Foundations of Multimedia Technologies

Dr. Firtha Gergely

April 29, 2020

Contents

Chapter 1

Basics of image and video compression

The previous chapter introduced the basic properties of consumer and professional, studio video parameters.

The active spatial resolution and the resulting bit rates of frequently used digital video formats are summarized in Table 1.1.

Table 1.1: The active bitrate of frequently used video formats along with the size, required for storing 1 hour of video stream

Format	**Active resolution**	**Active bitrate 4:2:2**	**Active bitrate 4:2:0**	**Size of 1 hour video**
SIF ($i59.54$)	352×240	40.6 Mbit/s	30.4 Mbit/s	13.7 Gbyte
CIF ($i59.54$)	352×288	48.6 Mbit/s	36.5 Mbit/s	16.4 Gbyte
$576i50$	576×720	199 Mbit/s	149.1 Mbit/s	67.1 Gbyte
$720p60$	1280×720	883 Mbit/s	662.8 Mbit/s	298.3 Gbyte
$1080i30$	1920×1080	994 Mbit/s	745.8 Mbit/s	335.6 Gbyte
$1080p60$	1920×1080	1.99 Gbit/s	1.49 Gbit/s	671.2 Gbyte
$2160p60$ (10 bits)	3840×2160	9.95 Gbit/s	7.47 Gbit/s	3.36 Tbyte
$4320p60$ (10 bits)	7680×4320	39.8 Gbit/s	29.9 Gbit/s	13.44 Tbyte

In the table SIF and CIF abbreviate Source Input Format and Common Intermediate Format respectively. Both formats were introduced for the consumer digital representation of NTSC and PAL videos—with CIF being the default video format of the H.261 encoder and SIF being that for the MPEG-1 standard— with a halved vertical resolution when compared to the professional ITU-601 studio standard.

As the table verifies it, the generated data rate of video formats—and thus the required storage space—grows exponentially with higher spatial and temporal resolution. Modern studio and consumer interfaces—variants of the SDI interface for studio applications and HDMI or DisplayPort for consumer use—allow the transmission of the data rates of uncompressed video over short ranges, e.g. between local devices. However, the storage and broadcasting of such high data rates is virtually impossible: the compression of digital video data is indispensable.

Fortunately, real-life sequence of images contain significant amount of redundant information: Statistically speaking within single frames the neighboring pixels are usually highly correlated. Similarly, consequent frames are usually very similar to each other, even if they contain objects under motion. In video signals, the redundancy can be classified as spatial, temporal, coding and psychovisual redundancies:

- Spatial redundancy (or intraframe/interpixel redundancy) is present in areas of images or video frames where pixel values vary only by small amounts.

- Temporal redundancy (or interframe redundancy) is present in video signals when there is significant similarity between successive video frames.

- Coding Redundancy is present if the symbols produced by the video encoder are inefficiently mapped to a binary bitstream. Typically, entropy coding techniques can be used in order to exploit the statistics of the output video data where some symbols occur with greater probability than others.

- Psychovisual redundancy is present either in a video signal or a still image containing perceptually unimportant information: The eye and the brain do not respond to all visual information with same sensitivity, some information is neglected during the processing by the brain. Elimination of this information does not affect the interpretation of the image by the brain and may lead to a significant compression. Psychovisual redundancy is usually removed by appropriate requantization of the video data, so that the quantization noise remains under the threshold of visibility.

In order to achieve a high compression ratio, all the above redundancy types should be eliminated, being the basic goal of a **source encoder**.

Generally speaking, the aim of source encoding is reducing the source redundancy by keeping only the relevant information, based on the properties of the source and the sink. The source in this case is the video (or possibly audio) sequence, and the sink is the human visual system (or the auditory system for audio info). The general structure of a source encoder, valid both for video or audio inputs is depicted in Figure 1.1. The reduction of the different types of redundancy is performed by the following steps:

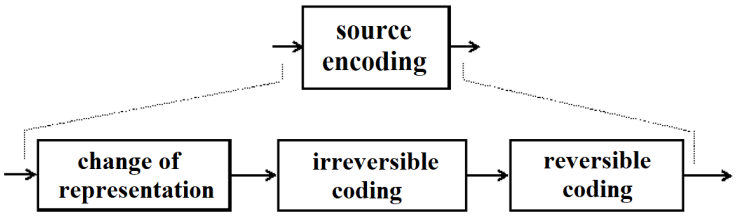


Figure 1.1: Block scheme of a general video/audio source encoder.

- Change of representation: in order to reduce spatial and temporal the input data is represented in a new data space containing less redundancy. The change of representation can be performed by
  - Differential coding (DPCM: Differential Pulse Code Modulation)
  - Transformation coding
  - Sub-band coding
- Irreversible coding: the accuracy of representation is reduced by removing irrelevant information, hence, eliminating psychovisual redundancy. Irreversible coding is achieved by
  - requantization of the data
  - spatial and temporal subsampling
- Reversible coding: an efficient code-assignment is established reducing statistical redundancy. Types of reversible entropy coding applied often in video, image and audio processing are
  - Variable Length Coding (VLC)
  - Run-Length Coding (RLC)

In the following this chapter introduces the basic concepts of compression methods, based on differential coding and transformation coding. The basic concepts are introduced for the generalized case of arbitrary one and two dimensional input signals, and later specialized to video signal inputs.

1.1 Differential quantization

Differential quantization is a compression technique, utilizing linear prediction along with the requantization of the predicted data (i.e. performing both a change of representation and irreversible coding): instead of the direct quantization and transmission of the input signal, the actual input sample is predicted with an appropriately chosen prediction algorithm, and only the discrepancy between the actual and the estimated sample is further processed. In the receiver the same prediction is performed as in the

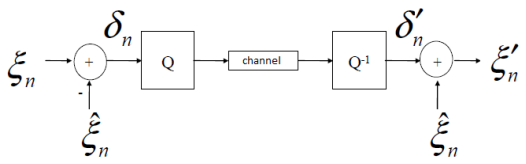


Figure 1.2: Block scheme of a general differential encoder and decoder.

source side, and the output sample is obtained as the sum of the estimated signal and the error of estimation.

The signal processing steps in a differential encoder and decoder are shown in Figure 1.2 with the following notation:

- $\xi(n)$ is the input source sample
- $\hat{\xi}(n)$ is the predicted input sample
- $\delta(n)$ is the error of prediction/differential signal
- Q is the quantization of the signal
- Q^{-1} is the inverse quantization
- $\delta'(n)$ is the quantized differential signal
- $\xi'(n)$ is the quantized, reconstructed input sample

In the block diagram quantization is performed by rescaling the input signal to match the dynamic range of the quantizer, followed by the rounding of the signal level to the nearest integer. Inverse quantizer, on the other hand scales back the quantized signal to the original dynamic range (obviously, information loss can not be reversed).

The basic idea behind differential quantization is the following: Assuming an efficient prediction the dynamic range of the differential signal is significantly smaller than that of the original input signal. Therefore, discretizing the error signal means the division of a smaller dynamic range to the same number of intervals (2^N in case of N bits representation) than in case of quantizing the input signal directly, resulting in an increased resolution, or mathematically speaking, in an increased signal-to-noise ratio. Alternatively, the same signal-to-noise ratio may be achieved by using lower bit depths utilizing differential quantization.

In order to give a mathematical description on differential quantization and quantify the introduced quantities, first a brief summary of stochastic processes is given.

1.1.1 Basic stochastic concepts

A stochastic process is any process describing the evolution in time or space of a random phenomenon, given by an indexed sequence of random samples. Each sample is a random variable with a given probability distribution, and with the probability usually depending on the previous samples. For the sake of simplicity it is implied here that the

process evolves over time, but all the following can be easily extended for e.g. spatially dependent processes.

Let ξ denote a stochastic process, and the sample index denoted by n, hence for each index $\xi(n)$ is a random variable. A stochastic process is fully described by its joint distribution function, which is, however, rarely available either by measurement or analytically. Instead, more often stochastic processes are characterized in a simplified manner by their **moments** (being the **mean value** its first and the **variance** its second moment) and the **autocorrelation function**.

Wide-sense stationary processes: In the following only **stationary processes** are investigated, that's statistical properties do not change over time. Strict stationary requires the entire joint distribution function of the process to be time invariant. In most applications it is sufficient to require the process to be **wide-sense stationary (WSS)**, defined by the following properties:

- The mean/expected value of a WSS process is constant, invariant of n:

$$m_\xi(n) = m_\xi \tag{1.1}$$

  Once the above relation holds, the expected values of the process can be approximated as the average of a realization of length N according to

$$m_\xi = \mathbb{E}(\xi(n)) = \frac{1}{N}\sum_{n=1}^{N}\xi(n) \tag{1.2}$$

- For a general process the autocorrelation function can be defined for two distinct samples, i.e. it is a two-dimensional function

$$r_\xi(n_1, n_2) = \mathbb{E}(\xi(n_1)\cdot\xi(n_2)), \tag{1.3}$$

  loosely speaking measuring the linear dependence between samples $\xi(n_1)$ and $\xi(n_2)$. If two samples are uncorrelated—i.e. $r_\xi(n_1, n_2) = 0$—it implies that no linear relation exists between them, however, higher order dependence may be present. Therefore, uncorrelatedness does not imply independence (while independence strictly ensures uncorrelatednes).

  For a WSS process this linear dependence is translation invariant

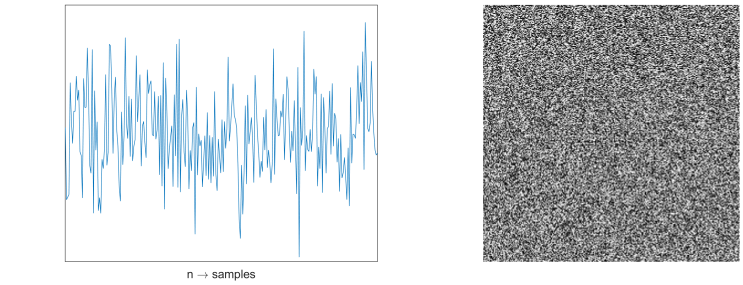
$$r_\xi(n_1, n_2) = r_\xi(n_1 + d, n_2 + d), \qquad \forall d \in \mathcal{N} \tag{1.4}$$

  therefore autocorrelation depends only on the distance of the two samples (denoted now by d)

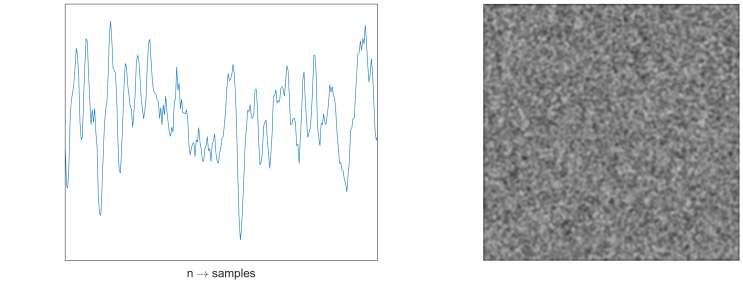
$$r_\xi(n_1 - n_2) = r_\xi(d). \tag{1.5}$$

  If the above relation holds, autocorrelation can be statistically approximated from a realization of the process as

$$r_\xi(d) = \mathbb{E}(\xi(n)\cdot\tilde{\xi}(n+d)) = \frac{1}{N}\sum_{n=0}^{N}\xi(n)\xi(n+d) \tag{1.6}$$



(a)



(b)

Figure 1.3: One dimensional and two dimensional white noise process (a) and correlated noise process (b).

- As a further property for WSS process the auto-correlation function at zero lag ($d = 0$) gives the mean value of the squared samples, i.e. the mean energy of the process, being obviously also time invariant

$$r_\xi(0) = E_\xi = \mathbb{E}(\xi(n)^2) = \frac{1}{N}\sum_{n=0}^{N}\xi(n)^2. \tag{1.7}$$

Noise processes: As the most simple stochastic example, an uncorrelated random process is considered, meaning that linear relation exists between neighboring samples. For such a process the autocorrelation is zero valued everywhere, except for zero lag ($d = 0$), where the autocorrelation value is the energy of the random process. The autocorrelation, therefore, is a Kronecker delta (discrete Dirac delta) function at the origin, given by

$$r_\xi(n) = E_\xi \cdot \delta(n) = \begin{cases} 0, & \text{if } n = 0 \\ E_\xi, & \text{elsewhere.} \end{cases} \tag{1.8}$$

Such a stochastic process is called **white noise**. The distribution of the individual samples is arbitrary, most often the samples are drawn from uniform or Gaussian normal distribution.

The terminology originates from the **power spectral density**, defined as the Fourier transform of the autocorrelation function , describing the frequency content of the stochastic process. For white noise the spectral density function is constant, similarly to the spectrum of white light containing all lights with all the visible wavelengths equally. A simple example realization of white noise process is depicted in Figure 1.3 (a) in one and two dimensions.

A correlated process can be most easily generated from white noise by linear filtering (e.g. FIR filtering): since after filtering each output sample is produced as the linear combination of the previous samples, therefore neighboring samples become correlated, and the autocorrelation is described by the filtering coefficients themselves. Correlated noise, obtained by filtering of the exemplary white noise realization is depicted in Figure 1.3.

1.1.2 The goal of differential quantization

Having introduced basic stochastic concepts differential quantization can be discussed mathematically.

In the model applied the input signal $\xi(n)$ is assumed to be a wide sense stationary process. The effect of quantization can be most easily modeled as an additive noise $\epsilon(n)$, added to the quantized signal. Efficiency of quantization is usually described by the signal-to-quantization-noise ratio, defined as the ratio of the energy of the quantized signal and the quantization noise, written as

$$\mathrm{SQNR} = \frac{\mathbb{E}(\xi(n)^2)}{\mathbb{E}(\epsilon(n)^2)}, \tag{1.9}$$

assuming that the quantized signal is the input signal directly.

In an ideal case where the quantization error is uniformly distributed and the signal has a uniform distribution covering all quantization levels the quantization noise can be calculated as

$$\mathrm{SQNR} = 20\log_{10}2^N, \tag{1.10}$$

where N is the bit depth. In case that differential quantization is applied, two statements can be made

- Assuming that in the receiver side the input signal can be regenerated from the quantized differential signal the final signal-to-noise ratio can be calculated as

$$\mathrm{SNR} = \frac{\mathbb{E}(\tilde{\xi}(n)^2)}{\mathbb{E}(\epsilon(n)^2)}, \tag{1.11}$$

- However, instead of the input signal, the differential signal is quantized, setting the quantization SNR to

$$\mathrm{SQNR} = \frac{\mathbb{E}(\delta(n)^2)}{\mathbb{E}(\epsilon(n)^2)} = 20\log_{10}2^N. \tag{1.12}$$

Rewriting the above equations results in the total SNR of

$$\mathrm{SNR} = \frac{\mathbb{E}(\tilde{\xi}(n)^2)}{\mathbb{E}(\epsilon(n)^2)} = \frac{\mathbb{E}(\tilde{\xi}(n)^2)}{\mathbb{E}(\delta(n)^2)} \cdot \underbrace{\frac{\mathbb{E}(\delta(n)^2)}{\mathbb{E}(\epsilon(n)^2)}}_{20\log_{10}2^N}, \tag{1.13}$$

revealing that compared to the direct quantization of the input signal the signal-to-noise ratio is increased by a factor of

$$\mathrm{G}_p = \frac{\mathbb{E}(\tilde{\xi}(n)^2)}{\mathbb{E}(\delta(n)^2)} \tag{1.14}$$

termed as the **prediction gain**, being a large number, assuming that the input signal can be predicted precisely. This arises the question, how the actual input sample can be estimated based on the previous samples only.

1.1.3 The optimal prediction coefficients

As the most simple approach the actual input sample $\xi(n)$ can be predicted as the linear combination of the previous N number of samples, written in the form of

$$\tilde{\xi}(n) = \sum_{m=1}^{N} p(m)\xi(n-m) = \mathbf{p}^{\mathrm{T}}\xi_{n-1}, \tag{1.15}$$

written in a vectorial form. In the expression vector $\mathbf{p} = [p(1), p(2), ..., p(N)]^{\mathrm{T}}$ contains the weights of the previous input samples used for prediction, and vector $\xi_{n-1} = [\xi(n-1), \xi(n-2), ..., \xi(n-N)]^{\mathrm{T}}$ contains the previous N number of the input samples.

The goal is to minimize the expected energy of the difference between the actual input sample $\xi(n)$ and the prediction $\hat{\xi}(n)$ by optimizing the prediction weights $\mathbf{p}^{\mathrm{T}}$ so that

$$\arg\min_{\mathbf{p}} : \mathbb{E}\left(|\xi(n) - \tilde{\xi}(n)|^2\right) = \arg\min_{\mathbf{p}} : \mathbb{E}\left(|\xi(n) - \mathbf{p}^{\mathrm{T}}\xi_{n-1}|^2\right) \tag{1.16}$$

holds. The quadratic expression can be expounded to

$$\begin{aligned}\mathbb{E}\left(|\xi(n)^2 - \mathbf{p}^{\mathrm{T}}\xi_{n-1}|^2\right) = \mathbb{E}\left(\xi(n) - 2\xi(n)\mathbf{p}^{\mathrm{T}}\xi_{n-1} + \mathbf{p}^{\mathrm{T}}\xi_{n-1}\xi_{n-1}^{\mathrm{T}}\mathbf{p}\right) = \\ \mathbb{E}\left(\xi(n)^2\right) - 2\mathbf{p}^{\mathrm{T}}\,\mathbb{E}\left(\xi(n)\xi_{n-1}\right) + \mathbf{p}^{\mathrm{T}}\,\mathbb{E}\left(\xi_{n-1}\xi_{n-1}^{\mathrm{T}}\right)\mathbf{p}\end{aligned} \tag{1.17}$$

with exploiting the linearity of expected value operator and collecting non-stochastic quantities outside of it. The expected value of the scalar-vector product and the dyadic product terms of the expression can be recognized as the autocorrelation values of the input signals, rewritten in a matrix form as

$$\mathbb{E}\left(|\xi(n)^2 - \mathbf{p}^{\mathrm{T}}\xi_{n-1}|^2\right) = r_\xi(0) - 2\mathbf{p}^{\mathrm{T}}\mathbf{r}_\xi + \mathbf{p}^{\mathrm{T}}\mathbf{R}_\xi\mathbf{p} \tag{1.18}$$

with denoting the signal energy, and the autocorrelation vector and matrix as

$$r_\xi(0) = \mathbb{E}\left(\xi(n)^2\right), \qquad \mathbf{r}_\xi = \begin{bmatrix} r_\xi(1) \\ r_\xi(2) \\ \dots \\ r_\xi(N) \end{bmatrix},$$
$$\mathbf{R}_\xi = \begin{bmatrix} r_\xi(0) & r_\xi(1) & \dots & r_\xi(N-1) \\ r_\xi(1) & r_\xi(0) & \dots & r_\xi(N-2) \\ \dots & & & \\ r_\xi(N-1) & r_\xi(N-2) & \dots & r_\xi(0) \end{bmatrix}. \tag{1.19}$$

Expression 1.18 has to be minimized with respect to vector $\mathbf{p}^{\mathrm{T}}$. The minimization can be performed by finding the zero of the derivative of the expression with respect to vector $\mathbf{p}^{\mathrm{T}}$, reading

$$\frac{\partial}{\partial \mathbf{p}^{\mathrm{T}}} \mathbb{E}\left(|\xi(n)^2 - \mathbf{p}^{\mathrm{T}}\xi_{n-1}|^2\right) = -2\mathbf{r}_\xi + 2\mathbf{R}_\xi\mathbf{p} = 0. \tag{1.20}$$

Finally, from the above equation the optimal prediction coefficient vector can be expressed as

$$\mathbf{p}^{\mathrm{T}} = \mathbf{R}_\xi^{-1}\mathbf{r}_\xi. \tag{1.21}$$

The above coefficients are the so-called **Wiener filter** coefficients for the estimation of a stationary stochastic process.

From the form of the optimal prediction coefficients it is clear that the signal estimation is based on the measured correlation of the previous source samples, therefore prediction is efficient as long as neighboring samples are linearly related. Hence the optimal prediction is often termed as **linear prediction**. The above Wiener filter is, therefore, capable of the estimation of the correlated part of the input signal.

1.1.4 Prediction as FIR filtering

It should be noted that linear prediction (1.15) describes the discrete linear convolution of vectors $\mathbf{p}$ and ξ_{n-1}. This means that the estimation of the actual sample can be obtained by the simple FIR filtering[1] of the input stream with the coefficient vector $\mathbf{p}$. The result of estimation is subtracted from the input sample, generating the differential signal, which, therefore, can be written as

$$\delta(n) = \xi(n) - \sum_{m=1}^{N} p(m)\xi(n-m), \tag{1.22}$$

[1]The term FIR (Finite Impulse Response) filtering refers to the fact that the applied filter contains no feedback, thus it is ensured that to an excitation with finite extent the filter output is of finite extent. The actual filter impulse response is described by the coefficient vector itself.

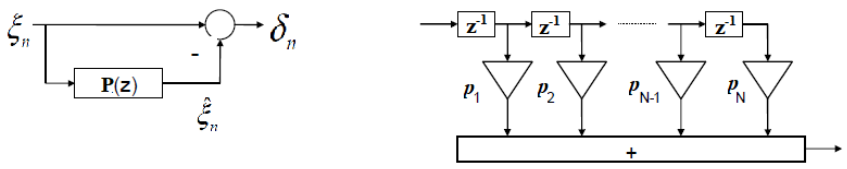


Figure 1.4: One dimensional and two dimensional white noise process (a) and correlated noise process (b).

or, transforming the equation to the z-transform domain—by exploiting that delay by one sample is a multiplication by z^{-1} in the z-domain—as

$$\delta(z) = \xi(z)\left(1 - \sum_{m=1}^{N} p_m(z)\, z^{-m}\right) = \xi(z)\,(1 - P(z))\,. \tag{1.23}$$

The realization of linear prediction with FIR is depicted in Figure 1.4. The structure of FIR filtering is depicted in Figure 1.4 (b). The prediction filter can be also interpreted as an accumulator, or memory, containing the previous samples, added with different weights to the output.

It is important to note that the prediction filter $P(z)$ is capable of identifying linear tendencies in the input signal and the actual sample is estimated based on the assumed linear relationship between previous samples Once all the correlated part of the input signal is removed, by definition, in the remaining differential signal each sample is uncorrelated from the previous samples. Thus, with theoretical optimal prediction, filter $(1 - P(z))$ **decorrelates** the input signal, and the differential signal is a **white noise process**: The filter is often referred to as whitening filter, since in the optimal case it transforms the input signal into white noise.

Figure 1.5 illustrates the whitening process in case of a simple 1D input signal, in the present example being an audio stream. The correlation of the signal can be estimated based on the input stream according to (1.6) from which the linear prediction coefficients are obtained (from (1.21)). These coefficients are applied to perform whitening filtering process described by 1.23.

Figure 1.5 (a) compares the time histories of the input and the differential signals. As it is illustrated the dynamics range of the input signal is significantly reduced by subtracting the predicted signal. Generally speaking, periodic signals can be predicted efficiently by measuring correlation, therefore, harmonic signals are almost entirely removed from the input. Obviously, transients can not be predicted based on previous samples, therefore, they are still present in the differential signals.

Figure 1.5 (b) verifies that as the effect of filtering the autocorrelation of the differential signal became approximately a delta function, hence the output stream is nearly uncorrelated. This is also verified by comparing the input and output spectral density functions, with the output spectrum being nearly constant as depicted in Figure 1.5 (c).

Note that since the input signal is typically of low-pass filtered characteristics,

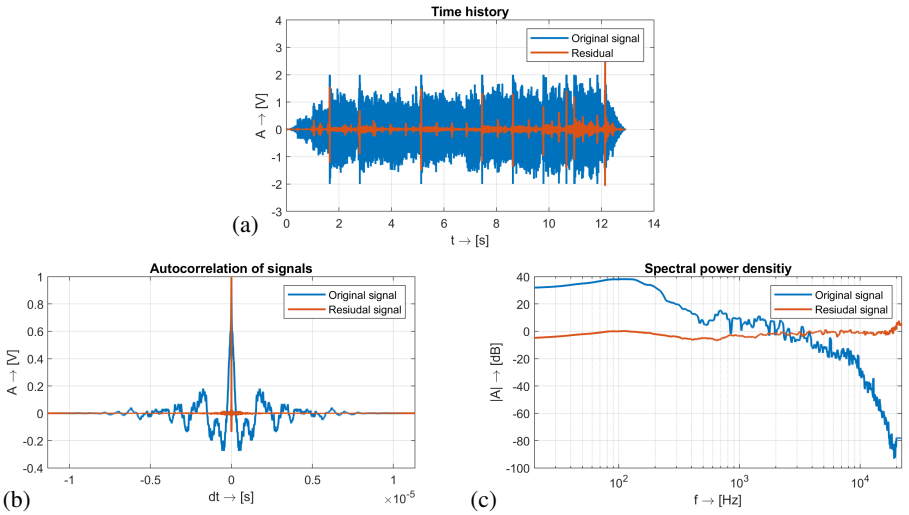


Figure 1.5: Example for the optimal linear prediction of an audio input stream.

therefore, the whitening filter has to be of high-pass characteristics in order to flatten out the output spectrum. This statement can be generalized to typical audio and video signals: since natural signals are usually dominated by low-frequency content, therefore, in practical applications the differential signal (i.e. prediction) is obtained as the high-pass filtered version of the input signal.

It has been highlighted that once the autocorrelation of the input signal can be estimated (e.g. by measurement) nearly optimal prediction coefficients can be defined. Hence, the prediction filter $P(z)$ depends on the actual input signal. Obviously, the decoding of the differential signal also requires the knowledge of coefficients $P(z)$ in the decoder side. In order to avoid the transmission of the prediction filter coefficients, in practice as a sub-optimal solution fixed prediction coefficients are applied for high-pass filtering. In the following only this fix-coefficient approach is considered.

1.1.5 Problem of feedforward prediction

Although being a very simple approach, the direct FIR filtering scheme, presented in Figure 1.4 (a) is never used directly in practice, due to the following reason:

So far only the prediction and generation of the differential signal have been discussed in details, without taking the receiver side into consideration. In the receiver—with assuming that the prediction filter coefficients are known—the original samples are reconstructed by adding the previously accumulated, weighted samples to the residual signal. With denoting the decoded signal by $\tilde{\xi}$ it can be written as

$$\tilde{\xi}(z) = \delta'(z) + \sum_{m=1}^{N} p_m(z)\tilde{\xi}(z)z^{-m} \quad \rightarrow \quad \frac{\tilde{\xi}(z)}{\tilde{\delta}(z)} = \frac{1}{1-P(z)}, \tag{1.24}$$

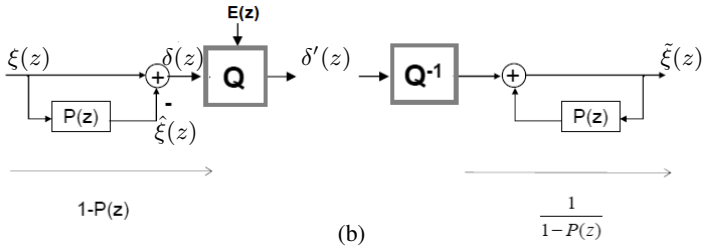


Figure 1.6: Example for the optimal linear prediction of an audio input stream.

hence, in the receiver the original signal can be reconstructed by **inverse filtering** and the transfer function of the inverse filter is the reciprocal of the forward filter. Obviously, $\frac{1}{1-P(z)}$ describes an IIR (Infinite Impulse Response) filter, since the reconstruction in the receiver is performed by feedbacking the output signal to the input of the receiver. Hence, the zeros of the forward FIR filter are mapped to the poles of the inverse filter, leading to stability problems in the receiver.

These stability problems causes serious artifacts due to the presence of the quantizer: As discussed earlier the presence of quantizer can be modeled as introducing quantization noise, added directly to the differential signal. Hence, with denoting the z-transform of the additive quantization noise by $\epsilon(z)$ and taking both the encoding and decoding steps into consideration the decoded signal can be written as

$$\tilde{\xi}(z) = \left(\underbrace{\xi(z)\,(1-P(z))}_{\delta(z)} + \epsilon(z) \right) \cdot \frac{1}{1-P(z)} = \xi(z) + \epsilon(z)\frac{1}{1-P(z)}. \qquad (1.25)$$

The decoded signal, therefore, consists of the original input signal and the quantization noise, filtered with the potentially unstable inverse filter.

In order to highlight the resulting effect as a simple example DPCM encoding is investigated.

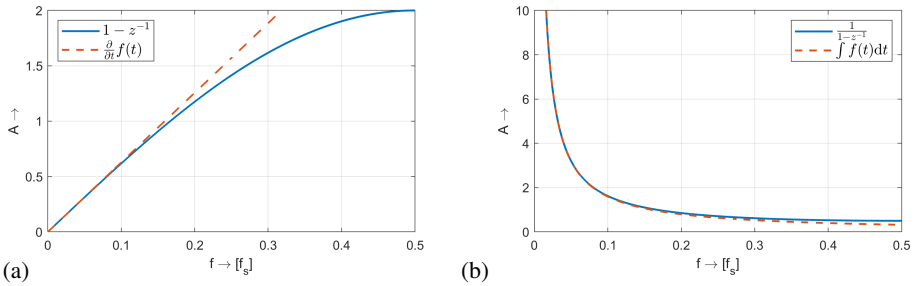
DPCM coding with feedforward prediction:
As the simplest approach for differential coding the basis of prediction is simply the previous sample of the input signal, hence

$$\begin{aligned} \delta(n) &= \xi(n) - \xi(n-1) \\ P(z) &= z^{-1} \quad \rightarrow \quad 1 - P(z) = 1 - z^{-1}. \end{aligned} \qquad (1.26)$$

The differential sample is simply given as the difference of the actual and the previous input samples. The approach is termed as **Differential Pulse-Code Modulation**.

Generally speaking, the derivative of the a continuous function most simply can be approximated numerically by the finite difference

$$\frac{\partial}{\partial t} f(t) \approx \frac{f(t) - f(t-T)}{T} = \frac{1}{T}\left(f(n) - f(n-1)\right), \qquad (1.27)$$



(a) (b)

Figure 1.7: Example for the optimal linear prediction of an audio input stream.

termed as the backward Euler scheme, with T being the sampling interval. Comparison of this expression with (1.26) clearly reveals that differential coding realizes the approximate differentiation of the input signal prior to quantization. The frequency response of this simple differentiation is obtained analytically by evaluation the z-transform along the unit circle, i.e. by the substitution $z = \mathrm{e}^{\mathrm{j}2\pi f/f_s}$, with f_s being the sampling frequency

$$|1 - z^{-1}| = |1 - \mathrm{e}^{-\mathrm{j}2\pi f/f_s}| = 2\left|\sin\frac{\pi f}{f_s}\right|. \tag{1.28}$$

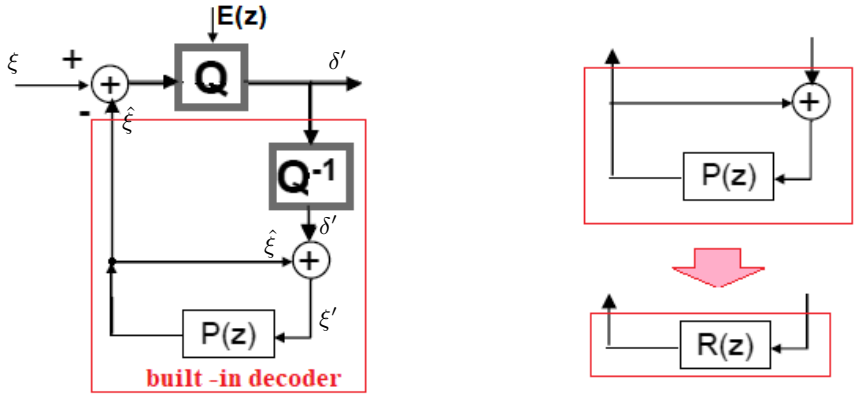
The frequency response of the filter is shown in Figure 1.7 (a) along with the frequency response of an ideal differentiator. The filter has a zero at $z = 1$, at zero frequency (as the derivative of a constant signal is zero), and approximates an ideal differentiator perfectly in the low frequency region.

Obviously, the inverse operation of differentiation is integration, i.e. the inverse filter $\frac{1}{1-P(z)}$ describes the numerical approximation of integration. This is verified by Figure 1.7 (b), depicting the frequency response of the inverse filter and that of an ideal integrator, with a pole at zero frequency (the integral of constant signal tends to infinity).

As a conclusion, the drawback of the simple feedforward processing scheme—where the encoder performs prediction directly from the input signal—is the following: According to (1.25) the quantization noise is reproduced in the output of the decoder filtered with the inverse filter $\frac{1}{1-z^{-1}}$. Since this filter response describes the integration of the filtered signal, therefore, the quantization noise is integrated, **accumulated** in the decoder. Less formally speaking: the source of accumulation of quantization error is the fact that due to the presence of quantization the basis of prediction is different in the encoder and the decoder side. While the encoder predicts the next sample from the original signal, in the decoder side only the quantized, decoded previous samples are available for the next prediction.

1.1.6 Feedback prediction loop

In order to overcome the error of feedforward prediction and to avoid the accumulation of the quantization error it has to be ensured that the encoder and the decoder uses



(a) (b)

Figure 1.8: One dimensional and two dimensional white noise process (a) and correlated noise process (b).

the same past samples for prediction. This can be achieved only by ensuring that the encoder predicts from the quantized input samples as well. Hence, in the encoder side the quantized differential signal has to be decoded, which decoded signal will serve as the basis for the next prediction: the decoder has to be built in the encoder in a feedback loop.

The concept of differential quantizer with built-in decoder stage is depicted in Figure 1.8 (a): the accumulator/filter $P(z)$ contains the previous quantized samples (in the previous example only the previous, quantized sample) and produces the actual estimation $\hat{\xi}'$ as their weighted sum. In the actual time step the estimation is subtracted from the actual input signal, and the difference is quantized, producing the quantized differential output signal. Besides transmission to the receiver, the quantized differential signal is added to the estimated sample, producing the decoded, quantized signal ξ', which is pushed into the accumulator and will serve as the basis of the prediction in the next time step. Hence, it is ensured that the basis of prediction is the quantized signal, similarly to the receiver side.

Mathematically the entire built in decoder can be modeled as a single transfer function: The input of the decoder block is the quantized differential signal δ', while the output is the estimation $\hat{\xi}$, i.e.

$$R(z) = \frac{\hat{\xi}(z)}{\delta'(z)} = \frac{P(z)\left(\hat{\xi}'(z) + \delta'(z)\right)}{\delta'(z)} \quad \rightarrow \quad R(z) = \frac{P(z)}{1 - P(z)} \tag{1.29}$$

holds. The transmitted differential signal can be written with taking the quantization error into consideration as

$$\begin{aligned} \delta'(z) &= \xi(z) - R(z)\delta'(z) + \epsilon(z) \quad \rightarrow \quad (1 + R(z))\,\delta'(z) = \xi(z) + \epsilon(z) \\ \delta'(z) &= \frac{1}{1 - R(z)}\left(\xi(z) + \epsilon(z)\right) = (1 - P(z))\left(\xi(z) + \epsilon(z)\right) \end{aligned} \tag{1.30}$$

and finally after decoding by applying the unchanged decoder with the transfer function of $\frac{1}{1-P(z)}$ the decoded signal reads

$$\tilde{\xi}(z) = \frac{1}{1-P(z)}(1-P(z))\left(\xi(z)+\epsilon(z)\right) = \xi(z)+\epsilon(z). \tag{1.31}$$

Therefore, without the accumulation or the error, besides the original input the decoded signal contains only the actual quantization noise. Due to this reason in differential coding exclusively the above feedback structure is applied with the encoder containing the built-in decoder stage.

1.1.7 Application of predictive coding in video technologies

Predictive coding is extensively applied in the field of image and video compression. Prediction can be applied either between pixels or blocks of the image, performing spatial prediction, or between consequent images, applying temporal prediction:

- **Temporal prediction:** MPEG video encoders apply block based temporal DPCM prediction, meaning that each block of a frame is predicted from a block from the previous frame. Hence, the prediction filter/accumulator contains a single block of the previous image, which is subtracted from the actual input block. Newer compression methods, like H.264, allows prediction using multiple reference blocks with arbitrary weights, therefore, the length of the prediction filter may be larger than one, approximating the optimal prediction case, discussed in the foregoing. The block in the prediction filter, termed as the **reference block**, does not necessarily located in the same position as the actual input block, but its location is found where the the reference block resembles the most to the actual input block. Finding the position of the reference block is the basic task of **motion estimation.** Motion estimation and temporal prediction of the MPEG encoder is discussed in details in the following chapter.

- **Spatial prediction:** Alternatively, within an image each pixel may be predicted based on the neighboring pixel values, followed by requantization of the difference pixel. Assuming that pixel values change slowly over the image, storing only the difference from the previous pixels may result in significant data compression. Pixel based predictive coding is applied by the PNG image encoder as discussed in the following, while modern video compression methods, e.g. H.264 allows more sophisticated block based intra-image prediction.

  Furthermore, all JPEG and MPEG block-based video encoders stores the average luminance/color of the actual block predictivly compared to the previous block in a DPCM loop.

  As a rule of thumb, since in the decoder side only the requantized difference pixels/blocks are available, in order to avoid the accumulation quantization error predictive coding is in all cases performed within a feedback prediction loop.

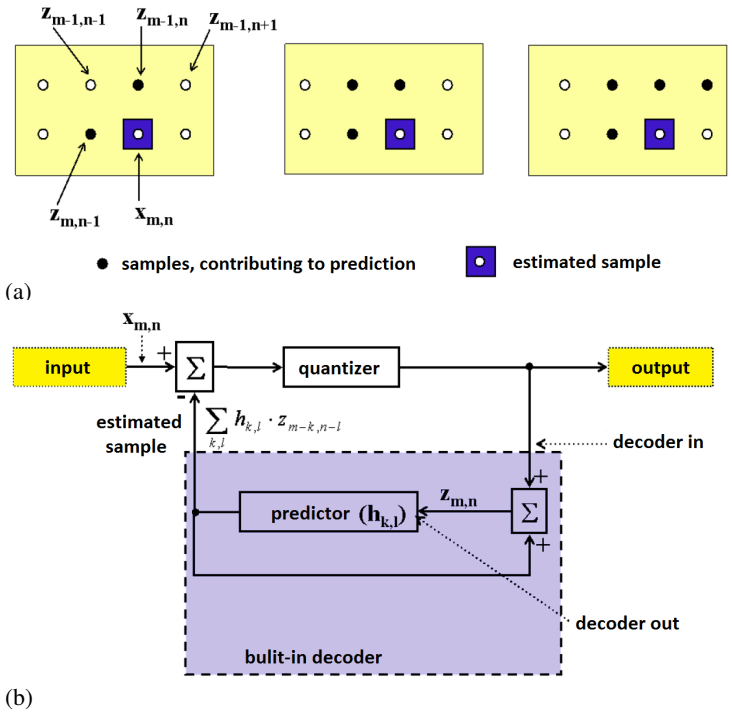


Figure 1.9: Prediction schemes (a) and block diagram (b) of PNG encoder.

PNG encoding:
As a simple example for spatial prediction the PNG encoder is discussed briefly. PNG (Portable Network Graphics), being one of the simplest image compression methods apply a lossless spatial prediction along with lossless entropy coding, with the following properties:

- Since PNG is a lossless compression, therefore, no quantization is present in the encoder.
- As PNG is lossless, luma/chroma separation is not beneficiary, but direct RGB coordinates are encoded.
- Compression is achieved by the fact that the result of spatial prediction contains mainly small pixel intensities, clustered around 0, which can be efficiently entropy coded with a properly chosen variable-length coding. The entropy coder of PNG is termed as DEFLATE coding, being originally introduced for the early version of ZIP comression.
- Before DEFLATE compression the actual pixel, denoted in Figure 1.9 by $x_{m,n}$ is predicted by the neighboring pixels. Obiviously, only previously encoded and transmitted pixels can be the reference of prediction, denoted by $z_{m,n}$. The number of reference pixels can be chosen by the encoder from the prediction

schemes, shown in Figure 1.9. Therefore, in this case linear prediction is based on a 2D FIR filter, containing 2, 3 or 4 values in the prediction accumulator, denoted by $h_{k,l}$.