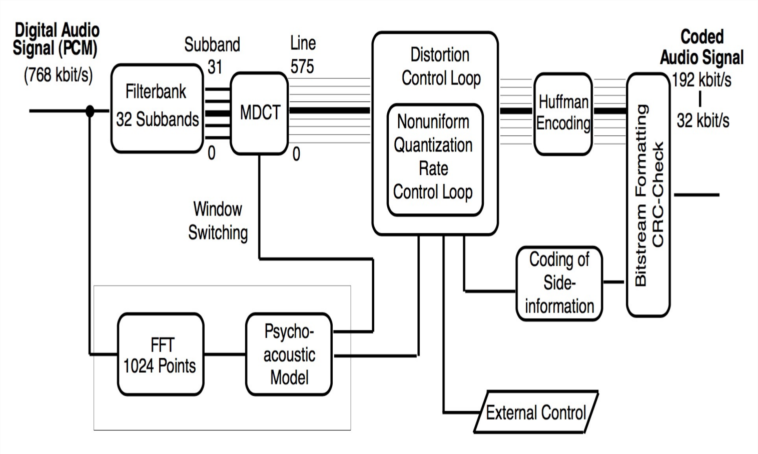
Algorithm 1 *

- **initialization** : 0 bits for each sub-band
- **while** $R > 0$ or $NMR(k) > 0, \forall k$ **do**
  - **Compute NMR** in each channel
  - $\widehat{k} = \arg\max_k (MNR)$
  - Find channel with **higher NMR**
  - Improve that channel by **allocating 1 bit** **(3 dB in SNR reduction)**
  - Update number of remaining bits available R
- **end while**

An iterative algorithm for the bit allocation in MPEG-1, Layer 1



MPEG-1, Layer-3 : Architecture



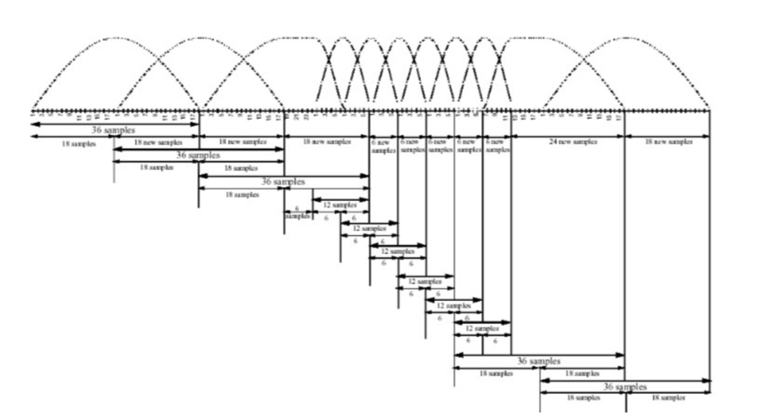
MPEG-1 layer-3

Some of the key advances of the Layer-3 format

- MDCT (Modified Discrete Cosine Transform)
  Transform with real values, 4 times less values than the STFT!
- Variable length Windows
  Short / long windows are used to encode fast / slow time variations. The choice of the window is decided after measuring the entropy of the audio signal.
- Entropic encoding
  An entropic encoding, based on the Huffman algorithm, is used following the perceptual coding, for additional lossless compression
- Bits reservoir
  Bits are not necessarily used uniformingly at each time. Unused bits are stored and can be further used forward

Variable window length

- Short window (12 samples)
  Good time resolution, non-stationary/transients segments
- Long window (36 samples)
  Good frequency resolution, stationary segment



Bits reservoir

- Bit reservoir is used to encode the next time frames