# Stellingen

behorende bij het proefschrift

# Overlapped Transform Coding of Images: Theory, Application and Realization

van

Richard Heusdens

# I

Het binnen de spraakcodering gebruikte sinusoïdale signaalmodel om een signaal $s$ (per frame) te representeren als

$$s(n) = \sum_{k=1}^{d} \alpha_k e^{i(\omega_k n + \varphi_k)},$$

waar $\alpha_k, \omega_k, \varphi_k \in \mathbb{R}$, leidt niet tot minimale representaties van audiosignalen.

# II

Het is geen goede zaak dat het basiswerk *Rate Distortion Theory; A Mathematical Basis for Data Compression* van Toby Berger [1] niet meer wordt uitgegeven.

[1] T. Berger. *Rate Distortion Theory; A Mathematical Basis for Data Compression.* Prentice Hall Series in Information and System Science. Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1971.

# III

De in [1] beschreven iteratieve *structured total least squares* (STLS) methode voor het oplossen van een overgedetermineerd systeem $Ax \approx b$ waarbij de datamatrix $[A \mid b]$ Toeplitz is, is moeilijk bruikbaar in real-time applicaties.

[1] S. van Huffel, H. Park, and J.B. Rosen. Formulation and solution of structured total least norm problems for parameter estimation. *IEEE Trans. on Signal Processing*, 44(10):2464–2474, October 1996.

# IV

Het gebruik van tijd-variërende filterbanken in datacompressiesystemen kan voor hoge bitrates leiden tot een significante kwaliteitsverbetering in vergelijking met systemen die gebaseerd zijn op tijd-invariante filterbanken.

# V

Recentelijk zijn de Rényi entropieën, die voor genormaliseerde, reëelwaardige tijd-frequentie ditributies van een signaal $s$ gedefiniëerd zijn als

$$H_\alpha(C_s) = \frac{1}{1-\alpha}^{2} \log \left( \iint C_s^\alpha(t, f)\, dt\, df \right), \tag{1}$$

geïntroduceerd als een maat voor signaalinformatie en complexiteit [1, 2]. Hoewel voor oneven $\alpha \geq 3$, alle $s \in L^2(\mathbb{R})$ en alle $C_s$ uit de Cohen-klasse van bilineaire tijd-frequentiedistributies de Rényi entropieën asymptotisch invariant zijn voor kruiscomponenten, is deze informatiemaat praktisch moeilijk toepasbaar.

[1] P. Flandrin, R.G. Baraniuk, and O. Michel. Time-frequency complexity and information. In *Proceedings IEEE Int. Conf. on ASSP*, pages 329–332, 1994.

[2] R.G. Baraniuk, P. Flandrin, and O. Michel. Measuring time-frequency information and complexity using the Rényi entropies. In *Proceedings IEEE Int. Symp. on Information Theory*, pages 426–430, Whistler, Canada, September 1995.

## VI

De codeer-efficiëntie van de discrete-wavelettransformatie in datacompressiesystemen wordt overschat.

## VII

Stellingen in de vorm van resultaten beschreven in het proefschrift kunnen beter achterwege gelaten worden.

## VIII

De reisplanning van de Nederlandse Spoorwegen vertoont grote analogie met het binnen het internet-netwerk gebruikte *transmission control protocol* (TCP). Beide garanderen een betrouwbare transmissie over een onbetrouwbaar netwerk, maar men weet nooit wanneer een "pakket" wordt afgeleverd en via welke route.

## IX

De afname van het aantal ernstige verkeersongevallen op de A2 sinds de invoering van de radarcontrole is eerder een gevolg van de verhoogde staat van oplettendheid van de verkeersdeelnemer dan van het zich houden aan de maximumsnelheid.

# Overlapped Transform Coding of Images: Theory, Application and Realization

# Overlapped Transform Coding of Images: Theory, Application and Realization

PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus Prof. dr. ir. J. Blaauwendraad,
in het openbaar te verdedigen ten overstaan van een commissie,
door het College van Dekanen aangewezen,
op maandag 17 maart 1997 te 13.30 uur

door

Richard HEUSDENS

elektrotechnisch ingenieur

geboren te Alkmaar

Dit proefschrift is goedgekeurd door de promotor:
Prof. dr. ir. J. Biemond.

Toegevoegd promotor:
Dr. ir. E.F. Deprettere.

Samenstelling promotiecommissie:

Rector Magnificus, voorzitter,
Prof. dr. ir. J. Biemond, Technische Universiteit Delft, promotor,
Dr. ir. E.F. Deprettere, Technische Universiteit Delft, toegevoegd promotor,
Prof. dr. ir. P.M. Dewilde, Technische Universiteit Delft,
Prof. dr. S. Vassiliadis, Technische Universiteit Delft,
Prof. dr. ir. J.B.H. Peek, Katholieke Universiteit Nijmegen,
Prof. dr. E.H.L. Aarts, Technische Universiteit Eindhoven,
Dr. ir. M. Breeuwer, Philips Medical Systems.

The work described in this thesis was carried out at the
Philips Research Laboratories in Eindhoven, The Netherlands,
as part of the Philips Research programme.

CIP-DATA KONINKLIJKE BIBLIOTHEEK, DEN HAAG

Heusdens, R.

Overlapped Transform Coding of Images: Theory, Application and Realization
R. Heusdens. - Eindhoven : Philips Research Laboratories
Thesis Technische Universiteit Delft. - With ref.
ISBN 90-74445-33-0
Subject headings: data compression / video coding / digital signal processing.

# Overlapped Transform Coding of Images: Theory, Application and Realization

*R. Heusdens*

*to my family: Karin,*
*Vianne, Naomi and Devi*

# *Summary*

Over the last few decades, images have increasingly been represented in a digital format. Digital representation of images has a number of important advantages over analogue representation, such as flexibility, robustness and suitability for a variety of signal processing techniques. Representing images digitally also has a disadvantage, namely, that the storage or transmission of these signals requires fast interfaces and a large storage capacity or transmission bandwidth, respectively. One way to reduce the number of bits, and thereby reduce the storage or transmission cost, is to apply *data compression*.

Transform coding is one of the most efficient methods of data compression of correlated signals. The use of a discrete-time signal transform before the actual coding (i.e. before the quantization and the mapping onto binary codewords) can significantly reduce the complexity of the coder. Suitable signal transformation leads to signal representations that are easy to code. This thesis describes the design, application and realization of discrete-time signal transforms for the purpose of data compression, in particular, data compression of images.

In Chapter 2, we show that a discrete-time signal transform can be regarded as a $(K, L, m)$ filter bank, where $K$ denotes the number of filter channels, $L$ the filter length and $m$ the down-sample factor. Placing discrete-time signal transforms into this more general framework thus means that the design of signal transforms can be formulated as finding a suitable filter-bank architecture (choice of $K$, $L$ and $m$), and choosing suitable constituent filter responses. In order to investigate how different choices of the parameters $K$, $L$ and $m$ affect the coding efficiency of the filter bank, we apply rate-distortion theory to transform coding systems.

In Chapter 3, we concentrate on the actual design of filter banks. Here we discuss which factors, besides coding efficiency, influence the choice of an architecture and its constituent filter responses. These factors are, for example, perceptual quality and numerical sensitivity. Based on this discussion we derive a list of desirable constraints which we use to choose a suitable filter-bank architecture. Moreover, we show how to design proper filter responses, i.e., filter responses which satisfy all the stated constraints. We show that the class of *lapped orthogonal transforms* (LOTs) is suitable for the application of image coding.

In Chapter 4, we discuss the application of overlapped transform coding with LOTs to X-ray cardio-angiographic image series. With lossless compression techniques the bit rate of these X-ray image series can be reduced by a factor of about 2.5 – 3.5 [1, 2]. The aim of the method described here, however, is to reach reduction factors in the order of 8 – 16. To achieve this goal we rely on lossy compression techniques, in particular, those that are based on overlapped transform coding with LOTs. Firstly, we discuss digital X-ray angiography. This specific application imposes some additional requirements on the parameter setting of the filter bank. Secondly, we discuss the complete transform coding system in more detail and describe the results obtained with a software implementation of this method. Philips has proposed that this coding scheme be included in the discussion on standardization of lossy data-compression algorithms which is being organized by the ACR-NEMA committee with the support of the National Electrical Manufacturers' Association (NEMA) and the American College of Radiology (ACR). We show that LOTs can effectively decorrelate the X-ray images, which simplifies encoding of the transformed signal. Moreover, we show that LOTs are particularly well suited to adapt the compression to the properties of the human visual system and possible post-processing, such as image enhancement.

We end this thesis with the realization of discrete-time signal transforms in Chapter 5. The design of realizations, in particular, those which are suitable for *very large scale integration* (VLSI) technology, involves choosing a suitable algorithm for computing the signal transformation and mapping this algorithm onto an architecture. We show that, by considering *both* algorithm *and* architecture design simultaneously, we can design numerically robust and implementation efficient realizations of filter banks. Moreover, these realizations can be made fully programmable in the sense that the architecture can be used for arbitrary signal transforms, including discrete cosine transforms, LOTs or discrete wavelet transforms.

# *Contents*

# *List of figures*

# List of tables

# Chapter 1

# Introduction

Over the last few decades, images have increasingly been represented in a digital format. Examples of applications in which digital images are used include the videophone, teleconferencing, digital television and digital radiology. Digital representation of images has a number of important advantages over analogue representation, such as flexibility, robustness and suitability for a variety of signal processing techniques. Representing images digitally also has disadvantages, specifically, the capacity required for their storage and the wide transmission bandwidth. For example, standard-definition television signals of the CCIR 601 format [3] have a bit rate of 166 Mbit/s. The storage or transmission of these signals, therefore, requires fast interfaces and a large storage capacity or transmission bandwidth, respectively. One way to reduce the bit rate, and thereby reduce the storage or transmission cost, is to apply data compression. Data compression can be divided into two categories: lossless and lossy compression. In this thesis we concentrate on lossy compression.

Various lossy data compression methods have been introduced over the last twenty years. A relatively old lossy data compression technique is predictive coding [4]. The compression ratio obtained with predictive coding is rather low and, therefore, has not been considered in our investigations. At present, non-overlapped transform coding is the best-known method. In 1991, a non-overlapped transform coding method was standardized by the Joint Photographic Experts Group (JPEG) of the International Standardization Organization (ISO) [5, 6]. The method was originally intended for compression of still images, but it can also be used to compress image series.

For the coding of video-image series, the Moving Picture Experts Group (MPEG) of the ISO recently standardized two motion-compensated inter-frame

non-overlapped transform-coding methods [7, 8, 9]. The JPEG and MPEG methods are based on compression of non-overlapping blocks of $8 \times 8$ pixels. The disadvantage of these methods is that "blocking artifacts" are introduced at low bit rates. These blocking artifacts are observed as equally-spaced discontinuities in the decoded images. In many applications, like medical imaging, the introduction of blocking artifacts may not be acceptable. For example, in medical imaging the artifacts not only give the images an unpleasant appearance, but in the worst case the presence of severe blocking artifacts may lead to a mis-interpretation of the image by the radiologist or cardiologist.

Several relatively new data compression methods are available which can avoid the occurrence of blocking artifacts and which have the potential to reach high compression ratios. The most promising methods are overlapped transform coding [10, 11, 12], wavelet coding [13, 14, 15] and subband coding [16, 17, 18, 19]. All these data-compression methods are based on transforming the image to some spatial frequency-domain representation, quantizing the transformed signal and mapping the quantized signals onto binary codewords. In fact, overlapped transforms, wavelet transforms, subband filtering as well as non-overlapped transforms can be seen as a special case of the more general framework of *discrete-time signal transforms* or *multi-rate filter banks* [12, 20, 21].

This thesis describes the design, application and realization of discrete-time signal transforms for the purpose of data compression, in particular, data compression of images.

The design of discrete-time signal transforms consists of choosing a suitable filter bank architecture (choice of the number of channels, filter lengths, downsample factor) and choosing suitable constituent filter responses. The choice of these parameters depends largely on the underlying application, which, in our case, is data compression of images. This means that the transform should have a good coding efficiency, be numerically robust and not introduce blocking artifacts.

The design of realizations, in particular, realizations which are suitable for *very large scale integration* (VLSI) technology, consists of choosing a suitable algorithm for computing the signal transformation and mapping this algorithm onto an architecture. We try to find realizations that behave *optimally* in the sense that they exhibit low sensitivity to finite-length register effects and enable efficient VLSI implementation by considering *both* algorithm *and* architecture design simultaneously. We argue that established structures, in particular, butterfly structures (the perfect shuffle architectures), are not naturally the "best possible" structures as is commonly claimed and believed. Butterfly structures are used to compute Fourier-like transforms such as the discrete cosine transform (DCT)

which is used in the JPEG and MPEG methods. These structures have emerged from the fast transforms which require only $\frac{m}{2} \log m$ multiplications instead of $m^2$ for a size $m \times m$ signal transform. This is a substantial saving in computations for almost all practical values of $m$ (typically $m \geq 8$). However, as the transforms commonly used are orthogonal, large word lengths are needed to guarantee that the finite precision transform preserves that desirable property. We claim that there is an alternative structure available which is of low-cost in terms of the number of operations required and the implementation complexity of the operations.

## Organization of this thesis

The structure of this thesis is as follows. In Chapter 2 and Chapter 3 we discuss the design of discrete-time signal transforms for data compression of images. Firstly, in Chapter 2, we apply rate-distortion theory to transform coding systems in order to investigate the influence of different parameter settings of the filter bank on the coding efficiency. Secondly, in Chapter 3, we discuss factors affecting perception and implementation and derive a list of constraints on the choice of parameters imposed by the image coding application. Based on this we choose a suitable filter bank architecture and show how to design proper filter responses, i.e., filter responses which satisfy all the stated constraints. We show that the class of *lapped orthogonal transforms* (LOTs) is a suitable candidate for our specific application.

In Chapter 4 we discuss the application of overlapped transform coding with LOTs to X-ray cardio-angiographic image series. With lossless compression techniques the bit rate of the X-ray image series can be reduced by a factor of about 2.5 – 3.5 [1, 2]. The aim of the method described here, however, is to reach reduction factors in the order of 8 – 16 with lossy compression techniques. Since digital X-ray angiography plays an important role throughout this chapter, we first give a brief review of the diagnostic X-ray system and consider some X-ray chain properties which affect the selection of an appropriate compression method. We also discuss some further constraints that are imposed on the method of data compression by the clinical procedure used during digital X-ray angiography and on the basis of these constraints, we select a suitable compression technique. Next, we discuss the selected technique in more detail and describe the results obtained with a software implementation of this method. Philips has proposed that this coding scheme be included in the discussion on standardization of lossy data-compression algorithms which is being organized by the ACR-NEMA committee with the support of the National Electrical Manufacturers' Association (NEMA) and the American College of Radiology (ACR). The signal transformation, quan-

tization and the actual coding are discussed. We show that LOTs can effectively decorrelate X-ray images, which simplifies encoding of the quantized transformed signal. Moreover, we show that LOTs are well suited to adapt the compression to the properties of the human visual system and post-processing, such as image enhancement. We believe that other compression methods are much less easily adaptable, and we therefore believe that overlapped transform coding will give a higher performance.

In Chapter 5 we consider the design of realizations for discrete-time signal transforms, in particular, LOTs. We examine the numerical behaviour of different realizations under finite word-length arithmetic and how they can be implemented at the same time. We propose an algorithm which gives *optimal realizations* for arbitrary signal transforms in the sense that they are numerically robust *and* allow efficient VLSI implementation. These realizations can be made fully programmable such that the architecture can be used for arbitrary signal transforms, including DCTs, LOTs or discrete wavelet transforms.

# Chapter 2

# *Rate-distortion performance of signal transforms*

**Contents**



## 2.1 Introduction

The need to communicate at increased speed often requires that information be sent over a communication channel at a rate that exceeds its capacity. In such situations some distortion inevitably results. In order to keep this distortion to a minimum, it is necessary first to order the data produced by the source in accordance with its importance at the eventual destination, and then either condense or delete the less significant information prior to actual transmission. Schemes devised to extract only desired information from the output of a source and to eliminate redundant and irrelevant material are called *data-compression schemes.* The theoretical discipline (based on a foundation provided by information theory) dealing with data compression is called *rate-distortion theory.*

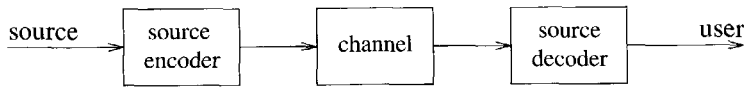



**Figure 2.1:** *Block diagram of a communication system.*

The foundations of information theory as a whole, and rate-distortion theory in particular, were laid down by C.E. Shannon in his book "A mathematical theory of communication", published in 1949 [22]. Since that time the literature on rate-distortion theory has grown rapidly. Most investigations to date have been primarily theoretical in nature. The performance achieved by various data-compression systems has been compared with absolute bounds derived from rate-distortion theory in several relatively simple cases. Attempts are currently being made to extend the theory to include highly redundant, non-stationary sources and better fidelity criteria. It is anticipated that these attempts will result in the design of more sophisticated data-compression systems. These systems, incorporating techniques based on rate-distortion theory, will be better at separating the redundant and irrelevant information from the relevant information and thereby achieve the required fidelity far more economically.

Figure 2.1 shows a diagram of a communication system which can be considered as a source encoder, a channel and a source decoder. The function of a communication system is to transmit information (data) from one point, usually referred to as the source, to another point, usually referred to as the user or destination. The purpose of the source encoder is to represent the source output by a sequence of binary digits, called bits, and one of the major questions is to determine how many bits per unit time are required to represent the output of a given source model. This number of bits per unit time (or bits per source symbol where the channel represents a storage medium) is called the bit rate. The purpose of the source decoder is to reproduce the information from the transmitted or stored sequence of bits. Since the decoder acts practically as the inverse of the encoder, we will mainly concentrate on the encoder and, where necessary, just discuss those aspects of the decoder which do not simply match their counterparts in the encoder. The channel might represent, for example, a telephone line or a storage medium. The channel is usually subject to various types of noise disturbances, which on a telephone line, for example, can take the form of crosstalk from other lines, and for a storage medium could be the coding noise introduced by the source encoder.

In practical source-coding systems, data reduction is obtained by using a quantizer and a lossless coder. The quantizer causes a substantial part of the bit-rate reduction. The quantizer maps an input symbol onto a quantizer-output symbol. Usually, the number of distinct quantizer-output values is smaller than the number of quantizer-input values. Therefore, the process of quantization is irreversible. Examples of quantizers are scalar quantizers [4, 23, 24] and vector quantizers [4, 25, 26]. The coder has two functions. The first is to map the quantizer-output symbols onto binary codewords that are transmitted or stored. The second function is to decrease the bit rate required to transmit or store the data. This is done by removing redundancy which is a reversible process. Examples of coding techniques are Huffman coding [27], Ziv-Lempel coding [28, 29, 30] and arithmetic coding [31, 32, 33, 34].

The use of a discrete-time signal transform before quantization and coding can significantly reduce the complexity of the coder [4, 26, 35]. Source-coding schemes using a signal transform are usually referred to as transform-coding schemes. A suitable signal transformation leads to signal representations that are easy to code after quantization. A second important reason for applying signal transformations is that the inverse map on the reconstruction side can transform the quantization errors in such a way that the perceptual distortion is small. This is sometimes called noise shaping. We shall concentrate on the case of separable transforms, i.e. a two-dimensional transform is performed by independently applying a one-dimensional transform in both the horizontal and vertical directions. For the purpose of the following discussion, we will restrict ourselves to one-dimensional transforms.

In [36] it was shown that a discrete-time signal transform can be regarded as a multi-rate filter bank. Placing signal transforms into this more general framework has the advantage that the basis functions can be viewed as the impulse responses of filters. Therefore, all properties known from multi-rate filter bank coding can be directly translated into properties of transform coding.

In this chapter we apply rate-distortion theory to transform-coding systems in order to investigate the influence of different parameter settings of the filter bank on coding efficiency. We show that a discrete-time signal transform can be regarded as a $(K, L, m)$ filter bank, where $K$ denotes the number of filter channels, $L$ the filter length and $m$ the down-sample factor, and we show how different filter banks (different choices of $K, L, m$) affect the coding performance in a rate-distortion sense. We do not attempt to tackle the problem of defining suitable distortion measures for given combinations of source and user. Rather, we develop some general results for various classes of distortion measures. To date,

little is known about rate-distortion theory for transform coding. Investigations have only be concerned with simple situations involving Gaussian sources and the squared-error distortion criterion [37, 38, 39, 40]. These investigations mainly concentrate on the separate encoding of filter bank channel signals. Relations have been derived between frequency selectivity and information-theoretic performance of two-channel quadrature mirror filters (QMFs) for small distortions.

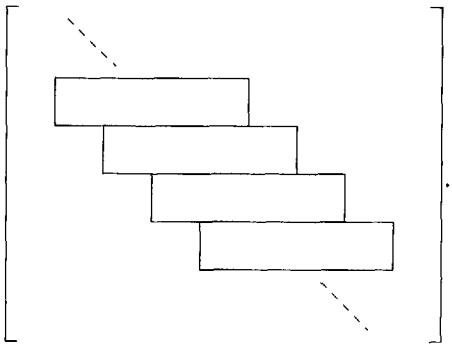
### Organization of this chapter

The remainder of this chapter is organized as follows. In Section 2.2 we discuss transform coding in some detail. We provide a precise mathematical definition of transform coding and show the equivalence of discrete-time signal transforms and multi-rate filter banks. In Section 2.3 we define the rate-distortion function of a source where a discrete-time signal transform is applied before the actual coding takes place. In Section 2.4, we investigate the influence of the parameter $m$ on the bit rate needed to represent the output of a source, as well as its influence on the complexity of the source encoder. The results thus obtained do not require Gaussian input distributions and hold true for arbitrary distortion measures. In Section 2.5, we show under what conditions we can separately encode the $K$ filter bank channels while still reaching the rate-distortion bound. The results obtained do not require QMFs and the validity of the results is not restricted to small distortions. Rather, we derive results that hold true for any distortion and for arbitrary filter banks. Moreover, we show how we can use these conditions to design proper filters for Gaussian input distributions. In Section 2.6, we discuss the problem of non-stationary sources. Under some reasonable assumptions about the nature of the non-stationarity, we show how this determines the optimal filter length $L$. Finally, in Section 2.7, we draw some conclusions.

## 2.2 Transform coding

Let $\{X_n, n \in \mathbb{Z}\}$ be a discrete source defined on a probability space $(\Omega, \mathcal{A}, \mathcal{P})$ (see Appendix A) and let $M$, $N$ be positive integers. Denote by $x$ any sequence of $N$ successive sample values of $\{X_n, n \in \mathbb{Z}\}$. That is, $x = (x(n))_{n=1}^{N}$ where $x(n) = X_n(\omega)$, $\omega \in \Omega$. Also, let the sequence $y = (y(n))_{n=1}^{N}$ of sample values of a random process $\{Y_n, n \in \mathbb{Z}\}$ denote the reproduction of the source sequence $x$. A linear discrete-time signal transform is a map $T_a \in \mathbb{C}^{M \times N} : x \mapsto u = T_a x$. $T_a$ is called the *analysis map* and is commonly a block-banded Toeplitz map, i.e. it is

of the form A common example of an analysis map is the discrete cosine trans-

$T_a =$ 

form (DCT). In this case the block entries of $T_a$ are square and non-overlapped. Instead of encoding the source sequence $x$ directly, we encode the transformed sequence $u = (u(n))_{n=1}^{M}$. After decoding, the reproducing sequence $v = (v(n))_{n=1}^{M}$ is inversely transformed by a linear map $T_s \in \mathbb{C}^{N\times M} : v \mapsto y = T_s v$, called the *synthesis map*. In the remainder of this chapter we consider the sequences $u$ and $v$ as sample sequences of random processes $\{U_n, n \in \mathbb{Z}\}$ and $\{V_n, n \in \mathbb{Z}\}$, respectively. Figure 2.2 shows a block diagram of a source-coding system in which a discrete-time signal transform is used. The boxes labelled $T_a$, $Q$, $C$, $D$, $R$ and $T_s$ denote the analysis map, quantization, lossless coding, decoding, reconstruction and the synthesis map, respectively.

As mentioned before, the use of a signal transform prior to the actual coding can significantly reduce the coder complexity required to code the data at a minimum rate. An appropriate transform removes the dependency in the source sequence, resulting in a sequence in which the sum of the entropies of the coefficients is minimized [4, 26, 35]. The optimal transform maps $x$ onto a sequence $u$ where the sample values are statistically independent. Clearly, this transform is signal dependent and, therefore, very complex. In practice, optimal transforms are approximated by less complex signal-independent transforms which are either constant or piecewise constant in time. In the former case we speak of a time invariant transform, whereas in the latter case, the transform is said to be time varying.

Let $K, L, m$ be positive integers. If $T_a$ is time invariant, i.e., it has a block-banded Toeplitz structure, it can be represented by a $(K, L, m)$ multi-rate filter bank as shown by the diagram in Figure 2.3. That is, we have collections $\mathcal{A} = (h_k)_{k=1}^{K}, \mathcal{S} = (f_k)_{k=1}^{K}$ of $K$ filters of length $L$ with impulse responses $h_k = (h_k(n))_{n=1}^{L}$, $f_k = (f_k(n))_{n=1}^{L}$ having $\mathcal{Z}$-transforms $H_k(z)$, $F_k(z)$, respec-

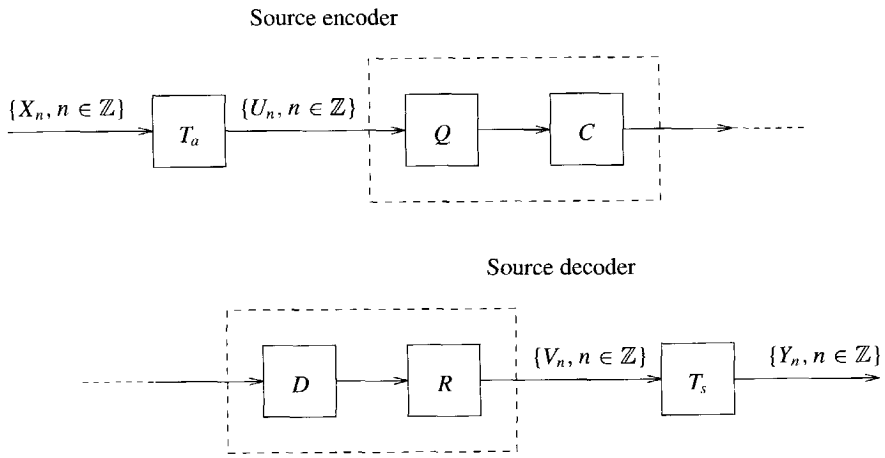



**Figure 2.2:** *Block diagram of a waveform encoder/decoder system.*

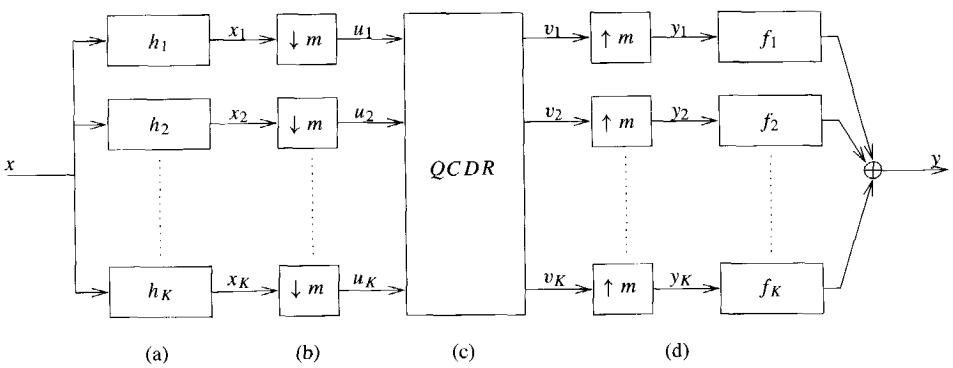



**Figure 2.3:** *Diagram of a multi-rate filter bank.*

tively. The set $\mathcal{A} = (h_k)_{k=1}^K$ is the set of analysis filters, the set $\mathcal{S} = (f_k)_{k=1}^K$ is the set of synthesis filters. The boxes $\downarrow m$ in Figure 2.3b are decimators, taking every $m$th element from $x_k$, $k = 1, \ldots, K$, i.e. they down-sample the filter outputs $x_k$ to $u_k$. In many applications the signals $u_k$ are coded, transmitted or stored, and decoded (see Figure 2.3c in which the box labelled $QCDR$ represents the concatenation of quantization, lossless coding, decoding and reconstruction), after which the original signal is reproduced using an inverse transform. This is essentially the sum of the interpolated received band signals $v_k$ (see Figure 2.3d). The boxes $\uparrow m$ indicate that $m-1$ zero-valued samples are inserted between every two samples of the received signals $v_k$. Note that the integers $K, L, m$ are the row dimension, the column dimension and the displacement of the block entries of $T_a$, respectively, and that $u = T_a x = (u_k)_{k=1}^K$ and the reproducing sequence is given by $v = (v_k)_{k=1}^K$. A more detailed description of multi-rate filter banks can be found in [41, 42, 43, 44].

## 2.3 Rate-distortion function

In the previous section we provided a precise mathematical definition of transform coding and showed the equivalence of discrete-time signal transforms and multi-rate filter banks. In this section we define the rate-distortion function of a source where a discrete-time signal transform is applied before actual coding takes place.

When information is transmitted from one point to another, the receiver often does not require a perfect copy of the source output but will settle for a sufficiently accurate approximation. In order to determine quantitatively whether or not the performance of a communication system is satisfactory, it is necessary to assign numerical values to the various errors that the system may make. Since the impact of an error usually depends critically on the context in which it occurs, measures of distortion can be quite complex.

Let $\{X_n, n \in \mathbb{Z}\}$ be a discrete source defined on a probability space $(\Omega, \mathcal{A}, \mathcal{P})$ and consider the problem of transmitting the output of $\{X_n, n \in \mathbb{Z}\}$ to the destination within a certain prescribed accuracy. Let $x = (x(n))_{n=1}^N$ be any sequence of $N$ successive sample values of $\{X_n, n \in \mathbb{Z}\}$ and let the sequence $y = (y(n))_{n=1}^N$ of sample values of a random process $\{Y_n, n \in \mathbb{Z}\}$ denote a reproduction of the sequence $x$. To determine whether $y$ is sufficiently close to $x$, we need a rule for quantitatively assigning a distortion value to every possible approximation of $x$.

Following [22, 45, 46, 47], let us assume for this purpose that a non-negative cost function is given, say $\rho_N(x, y)$, which specifies the penalty for approximating

the sequence $x$ by the sequence $y$. A sequence of distortion measures $F_\rho = (\rho_l)_{l=1}^{\infty}$ is called a *fidelity criterion*. A fidelity criterion composed of distortion measures of the form

$$\rho_N(x, y) = \frac{1}{N} \sum_{n=1}^{N} \rho(x(n), y(n)), \tag{2.1}$$

is called a *single-letter fidelity criterion*. In that case the distortion between a source sequence and a reproduction sequence is simply the arithmetic average of the distortions between their corresponding sample values specified by a fixed single-letter distortion measure $\rho$. As an example of such a distortion measure, we might have $\rho(x(n), y(n)) = 0$ for $x(n) = y(n)$ and $\rho(x(n), y(n)) = 1$ for $x(n) \neq y(n)$. Such a distortion measure would be appropriate if we were interested in reproducing a source sequence exactly and considered all errors as being equally serious. Other, more frequently used distortion measures, are the *squared-error distortion measure* defined by

$$\rho_N(x, y) = \frac{1}{N} \|x - y\|_2^2,$$

and the *magnitude-error distortion measure* defined by

$$\rho_N(x, y) = \frac{1}{N} \|x - y\|_1,$$

the latter being popular because of its low computational complexity.

To every conditional probability density $q(y|x)$ defined on $\mathcal{A}$ we assign both an average distortion

$$d(x, y) = \int\int p(x)q(y|x)\rho_N(x, y)dxdy,$$

and an average mutual information

$$I(x; y) = \int\int p(x)q(y|x) \log \frac{q(y|x)}{q(y)} dxdy.$$

The rate-distortion function of $\{X_n, n \in \mathbb{Z}\}$ with respect to $F_\rho$, then, is defined by

$$R_x(D) = \lim_{N \to \infty} R_{x,N}(D),$$

where

$$R_{x,N}(D) = \frac{1}{N} \inf_{q \in \mathcal{Q}_D} I(x; y),$$

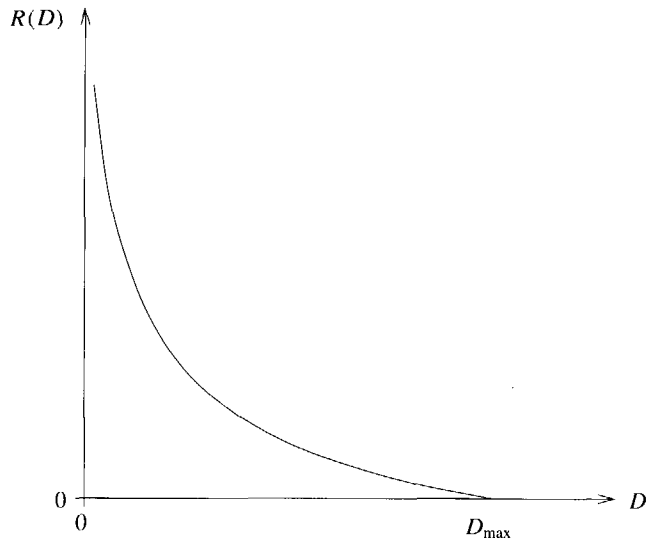



**Figure 2.4:** *A typical rate-distortion function.*

and

$$\mathcal{Q}_D = \{q(y|x) : d(x, y) \leq D\}.$$

See also [22, 45, 46, 47] for further details. The subscript $x$ is used to indicate that the source $\{X_n, n \in \mathbb{Z}\}$ itself is transmitted rather than a transformed process, say $\{U_n, n \in \mathbb{Z}\}$, a situation we shall discuss extensively in the following sections. Thus, $R_x(D)$ is the effective rate at which the source $\{X_n, n \in \mathbb{Z}\}$ generates information when a distortion less than or equal to $D$ is required with respect to $F_\rho$. Further, from its definition it is clear that $R_x(D)$ is a monotonically decreasing function of $D$. The rate at which a source produces information subject to the requirement of perfect reproduction is called the entropy of the source. It follows that the rate-distortion function is a generalization of the concept of entropy. Indeed, if the distortion measure is such that a perfect reproduction is assigned zero distortion, then $R_x(0)$ equals the source entropy. As $D$ increases, $R_x(D)$ decreases monotonically and usually becomes zero at some finite value of the distortion $D$. With continuous amplitude sources, the (absolute) entropy is infinite and the $R_x(D)$ curves become unbounded as $D$ approaches zero. Figure 2.4 shows a typical rate-distortion function.

The calculation of the rate-distortion function for sources with memory is

quite difficult. To date, this function is known specifically only in situations involving Gaussian sources and certain modifications of the squared-error criterion [48], and some special non-Gaussian cases. The usual approach to minimize the function $I(x; y)$ subject to the constraint $d(x, y) \leq D$, is to use a Lagrange multiplier, say $s$, and to minimize the Lagrangian function

$$L_x(q, s) = \frac{1}{N} I(x; y) + sd(x, y),$$

over the set of conditional probabilities $q(y|x)$. Then, by varying $s \geq 0$ we can find $R_x(D)$ for all values of $0 < D \leq D_{\max}$. In this approach, the multiplier $s$ has the geometric significance of being the magnitude of the slope of the $R_x(D)$ curve at the point generated by that value of $s$. For a clear introduction to the Lagrange-multiplier technique the reader is referred to [49].

**Remark 2.3.1** *The Lagrange-multiplier technique is applied to minimization problems subject to a set of equality constraints. Since it can be shown that $I(x; y)$ is a convex function on the closed convex region $\mathcal{Q}_D$, there is one and only one minimum, $R_{x,N}(D)$, and it always occurs at a point having an average distortion $d(x, y)$ that equals $D$, i.e. the minimum value of $I(x; y)$ in $\mathcal{Q}_D$ is achieved at the boundary of $\mathcal{Q}_D$. This is the reason that it is allowed to minimize $I(x; y)$ subject to $d(x, y) = D$.*

Let $T_a$ and $T_s$ be fixed transforms and let $R_u(D)$ denote the rate-distortion function of $\{X_n, n \in \mathbb{Z}\}$ with respect to $F_\rho$ where the transformed process $\{U_n, n \in \mathbb{Z}\}$ is transmitted rather than the source process $\{X_n, n \in \mathbb{Z}\}$ itself. Since we are interested in the effective rate at which the transformed process $\{U_n, n \in \mathbb{Z}\}$ generates information when the distortion between the sequences $x$ and $y$ must be less than or equal to $D$, we define $R_u(D)$ by

$$R_u(D) = \lim_{M \to \infty} R_{u,M}(D),$$

where

$$R_{u,M}(D) = \frac{1}{M} \inf_{r \in \mathcal{R}_D} I(u; v),$$

and

$$\mathcal{R}_D = \{r(v|u) : d(x, y) \leq D\}.$$

This means that instead of minimizing $I(x; y)$ subject to $d(x, y) \leq D$ we minimize $I(u; v)$ subject to $d(x, y) \leq D$.

## 2.4 Influence of the down-sample factor on $R(D)$

Commonly used discrete-time signal transform systems, such as subband coding systems [16, 18, 19, 50] and systems based on the DCT [5, 51], are critically sampled, i.e. $M = N$ (or equivalently $m = K$). The main reason for choosing $M = N$ is not quite clear from the literature but in the author's opinion this is because it leads to the lowest possible sample rate at which perfect reconstruction is still possible and thus to the minimum number of arithmetic operations per sample per unit time.

The situation in which $M > N$ can offer some advantages over the critically sampled case. For example, let's assume that $T_s T_a = I$, i.e., $T_s$ is a left inverse of $T_a$, and assume that we want to impose some frequency-domain constraints on the analysis filters, e.g., with a minimal ripple in the pass-band or with a certain attenuation in the stop-band. Let us also assume that we want to impose some time-domain constraints on the synthesis filters, like smoothness or length of impulse responses [52]. In critical sampling the analysis and synthesis maps are square, i.e., $N = M$, so that given $T_a$, $T_s$ is uniquely determined. In other words, there is a one-to-one correspondence between the analysis and synthesis filters. This means that all requirements imposed on the synthesis filters can be directly translated into requirements imposed on the analysis filters, and vice versa. Imposing both time-domain and frequency-domain requirements on the filters, however, can lead to suboptimal solutions. However, if we choose $M > N$, then the representation is redundant and $T_s$ is not uniquely determined by $T_a$. In fact, for any $T_a$ there are an infinite number of $T_s$ satisfying $T_s T_a = I$. The conclusion we can draw from this is that non-critical sampling can give us some extra freedom to optimize $T_a$ and $T_s$ simultaneously.

Recently, analysis/synthesis systems in which $M > N$, also called oversampled filter banks, have been extensively studied. In [53, 54, 55] oversampling has been studied using the theory of polynomial matrices while in [56, 57] oversampled systems have been studied using the theory of Weyl-Heisenberg frames. A vector-filter framework for the study of oversampled analysis/synthesis systems has been proposed in [58]. Oversampled filter banks are known to have improved noise immunity compared with critically sampled filter banks [15, 56, 59]. The improved immunity of redundant representations allows coarser quantization of the filter bank channel. This improved noise immunity, however, does not imply that non-critically sampled filter banks are more suitable than critically sampled systems for source-coding applications, since no consideration is given to the associated number of bits needed to reproduce the redundant representations. It is

the author's belief that no significant conclusion can be drawn by considering rate or distortion separately, but that they have to be considered simultaneously in a rate-distortion sense.

The questions that arise here are what is the increase in sample rate costs in terms of bit rate needed to represent the source process $\{X_n, n \in \mathbb{Z}\}$ with an average distortion less than or equal to $D$, and what is the impact on the source-encoder complexity. In this section we investigate these problems and we show what the relation is between the oversample ratio $\mu = \frac{M}{N}$ and the rate-distortion function. To do this, we first show what the relation is between $I(u; v)$ and $I(x; y)$. Based on this we derive a relation between $R_u(D)$ and $R_x(D)$.

To facilitate the discussion we have adopted a few conventions. Let $\{X_n, n \in \mathbb{Z}\}$ be an arbitrary random process defined on a probability space $(\Omega, \mathcal{A}, \mathcal{P})$. A sample sequence $X(\omega) = (X_n(\omega))_{n=1}^N$ will be denoted by $x = (x_n)_{n=1}^N$ where $x \in \Gamma_x$, the source alphabet defined by $\Gamma_x = \{X(\omega) : \omega \in \Omega\}$.

We now introduce the notion of *equivalent elements*. Equivalent elements play a central role in the derivation of the relation between $R_u(D)$ and $R_x(D)$. As we will show, it appears that the mutual information $I(u; v)$ equals the mutual information $I(x; y)$ if and only if no non-equivalent elements in $\Gamma_x$ and $\Gamma_v$ are ever confused after the analysis and synthesis map has been applied to $X$ and $V$, respectively.

**Definition 2.4.1 (equivalent elements)** *Let $X, Y$ and $Z$ be jointly distributed random variables defined on a common probability space $(\Omega, \mathcal{A}, \mathcal{P})$. The elements $y \in \Gamma_y, z \in \Gamma_z : p(y) > 0, p(z) > 0$ are said to be equivalent modulo $X$ if and only if $p(x|y) = p(x|z)$ for all $x \in \Gamma_x$ and we write $y \equiv z \pmod{X}$.*

We have the following equivalence relation.

**Lemma 2.4.1** *Let $X, Y$ and $Z$ be jointly distributed random variables defined on a common probability space $(\Omega, \mathcal{A}, \mathcal{P})$.*

1. *If $y = z$, then $y \equiv z \pmod{X}$.*
2. *If $y \equiv z \pmod{X}$, then $z \equiv y \pmod{X}$.*
3. *If $y_1 \equiv z \pmod{X}$ and $y_2 \equiv z \pmod{X}$, then $y_1 \equiv y_2 \pmod{X}$.*

**Proof:** The proof of assertions 1 and 2 is trivial. To prove assertion 3 we conclude that since $p(x|y_1) = p(x|z)$ and $p(x|y_2) = p(x|z)$ for all $x \in \Gamma_x$ we have $p(x|y_1) = p(x|y_2)$ for all $x \in \Gamma_x$ so that $y_1 \equiv y_2 \pmod{X}$. □

The property of being equivalent modulo $X$ is independent of the source distribution and only depends on the channel. This can be seen as follows. Using Bayes' rule, we have

$$p(x|y) = \frac{p(x)}{p(y)} p(y|x).$$

Hence, if $y \equiv z \pmod{X}$ we have $p(x|y) = p(x|z)$ for all $x$ and we conclude that

$$\frac{p(y|x)}{p(z|x)} = \frac{p(y)}{p(z)} \quad \text{for all} \quad x \in \Gamma_x.$$

Let $p(y|x) = \alpha p(z|x)$. If we let $q(x)$ be another input distribution, we have $q(x, y) = q(x)p(y|x) = \alpha q(x)p(z|x) = \alpha q(x, z)$ so that $q(y) = \alpha q(z)$ and we conclude that

$$\frac{p(y|x)}{p(z|x)} = \frac{q(y)}{q(z)} \quad \text{for all} \quad x \in \Gamma_x,$$

and $y$ and $z$ are still equivalent in the new distribution. We illustrate the notion of equivalent elements with an example.

**Example 2.4.1** *Let $\Gamma_x = \{x_1, x_2\}$, $\Gamma_y = \{y_1, y_2, y_3, y_4\}$ and let the conditional probabilities $p(y_i|x_j)$ be given as depicted in Figure 2.5. Since $p(y_1|x_i) = p(y_4|x_i)$ and thus $p(y_1, x_i) = p(y_4, x_i)$ for $i = 1, 2$, we conclude that $p(y_1) = p(y_4)$ and, therefore, $p(x_i|y_1) = p(x_i|y_4)$ for $i = 1, 2$. Hence, the elements $y_1$ and $y_4$ are equivalent modulo $X$. A similar derivation can be made for $y_2$ and $y_3$. Note that if $p = q = \frac{1}{2}$ we have $y_i \equiv y_j \pmod{X}$ for all $i, j = 1, \ldots, 4$, since in that case no information is transmitted at all, i.e. the capacity of the channel is zero.*

We are now in the position to derive one of the main results of this section which is the relation between $I(x; y)$ and $I(u; v)$. Instead of limiting ourselves just to linear operators $T_a$ and $T_s$, we shall obtain some results for rather arbitrary mappings, either deterministic or stochastic, and translate these results to transform-coding systems. In what follows, we will repeatedly use the following results.

**Theorem 2.4.1** *Let $X$, $Y$ and $Z$ be jointly distributed random variables defined on a common probability space $(\Omega, \mathcal{A}, \mathcal{P})$ and let $f$ be a measurable function.*

1. *If $Z = f(Y)$, then $I(x; y|z) = 0$ if and only if $\forall y_1, y_2 \in \Gamma_y : f(y_1) = f(y_2) \Rightarrow y_1 \equiv y_2 \pmod{X}$.*

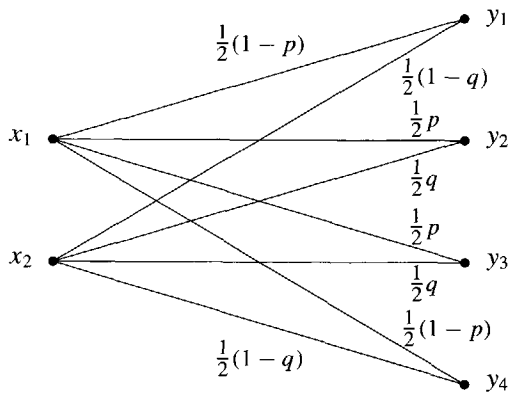



**Figure 2.5:** *An example of equivalent symbols.*

2. *If* $Y = f(X)$, *then* $I(z; y|x) = 0$ *if and only if* $\forall x_1, x_2 \in \Gamma_x : f(x_1) = f(x_2) \Rightarrow x_1 \equiv x_2 \pmod{Z}$.

The proof of Theorem 2.4.1 is given in Appendix B. Here follows an example to illustrate these results.

**Example 2.4.2** *Consider* $\Gamma_x$, $\Gamma_y$ *and the conditional probabilities* $p(y_i|x_j)$ *as given in Example 2.4.1. Let* $Z = f(Y)$ *be given by*

$$z = \begin{cases} z_1 & \text{if } y = y_1, \\ z_2 & \text{if } y = y_2 \text{ or } y = y_3, \\ z_3 & \text{if } y = y_4. \end{cases}$$

*Figure 2.6 shows this situation. Since* $y_2 \equiv y_3 \pmod{X}$ *(see Example 2.4.1), we have* $\forall y_i, y_j \in \Gamma_y : f(y_i) = f(y_j) \Rightarrow y_i \equiv y_j \pmod{X}$ *and we conclude that* $I(x; y) = I(x; z)$. *Hence, the second channel destroys no information about* $X$ *since no non-equivalent elements in* $\Gamma_y$ *are ever confused in the second channel.*

Next let $\{U_n, n \in \mathbb{Z}\}$, $\{V_n, n \in \mathbb{Z}\}$, $\{X_n, n \in \mathbb{Z}\}$ and $\{Y_n, n \in \mathbb{Z}\}$ be jointly distributed random processes defined on a common probability space $(\Omega, \mathcal{A}, \mathcal{P})$. We have the following result which gives the relation between $I(x; y)$ and $I(u; v)$.

**Theorem 2.4.2** *Let* $f$ *and* $g$ *be measurable functions. If* $U = f(X)$ *and* $Y = g(V)$, *then* $I(u; v) \geq I(x; y)$ *with equality if and only if* $\forall x_1, x_2 \in \Gamma_x : f(x_1) = f(x_2) \Rightarrow x_1 \equiv x_2 \pmod{Y}$ *and* $\forall v_1, v_2 \in \Gamma_v : g(v_1) = g(v_2) \Rightarrow v_1 \equiv v_2 \pmod{U}$.

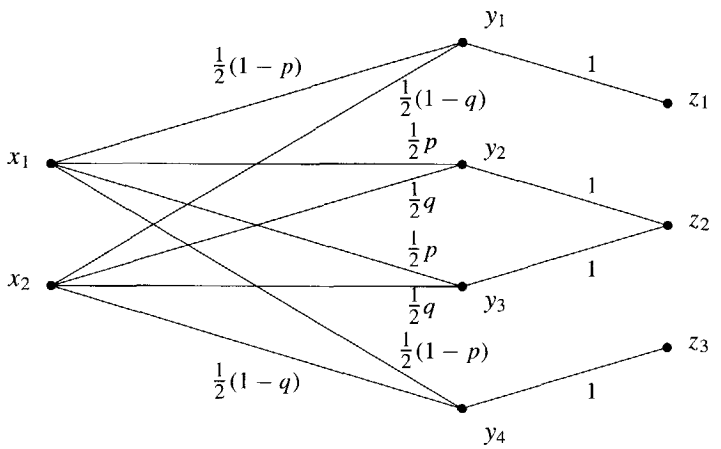



**Figure 2.6:** *An example of an information-preserving map, i.e.* $I(x;y) = I(x;z)$.

**Proof:** Using the data-processing inequality (see Theorem A.3.2), we have $I(u;v) = I(u;y) + I(u;v|y) = I(x;y) + I(u;y|x) + I(u;v|y) \geq I(x;y)$ with equality if and only if $I(u;v|y) = 0$ and $I(u;y|x) = 0$. Thus, using the results of Theorem 2.4.1, we conclude that $I(u;v) = I(x;y)$ if and only if $\forall v_1, v_2 \in \Gamma_v : g(v_1) = g(v_2) \Rightarrow v_1 \equiv v_2 \pmod{U}$ and $\forall x_1, x_2 \in \Gamma_x : f(x_1) = f(x_2) \Rightarrow x_1 \equiv x_2 \pmod{Y}$. □

At this point we are almost ready to derive the relation between $R_u(D)$ and $R_x(D)$. To do this, we need the following result.

**Theorem 2.4.3** *Let $f$ and $g$ be measurable functions. If $U = f(X)$ and $Y = g(V)$, then*

$$\inf_{r \in \mathcal{R}_D} I(u;v) \geq \inf_{q \in \mathcal{Q}_D} I(x;y), \tag{2.2}$$

*with equality if and only if $\forall x_1, x_2 \in \Gamma_x : f(x_1) = f(x_2) \Rightarrow x_1 \equiv x_2 \pmod{Y}$ and $\forall v_1, v_2 \in \Gamma_v : g(v_1) = g(v_2) \Rightarrow v_1 \equiv v_2 \pmod{U}$.*

**Proof:** Obviously, we have

$$\inf_{r \in \mathcal{R}_D} I(u;v) \geq \inf_{q \in \mathcal{Q}_D} I(x;y),$$

since the rate of any code exceeds the rate-distortion function $R_x(D)$ evaluated at the distortion level $D$ achieved by that code.

First let us assume that we have equality in (2.2). In that case there exist $r(v|u) \in \mathcal{R}_D$ and $q(y|x) \in \mathcal{Q}_D$ such that $I(u;v) = I(x;y)$ and we conclude from Theorem 2.4.2 that $\forall x_1, x_2 \in \Gamma_x : f(x_1) = f(x_2) \Rightarrow x_1 \equiv x_2 \pmod{Y}$ and $\forall v_1, v_2 \in \Gamma_v : g(v_1) = g(v_2) \Rightarrow v_1 \equiv v_2 \pmod{U}$.

On the other hand, if we assume that $\forall x_1, x_2 \in \Gamma_x : f(x_1) = f(x_2) \Rightarrow x_1 \equiv x_2 \pmod{Y}$ and $\forall v_1, v_2 \in \Gamma_v : g(v_1) = g(v_2) \Rightarrow v_1 \equiv v_2 \pmod{U}$ we have $I(u;v) = I(x;y)$ by Theorem 2.4.2. To prove equality in (2.2) we show that the left-hand side of (2.2) is less than or equal to the right-hand side. Since $\forall x_1, x_2 \in \Gamma_x : f(x_1) = f(x_2) \Rightarrow x_1 \equiv x_2 \pmod{Y}$ there exists a measurable function, say $h$, such that if $u = f(x)$, then $h(u) \equiv x \pmod{Y}$ and therefore $q(y|h(u)) = q(y|x)$ for all $y \in \Gamma_y$. Hence, for every $q(y|x) \in \mathcal{Q}_D$ there exist $r(v|u) \in \mathcal{R}_D$ such that $r(v|u) = q(y|h(u)) = q(y|x)$ and we conclude that

$$\inf_{r \in \mathcal{R}_D} I(u;v) \leq \inf_{q \in \mathcal{Q}_D} I(x;y),$$

which is a contradiction. □

This brings us to the second main theorem of this section, which gives the relation between $R_u(D)$ and $R_x(D)$.

**Theorem 2.4.4** *Let $f$ and $g$ be measurable functions such that $U = f(X)$ and $Y = g(V)$. If $x, y$ are $N$-element sequences and $u, v$ are $M$-element sequences, then*

$$R_u(D) \geq \mu^{-1} R_x(D), \qquad \mu = \frac{M}{N}, \tag{2.3}$$

*with equality if and only if $\forall x_1, x_2 \in \Gamma_x : f(x_1) = f(x_2) \Rightarrow x_1 \equiv x_2 \pmod{Y}$ and $\forall v_1, v_2 \in \Gamma_v : g(v_1) = g(v_2) \Rightarrow v_1 \equiv v_2 \pmod{U}$.*

**Proof:** We have

$$\begin{aligned}
R_{u,M}(D) &= \frac{1}{M} \inf_{r \in \mathcal{R}_D} I(u;v) \\
&\geq \frac{1}{M} \inf_{q \in \mathcal{Q}_D} I(x;y) \\
&= \frac{N}{M} \left( \frac{1}{N} \inf_{q \in \mathcal{Q}_D} I(x;y) \right) \\
&= \frac{N}{M} R_{x,N}(D) \\
&= \mu^{-1} R_{x,N}(D),
\end{aligned}$$

and thus

$$\begin{aligned} R_u(D) &= \lim_{M\to\infty} R_{u,M}(D) \\ &\geq \mu^{-1} \lim_{N\to\infty} R_{x,N}(D) \\ &= \mu^{-1} R_x(D), \end{aligned}$$

with equality if and only if $\forall x_1, x_2 \in \Gamma_x : f(x_1) = f(x_2) \Rightarrow x_1 \equiv x_2 \pmod{Y}$ and $\forall v_1, v_2 \in \Gamma_v : g(v_1) = g(v_2) \Rightarrow v_1 \equiv v_2 \pmod{U}$, according to Theorem 2.4.3. □

Theorem 2.4.4 says that we can reach the rate-distortion bound if and only if no non-equivalent elements in $\Gamma_x$ and $\Gamma_v$ are ever confused after the analysis and synthesis map have been applied to $X$ and $V$, respectively. In the remainder of this thesis we will refer to signal transforms having this property as *information-preserving transforms*.

Some remarks are in order here. To start with, the result of Theorem 2.4.4 shows that oversampled analysis/synthesis systems have improved noise immunity compared with critically sampled systems, since the effective rate at which the source $\{U_n, n \in \mathbb{Z}\}$ generates information when a distortion $d(x, y)$ is required less than or equal to $D$ with respect to $F_\rho$ decreases linearly with the oversample ratio $\mu$. This does not imply, however, that the total rate needed to store or transmit the data decreases with the ratio $\mu$. This can be seen as follows.

By definition, $R(D)$ gives the effective rate at which the source generates information in bits per sample. Hence, the total rate needed to store or transmit the data equals $R(D)$ times the number of samples. Let $B_x(D)$ and $B_u(D)$ denote the total rate needed to store or transmit the sequence of sample values $x$ and $u$, respectively. In that case we have $B_u(D) = MR_u(D) \geq M\left(\mu^{-1}R_x(D)\right) = M\left(\frac{N}{M}R_x(D)\right) = NR_x(D) = B_x(D)$, so that an increase of the oversample ratio $\mu$ does not decrease the total rate needed to reproduce the source $\{X_n, n \in \mathbb{Z}\}$. This is equivalent by saying that, given a fixed rate, an increase of the oversample ratio $\mu$ does *not* decrease the average distortion $d(x, y)$. Figure 2.7 shows an example of the rate-distortion functions $R_x(D)$ of the source and $R_u(D)$ of the source when the transformed process is transmitted where the oversample ratio equals $\mu = 2$. So, with regard to rate-distortion, oversampling does not offer advantages over critical sampling. In fact, non-critically sampled filter banks have a major disadvantage over critically sampled ones, namely the coder complexity. The reason for greater coder complexity lies in the fact that the transformed sequence $u$ in oversampled systems is a redundant representation of the original

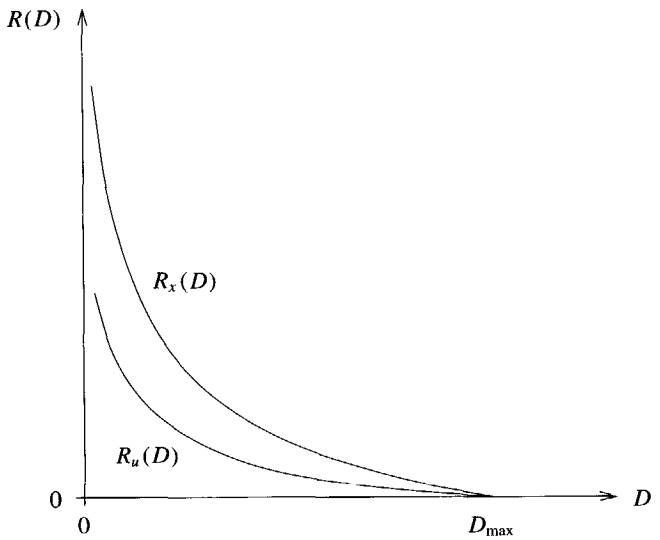



**Figure 2.7:** *A typical rate distortion function $R_x(D)$ of the source and $R_u(D)$ of the source when the transformed process is transmitted ($\mu = 2$).*

source sequence $x$. To code the sequence $u$ with a minimum number of bits, the source encoder should be able to remove the redundancy introduced by the oversampling. This inevitably implies an increase in encoder complexity compared with the critically sampled situation, thus obviating the fact that the purpose of the signal transform is to *reduce* coder complexity. A second disadvantage of oversampling is that it increases the sample rate in the channel. This increase in sample rate can give severe problems in real-time data-compression applications as it increases the computational complexity of the hardware implementation and, thereby, its silicon area in the case of a VLSI implementation.

In conclusion, it should be noted that the results derived in this section are independent of the perfect reconstructing condition such that even where $T_s T_a \neq I$, we can have equality in (2.3).

## 2.5 Independent coding of the filter bank channels

As stated above, the use of a signal transform prior to the actual coding can significantly reduce the coder complexity required to code the data at a minimum

rate. The optimal transform maps the input sequence onto a sequence in which the coefficients are statistically independent. This transform, however, is far too complex to implement and, therefore, we will approximate it by a less complex signal-independent transform. This, of course, at the expense of increased coder complexity. If we can design a filter bank in which the channel signals can be coded separately, while still reaching the rate-distortion bound, we can trade-off between coder complexity and filter bank complexity by varying $K$, the number of channels. In this section we show under what conditions this is possible. We will also show how this can be used to design proper filters for Gaussian input distributions. We assume that $\rho_M(u, v)$, the penalty charge for reproducing the sequence $u$ by the sequence $v$, is a single-letter distortion measure, that is, $\rho_M(u, v)$ is of the form (2.1).

Let us assume that the transformed process $\{U_n, n \in \mathbb{Z}\}$ is transmitted rather than the process $\{X_n, n \in \mathbb{Z}\}$ itself and that the channel sequences $u_k, k = 1, \ldots, K$, are coded independently. This means that the reproducing sequence $v_k$ depends only on the sequence $u_k$ so that no information from the other channel sequences $u_k$, $k \neq l$, is used. Formally stated, independent coding of the filter bank channels means that we have $I(v_k; u|u_k) = 0$ for all $k$. An equivalent interpretation of this statement is that, due to the independent coding, we impose some restrictions on the set of possible conditional probabilities $r(v|u)$. Let $\mathcal{R}'_D$ be this restricted set of conditional probabilities $r(v|u)$ which gives rise to an average distortion $d(x, y)$ less than or equal to $D$, i.e., from (A.4),

$$\mathcal{R}'_D = \{r(v|u) \in \mathcal{R}_D : r(v_k, u|u_k) = r(v_k|u_k)r(u|u_k),\ k = 1, \ldots, K\}.$$

From Lemma B.1.1 we conclude that independent coding implies that $V_k \to U_k \to U_l$ or, equivalently, $U_l \to U_k \to V_k$, corresponding to what is intuitively suspected. In that case we can define the rate-distortion function of $\{X_n, n \in \mathbb{Z}\}$ with respect to $F_\rho$ as

$$R'_u(D) = \lim_{M \to \infty} R'_{u,M}(D),$$

where

$$R'_{u,M}(D) = \frac{1}{M} \inf_{r \in \mathcal{R}'_D} \sum_{k=1}^{K} I(u_k; v_k).$$

The main result of this section is the following theorem.

**Theorem 2.5.1** *Let $T_a$ and $T_s$ be information-preserving signal transforms such that $u = T_a x$ and $y = T_s v$. If $x, y$ are $N$-element sequences and $u, v$ are $M$-element sequences, and $R_x(D)$ and $R'_u(D)$ denote the rate-distortion function of the source process and the transformed process where the filter bank channels are coded separately, respectively, then $R'_{u,M}(D) \geq \mu^{-1} R_{x,N}(D)$, $\mu = \frac{M}{N}$, with equality if and only if*

$$r(u) = \prod_{k=1}^{K} r(u_k).$$

The proof of Theorem 2.5.1 is given in Appendix C. A consequence of this theorem is that if we want to reach the rate-distortion bound $R_x(D)$, we have to design our discrete-time signal transform such that the filter bank channel signals are mutually independent, i.e., if and only if $I(u_k; u_1 \cdots u_{k-1}) = 0$ for all $k = 1, \ldots, K$. In the case of a Gaussian source it is then sufficient to make the signals $u_k$ mutually uncorrelated. In the remainder of this section we will use this property to design proper signal transforms for wide-sense stationary (w.s.s.) Gaussian input distributions.

Let us consider the analysis filter bank and let the impulse responses be $h_k \in l^2(\mathbb{Z})$. Also, let the source process $\{X_n, n \in \mathbb{Z}\}$ be a w.s.s. process. Thus we conclude from Theorem A.3.4 that the autocorrelation function $R$ of $\{X_n, n \in \mathbb{Z}\}$ is of the form

$$R(h) = \frac{1}{2\pi} \int_{-\pi}^{\pi} e^{i\xi h} S(\xi) d\xi,$$

where $h \in \mathbb{Z}$ and $S$ is the spectral-density function of $\{X_n, n \in \mathbb{Z}\}$.

It is natural to view the output $X_{k,n}(\omega), n \in \mathbb{Z}$, of the filter $h_k$ as a sample function of another random process and write

$$X_{k,n}(\omega) = \sum_{l=-\infty}^{\infty} h_k(n-l) X_l(\omega), \qquad k = 1, \ldots, K. \tag{2.4}$$

The decimators in the analysis bank each down-sample the sample functions $X_{k,n}(\omega)$ of $\{X_{k,n}, n \in \mathbb{Z}\}$ by a factor of $m$ to the sample functions $U_{k,n}(\omega)$ of $\{U_{k,n}, n \in \mathbb{Z}\}$ given by

$$U_{k,n}(\omega) = X_{k,mn}(\omega), \qquad k = 1, \ldots, K.$$

The result will be that $U_k$ and $U_l$ are uncorrelated if $EU_{k,n}\overline{U_{l,j}} = 0$ for all $j, n \in \mathbb{Z}$. So, to design proper signal transforms, we need an expression for $EU_{k,n}\overline{U_{l,j}}$ in

terms of the filters $h_k$ and $h_l$, the down-sample factor $m$ and the spectral-density function $S$ of the source $\{X_n, n \in \mathbb{Z}\}$. This relation will be given in Theorem 2.5.3 below, for which we need the following lemmas which show that both $\{X_{k,n}, n \in \mathbb{Z}\}$ and $\{U_{k,n}, n \in \mathbb{Z}\}$ are w.s.s. processes.

**Lemma 2.5.1** *Let $\{X_n, n \in \mathbb{Z}\}$ be a w.s.s. process and let $\{X_{k,n}, n \in \mathbb{Z}\}$, $k = 1, \ldots, K$, be the output of filter $h_k \in l^2(\mathbb{Z})$ given by (2.4). Then, $\{X_{k,n}, n \in \mathbb{Z}\}$ is w.s.s. for all $k = 1, \ldots, K$.*

**Proof:** See [60, 61, 62]. □

**Lemma 2.5.2** *Let $\{X_n, n \in \mathbb{Z}\}$ be a w.s.s. process and let $\{U_n, n \in \mathbb{Z}\}$ be the down-sampled process, i.e., $U_n = X_{mn}$ for all $n \in \mathbb{Z}$ where $m$ is the down-sample factor. Then, $\{U_n, n \in \mathbb{Z}\}$ is w.s.s.*

**Proof:** See [61, 62]. □

Hence, given that $\{X_n, n \in \mathbb{Z}\}$ is w.s.s., the processes $\{U_{k,n}, n \in \mathbb{Z}\}$, $k = 1, \ldots, K$, are w.s.s. as well. This brings us to the following result.

**Theorem 2.5.2** *Let $\{X_n, n \in \mathbb{Z}\}$ be a w.s.s. process and let $\{X_{k,n}, n \in \mathbb{Z}\}$, $k = 1, \ldots, K$, be the output of filter $h_k \in l^2(\mathbb{Z})$ given by (2.4). Then,*

$$E X_{k,n+h} \overline{X_{l,n}} = \frac{1}{2\pi} \int_{-\pi}^{\pi} \hat{h}_k(\xi) \overline{\hat{h}_l(\xi)} e^{i\xi h} S(\xi) d\xi.$$

**Proof:** From Lemma 2.5.1 we conclude that $\{X_{k,n}, n \in \mathbb{Z}\}$ is a w.s.s. process. Hence

$$\begin{aligned}
E X_{k,n+h} \overline{X_{l,n}} &= \sum_{s,t} h_k(s) \overline{h_l(t)} E X_{n+h-s} \overline{X_{n-t}} \\
&= \frac{1}{2\pi} \sum_{s,t} h_k(s) \overline{h_l(t)} \int_{-\pi}^{\pi} e^{i(t-s+h)\xi} S(\xi) d\xi \\
&= \frac{1}{2\pi} \int_{-\pi}^{\pi} \left( \sum_s h_k(s) e^{-is\xi} \right) \left( \sum_t \overline{h_l(t)} e^{it\xi} \right) e^{ih\xi} S(\xi) d\xi \\
&= \frac{1}{2\pi} \int_{-\pi}^{\pi} \hat{h}_k(\xi) \overline{\hat{h}_l(\xi)} e^{ih\xi} S(\xi) d\xi.
\end{aligned}$$

□

**Theorem 2.5.3** *Let* $\{X_n, n \in \mathbb{Z}\}$ *be a w.s.s. process and let* $\{U_{k,n}, n \in \mathbb{Z}\}$, $k = 1, \ldots, K$, *be the down-sampled output of filter* $h_k \in l^2(\mathbb{Z})$. *Then*

$$EU_{k,n+h}\overline{U_{l,n}} = \frac{1}{2\pi}\int_{-\pi}^{\pi} \hat{h_k}(\xi)\overline{\hat{h_l}(\xi)}e^{i\xi mh}S(\xi)d\xi.$$

**Proof:** This follows on directly from the fact that $EU_{k,n+h}\overline{U_{l,n}} = EX_{k,n+mh}\overline{X_{l,n}}$ and Theorem 2.5.2. □

Theorem 2.5.3 shows that the conclusion often drawn that the filter bank channel signals $U_{k,n}$ are mutually uncorrelated in the case of orthogonal analysis filters is incorrect in general. One exception is where $\{X_n, n \in \mathbb{Z}\}$ itself is uncorrelated. In this event we have $S(\xi) = R(0)$ is constant on the interval $(-\pi, \pi]$ so that

$$EU_{k,n+h}\overline{U_{l,n}} = \frac{R(0)}{2\pi}\int_{-\pi}^{\pi} \hat{h_k}(\xi)\overline{\hat{h_l}(\xi)}d\xi,$$

which is zero in the case of orthogonal filters. In practical coding schemes, however, the statistics of the source are generally not known. For any unknown source statistics it is clear that $EU_{k,n+h}\overline{U_{l,n}}$ vanishes if and only if $\hat{h_k}(\xi)\overline{\hat{h_l}(\xi)} = 0$ for all $\xi \in (\pi, \pi]$. In other words, this is only satisfied if the analysis filters are ideal pass/stop filters. We will refer to these kinds of filters as (ideal) brick-wall filters. Although such filters cannot be realized in practice, we conclude that it is important to design our analysis filters such that they have good frequency-discriminating properties. In that case the quantity $\hat{h_k}(\xi)\overline{\hat{h_l}(\xi)}$ is small and so is $EU_{k,n+h}\overline{U_{l,n}}$.

Moreover, where there are spectral lines and we consider arbitrary distribution functions $F$, the proofs given above can easily be modified by considering distributions rather than spectral-density functions. In the more general case we have

$$EU_{k,n+h}\overline{U_{l,n}} = \frac{1}{2\pi}\int_{-\pi}^{\pi} \hat{h_k}(\xi)\overline{\hat{h_l}(\xi)}e^{i\xi mh}dF(\xi),$$

and our conclusion is that it is still important to design our analysis filters such that $\hat{h_k}(\xi)\overline{\hat{h_l}(\xi)}$ is small.

We will illustrate the previous discussion with an example. We have used two source models for the source process to be coded, the spectral densities of which are shown in Figure 2.8. The dashed line corresponds to a zero-mean white Gaussian source whereas the solid line represents a zero-mean non-white, or coloured,

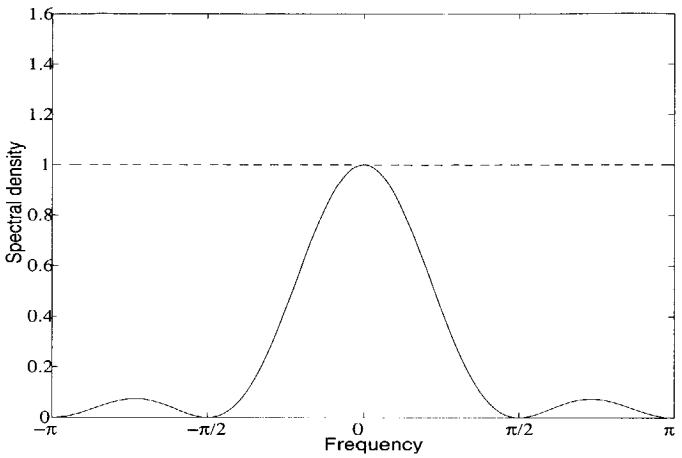



**Figure 2.8:** *Spectral densities of the zero-mean Gaussian sources.*

Gaussian source. Both sources were coded using three different orthogonal filter banks, all three with 8 channels. The filter bank filters were assumed to be the DCT filters, the LOT filters and the brick-wall filters, the frequency responses of which are shown in Figure 2.9, from top to bottom, respectively. The distortion measure we have used is the squared-error distortion measure. It is well known [45, 47, 63] that the rate-distortion function for this distortion criterion has the parametric representation

$$D(\theta) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \min(\theta, S(\xi)) d\xi,$$

and

$$R(\theta) = \frac{1}{4\pi} \int_{-\pi}^{\pi} \max\left(0, \log\left(\frac{S(\xi)}{\theta}\right)\right) d\xi.$$

The non-zero portion of the $R(D)$ curve is generated as the parameter $\theta$ traverses the interval $0 \leq \theta \leq \sup S(\xi)$. Figure 2.10 shows the rate-distortion curves for the white Gaussian source. As discussed above, the three curves coincide, since the source itself is uncorrelated so that $EU_{k,n+h}\overline{U_{l,n}}$ becomes zero for orthogonal filters. Figure 2.11 shows the rate-distortion curves for the coloured Gaussian source. It is clear that the performance increases with increasing frequency-discriminating properties. An important conclusion we can draw from Figure 2.11

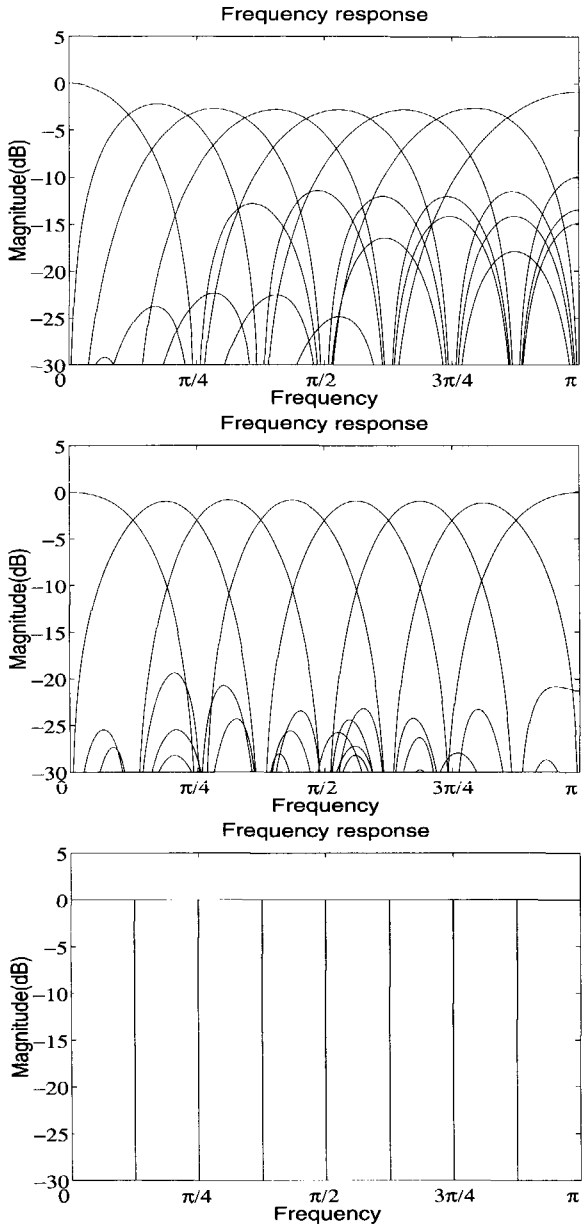



**Figure 2.9:** *Frequency responses of DCT filters, LOT filters and ideal brick-wall filters, from top to bottom.*

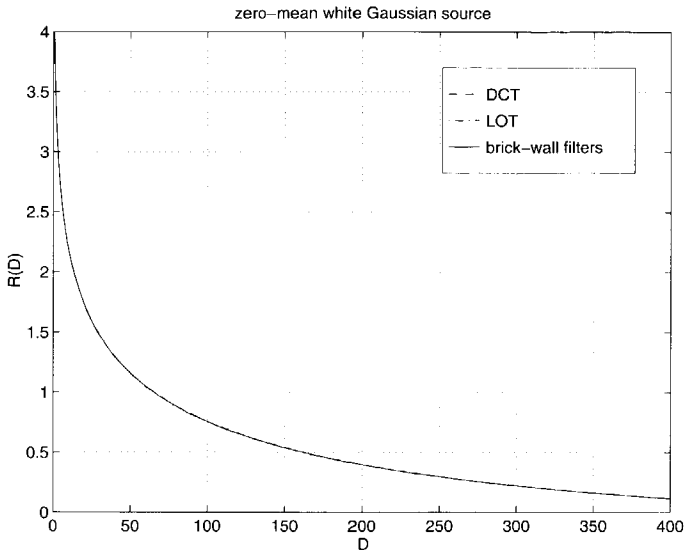



**Figure 2.10:** *Rate-distortion function for the white Gaussian source.*

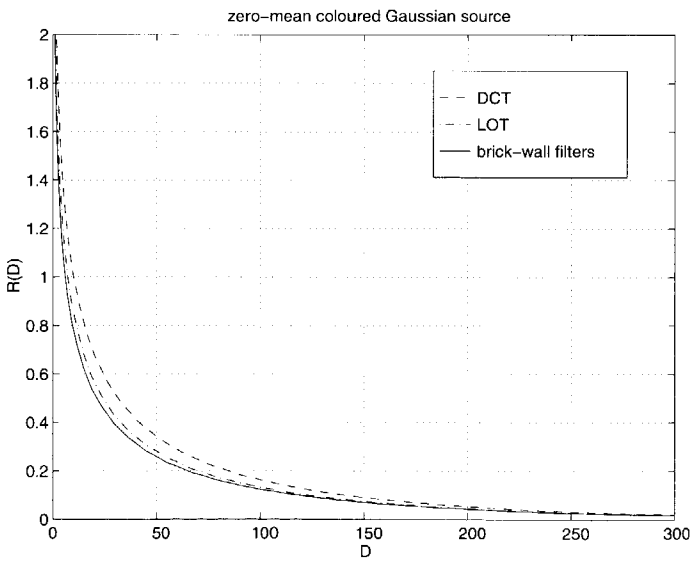



**Figure 2.11:** *Rate-distortion function for the coloured Gaussian source.*

is that the performance of the LOT already approaches the performance of the ideal brick-wall filters very closely, these being optimal in terms of rate-distortion. This means that it is not necessary to use long filter responses in order to obtain very high attenuation (> 20 dB) of the stop bands. Sufficient attenuation of the stop band is obtained with the LOT.

## 2.6 Coding of non-stationary sources

The third and last point concerning the rate-distortion performance of discrete-time signal transforms we are investigating here is how to choose the filter length $L$. Let $\{X_n, n \in \mathbb{Z}\}$ be a second-order Gaussian random process and assume that the transformed process $\{U_n, n \in \mathbb{Z}\}$ is transmitted rather than $\{X_n, n \in \mathbb{Z}\}$ itself. Let $\Phi_x = \left(\phi_{k,l}\right)_{k,l=1,\ldots,N}$ the (non-negative definite) autocorrelation matrix of any succession of $N$ components of the source $\{X_n, n \in \mathbb{Z}\}$, and $\lambda_k$, $k = 1, \ldots, N$, its eigenvalues. Similarly, let $\Phi_u$ be the $N \times N$ autocorrelation matrix of the corresponding transformed process $\{U_n, n \in \mathbb{Z}\}$. The next theorem shows the existence of an analysis map $T_a \in \mathbb{C}^{N \times N}$ and a synthesis map $T_s \in \mathbb{C}^{N \times N}$ satisfying $T_s = T_a^*$ such that $\Phi_u = \Lambda_x = \mathrm{diag}(\lambda_1, \ldots, \lambda_N)$ is the eigenvalue matrix of $\Phi_x$. This also holds where the source $\{X_n, n \in \mathbb{Z}\}$ is non-stationary. The resulting map (*KL-transform*) has $K = L = N$, is data dependent and computationally complex whenever $N$ becomes large, as usually occurs. To overcome this complexity problem, we can reduce $K$ (and $L$) without the rate-distortion performance deteriorating, provided Theorem 2.5.1 is satisfied. This results in a transform which is block-banded, the band consisting of overlapping blocks shifted over $m \leq K$ positions.

**Theorem 2.6.1** *Let $\{X_n, n \in \mathbb{Z}\}$ be a second-order random process with autocorrelation matrix $\Phi_x$ and eigenvalues $\lambda_k$, $k = 1, \ldots, N$. Then, $\Phi_x$ can be decomposed as*

$$\Phi_x = U \Lambda_x U^*,$$

*where $U$ is unitary and $\Lambda_x = \mathrm{diag}(\lambda_1, \ldots, \lambda_N)$.*

**Proof:** Since $\phi_{k,l} = E X_k \overline{X_l} = \langle X_k, X_l \rangle = \overline{\langle X_l, X_k \rangle} = E \overline{X_l} X_k = \overline{\phi_{l,k}}$ we conclude that $\Phi_x$ is selfadjoint and thus $\Phi_x = U \Lambda_x U^*$ where $U$ is unitary and $\Lambda_x$ is the eigenvalue matrix of $\Phi_x$. □

Next, let us assume that the source $\{X_n, n \in \mathbb{Z}\}$ is piecewise stationary, satisfying

$$\Phi_x = \mathrm{diag}(\Phi_{x,1}, \ldots, \Phi_{x,l}), \qquad \Phi_{x,j} \in \mathbb{C}^{s_j \times s_j}, \tag{2.5}$$

i.e., the piecewise stationary parts of $\{X_n, n \in \mathbb{Z}\}$ are mutually independent. Obviously, we have $\sum_{j=1}^{l} s_j = N$. This brings us to the following result.

**Theorem 2.6.2** *Let $\{X_n, n \in \mathbb{Z}\}$ be a second-order piecewise stationary process with autocorrelation matrix $\Phi_x$ as given by (2.5). Then, $\Phi_x = U \Lambda_x U^*$ with*

$$U = \text{diag}(U_1, \ldots, U_l), \qquad U_j \in \mathbb{C}^{s_j \times s_j}.$$

**Proof:** Since $\Phi_x$ is an autocorrelation matrix of a second-order random process, $\Phi_x$ is non-negative definite. As a consequence any principal submatrix of $\Phi_x$, say $\Phi_{x,j} \in \mathbb{C}^{s_j \times s_j}$, is non-negative definite and from Theorem 2.6.1 we conclude that there exists a unitary matrix $U_j \in \mathbb{C}^{s_j \times s_j}$ such that $U_j^* \Phi_{x,j} U_j = \Lambda_{x,j}$, the eigenvalue matrix of $\Phi_{x,j}$. Therefore, we have

$$\begin{aligned} U \Lambda_x U^* &= \Phi_x \\ &= \text{diag}(\Phi_{x,1}, \ldots, \Phi_{x,l}) \\ &= \text{diag}(U_1 \Lambda_{x,1} U_1^*, \ldots, U_l \Lambda_{x,l} U_l^*) \\ &= \text{diag}(U_1, \ldots, U_l)\text{diag}(\Lambda_{x,1}, \ldots, \Lambda_{x,l})\text{diag}(U_1^*, \ldots, U_l^*) \\ &= \text{diag}(U_1, \ldots, U_l) \Lambda_x \text{diag}(U_1, \ldots, U_l)^* \end{aligned}$$

□

Theorem 2.6.2 asserts that the optimal signal transform for a random process with autocorrelation matrix $\Phi_x$ as given by (2.5) is of the form

$$T_s = T_a^* = \text{diag}(T_1, \ldots, T_l), \qquad T_j \in \mathbb{C}^{s_j \times s_j},$$

satisfying $T_j \Phi_{x,j} T_j^* = \Lambda_{x,j}$, the eigenvalue matrix of $\Phi_{x,j}$. This means that the big orthogonal transform becomes a block-diagonal transform with orthogonal blocks $T_k \neq T_l$ for $k \neq l$, and $K_j = L_j = s_j$, $j = 1, \ldots, l$. It also means that the filter length of the optimal transform becomes larger as the degree of stationarity of the source process increases and becomes smaller as the degree of stationarity decreases. It should be noted that this does not depend on the correlation between the samples to be transformed. Indeed, if the source is completely stationary but nearly uncorrelated, we have $l = 1$ and the filter length becomes $s_1 = N$ as shown before. On the other hand, if the source is highly correlated but the length of the data segments on which this highly correlated signal is stationary is very small, i.e. $s_j \ll N$ for all $j$, the filter length should be short, namely $s_j$. Again, when $s_j$ becomes too large for some $j$, similar solutions can be found as discussed above, and the $T_j$ can be replaced with a "banded" block.

As mentioned in Section 2.1, in nearly all practical coding schemes the transform used is signal-independent and time-invariant ($K$, $L$ and $m$ constant). That is, we have $T_j = T \in \mathbb{C}^{s \times s}$ for all $j = 1, \ldots, l$. In that case we should choose a filter length which is proportional to the average degree of stationarity. This means that relatively stationary processes require longer filter lengths than processes where the statistics vary quite rapidly. We will end this section by illustrating the previous discussion with an example.

### Practical coding example

The source sequence we used in this experiment is the image depicted in Figure 2.12, which is a single frame (only the Y-component) of the "mobile-calendar" sequence. The size of the image is 576 lines with 720 pixels each, where each pixel is represented by 8 bits. We used a time-invariant signal transform in which the blocks $T_j = T = \mathbb{C}^{s \times s}$ for all $j = 1, \ldots, l$ were taken to be the DCT. The coding was carried out using a simple uniform quantizer and Huffman coder. The distortion measure we used was the squared-error distortion function. The coding experiment was performed for four different filter lengths, viz. $s = 4, 8, 16$ and 32. Figure 2.13 shows the resulting rate-distortion curves. We can see that the optimal filter length for the "mobile-calendar" image is $s = 8$ which corresponds to the lower curve. If we either increase or decrease the filter length to $s = 16$ or $s = 4$, respectively, the rate needed to reproduce the image with some fixed distortion $D$ will increase in both cases.

Next, consider the two images shown in Figure 2.14 and Figure 2.16. Figure 2.14 is a sub-image of Figure 2.12, $32 \times 32$ pixels in size and is part of the sheep. We will refer to this image as the "sheep" image. Figure 2.16 is also a sub-image of Figure 2.12, $32 \times 32$ pixels in size but is part of the goat and will henceforth be referred to as the "goat" image. It is obvious that the "sheep" image is a very low correlated but stationary image while the "goat" image is a very high correlated stationary image. Figure 2.15 shows the rate-distortion curves for the "sheep" image and Figure 2.17 shows the rate-distortion curves for the "goat" image. The filter lengths used in these experiments were $s = 8, 16$ and 32. The figures show that, for stationary sources, the coding performance increases when filter length is increased, irrespective of the correlation between the samples to be coded. Although the "sheep" image is considerably more difficult to code (requiring a higher bit rate for a given distortion) than the "goat" image, the optimal filter length in both cases is the same as the dimension of the image, namely $s = 32$. It should be noted that while the "sheep" image is much more difficult to code than

the complete "mobile-calendar" image (in terms of bits per sample), it still needs a larger filter length. Based on the foregoing discussion we can conclude that the filter length should be chosen such that it corresponds to the degree of stationarity in the source process and is independent of the correlation between samples.

## 2.7 Conclusions

We have investigated the rate-distortion function of a (not necessarily stationary) stochastic process where a discrete-time signal transform is used in the coder. The main results can be summarized as follows.

- Given a fixed bit rate, oversampling does not reduce the reconstruction error. The effective rate at which the source $\{X_n, n \in \mathbb{Z}\}$ generates information when a distortion $d(x, y)$ less than or equal to $D$ is required with respect to some fidelity criterion decreases linearly with the oversample ratio $\mu$. However, since the total number of samples needed to represent the transformed process $\{U_n, n \in \mathbb{Z}\}$ increases linearly with the oversample ratio, the total rate needed to store or transmit the source process will remain the same, independent of the value of $\mu$.

- Where we use non-critically sampled filter banks, the source encoder should be able to remove the redundancy introduced by the oversampling. This inevitably implies an increase in encoder complexity compared with the critically sampled situation. Moreover, oversampling increases the sample rate in the channel, which can cause severe problems in real-time data-compression applications.

- If the filter bank channels are coded separately, we can reach the rate-distortion bound $R_x(D)$ if and only if the filter bank channel signals are statistically independent. In practical coding systems this requirement will not generally be met. In order to minimize the loss introduced by the independent encoding of the filter bank channel signals, the analysis filters should be designed such that they have sufficient frequency-discriminating properties, i.e., the suppression of the side lobe amplitudes must be sufficient. As an example we showed that an attenuation of approximately 20 dB of the first side lobe is already sufficient to reach the rate-distortion bound very closely.

**Figure 2.12:** *The "mobile-calendar" image.*

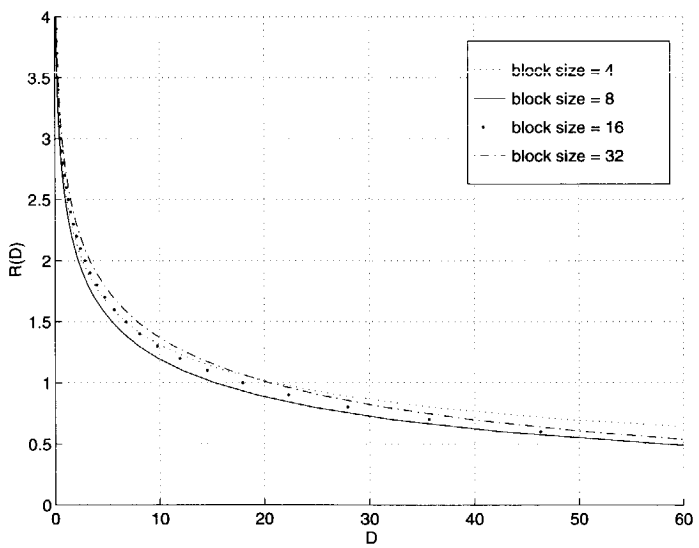



**Figure 2.13:** *Rate-distortion curves for the "mobile-calendar" image.*

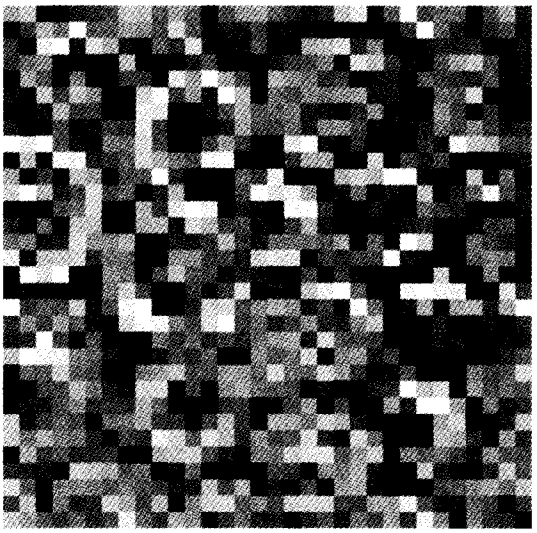


**Figure 2.14:** *The "sheep" image.*

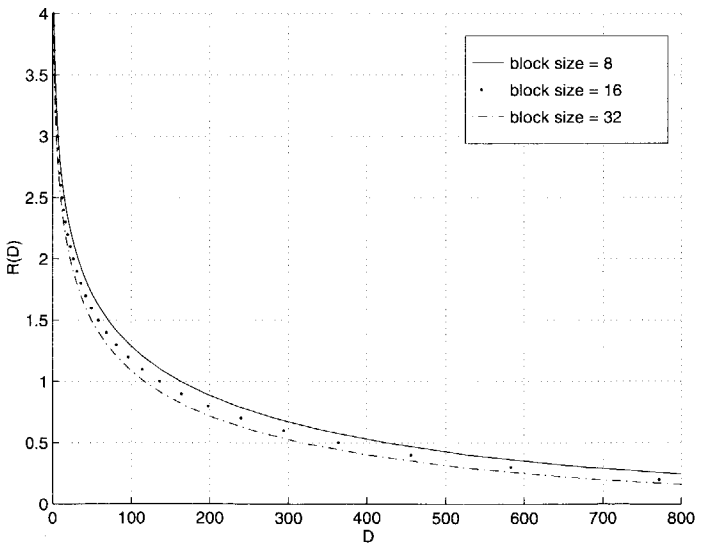



**Figure 2.15:** *Rate-distortion curves for the "sheep" image.*

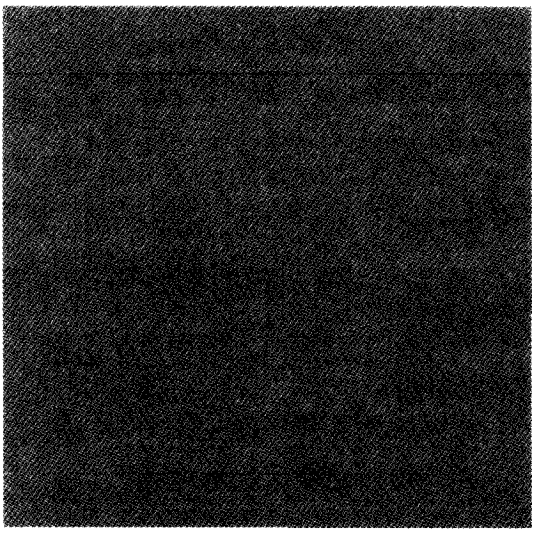


**Figure 2.16:** *The "goat" image.*

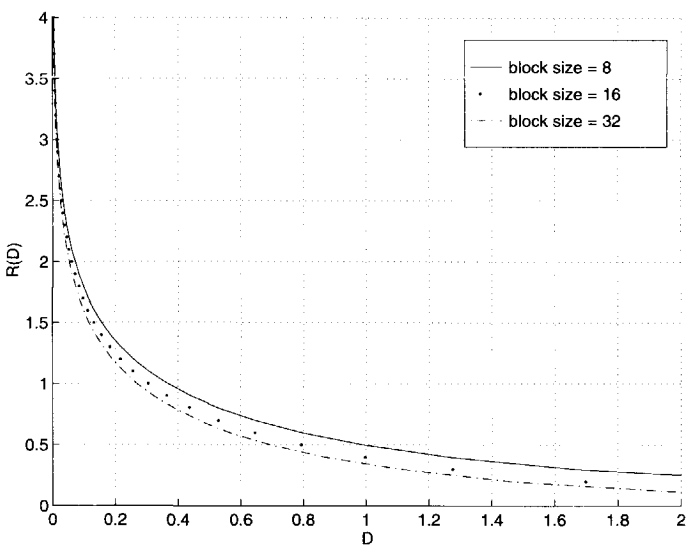



**Figure 2.17:** *Rate-distortion curves for the "goat" image.*

- If we assume the source process $\{X_n, n \in \mathbb{Z}\}$ to be Gaussian and piecewise stationary, we showed that the optimal discrete-time signal transform corresponds to a unitary time-varying filter bank. Moreover, we demonstrated that the filter length should be chosen such that it corresponds to the degree of stationarity in the source process and is independent of the correlation between samples. This means that relatively stationary processes require longer filter lengths than processes whose statistics vary quite rapidly. This property can be used as a basis for segmentation algorithms used in region-based coding systems, i.e. in coding systems based on time-varying filter banks [64, 65]. When a time-invariant transform is used, this property implies that the fixed filter length should correspond to the average degree of stationarity.

# Chapter 3

# Design of lapped orthogonal transforms

**Contents**



## 3.1 Introduction

In the previous chapter we applied rate-distortion theory to transform-coding systems in order to investigate the influence of different parameter settings of the filter bank on the coding efficiency. In this chapter, using the results of the previous chapter, we design analysis and synthesis filters for data compression of images.

In traditional transform coding, such as JPEG and MPEG [5, 7, 9], non-overlapped orthogonal transforms are commonly used. This is done by mapping the input sequence $x$ onto an output sequence $u$ using an orthogonal, block-diagonal Toeplitz operator, say $T_a$, i.e. $u$ is given by

$$u = T_a x = \begin{bmatrix} \ddots & & & \\ & A_1 & & \\ & & A_1 & \\ & & & \ddots \end{bmatrix} x,$$

where $A_1 \in \mathbb{C}^{K \times K}$ of full rank $K$. The term non-overlapped refers to the fact that the blocks on the main diagonal of $T_a$ are non-overlapping. Since $T_a$ has a block-diagonal Toeplitz structure, the corresponding filter bank is time-invariant. In general, the sequence $u$ is quantized, coded, transmitted and/or stored and reconstructed before being inversely-mapped by the synthesis operator $T_s$. The resulting sequence $y$, therefore, will generally be an approximation of the input sequence $x$. If $T_s T_a = I$, hence $y = x$, we say that $y$ is a perfect reconstruction (PR) of the input sequence $x$.

A common example of a non-overlapped orthogonal transform is the discrete cosine transform (DCT) [4, 26, 35, 51]. It is well known that the disadvantage of non-overlapped transforms is that "blocking artifacts" are introduced at low bit rates. This can be seen as follows. The effect of quantization can be regarded as adding an additional noise signal, say $n$, to the sequence $u$. Hence, the reconstruction $y$ becomes $y = T_s(u + n) = x + \varepsilon$, i.e. $y$ can be written as $x$, a perfect replica of the input signal, and an error term $\varepsilon$, which is a linear combination of the columns of $T_s$. We shall refer to these columns as the basis functions of the signal transform.

With non-overlapped transforms, the basis functions change abruptly at the endpoints of their support, which causes equally spaced discontinuities in the error term $\varepsilon$. For example, Figure 3.1 shows the first even-symmetric and odd-symmetric DCT basis functions. The result of reconstructing a ramp-function with only these two basis functions, using $K = 32$, is shown in Figure 3.2. The clearly visible discontinuities are generally referred to as blocking artifacts. To avoid these artifacts, we must choose basis functions without abrupt changes that tend to zero smoothly at the endpoints of their supports. However, this is not possible with non-overlapped transforms when the PR condition has to be maintained at the same time.

A class of transforms which can eliminate the blocking artifacts is the class of 50% overlapped transforms [10, 11, 36, 66]. In this case, the analysis map $T_a$ has an upper-triangular, block-banded Toeplitz structure, where the non-zero parts of the basis functions overlap one another by 50%. The orthogonal transforms of this sub-class are collectively referred to as the lapped orthogonal transform (LOT). In [36, 10, 11] it was shown that the LOT can be constructed using a DCT. However, the basis functions thus obtained still have a certain amount of discontinuity, which will also cause blocking artifacts. In [67] a solution is proposed for overcoming this problem. The discontinuities can be eliminated by using an additional scaling of $\sqrt{2}$ of the first DCT basis function. The signal transform thus obtained

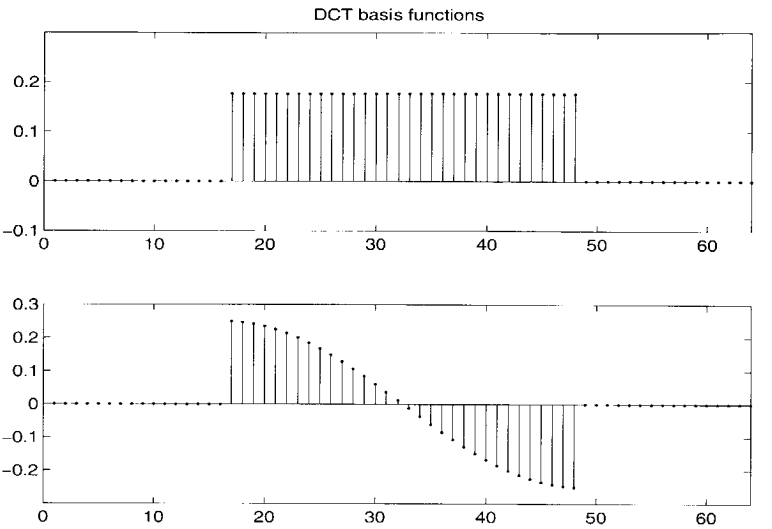



**Figure 3.1:** *First even and odd-symmetric DCT basis functions.*

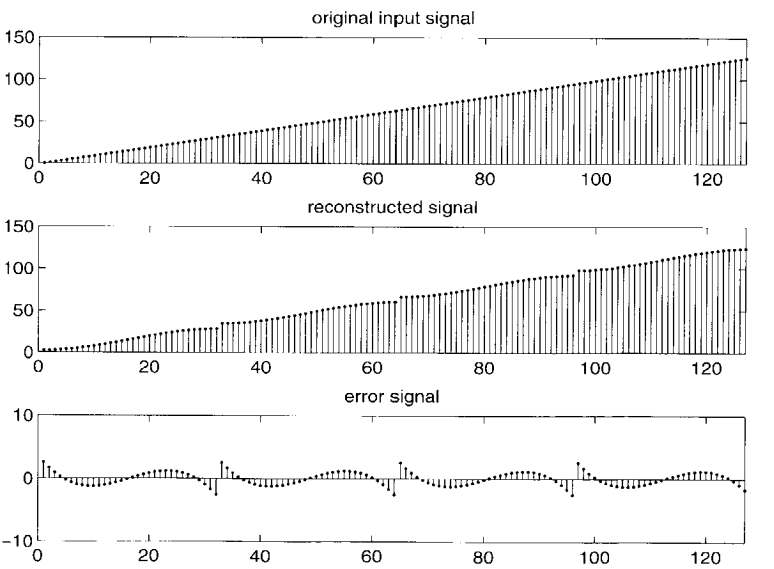



**Figure 3.2:** *Example of a reconstruction with the first two DCT basis functions.*

is usually referred to as the modified lapped transform (MLT)[1]. However, this solution leads to non-orthogonal transforms which is a disadvantage in terms of implementation.

In this chapter we design LOTs which do not introduce blocking artifacts. These LOTs, however, cannot be constructed with DCTs. Figure 3.3 shows the first even and odd-symmetric LOT basis functions as an example of a perceptually relevant LOT. These functions are smooth and decay to zero at the endpoints of their support and, therefore, cannot cause blocking artifacts. The result of reconstructing a ramp-function with these two basis functions, again using $K = 32$, is shown in Figure 3.4. From Figure 3.2 and Figure 3.4, we conclude that it is indeed possible to eliminate the blocking artifacts completely.

### Organization of this chapter

The remainder of this chapter is organized as follows. Since the aim of this chapter is to design analysis and synthesis filters for data compression of images, it would be useful to start by discussing those factors which influence the choice of an architecture (choice of $K$, $L$, $m$) and the sets $\mathcal{A}$ and $\mathcal{S}$. These factors are, for example, coding efficiency, perceptual quality, applying post-processing to the reconstructed images and implementation considerations. This is done in Section 3.2. Based on this discussion, we give a list of desirable constraints. This leads to a suitable filter bank architecture. Given this architecture, we derive some algebraic properties of LOTs in Section 3.3, which can be used to significantly simplify the design of the sets $\mathcal{A}$ and $\mathcal{S}$. In Section 3.4 we concentrate on the choice of the constituent filters. We show how we can design proper analysis and synthesis filters banks, i.e., filter banks in which the filters satisfy all the constraints referred to in Section 3.2. Finally, in Section 3.5, we draw some conclusions.

## 3.2 Constraints on the choice of filter bank

In this section we give a list of constraints on the choice of the filters, imposed by the application of image compression. Based on this we select a suitable filter bank architecture (choice of $K$, $L$, $m$). The constraints can roughly be divided

[1] In [11, 68, 69] the term MLT is used as an abbreviation for modulated lapped transforms. As these transforms are better known as cosine-modulated filter banks [70, 71, 72], we will use the term MLT only for the modified lapped transform.

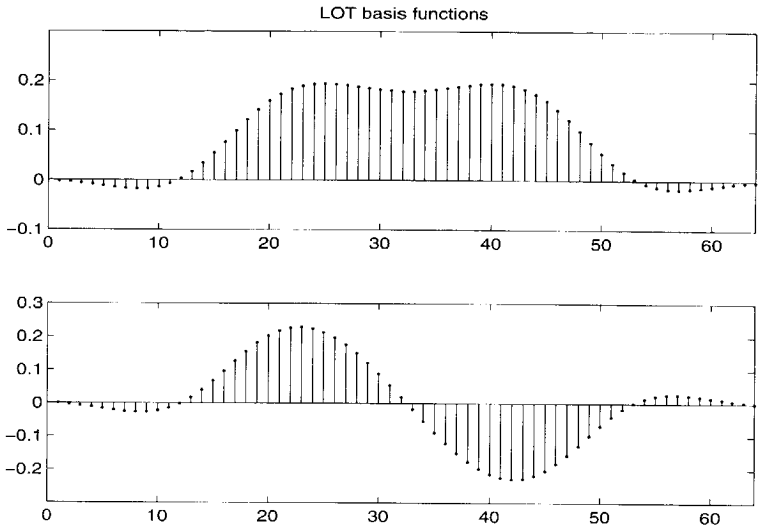



**Figure 3.3:** *First even and odd-symmetric LOT basis functions.*

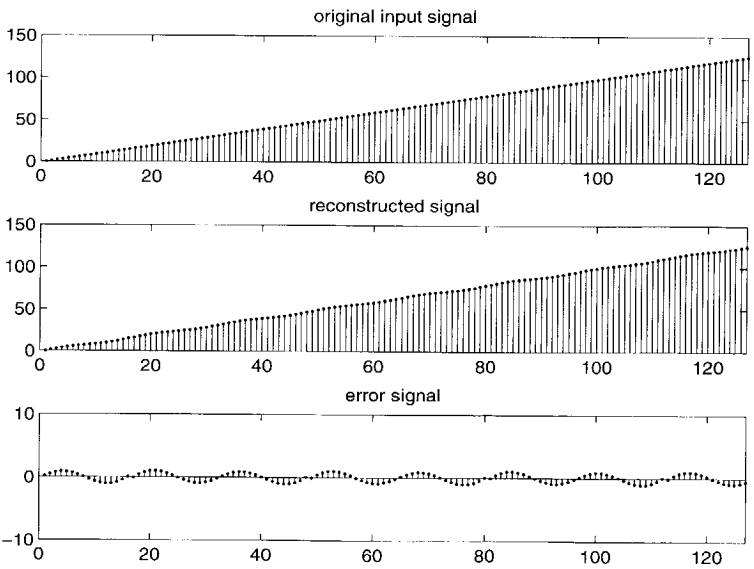



**Figure 3.4:** *Example of a reconstruction with the first two LOT basis functions.*

into three categories: coding-efficiency, perception and implementation-related constraints.

### Coding-efficiency-related constraints

As discussed in Section 2.4, oversampling does not reduce the reconstruction error at a fixed bit rate. In fact, oversampling leads to complex source encoders and can give severe problems in real-time implementations of the signal transforms. For these reasons, we use critically sampled rather than non-critically sampled filter banks ($m = K$). Moreover, we showed that $T_s T_a = I$ is a sufficient (but not necessary) condition to reach the rate-distortion bound for *any* distortion $d(x, y) = D$. If we choose $T_s T_a \neq I$, however, we have to increase the complexity of the source encoder. This can be most easily seen by considering the reconstructability of the input sequence $x$ in the absence of a quantizer. Also, as we showed in Section 2.5, we can reduce the coder complexity where the analysis filters have good frequency discriminating properties.

From a coding point of view it is also very desirable to have both analysis and synthesis filters with polynomial transfer functions which all have a zero in the complex $z$ plane at $z = 1$, except for the low-pass filter. In that case we can represent DC information with the aid of only one coefficient, which will lead to the most efficient representation of flat backgrounds, for example. To summarize, we have

$$
\begin{aligned}
\text{coder complexity} &\Rightarrow \text{critical sampling } (m = K), \\
&\Rightarrow \text{perfect reconstruction } (T_s T_a = I), \\
&\Rightarrow \text{good frequency discriminating properties of the analysis filters,} \\
\text{coding efficiency} &\Rightarrow \text{zeros at } z = 1, \text{ except for low-pass filters.}
\end{aligned}
$$

### Perception-related constraints

A desirable property in image coding applications is that the overall transfer function $T(z) = Y(z)/X(z)$ is a monomial in $z$, i.e. $T(z) = z^k$ for some $k \in \mathbb{Z}$. This means that the overall (group) delay is equal to $k$. Moreover, since the contrast sensitivity function of the human visual system is roughly isotropic (rotation-invariant) [73, 74] and the visual sensitivity of human observers decreases symmetrically at and on both sides of luminance changes [75], a desirable property of the synthesis filters is that they have symmetry. This means that not only the

overall transfer function $T(z)$ should have the linear-phase property, but the constituent synthesis filters as well.

An important requirement with respect to the type of filter to be used is that blocking artifacts must be avoided. Therefore, the filter responses should be smooth and decay to zero at the endpoints of their supports. Thus, in the case of a PR analysis/synthesis system this implies $l \geq 2$. However, long filter responses usually give rise to ringing effects and, therefore, must be limited. This implies short synthesis filters.

Finally, in almost all coding systems a noise-shaping quantizer is used. This can be done by combining the quantizer with a filter which feeds back the error to the quantizer's input [76, 77, 78]. A disadvantage of noise-shaping quantizers is that they are relatively complex compared with ordinary quantizers, and the quantizer and filter must be chosen carefully to avoid instabilities caused, for example, by overload errors in the quantizer. In transform coding systems, however, we can do similar things by using different step sizes for the different filter bank channels. This relatively simple method results in a spectral shaping of the coding error and, therefore, can be seen as another form of noise shaping. This methods can only be used properly if $K$ is sufficiently large and the constituent filters have good frequency discriminating properties. To summarize, we have

| | |
|---|---|
| no phase errors | $\Rightarrow$ linear-phase overall transfer function, |
| symmetric sensitivity | $\Rightarrow$ linear-phase synthesis filters, |
| no blocking artifacts | $\Rightarrow$ smooth decaying to zero synthesis impulse responses, |
| no ringing effect | $\Rightarrow$ short synthesis filters, |
| simple noise shaping | $\Rightarrow$ $K$ sufficiently large. |

### Implementation-related constraints

Error blow-up is minimal when $(\mathcal{A}, \mathcal{S})$ is para-unitary, i.e. the impulse responses $h_n$ together with their shifted versions (over $m$ samples) must form an orthonormal set and the same holds for $f_n$s. Moreover, para-unitary filter banks have the nice property of having the same structure and entries for both the analysis and synthesis filter bank, i.e., $T_s = T_a^*$. If desired, we can exploit this property by implementing both the analysis and synthesis filter bank with the same hardware. Especially, in applications where the analysis and synthesis mapping is not computed concurrently, this solution saves 50% of the silicon area. Finally, in order

to minimize the sample rate in the filter bank, and thereby minimize the computational complexity, we down-sample the filter outputs $x_k$ to their Nyquist rate. To summarize, we have

| | |
|---|---|
| minimal error blow-up | $\Rightarrow$ orthonormality, |
| $\mathcal{A}$ and $\mathcal{S}$ the same structure | $\Rightarrow$ para-unitary, |
| minimum sample rate | $\Rightarrow$ critical sampling ($m = K$). |

Apart from the final choice of the constituent filters to be used, we can conclude from what has been said that an architecture in which $m = K$ (critical sampling) and $L = 2K$ ($l = 2$, a 50% overlapped transform) could be very suitable for data compression of images. Such an architecture would enable us to reduce the blocking effects and at the same time limit ringing effects due to long filter responses. Therefore, for the remainder of this chapter we shall assume that $m = K$ and $L = 2K$. The synthesis filters have symmetry (linear-phase filters) and the overall transfer function $T(z)$ must be a monomial in $z$. Adding the para-unitary requirement to this architecture, we conclude that the LOT is suitable for our specific application.

In the next section we shall derive some algebraic properties of LOTs which can be used to simplify the design procedure considerably.

## 3.3 Lapped orthogonal transforms

As stated above, we shall assume that $m = K$ (critical sampling), $L = 2K$ (a 50% overlapped transform) and the synthesis filters have the linear-phase property. In this section we will derive some algebraic properties of LOTs which can be used to design suitable filters.

We arrange the analysis filters and the synthesis filters in $K \times 2K$ matrices

$$A = [h_k(L + 1 - n)]_{k=1,\ldots,K,\ n=1,\ldots,L} = [A_1\ A_2],$$

and

$$S = [f_k(n)]_{k=1,\ldots,K,\ n=1,\ldots,L} = [S_1\ S_2],$$

where $A_j$, $S_j$, $j = 1, 2$, are matrices of size $K \times K$. The corresponding transform is performed by a map $T_a$ which has an upper-triangular block-banded Toeplitz

structure, where the non-zero parts of the basis functions overlap one another by 50%. Thus, $T_a$ is given by

$$T_a = \begin{bmatrix} \ddots & \ddots & & & & \\ & A_1 & A_2 & & & \\ & & A_1 & A_2 & & \\ & & & A_1 & \ddots & \\ & & & & \ddots & \end{bmatrix},$$

and similarly for $T_s$ it holds that

$$T_s = \begin{bmatrix} \ddots & & & & & \\ \ddots & S_1^t & & & & \\ & S_2^t & S_1^t & & & \\ & & S_2^t & S_1^t & & \\ & & & \ddots & \ddots & \end{bmatrix}.$$

Perfect reconstruction is obtained if the following necessary and sufficient conditions are satisfied (see [43])

$$\sum_{l=1}^{2} S_l^t A_{l-k} = \delta_{ko} I, \qquad k = -1, \ldots, 1, \ 1 \leq l - k \leq 2, \tag{3.1}$$

or equivalently with critical sampling

$$\sum_{l=1}^{2} A_l^t S_{l-k} = \delta_{ko} I, \qquad k = -1, \ldots, 1, \ 1 \leq l - k \leq 2, \tag{3.2}$$

where $\delta_{ko}$ denotes the Kronecker delta.

In this section we show that if the filter bank is PR, the filter bank is para-unitary if and only if $(\mathcal{A}, \mathcal{S})$ is restricted orthonormal, i.e. the sets $\mathcal{A}$ and $\mathcal{S}$ themselves are orthonormal. The proof can be divided into two parts. In section 3.3.1 we elaborate the PR conditions (3.1) and (3.2) together with the linear-phase synthesis conditions. In section 3.3.2 we show that the resulting conditions imply that $(\mathcal{A}, \mathcal{S})$ is para-unitary if and only if $(\mathcal{A}, \mathcal{S})$ is restricted orthonormal. As a consequence, we can concentrate on the sets $\mathcal{A}$ and $\mathcal{S}$ only, and we do not have to consider the orthonormality of shifted versions of the filter responses, which greatly simplifies the filter design.

### 3.3.1 PR and linear-phase synthesis filters

**Theorem 3.3.1** *Let $(\mathcal{A}, \mathcal{S})$ be a PR filter bank with even length filters. The analysis bank has linear-phase filters if and only if the synthesis bank has linear-phase filters. Moreover, half of the synthesis filters have even symmetry, i.e. $f_k(L+1-n) = f_k(n)$ and half of the synthesis filter have odd symmetry, i.e. $f_k(L+1-n) = -f_k(n)$ and the same applies to the analysis filters.*

**Proof:** see [79, 80] □

Let $J$ denote the exchange matrix in which all entries are zero, except for the main anti-diagonal entries, which are all one. We partition $A$ and $S$ as

$$A = \begin{bmatrix} A_{1,1} & A_{1,1}J \\ A_{2,1} & -A_{2,1}J \end{bmatrix},$$

and

$$S = \begin{bmatrix} S_{1,1} & S_{1,1}J \\ S_{2,1} & -S_{2,1}J \end{bmatrix},$$

where $A_{k,1}, S_{k,1} \in \mathbb{C}^{\frac{K}{2} \times K}$, $k = 1, 2$. Hence, we have arranged the even-symmetric filters in the first $\frac{K}{2}$ rows and the odd-symmetric filters in the last $\frac{K}{2}$ rows. Using these decompositions, (3.1) becomes

$$S_{1,1}^t A_{1,1} + J S_{2,1}^t A_{2,1} J = \tfrac{1}{2} I, \tag{3.3}$$

$$S_{1,1}^t A_{1,1} = S_{2,1}^t A_{2,1}. \tag{3.4}$$

We now can derive some useful properties.

**Lemma 3.3.1** $\mathrm{ran}(A_{k,1}) = \mathrm{ran}(S_{k,1}) = \frac{1}{2}K$ *for* $k = 1, 2$.

**Proof:** From (3.3) we conclude $\frac{1}{2}K \leq \mathrm{ran}(S_{1,1}^t A_{1,1}) \leq \mathrm{ran}(A_{1,1}) \leq \frac{1}{2}K$ and $\mathrm{ran}(A_{1,1}) = \frac{1}{2}K$. The proof for $A_{2,1}$, $S_{1,1}$ and $S_{2,1}$ is similar. □

Since both $A_{1,1}$ and $A_{2,1}$ are of full rank we conclude that $A_{2,1} = B_a A_{1,1}$ with $B_a$ is non-singular. Similarly, we have $S_{2,1} = B_s S_{1,1}$. Substituting this result in (3.4) yields $S_{1,1}^t A_{1,1} = S_{1,1}^t B_s^t B_a A_{1,1}$ from which we conclude that $B_s^t = B_a^{-1}$ since both $A_{1,1}$ are $S_{1,1}$ are of full rank by Lemma 3.3.1.

Let $A_e = \frac{1}{2}(A_{1,1} + A_{1,1}J)$ and $A_o = \frac{1}{2}(A_{1,1} - A_{1,1}J)$ so that $A_e J = A_e$ and $A_o J = -A_o$. It then follows that $A$ and $S$ take the following form

$$A = \begin{bmatrix} I & \\ & B_a \end{bmatrix} \begin{bmatrix} A_e + A_o & (A_e + A_o)J \\ A_e + A_o & -(A_e + A_o)J \end{bmatrix}, \tag{3.5}$$

and

$$S = \begin{bmatrix} I & \\ & B_a^{-t} \end{bmatrix} \begin{bmatrix} S_e + S_o & (S_e + S_o)J \\ S_e + S_o & -(S_e + S_o)J \end{bmatrix}. \tag{3.6}$$

Note that a similar expression was postulated in [10] for a LOT. However, it turns out that the conditions of PR, linear-phase synthesis filters and 50%-overlapped transform already imply the signal transform to be characterized by (3.5) and (3.6).

Since $A_e J = A_e$ and $A_o J = -A_o$ we conclude that $A_e A_o^* = (A_e J)(A_o J)^* = -A_e A_o^* = O$. Using the PR conditions (3.2), which only hold of a critically sampled $(\mathcal{A}, \mathcal{S})$, it follows that $A_{1,1} S_{1,1}^t = \frac{1}{2} I$ and $A_{1,1} J S_{1,1}^t = O$ or equivalently

$$A_e S_e^t = A_o S_o^t = \tfrac{1}{4} I, \tag{3.7}$$

and thus $\operatorname{ran}(A_e) = \operatorname{ran}(A_o) = \operatorname{ran}(S_e) = \operatorname{ran}(S_o) = \frac{1}{2} K$.

### 3.3.2 Orthonormality

**Definition 3.3.1** *Let $Z$ be the right (or causal) bilateral shift operator, defined by $Za = Z(\alpha_k)_{k=-\infty}^{\infty} = (\alpha_{k-1})_{k=-\infty}^{\infty}$. Then the set $\mathcal{X} = (x_k)_{k=1}^{K}$ is said to be restricted orthonormal if $\mathcal{X}$ satisfies*

$$\langle x_m, x_n \rangle = \delta_{mn} \quad \textit{for all} \quad m, n = 1, \ldots, K,$$

*and (fully) orthonormal if $\mathcal{X}$ satisfies*

$$\langle x_m, Z^{kK} x_n \rangle = \delta_{mn} \delta_{ko} \quad \textit{for all} \quad m, n = 1, \ldots, K, \quad \textit{and all} \quad k \in \mathbb{Z}.$$

*A $K$-channel $(\mathcal{A}, \mathcal{S})$ filter bank is said to be restricted orthonormal or fully orthonormal if both the sets $\mathcal{A}$ and $\mathcal{S}$ are restricted orthonormal or fully orthonormal, respectively.*

Considering $AA^*$ and $SS^*$, we have

$$AA^* = 2 \begin{bmatrix} I & \\ & B_a \end{bmatrix} \begin{bmatrix} A_{1,1} A_{1,1}^* & \\ & A_{1,1} A_{1,1}^* \end{bmatrix} \begin{bmatrix} I & \\ & B_a^* \end{bmatrix},$$

and

$$SS^* = 2 \begin{bmatrix} I & \\ & B_a^{-t} \end{bmatrix} \begin{bmatrix} S_{1,1} S_{1,1}^* & \\ & S_{1,1} S_{1,1}^* \end{bmatrix} \begin{bmatrix} I & \\ & \overline{B}_a^{-1} \end{bmatrix}.$$

Hence, we conclude that restricted orthonormality of $(\mathcal{A}, \mathcal{S})$ implies

$$A_{1,1}A_{1,1}^* = A_eA_e^* + A_oA_o^* = \tfrac{1}{2}I, \tag{3.8}$$

$$S_{1,1}S_{1,1}^* = S_eS_e^* + S_oS_o^* = \tfrac{1}{2}I, \tag{3.9}$$

and that $B_a$ is unitary. Before proving the main theorem of this chapter, we need the following lemma.

**Lemma 3.3.2** *If $P, Q \in \mathbb{C}^{\frac{K}{2}\times K}$ are even-symmetric, i.e. $PJ = P$ and $QJ = Q$ satisfying $A_eP^* = A_eQ^*$, then $P = Q$.*

**Proof:** Since $A_eA_o^* = O$ we conclude from $A_e(P-Q)^* = O$ that $P-Q = TA_o$ for some $T \in \mathbb{C}^{\frac{K}{2}\times\frac{K}{2}}$. Therefore we have $P - Q = (P - Q)J = TA_oJ = -TA_o = -(P - Q) = O$ which completes the proof. □

**Theorem 3.3.2** *Let $(\mathcal{A}, \mathcal{S})$ be a PR critically sampled linear-phase filter bank. Then $(\mathcal{A}, \mathcal{S})$ is fully orthonormal if and only if $(\mathcal{A}, \mathcal{S})$ is restricted orthonormal.*

**Proof:** Assume that $(\mathcal{A}, \mathcal{S})$ is fully orthonormal, thus para-unitary. Then $S = \overline{A}$, hence $S_e = \overline{A}_e$ and $S_o = \overline{A}_o$ so that (3.7) becomes $A_eA_e^* = A_oA_o^* = \frac{1}{4}I$.

Conversely, taking (3.7) and the fact that $A_eA_e^*$ is non-singular, we have

$$\begin{aligned} A_eS_e^t &= \tfrac{1}{4}I \\ &= \tfrac{1}{4}(A_eA_e^*)(A_eA_e^*)^{-1}. \end{aligned}$$

Let $P = \overline{S}_e$ and $Q = \frac{1}{4}(A_eA_e^*)^{-1}A_e$. Then $PJ = \overline{S}_eJ = \overline{S}_e = P$ and $QJ = \frac{1}{4}(A_eA_e^*)^{-1}A_eJ = \frac{1}{4}(A_eA_e^*)^{-1}A_e = Q$. Therefore we conclude from Lemma 3.3.2 that $S_e^t = \frac{1}{4}A_e^*(A_eA_e^*)^{-1}$ so that

$$\begin{aligned} \overline{S_eS_e^*} &= (S_e^t)^*S_e^t \\ &= \tfrac{1}{16}(A_eA_e^*)^{-1}A_eA_e^*(A_eA_e^*)^{-1} \\ &= \tfrac{1}{16}(A_eA_e^*)^{-1}. \end{aligned}$$

Similarly, we have $\overline{S_oS_o^*} = \frac{1}{16}(A_oA_o^*)^{-1}$. Combining these results with (3.8) and (3.9), we get

$$A_eA_e^* + A_oA_o^* = \tfrac{1}{2}I,$$

and

$$\tfrac{1}{16}(A_eA_e^*)^{-1} + \tfrac{1}{16}(A_oA_o^*)^{-1} = \tfrac{1}{2}I.$$

From these two equations for $A_e A_e^*$ and $A_o A_o^*$ we readily obtain $A_e A_e^* = A_o A_o^* = \frac{1}{4} I$. Combining this result with (3.7), and again using Lemma 3.3.2 where $P = A_e$ and $Q = \overline{S_e}$, we conclude that $A_e = \overline{S_e}$ and similarly for $A_o$, $S_o$, we conclude that $A_o = \overline{S_o}$, which completes the proof. □

In conclusion, the design of LOTs is fully characterized by a unitary map $B_a$ and $A_e$, $A_o$ satisfying

$$A_e A_e^* = A_o A_o^* = \tfrac{1}{4} I. \tag{3.10}$$

## 3.4 Design of perceptually-relevant LOTs

In this section we consider the problem of designing suitable para-unitary linear-phase filter banks. So far, we have satisfied all the constraints mentioned in Section 3.2, except for three, namely,

1. $H_k(z)$, $F_k(z)$, $k = 2, \ldots, K$, have zeros at $z = 1$,
2. good frequency discriminating properties of the analysis filters,
3. smooth synthesis impulse responses, decaying to zero at the endpoints of their supports.

The first condition has been added because we can then represent DC information with the aid of only one coefficient. This will give to the most efficient representation of flat backgrounds. The second condition enables us to reduce coder complexity, while the third condition ensures that blocking artifacts are avoided. To satisfy these conditions, we can restrict our attention to $A_e$ and $A_o$ because the design of $B_a$ can be done independently as it only affects the odd-symmetric filters. For this reason, in the next subsection we will focus on the design of $A_e$ and $A_o$. The design of $B_a$ is discussed separately in subsection 3.4.2.

### 3.4.1 Design of $A_e$ and $A_o$

In this subsection we show how the above conditions, which are imposed on the composite matrix $A$, can be turned into constraints on the submatrices $A_e$ and $A_o$. To do this, let $A_e = (a_e(k, n))_{k,n}$ and $A_o = (a_o(k, n))_{k,n}$, and stack the matrices $A_e$ and $A_o$ in a matrix $U \in \mathbb{C}^{K \times K}$,

$$U = \begin{bmatrix} A_e \\ A_o \end{bmatrix}. \tag{3.11}$$

We then have the following result.

**Theorem 3.4.1** *Let* $s = (1, \ldots, 1)^t \in \mathbb{C}^K$ *and* $e_K = (1, 0, \ldots, 0)^t \in \mathbb{C}^K$. *The polynomial transfer functions* $H_k(z)$, $F_k(z)$, $k = 2, \ldots, K$, *have zeros at* $z = 1$ *if and only if*

$$Us = \gamma e_K, \qquad |\gamma| = \tfrac{1}{2}\sqrt{K},$$

*and*

$$a_e(1, n) = \frac{\gamma}{K} \qquad \textit{for all } n = 1, \ldots, K.$$

**Proof:** Assume that the analysis filters, except for the first one, have zeros at $z = 1$. Since the odd-symmetric filters have a zero at $z = 1$ by definition, we can concentrate on the even-symmetric filters only. It follows that $A_{1,1}s = \gamma e_{K/2}$ so that $A_e s = \gamma e_{K/2}$ and $Us = \gamma e_K$ for some $\gamma \in \mathbb{C}$. Moreover, since $UU^* = \frac{1}{4}I$ and thus $U^*U = \frac{1}{4}I$, we get $s = 4U^*Us = 4U^*\gamma e_K$ so that $a_e(1, n) = (4\overline{\gamma})^{-1}$ for all $n = 1, \ldots, K$. Therefore, since $Us = \gamma e_K$, we conclude that $K(4\overline{\gamma})^{-1} = \gamma$, so that $|\gamma| = \frac{1}{2}\sqrt{K}$ and thus $a_e(1, n) = (4\overline{\gamma})^{-1} = \frac{\gamma}{K}$ for all $n = 1, \ldots, K$. The proof of the converse is trivial. □

A direct consequence of Theorem 3.4.1 is that the only freedom we have in designing the first LOT filter is in choosing a suitable sequence $a_o(1, \cdot)$. The higher-order filters ($k > 1$) can be obtained by a joint optimalization of both $a_e(k, \cdot)$ and $a_o(k, \cdot)$. To satisfy condition 2, we conclude from (3.5), that the sequences $a_e(k, \cdot)$ and $a_o(k, \cdot)$ must have good frequency-discriminating properties. Moreover, condition 3 is satisfied if we take smooth sequences $(a_e(k, n))_{n=1}^K$ and $(a_o(k, n))_{n=1}^K$ of which the sum $a_e(k, n) + a_o(k, n)$ decays smoothly to zero at the left endpoints of their supports. Figure 3.5 and Figure 3.6 show an example of suitable sequences $a_e(1, \cdot)$ and $a_o(1, \cdot)$ and the resulting sum sequence $a_e(1, \cdot) + a_o(1, \cdot)$, respectively, when $K = 32$. Note that $a_e(1, n) = \frac{1}{2}\sqrt{\frac{1}{K}} \approx 0.088$ for all $n = 1, \ldots, 32$. In the following we discuss a recursive method for the design of perceptually-relevant LOTs, these being LOTs which satisfy all the above conditions. The term recursive reflects the fact the we optimize one filter after the other. Such a procedure clearly has advantages over a direct approach, because otherwise the number of variables to be optimized becomes unfeasible for large values of $K$.

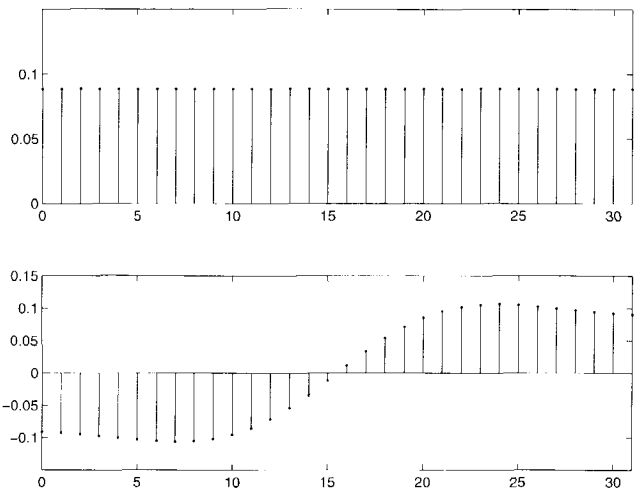



**Figure 3.5:** *Example of smooth sequences* $a_e(1,\cdot)$ *and* $a_o(1,\cdot)$.

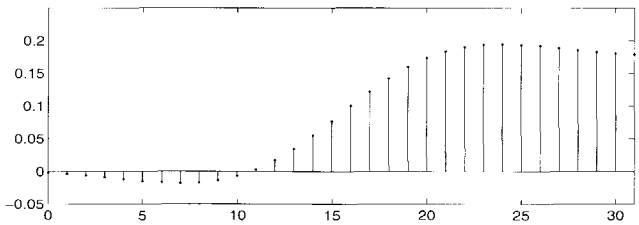



**Figure 3.6:** *Sum sequence* $a_e(1,\cdot) + a_o(1,\cdot)$.

### A recursive method for the design of perceptually-relevant LOTs

From the DCT, we may conclude that the DCT basis functions satisfy all the above conditions, except for one of them, which is that the sum sequence $a_e(1,\cdot) + a_o(1,\cdot)$ does not decay smoothly to zero at the left endpoint of its support. The design algorithm we propose here exploits these properties and modifies the sequence $a_o(1,\cdot)$ such that the resulting sum sequence does have the desired property of smoothness. This is accomplished by weighting the coefficients $a_o(1,n)$ with some properly chosen weighting function and normalizing the resulting sequence to unity, recursively. In order to preserve the symmetry properties of $A_o$, the weighting function must be even-symmetric. Since a modification of

$a_o(1, \cdot)$ affects the orthonormality of the rows of $A_o$, we can apply a Gram-Schmidt orthogonalization procedure. The Gram-Schmidt process says that if $\mathcal{W} = (w_n)_{n=1}^N$ is a linearly independent set of vectors, then there is an orthogonal set $\mathcal{V} = (v_n)_{n=1}^N$ such that $v_n \neq o$ and $v_n$ is a linear combination of $w_1, \ldots, w_n$ for every $n = 1, \ldots, N$. The $v_n$ are constructed by

$$v_n = w_n - \sum_{k=1}^{n-1} \frac{\langle w_n, v_k \rangle}{\|v_k\|_2^2} v_k. \tag{3.12}$$

Clearly, the idea of the Gram-Schmidt process is to subtract from every new vector its components in the directions that are already settled and normalize it. As a consequence, after we have optimized $a_o(1, \cdot)$, we have to optimize the remaining sequences as well since they all are affected by the Gramm-Schmidt procedure. As mentioned above, for $k > 1$, a joint optimalization of $a_e(k, \cdot)$ and $a_o(k, \cdot)$ can be performed. However, for the purpose of implementation, it is better to optimize $A_o$ only, and leave $A_e$ unchanged. We will return to this point in more detail in Chapter 5 and leave this issue aside for the moment. Algorithm 3.1 shows the pseudo-code of the recursive design algorithm. Figure 3.7 shows an example of LOT impulse responses (only the first and fourth filter responses are shown) obtained with Algorithm 3.1 when $K = 32$ and $B_a$ has been designed as discussed in the next subsection. It can be seen that these functions smoothly decay to zero at the endpoints of their supports. Figure 3.8 shows the frequency responses (only the first eight responses are shown).

An alternative recursive design method, which is computationally more expensive but robust with respect to any initial solution (the sequences $a_e(k, \cdot)$ and $a_o(k, \cdot)$ can be chosen randomly), is discussed in [81], where an optimization procedure is proposed which is formulated entirely in the frequency domain. The results obtained with this method are very close to the solutions obtained with Algorithm 3.1, and where almost independent of the initial solution chosen. This indicates that in the orthonormal case, Algorithm 3.1 leads to "optimal" LOTs. When the signal transform to be designed is biorthogonal or oversampled, however, Algorithm 3.1 fails, whereas the algorithm proposed in [81] does not.

### 3.4.2 Design of $B_a$

In this subsection we concentrate on the design of $B_a$ and have assumed that the even-symmetric filters are properly designed so that they satisfy the above conditions.

```
/* initialization */
A_e = 0.5*even_dct_functions (K);
A_o = −0.5*odd_dct_functions (K);
α = 0.05;
/* program body */
for k = 1 to K/2,
do
        Δ = a_e(k, 1) + a_o(k, 1);
        while |Δ| > threshold(k),
        do
                if Δ > 0,
                        window = 1 + α*hanning (K);
                else
                        window = 1 − α*hanning (K);
                fi;
                weight_sequence (a_o(k, ·), window);
                Gramm_Schmidt (A_o);
                Δ = a_e(k, 1) + a_o(k, 1);
        od;
od;
```

**Algorithm 3.1:** *Recursive design algorithm for LOTs.*

One way to obtain the odd-symmetric filter responses is to shift the frequency responses of the even-symmetric filters over $\pi$ radians. Thus, let $h_k$ be an even-symmetric filter. Then the corresponding odd-symmetric filter, say $h_l$, satisfies

$$\begin{aligned}\hat{h_l}(\xi) &= \sum_{n=-\infty}^{\infty} h_l(n)e^{in\xi} \\ &= \sum_{n=-\infty}^{\infty} h_k(n)e^{in(\xi-\pi)} \\ &= \sum_{n=-\infty}^{\infty} (-1)^n h_k(n)e^{in\xi},\end{aligned}$$

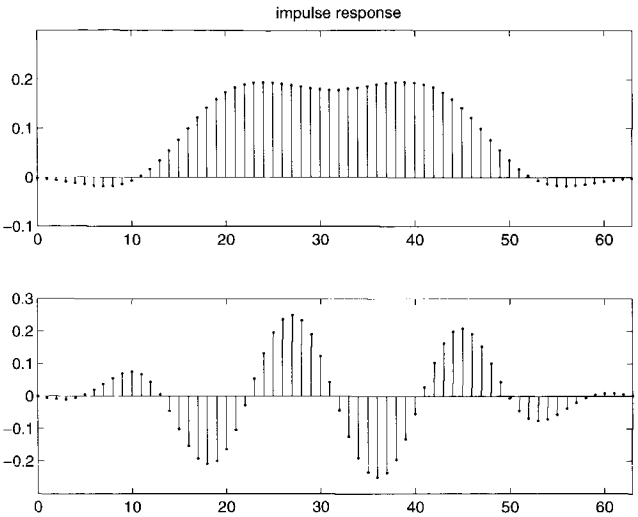



**Figure 3.7:** *Impulse responses of a size* 32 × 64 *LOT. Only the first (top) and the fourth (bottom) filter responses are shown.*

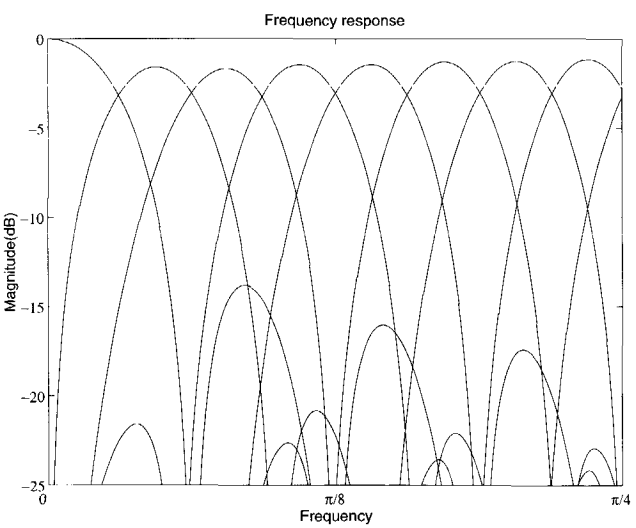



**Figure 3.8:** *Frequency responses of a size* 32 × 64 *LOT. Only the first eight responses are shown.*

so that

$$h_l(n) = (-1)^n h_k(n), \tag{3.13}$$

for all $n = 1, \ldots, L$. It is easily verified that $l = K + 1 - k$. Let $D = \text{diag}(d_1, \ldots, d_K)$ with $d_n = (-1)^n$ for all $n = 1, \ldots, K$. We then can express (3.13) for all the $\frac{K}{2}$ odd-symmetric filter responses as

$$A_{2,1} = JA_{1,1}D.$$

Due to the PR condition together with the linear-phase conditions we have to satisfy $A_{2,1} = B_a A_{1,1}$. The row space of $B_a A_{1,1}$, however, does not generally coincide with the row space of $JA_{1,1}D$. In the case that the two row spaces do not coincide we will approximate $JA_{1,1}D$ in a least-square sense, i.e. we will approximate $JA_{1,1}D$ by the orthogonal projection of the row space of $JA_{1,1}D$ onto the row space of $B_a A_{1,1}$. This means that

$$\begin{aligned} B_a A_{1,1} &= JA_{1,1}D\left(B_a A_{1,1}\right)^*\left(B_a A_{1,1}\left(B_a A_{1,1}\right)^*\right)^{-1} B_a A_{1,1} \\ &= JA_{1,1}DA_{1,1}^*\left(A_{1,1}A_{1,1}^*\right)^{-1} A_{1,1}, \end{aligned}$$

so that $B_a$ is given by

$$B_a = JA_{1,1}DA_{1,1}^*\left(A_{1,1}A_{1,1}^*\right)^{-1}.$$

Since both $J$, $D$ and $A_{1,1}$ are of full rank we conclude that $B_a$ is non-singular, as required.

In order to have $B_a$ unitary, however, we can apply a Gram-Schmidt orthogonalization procedure to $B_a$. This can be done either to the rows or the columns of $B_a$. For perceptual reasons, however, we apply the procedure to the rows. This can be seen as follows. From (3.12), we conclude that the Gram-Schmidt process subtracts from every new vector its components in the directions that are already settled and normalize it. As a consequence, the deviation of the orthogonalized vectors with respect to the original non-processed vectors increases with increasing index, i.e., $\|w_{n+1} - v_{n+1}\|_2 \geq \|w_n - v_n\|_2$ for all $n = 1, \ldots, N-1$. Next consider the construction of $A_{2,1}$ and assume that we have ordered the even-symmetric analysis filters in $A$ such that the pass-band centre-frequency of the frequency responses increases with increasing row index $k$. Obviously, as $A_{2,1} = B_a A_{1,1}$, the $k$th row of $A_{2,1}$ is only affected by $A_{1,1}$ and the $k$th row of $B_a$. Now, since the sensitivity of the human visual system varies as a function of spatial frequency in the sense that it is less sensitive to the higher spatial frequencies than to the middle and lower spatial frequencies [73, 74], we conclude that we are less sensitive

to errors in the higher order rows of $B_a$ than to errors in the lower-order rows. So, based on what was said above, we can exploit this property by applying the Gram-Schmidt procedure to the rows rather than to the columns of $B_a$, starting with the first row, then the second one and so forth.

We end this section by showing under what conditions of $A_e$, $A_o$ and $B_a$ we satisfy $B_a A_{1,1} = J A_{1,1} D$. To do so, we need the following lemma.

**Lemma 3.4.1** *$A_e D = T A_o$ and $A_o D = T^{-1} A_e$ for some non-singular $T \in \mathbb{C}^{\frac{K}{2} \times \frac{K}{2}}$.*

**Proof:** Since $JDJ = -D$ we have $A_e DJ = (A_e J)(JDJ) = -A_e D$ and $A_o DJ = (A_o J)(JDJ) = A_o D$ so that $A_e D$ is odd-symmetric and $A_o D$ is even-symmetric. Therefore, we have $A_e D = T A_o$ for some non-singular $T \in \mathbb{C}^{\frac{K}{2} \times \frac{K}{2}}$ and thus $A_o D = T^{-1} A_e$, which completes the proof. □

The next theorem finally gives the required conditions.

**Theorem 3.4.2** *$B_a A_{1,1} = J A_{1,1} D$ if and only if $B_a A_o = J A_e D$ and $B_a = J B_a^{-1} J$.*

**Proof:** Let us assume that $B_a A_{1,1} = J A_{1,1} D$. Then $B_a(A_e + A_o) = J(A_e + A_o)D$. Since both $A_e$ and $A_o$ are of full rank and $A_e D$ and $A_o D$ are odd and even-symmetric, respectively, by Lemma 3.4.1, we conclude that $B_a A_o = J A_e D$ and $B_a A_e = J A_o D$. Moreover, from $B_a A_o = J A_e D$ we conclude that $J A_o D = J B_a^{-1} J A_e$ so that $B_a = J B_a^{-1} J$.

Conversely, if we assume that $B_a A_o = J A_e D$ and $B_a = J B_a^{-1} J$. We then have $J A_o D = J B_a^{-1} J A_e = B_a A_e$. Hence, $J A_{1,1} D = J(A_e + A_o)D = J A_e D + J A_o D = B_a A_o + B_a A_e = B_a(A_e + A_o) = B_a A_{1,1}$ as required. □

## 3.5 Conclusions

We have investigated the design of analysis and synthesis filters for the purpose of data compression, especially the data compression of images. We indicated desirable constraints imposed by the application, on the basis of which we derived a suitable filter bank architecture (choice of $K, L, m$). We concluded that LOTs are suitable for our specific application. These transforms correspond to critically sampled para-unitary linear-phase filter banks. LOTs enable us to reduce blocking effects and at the same time limit ringing effects due to long filter responses. We

showed that any PR linear-phase filter bank with (fully) orthonormal analysis and synthesis filters can be generated by (3.5) with $S = \overline{A}$, $B_a$ unitary, and $A_e$, $A_o$ satisfying (3.10). Furthermore, we turned the perception-related constraints on the filter responses into constraints on the constituent matrices $A_e$ and $A_o$. Finally, we showed how to design a suitable matrix $B_a$ such that the resulting odd-symmetric filters also satisfy the given constraints.

# Chapter 4

# *Application to X-ray cardio-angiograms*

**Contents**



## 4.1 Introduction

The previous chapters touched upon the design of discrete-time signal transforms, in particular the design of LOTs, for data compression of images. In this chapter we discuss the application of overlapped transform coding with LOTs to X-ray cardio-angiographic image series.

*X-ray cardio-angiography*[1], which can be used very effectively in adult coronary and left ventriculography, is a useful tool in the rapid diagnosis of cardiac abnormalities and in clinical decision-making [82]. X-ray angiography is one of the major imaging modalities in radiology. Other modalities, for example, are

[1] X-ray cardio-angiography visualizes the cavities of and the large arteries around the heart using röntgenology.

computed tomography, magnetic resonance imaging, nuclear medicine and digital fluorography [83].

The processing trend in medical radiologic imaging is increasingly towards digital. The rationale behind this trend is the wide variety of opportunities it offers to support image transfer and archiving, and to manipulate visual diagnostic information in useful and novel ways, such as image enhancement and volume rendering [84]. Another impulse comes from the *picture archiving and communication systems* (PACS) community who envisions an all-digital radiology environment in hospitals for the acquisition, storage, communication and display of large volumes of images in various modalities [85]. Image compression is a major contribution in using PACS as an alternative to analogue film-based systems by reducing the bit rate required to store or transmit images, while maintaining the relevant diagnostic information. It also enables fast transmission of large medical images over a PACS network to display them on a workstation for diagnostic, review and teaching purposes. A rather new application, which has received much attention over the last few years and which inevitably requires data compression, is teleradiology. Teleradiology is a wide area network (WAN) application that aims to bring expert radiological service, which is available in major urban medical centres into rural areas and small towns, usually via low-bandwidth channels, such as a long-distance telephone line or an integrated services digital network (ISDN) supporting data rates of 144 Kbit/s [86, 87].

At present, resolutions of $512 \times 512$ pixels per image are used in digital X-ray angiography, but an increase in the resolution to $1024 \times 1024$ pixels may be expected in the near future. Philips, for example, has introduced the Integris Cardiac Imaging system [88], which enables recording of $1024 \times 1024$ images. Image rates up to 15 Hz are most frequently used, but an increase to 30 Hz or more can be expected in the near future. Storage of these image series will require fast interfaces and a large amount of storage space. One way to reduce the access speed and storage space, and thereby reduce the storage cost, is to apply data compression.

In this chapter we investigate the performance of data compression of X-ray cardio-angiographic image series. Data compression can be divided into two categories: lossless and lossy compression. With lossless compression the bit rate of the image series can be reduced by a factor of about 2.5 – 3.5 [1, 2]. The aim of the work described here, however, is to reach reduction factors in the order of 8 – 16. To achieve this goal, we rely on lossy compression techniques, in particular those that are based on overlapped transform coding. Our objective is to produce images that do not contain perceptually annoying artifacts and preserve

the original diagnostic quality.

### Organization of this chapter

The outline of this chapter is as follows. Digital X-ray cardio-angiographic images play an important role throughout this chapter, and it thus seems appropriate to start with a brief review of this imaging modality. In Section 4.2 we briefly review the digital diagnostic X-ray system and discuss some X-ray chain properties, such as transfer function and noise characteristics. This review is not intended to serve as a detailed description of the X-ray system but to understand those system features that play a role in selecting an appropriate compression technique and analyzing its performance. Moreover, we discuss some additional constraints that are imposed on the method of data compression by the clinical procedure used during digital X-ray angiography and, on the basis of these constraints, we select a suitable compression technique. In Section 4.3, we concentrate on the complete transform-coding system. Philips has proposed that this coding system be included in the discussion on standardization of lossy data-compression algorithms which is being organized by the ACR-NEMA committee with the support of the National Electrical Manufacturers' Association (NEMA) and the American College of Radiology (ACR) [89]. We discuss signal transformation, quantization and lossless encoding and we show how we can adapt the compression to the characteristics of the human visual system and to a post-processing technique called image enhancement. In Section 4.4, we consider computer simulation results of this compression method. Finally we draw some conclusions in Section 4.5.

## 4.2 X-ray cardio-angiographic image series

In this section we will briefly review the digital diagnostic X-ray system and discuss some constraints that are imposed on the method of data compression by the clinical procedure used during digital X-ray angiography.

Figure 4.1 shows the X-ray acquisition chain in the form of a diagram. In the X-ray tube a beam of electrons, accelerated by a potential of several thousand volts, hits a metallic target called the anode. In the anode the electrons are decelerated as a result of which a substantial part of the total energy from the charge is emitted as radiation, called deceleration radiation or more commonly, *bremsstrahlung*. The production of X-rays is accompanied by a large amount of heat. To avoid anode deterioration, the target used is a rotating disc so that the

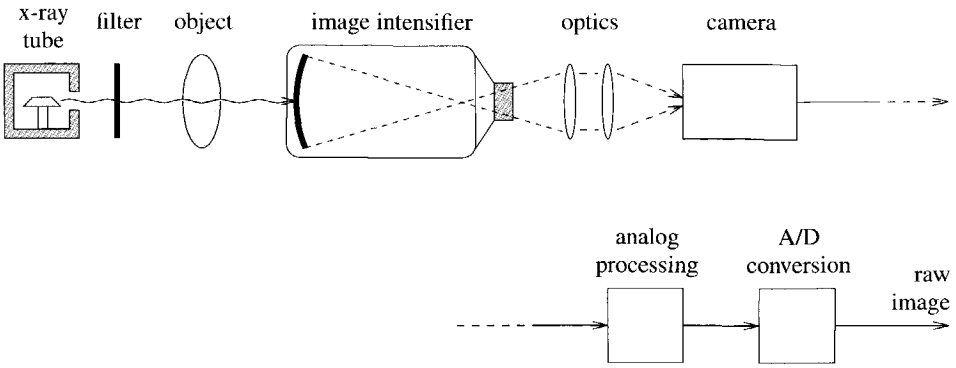



**Figure 4.1:** *A diagram of the X-ray acquisition chain.*

heat is equally distributed over the target. The X-ray tube is followed by a filter which removes the low-energy photons. These low-energy photons do not contribute to the image formation because they will be completely absorbed by the object being radiated. Since any amount of X-radiation causes damage to living tissues and organisms, it is important to filter out these low-energy photons. The filter is followed by an image intensifier. The input screen of the image intensifier, a cesium iodide (CsI) X-ray phosphor layer, converts the X-ray photons into electrons. These electrons are accelerated and focussed at an output phosphor screen which converts the electrons into light photons. The output of the image intensifier is connected, via some optics, to a camera which converts the light photons into an electrical (analog) signal. One of the last steps in the acquisition chain is the reduction of the voltage range. This is a non-linear operation resulting in a more equally distributed error, introduced in the digitalization of the signal, over the dark and bright parts of the image. This operation is commonly referred to as *white compression*. After white compression the signal is finally digitized in the analog-to-digital convertor.

Several components are involved in the X-ray acquisition chain, each of which has an impact on the final image quality. Two major features of the X-ray chain are its *modulation transfer function* (MTF), and its *noise characteristics*. The MTF is the response of the acquisition chain in the frequency domain. It has a low-pass characteristic which is mainly determined by the behaviour of the image intensifier and the camera. The impinging X-ray photons are thus low-pass filtered by the imaging system. As a consequence, most of the signal energy can be expected

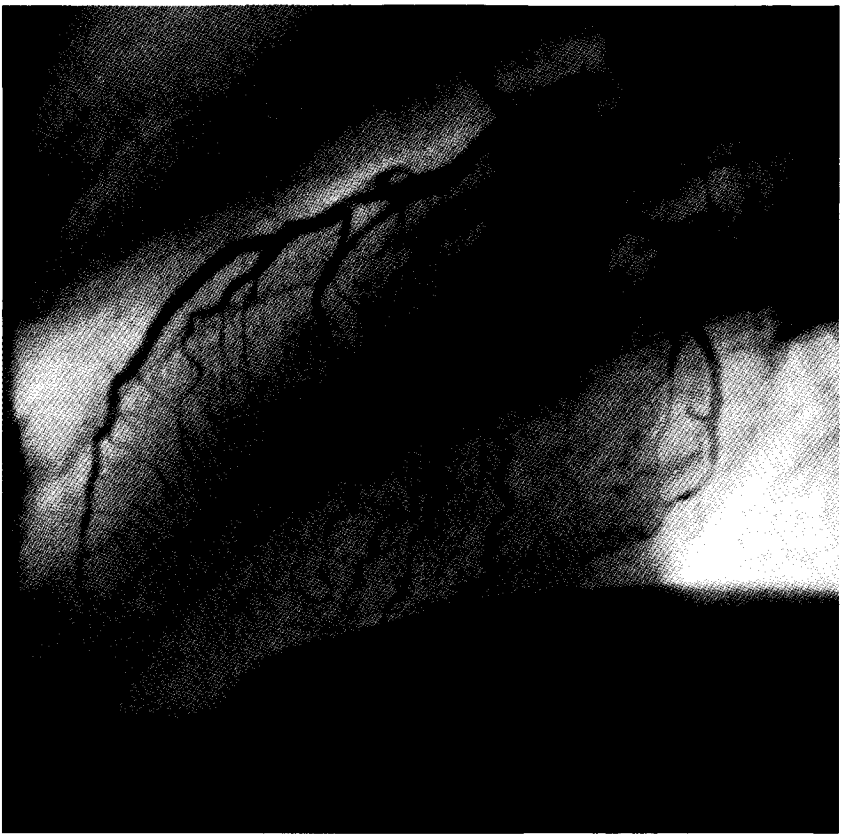


**Figure 4.2:** *A typical (raw) X-ray cardio-angiogram.*

to be concentrated in the lower spatial frequency band. X-ray systems are subject to different types of noise, including camera noise, noise introduced in the image intensifier and quantum noise. Quantum noise is the most dominant noise component and is caused by spatial fluctuations in the number of impinging photons. Quantum noise can be modelled as a Poisson process in which the variance is proportional to the X-ray dose [90]. Therefore, the high-intensity parts of the image contain more quantum noise than the low-intensity parts. Figure 4.2 shows a typical example of a (raw) X-ray cardio-angiogram. The term "raw" reflects the fact that no signal processing (e.g., sharpness perception enhancement) has been applied to the images.

This given information can be used in a data compression scheme. Thus we can exploit the fact that the system MTF has a low-pass characteristic in the allocation of available bits to the frequency components. Similarly, we can exploit the fact that the high-intensity parts of the image contain more noise than the low-intensity parts by accepting larger quantization errors in areas with a larger intensity, thereby achieving a better balance between the perceptual image quality in the dark and bright areas [91].

### Clinical procedure

In the past, the interpretation of angiograms was performed (off-line) using 35 mm cinefilm. Most angiograms were reviewed visually, hence, subjectively. A great advantage though of the off-line cinefilm is its high spatial resolution. Nowadays, the diagnosis is performed both objectively and subjectively. Recent developments in digital cardiac imaging systems have been directed towards obtaining on-line quantitative measurements, i.e., from digitized video images during the catheterization procedure. The digital systems provide a high density resolution of 8 to 10 bits per pixel and an image rate of up to 30 Hz. These features are more suitable for quantitative analysis than the conventional cinefilm approach can offer.

Figure 4.3 shows a block diagram of the various steps involved in the angiogram interpretation process. The quantitative analysis of vessel diameter and wall motion measurements [82], in particular, is usually performed using the raw (decoded) data, i.e. without any further processing of the image. Prior to displaying the images, however, they are commonly enhanced to increase the sharpness perception. This enhancement is in fact an emphasis of the middle and high spatial frequencies in the image using unsharp masking techniques [92]. Figure 4.4 shows the enhanced version of the raw image in Figure 4.2.

It should be noted that the image not only looks sharper than the original raw image but also looks more noisy. We should be aware of the fact that besides the original information, the quantization errors introduced by the compression system will also be emphasized by the enhancement. No coding artifact should be visible in the enhanced image.

During visual inspection of the image series, the cardiologist may want to access images in an arbitrary order. Each image should, therefore, be separately accessible from the storage medium. This means that each image should be encoded separately. This type of compression is usually referred to as intra-frame coding. Finally, the quality of the decompressed images must be such that the cardiologist can use them for the diagnosis of abnormalities and for decision-making.

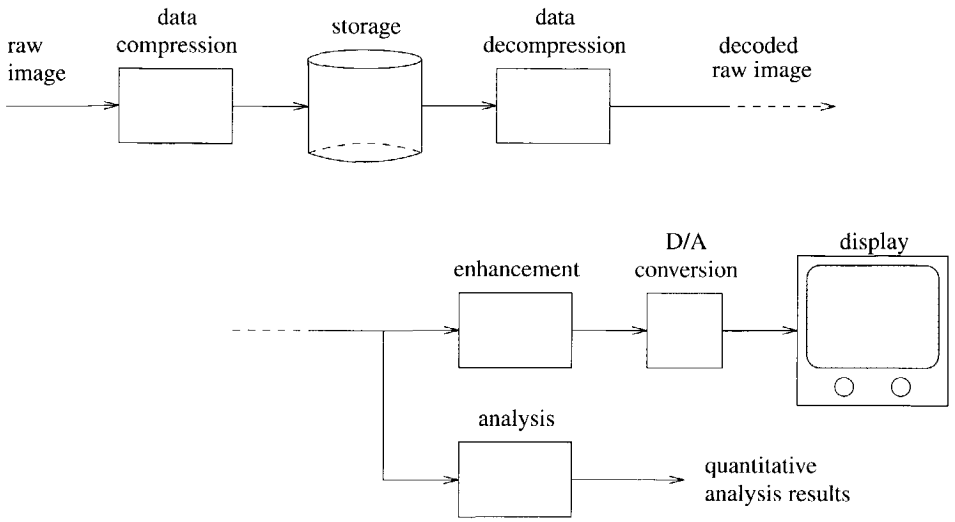



**Figure 4.3:** *Block diagram of the steps involved in the interpretation of X-ray cardio-angiograms.*

## 4.3 The overlapped transform coding system

Based on what has been said above, we conclude that the data compression technique for compressing X-ray cardio-angiographic image series has to be an intra-frame lossy coding technique. Figure 4.5 gives a block diagram of such a coding system. The boxes labelled $T_a$, $Q$ and $C$ denote (analysis) signal transformation, quantization and lossless coding, respectively. In the following subsections, we will describe the individual blocks of the encoder in more detail. We will not consider the source decoder here since it behaves more or less as the inverse of the source encoder.

### 4.3.1 Signal transformation

In Chapters 2 and 3, we discussed the design of discrete-time signal transforms for the compression of images, and we concluded that LOTs are suitable candidates. These transforms provide a sufficient degree of freedom to reduce the blocking effects as well as limiting ringing effects due to long filter responses. We interpreted the LOT as a $(K, L, m)$ multi-rate filter bank for which $K = m = 2L$. There is

**Figure 4.4:** *A typical enhanced X-ray cardio-angiogram.*

thus one independent parameter which has to be determined from application specific input data. This parameter is the filter length $L$. Based on the reasoning in Section 2.6, we know that $L$ must reflect the average "degree" of stationarity in the image. This means that images containing relatively stationary data require longer filter lengths than images in which the statistics vary quite rapidly.

In order to determine the optimal filter length for coding X-ray cardio-angiograms, we can set up an experiment similar to the one described in Section 2.6. The source material we use in this experiment is the enhanced image shown in Figure 4.4. The size of the image is 512 lines with 512 pixels each, where each pixel is represented by 8 bits. The coding is performed using a (dead-zone) uni-

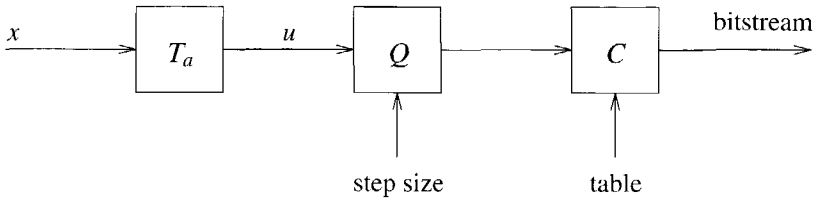



**Figure 4.5:** *Block diagram of the transform coding encoder.*

form quantizer and a Huffman encoder [26]. The distortion measure we use is the squared-error distortion measure. We repeat the coding experiment using LOTs of different block sizes. These LOTs are designed using the recursive Algorithm 3.1 which produces perceptually-relevant transforms. The resulting rate-distortion curves for the block sizes $K = 16$, 32 and 64 are shown in Figure 4.6. It is seen that the optimal block size for coding the angiograms is $K = 32$, which corresponds to a filter bank with filters of length 64. If we either increase or decrease the block size to 64 or 16, respectively, the total rate needed to reproduce the image with some fixed distortion increases in both cases. It should be noted, however, that the differences are quite small for this type of imaging modality. One of the reasons is that, besides the diagnostic information, these images contain a significant amount of quantum noise. This noise component is stationary over and between the images and, therefore, requires a large block size for minimum rate coding. The diagnostic information, however, is non-stationary and needs a smaller block size. It turns out that a block size of 32 serves as a good compromise to code both components efficiently. Figure 4.7 shows basis functions (only the first ten basis functions are shown) of the $32 \times 64$ LOT. This is the LOT that we will be using in the remainder of this chapter.

For video material, like the "mobile-calender" image used in the experiment in Section 2.6, it turned out that the optimal block size equals $K = 8$, as it is chosen in the JPEG and MPEG standards [5, 7]. This value is significantly smaller than the optimal block size found for the X-ray angiograms. Therefore, we may conclude that X-ray cardio-angiograms have a much more stationary character than video images, so that apart from the blocking artifact argument, this provides a second argument for not using JPEG or MPEG for the compression of X-ray cardio images.

Another important conclusion that we can draw here is the following. If we want to compress different image modalities efficiently, like X-ray vascular, com-

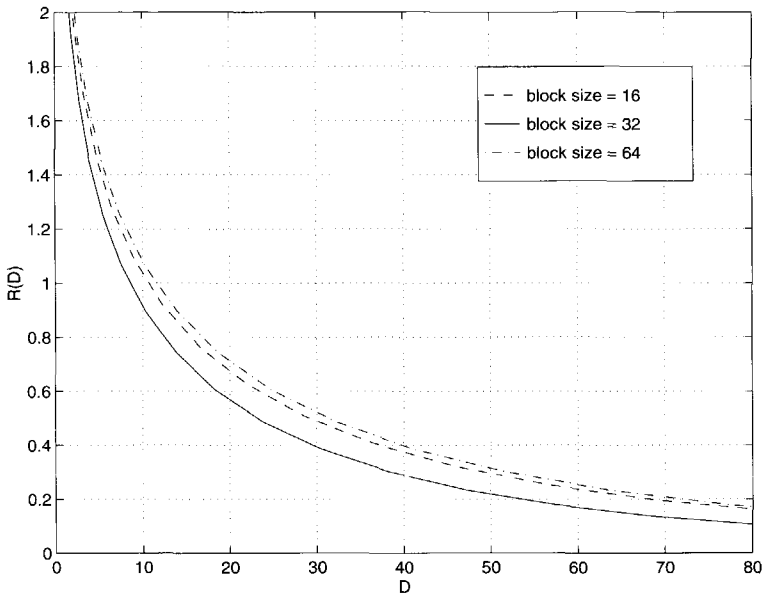



**Figure 4.6:** *Rate-distortion curves for the X-ray angiographic image shown in Figure 4.4.*

puted tomography and magnetic resonance imaging, with one and the same compression system, the signal transform must not be a fixed in size. This limitation on the transform does affect for the hardware implementation since it requires a certain degree of flexibility. We will return back to this issue in the next chapter, where we discuss an architecture which is fully programmable in the sense that it can be used for arbitrary signal transforms.

In the remainder of this chapter we consider two-dimensional signals since some functions in the overall coding system cannot be described in terms of one-dimensional signals only.

### 4.3.2 Quantization

The purpose of quantization is to remove information that cannot or need not be perceived by the receiver. This is accomplished by mapping the incoming sequence of transform coefficients $u$ onto quantizer-output symbols. Usually, the number of distinct quantizer-output symbols is much smaller than the number of

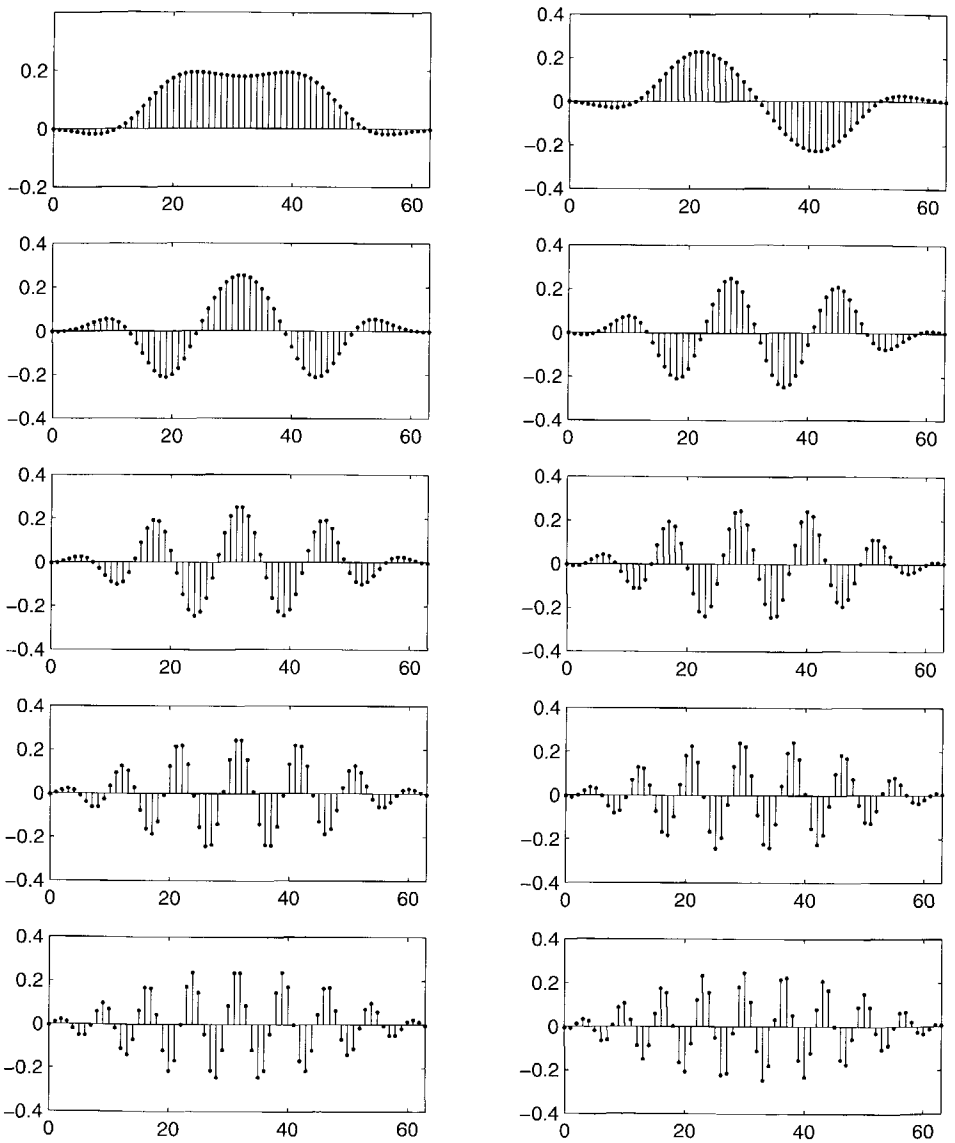



**Figure 4.7:** *Impulse responses of a size* $32 \times 64$ *perceptually-relevant LOT (only the first ten basis functions are shown).*

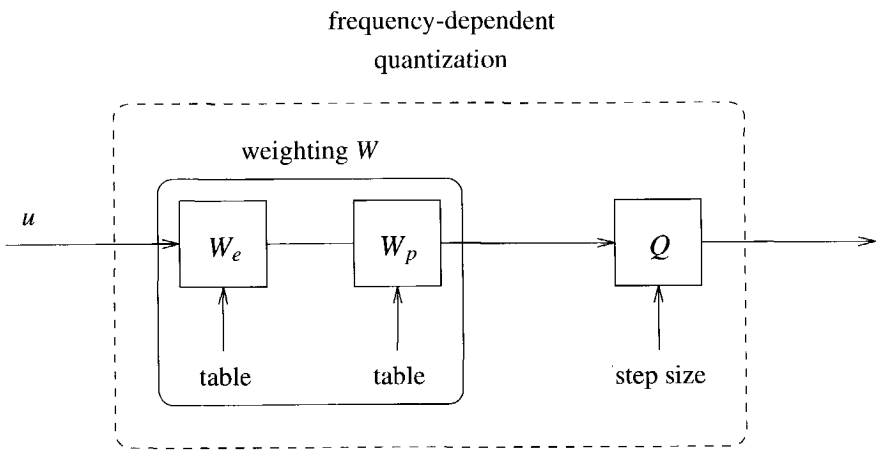



**Figure 4.8:** *Block diagram of the quantization operator.*

quantizer-input values. The quantizer symbols are transmitted to the decoder. The number of bits required to represent the quantizer-output symbols (after lossless coding) is generally smaller than the number of bits required to represent the original transform coefficients. Figure 4.8 shows a block diagram of the quantization operation.

The block denoted $Q$ is a mid-tread dead-zone uniform quantizer [93, 26]. The characteristic of this quantizer is almost the same as the characteristic of a purely uniform quantizer, except for a somewhat larger quantization interval around zero, usually referred to as the *dead zone*. The dead zone ensures that very small coefficient values are quantized to zero, which significantly decreases the entropy of the quantized coefficients and thus reduces the number of bits needed to represent them. It does not, however, introduce significantly more visible quantization errors than purely uniform quantization would do. The size of the quantization intervals, except for the one around zero, is usually referred to as the *step size* of the quantizer. The box denoted $W_p$ performs a (perceptual) weighting of the transform coefficients $u$ to adapt the quantization to the human visual system. The box $W_e$ performs another weighting of the transform coefficients. The objective of this weighting is to adapt the quantization function to the post-processing enhancement, which is very often applied to medical images before they are viewed on a monitor or printed in hard copy. We will discuss both adaptations in more detail below.

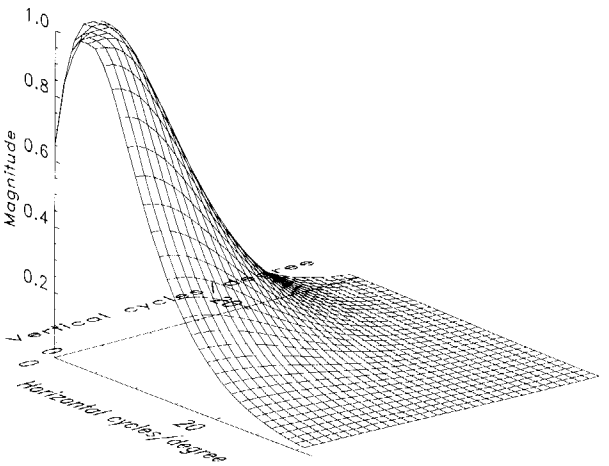



**Figure 4.9:** *A typical (normalized) CSF.*

### Adaptation to the human visual system: the perceptual weighting

The sensitivity of the human visual system to intensity variation is a function of the spatial frequency. This function is usually referred to as the contrast sensitivity function (CSF) and is roughly isotropic (rotation-invariant) [73, 74]. Figure 4.9 shows a typical (normalized) CSF [74]. The CSF reflects the fact that the human visual system is less sensitive to the higher spatial frequencies than to the mid and lower spatial frequencies.

The overlapped transform decomposes the image into spatial frequency components (transform coefficients). We can exploit the non-uniform frequency sensitivity of the human visual system by quantizing the transform coefficients with varying accuracies (i.e. non-uniform step size). We can accomplish this in practice by first multiplying the sequence of transform coefficients $u = (u(k,l))_{k,l}$ by weighting factors $\omega_{k,l}$, which are positive and bounded by one. These factors are sample values of the contrast sensitivity function of the human visual system at the frequencies $(k,l)$. The coefficients weighted this way can then be quantized with a uniform accuracy quantizer. The net result is that the lower the value of $\omega_{k,l}$, the more coarsely the coefficient $u(k,l)$ is quantized.

Let $\xi_h$ and $\xi_v$ denote horizontal and vertical spatial frequencies, respectively,

and let $\xi_r^2 = \xi_h^2 + \xi_v^2$. The CSF is defined [74] as

$$f(\xi_r) = (c + \tfrac{\xi_r}{\xi_{\max}})e^{-\left(\frac{\xi_r}{\xi_{\max}} - 1 + c\right)},$$

where the parameters $c$ and $\xi_{\max}$ are 0.05 and 8 cycles per degree, respectively. This function is modelled in units of cycles per degree and has to be mapped onto the two-dimensional frequency domain using the calibration parameters of viewing distance and pixel spacing. As a consequence, the weighting factors have to be adapted to the conditions under which the images are viewed. In the present application, it would be unrealistic to fix the viewing distance at a single value, yet we may assume that the observer will stay within a range of viewing distances, say $\Lambda_v \subset \mathbb{R}$. We, therefore, consider the worst case situation for the weighting function, which implies that we take the maximum sensitivity of the human visual system over the complete set $\Lambda_v$. For this reason, we use a modified CSF as follows,

$$f(\xi_r) = \max_{\Lambda_v} \left( (c + \tfrac{\xi_r}{\xi_{\max}})e^{-\left(\frac{\xi_r}{\xi_{\max}} - 1 + c\right)} \right).$$

Figure 4.10 shows the modified function for a viewing distance ranging from three times the screen height to infinity. This approach ensures that the modified curve can be used over the complete set $\Lambda_v$.

### Adaptation to post-processing: the enhancement weighting

Very often some form of post-processing is applied to medical images before they are viewed. X-ray images, for example, are often enhanced in order to increase the impression of sharpness. This enhancement is in fact an emphasis of the middle and high spatial frequencies using digital filtering techniques.

Image enhancement is usually based on a technique which is referred to as unsharp masking [92]. With the method of unsharp masking, the input (raw) image is first low-pass filtered. The low-pass image thus obtained (the unsharp mask) is then subtracted from the input image to obtain a high-pass version of the raw image. This high-pass image, multiplied by a gain factor, is then added to the input image. This method is depicted in Figure 4.11. Here, $x_r$ denotes the raw image, $x_e$ the enhanced image, $h$ the low-pass filter used to obtain the unsharp mask and $\gamma$ a gain factor. In general, the gain factor will have a value of around 5. However, if so desired, it can be adjusted by the user. Let $h_{enh}$ denote the effective enhancement filter, i.e., $x_e(n) = (h_{enh} * x_r)(n)$ for all $n$, and let $\hat{h}_{enh}$ denote the

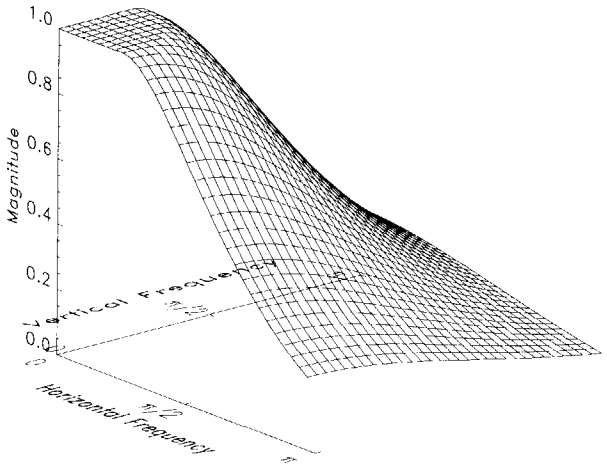



**Figure 4.10:** *Modified CSF for a range of viewing distances.*

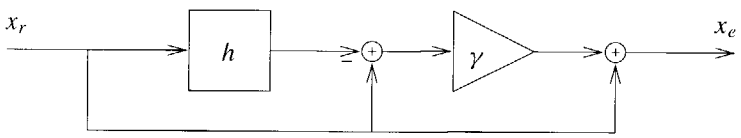



**Figure 4.11:** *Enhancement method using unsharp masking.*

Fourier transform of $h_{enh}$. Figure 4.12 shows an example of a two-dimensional magnitude response $|\hat{h}_{enh}|$ where $h(k,l) = \frac{1}{5}$ for all $k, l = 1, \ldots, 5$, and $\gamma = 5$. It is clear that this enhancement is in fact an emphasis of the middle and high spatial frequencies.

In coded images, the enhancement not only emphasizes the relevant medical information, but the coding noise as well. Steps should be taken to ensure that coding artifacts, which were below the threshold of visibility in the decoded raw images, do not become visible in the enhanced version of these images. Of course, one way of solving this problem is to perform the enhancement before the images are coded. That way, the decoded data are suitable for viewing, but are not in the correct format for the quantitative measurements such as vessel diameter. Moreover, as different physicians prefer different settings of the image enhancement

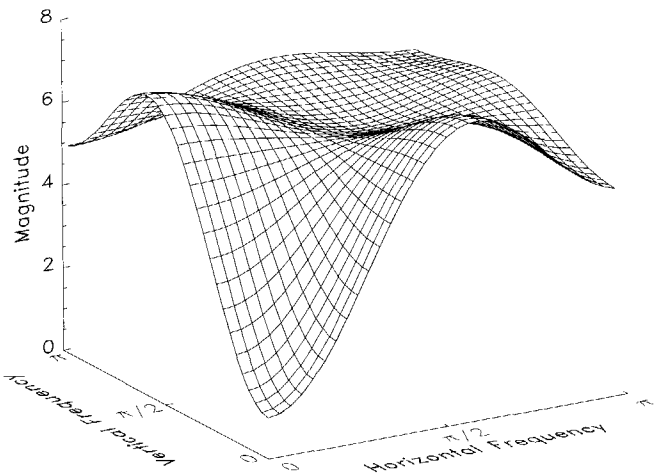



**Figure 4.12:** *Magnitude response* $|\hat{h}_{enh}|$.

(different gain factors), the non-enhanced data should be compressed. The compression method, therefore, needs to be adapted to the enhancement which may be applied later.

The overlapped transform coding method is well suited to adapt the quantization to the image enhancement. Since it divides the spatial frequency spectrum into a large number of narrow frequency bands ($K = 32$), we can quantize the frequency bands that will be enhanced later on more accurately, i.e. with a smaller step size than those that will not be enhanced. This can be achieved by applying an *enhancement weighting*, in addition to the perceptual weighting. This enhancement weighting can be calculated as follows.

Assume that we apply the enhancement to the images before they are coded. In that case, due to the associative property of the linear convolution, the filter bank channel signals $u_{k,l}$ are given by

$$\begin{aligned} u_{k,l} &= x_e * h_{k,l} \\ &= (x_r * h_{enh}) * h_{k,l} \\ &= x_r * \left(h_{enh} * h_{k,l}\right). \end{aligned}$$

Hence, the transform coefficients can be calculated by convolving the raw input data with an effective filter $h_{enh} * h_{k,l}$. Next, consider the coding of raw

images using the enhancement weighting. Here the weighting coefficients, say $\eta_{k,l}$, $k, l = 1, \ldots, K$, should be chosen such that the weighted transform coefficients approximate the transform coefficients obtained by coding the enhanced data. For this reason, we minimize

$$\Phi_{k,l} = \|\eta_{k,l} h_{k,l} - h_{enh} * h_{k,l}\|_{\alpha}^2, \quad \text{for all } k, l = 1, \ldots, K, \tag{4.1}$$

for some suitable choice of $\alpha$. We choose $\|\cdot\|_{\alpha}$ to be the Frobenius norm, which results in a least-square solution.

Let $\tilde{\eta}_{k,l}$, $k, l = 1, \ldots, K$, denote the weighting coefficients which minimize (4.1). By setting the derivative with respect to $\eta_{k,l}$ on the right-hand side of (4.1) to zero, we have

$$\left.\frac{\partial \Phi_{k,l}}{\partial \eta_{k,l}}\right|_{\eta_{k,l}=\tilde{\eta}_{k,l}} = 2\langle h_{k,l}, \tilde{\eta}_{k,l} h_{k,l} - h_{enh} * h_{k,l}\rangle = 0,$$

from which we conclude that

$$\tilde{\eta}_{k,l} = \frac{\langle h_{k,l}, h_{enh} * h_{k,l}\rangle}{\|h_{k,l}\|_F^2}, \tag{4.2}$$

for all $k, l = 1, \ldots, K$. Note that for separable signal transforms we have $\|h_{k,l}\|_F = \|h_k\|_2 \|h_l\|_2$, where $h_k$ and $h_l$ are the one-dimensional filters constructing $h_{k,l}$.

The resulting enhancement weighting function for the LOT described in subsection 4.3.1, obtained by calculating (4.2) for all $k, l$, is shown in Figure 4.13. As expected, its shape is similar to that of the frequency response of the enhancement filter. Obviously, we can combine the enhancement weighting and the perceptual weighting of the transform coefficients without any additional complexity being introduced. By doing so, we obtain the combined weighting function $W$, given in Figure 4.14.

### 4.3.3 Lossless coding

The purpose of the lossless coding is twofold. The first function is to remove redundancy in the quantizer-output symbols. The quantizer-output symbols are highly redundant since the quantizer maps the incoming sequence of transform coefficients onto a smaller number of output symbols, which results in a peaked probability density function. This redundancy can be eliminated by mapping the quantizer-output symbols onto another, more efficient, representation. The mapping is reversible (lossless), i.e., the quantizer-output symbols can be exactly reconstructed in the decoder. The second function is to map the quantizer-output symbols onto binary codewords that are transmitted or stored.

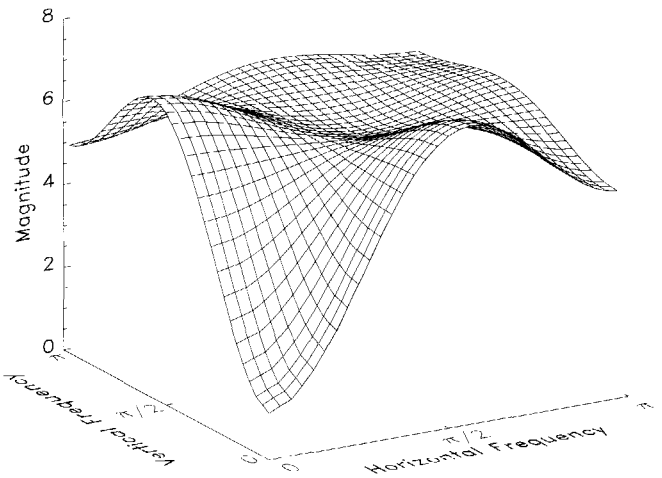



**Figure 4.13:** *Enhancement weighting.*

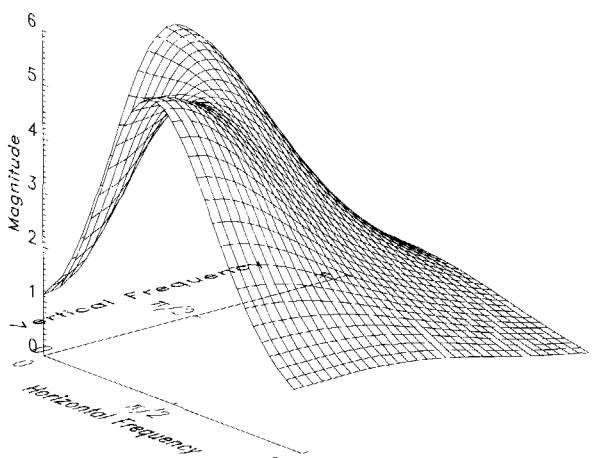



**Figure 4.14:** *Total combined perceptual and enhancement weighting function.*

The quantizer-output symbols are coded block-by-block, i.e., each block of $K \times K$ transform coefficients is treated independently. Within a block, the symbols are first converted from a doubly-indexed sequence to a singly-indexed sequence by using a *scanning pattern*. We have adopted the frequently used *zig-zag* scanning pattern [26]. Another scanning pattern, for example, is the Hilbert or Peano scanning pattern [94, 95]. The scanned quantizer-output symbols are coded by using a combination of *variable-length coding* (VLC) and *run-length coding* (RLC) which, in principle, is equivalent to the method used in the MPEG image-coding standard [8, 9]. It works as follows.

Let $q = (q_k)_k$ denote the singly-indexed sequence of quantizer-output symbols. The first element, which corresponds to the DC information within a block, is coded separately using a fixed-length binary codeword. The actual length depends on the original pixel resolution and the first basis function of the signal transform. The run-length coding then converts the remaining elements to a sequence of triples $(r_n, m_n, s_n)$, where $r_n$ denotes the length of a concatenation of zero-valued symbols (called the *run-length*), $m_n$ denotes the magnitude of the first non-zero symbol following these zero-valued symbols and $s_n$ denotes the sign of this non-zero symbol. The last element of the sequence is a special symbol (called the *end-of-block* (EOB) symbol), indicating that the last non-zero coefficient has been encountered in the block. For example, let's assume that the sequence of quantizer-output symbols $q$ is given by

$$q = (127, 80, -20, 0, 5, 0, 0, 0, 1, 0, \ldots, 0).$$

The first element (with value 127) is fixed-length coded. The remaining elements are converted by the run-length coding to a sequence of triples $(r_n, m_n, s_n)$, terminated by the EOB symbol, as

$$((0, 80, 1), (0, 20, 0), (1, 5, 1), (3, 1, 1), \text{EOB}),$$

were $s_n = 1$ indicates a positive value and $s_n = 0$ a negative value. The variable-length coding maps the sequence of triples (including the EOB symbol) onto variable-length binary codewords originating from a two-dimensional variable-length coding table which has entries for the run-length and the magnitude. The sign is separately coded with one bit following the variable-length codeword.

The number of possible combinations of run-lengths and magnitudes is very large. Codewords are given in the table only for the most frequently occurring combinations. For other combinations a special *escape* (ESC) codeword, which is part of the table, is generated, followed by the fixed-length binary coded run-length, magnitude and sign. Table 4.1 shows an example of the format of the

| $(r_n, m_n)$ | codeword |
|---|---|
| (0,1) | 0 |
| (0,2) | 100 |
| (0,3) | 101 |
| (1,1) | 1110 |
| (1,2) | 11010 |
| (2,1) | 11011 |
| EOB | 1100 |
| ESC | 1111 |

**Table 4.1:** *Example of a two-dimensional Huffman table.*

variable-length coding table assuming that only 8 codewords are used. The number of bits needed to represent the run-length depends on the block size of the transformation. For example, if the block size is $32 \times 32$, each block contains $2^{10}$ symbols so that 10 bits are needed to represent the run-length. The number of bits needed to represent the magnitude again depends on the original pixel resolution and the basis functions of the signal transform. In addition to the quantizer-output symbols, the step sizes used during the quantization are coded in the bit stream.

Since we assign short codewords to quantizer-output symbols with a high probability and large codewords to those with a low probability, the bit rate per coded image varies from image to image. Where it is necessary to produce a constant number of bits per image, we can design some controlling mechanism which ensures that, within a certain degree of accuracy, each image is encoded with the required number of bits by varying the step size over the images [96, 97, 98, 99]. A bit-rate control mechanism of this kind is based on a feedback control system in which the step size is updated from block to block, depending on the number of bits actually produced.

A disadvantage of varying step sizes within an image is that some parts of the decoded image may contain more clearly perceivable coding artifacts than others. To avoid this "inhomogeneous distribution" of the coding artifacts, the control systems aim to minimize the variation in the step size as much as possible.

To ensure that the control system is fast and accurate, the number of blocks of $K \times K$ transform coefficients should be sufficiently large [98]. In our transform-coding system, we use fairly large blocks of transform coefficients ($32 \times 32$ coefficients). An image of $512 \times 512$ pixels, for example, therefore contains only 256 blocks, which has proven to be insufficient for proper control. In [100] this

problem is solved by dividing each block of transform coefficients into, say $n$, smaller segments and updating the step size from segment to segment. Computer simulations have shown that good results can be obtained for $n \geq 4$. An almost constant step size can be obtained with this method and the required bit rate can be accurately achieved.

## 4.4 Computer simulations

In this section we consider computer simulation results of the compression method described in the previous section.

### 4.4.1 Experimental set-up

A large number of X-ray images has been compressed at compression ratios in the range of 8 – 32. The image material originated from various medical centres, viz. the Hôpital Cardiologique BP Lyon Monchat, France, the Hôpital Cardiologique, Service Hémodynamique, CHRU de Lille, France and the University of Texas, Hermann Hospital, Cardiology Division, Houston, USA. Compression was applied to the raw images. The bit-rate control method described in [100] was used to ensure that a given compression ratio was indeed accurately achieved.

Original and decoded images were compared using a 17 inch medical black-and-white monitor at a viewing distance of three times the screen height or more. Before viewing, the images were first enhanced with an enhancement filter as described in subsection 4.3.2.

We compared our results with those obtained using the conventional non-overlapped DCT based coding methods JPEG and MPEG-1. For JPEG we used the baseline version. For MPEG we chose groups of pictures (GOPs) of length two, each consisting of one I and one P picture. This means that with MPEG-1 two images are encoded in combination, so that only groups of two images can be retrieved and decoded as a unit. MPEG is actually a standard for coding colour video signals consisting of one luminance component and two colour-difference components. We have simply assumed that the colour components are equal to zero and coded our images as the luminance component. The compression ratio we used in the comparison was 8, 12 and 16.

We used the *peak signal-to-noise ratio* (PSNR or simply SNR) as an objective quality measure. Let $x = (x_{k,l})_{k,l=1}^{N}$ and $\hat{x} = (\hat{x}_{k,l})_{k,l=1}^{N}$ denote the original and decoded image, respectively, and let $e = x - \hat{x} = (e_{k,l})_{k,l=1}^{N}$ denote the

coding noise introduced by the compression. Assuming that $e$ is a zero-mean noise process, the variance, say $\sigma_e^2$, is estimated by

$$\sigma_e^2 = \frac{1}{N^2} \sum_{k,l=1}^{N} |e_{k,l}|^2.$$

The SNR is then defined by

$$\text{SNR} = 10^{10} \log \left( \frac{255^2}{\sigma_e^2} \right) \quad \text{(dB)}.$$

### 4.4.2 Results

In this subsection we discuss results obtained by computer simulations of the overlapped transform-coding method. The observations described here are those of the author and several of his colleagues, who include a number of medical physicists. At present, more carefully defined perceptual evaluations are being performed in the clinical environment, with radiologists and cardiologists acting as viewers. The preliminary results of these evaluations will be described here. A more detailed description will be published in the near future.

#### Performance of the overlapped transform system

We observed that the type of artifacts introduced by our overlapped transform-coding method was completely different from the type of artifacts introduced by JPEG and MPEG. At compression ratios higher than about 8, JPEG and MPEG introduce blocking artifacts, i.e. individual blocks of $8 \times 8$ pixels become clearly visible and small details inside the blocks, such as small blood vessels, disappear or are inaccurately represented. Our overlapped transform-coding system does not introduce blocking artifacts. At large compression ratios (greater than 16), however, some change in the characteristics of the noise which is present in the original images, can be observed. Furthermore, small details become a little unsharp.

Figures 4.18, 4.19, 4.20 and 4.21 show the decoded versions of the original image depicted in Figure 4.4, at compression ratios of 8, 12, 16 and 32, respectively. The image shown in Figure 4.4 is one out of a sequence (52 images) recorded at the Hôpital Cardiologique Lille. The images in this sequence measured 512 lines with 512 pixels each, where each pixel is represented by 8 bits, and has an image rate of 12.5 Hz. The corresponding difference images are shown

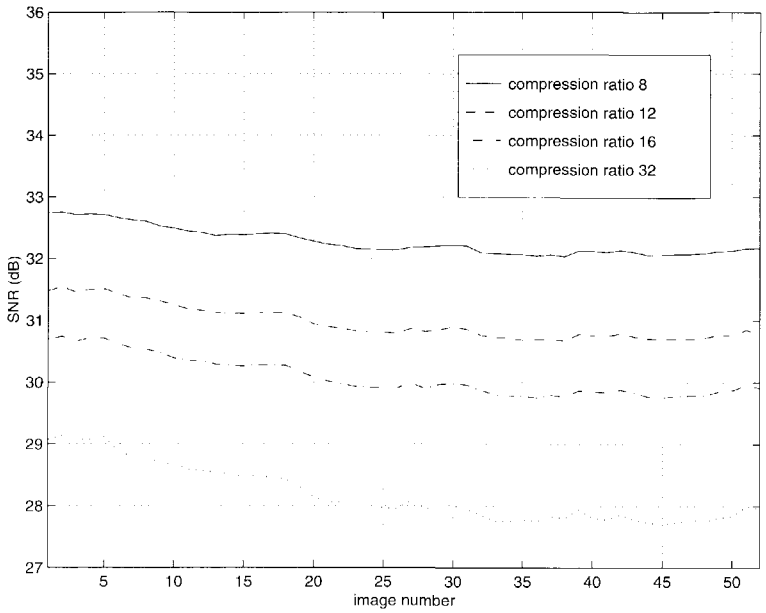



**Figure 4.15:** *SNRs of the coded and enhanced X-ray cardio-angiographic image series at compression ratios 8, 12, 16 and 32.*

in Figures 4.22, 4.23, 4.24 and 4.25. To make the coding errors more visible, the differences have been multiplied by 5 and a value of 128 (middle grey) has been added. Figure 4.15 shows the SNRs of the complete decoded and enhanced sequence at compression ratios of 8, 12, 16 and 32.

The image quality perceived at different compression ratios is as follows. At compression ratios up to about 12, the overlapped transform-coded images can hardly be distinguished from the original images. At a compression ratio of 16, the artifacts described above start to become visible, but the perceived quality is still good. We also observed that the image quality degrades very gradually as the compression ratio increases; the images still look reasonable at a compression ratio of 32.

### Overlapped transform coding versus JPEG and MPEG

In order to make a significant comparison between the overlapped transform method and the non-overlapped methods JPEG and MPEG, we also applied an

enhancement weighting in the DCT based systems. At a compression ratio of 8, the MPEG-coded images look reasonably good, whereas some blocking artifacts can already be observed in the JPEG-coded images. At compression ratios of 12 and 16, both the JPEG-coded and the MPEG-coded images clearly show blocking artifacts. These artifacts are most apparent in the JPEG-coded images. If the default viewing conditions were changed, by applying zoom for instance, the MPEG and JPEG-coded images look poor: the blocking artifacts are emphasized, the images look blurred and the contours of large vessels are not very well defined and would be difficult to determine in a quantitative coronary analysis. The zoomed overlapped transform-coded images are of good quality, but slightly lacking in definition. Figures 4.26 and 4.27 show the decoded images when the JPEG and MPEG coding method is used, respectively, at a compression ratio of 12, and Figures 4.28 and 4.29 show the corresponding difference images (again the coding errors have been multiplied by 5 and a value of 128 (middle grey) has been added). Figure 4.16 shows us the SNRs of the complete decoded and enhanced sequence at the same compression ratio for the JPEG, MPEG and overlapped transform-coded method.

### Influence of the enhancement weighting

In order to determine how well the performance of the coding system using the enhancement weighting approximates the performance of the coding system in which the input data are enhanced before being coded, we conducted the following experiments. In one experiment we performed the enhancement before the images were coded. We shall refer to this situation as "pre-enhancement". We compared the outcome of this experiment with the outcome of an experiment in which the raw data are coded using the enhancement weighting, and enhanced afterwards. We shall refer to this situation as "post-enhancement". Note that the pre-enhancement configuration is optimal in the sense that coding errors which are below the threshold of visibility in the decoded images do not become visible after applying the image enhancement, as is the case with the post-enhancement configuration.

In a practical situation, the reconstructed images are stored on a storage medium. This implies that the image decoding is followed by an additional quantization which maps the reconstructed (floating-point) results onto the original pixel resolution. This quantization introduces additional noise in the image. In the post-enhancement configuration, this additional noise will be amplified by the enhancement filter. To see what influence this has on the SNR and the image

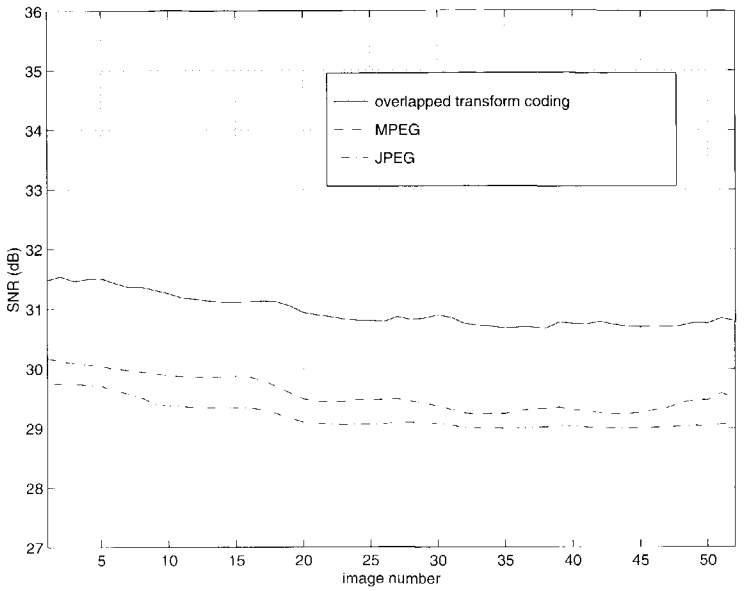



**Figure 4.16:** *SNRs of the coded and enhanced X-ray cardio-angiographic image series using the JPEG, MPEG and overlapped transform coding methods at a compression ratio of 12.*

quality, the post-enhancement experiment was performed both with and without mapping onto the 8 bits per pixel resolution. In the latter case, this means that the floating-point data produced by the synthesis mapping were directly enhanced. We will refer to this configuration as the floating-point post-enhancement configuration, or flt. post-enhancement, for short. To summarize, we considered the following cases:

- pre-enhancement,
- post-enhancement using enhancement weighting with additional mapping to 8 bits/pixel,
- post-enhancement using enhancement weighting without additional mapping to 8 bits/pixel.

Figure 4.17 shows the SNRs of the decoded images in the test sequence at a reduction factor of 12. The use of post-enhancement together with the enhancement

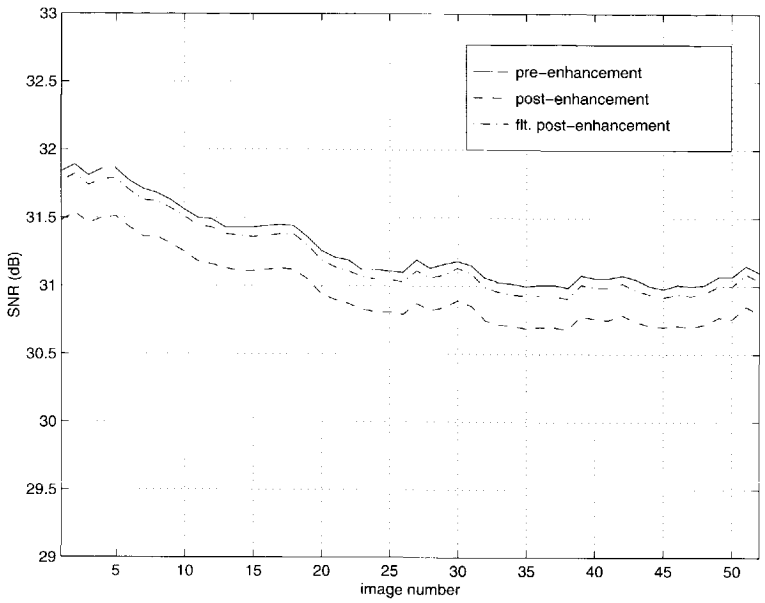



**Figure 4.17:** *SNRs of the coded X-ray cardio-angiographic image series using different enhancement methods at a compression ratio of 12.*

weighting approximates the pre-enhancement configuration very closely. When mapped onto the original pixel resolution, however, the SNR of the enhanced images decreases compared with the optimal situation. It should be noted that this difference becomes relatively small for high compression ratios. Indeed, the total coding noise $e$ can be thought of as the combination of "real" coding noise with variance, say $\sigma_c^2$, and noise produced by the additional mapping with variance, say $\sigma_m^2$. Allowing for the two noise components, the variance of the total coding noise becomes

$$\sigma_e^2 \leq \sigma_c^2 + \sigma_m^2,$$

with equality if and only if both noise sources are statistically independent. With this, the SNR of the decoded images can be expressed as

$$\mathrm{SNR} \geq 10^{10} \log \left( \frac{255^2}{\sigma_c^2 + \sigma_m^2} \right).$$

It is clear that the influence of the errors introduced by the mapping will decrease with increasing reduction factors.

To determine the perceptual influence of the introduced quantization noise, we compared the pre-enhanced sequence with the post-enhanced sequences. Figures 4.30, 4.31 and 4.32 show the results (enlarged) for the pre-, post- and flt. post-enhancement scheme at a reduction factor of 12. Although the SNR plots would suggest that the flt. post-enhancement is significantly better than the post-enhancement with the additional mapping, the photographs show that this effect is hardly visible. However, at very high compression ratios (greater than 40), these errors become clearly visible as contouring. This phenomenon will only occur in flat areas since in these regions almost all grey values will be mapped onto the same quantization level before being enhanced.

### Clinical evaluations

In the ongoing clinical evaluations [101], images compressed with the overlapped transform coding method have been presented to radiologists and cardiologists. The compression ratio used was 12. The viewing sessions took place in the normal clinical environment. Of the set of patient images reviewed in a single session, some were original, some were coded and the observers were not informed which was which. The observers were requested to do the same diagnostic task as they perform in clinical practice in their cardiology departments with both still and moving images. They were free to review each image series at their own pace, to move forward and backward and change the speed. Changes in the post-processing, such as increasing the default edge enhancement, applying zoom, changing the contrast and brightness, were not allowed. There was no significant difference found in the interpretations based on original and the overlapped transform-coded angiograms. The variability in the diagnosis introduced by the fact that different observers have different professional judgements is significantly greater than the variability introduced by the compression. This suggests that higher compression ratios may also be appropriate.

## 4.5 Conclusions

We have investigated the potential of overlapped transform coding for lossy data compression of X-ray cardio-angiographic image series. We observed that overlapped transform coding does not introduce any blocking artifacts, in contrast to conventional non-overlapped transform coding. With non-overlapped transform coding, blocking artifacts start to become visible in enhanced images at compres-

sion ratios of about 8. With overlapped transform coding, however, the original images can hardly be distinguished from the coded images at compression ratios up to about 12. At higher compression ratios, the perceived quality is still high, but some change in the noise structure that is usually present in the original image, and in small details such as small blood vessels, can be observed. Moreover, since the overlapped transform coding method divides the spatial frequency spectrum into a large number of narrow frequency bands ($K = 32$), it is particularly well suited to adapt the compression to the properties of the human visual system and to post-processing by quantizing the frequency bands to a different degree.

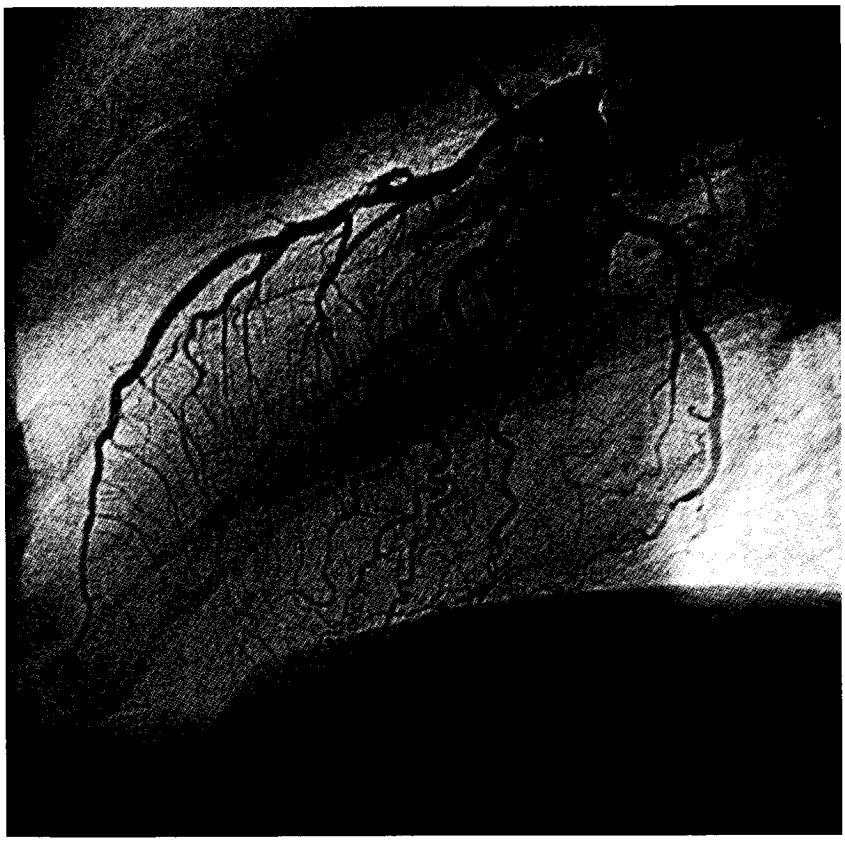


**Figure 4.18:** *Coded, enhanced X-ray cardio-angiographic image at a compression ratio of 8 (first encoded and decoded, then enhanced).*

**Figure 4.19:** *Coded, enhanced X-ray cardio-angiographic image at a compression ratio of 12 (first encoded and decoded, then enhanced).*

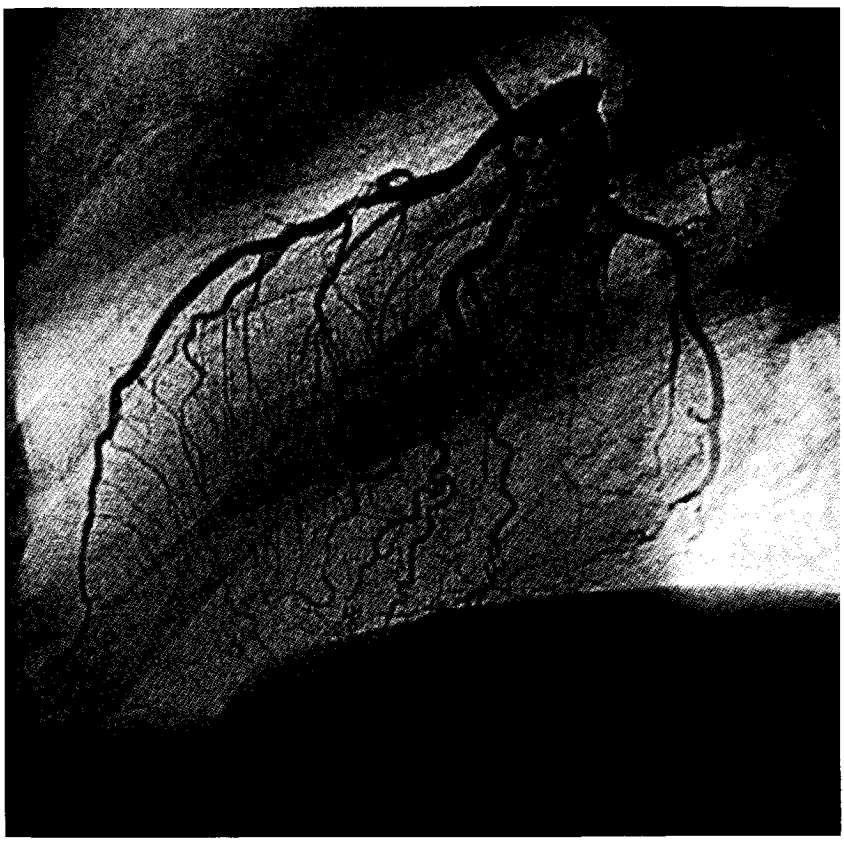


**Figure 4.20:** *Coded, enhanced X-ray cardio-angiographic image at a compression ratio of 16 (first encoded and decoded, then enhanced).*

**Figure 4.21:** *Coded, enhanced X-ray cardio-angiographic image at a compression ratio of 32 (first encoded and decoded, then enhanced).*

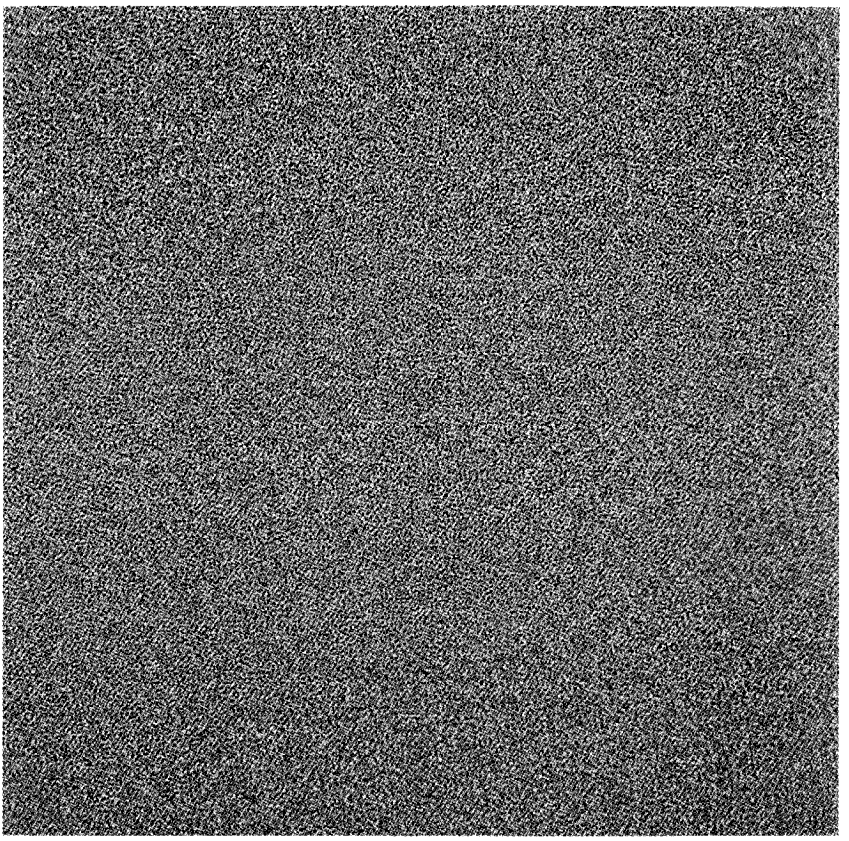


**Figure 4.22:** *Difference image at a compression ratio of 8 (scaled).*

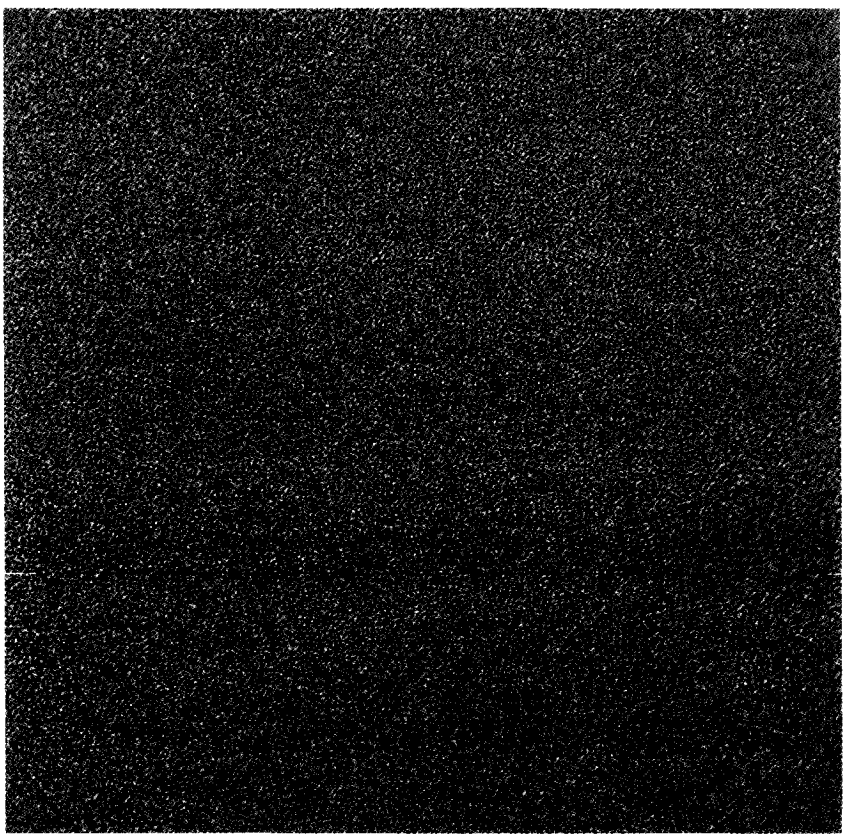


**Figure 4.23:** *Difference image at a compression ratio of 12 (scaled).*

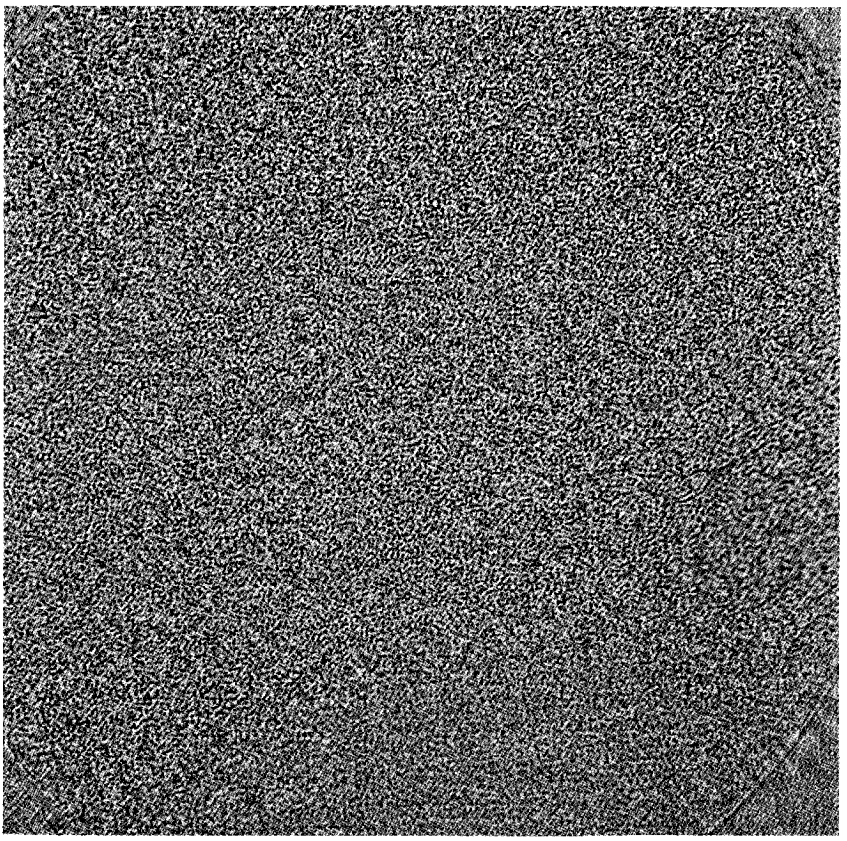


**Figure 4.24:** *Difference image at a compression ratio of 16 (scaled).*

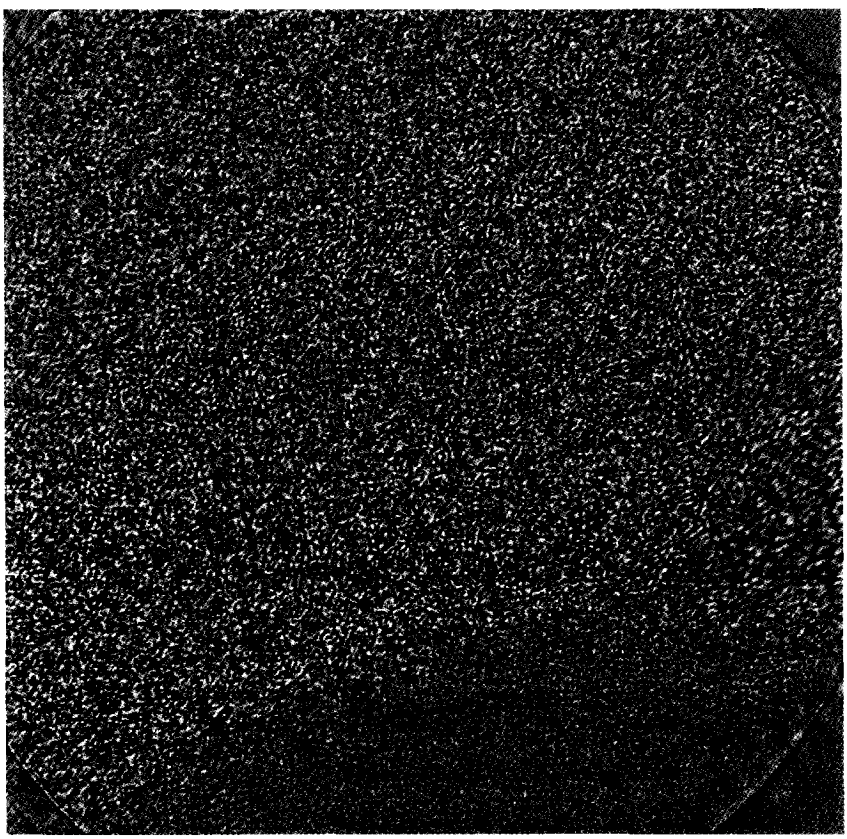


**Figure 4.25:** *Difference image at a compression ratio of 32 (scaled).*

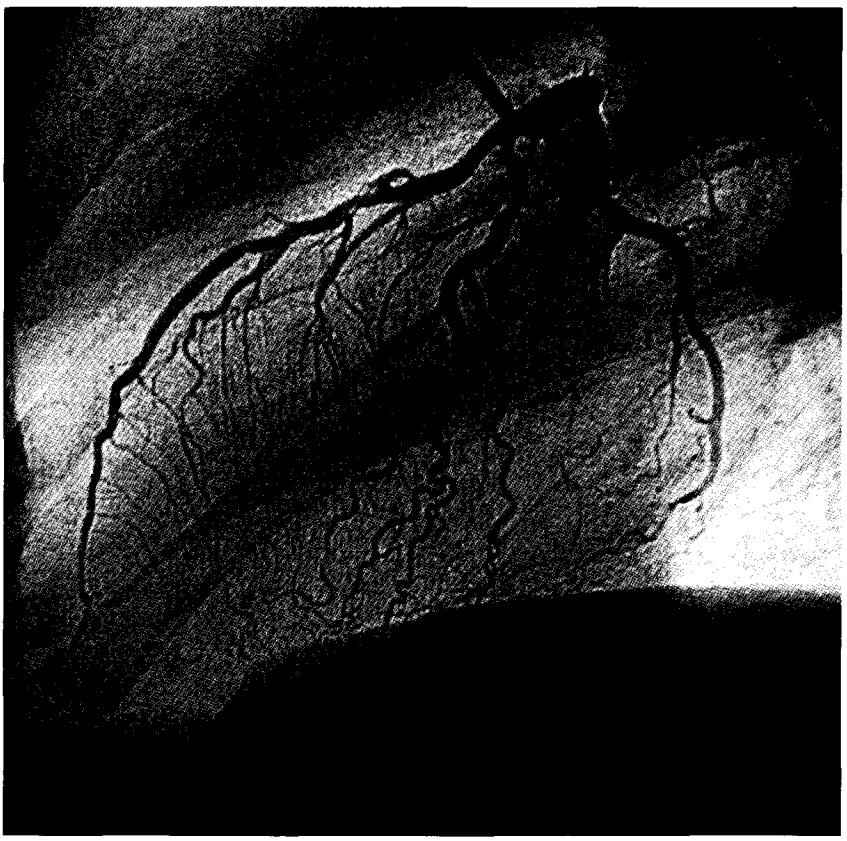


**Figure 4.26:** *JPEG coded, enhanced X-ray cardio-angiographic image at a compression ratio of 12 (first encoded and decoded, then enhanced).*

**Figure 4.27:** *MPEG coded, enhanced X-ray cardio-angiographic image at a compression ratio of 12 (first encoded and decoded, then enhanced).*

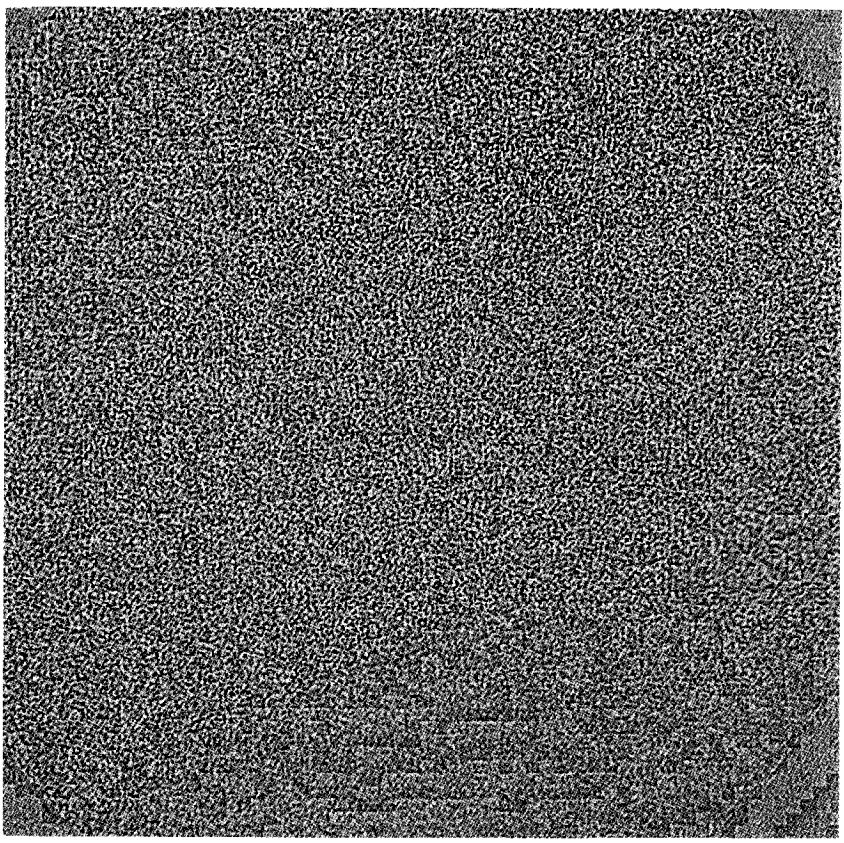


**Figure 4.28:** *Difference image of the JPEG coded image at a compression ratio of 12 (scaled).*

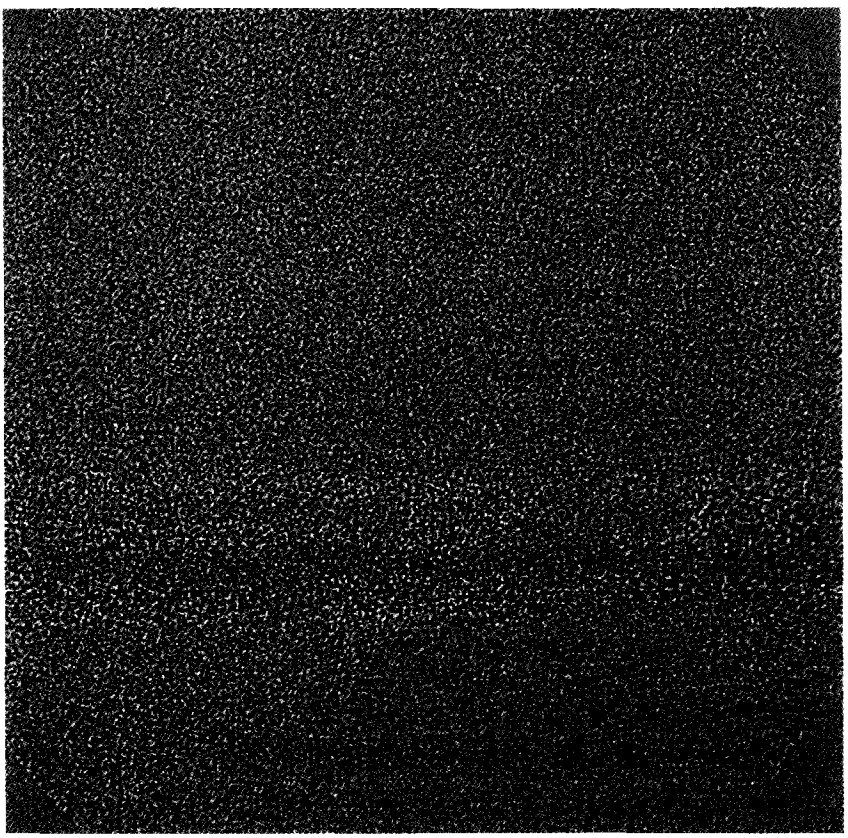


**Figure 4.29:** *Difference image of the MPEG coded image at a compression ratio of 12 (scaled).*

**Figure 4.30:** *Coded, pre-enhanced X-ray cardio-angiographic image (enlarged) at a compression ratio of 12.*

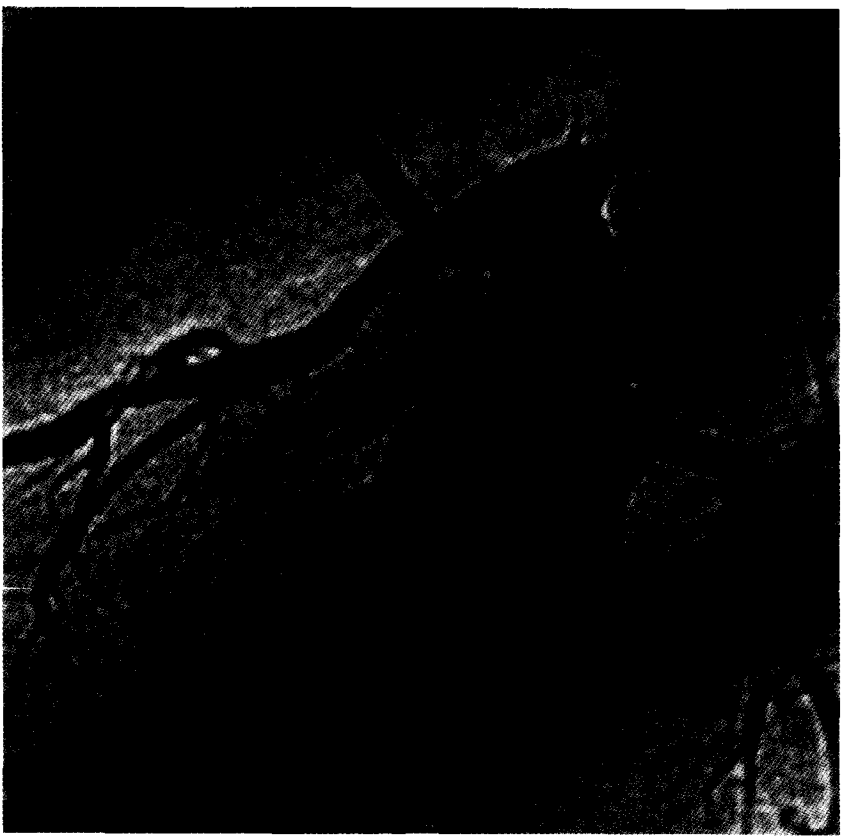


**Figure 4.31:** *Coded, post-enhanced X-ray cardio-angiographic image (enlarged) with enhancement weighting at a compression ratio of 12.*

**Figure 4.32:** *Coded, flt. post-enhanced X-ray cardio-angiographic image (enlarged) with enhancement weighting at a compression ratio of 12.*

# Chapter 5

# *Realizations for discrete-time signal transforms*

**Contents**



## 5.1 Introduction

In the previous chapters we touched upon the design of analysis and synthesis filters for data compression of images and we concluded that LOTs were suitable for this application. In this chapter we concentrate on the realization of these filters, in particular on realizations which are suitable for *very large scale integration* (VLSI) technology. Rather than designing realizations for these transforms directly, we will focus on the computing of

$$y = Ax,$$

where $A \in \mathbb{C}^{m \times n}$, $x \in \mathbb{C}^n$ can be arbitrary. After showing how we can deal with this problem, we will apply this solution to the LOT itself, to its constituent matrices $A_e$, $A_o$ and $B_a$, in particular (see Chapter 3).

In this chapter we try to find realizations that behave *optimally* in the sense that they exhibit low sensitivity to perturbations in the coefficients. However, realizations that are insensitive to coefficient perturbations do not need to be truly optimal. It might very well turn out that this notion of optimality is not appropriate from the point of view of implementation. A realization which is optimal in the sense that it is numerical robust is not necessarily efficient in terms of implementation, i.e., its computational complexity or its silicon area in the case of a VLSI implementation may be too high. We show that, by considering *both* algorithm *and* architecture design simultaneously, we can design numerically robust and implementation-efficient realizations of filter banks. Moreover, such realizations can be made fully programmable in the sense that the architecture can be used for arbitrary signal transforms, including the DCT, the LOT or discrete wavelet transforms. We argue that established structures, in particular butterfly structures (the perfect shuffle architectures), used to compute Fourier-like transforms such as the DCT, are not naturally the "best possible" structures as is commonly claimed and believed. Butterfly structures have emerged from the fast transforms which require only $\frac{m}{2} \log m$ multiplications instead of $m^2$ for a size $m \times m$ signal transform. This is a substantial saving in computation effort for almost all practical values of $m$ (typically $m \geq 8$). However, as the transforms commonly used are unitary, large word lengths are needed to ensure that the finite precision transform preserves this desirable property. We claim that there are alternative structures which are low-cost in terms of the number of operations and the implementation complexity of the operations.

### Organization of this chapter

This chapter is organized as follows. For the purpose of selecting an algorithm which admits a suitable realization, in Section 5.2 we will investigate the behaviour of different realizations and their corresponding implementations. We first provide a precise definition of *optimal realizations*. We show that (minimal) *orthogonal realizations* are optimal according to that definition and that they are obtained through a factorization of $A$ into elementary Givens *rotation* matrices. Next we turn to the VLSI architectures to which these realizations can be mapped. It is known that the Givens rotation can be implemented in CORDIC arithmetic. The CORDIC computation technique was originally introduced by Volder [102]

and later generalized by Walter [103]. CORDIC performs a plane rotation over an angle $\theta = \sum_{i \in \mathcal{I}} \sigma_i \theta_i$ in which $\sigma_i \in \{-1, 1\}$ and $(\theta_i)_{i \in \mathcal{I}}$ is a fixed-length sequence of shift-add operations. The length $|\mathcal{I}|$ depends on the precision required and the chosen set of shift-adds. For example, a rotation over an angle $\theta$ with $s$ bits accuracy takes $s + 1$ elementary shift-add stages [102, 103, 104]. However, CORDIC arithmetic can be more broadly defined by adding to the range $0 \leq |\theta| \leq \pi$ of angles a discrete set of feasible angles which admit simple rotations, called *micro-rotations* or *$\mu$-rotations*, which were first introduced in [105, 106] to solve the symmetric eigenvalue problem.

For $\mu$-rotation-based realizations to be feasible, it is necessary to first identify a suitable, numerically robust algorithm to compute $y = Ax$. This is done in Section 5.3. We propose an algorithm which is based on the RQ-factorization of a properly chosen matrix, say $X \in \mathbb{C}^{m \times n}$, $m \leq n$. By properly chosen, we mean that $XX^*$ is a multiple of the identity operator. In Section 5.4 we show how we can apply this algorithm to the analysis and synthesis maps of a transform coding system. We discuss both analysis and synthesis operator realizations and we derive the conditions under which both operators can be given the same realization. The RQ-factorization is commonly computed using a concatenation of Givens rotation matrices [107, 108]. The purpose of the rotations in the RQ-algorithm is to nullify the super-diagonal entries of $X$ to reduce it to a lower-triangular form. This is usually done by first nulling the super-diagonal entries of the last column of $X$, then those of the second-last column, and so forth until a lower-triangular form is eventually reached. Note that if $XX^*$ is diagonal, the resulting lower-triangular matrix will also be diagonal. The use of $\mu$-rotations instead of exact Givens rotations, however, may lead to a different ordering in which the super-diagonal entries of $X$ are zeroed. Indeed, since the set of feasible angles is a subset of $[0, \pi]$, entries of $X$ can only be made approximately zero and the accuracy depends on the entry's value. In Section 5.5 we propose an iterative algorithm to compute the RQ-factorization of $X$. The idea behind this algorithm is that we apply a sequence of rotations to $X$ with the property that each new $X$ is "more diagonal" than its predecessor. Eventually, the off-diagonal entries of $X$ are small enough to be declared zero. In Section 5.6 we discuss the architecture imposed by this algorithm, in which the realization of LOTs can be mapped. We discuss how to increase the efficiency of the VLSI implementation by introducing concurrent processing and designing LOTs which perfectly match hardware constraints, i.e., LOTs which are built on $\mu$-rotations. The use of such transforms can save more than 25% of the total silicon area.

In Section 5.7, we make some concluding remarks on the final VLSI imple-

mentation. Here, we discuss complexity issues such as layout and total silicon area for the implementation of the LOT, and compare these results with alternative implementations, viz., a conventional butterfly-based VLSI implementation and an implementation based on a TriMedia digital signal processor. Finally, in Section 5.8, we draw some conclusions.

## 5.2 Optimal realizations

### 5.2.1 Introduction

Digital filters can be realized in a variety of structures. Which realization to use may depend on various measures of performance. Thus the cost of the realization measured in terms of the number of multiplications may be critical. Another measure may be the sensitivity of the input-output map with respect to perturbations in the realization parameters, as different realizations can produce markedly different performances due to errors incurred by the use of finite word-length registers. To investigate these issues in more detail, it is necessary to know how the actual computation of the output is achieved. This includes knowing what internal variables are used and in what order what variables are stored for later use, and other detailed information about the computational structure. One convenient model for analysing filter structures is the *signal flow graph* (SFG).

A signal flow graph is a network of interconnected operations which takes an input signal at its input terminal and computes the output signal delivered by the output terminal. For shift-invariant SFGs, the relation between output and input signals can be expressed by means of the network's transfer function. Thus a filter SFG consists of nodes and directed branches. The branches are commonly labelled with simple branch maps, such as the unit delay with a transfer function $z^{-1}$ and multiplication constants with a transfer function equal to that gain. Figure 5.1 shows a simple example where the corresponding transfer function is given by $H(z) = h_0 + h_1 z^{-1} + h_2 z^{-2}$.

A rational function $H(z)$ of the form

$$H(z) = \frac{B(z)}{A(z)} = \frac{z^N(b_0 + b_1 z^{-1} + \cdots + b_q z^{-q})}{z^N(1 + a_1 z^{-1} + \cdots + a_p z^{-p})}, \tag{5.1}$$

where $z \in \mathbb{C}$, $N = \max(p, q)$ and $A(z) \neq 0$ in $|z| \geq 1$, is a transfer function of a (causal) discrete-time or digital filter with the impulse response

$$h(n) = \frac{1}{2\pi i} \oint_C H(z) z^{n-1} dz \quad \text{for all } n \in \mathbb{Z},$$

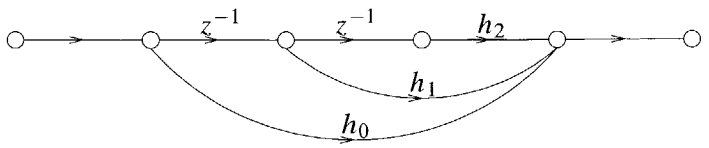



**Figure 5.1:** *A signal flow graph corresponding to the transfer function* $H(z) = h_0 + h_1 z^{-1} + h_2 z^{-2}$.

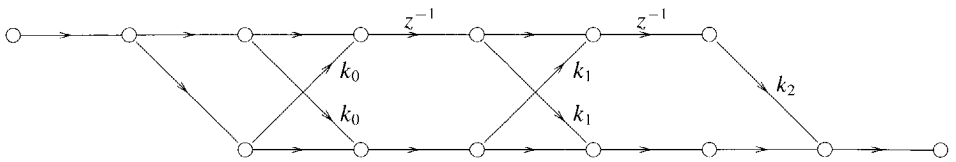



**Figure 5.2:** *A second realization of the transfer function* $H(z) = h_0 + h_1 z^{-1} + h_2 z^{-2}$.

where $C$ is any closed counterclockwise contour in $|z| \geq 1$ and $h(n)$ is absolutely summable. The integer $N$ in (5.1) is called the *degree* of $H(z)$.

A digital filter *realization* of $H(z)$ is any (finite) signal flow graph which provides a receipt for the computation of a signal $y$ from a signal $x$ satisfying $Y(z) = H(z)X(z)$, where $Y(z)$ and $X(z)$ are the $\mathcal{Z}$-transforms of the signals $y$ and $x$, respectively. A (digital) *implementation* of a digital filter realization is any *mapping of the realization on a physical (digital) processor*, be it a computer main frame, a mini-computer, a micro-computer, a signal processor or a VLSI chip. A given transfer function $H(z)$ has neither a unique realization nor a unique implementation. For example, Figure 5.1 and Figure 5.2 show two different realizations of the same transfer function $H(z) = h_0 + h_1 z^{-1} + h_2 z^{-2}$ where $h_0 = 1 + k_0$, $h_1 = k_1(1 + k_0)(1 + k_2)$ and $h_2 = (1 + k_0)k_2$. Also, both realizations can be mapped onto various processors and circuits.

In order to arrive at a useful definition for *optimal realization* of transfer functions, we will start with some preliminary definitions to facilitate the following discussion. We illustrate the definitions using the realizations shown in Figure 5.1 and Figure 5.2 as examples.

**Definition 5.2.1 (irreducible transfer function)** *A transfer function* $H(z) = B(z)/A(z)$ *is said to be irreducible if the polynomials* $B(z)$ *and* $A(z)$ *are rela-*

*tively prime or coprime, i.e., have no common zeros.*

**Definition 5.2.2 (minimal realization)** *A realization is minimal if the number of delay operators $z^{-1}$ is minimal, i.e., is equal to the degree of the corresponding irreducible transfer function.*

In our example, both the realization in Figure 5.1 and Figure 5.2 are minimal since the number of delays equals the degree of $H(z)$.

**Definition 5.2.3 (direct realization)** *A (minimal) realization is said to be direct if the parameters in the SFG are the coefficients of the (irreducible) transfer function.*

The realization in Figure 5.1 is a direct realization whereas the realization in Figure 5.2 is not.

**Definition 5.2.4 (canonical realization)** *A minimal realization is said to be canonical if the number of parameters in the SFG equals the number of coefficients in the (irreducible) transfer function.*

Note that minimal realizations may be canonical, without being direct realizations. For example, the SFG in Figure 5.1 is a direct canonical realization whereas the SFG in Figure 5.2 is canonical but not direct. Indeed, there is a one-to-one mapping $(h_0, h_1, h_2) \mapsto (k_0, k_1, k_2)$. We can use the freedom we have in selecting a realization for $H(z)$ to optimize criteria related to the implementation of $H(z)$. Implementation considerations can influence the structure of the algorithm or realization, and vice versa, the structure of the algorithm often suggests an implementation.

In general, the operations in digital filters are implemented in finite-precision arithmetic. The effect of parameter word-length limitation on the transfer function is of a *static* nature and only depends on the realization structure. However, there is also a *dynamic* word-length limitation effect which is caused by the *truncation* or *rounding* of the signals in the graph, in particular of the state variables that have to be stored in finite-length registers. A third finite-length register effect is the *limit-cycle* effect. Limit cycles are periodic signals (oscillations) that are sustained by the signal quantization non-linearities.

Realizations that are insensitive to parameter perturbations and that produce low output quantization noise, are not necessarily truly optimal. The reason is that, even in such cases, the complexity may turn out to be greater than strictly required.

By complexity, we mean implementation complexity, mainly size and dissipation in VLSI implementations. For example, if the realization in Figure 5.2 was be insensitive to perturbations of the multiplier coefficients $k_i$, then the multiplication cost could be reduced dramatically by substituting for $k_i$ the "nearest" value $\hat{k}_i$ which is in an almost trivial canonical signed-digit form.

We thus arrive at a useful definition of the notion of *optimal realization* of a transfer function, which is the following.

**Definition 5.2.5 (optimal realization)** *A realization will be said to be optimal if*

1. *it is minimal and canonical,*
2. *it exhibits low sensitivity to perturbations in the parameters and quantization of the internal signals,*
3. *it does not sustain any limit cycles due to feedback of both coarsely and granularly distorted signals,*
4. *it is suitable for low-cost high-accuracy VLSI implementation.*

Deriving optimal realizations satisfying the above mentioned properties is not easy. Very many of the commonly used realizations violate one or more of these optimality conditions. Direct implementations, for example, are not optimal because they require $\mathcal{O}(N^2)$ multiplications per output sample [60]. Furthermore, direct implementations of rational functions are numerically unstable, i.e. a small variation in the input can cause very large variations in the output, which requires large word lengths as a result [109]. We will return to this issue later. Notwithstanding these drawbacks, directly implemented transfer functions are attractive because of their simplicity and the way they match common multiply-accumulate operations in DSP architectures. Thus, although there are alternative non-direct realizations that come close to optimal, they mostly violate the fourth of the optimality conditions, i.e., their structure tends to prevent efficient mapping onto silicon. However, leaving the mapping issue aside for the moment, there is one class of realizations that deserves our attention because – as we shall show – it can be exploited to the utmost optimality. The realizations we are referring to are the class of *orthogonal realizations* which we will discuss in more detail in the next subsection.

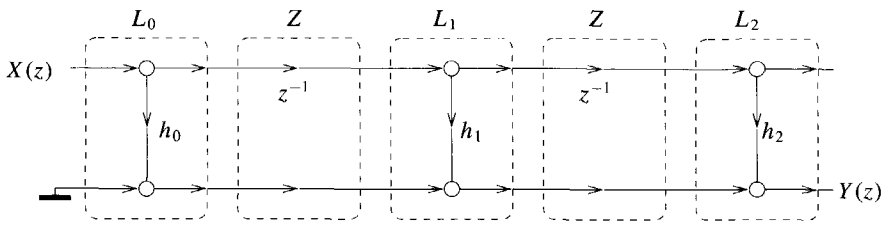



**Figure 5.3:** *The redrawn SFG of the direct realization.*

### 5.2.2 Orthogonal realizations

let us consider the example in Section 5.2, Figure 5.1. Remember that this is a direct canonical realization of the transfer function $H(z) = h_0 + h_1 z^{-1} + h_2 z^{-1}$. We can redraw this structure as shown in Figure 5.3. This SFG is clearly a cascade of elementary sections performing either a multiply-accumulate operation or a delay operation. It is this cascade that we now use for error analysis. The input-output map is

$$Y(z) = \begin{bmatrix} 0 & 1 \end{bmatrix} L_2 Z L_1 Z L_0 \begin{bmatrix} 1 \\ 0 \end{bmatrix} X(z),$$

where

$$L_2 Z L_1 Z L_0 = \begin{bmatrix} 1 & 0 \\ h_2 & 1 \end{bmatrix} \begin{bmatrix} z^{-1} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ h_1 & 1 \end{bmatrix} \begin{bmatrix} z^{-1} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ h_0 & 1 \end{bmatrix}.$$

Note that $L_0$, $L_1$ and $L_2$ are lower-triangular matrices and are, therefore, generally sensitive to perturbations in the coefficients $h_i$ (see Appendix D). In fact, the matrices $L_0$, $L_1$ and $L_2$ are elementary Gauss matrices. It is well known (see also [110, 111, 112]) that Gaussian elimination is an unstable method; Gaussian elimination can give arbitrarily poor results, even for well-conditioned problems. We conclude that such direct canonical realizations are sensitive and, therefore, not suitable for finite-precision arithmetic.

We can overcome the stability problems of the Gaussian elimination by choosing a realization which is based on a factorization into unitary and diagonal (scaling) matrices. One example of such a realization is shown in Figure 5.4, where $c_n$ and $s_n$, $n = 1, \ldots, 3$, denote $\cos(\theta_n)$ and $\sin(\theta_n)$, respectively. Note that this realization is minimal and canonical. The input-output map is

$$Y(z) = \begin{bmatrix} 0 & 1 \end{bmatrix} U_2 Z U_1 Z U_0 \begin{bmatrix} 1 \\ 1 \end{bmatrix} X(z),$$

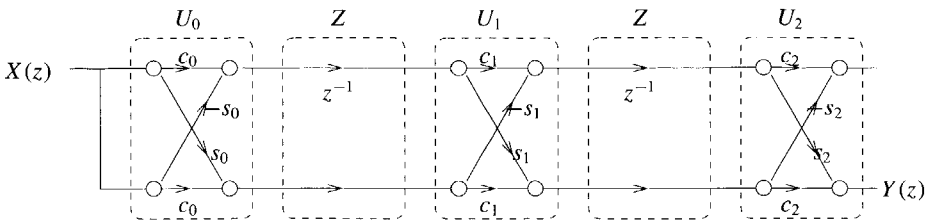



**Figure 5.4:** *An SFG based on unitary sections.*

where

$$U_2 Z U_1 Z U_0 = \begin{bmatrix} c_2 & -s_2 \\ s_2 & c_2 \end{bmatrix} \begin{bmatrix} z^{-1} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} c_1 & -s_1 \\ s_1 & c_1 \end{bmatrix} \begin{bmatrix} z^{-1} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} c_0 & -s_0 \\ s_0 & c_0 \end{bmatrix}.$$

The matrices $U_0$, $U_1$ and $U_2$ are elementary Givens *rotation* matrices. The relation between $(h_0, h_1, h_2)$ and $(\theta_0, \theta_1, \theta_2)$ is given by

$$\begin{aligned} h_0 &= (c_0 + s_0)c_1c_2, \\ h_1 &= (c_0 - s_0)s_1c_2 + (c_0 + s_0)s_1s_2, \\ h_2 &= (c_0 - s_0)c_1s_2. \end{aligned}$$

To show that this *orthogonal realization* is numerically stable, we will rely on the theory of the *condition* of a set of linear equations [113, 111]. Consider the computation of

$$y = Ax, \tag{5.2}$$

where $A \in \mathbb{C}^{n \times n}$ is of full rank and $x \in \mathbb{C}^n$. The condition of (5.2) measures, by definition, the worst relative variation of $y$ with respect to the corresponding allowable variation of $A$ and $x$. It holds (see Appendix D) that

$$\frac{\|dy\|}{\|y\|} \leq \kappa(A) \left( \frac{\|dA\|}{\|A\|} + \frac{\|dx\|}{\|x\|} \right),$$

where

$$\kappa(A) = \|A^{-1}\| \|A\|,$$

is called the *condition number* of a set of linear equations which quantifies the sensitivity or numerical stability of the computing of (5.2). Note that $\kappa(\cdot) \geq 1$

and that it depends on the underlying norm. When the norm is to be stressed, we use a subscript. For the 2-norm, unitary matrices are perfectly conditioned in that $\kappa_2(A) = 1$ if $A$ is unitary. If we return to the realization in Figure 5.4. Put $A = U_2 Z U_1 Z U_0$. Since $Z$, $U_0$, $U_1$ and $U_2$ are unitary, we have

$$\kappa_2(A) = \kappa_2(U_2)\kappa_2(Z)\kappa_2(U_1)\kappa_2(Z)\kappa_2(U_0) = 1,$$

and we conclude that these orthogonal realizations exhibit low sensitivity to perturbations in the parameters $\theta_n$, statically as well as dynamically. The class of (minimal) orthogonal realizations also satisfy the third optimality condition, i.e. if properly implemented the realization cannot sustain any limit cycles (see also [60]). More about limit cycles can be found in [114, 115].

It is well known that Givens rotations can be implemented in CORDIC arithmetic. In CORDIC arithmetic, the matrix

$$G(\theta) = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix},$$

is a function of the single parameter $\theta$, i.e., it is a rotation operator, whereas in multiply-accumulate arithmetic this matrix leads to four independent multiplications. There are many ways to implement the rotation in CORDIC arithmetic [102, 104, 116, 117]. Most of them are defined for all $0 \leq |\theta| \leq \pi$. However, if the angle support is chosen appropriately, the rotations can be implemented at low cost. These "simple" rotations are called *micro-rotations* or *$\mu$-rotations* and were first introduced in [105, 106]. In the next subsection we briefly discuss $\mu$-rotations and study their numerical behaviour in finite word-length arithmetic.

### 5.2.3 Fast $\mu$-rotations

In order to facilitate the following discussion, we have adopted certain conventions. Let $Q$ be the quantization operator. For computations with $s$ digits after the binary point, $Q$ satisfies $Qx = x + \varepsilon$, where $|\varepsilon| \leq \eta$ and $\eta$ is the *unit round-off*. For example, in rounded arithmetic we have $\eta = 2^{-(s+1)}$ whereas $\eta = 2^{-s}$ in chopped arithmetic. If $x_1, x_2 \in \mathbb{C}^n$ satisfies $\|x_1 - x_2\|_\infty < \eta$, we cannot distinguish between them. We say that $x_1$ and $x_2$ are *equivalent modulo* $\eta$ and write $x_1 \equiv x_2 \pmod{\eta}$. In the remainder of this chapter we assume that scaling factors are introduced if necessary, so that every number $x$ lies in the standard range $|x| \leq 1$.

We are now in the position to discuss $\mu$-rotations and to study their numerical behaviour in finite word-length arithmetic. We define a fast $\mu$-rotation $F \in \mathbb{C}^{2\times 2}$ as

$$F = \begin{bmatrix} \hat{c} & e^{i\gamma}\hat{s} \\ -e^{-i\gamma}\hat{s} & \hat{c} \end{bmatrix}, \qquad \hat{c}, \hat{s}, \gamma \in \mathbb{R}, \tag{5.3}$$

where $0 \leq \hat{c}, \hat{s} \leq 1$. Since a complex $\mu$-rotation can be implemented as a composition of real $\mu$-rotations (see also [118]), in this section we have restricted ourselves to $\gamma \in \{0, \pi\}$, and thus $e^{i\gamma} \in \{-1, 1\}$.

**Lemma 5.2.1** *From the above definition and notation we have the following properties:*

1. *Let $F = U\Sigma V^*$ be the singular value decomposition of $F$. Then*

$$\sigma_1 = \sigma_2 = \sqrt{\hat{c}^2 + \hat{s}^2}.$$

2. *If $G(\theta)$ is a Givens rotation matrix, then*

$$F = \sigma_1 G(\theta),$$

*with*

$$\theta = \arctan\left(\frac{\hat{s}}{\hat{c}}\right).$$

**Proof:**

1. From (5.3) we conclude that $FF^* = \left(\hat{c}^2 + \hat{s}^2\right) I$, so that

$$\begin{aligned} \Sigma\Sigma^* &= U^*\left(FF^*\right)U \\ &= \left(\hat{c}^2 + \hat{s}^2\right) I, \end{aligned}$$

whence $\sigma_1 = \sigma_2 = \sqrt{\hat{c}^2 + \hat{s}^2}$.

2. From assertion 1, we have $F = U\Sigma V^* = \sigma_1 UV^* = \sigma_1 G(\theta)$ and, therefore, $G(\theta) = \sigma_1^{-1} F$. By inspection of (5.3), we conclude that

$$\sigma_1^{-1} F = \begin{bmatrix} c & e^{i\gamma} s \\ -e^{-i\gamma} s & c \end{bmatrix},$$

where $c = \sigma_1^{-1}\hat{c}$ and $s = \sigma_1^{-1}\hat{s}$, so that

$$\begin{aligned} \theta &= \arctan\left(\frac{s}{c}\right) \\ &= \arctan\left(\frac{\hat{s}}{\hat{c}}\right). \end{aligned}$$

□

Lemma 5.2.1 shows that a $\mu$-rotation can be regarded as a true plane rotation and an additional scaling. By choosing $\sigma_1 \approx 1$, the matrix $F$ is "close-to-orthonormal" and, therefore, numerically robust. Moreover, if the entries $\hat{c}$ and $\hat{s}$ admit a low-cost implementation, the $\mu$-rotation thus defined satisfies all the optimality conditions stated in subsection 5.2.1.

Let $k$ be a non-negative integer. Following [119, 120], we have four types of $\mu$-rotations in which the entries $\hat{c}$ and $\hat{s}$ are based on the Taylor series expansion of $\cos(2^{-k})$ and $\sin(2^{-k})$, respectively. The first and most simple $\mu$-rotation is defined by

$$\begin{aligned} \hat{c} &= 1, \\ \hat{s} &= 2^{-k}, \end{aligned}$$

so that $\sigma_1$ is given by

$$\sigma_1 = (1 + 2^{-2k})^{\frac{1}{2}}.$$

We will refer to this type of $\mu$-rotation as a type I rotation, denoted by $F_I(k)$. Let $x, y \in \mathbb{C}^2$. The computation $y = F_I(k)x$ then takes only two shift-add operations.

Obviously, $\sigma_1$ is a function of $k$ and gets smaller as $k$ gets larger. In other words, the larger the value of $k$, the more $F$ approaches a true orthonormal matrix $G$. Clearly, for some $k \geq k_I$ we have $Fx \equiv Gx \pmod{\eta}$ for any $x \in \mathbb{C}^2$ and we cannot distinguish between them. Using the property $\|A\|_\infty \leq \sqrt{m}\|A\|_2$ for any $A \in \mathbb{C}^{m \times n}$ [111, p. 57], we conclude that

$$\begin{aligned} \|(F - G)x\|_\infty &\leq \|F - G\|_\infty \|x\|_\infty \\ &\leq \|F - G\|_\infty \end{aligned}$$

$$\begin{aligned} &\leq \sqrt{2}\|F - G\|_2 \\ &= \sqrt{2}\|U(\Sigma - I)V^*\|_2 \\ &= \sqrt{2}\|\Sigma - I\|_2 \\ &= \sqrt{2}(\sigma_1 - 1). \end{aligned}$$

This means that if $\sigma_1 < 1 + \frac{1}{2}\sqrt{2}\eta$, then $Fx \equiv Gx \pmod{\eta}$. Using the Taylor series expansion

$$(1+x)^{\frac{1}{2}} = 1 + \tfrac{1}{2}x + \mathcal{O}(x^2), \quad \mathcal{O}(x^2) < 0,$$

for any non-zero $x \in \mathbb{R}$, we conclude that $\sigma_1 = 1 + 2^{-(2k+1)} + \mathcal{O}(2^{-4k}) < 1 + 2^{-(2k+1)}$. Therefore, we have $\sigma_1 < 1 + \frac{1}{2}\sqrt{2}\eta$ for any $k \geq k_I$ given by

$$k_I = -\tfrac{1}{2}\left(^2\log\eta + \tfrac{1}{2}\right).$$

A more accurate $\mu$-rotation, denoted by $F_{II}(k)$, is defined by

$$\begin{aligned} \hat{c} &= 1 - 2^{-(2k+1)}, \\ \hat{s} &= 2^{-k}, \end{aligned}$$

so that $\sigma_1$ becomes

$$\sigma_1 = (1 + 2^{-(4k+2)})^{\frac{1}{2}}.$$

Hence, the computation $y = F_{II}(k)x$ requires four shift-add operations. With this type of $\mu$-rotation, we have $\sigma_1 < 1 + \frac{1}{2}\sqrt{2}\eta$ for any $k \geq k_{II}$ given by

$$k_{II} = -\tfrac{1}{4}\left(^2\log\eta + \tfrac{5}{2}\right).$$

A type III $\mu$-rotation is defined by

$$\begin{aligned} \hat{c} &= 1 - 2^{-(2k+1)}, \\ \hat{s} &= 2^{-k} - 2^{-(3k+3)}, \end{aligned}$$

so that

$$\sigma_1 = (1 + 2^{-(6k+6)})^{\frac{1}{2}}.$$

Hence, the lower bound $k_{III}$ for this type of rotation is given by

$$k_{III} = -\tfrac{1}{6}\left(^2\log\eta + \tfrac{13}{2}\right).$$

In this case $y = F_{III}(k)x$ requires six shift-add operations.

As each of the previous types is limited to a certain lower bound of $k$, we need a $\mu$-rotation which can be scaled, i.e., a $\mu$-rotation where $\sigma_1$ can be made arbitrarily close to unity, thereby making it suitable for any value of $k$. This cannot be done easily any of the above types. The problem with scaling these $\mu$-rotations is the square root function of $\sigma_1$. Following [117], we can overcome this problem by rotating twice over approximately half the angle so that the square root function is eliminated. For this reason, a type IV $\mu$-rotation is defined to be a double type I rotation, i.e., $F_{IV}(k) = F_I(k+1)F_I(k+1)$. In doing so, we have

$$\hat{c} = 1 - 2^{-2(k+1)},$$
$$\hat{s} = 2^{-k},$$

such that $\sigma_1$ becomes

$$\sigma_1 = 1 + 2^{-2(k+1)}.$$

Note that the scaling is no longer a square root function. We can exploit this fact to obtain rapidly converging scaling sequences. These sequences are based on the well-known polynomial factorization

$$1 - x^{2^m} = (1-x)(1+x)(1+x^2)(1+x^4)\cdots(1+x^{2^{(m-1)}}).$$

Thus, let $m$ denote the number of scaling steps. We define the scaling as

$$K_m = \prod_{i=1}^{m} k_i,$$

where

$$k_i = \begin{cases} 1 - 2^{-2(k+1)} & \text{if } i = 1, \\ 1 + 2^{-2^i(k+1)} & \text{otherwise.} \end{cases}$$

With this, the singular value $\sigma_1(m)$ of a scaled type IV $\mu$-rotation becomes

$$\begin{aligned} \sigma_1(m) &= \sigma_1 K_m \\ &= \left(1 + 2^{-2(k+1)}\right) K_m \\ &= 1 - 2^{-2^{(m+1)}(k+1)}. \end{aligned}$$

As for the unscaled types, we can derive a lower bound for $k$, given the number of scaling steps, and we conclude that

$$k_{IV}(m) = -2^{-(m+1)} \left( {}^2\log\eta + 2^{(m+1)} - \tfrac{1}{2} \right).$$

| shift factor $k$ | type | angle $\theta_k$ | no. rot. | no. scl. |
|---|---|---|---|---|
| 0 | IV | $9.27295\ 10^{-1}$ | 4 | 10 |
| 1 | IV | $4.89957\ 10^{-1}$ | 4 | 8 |
| 2 | III | $2.46845\ 10^{-1}$ | 6 | 0 |
| 3 | III | $1.24596\ 10^{-1}$ | 6 | 0 |
| 4 | II | $6.24797\ 10^{-2}$ | 4 | 0 |
| 5 | II | $3.12475\ 10^{-2}$ | 4 | 0 |
| 6 | II | $1.56127\ 10^{-2}$ | 4 | 0 |
| 7 | II | $7.81246\ 10^{-3}$ | 4 | 0 |
| 8 | I | $3.90623\ 10^{-3}$ | 2 | 0 |
| 9 | I | $1.95312\ 10^{-3}$ | 2 | 0 |
| 10 | I | $9.76562\ 10^{-4}$ | 2 | 0 |
| 11 | I | $4.88281\ 10^{-4}$ | 2 | 0 |
| 12 | I | $2.44121\ 10^{-4}$ | 2 | 0 |
| 13 | I | $1.22070\ 10^{-4}$ | 2 | 0 |
| 14 | I | $6.10351\ 10^{-5}$ | 2 | 0 |
| 15 | I | $3.05176\ 10^{-5}$ | 2 | 0 |
| 16 | I | $1.52588\ 10^{-5}$ | 2 | 0 |
| 17 | I | 0 | 2 | 0 |

**Table 5.1:** *The set of feasible angles for fixed-point (chopped) arithmetic with 16-bit accuracy, the type to be used and the number of shift-add operations required.*

Note that the rotation itself requires four shift-add operations, while the scaling requires $m$ times two shift-add operations. To illustrate the previous discussion, Table 5.1 shows which angles can be used with the different types of $\mu$-rotation, as well as the requirement in terms of shift-add operations for fixed-point chopped arithmetic where $s = 16$ digits.

Table 5.1 clearly shows that the unscaled types are relatively cheap to implement while the scaled type IV rotation is in general very expensive to implement. Its use should therefore be limited. At first sight this would imply that rotations over large angles should be avoided. However, this is not so. The reason for this is that a rotation over a multiple of $\frac{\pi}{2}$ radians is an almost trivial operation, because

$$G(\alpha\frac{\pi}{2}) = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}^{\alpha},$$

so that the set of feasible angles is significantly larger than the angles given in Table 5.1. If we concentrate on the feasible angles $\theta_k$ in the first quadrant only, we can rotate over $\frac{\pi}{2} - \theta_k$ as well. This means that large angles can still be used efficiently and that we can indeed limit the use of method IV rotations. Let $\varsigma = e^{i\gamma}$ and thus $\varsigma \in \{-1, 1\}$. Clearly, for finite precision arithmetic with $s$-bit accuracy, the total set of feasible angles, denoted by $\mathcal{F}$, is given by

$$\mathcal{F} = \{\alpha\frac{\pi}{2} + \varsigma\theta_k : \alpha \in \mathbb{Z},\ k = 0, \ldots, s+1\}. \tag{5.4}$$

Figures 5.5, 5.6 and 5.7 show the realizations of $F_I(k)$, $F_{II}(k)$ and $F_{III}(k)$, respectively. The realizations of $F_{II}(k)$ and $F_{III}(k)$ are implemented as a cascade of simple stages by using the relation

$$2^{-(3k+3)} = 2^{-k}2^{-(k+1)}2^{-(k+2)}.$$

The realization of $F_{IV}(k)$ consists of a concatenation of two $F_I(k+1)$ stages, followed by, say $m$, scaling stages. Figure 5.8 shows an example of these scaling stages where $m = 3$.

We can implement the rotation and scaling stages of the different types of $\mu$-rotations in one single, generalized stage, as shown in Figure 5.9. This stage forms the basis of the fast $\mu$-rotation architecture. The 1-bit control signals $s_0$ and $s_1$ can be used to select the desired input data. By doing so, the generalized stage is fully controlled by the control sequence $\{s_0, s_1, k, \varsigma_u, \varsigma_l\}$ which requires $4 + {}^2\log s$ bits to store.

## 5.3 Unitary factorization of embedded transforms

Based on what was said in the previous section, we conclude that realizations which can be implemented using $\mu$-rotations are very suitable for VLSI technology; they are numerically robust and can be efficiently implemented. For such realizations to be feasible for our application, the realization of signal transforms, it is first necessary to identify a suitable, numerically robust algorithm to calculate $y = Ax$. This will be done in this section.

Let $A \in \mathbb{C}^{m\times m}$ be an arbitrary non-singular matrix. It is well known that for any such matrix there exists an invertible lower-triangular matrix $R \in \mathbb{C}^{m\times m}$ and a unitary matrix $Q \in \mathbb{C}^{m\times m}$ such that $A = RQ$. We will refer to this factorization as the RQ-factorization of $A$. Note that this factorization is simply the better-known QR-factorization of $A^*$ [110, 111, 112]. Let $I_j$ denote the identity in

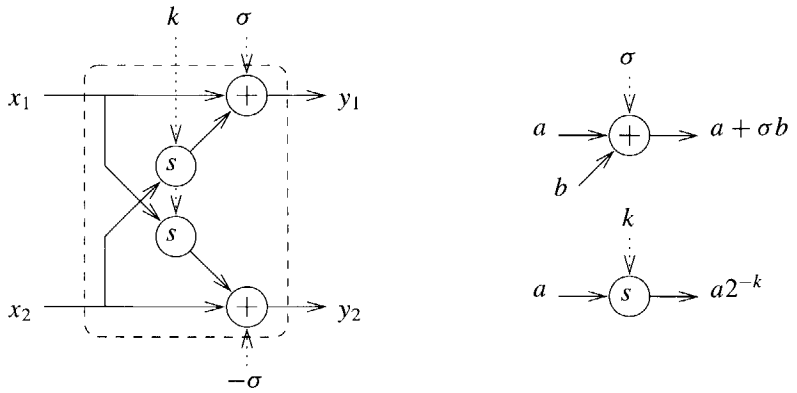



**Figure 5.5:** *Realization for $F_I(k)$.*

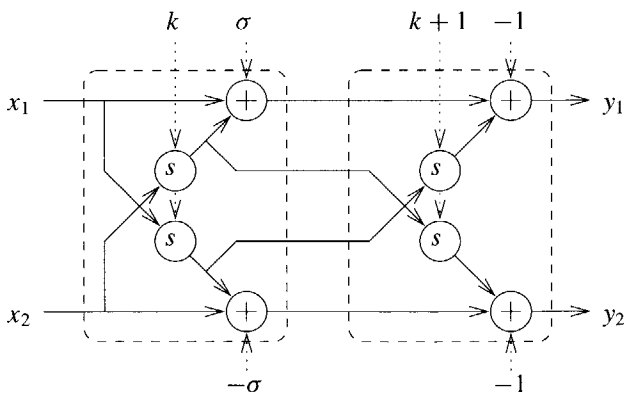



**Figure 5.6:** *Cascaded realization for $F_{II}(k)$.*

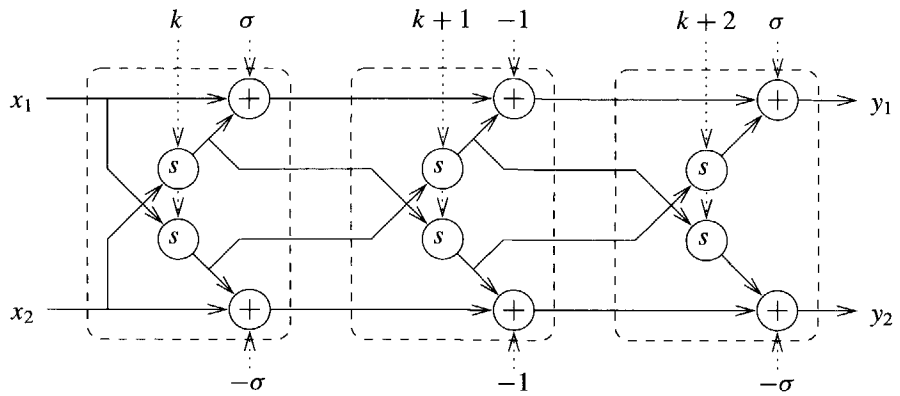



**Figure 5.7:** *Cascaded realization for* $F_{III}(k)$.

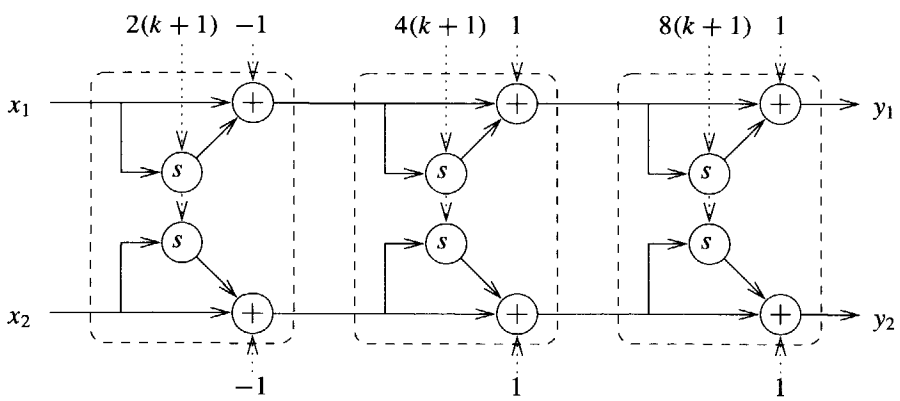



**Figure 5.8:** *Example of scaling sections* ($m = 3$) *for* $F_{IV}(k)$.

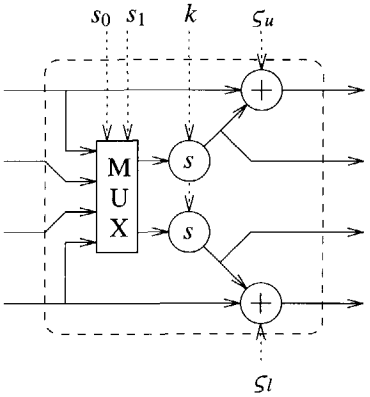



**Figure 5.9:** *Generalized stage for a fast $\mu$-rotation architecture.*

$\mathbb{C}^{j\times j}$. The mapping $Q$ thus obtained can be implemented by either a concatenation of embedded elementary Givens rotation matrices $G_m(k, l, \theta, \alpha, \beta) \in \mathbb{C}^{m\times m}$, defined by

$$G_m(k,l,\theta,\alpha,\beta) = \begin{bmatrix} I_{k-1} & & & & \\ & e^{i\alpha}\cos(\theta) & & e^{i\beta}\sin(\theta) & \\ & & I_{l-k-1} & & \\ & -e^{-i\beta}\sin(\theta) & & e^{-i\alpha}\cos(\theta) & \\ & & & & I_{m-l} \end{bmatrix}, \quad (5.5)$$

where $\alpha, \beta, \theta \in \mathbb{R}$, $0 \leq \theta \leq \pi$, or a concatenation of elementary Householder *reflection* matrices $H_m(v) \in \mathbb{C}^{m\times m}$, $v \in \mathbb{C}^m$, $v \neq o$, defined by

$$H_m(v) = I_m - 2\frac{vv^*}{v^*v}.$$

Thus, if $x \in \mathbb{C}^m$, and $x_j$ denotes the $j$th entry of $x$, then $G_m(k, l, \theta, \alpha, \beta)x$ is essentially a *plane rotation* of the vector $[x_k \;\; x_l]^t$ over $\theta$ radians and $H_m(v)x$ is a reflection of $x$ in $\ker(v^*)$. We shall not pursue the Householder factorization of $Q$.

The purpose of the rotations in the RQ-process is to nullify the super-diagonal elements of $A$, thereby reducing $A$ to a lower-triangular form. This is usually accomplished by first nulling the super-diagonal entries of the last column of $A$,

then those of the second-last column, and so forth, until a lower-triangular form is eventually reached [111]. Thus, the lower-triangular matrix $R$ is obtained by

$$AG_m^*(k_1, l_1, \theta_1, \alpha_1, \beta_1) \cdots G_m^*(k_n, l_n, \theta_n, \alpha_n, \beta_n) = R,$$

where $n = \frac{1}{2}m(m-1)$ denotes the number of super-diagonal entries of $A$. The matrix $R$ thus obtained is essentially unique, i.e., if $R_0$ is another such matrix then $R = R_0 D$ for a unitary diagonal matrix $D$ [110, p. 241].

We now consider the case where $A$ is unitary, i.e., $A$ satisfies $A^*A = AA^* = I$ so that $R$ is unitary and thus $R = I$. As a consequence, we have $A = RQ = Q = G_m(k_1, l_1, \theta_1, \alpha_1, \beta_1) \cdots G_m(k_n, l_n, \theta_n, \alpha_n, \beta_n)$ so that

$$y = Ax = G_m(k_n, l_n, \theta_n, \alpha_n, \beta_n) \cdots G_m(k_1, l_1, \theta_1, \alpha_1, \beta_1)x.$$

We can thus compute $y = Ax$ by applying a sequence of plane rotations to the input data $x$. In general, however, $A$ will not be unitary.

In this section we propose an algorithm which leads to orthogonal realizations for arbitrary mappings $A \in \mathbb{C}^{m \times n}$, $m \leq n$. The basic idea behind this algorithm is the following. Since $m \leq n$ there exists a lower-triangular matrix $R \in \mathbb{C}^{m \times n}$ and a unitary matrix $Q \in \mathbb{C}^{n \times n}$ such that

$$A = RQ = \begin{bmatrix} L & O \end{bmatrix} Q,$$

where $L \in \mathbb{C}^{m \times m}$ is lower-triangular. Therefore, $y = Ax$ can be computed as $y = Ax = RQx$. When $AA^* = |\alpha|^2 I$, $\alpha \in \mathbb{C}$, i.e., the rows of $A$ form an orthogonal set, we have $L = \alpha I$. In that case the problem is similar to the situation in which $A$ is unitary and we only have to factorize $Q$ in order to achieve an orthogonal realization. Where the rows of $A$ do not form an orthogonal set, i.e., $AA^* \neq |\alpha|^2 I$, this procedure does not work since $L$ will not be diagonal anymore and, therefore, $RQ$ cannot be factorized into unitary factors. We propose an algorithm which overcomes this problem. The idea is to embed the matrix $A$ into a larger matrix, say $X$, satisfying $XX^* = |\alpha|^2 I$, for some $\alpha \in \mathbb{C}$.

Let $A \in \mathbb{C}^{m \times n}$, $m \leq n$, and let $k$ be a non-negative integer. We construct an augmented matrix $X \in \mathbb{C}^{m \times (n+k)}$ as follows

$$X = \begin{bmatrix} P & A \end{bmatrix}, \tag{5.6}$$

where $P \in \mathbb{C}^{m \times k}$. Since $m \leq n + k$ for all $k \geq 0$, the RQ-factorization of $X$ exists and we have

$$X = RQ = \begin{bmatrix} L & O \end{bmatrix} Q, \tag{5.7}$$

where $L \in \mathbb{C}^{m\times m}$ is lower triangular. Let $x \in \mathbb{C}^n$, $v \in \mathbb{C}^m$ and $w \in \mathbb{C}^{n+k-m}$. We show that if

$$Q\begin{bmatrix} o \\ x \end{bmatrix} = \begin{bmatrix} v \\ w \end{bmatrix}, \tag{5.8}$$

then $y = Lv$. Hence, in order to compute $y = Ax$ we first compute the RQ-factorization (5.7), then compute (5.8) after which we compute $Lv$ to obtain the desired data $y = Ax$.

The purpose of $P$ is to augment the matrix $A$ to $X$ such that $L$ becomes diagonal, say $L = \alpha I$ for some $\alpha \in \mathbb{C}$. In that case we have $XX^* = RQQ^*R^* = RR^* = LL^* = |\alpha|^2 I$, which means that the rows of $X$ form an orthogonal set. Apparently, the function of $P$ is to make all the singular values of $X$ equal to $\alpha$. Since $\|X\|_2 \geq \|A\|_2 = \sigma_1$ we conclude that the largest singular value of $X$ is at least $\sigma_1$. Hence, if $\mu$ denotes the algebraic multiplicity of $\sigma_1$, $P$ must contain at least $m - \mu$ non-zero columns in order to have equal singular values. The solution which yields $\|X\|_2 = \|A\|_2 = \sigma_1$ will henceforth be referred to as the minimum norm solution for $P$. As an example, let us consider a map $A$ satisfying $AA^* = I$ (or scaled by a constant). We then have $\mu = m$ so that the minimum norm solution satisfies $\text{rank}(P) = 0$ and we do not have to extend $A$ at all. In this case the proposed algorithm automatically degenerates to the ordinary RQ-factorization of $A$, with $L = I$. Assuming that the number of operations needed to compute $Lv$ can be neglected, this algorithm leads to canonical realizations for any $A \in \mathbb{C}^{m\times n}$ if we take $P$ to be the lower-triangular minimum norm solution. Indeed, the complete RQ-factorization of $X$ then can be accomplished using $mn - \frac{1}{2}(m + 1 - k)(m - k)$ rotations, the number of non-zero super-diagonal entries of $X$. We shall return to the lower-triangular minimum norm solution.

To prove the main result of this subsection, we need the following theorem.

**Theorem 5.3.1** *Let $G \in \mathbb{C}^{n\times n}$ be Hermitian with* $\text{rank}(G) = r$. *If the leading $r \times r$ principal submatrix of $G$ is non-singular, the equation*

$$SS^* = G, \quad S \in \mathbb{C}^{n\times n} \text{ lower-triangular}, \tag{5.9}$$

*either has no solution, or all solutions are of the form $S = S_0 D$, where $S_0$ is any particular solution to (5.9) and $D \in \mathbb{C}^{n\times n}$ is a unitary diagonal matrix.*

**Proof:** If $G$ has negative eigenvalues then (5.9) has no solution. Indeed, in that case there exist $x \in \mathbb{C}^n : x^*Gx < 0$ so that $0 > x^*Gx = x^*SS^*x = \|S^*x\|^2 \geq 0$ which is a contradiction.

Let $G$ be non-negative definite. We partition $G$ as

$$G = \begin{bmatrix} G_{1,1} & G_{1,2} \\ G_{2,1} & G_{2,2} \end{bmatrix}, \tag{5.10}$$

where $G_{1,1} \in \mathbb{C}^{r\times r}$, $G_{1,2}, G_{2,1}^t \in \mathbb{C}^{r\times(n-r)}$ and $G_{2,2} \in \mathbb{C}^{(n-r)\times(n-r)}$. Since $G$ is Hermitian, we have $G = LDL^*$ where $L$ is a (not necessarily unique) unit lower-triangular matrix and $D$ a diagonal matrix. Using the same partitioning for $L$ and $D$ as in (5.10), we have

$$\begin{bmatrix} G_{1,1} & G_{1,2} \\ G_{2,1} & G_{2,2} \end{bmatrix} = \begin{bmatrix} L_{1,1} & O \\ L_{2,1} & L_{2,2} \end{bmatrix} \begin{bmatrix} D_1 & O \\ O & D_2 \end{bmatrix} \begin{bmatrix} L_{1,1}^* & L_{2,1}^* \\ O & L_{2,2}^* \end{bmatrix}. \tag{5.11}$$

Since $G_{1,1} = L_{1,1}D_1L_{1,1}^*$ is non-singular, both $L_{1,1}$ and $D_1$ are unique [111, p. 137] and $D_1$ a positive diagonal matrix of order $r$. Therefore, $G_{2,1} = L_{2,1}D_1L_{1,1}^*$ with $D_1$ and $L_{1,1}$ non-singular. We conclude that $L_{2,1} = G_{2,1}(D_1L_{1,1}^*)^{-1}$ is unique by the uniqueness of $G_{2,1}$, $L_{1,1}$ and $D_1$. Note that $L_{2,2}$ is not unique. Moreover, from (5.11) we conclude that $D_2L_{2,2}^* = 0$. Since $L_{2,2}$ is unit lower triangular, it is non-singular so that $D_2 = 0$. Thus, by setting

$$S = \begin{bmatrix} L_{1,1}\sqrt{D_1} & O \\ L_{2,1}\sqrt{D_1} & L_{2,2}\sqrt{D_2} \end{bmatrix} = \begin{bmatrix} L_{1,1}\sqrt{D_1} & O \\ L_{2,1}\sqrt{D_1} & O \end{bmatrix}, \tag{5.12}$$

we have $G = SS^*$, where $S$ is lower triangular with $n-r$ columns being zero. The proof that $S$ is essentially unique follows from $D_1 = (\sqrt{D_1}Q)(\sqrt{D_1}Q)^*$ where $Q$ is a unitary diagonal matrix. □

It is essential in Theorem 5.3.1 that the leading $r\times r$ principal submatrix is positive definite. If this requirement is not met there will be an infinite number of solutions, all differing from one another by a unitary matrix. For example, we have

$$\begin{bmatrix} 0 & 0 \\ 0 & 25 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} 0 & 3 \\ 0 & 4 \end{bmatrix},$$

as well as

$$\begin{bmatrix} 0 & 0 \\ 0 & 25 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 5 \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & 5 \end{bmatrix}.$$

However, if we consider

$$\begin{bmatrix} 25 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 5 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 5 & 0 \\ 0 & 0 \end{bmatrix},$$

this solution is essentially unique.

Now we are ready to prove our main result.

**Theorem 5.3.2** *Let $k$ be a non-negative integer, $A \in \mathbb{C}^{m\times n}$, $m \leq n$, and $P \in \mathbb{C}^{m\times k}$. Put*

$$X = \begin{bmatrix} P & A \end{bmatrix}. \tag{5.13}$$

*We can write $X = RQ$, where $Q \in \mathbb{C}^{(n+k)\times(n+k)}$ is unitary and $R \in \mathbb{C}^{m\times(n+k)}$ is lower triangular,*

$$R = \begin{bmatrix} L & O \end{bmatrix}, \tag{5.14}$$

*with $L \in \mathbb{C}^{m\times m}$ lower triangular.*

1. *If $x \in \mathbb{C}^n$, $v \in \mathbb{C}^m$ and $w \in \mathbb{C}^{n+k-m}$ such that*

$$Q\begin{bmatrix} o \\ x \end{bmatrix} = \begin{bmatrix} v \\ w \end{bmatrix},$$

*then $Ax = Lv$.*

2. *If $L = \alpha I$ for some $\alpha \in \mathbb{C}$, then $|\alpha| \geq \sigma_1$, the largest singular value of $A$, and*

$$k \geq \begin{cases} m & \text{if } |\alpha| > \sigma_1, \\ m - \mu & \text{if } |\alpha| = \sigma_1, \end{cases}$$

*where $\mu$ denotes the algebraic multiplicity of $\sigma_1$.*

3. *Let $\alpha \in \mathbb{C}$ and let*

$$k = \begin{cases} m & \text{if } |\alpha| > \sigma_1, \\ m - \mu & \text{if } |\alpha| = \sigma_1. \end{cases}$$

*Also, let $\Pi \in \mathbb{C}^{m\times m}$ be a permutation matrix. If the leading $k \times k$ principal submatrix of*

$$\Pi\left(|\alpha|^2 I - AA^*\right)\Pi^*,$$

*is non-singular, then the matrix $P \in \mathbb{C}^{m\times k}$ can be chosen such that the matrix $L$ satisfies $L = \alpha I$ and $\Pi P$ is a rank $k$ lower-triangular matrix. Moreover, $P$ is essentially unique, i.e., if $P_0$ is another such matrix, then $P_0 = PD$ for a unitary diagonal matrix $D$.*

**Proof:**

1. We have

$$\begin{aligned} Ax &= X\begin{bmatrix} o \\ x \end{bmatrix} \\ &= RQ\begin{bmatrix} o \\ x \end{bmatrix} \\ &= R\begin{bmatrix} v \\ w \end{bmatrix} \\ &= Lv. \end{aligned}$$

2. We have $|\alpha| = \|L\|_2 = \|R\|_2 = \|XQ^*\|_2 = \|X\|_2 \geq \|A\|_2 = \sigma_1$. Let $\mu$ denote the algebraic multiplicity of $\sigma_1$ and let $\Sigma$ denote the singular value matrix of $A$. Since $AA^*$ is Hermitian the SVD of $AA^*$ is given by $AA^* = U\Sigma\Sigma^*U^*$. Therefore we have

$$\begin{aligned} U^*\left(PP^*\right)U &= U^*\left(LL^* - AA^*\right)U \\ &= U^*\left(|\alpha|^2 I - AA^*\right)U \\ &= |\alpha|^2 - \Sigma\Sigma^*, \end{aligned}$$

where $\Sigma\Sigma^* = \operatorname{diag}(\sigma_1^2, \ldots, \sigma_m^2)$. Hence, $PP^*$ and $AA^*$ are diagonalized by the same unitary matrix $U$, and, since $|\alpha| \geq \sigma_1$, the quantities $\pi_1, \ldots, \pi_m, \pi_j = (|\alpha|^2 - \sigma_j^2)^{\frac{1}{2}} \geq 0$ for all $1 \leq j \leq m$ are the singular values of $P$. Hence, if $|\alpha| > \sigma_1$ we have $\pi_j > 0$ for all $j$ so that $\operatorname{rank}(P) = m$ and thus $k \geq m$, and if $|\alpha| = \sigma_1$ we have $\pi_1 = \cdots = \pi_\mu = 0$ and we conclude that $\operatorname{rank}(P) = m - \mu$ so that $k \geq m - \mu$.

3. From Theorem 5.3.1 we conclude that there exists an essentially unique lower-triangular matrix $S \in \mathbb{C}^{m\times m}$ of rank $k$ which satisfies

$$SS^* = \Pi\left(|\alpha|^2 I - AA^*\right)\Pi^*.$$

If we arrange the $k$ non-zero columns of $S$ in a matrix $P \in \mathbb{C}^{m\times k}$ such that $P = \Pi^* S$, we then have

$$\begin{aligned} |\alpha|^2 I &= \Pi^*\left(SS^*\right)\Pi + AA^* \\ &= PP^* + AA^* \\ &= LL^*. \end{aligned} \tag{5.15}$$

Obviously, $LL^*$ is Hermitian so that $L = |\alpha| D$ by Theorem 5.3.1, where $D$ is a unitary diagonal matrix. However, due to the essential uniqueness of the RQ-factorization, we can take $Q$ such that $L = \alpha I$. By construction, $\Pi P$ is a lower-triangular matrix of rank $k$ and the essentially uniqueness of $P$ follows directly from the uniqueness of $\Pi P$. □

## 5.4 Analysis and synthesis operator realization of LOTs

In this section we show how we can apply the previously developed theory to implement the analysis and synthesis maps of a transform coding system as described in Chapter 3. We discuss both analysis and synthesis operator realizations and we derive the conditions under which both operators can be given the same realization.

### 5.4.1 Analysis operator realization

As discussed in the foregoing chapters, in transform coding systems the input sequence $x$ is mapped onto an output sequence $u$ using an upper-triangular block-banded Toeplitz operator, say $T_a$, with block entries $A \in \mathbb{C}^{m \times n}$, $m \leq n$. Since the analysis operator is Toeplitz we can concentrate on a single block entry $A$. Indeed, denoting $x_{i,j} = (x(im+1), \ldots, x(im+j))^t$, the complete transformed sequence $u = T_a x$ can be calculated by computing $u_{k,m} = Ax_{k,n}$ for all $k \in \mathbb{Z}$. Hence, the analysis operator $T_a$ is fully characterized by $A$ and for this reason we will focus on computing $u_{k,m} = Ax_{k,n}$.

In the previous section we saw how to obtain optimal realizations for the execution of the matrix-vector product $u_{k,m} = Ax_{k,n}$. This result was obtained independently of the structure of the matrix $A$. However, the matrix $A$ is often so structured that it can be characterized with sometimes remarkably few angles. This effect can already be observed when $A$ exhibits some sort of symmetry as is the case with the LOT. Thus applying the factorization algorithm to the LOT matrix directly leads to realizations that are not canonical. The strategy we follow to overcome this problem is to first decompose the matrix $A$ into submatrices which fully characterize $A$ in a canonical way and to apply the factorization to these submatrices individually.

In Chapter 3, we saw that the LOT matrix $A \in \mathbb{C}^{K \times 2K}$ can be factored as follows,

$$A = \begin{bmatrix} I & \\ & B_a \end{bmatrix} \begin{bmatrix} A_{1,1} & A_{1,1}J \\ A_{1,1} & -A_{1,1}J \end{bmatrix}$$

$$= \begin{bmatrix} I & \\ & B_a \end{bmatrix} \begin{bmatrix} A_{1,1} & \\ & A_{1,1} \end{bmatrix} \begin{bmatrix} I & J \\ I & -J \end{bmatrix}, \tag{5.16}$$

where $B_a \in \mathbb{C}^{\frac{K}{2} \times \frac{K}{2}}$ is unitary, and $A_{1,1} \in \mathbb{C}^{\frac{K}{2} \times K}$ obeys $A_{1,1} A_{1,1}^* = \frac{1}{2} I$. Put $x_{k,2K} = (x_{k,K}^t \; x_{k-1,K}^t)^t$. Using the factorization (5.16), the matrix-vector product $u_{k,K} = A x_{k,2K}$ can be written as

$$u_{k,K} = \begin{bmatrix} I & \\ & B_a \end{bmatrix} \begin{bmatrix} A_{1,1} & \\ & A_{1,1} \end{bmatrix} \begin{bmatrix} I & J \\ I & -J \end{bmatrix} \begin{bmatrix} x_{k,K} \\ x_{k-1,K} \end{bmatrix}. \tag{5.17}$$

This means that for each time instant $k$, we have to compute both $A_{1,1} x_{k,K}$ and $A_{1,1} J x_{k-1,K}$. Similar to what we did in Section 3.3, we can express $A_{1,1}$ as the sum of its even-symmetric and odd-symmetric parts, i.e., $A_{1,1} = A_e + A_o$ and $A_{1,1} J = (A_e + A_o) J = A_e - A_o$, and stack the matrices $A_e$ and $A_o$ in a matrix $U \in \mathbb{C}^{K \times K}$,

$$U = \begin{bmatrix} A_e \\ A_o \end{bmatrix}. \tag{5.18}$$

We can then compute $A_{1,1} x_{k,K}$ and $A_{1,1} J x_{k-1,K}$ by computing $U x_{k,K}$ and $U x_{k-1,K}$ followed by some additions and subtractions to obtain $A_{1,1} x_{k,K}$ and $A_{1,1} J x_{k-1,K}$. Moreover, temporary storage of $U x_{k,K}$ at time $k$ allows us to reuse this product at time $k+1$, as we then have to compute $U x_{k+1,K}$ and $U x_{k,K}$. Since $A_{1,1} A_{1,1}^* = \frac{1}{2} I$ and thus $A_e A_e^* = A_o A_o^* = \frac{1}{4} I$, we have

$$U U^* = U^* U = \frac{1}{4} I. \tag{5.19}$$

This property enables us to reduce further the computational complexity of calculating $u_{k,K} = A x_{k,2K}$. Since $U$ itself contains $\frac{K}{2}$ even-symmetric and $\frac{K}{2}$ odd-symmetric rows, we can partition $U$ similar to $A$,

$$U = \begin{bmatrix} U_{1,1} & U_{1,1} J \\ U_{2,1} & -U_{2,1} J \end{bmatrix},$$

where $U_{1,1}, U_{2,1} \in \mathbb{C}^{\frac{K}{2} \times \frac{K}{2}}$. Moreover, from (5.19), we have

$$U_{1,1} U_{1,1}^* = U_{1,1}^* U_{1,1} = \tfrac{1}{8} I,$$

$$U_{2,1} U_{2,1}^* = U_{2,1}^* U_{2,1} = \tfrac{1}{8} I.$$

There is therefore a unique unitary matrix, say $B_u$, such that $U_{2,1} = B_u U_{1,1}$, and we can write

$$U = \begin{bmatrix} I & \\ & B_u \end{bmatrix} \begin{bmatrix} U_{1,1} & \\ & U_{1,1} \end{bmatrix} \begin{bmatrix} I & J \\ I & -J \end{bmatrix}. \tag{5.20}$$

In general, the matrix $U_{1,1}$ will be unstructured so that where the LOT is fully canonically characterized by the submatrices $B_a$, $B_u$ and $U_{1,1}$. When $U_{1,1}$ itself consists of $\frac{K}{4}$ even-symmetric and $\frac{K}{4}$ odd-symmetric rows, however, a further factorization for $U_{1,1}$ can be carried out. This situation arises, for example, when $U$ is a discrete cosine transform (DCT). In that case the matrix $U_{1,1}$ is a DCT half the size of $U$ and, therefore, can be further factorized. In fact, the factorization (5.20) can then be continued recursively until we end up with trivial scalar multiplications.

It is important to note that a necessary and sufficient condition for complete recursive factorization of $A$ is that the matrix $U_{1,1}$ can be factorized recursively. This means that LOTs with even-symmetric rows of $U$ as the even-symmetric DCT functions satisfy this condition. This is the main reason why Algorithm 3.1 uses the DCT as its initial solution and only affects $A_o$, the odd-symmetric functions, during the recursive procedure. It should be noted that many Fourier-like signal transforms have this recursion property, i.e., a size $n$ linear-phase transform can be constructed with two size $\frac{n}{2}$ linear-phase transforms. For fast decompositions of Fourier-like transforms the reader is referred to [121, 122, 123, 124].

In conclusion, we showed that in order to compute $u_{k,K} = Ax_{k,2K}$ we do not apply the iterative RQ-factorization directly to the matrix $A$. Instead, we first factorize the matrix as in (5.16) and in (5.20), and we then apply the iterative algorithm to the matrices $B_a$, $B_u$ and $U_{1,1}$.

### 5.4.2 Synthesis operator realization

In this subsection we focus on the realization of the synthesis mapping $T_s : u \mapsto y$. In Chapter 3 we saw that the synthesis mapping $T_s$, too, has a triangular block-banded Toeplitz structure as follows

$$T_s = \begin{bmatrix} \ddots & & & & \\ \ddots & S_1^t & & & \\ & S_2^t & S_1^t & & \\ & & S_2^t & S_1^t & \\ & & & \ddots & \ddots \end{bmatrix}.$$

In order to exploit the symmetry properties of the synthesis operator, which are in the columns rather than in the rows, we characterize $T_s$ in terms of the matrix $S = [S_1 \; S_2]$ containing the synthesis filters. We can then calculate $y = T_s u$ by repeatedly calculating $y_{k,2K} = S^t u_{k,K}$ for all $k \in \mathbb{Z}$, and adding together the results thus obtained with a mutual overlap of $K$ samples. For this reason, in this subsection we will focus on computing $y_{k,2K} = S^t u_{k,K}$.

It is advantageous to design an optimal realization which can be used for both the analysis and synthesis mappings. Such a solution, if it exists, has two major advantages over *any* other realization. Firstly, in applications where the analysis and synthesis mappings are not computed concurrently, this solution would save 50% of the total chip area. Secondly, and even more importantly, $T_s T_a = I$ is maintained even under perturbation of the angles characterizing both $A$ and $S$, assuming the rotations are properly implemented. We shall come back to this point and will now proceed with the determination of a unique realization for both $A$ and $S$. We have the following result.

**Theorem 5.4.1** *Using the definition and notation from above, let $T_s T_a = I$. For any $P \in \mathbb{C}^{m \times k}$, let*

$$X = \begin{bmatrix} P & A \end{bmatrix},$$

*have RQ-factorization $X = RQ$, where*

$$R = \begin{bmatrix} L & O \end{bmatrix},$$

*and $L = \alpha I \in \mathbb{C}^{m \times m}$ for some $\alpha \in \mathbb{C}$. If $x \in \mathbb{C}^m$, $v \in \mathbb{C}^k$ and $w \in \mathbb{C}^n$ such that*

$$Q^* \begin{bmatrix} x \\ o \end{bmatrix} = \begin{bmatrix} v \\ w \end{bmatrix}, \tag{5.21}$$

*then $w = S^t x$ if and only if $T_a T_a^* = \alpha I$.*

**Proof:** First assume that $T_a T_a^* = \alpha I$. Since $T_s T_a = I$ we conclude that $T_a^* = \alpha T_s$ and thus $S^t = \alpha^{-1} A^*$. Partitioning $Q$ as

$$Q = \begin{bmatrix} Q_{1,1} & Q_{1,2} \\ Q_{2,1} & Q_{2,2} \end{bmatrix},$$

where $Q_{1,1} \in \mathbb{C}^{k \times k}$, $Q_{1,2} \in \mathbb{C}^{k \times n}$, $Q_{2,1} \in \mathbb{C}^{n \times k}$ and $Q_{2,2} \in \mathbb{C}^{n \times n}$, we conclude that $A = \alpha Q_{1,2}$ and thus, from (5.21), $w = Q_{1,2}^* x = \alpha^{-1} A^* x = S^t x$.
On the contrary, if we assume that $w = S^t x$, again, we have $w = Q_{1,2}^* x$ so that

$w = \alpha^{-1}A^*x$, whence $S^t = \alpha^{-1}A^*$ since both $A$ and $S$ are of full rank. Therefore, $T_s = \alpha^{-1}T_a^*$ and, since $T_sT_a = I$, $T_aT_a^* = \alpha I$. □

Theorem 5.4.1 shows that when $A$ is isometric, i.e., $A$ obeys $AA^* = \alpha I$ for some $\alpha \in \mathbb{R}$, both the analysis operator and synthesis operator can be given the same realization. Note that this property also holds for non-square signal transforms. To show that $T_sT_a = I$ is maintained when the angles characterizing both $A$ and $S$ are perturbed, remember that we factor $A$ as in (5.16) and in (5.20), and apply the RQ-factorization to the matrices $B_a$, $B_u$ and $U_{1,1}$, which are all unitary. Hence, if no overflows are introduced in the additions and the subtractions and the realizations of $B_a$, $B_u$ and $U_{1,1}$ are implemented in CORDIC arithmetic, the inverse matrices will be characterized by the same angles. Hence, $B_a$, $B_u$ and $U_{1,1}$ can be perfectly inverted, no matter what the values of the angles are.

## 5.5 Calculating the RQ-factorization

In the previous sections we identified a numerically robust algorithm to calculate $y = Ax$, which is based on the RQ-factorization of a matrix $X$ satisfying $XX^* = |\alpha|^2 I$ for some $\alpha \in \mathbb{C}$. As mentioned before, this factorization can be computed using a concatenation of Givens rotation matrices by first nulling the super-diagonal entries of the last column of $X$, then those of the second-last column, and so forth until a lower-triangular form is eventually reached. If we use $\mu$-rotations instead of exact Givens rotations, however, the ordering in which the super-diagonal entries of $X$ are zeroed may change. Indeed, since we added to the range $0 \leq \theta \leq \pi$ of angles a discrete set $\mathcal{F}$ of feasible angles which admit simple and fast rotations, entries of $X$ can only be made approximately zero and accuracy will depend on the value of the entry. In this section we propose an iterative algorithm for calculating the RQ-factorization of a matrix $X \in \mathbb{C}^{m \times l}$, $m \leq l$, satisfying $XX^* = |\alpha|^2 I$ for some $\alpha \in \mathbb{C}$. The idea behind the iterative RQ-algorithm is that we apply a sequence of rotations to $X$ with the property that each new $X$ is "more diagonal" than its predecessor. Eventually, the off-diagonal entries of $X$ are small enough to be declared zero. In subsection 5.5.1 we discuss this algorithm in more detail. We prove the convergence in subsection 5.5.2.

### 5.5.1 Iterative RQ-factorization

Let $X \in \mathbb{C}^{m \times l}$, $m \leq l$ satisfying $XX^* = |\alpha|^2 I$ for some $\alpha \in \mathbb{C}$. Since $XX^*$ is diagonal, we can compute the RQ-factorization of $X$ by systematically reducing

the "off-diagonal" energy, defined by

$$\text{off}(X) = \sum_{i=1}^{m} \sum_{\substack{j=1 \\ j \neq i}}^{l} |x_{ij}|^2.$$

The tools for doing this are embedded $\mu$-rotations $F(p, q, \theta_k, \gamma) \in \mathbb{C}^{l \times l}$, $1 \leq p \leq m$, $p < q \leq l$, of the form

$$F(p, q, \theta_k, \gamma) = \begin{bmatrix} I_{p-1} & & & & \\ & \hat{c} & & e^{i\gamma}\hat{s} & \\ & & I_{q-p-1} & & \\ & -e^{i\gamma}\hat{s} & & \hat{c} & \\ & & & & I_{l-q} \end{bmatrix},$$

where $\gamma \in \mathbb{R}$ and $\theta_k \in \mathcal{F}$, a set of feasible angles. If $X^{(i)} = (x_{k,l}^{(i)})_{k,l}$ denotes the updated $X$ matrix after $i$ iteration steps, then the updated $X$ matrix after $i+1$ iterations will be $X^{(i+1)} = X^{(i)}F(p, q, \theta_k, \gamma)$. The basic step in the iterative RQ procedure involves choosing an index pair $(p, q)$ and a pair $(\theta_k, \gamma)$ such that the off-diagonal energy of $X$ is minimized. Hence, a precise statement of this procedure is

$$\begin{array}{ll} \text{minimize} & \text{off}(X), \\ \text{subject to} & \theta_k \in \mathcal{F}. \end{array}$$

As the iteration $X^{(i+1)} = X^{(i)}F(p, q, \theta_k, \gamma)$ only affects the columns $p$ and $q$ of $X^{(i)}$ and since a decrease in the off-diagonal energy of $X^{(i)}$ can only be accomplished by rotating energy from the entries $x_{pq}$ and $x_{qp}$ (assuming that $q \leq m$) to the diagonal elements $x_{pp}$ and $x_{qq}$, respectively, it is sufficient to concentrate on computing

$$\begin{bmatrix} x_{pp}^{(i+1)} & x_{pq}^{(i+1)} \\ x_{qp}^{(i+1)} & x_{qq}^{(i+1)} \end{bmatrix} = \begin{bmatrix} x_{pp}^{(i)} & x_{pq}^{(i)} \\ x_{qp}^{(i)} & x_{qq}^{(i)} \end{bmatrix} \begin{bmatrix} \hat{c} & e^{\gamma}\hat{s} \\ -e^{\gamma}\hat{s} & \hat{c} \end{bmatrix}. \tag{5.22}$$

The case $m < q \leq l$ can be simply tackled by extending $X$ with $l - m$ zero rows to a square $l \times l$ matrix and making the extended matrix diagonal. This means that if $m < q \leq l$ we have $x_{qp} = x_{qq} = 0$ so that (5.22) reduces to a true RQ step. For this reason we will henceforth assume, without loss of generality, that $X$ is square of size $l \times l$.

Since the Frobenius norm is rotation invariant, we conclude that

$$\begin{aligned} \operatorname{off}(X^{(i+1)}) &= \|X^{(i+1)}\|_F^2 - \sum_{j=1}^{l} |x_{jj}^{(i+1)}|^2 \\ &= \|X^{(i)}\|_F^2 - \sum_{\substack{j=1 \\ j\neq p,q}}^{l} |x_{jj}^{(i)}|^2 - \left(|x_{pp}^{(i+1)}|^2 + |x_{qq}^{(i+1)}|^2\right). \end{aligned} \tag{5.23}$$

Let $\Delta_{p,q}^{(i)}$ denote the increase in diagonal energy caused by a rotation in the $(p, q)$ plane at iteration $i$, i.e.,

$$\Delta_{p,q}^{(i)} = |x_{pp}^{(i+1)}|^2 + |x_{qq}^{(i+1)}|^2 - \left(|x_{pp}^{(i)}|^2 + |x_{qq}^{(i)}|^2\right). \tag{5.24}$$

From this, (5.23) can be written as

$$\begin{aligned} \operatorname{off}(X^{(i+1)}) &= \|X^{(i)}\|_F^2 - \sum_{j=1}^{l} |x_{jj}^{(i)}|^2 - \Delta_{p,q}^{(i)} \\ &= \operatorname{off}(X^{(i)}) - \Delta_{p,q}^{(i)}. \end{aligned}$$

Hence, in order to minimize the off-diagonal energy of $X$ at iteration $i$, we have to maximize $\Delta_{p,q}^{(i)}$.

Let $\operatorname{Re}(x)$ and $\operatorname{Im}(x)$ denote the real and imaginary part of $x \in \mathbb{C}$, respectively. From (5.22) and (5.24) we find that

$$\begin{aligned} \Delta_{p,q}^{(i)} &= \hat{s}^2 \left(|x_{pq}^{(i)}|^2 + |x_{qp}^{(i)}|^2 - |x_{pp}^{(i)}|^2 - |x_{qq}^{(i)}|^2\right) + \\ &\quad + 2\hat{c}\hat{s}\operatorname{Re}\left(e^{i\gamma}(x_{qp}^{(i)}\bar{x}_{qq}^{(i)} - x_{pp}^{(i)}\bar{x}_{pq}^{(i)})\right). \end{aligned} \tag{5.25}$$

By using some elementary calculus and setting the derivative with respect to $\gamma$ of the right-hand side of (5.25) to zero, we have

$$\gamma = -\arctan\left(\frac{\operatorname{Im}(x_{qp}^{(i)}\bar{x}_{qq}^{(i)} - x_{pp}^{(i)}\bar{x}_{pq}^{(i)})}{\operatorname{Re}(x_{qp}^{(i)}\bar{x}_{qq}^{(i)} - x_{pp}^{(i)}\bar{x}_{pq}^{(i)})}\right) \bmod \pi, \tag{5.26}$$

and similarly for $\theta$, we obtain

$$\theta = \frac{1}{2}\arctan\left(\frac{2\operatorname{Re}\left(e^{i\gamma}(x_{qp}^{(i)}\bar{x}_{qq}^{(i)} - x_{pp}^{(i)}\bar{x}_{pq}^{(i)})\right)}{|x_{pp}^{(i)}|^2 + |x_{qq}^{(i)}|^2 - |x_{pq}^{(i)}|^2 - |x_{qp}^{(i)}|^2}\right) \bmod \frac{\pi}{2}. \tag{5.27}$$

$i = 1;$
**while** off$(X^{(i)}) >$ threshold,
**do**
$(p, q, \theta, \gamma) = compute\_opt\_angle(X^{(i)});$
select $\theta_k, \theta_{k+1} \in \mathcal{F} : \theta_{k+1} \leq \theta < \theta_k;$
**if** $\Delta_{p,q}^{(i)}(\theta_{k+1}) \geq \Delta_{p,q}^{(i)}(\theta_k),$
**then**
$X^{(i+1)} = X^{(i)} F(p, q, \theta_{k+1}, \gamma);$
**else**
$X^{(i+1)} = X^{(i)} F(p, q, \theta_k, \gamma);$
**fi;**
$i = i + 1;$
**od;**

**Algorithm 5.1:** *Iterative RQ-factorization.*

Clearly, $\gamma$ and $\theta$ maximize the diagonal energy if the second derivatives of the right-hand side of (5.25) are negative for those particular values of $\gamma$ and $\theta$. It is easily verified that if we take $\gamma$ such that the quantity $e^{i\gamma}(x_{qp}^{(i)}\bar{x}_{qq}^{(i)} - x_{pp}^{(i)}\bar{x}_{pq}^{(i)})$ is non-negative, then the diagonal energy is maximized by taking $0 < \theta < \frac{\pi}{4}$.

Algorithm 5.1 gives the outline of the iterative RQ-factorization. The procedure *compute_opt_angle* $(\cdot)$ computes the optimum pair $(\theta, \gamma)$ according to (5.26) and (5.27) for all pairs $(p, q)$ for which $1 \leq p < q \leq l$ and returns the quadruple $(p, q, \theta, \gamma)$ which maximizes (5.25).

### 5.5.2 Convergence of the iterative RQ-factorization

In this subsection we consider the convergence of the iterative RQ-algorithm. In order to prove the convergence of this "greedy" RQ-algorithm, it is sufficient to show that

$$\exists (p, q) : 1 \leq p < q \leq l, \ \Delta_{p,q}^{(i)} > 0, \tag{5.28}$$

for each iteration step $i = 1, \ldots, i_{\max}$, where $i_{\max}$ denotes the last iteration step in the sense that off$(X^{(i)})$ is small enough to be declared zero for all $i \geq i_{\max}$. By small enough to be declared zero we mean that $|x_{pq}^{(i_{\max})}| < \eta$, the unit round-off, for all $p, q = 1, \ldots, l, \ p \neq q$. For finite precision arithmetic with $s$-bit accuracy,

the smallest non-zero fast rotation angle, say $\theta_{\min}$, is given by $\theta_{\min} = \arctan(2^{-s})$. Clearly, assuming that the algorithm converges, we have $|x_{pq}^{(i_{\max})}| \leq \sin(\frac{1}{2}\theta_{\min})$ for all $p, q : p \neq q$ so that

$$\begin{aligned} |x_{pq}^{(i_{\max})}| &\leq \sin(\tfrac{1}{2}\theta_{\min}) \\ &< \tfrac{1}{2}\theta_{\min} \\ &= \tfrac{1}{2}\arctan(2^{-s}) \\ &< \tfrac{1}{2}2^{-s} \\ &\leq \eta, \end{aligned}$$

and we conclude that the off-diagonal entries of $X$ are guaranteed to be smaller than the unit round-off $\eta$ and can, therefore, be declared zero. Note that this property is independent of the use of rounded or chopped arithmetic.

To see whether we can satisfy (5.28) for all $i = 1, \ldots, i_{\max}$, let $\theta_{\text{opt}} \bmod \pi$ be the solution of (5.27) which maximizes $\Delta_{p,q}^{(i)}$. If we have $\theta_{\text{opt}} \neq 0 \bmod \pi$, then $\Delta_{p,q}^{(i)} > 0$. If we have $\theta_{\text{opt}} = 0 \bmod \pi$, however, we conclude that $\Delta_{p,q}^{(i)} = 0$, which has the interpretation that no off-diagonal energy can be rotated to the main diagonal of $X$ at iteration $i$ by a rotations in the $(p, q)$ plane. Obviously, if this situation occurs for all $1 \leq p < q \leq m$, we cannot satisfy (5.28). By inspection of the second derivative with respect to $\theta$ of (5.25), we conclude that

$$\left.\frac{\partial^2 \Delta_{p,q}^{(i)}}{\partial \theta^2}\right|_{\theta = 0 \bmod \pi} = -2\left(|x_{pp}^{(i)}|^2 + |x_{qq}^{(i)}|^2 - |x_{pq}^{(i)}|^2 - |x_{qp}^{(i)}|^2\right),$$

so that $\theta_{\text{opt}} = 0 \bmod \pi$ implies that

$$\begin{aligned} &x_{pp}^{(i)}\bar{x}_{pq}^{(i)} = x_{qp}^{(i)}\bar{x}_{qq}^{(i)}, \\ &|x_{pp}^{(i)}|^2 + |x_{qq}^{(i)}|^2 \geq |x_{pq}^{(i)}|^2 + |x_{qp}^{(i)}|^2. \end{aligned} \tag{5.29}$$

Hence, the convergence breaks down if both conditions are satisfied simultaneously for all $1 \leq p < q \leq l$ for some $i < i_{\max}$. This situation, however, is very unlikely to occur, as we shall show later on. Where this situation does occur, we can overcome the convergence problem by using a double-sided rotation rather than a single-sided rotation. Therefore, the updated $X$ matrix after iteration $i$ will be $X^{(i)} = F^*(p, q, \theta_k, \gamma)X^{(i-1)}F(p, q, \theta_l, \gamma')$. In fact, this is the basic step in the Kogbetliantz algorithm [119, 125, 126]. If $X^{(i)}$ is Hermitian, this step reduces to a basic step of the Jacobi method [110, 111, 119, 127, 128]. It should be noted

that the algorithm proposed in Section 5.2.2 for computing $y = Ax$ does not break down with the use of double-sided rotations.

In the remainder of this section we concentrate on matrices for which (5.29) is satisfied for all $1 \leq p < q \leq m$. To determine the algebraic structure of such matrices, first assume that $m < q \leq l$. As mentioned above, in that case we have $x_{qp}^{(i)} = x_{qq}^{(i)} = 0$ for all $i = 1, \ldots, i_{\max}$, so that from (5.29) we conclude that $\theta_{\text{opt}} \neq 0 \bmod \pi$ (unless $x_{pq} = 0$ and nothing has to be done), and thus $\Delta_{p,q}^{(i)} > 0$ for all $i$. This means that we can always nullify the last $l - m$ columns of $X$ so that $X$ becomes zero, except for the $m \times m$ leading principal submatrix, say $X_m$. We then have $X_m X_m^* = XX^* = |\alpha|^2 I$. To facilitate the following discussion we assume, without loss of generality, that $|\alpha| = 1$, so that we can concentrate on the algebraic structure of $m \times m$ unitary matrices satisfying (5.29) for all $1 \leq p < q \leq m$.

Let $\mathcal{U}_m$ denote the set of $m \times m$ unitary matrices which do not converge to the desired diagonal form. Thus, if $A \in \mathcal{U}_m$, then

$$\begin{aligned} &a_{pp}\bar{a}_{pq} = a_{qp}\bar{a}_{qq}, \\ &|a_{pp}|^2 + |a_{qq}|^2 \geq |a_{qp}|^2 + |a_{pq}|^2, \end{aligned} \tag{5.30}$$

for all $1 \leq p < q \leq m$. We have the following results.

**Theorem 5.5.1** *Let $A \in \mathcal{U}_m$. Let $D \in \mathbb{C}^{m \times m}$ be a unitary diagonal matrix and $\Pi \in \mathbb{C}^{m \times m}$ a permutation matrix. Then*

$$\begin{aligned} &\Pi(DA)\Pi^* \in \mathcal{U}_m, \\ &\Pi(AD)\Pi^* \in \mathcal{U}_m. \end{aligned}$$

**Proof:** We will prove that $DA \in \mathcal{U}_m$ and $AD \in \mathcal{U}_m$. To do so, let $D = \text{diag}(\lambda_1, \ldots, \lambda_m)$, $\lambda_j \in \mathbb{C}$ and $|\lambda_j| = 1$ for all $j$. Since $A \in \mathcal{U}_m$, we have for $DA$,

$$\begin{aligned} (\lambda_p a_{pp})(\overline{\lambda_q a_{pq}}) &= \lambda_p \bar{\lambda}_q a_{pp}\bar{a}_{pq} \\ &= \lambda_p \bar{\lambda}_q a_{qp}\bar{a}_{qq} \\ &= (\lambda_p a_{qp})(\overline{\lambda_q a_{qq}}), \end{aligned}$$

and similarly for $AD$,

$$\begin{aligned} (a_{pp}\lambda_p)(\overline{a_{pq}\lambda_p}) &= |\lambda_p|^2 a_{pp}\bar{a}_{pq} \\ &= |\lambda_q|^2 a_{qp}\bar{a}_{qq} \\ &= (a_{qp}\lambda_q)(\overline{a_{qq}\lambda_q}), \end{aligned}$$

for all $1 \leq p < q \leq m$. Moreover, since

$$\begin{aligned} |\lambda_p a_{pp}|^2 + |\lambda_q a_{qq}|^2 &= |a_{pp}|^2 + |a_{qq}|^2 \\ &\geq |a_{qp}|^2 + |a_{pq}|^2 \\ &= |\lambda_p a_{qp}|^2 + |\lambda_q a_{pq}|^2, \end{aligned}$$

for all $1 \leq p < q \leq m$, we conclude that $DA \in \mathcal{U}_m$ and $AD \in \mathcal{U}_m$. The remaining part of the proof is trivial. □

From Theorem 5.5.1 we conclude that we can concentrate on matrices $A \in \mathcal{U}_m$ with real non-negative diagonal entries satisfying $a_{11} \geq a_{22} \geq \cdots \geq a_{mm}$.

**Theorem 5.5.2** *Let $A \in \mathcal{U}_m$ have real non-negative diagonal entries such that $a_{11} \geq a_{22} \geq \cdots \geq a_{mm}$. Moreover, let $m_j, \mu_j$, $j = 1, \ldots, k \leq m$ be real non-negative numbers such that* $\operatorname{diag}(a_{11}, \ldots, a_{mm}) = \operatorname{diag}(m_1 I_1, \ldots, m_k I_k)$ *where $I_j \in \mathbb{C}^{\mu_j \times \mu_j}$. Then*

$$A = \operatorname{diag}(A_1, \ldots, A_k), \quad A_j \in \mathbb{C}^{\mu_j \times \mu_j}, \tag{5.31}$$

*satisfying*

$$a_{qp} = \bar{a}_{pq}. \tag{5.32}$$

**Proof:** The proof is by induction to $k$. For $k = 1$ we have $a_{pp} = m_1$ for all $1 \leq p \leq m$ so that $a_{pp}\bar{a}_{pq} = a_{qp}\bar{a}_{qq}$ implies $a_{qp} = \bar{a}_{pq}$. Let $k > 1$. We then have

$$\begin{aligned} |a_{pp}|^2 &= |a_{pp}|^2 \sum_{q=1}^{m} |a_{pq}|^2 \\ &= \sum_{q=1}^{m} |a_{pp}\bar{a}_{pq}|^2 \\ &= \sum_{q=1}^{m} |a_{qp}\bar{a}_{qq}|^2 \\ &\leq m_1^2 \sum_{q=1}^{m} |a_{qp}|^2 \\ &= m_1^2. \end{aligned}$$

If $a_{pp} = m_1$ we have equality and we conclude that

$$a_{qp} = \begin{cases} \bar{a}_{pq} & \text{if } a_{qq} = m_1 \\ a_{pq} = 0 & \text{if } a_{qq} \neq m_1 \end{cases}.$$

Hence, $A = \text{diag}(A_1, B)$ where $B \in \mathbb{C}^{(m-\mu_1)\times(m-\mu_1)}$. By the induction hypothesis, $B$ satisfies (5.31) and (5.32) so that $A = \text{diag}(A_1, \ldots, A_k)$, $A_j \in \mathbb{C}^{\mu_j \times \mu_j}$ for all $j = 1, \ldots, k$, satisfying $a_{pq} = \bar{a}_{qp}$. □

From Theorems 5.5.1 and 5.5.2, we conclude that unitary matrices which do not converge to a diagonal form are (except for a symmetric permutation) are block-diagonal matrices where the block-diagonal elements are essentially unitary Hermitian with equal main-diagonal entries satisfying (5.30). For example,

$$A = \frac{1}{5}\begin{bmatrix} -3 & 2 & 2 & 2 & 2 \\ 2 & -3 & 2 & 2 & 2 \\ 2 & 2 & -3 & 2 & 2 \\ 2 & 2 & 2 & -3 & 2 \\ 2 & 2 & 2 & 2 & -3 \end{bmatrix},$$

does not converge to a diagonal form by single-sided rotations and the same holds for any other matrix $DA$ or $AD$ where $D$ is a unitary diagonal matrix. Note that

$$A = \frac{1}{3}\begin{bmatrix} -1 & 2 & 2 \\ 2 & -1 & 2 \\ 2 & 2 & -1 \end{bmatrix},$$

does satisfy $a_{pp}\bar{a}_{pq} = a_{qp}\bar{a}_{qq}$ but not $|a_{pp}|^2 + |a_{qq}|^2 \geq |a_{qp}|^2 + |a_{pq}|^2$ for all $1 \leq p < q \leq m$, so this example does not cause convergence problems. Indeed, a rotation over $\frac{\pi}{2}$ radians in any $(p, q)$ plane will decrease the off-diagonal energy of this matrix.

It should be noted that matrices which cause convergence problems is very unlikely to occur since we have to satisfy $2\binom{m}{2} = m(m-1)$ constraints simultaneously. Moreover, for $m \leq 3$ we cannot construct a matrix $A \in \mathcal{U}_m$. This property will be shown in Theorem 5.5.3 for which we need the following results.

**Lemma 5.5.1** *If $A \in \mathcal{U}_m$, then*

$$|a_{pp}| \geq |a_{qp}|,$$

*for all $1 \leq p < q \leq m$.*

**Proof:** From (5.30) and Theorems 5.5.1 and 5.5.2 we conclude that

$$\begin{aligned} |a_{pp}|^2 + |a_{qq}|^2 &\geq |a_{qp}|^2 + |a_{pq}|^2 \\ &= \begin{cases} 2|a_{qp}|^2 & \text{if } |a_{pp}| = |a_{qq}| \\ 0 & \text{otherwise} \end{cases}, \end{aligned}$$

so that $|a_{pp}| \geq |a_{qp}|$ for all $1 \leq p < q \leq m$, as required. □

**Corollary 5.5.1** *If $A \in \mathcal{U}_m$, then $|a_{pp}| > 0$ for all $1 \leq p \leq m$.*

**Proof:** Follows trivially from Lemma 5.5.1 and the fact that $A$ is of full rank $m$. □

**Theorem 5.5.3** *If $A \in \mathcal{U}_m$, then $m \geq 4$.*

**Proof:** For all $1 \leq p < q \leq m$ we have

$$\begin{aligned} 0 &= \sum_{j=1}^{m} a_{pj}\bar{a}_{qj} \\ &= a_{pp}\bar{a}_{pq} + a_{qp}\bar{a}_{qq} + \sum_{\substack{j=1 \\ j\neq p,q}}^{m} a_{pj}\bar{a}_{qj} \\ &= 2a_{pp}\bar{a}_{pq} + \sum_{\substack{j=1 \\ j\neq p,q}}^{m} a_{pj}\bar{a}_{qj}. \end{aligned}$$

Let $\mu$ denote the maximum absolute off-diagonal entry of $A$. We then conclude, using Lemma 5.5.1, that

$$\begin{aligned} 2|a_{pp}\bar{a}_{pq}| &= \left| \sum_{\substack{j=1 \\ j\neq p,q}}^{m} a_{pj}\bar{a}_{qj} \right| \\ &\leq \sum_{\substack{j=1 \\ j\neq p,q}}^{m} |a_{pj}||\bar{a}_{qj}| \\ &\leq \sum_{\substack{j=1 \\ j\neq p,q}}^{m} |a_{pp}||\bar{a}_{qj}| \\ &\leq (m-2)\mu|a_{pp}|. \end{aligned}$$

Consequently, as $|a_{pp}| > 0$ by Corollary 5.5.1, we have that $2|\bar{a}_{pq}| \leq (m-2)\mu$. Now take $p$ and $q$ such that $|\bar{a}_{pq}| = \mu$ and we conclude that $m \geq 4$. □

As an example of $A \in \mathcal{U}_m$ for $m = 4$ we have

$$A = \frac{1}{4}\begin{bmatrix} -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 \\ 1 & 1 & 1 & -1 \end{bmatrix}. \tag{5.33}$$

Note from the proof of Theorem 5.5.3 that for $m = 4$, the relations are satisfied with equality so that $|a_{qp}| = \mu$ for all $p, q = 1, \ldots m$. Hence, (5.33) is essentially unique.

## 5.6 Pipelined architecture for LOTs

In the previous section we discussed an iterative algorithm for computing the RQ-factorization of a matrix $X$ satisfying $XX^* = |\alpha|^2 I$ for some $\alpha \in \mathbb{C}$. In this section we will discuss the architecture imposed by this algorithm, in which the realization of LOTs can be mapped. To facilitate the following discussion, we will restrict ourselves to real valued matrices only.

### 5.6.1 Architecture for computing $y = Ax$

Let $k$ be a non-negative integer. Remember that we constructed an augmented matrix $X \in \mathbb{R}^{m \times (n+k)}$ as follows

$$X = \begin{bmatrix} P & A \end{bmatrix}, \tag{5.34}$$

where $P \in \mathbb{R}^{m \times k}$ is such that the RQ-factorization

$$X = RQ = \begin{bmatrix} L & O \end{bmatrix} Q,$$

satisfies $L = \alpha I$ for some $\alpha \in \mathbb{R}$. Let $x \in \mathbb{R}^n$, $v \in \mathbb{R}^m$ and $w \in \mathbb{R}^{n+k-m}$ such that

$$Q \begin{bmatrix} o \\ x \end{bmatrix} = \begin{bmatrix} v \\ w \end{bmatrix}, \tag{5.35}$$

and hence, $y = \alpha v$, by Theorem 5.3.2, assertion 1. Since $X$ is real, the RQ-factorization $X = RQ$ is unique with both $Q$ and $R$ real [111, p. 516].

When we obtain $Q$ by the iterative RQ-factorization algorithm, $Q$ is a concatenation of fast $\mu$-rotations and can be written as

$$Q = F(p_1, q_1, \theta_{k_1}, \gamma_1) \cdots F(p_{i_{\max}}, q_{i_{\max}}, \theta_{k_{i_{\max}}}, \gamma_{i_{\max}}), \tag{5.36}$$

where $1 \leq p_i < q_i \leq n + k$, $0 \leq \theta_{k_i} \leq \frac{\pi}{4}$ and $\gamma_i \in \{0, \pi\}$ for all $i = 1, \ldots, i_{\max}$. Hence, from (5.35) and (5.36) we conclude that the matrix-vector product $y = Ax$ can be computed by repeatedly performing $\mu$-rotations on the (extended) input data, i.e.,

$$\begin{bmatrix} y \\ \alpha w \end{bmatrix} = \alpha F(p_1, q_1, \theta_{k_1}, \gamma_1) \cdots F(p_{i_{\max}}, q_{i_{\max}}, \theta_{k_{i_{\max}}}, \gamma_{i_{\max}}) \begin{bmatrix} o \\ x \end{bmatrix}.$$

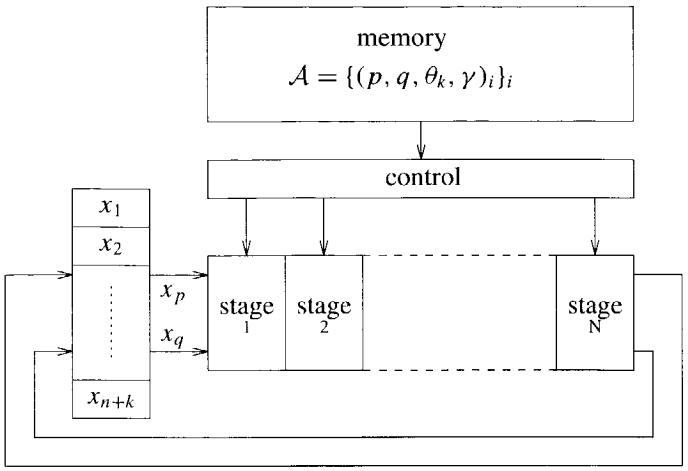



**Figure 5.10:** *Architecture for computing $y = Ax$ using unitary factorizations.*

Since a $\mu$-rotation is fully characterized by a quadruple $(p, q, \theta_k, \gamma)$, we have to store all quadruples $(p, q, \theta_k, \gamma)_i$, $1 \leq i \leq i_{\max}$, in some storage medium in order to compute $y = Ax$. Let $x = (x_j)_{j=1}^{n+k}$ denote the (extended) input data and $\mathcal{A} = \{(p, q, \theta_k, \gamma)_i\}_i$ the set of all quadruples $(p, q, \theta_k, \gamma)_i$, Figure 5.10 shows the architecture which can be used to compute (5.35). Each stage in the pipeline is one single generalized stage (see subsection 5.2.3), which itself is controlled by a $4 + {}^2\log s$ bit sequence $\{s_0, s_1, k, \varsigma_u, \varsigma_l\}$.

## 5.6.2 Concurrent processing

Consider the architecture depicted in Figure 5.10. If we take $N = 1$, i.e., the pipeline consists of one single stage, the architecture is purely sequential. Thus a type III $\mu$-rotation is performed in three cycles. By taking $N > 1$, we can introduce concurrent processing. However, we then have to take some precautions in the way we compute the RQ-factorization. This can be seen as follows. Assume that $N = 2$ and we want to compute a method III $\mu$-rotation. Naturally, we select elements $x_p$ and $x_q$ and feed them into the first stage. In the next cycle, the processed data are fed into the second stage while the input data needed for the succeeding $\mu$-rotation can be fed into the first stage at the same time. It is clear that a causality problem arises when either $x_p$ or $x_q$ are needed for this succeeding $\mu$-rotation since both data are being processed and, therefore, are tem-

| $(p, q)$ | $(\theta_{\text{opt}}, \gamma)$ | $\Delta_{p,q}$ |
|---|---|---|
| (1,2) | $(0.3218, 0)$ | $\Delta_{1,2} = 0.0833$ |
| (1,3) | $(0.4636, \pi)$ | $\Delta_{1,3} = 0.1667$ |
| (1,4) | $(0.4636, \pi)$ | $\Delta_{1,4} = 0.1667$ |
| (2,3) | $(0.7854, 0)$ | $\Delta_{2,3} = 0.2500$ |
| (2,4) | $(0.7854, 0)$ | $\Delta_{2,4} = 0.2500$ |

**Table 5.2:** *Possible indices, optimal angles and the corresponding reduction in the off-diagonal energy of $X$ given by (5.37).*

porarily not available. We can overcome this problem by modifying the procedure *compute_opt_angle* $(\cdot)$. After doing some bookkeeping, we select the quadruple $(p, q, \theta, \gamma)$ which maximizes (5.25) subject to the condition that the indices $p$ and $q$ are not in the pipeline. This solution, however, has the disadvantage that it influences the reduction of the off-diagonal energy inefficiently when the length of the pipeline becomes large relative to the number of columns in the augmented matrix $X$. This is illustrated by the following example. Assume that $X$ is given by

$$X = \frac{1}{6}\begin{bmatrix} 2\sqrt{6} & 0 & \sqrt{6} & \sqrt{6} \\ 3 & 3 & -3 & -3 \end{bmatrix}, \qquad XX^* = I, \tag{5.37}$$

and that the pipeline depth equals two. To select the optimal quadruple $(p, q, \theta, \gamma)$ we compute the optimal pair $(\theta_k, \gamma)$ for any index pair $(p, q)$ and the corresponding reduction in the off-diagonal energy of $X$. The results of these calculations are shown in Table 5.2. Obviously, for the first iteration we can choose either (2,3) or (2,4) as the index pair. Assuming that we choose the pair (2,3) (the essence of the discussion would not change if we had chosen the pair (2,4)). To select an index pair for the second iteration, we have to update Table 5.2. However, irrespective of the new values of $(\theta_k, \gamma)$ and $\Delta_{p,q}$, only the index pair (1,4) is left for the next iteration since indices 2 and 3 are currently being processed in the pipeline and therefore cannot be used. For the third iteration we have a similar situation since the indices 1 and 4 would not be available so that we would have to choose the index pair (2,3) again, and so forth. Clearly this procedure can cause some oscillation with the consequence that we can only reduce the off-diagonal energy of $X$ locally. Eventually, after the elements $x_{1,4}$ and $x_{2,3}$ have been made completely zero (up to the working precision) we have to introduce some dummy data into the pipeline to be able to nullify the other elements of $X$. We can partly overcome this undesired behaviour by introducing some queueing mechanism which has to

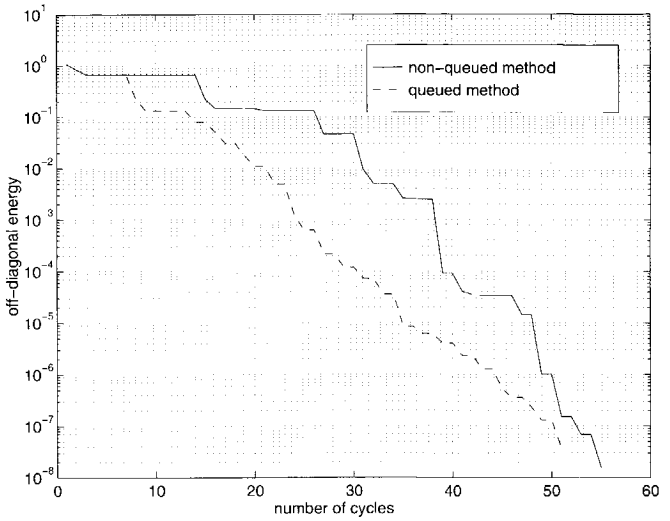



**Figure 5.11:** *Reduction of the off-diagonal energy of (5.37) with and without a queueing mechanism.*

ensure that the reduction in the off-diagonal energy is not held up by rotations which give little reduction. This can be done by feeding dummy data into the pipeline whenever the index pair $(p, q)$ corresponding to the maximum reduction in off-diagonal energy could cause a causality problem. Figure 5.11 shows the results with $s = 12$. In the non-queued situation, convergence is very slow during the first cycles, while in the queued situation this problem does not occur: the convergence is now more uniform. As the pipeline depth gets smaller relative to the number of columns of $X$, both methods will perform more equally since in that case there is enough freedom left to select a proper index pair $(p, q)$ at each iteration step. In the remainder of this chapter we proceed with the queued method.

### 5.6.3 Fixed-length versus variable-length rotation angle

The number of elementary stages needed to perform a rotation over a given angle increases as the angle gets larger. We will refer to this number as the length of the angle. Thus a type II $\mu$-rotation rotates over length two angles. In the case when the angle length is not a multiple of the number of stages in the pipeline, a

rotation over this angle will be completed before the processed data has reached the last pipeline stage, the one which is connected to the output bus. This means that we have to pass the processed data through the remaining stages. However, since these data cannot be addressed as long as they are in the pipeline, we can use these remaining stages to further reduce the off-diagonal energy. As an example, suppose the pipeline depth is three and we want to rotate over a length four angle. We then have to feed back the processed data into the pipeline after three cycles in order to compute the fourth and last cycle. After this operation, the rotated data are idle for the next two clock cycles. We can exploit this by performing, for example, one additional type II $\mu$-rotation or two additional type I $\mu$-rotations. By doing so we can further reduce the total number of cycles needed to zero off-diagonal entries. In fact, with this method we actually aim to use fixed-length angles, of which the length equals the pipeline depth, instead of using variable-length angles.

We illustrate the use of both methods by showing the results of applying the iterative RQ-factorization to LOTs. Thus, we factor $A \in \mathbb{C}^{K \times 2K}$ as in (5.16), (5.18) and (5.20) to obtain matrices $U_{1,1} \in \mathbb{C}^{\frac{K}{2} \times \frac{K}{2}}$, $B_u \in \mathbb{C}^{\frac{K}{2} \times \frac{K}{2}}$ and $B_a \in \mathbb{C}^{\frac{K}{2} \times \frac{K}{2}}$. The iterative RQ-factorization is then applied to these matrices.

Figures 5.12 and 5.13 show the results of using variable-length and fixed-length angles, respectively, in terms of off-diagonal energy compared with the number of cycles for the factorization of the matrix $B_a$ when $K = 32$, $s = 12$ and the depth of the pipeline varies from one to four. Figures 5.14 and 5.15 show these results for the factorization of $B_u$, and Figures 5.16 and 5.17 for the factorization of $U_{1,1}$. It can be seen that an increase in the pipeline depth decreases the total number of cycles needed to make the off-diagonal energy sufficiently small. The reason for this is that during the first iterations the rotations are performed over relatively large angles in method II, III or IV $\mu$-rotations which take more than one elementary stage. This means that we can process succeeding $\mu$-rotations in parallel, provided that they do not give causality conflicts with any of the elements being processed in the pipeline. As discussed previously, an increase in the pipeline depth decreases the degree of freedom we have in choosing the optimal quadruple $(p, q, \theta, \gamma)$, therefore, to minimize the off-diagonal energy. There is, therefore, a pipeline depth beyond which no further increase in efficiency can be obtained. Thus, when using fixed-length angles, the optimal pipeline depth for $B_a$, $B_u$ and $U_{1,1}$ equals four. Clearly, the optimal pipeline depth depends on the dimensions of the matrix to be factorized.

The number of cycles needed to zero the off-diagonal entries also depends on the required working precision $s$. If we increase the value of $s$, we need more

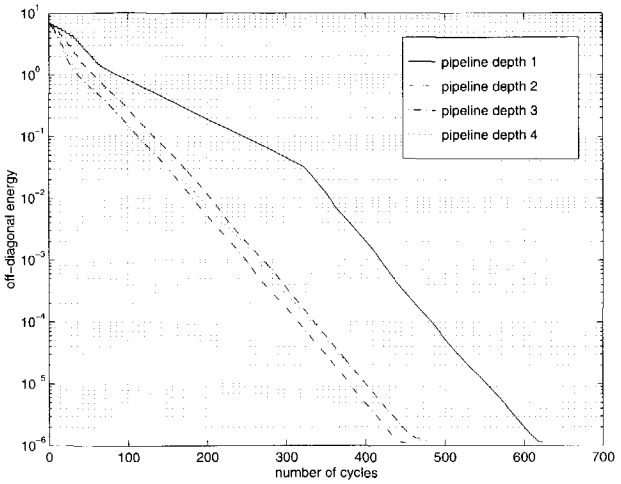



**Figure 5.12:** *Factorization of $B_a$ using variable-length angles: reduction of the off-diagonal energy compared with the number of cycles, for one to four pipeline stages.*

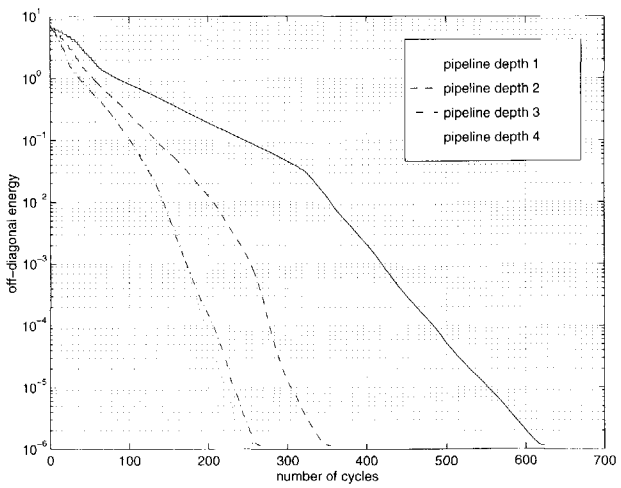



**Figure 5.13:** *Factorization of $B_a$ using fixed-length angles: reduction of the off-diagonal energy compared with the number of cycles, for one to four pipeline stages.*

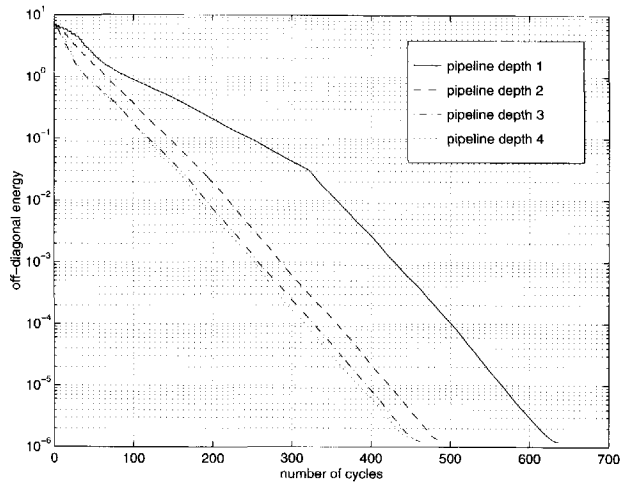



**Figure 5.14:** *Factorization of $B_u$ using variable-length angles: reduction of the off-diagonal energy compared with the number of cycles, for one to four pipeline stages.*

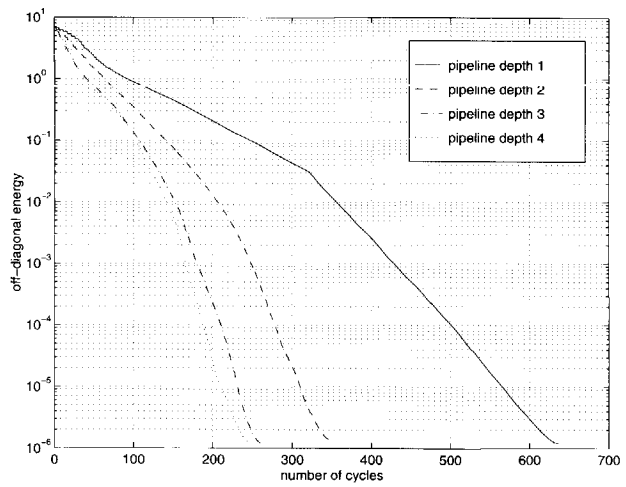



**Figure 5.15:** *Factorization of $B_u$ using fixed-length angles: reduction of the off-diagonal energy compared with the number of cycles, for one to four pipeline stages.*

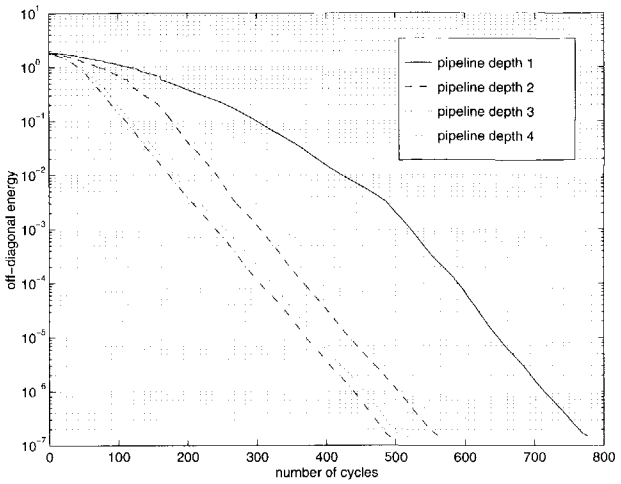



**Figure 5.16:** *Factorization of $U_{1,1}$ using variable-length angles: reduction of the off-diagonal energy the number of cycles, for one to four pipeline stages.*

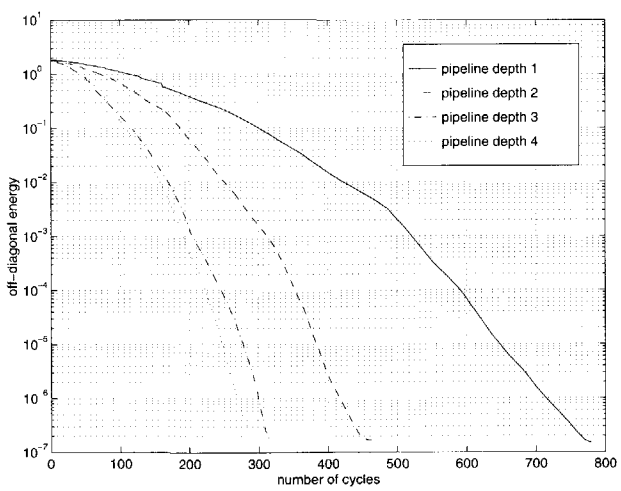



**Figure 5.17:** *Factorization of $U_{1,1}$ using fixed-length angles: reduction of the off-diagonal energy compared with the number of cycles, for one to four pipeline stages.*

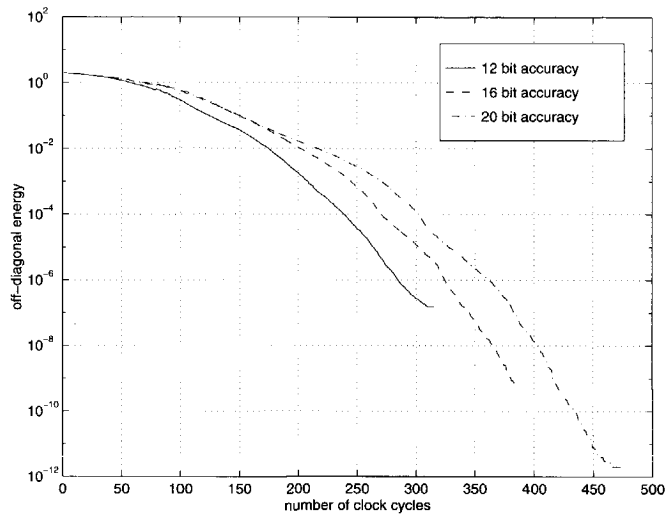



**Figure 5.18:** *Influence of the working precision on the number of iterations.*

accurate, and thus computationally more complex, $\mu$-rotations to implement the rotations. For example, a rotation over an angle which can be done using a type I $\mu$-rotation with 12 bit accuracy may need a type II $\mu$-rotation if the required accuracy becomes 16 bit. Hence, an increase in the accuracy results in an increase in the number of cycles. As a consequence, the proposed architecture makes it possible to trade-off between time and computational accuracy. Figure 5.18 shows the influence of different working precisions $s$ on the factorization of the matrix $U_{1,1}$. The pipeline depth has been fixed to 3 stages.

### 5.6.4 Approximated LOTs

We can further reduce the number of cycles by exploiting the orthonormality properties of the rows of the LOT matrix $A$, which are the filters of a $K$-channel paraunitary filter bank. The $K$ filters jointly satisfy the following property

$$\sum_{k=1}^{K} |\hat{h}_k(\xi)|^2 = 1, \qquad \text{for all } \xi \in (-\pi, \pi]. \tag{5.38}$$

Differentiation of (5.38) then yields

$$\sum_{k=1}^{K} |\hat{h}_k(\xi)| \partial |\hat{h}_k(\xi)| = 0.$$

Therefore, if $\hat{h}_k$s are frequency selective functions in the sense that $|\hat{h}_k(\xi)| \approx 0$ for all $\xi \notin \Pi_k$, where $\Pi_k$ denotes the pass-band region of filter $\hat{h}_k$, and $|\hat{h}_l(\xi)| \approx 0$ for all $\xi \in \Pi_k$, $k \neq l$, then

$$\partial |\hat{h}_k(\xi)| \approx 0, \quad \text{for all } \xi \in \Pi_k,$$

for all $k = 1, \ldots, K$. This means that a first-order perturbation of the filter parameters not only preserves orthogonality, but it also does not effect the behaviour of $|\hat{h}_k(\xi)|$, at least in its pass-band region.

We can exploit this property by terminating the iterative RQ-factorization before it converges. This can be seen as follows.

Let $\hat{Q}$ denote the concatenation of the first, say $n$, rotations. If we now apply $\hat{Q}$ to the (extended) input data rather than $Q$ itself, i.e.,

$$\hat{Q} \begin{bmatrix} o \\ x \end{bmatrix} = \begin{bmatrix} \hat{v} \\ \hat{w} \end{bmatrix},$$

the result $\hat{y} = \alpha \hat{v}$ can be regarded as the output of a filter bank with filters as approximations of the rows of $A$. Indeed, let $X = A$ (we do not have to augment $A$ here since the rows of $A$ already form an orthogonal set) with RQ-factorization $X = QR$, where $R = \alpha [\, I \quad O \,]$ for some $\alpha \in \mathbb{R}$, we define $\hat{A} = R\hat{Q}$. We then have

$$\begin{aligned} \hat{A}x &= \hat{A}\hat{Q}^* \hat{Q} x \\ &= R \begin{bmatrix} \hat{v} \\ \hat{w} \end{bmatrix} \\ &= \alpha \hat{v}. \end{aligned}$$

Clearly, termination of the iterative RQ-factorization causes a perturbation in the entries of $A$ and thus of the underlying filter coefficients, although, without violating the orthogonality property. Therefore, for small perturbations, the Fourier transforms of the rows of $\hat{A}$, the perturbed filters, closely match the Fourier transforms of the original filters, at least in their pass-band region. The following example serves to illustrate this important property. Figures 5.19 and 5.20 show approximation results of the LOT which was constructed with approximations of

$B_a$, $B_u$ and $U_{1,1}$, using a pipeline depth of four. Without any approximation in the above sense, the number of cycles can be read-off from Figures 5.13, 5.15 and 5.17, respectively. Thus, the total number of cycles needed to factorize the matrices $B_a$, $B_u$ and $U_{1,1}$ "completely" is approximately $N = 800$. The approximation of the LOT functions is obtained by simply terminating the factorization of each of the three submatrices after an equal number of cycles. It is not investigated here whether there is a better allocation of the number of cycles to the submatrices. For a detailed study of this problem, the reader is referred to [129, 130]. It can be seen seen from the figures that the behaviour of the filters in the pass-band region is insensitive to small parameter perturbations and that after $\frac{3}{4}$ of the number of cycles $N$ (here 800) for a "complete" factorization, the approximated filters become suitable filter-bank filters. In fact, the difference in the frequency domain between the "original" filters and the approximated ones after $N$ and $\frac{3}{4}N$ cycles, respectively, is less than $-50$ dB.

## 5.7 Implementation

In the previous sections, we touched upon the design of realizations of discrete-time signal transforms. As discussed in Section 5.2, an optimal realization must be capable of being efficiently mapped onto silicon. In this section we discuss issues of complexity such as layout and total silicon area of using the LOT, and compare these results with alternative implementations. This section is not intended to provide a detailed description of the chip design, but serves to affirm the fact that the above realizations are suitable for low-cost high-accuracy VLSI implementation.

In [131], an architecture for implementing the complete LOT is discussed. The architecture consists of a linear array of pipelined rotation processors, all working in parallel. Each pipelined processor consists of a register to store the input, output and intermediate results, a pipeline consisting of a concatenation of four generalized $\mu$-rotation stages (see Section 5.2.3) and a simple routing network which routes the data between the rotation pipe, the register and the input/output ports. The rotation processors themselves are governed by a global controller. This global controller stores the entire control sequence for the transform and delivers it to the first processor unit in the array of processors when the image data arrive. The control is then skewed in time and delivered to subsequent processor units by means of FIFOs. This architecture greatly reduces the control overhead as opposed to each processor having its own stored control sequence.

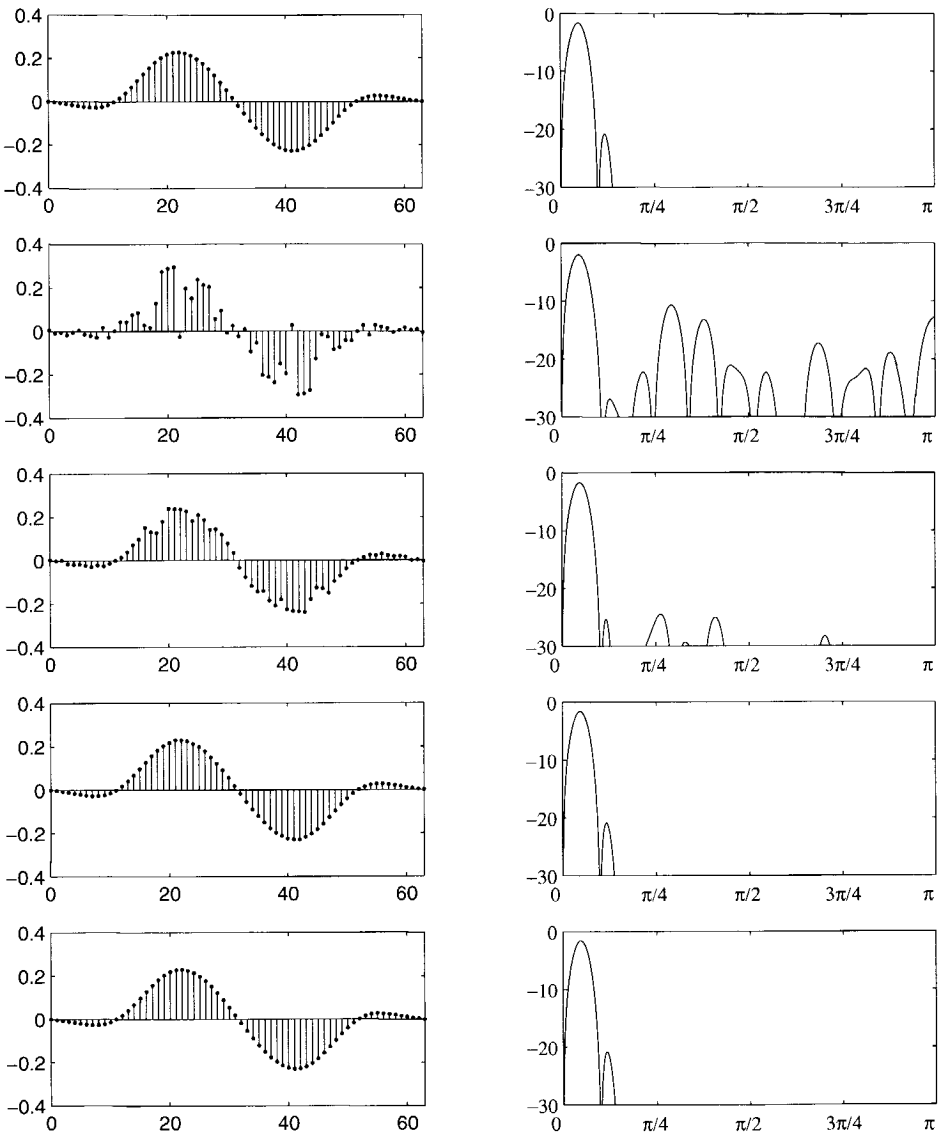



**Figure 5.19:** *Approximate filter responses of a size* $32 \times 64$ *LOT. The upper figure shows the original (low-frequency) filter response in time (left) and frequency (right) domain. The other figures show, from top down, the approximations after* $\frac{1}{4}N$, $\frac{1}{2}N$, $\frac{3}{4}N$ *and* $N$ *clock cycles;* $N = 800$.

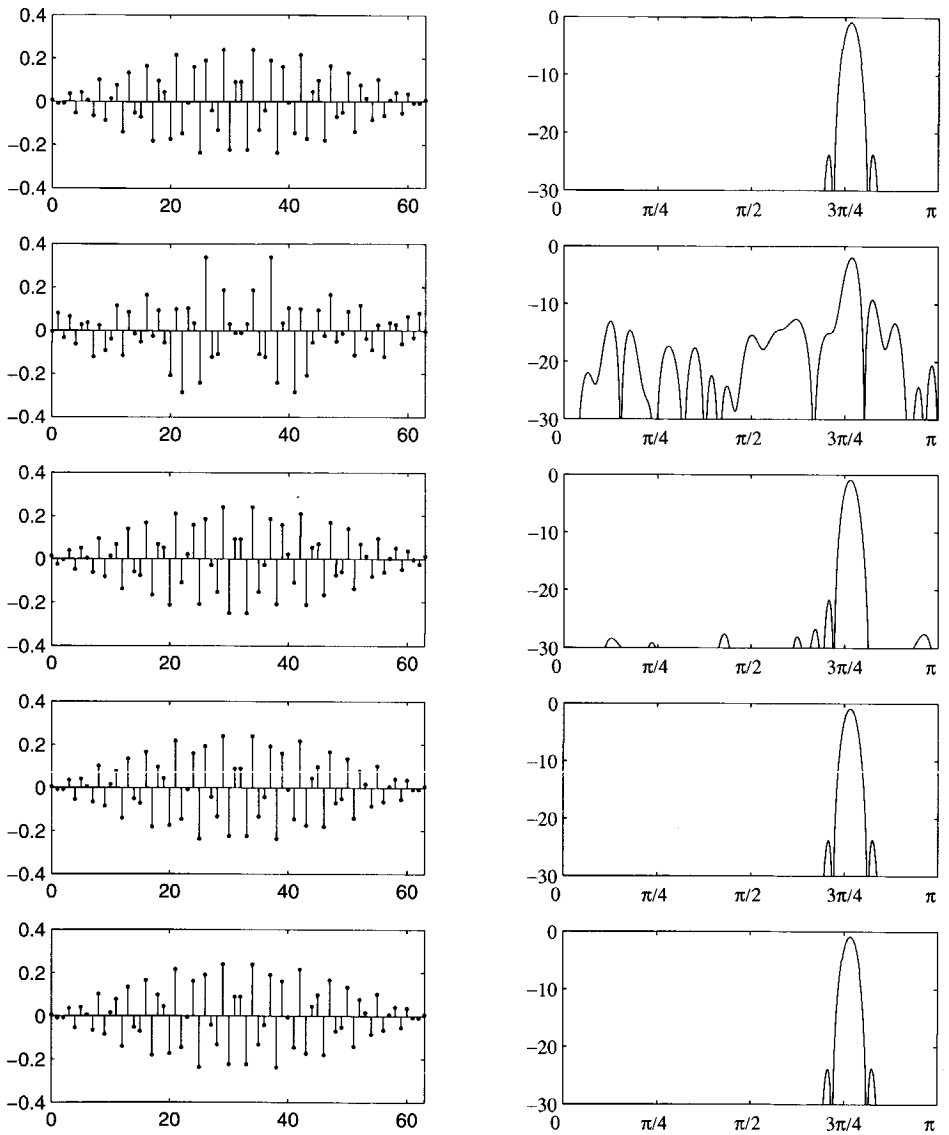



**Figure 5.20:** *Approximate filter responses of a size* $32 \times 64$ *LOT. The upper figure shows the original (high-frequency) filter response in time (left) and frequency (right) domain. The other figures show, from top down, the approximations after* $\frac{1}{4}N$, $\frac{1}{2}N$, $\frac{3}{4}N$ *and* $N$ *clock cycles;* $N = 800$.

Moreover, since the storage for the control sequence is implemented on a RAM, the system is (re)programmable for arbitrary signal transforms.

At Delft University of Technology a chip has been designed in $0.8\mu$-CMOS (Compass, ES2) that contains ten pipelined rotation processors. The chip operates at a clock frequency of 50 MHz, which is suitable for real-time signal transformation of images with

- an image rate of up to 30 Hz,
- an image resolution of up to $1024 \times 1024$ pixels,
- an pixel resolution of up to 10 bits per pixel.

The total chip area for this chip is about 160 $mm^2$ for a one-dimensional LOT. Currently, a prototype with three rotation processors is being produced.

In the remainder of this section, we compare the required chip area of the chip described above with alternative implementations, viz., a conventional implementation based on multiply-accumulate arithmetic and an implementation based on a TriMedia digital signal processor (DSP). We discuss both alternatives in more detail below.

### Conventional multiply-accumulate arithmetic implementation

A hardware feasibility study is reported in [132, 133] for a DCT based LOT, i.e., a LOT in which the matrix $U$ is taken to be the DCT. The block size of this LOT can be switched between 8, 16 and 32. The resulting chip operates at a clock frequency of 35 MHz and is capable of processing images of $1024 \times 1024$ pixels, with a pixel resolution of 10 bits and an image rate of 30 Hz. The required chip area in $0.8\mu$-CMOS (C150dm) is about 160 $mm^2$ for a one-dimensional LOT. In C100tm technology ($0.5\mu$-CMOS), however, the estimated chip area is 40 $mm^2$. A similar reduction in the chip area (factor four) may be expected for the chip designed at the Delft University when the C100tm technology is used.

### TriMedia DSP

The TriMedia TM-1 is a programmable DSP which is intended as a multi-standard video, audio, graphics accelerator for PCI-based personal computers [134]. It can also be used as the master CPU in stand-alone multimedia PCI-bus-based systems. The programming language for the TriMedia DSP family is C. The TriMedia compilation system translates C programs and generates machine code

for a particular TriMedia chip. The generated coded can be verified by computer simulation with the TriMedia simulator. The implementation of a DCT-based $32 \times 64$ LOT on a TM-1 processor is reported in [135]. It is estimated that a one-dimensional LOT could be implemented with four TriMedia DSPs. With a 200 Mhz version of the TriMedia DSP, which has been announced as the successor to the TM-1, this number will be halved to two.

## 5.8 Conclusions

We have investigated the realization of discrete-time signal transforms, in particular the realization of LOTs. We provided a precise definition of optimal realizations and we showed that (minimal) orthogonal realizations are optimal under that definition. We proposed a numerically robust algorithm for the execution of the matrix-vector product $y = Ax$ and applied this to the LOT or, more precisely, to its constituent matrices $A_e$, $A_o$ and $B_a$. The algorithm is based on the RQ-factorization of a properly chosen matrix $X$ which is computed using a concatenation of $\mu$-rotations. The realizations thus obtained are low-cost in terms of the number of operations and the implementation complexity of the operations. The total chip area needed for these realizations is about 160 mm$^2$ for a one-dimensional LOT. This is similar to the chip area obtained by using conventional multiply-accumulate based realizations. Our realization, however, has two major advantages over conventional ones. Firstly, the realizations can be made fully programmable in the sense that the architecture can be used for arbitrary signal transforms, including DCTs, LOTs or discrete wavelet transforms. Secondly, since both the analysis and synthesis operator can be given the same realizations, we can save 50% of the total chip area in applications where both mappings are not computed simultaneously.

# *Bibliography*


[1] P. Roos, M.A. Viergever, M.C.A. van Dijke, and J.H. Peters. Reversible intraframe compression of medical images. *IEEE Trans. on Medical Imaging*, 7(4):328–336, December 1988.

[2] P. Roos. *Reversible Compression of Medical Images.* PhD thesis, Delft University of Technology, Delft, The Netherlands, December 1991.

[3] CCIR (International Radio Consultative Committee). *Encoding Parameters of Digital Television for Studios.* 1982. Recommendation 601-1.

[4] N.S. Jayant and P. Noll. *Digital Coding of Waveforms; Principles and Applications to Speech and Video.* Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1984.

[5] G.K. Wallace. The JPEG still picture compression standard. *Communications of the ACM*, 34(4):30–44, April 1991.

[6] W.B. Pennebaker and J.L. Mitchell. *JPEG, Still Image Data Compression Standard.* van Nostrand Reinhold, New York, 1993.

[7] D. Le Gall. MPEG: A video compression standard for multimedia applications. *Communications of the ACM*, 34(4):47–58, April 1991.

[8] *ISO/IEC 11172, Coding of Moving Pictures and Associated Audio for Storage at up to about 1.5 Mbit/s.* November 1992.

[9] *ISO/IEC 13818, Generic Coding of Moving Pictures and Associated Audio Information.* November 1994.

[10] H.S. Malvar and D.H. Staedlin. The LOT: Transform coding without blocking effects. *IEEE Trans. on ASSP*, 37(4):553–559, April 1989.

[11] H.S. Malvar. Lapped transforms for efficient transform/subband coding. *IEEE Trans. on ASSP*, 38(6):969–978, June 1990.

[12] H.S. Malvar. *Signal Processing with Lapped Transforms*. Artech House, Boston, 1992.

[13] S.G. Mallat. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(7):674–693, July 1989.

[14] I. Daubechies. The wavelet transform, time-frequency localization and signal analysis. *IEEE Trans. on Information Theory*, 36(5):961–1005, September 1990.

[15] I. Daubechies. *Ten Lectures on Wavelets*. SIAM, Philadelpia, Pennsylvania, 1992.

[16] R.E. Crochiere, S.A. Webber, and J.L. Flanagan. Digital coding of speech in subbands. *Bell Systems Technical Journal*, 55(8):1069–1085, October 1976.

[17] J.W. Woods and S.D. O'Neil. Subband coding of images. *IEEE Trans. on ASSP*, 34(5):1278–1288, October 1986.

[18] P.H. Westerink. *Subband Coding of Images*. PhD thesis, Delft University of Technology, Delft, The Netherlands, October 1989.

[19] J.W. Woods. *Subband Image Coding*. Renselaer Polytechnic Institute Troy, New York, 1991.

[20] P.P. Vaidyanathan. *Multirate Systems and Filter Banks*. Prentice Hall Signal Processing Series. Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1993.

[21] M. Vetterli and J. Kovačević. *Wavelets and Subband Coding*. Prentice Hall Signal Processing Series. Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1995.

[22] C.F. Shannon and W. Weaver. *The Mathematical Theory of Communication*. The University of Illinois Press, Urbana, 1949.

[23] J. Max. Quantization for minimum distortion. *IEEE Trans. on Information Theory*, 6:7–12, 1960.

[24] S.P. Lloyd. Least squares quantization in PCM. *IEEE Trans. on Information Theory*, 28:129–137, 1982.

[25] R.M. Gray. Vector quantization. *IEEE ASSP Magazine*, 1(2):4–29, 1982.

[26] R.N.J. Veldhuis and M. Breeuwer. *An Introduction to Source Coding*. Prentice Hall, New York, 1993.

[27] D.A. Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9):1098–1103, September 1952.

[28] J. Ziv and A. Lempel. A universal algorithm for sequential data compression. *IEEE Trans. on Information Theory*, 23(3):337–343, May 1977.

[29] J. Ziv and A. Lempel. Compression of individual sequences via variable-rate coding. *IEEE Trans. on Information Theory*, 24(5):530–536, September 1978.

[30] T.A. Welch. A technique for high-performance data compression. *Computer*, pages 8–19, June 1984.

[31] N. Abramson. *Information Theory and Coding*. McGraw-Hill Book Company, Inc., New York, 1963.

[32] J.J. Rissanen. Generalized Kraft inequality and arithmetic coding. *IBM J. Res. Develop.*, 20(3):198–203, May 1976.

[33] G.G. Langdon. An introduction to arithmetic coding. *IBM J. Res. Develop.*, 28(2):135–149, March 1984.

[34] I.H. Witten, R.M. Neal, and J.G. Cleary. Generalized Kraft inequality and arithmetic coding. *Communications of the ACM*, 30(6):520–540, June 1987.

[35] R.J. Clarke. *Transform Coding of Images*. Academic Press, London, U.K., 1985.

[36] H.S. Malvar. The LOT: a link between block transform coding and multirate filter banks. In *Proceedings IEEE Int. Symp. on Circuits and Systems*, pages 835–838, 1988.

[37] R. Padmanabha Rao and W.A. Pearlman. On entropy of pyramid structures. *IEEE Trans. on Information Theory*, 37(2):407–413, March 1991.

[38] S. Nanda and W.A. Pearlman. Tree coding of image subbands. *IEEE Trans. on Image Processing*, 1(2):133–147, April 1992.

[39] T.R. Fischer. On the rate-distortion efficiency of subband coding. *IEEE Trans. on Information Theory*, 38(2):426–428, March 1992.

[40] A. Divakaran and W.A. Pearlman. Information-theoretic performance of quadrature mirror filters. *IEEE Trans. on Information Theory*, 41(6):2094–2100, November 1995.

[41] M. Vetterli. A theory of multirate filter banks. *IEEE Trans. on ASSP*, 35(3):356–372, March 1987.

[42] M. Vetterli and D. Le Gall. Perfect reconstruction FIR filter banks: Lapped transforms, pseudo QMFs and paraunitary matrices. In *Proceedings IEEE Int. Conf. on ASSP*, pages 2249–2253, 1988.

[43] M. Vetterli and D. Le Gall. Perfect reconstruction FIR filter banks: Some properties and factorizations. *IEEE Trans. on ASSP*, 37(7):1057–1071, July 1989.

[44] P.P. Vaidyanathan. Multirate digital filters, filter banks, polyphase networks, and applications: A tutorial. *Proceedings IEEE*, 78(1):56–93, January 1990.

[45] A.N. Kolmogorov. On the Shannon theory of information transmission in the case of continuous signals. *IRE Trans. on Information Theory*, 2:102–108, December 1956.

[46] R.G. Gallager. *Information Theory and Reliable Communication.* John Wiley and Sons, Inc., New York, 1968.

[47] T. Berger. *Rate Distortion Theory; A Mathematical Basis for Data Compression.* Prentice Hall Series in Information and System Science. Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1971.

[48] D.J. Sakrison. The rate distortion function of a Gaussian process with a weighted square error criterion. *IEEE Trans. on Information Theory*, 14:506–508, May 1968.

[49] D.G. Luenberger. *Introduction to Linear and Nonlinear Programming.* Addison-Wesley Publishing Company, 1973.

[50] T.A. Ramstad. IIR filterbank for subband coding of images. In *Proceedings IEEE Int. Symp. on Circuits and Systems*, pages 827–830, 1988.

[51] N. Ahmed, T. Natarajan, and K.R. Rao. Discrete cosine transform. *IEEE Trans. on Computers*, pages 90–93, January 1974.

[52] R. Heusdens. Design of lapped orthogonal transforms. *IEEE Trans. on Image Processing*, 5(8):1281–1284, August 1996.

[53] Z. Cvetković. Oversampled modulated filter banks and tight Gabor frames in $l^2(\mathbb{Z})$. In *Proceedings IEEE Int. Conf. on ASSP*, pages 1456–1459, Detroit, MI (USA), May 1995.

[54] Z. Cvetković and M. Vetterli. Oversampled filter banks. December 1994. Submitted to IEEE Trans. on Signal Processing.

[55] M. Vetterli and Z. Cvetković. Oversampled FIR filter banks and frames in $l^2(\mathbb{Z})$. In *Proceedings IEEE Int. Conf. on ASSP*, Atlanta, GA (USA), May 1996.

[56] H. Bölcskei, F. Hlawatsch, and H.G. Feichtinger. Frame-theoretic analysis of filter banks. February 1996. Submitted to IEEE Trans. on Signal Processing.

[57] H. Bölcskei, F. Hlawatsch, and H.G. Feichtinger. Oversampled FIR and IIR DFT filter banks and Weyl-Heisenberg frames. In *Proceedings IEEE Int. Conf. on ASSP*, Atlanta, GA (USA), May 1996.

[58] R.N.J. Veldhuis. A vector-filter notation for analysis/synthesis systems and its relation to frames. Technical Report 909, Institute for Perception Research, Eindhoven, June 1993.

[59] N.J. Munch. Noise reduction in tight Weyl-Heisenberg frames. *IEEE Trans. on Information Theory*, 38(2):608–616, March 1992.

[60] R.A. Roberts and C.T. Mullis. *Digital Signal Processing.* Addison-Wesley Publishing Company, 1987.

[61] L. Vandendorpe. CQF filter banks matched to signal statistics. *Signal Processing*, 29(3):237–249, December 1992.

[62] V.P. Sathe and P.P. Vadyanathan. Effects of multirate systems on the statistical properties of random signals. *IEEE Trans. on Signal Processing*, 41(1):131–146, January 1993.

[63] R.A. McDonald and P.M. Schultheiss. Information rates of Gaussian signals under criteria constraining the error spectrum. *Proceedings IEEE*, 52:415–416, April 1964.

[64] H.G.J. Theunis and E.F. Deprettere. Piecewise stationary perfect reconstruction filter banks. *AEÜ Int. Journal of Electr. and Commun*, 49(5/6):344–361, 1995.

[65] A.N. Lemma and E.F. Deprettere. Time-varying biorthogonal filter banks, a state-space approach. *IEEE Trans. on Circuits and Systems*, 1996.

[66] P. Cassereau. A new class of optimal unitary transforms for image processing. Master's thesis, Mass. Inst. Technology, Cambridge, May 1985.

[67] R.W. Young and N.G. Kingsbury. Video compression using lapped transforms for motion estimation/compensation and coding. In *Proceedings SPIE Visual Communications and Image Processing*, pages 276–288, 1992.

[68] H.S. Malvar. Fast algorithms for modulated lapped trasnforms. *Electronic Letters*, 27(9):775–776, April 1991.

[69] H.S. Malvar. Extended lapped transforms: Fast algorithms and applications. In *Proceedings IEEE Int. Conf. on ASSP*, pages 1797–1800, 1991.

[70] T.A. Ramstad and J.P. Tanem. Cosine-modulated analysis-synthesis filterbank with critical sampling and perfect reconstruction. In *Proceedings IEEE Int. Conf. on ASSP*, pages 1789–1792, Toronto, Ontario, Canada, May 1991.

[71] R.D. Koilpillai and P.P. Vaidyanathan. New results on cosine-modulated FIR filterbanks satisfying perfect reconstruction. In *Proceedings IEEE Int. Conf. on ASSP*, pages 1793–1796, July 1991.

[72] R.D. Koilpillai and P.P. Vaidyanathan. Cosine-modulated FIR filter banks satisfying perfect reconstruction. *IEEE Trans. on Signal Processing*, 40(4):770–783, April 1992.

[73] D.J. Sakrison and V.R. Algazi. Comparison of line-by-line and two-dimensional encoding of random images. *IEEE Trans. on Information Theory*, 17(4):386–398, July 1971.

[74] J.L. Mannos and D.J. Sakrison. The effect of a visual fidelity criterion on the encoding of images. *IEEE Trans. on Information Theory*, 20(4):525–536, July 1974.

[75] A.N. Netravali and B. Prasada. Adaptive quantization of picture signals using spatial masking. *Proceedings IEEE*, 65(4):536–548, April 1977.

[76] S.P. Lipshitz, J. Vanderkooy, and R.A. Wannamaker. Minimally audible noise shaping. *Journal of the Audio Engineering Society*, 39(11):836–852, November 1991.

[77] S.P. Lipshitz, R.A. Wannamaker, and J. Vanderkooy. Quantization and dither: A theoretical survey. *Journal of the Audio Engineering Society*, 40(5):355–375, May 1992.

[78] R.A. Wannamaker. Psychoacoustically optimal noise shaping. *Journal of the Audio Engineering Society*, 40(7/8):611–620, 1992.

[79] A.K. Soman, P.P. Vaidyanathan, and T.Q. Nguyen. Linear phase paraunitary filter banks: Theory, factorization and designs. *IEEE Trans. on Signal Processing*, 41(12):3480–3496, December 1993.

[80] A.A.C.M. Kalker and I.A. Shah. A group theoretic approach to multidimensional filter banks: Theory and applications. *IEEE Trans. on Signal Processing*, 1994. Scheduled to be published.

[81] H. Bölcskei, R. Heusdens, H.G.J. Theunis, and A.J.E.M. Janssen. A recursive method for the design of perceptually relevant lapped transforms. October 1996. Submitted to IEEE Trans. on Image Processing.

[82] J.H.C. Reiber. Angiography and digital radiography. *Current Opinion in Cardiology*, 4:853–862, 1989.

[83] A.E. James, J.H. Anderson, and C.B. Higgins, editors. *Digital Image Processing in Radiology*. Williams Wilkens, Baltimore, 1985.

[84] B.M. Hemminger, P.L. Molina, M.P. Braeuning, F.C. Detterbeck, T.M. Egan, E.D. Pisano, and D.V. Beard. Clinical applications of real-time volume rendering. In *Proceedings SPIE Medical Imaging*, pages 65–76, San Diego, California (USA), February 1995. Image Display.

[85] F.E. Linhardt. PACS, the Viborg project: clinical experiences with a totally digitalized hospital. PACS related changes of the hospital organization. In *Proceedings SPIE Medical Imaging*, pages 312–320, San Diego, California (USA), February 1995. PACS Design and Evaluation: Engineering and Clinical Issues.

[86] G.J. Blaine, S.M. Moore, J.R. Cox, and R.A. Whitman. Teleradiology via narrow-band integrated services digital network (N-ISDN) and joint photographic experts group (JPEG) image compression. *Journal of Digital Imaging*, 5:156–160, 1992.

[87] A. Docef and M.J.T. Smith. A robust model-based coding technique for ultrasound video. In *Proceedings SPIE Medical Imaging*, pages 203–213, San Diego, California (USA), February 1995. Image Display.

[88] M. Tosetto. Progressive coding of medical images using wavelets and arithmetic coding. Technical Report 032/96, Philips Research Laboratories, Eindhoven, October 1996.

[89] M. Breeuwer, R. Heusdens, R.B.M. Klein Gunnewiek, P. Zwart, and P. Béretta. *An Overlapped Transform-Coding Method for Data Compression of Medical Images*, September 1995. Proposal for the ACR-NEMA evaluation.

[90] C.H. Slump. On the restauration of noise aliasing and moiré distortion in ccd-based diagnostic x-ray imaging. In *Proceedings SPIE Medical Imaging '92*, Newport Beach, California, USA, February 23-27 1992.

[91] R. Heusdens and M. Breeuwer. Adaptive quantization in overlapped transform coding of medical x-ray images. In *Proceedings SPIE Medical Imaging*, pages 245–256, San Diego, California (USA), February 1995. Image Display.

[92] L.N.D. Loo, K. Doi, and C.E. Metz. Effect of unsharp masking on the detection of simple patterns. *Medical Physics*, 12(2):209–214, 1985.

[93] W.H. Chen and W.K. Pratt. Scene adaptive coder. *IEEE Trans. on Communications*, 32(3):225–232, 1984.

[94] W. Skarbek, T. Agui, and M. Nakajima. Compression of dithered binary images using Hilbert scan. *Trans. IEICE*, E-72(11):1235–1242, 1989.

[95] A. Pérez, S. Kamata, and E. Kawaguchi. N-dimensional Hilbert scanning and its applications to data processing. In *Proceedings SPIE Image Processing*, pages 430–441, 1991. Algorithms and Techniques II.

[96] P. Kauff, S. Rauthenberg, R. Ritter, M. Charatishvili, and M. Hahn. An improved coding scheme for studio recording of interlaced and progressive HDTV signals. In *Proceedings of the HDTV workshop*, pages 71–79, Kawasaki, 1992.

[97] K. Grüneberg, H. Klein, J. Nie, J.F. Schlüssler, and P. Stammnitz. Hardware implementation of the framestore and data rate control for a digital HDTV-VCR. In *Proceedings of the HDTV workshop*, pages 123–132, Kawasaki, 1992.

[98] G.J. Keesman, I.A. Shah, and R. Klein Gunnewiek. Bit-rate control for MPEG encoders. *Signal Processing: Image Communications*, 6(6):1906–1917, February 1995.

[99] G.J. Keesman. *Multi-Program Video Data Compression*. PhD thesis, Delft University of Technology, Delft, The Netherlands, October 1995.

[100] M. Breeuwer and R. Klein Gunnewiek. Bit-rate control in overlapped transform coding of medical x-ray images. In *Proceedings SPIE Medical Imaging*, pages 490–500, San Diego, California (USA), February 1995. Image Display.

[101] P. Beretta. Image data compression in cardiac angiography. Technical note, Laboratoire de Radiologie Expérimentale, Lyon (F), 1992.

[102] J.E. Volder. The CORDIC trigoniometric computing technique. *IRE Trans. on Electronic Computers*, 8:330–334, September 1959.

[103] J.S. Walter. An unified algorithm for elementary functions. *Proceedings Spring Joint Computer Conference*, 38:397, 1971. AFIPS press.

[104] E.F. Deprettere, P. Dewilde, and R. Udo. Pipelined CORDIC architectures for fast VLSI filtering and array processing. In *Proceedings IEEE Int. Conf. on ASSP*, pages 41A.6.1–41A.6.4, 1984.

[105] J. Götze and G.J. Hekstra. Adaptive approximate rotations for computing the symmetric evd. In M. Moonen and F. Catthoor, editors, *Algorithms and Parallel VLSI Architectures III*, pages 73–84. Elsevier, 1994.

[106] J. Götze and G.J. Hekstra. An algorithm and architecture based on orthonormal $\mu$-rotations for computing the symmetric EVD. *Integration, the VLSI Journal*, 20:21–39, 1995.

[107] W. Givens. Computation of plane unitary rotations transforming a general matrix to triangular form. *J. Soc. Indust. Appl. MAth.*, 6(1):26–50, March 1958.

[108] M. Gentleman. Error analysis of QR-decompositions by Givens transformations. *Linear Algebra and its Appl.*, 10:189–197, 1975.

[109] A.V. Oppenheim and R.W. Schafer. *Digital Signal Processing*. Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1975.

[110] J.H. Wilkinson. *The Algebraic Eigenvalue Problem*. Oxford University press, Oxford, 1965.

[111] G.H. Golub and C.F. Van Loan. *Matrix Computations*. North Oxford Academic, Oxford, second edition, 1983.

[112] G. Strang. *Linear Algebra and its Applications*. Harcourt Brace Jovanovich, San Diego, third edition, 1986.

[113] G.H. Golub and W. Kahan. Calculating the singular values and pseudo-invers of a matrix. *SIAM J. Num. Anal.*, 2:205–224, 1965.

[114] T.A.C.M. Claasen, W.F.G. Mecklenbräuker, and J.B.H. Peek. Some remarks on the classification of limit cycles in digital filters. *Philips Research Reports*, 8(4):297–305, August 1973.

[115] T.A.C.M. Claasen, W.F.G. Mecklenbräuker, and J.B.H. Peek. Frequency domain criteria for the absence of zero-input limit cycles in nonlinear discrete-time systems, with applications to digital filters. *IEEE Trans. on Circuits and Systems*, 22(3):232–239, March 1975.

[116] A.A.J. de Lange, A. van der Hoeven, J. Bu, and E.F. Deprettere. A floating-point pipeline CMOS CORDIC processor: Algorithm, automatic design, layout and performance. Technical report, Delft University of Technology, 1988. (See also Proceedings ISCAS'88).

[117] J.M. Delosme. CORDIC algorithms: Theory and extensions. In *Proceedings SPIE Advanced Algorithms and Architectures for Signal Processing*, pages 131–145, 1989.

[118] H.W. van Dijk, G.J. Hekstra, and E.F. Deprettere. Scalable parallel processor arry for jacobi-type matrix computations. *Integration, the VLSI Journal*, 1995.

[119] J. Götze. On the parallel implementation of Jacobi and Kogbetliantz algorithms. *SIAM J. Sci. Comput.*, 15(6):1331–1348, November 1994.

[120] E.F. Deprettere, G.J. Hekstra, and R. Heusdens. Fast VLSI overlapped transform kernel. In *Proceedings IEEE Int. Symp. on Circuits and Systems*, 1995.

[121] P. Yip and K.R. Rao. A fast computational algorithm for the discrete sine transform. *IEEE Trans. on Communications*, 28(2):304–307, February 1980.

[122] Z. Wang. A fast algorithm for the discrete sine transform implemented by the fast cosine transform. *IEEE Trans. on ASSP*, 30(5):803–816, October 1982.

[123] Z. Wang. Fast algorithms for the discrete W transform and for the discrete Fourier transform. *IEEE Trans. on ASSP*, 32(4):803–816, August 1984.

[124] B.G. Lee. A new algorithm to compute the discrete cosine transform. *IEEE Trans. on ASSP*, 32(6):1243–1245, December 1984.

[125] E.G. Kogbetliantz. Solution of linear equations by diagonalization of coefficients matrix. *Quarterly of Applied Mathematics*, 13(2):123–132, 1955.

[126] V. Hari and K. Veselić. On Jacobi methods for singular value decompositions. *SIAM J. Sci. Stat. Comput.*, 8(5):741–754, September 1987.

[127] W. Rath. Fast Givens rotations for orthogonal similarity transformations. *Numerische Mathematik*, 40:47–56, 1982.

[128] J. Götze, S. Paul, and M. Sauer. An efficient Jacobi-like algorithm for parallel eigenvalue computation. *IEEE Trans. on Computers*, 45(9):1058–1065, September 1993.

[129] M. Monari. Ricorsiva realizzazione approssimata della trasformazione di immagine con ortonormali rotazioni. Master's thesis, Bologna University, Bologna, Italy, july 1996.

[130] G.J. Hekstra, E.F. Deprettere, M. Monari, and R. Heusdens. Recursive approximate realization of image transforms with orthonormal rotations. In *Proceedings 3rd Int. Workshop in Image and Signal Processing Advances in Computer Intellegence*, Manchester, November 1996.

[131] G.J. Hekstra, E.F. Deprettere, R. Heusdens, and Z. Zeng. Efficient orthogonal realization of image transforms. *Advanced Signal Processing: Algorithms, Architectures and Implementations*, (VII), August 1996.

[132] M. Breeuwer, R. Klein Gunnewiek, W. Wiertsema, A. Gruijthuijsen, and P. Lippens. The feasibility of an IC-implementation of an overlapped transform-coding method for data compression of medical images: Summary of results. Technical Report 6900, Philips Research Laboratories, Eindhoven, May 1996. Company confidential till january 1998.

[133] R. Klein Gunnewiek, W. Wiertsema, A. Gruijthuijsen, and M. Breeuwer. The feasibility of an IC-implementation of an overlapped transform-coding method for data compression of medical images: Part II - The overlapped transform. Technical Report 6902, Philips Research Laboratories, Eindhoven, May 1996. Company confidential till january 1998.

[134] *TM-1 Preliminary Data Book*, February 1996. Philips Semiconductors.

[135] R. Klein Gunnewiek. Mapping a medical image compression algorithm on a TriMedia processor. Technical Report 6922, Philips Research Laboratories, Eindhoven, June 1996. Company confidential till january 1998.

[136] G. Helmberg. *Introduction to Spectral Theory in Hilbert Space*. North-Holland publishing company, Amsterdam, 1969.

[137] N. Young. *An introduction to Hilbert Space*. Cambridge Mathematical Textbooks, Cambridge, 1988.

[138] J.L. Doob. *Stochastic Processes.* Wiley Series in Statistics. John Wiley and Sons, Inc., New York, 1953.

[139] M. Loève. *Probability Theory.* The University Series in Higher Mathematics. D. van Nostrand Company, Inc., Princeton, 1955.

[140] M.S. Pinsker. *Information and Information Stability of Random Variables.* Holden-Day, San Francisco, 1964. Translation into English by A. Feinstein.

[141] T.M. Cover and J.A. Thomas. *Elements of Information Theory.* Wiley Series in Telecommunication. John Wiley and Sons, Inc., New York, 1991.

# *Appendix A*

# *Definitions and basic theory*

**Contents**



## A.1 Introduction

In this appendix we give some definitions and preliminaries. Its main purpose is to provide sufficient mathematical background for the understanding of the theorems stated and proven in this thesis.

Linear operators play an important role throughout, and it thus seems proper to start with a brief review of the definitions and results that are relevant to this thesis. This is done in Section A.2. Complicated or technical proofs that are not necessary for understanding the main line of thought have been omitted. For a more extensive but simple introduction to this theory the reader is referred to [136, 137].

Since the main subject of this thesis deals with data compression, we will unavoidably need the theory of random processes. To prepare for this, we provide some background in Section A.3. This section is not intended to serve as a detailed exposition on the fundamentals of random processes and information theory, since it is assumed that the reader is familiar with the rudiments of this subject. A more detailed description of random processes can be found in [138, 139].

The symbols $\mathbb{N}, \mathbb{Z}, \mathbb{R}$, and $\mathbb{C}$ will denote the set of natural, integer, rational, real and complex numbers, respectively. Real elements may be considered complex by a natural embedding of real numbers in the field of complex numbers. The symbols $a, b, \ldots, f, g, \ldots$ (lower case italics) denote arbitrary set elements, with the exception of $j, k, l, m, n$ which are used for integers, in particular, for indices. The *zero element* is denoted by $o$. The symbols $\alpha, \beta, \ldots$ (lower case greek letters) denote arbitrary numbers in $\mathbb{C}$. The Kronecker delta is denoted by $\delta_{ko}$. If $\alpha$ is a complex scalar, then $\overline{\alpha}$ denotes the complex conjugate of $\alpha$.

## A.2 Linear operators

In this section we consider an arbitrary Hilbert space $\mathcal{H}$.

**Definition A.2.1 (linear manifold and subspace)** *A non-empty subset $\mathcal{D}$ of $\mathcal{H}$ is called a linear manifold if for all $f, g \in \mathcal{D}$ and all $\lambda \in \mathbb{C}$, $f+g \in \mathcal{D}$ and $\lambda f \in \mathcal{D}$. A closed linear manifold is called a subspace.*

**Definition A.2.2 (operator)** *A mapping of a linear manifold $\mathcal{D} \subset \mathcal{H}$ onto $\mathcal{H}$ is called an operator in $\mathcal{H}$. A mapping of $\mathcal{H}$ onto $\mathcal{H}$ (i.e. $\mathcal{D} = \mathcal{H}$) is called an operator on $\mathcal{H}$.*

In the following we shall assume that the subset $\mathcal{D} \subset \mathcal{H}$ on which a mapping is defined (also called the *domain*) is a linear manifold. Linear maps of a linear space $\mathbb{C}^m$ onto some other linear space $\mathbb{C}^n$ are represented by matrices. If $f \in \mathbb{C}^m$, and $A = (\alpha_{k,l})_{k=1,\ldots,n, l=1,\ldots,m}$, then $g = Af \in \mathbb{C}^n$. We shall write $A \in \mathbb{C}^{n \times m}$.

**Definition A.2.3 (linear operator)** *A (not necessarily continuous) mapping $A : f \to Af$ of a linear manifold $\mathcal{M} \subset \mathcal{H}$ onto a Hilbert space $\mathcal{H}'$ is called linear if for all $f, g \in \mathcal{M}$ and all $\lambda \in \mathbb{C}$*

$$A(f+g) = Af + Ag,$$

$$A(\lambda f) = \lambda Af.$$

*A mapping $A$ of $\mathcal{M}$ onto $\mathcal{H}'$ is called isometric (or an isometry) if for all $f \in \mathcal{M}$ and all $g \in \mathcal{M}$*

$$\langle Af, Ag \rangle = \langle f, g \rangle.$$

*A linear and isometric mapping of $\mathcal{H}$ onto $\mathcal{H}'$ is called an isomorphism of $\mathcal{H}$ onto $\mathcal{H}'$. An isomorphism of $\mathcal{H}$ onto itself is called an automorphism.*

**Definition A.2.4 (bounded linear operator and matrix norm)** *A linear operator A, with domain $\mathcal{D} \subset \mathcal{H}$, onto a Hilbert space $\mathcal{H}'$ is said to be bounded if there exists a real number $\gamma \geq 0$ such that*

$$\|Af\| \leq \gamma \|f\| \quad \text{for all } f \in \mathcal{D}.$$

*If A is bounded and $\mathcal{D} \neq \emptyset$, then the non-negative number*

$$\|A\| = \sup_{f \in \mathcal{D}, f \neq o} \frac{\|Af\|}{\|f\|}, \tag{A.1}$$

*is called the norm of A.*

If $A$ is bounded, then (A.1) implies

$$\|Af\| \leq \|A\| \cdot \|f\| \quad \text{for all } f \in \mathcal{D}.$$

Note that for a bounded linear mapping $A$ on a domain $\mathcal{D}$, $\mathcal{D} \neq \emptyset$, (A.1) is equivalent to the two following equations [136]:

$$\|A\| = \sup_{f \in \mathcal{D}, \|f\| = 1} \|Af\|,$$

$$\|A\| = \sup_{f \in \mathcal{D}, \|f\| \leq 1} \|Af\|.$$

**Definition A.2.5 (adjoint operator)** *Let A be a linear operator on $\mathcal{H}$. The linear operator $A^*$ on $\mathcal{H}$ satisfying*

$$\langle Af, g \rangle = \langle f, A^* g \rangle \quad \text{for all } f, g \in \mathcal{H},$$

*is called the adjoint of A.*

It can be shown [136] that if $A$ is a linear operator on $\mathcal{H}$, then $A$ and $A^*$ are bounded.

**Definition A.2.6 (inverse operator)** *If the linear operator A on $\mathcal{H}$ is one-to-one on $\mathcal{H}$, then the linear operator $A^{-1}$ on $\mathcal{H}$ defined by*

$$A^{-1}(Af) = f \quad \text{for all } f \in \mathcal{H},$$

*is called the inverse of A.*

**Definition A.2.7 (self-adjoint, unitary and normal)** *A linear operator $A$ on $\mathcal{H}$ is called self-adjoint if $A^* = A$. A bounded linear operator $A$ on $\mathcal{H}$ is called* unitary *if $A^* = A^{-1}$. $A$ is called* normal *if $A^*A = AA^*$.*

Obviously, self-adjoint bounded linear operators as well as unitary ones are normal.

It is well known that every subspace $\mathcal{M} \subset \mathcal{H}$ gives rise to a bounded linear operator $P = P_{\mathcal{M}}$ on $\mathcal{H}$ which maps every vector $f \in \mathcal{H}$ onto its projection upon $\mathcal{M}$. This bounded linear operator $P$ is called the *projection operator* corresponding to $\mathcal{M}$ or, more briefly, the *projection* upon $\mathcal{M}$. If $P$ is the projection upon a subspace $\mathcal{M}$, then

$$\mathcal{M} = \{f : Pf = f\} = \{f : \|Pf\| = \|f\|\} = \{Pg : g \in \mathcal{H}\}.$$

A bounded linear operator $P$ on $\mathcal{H}$ is a projection if $P = P^2$. It is an orthogonal projection if $P = P^*$.

**Remark A.2.1** *It seems reasonable to denote the set $\{Pg : g \in \mathcal{H}\}$ by $P\mathcal{H}$. In general, if $A$ is any linear operator, then for any subset $\mathcal{U} \subset \mathcal{H}$ we shall write*

$$A\mathcal{U} = \{Af : f \in \mathcal{U}\}.$$

**Definition A.2.8 (range)** *If $A$ is a linear operator on $\mathcal{H}$, then the set $A\mathcal{H}$ is called the range or image of $A$ (notation:* $\operatorname{ran}(A)$*).*

According to the foregoing the range of a projection coincides with the corresponding subspace.

**Definition A.2.9 (kernel)** *If $A$ is a linear operator on $\mathcal{H}$, then*

$$\ker(A) = \{f \in \mathcal{H} : Af = 0\},$$

*is called the null-space or kernel of $A$.*

**Theorem A.2.1** *If $A$ is a bounded linear operator on $\mathcal{H}$, then* $\ker(A)$ *is a subspace.*

**Proof:** Suppose $g_1 \in \ker(A)$ and $g_2 \in \ker(A)$. Since $A(\alpha_1 g_1 + \alpha_2 g_2) = \alpha_1 A g_1 + \alpha_2 A g_2 = 0, \alpha_1 \in \mathbb{C}$ and $\alpha_2 \in \mathbb{C}$ we conclude that $\alpha_1 g_1 + \alpha_2 g_2 \in \ker(A)$. Thus $\ker(A)$ is a linear manifold. Now let $g \in \overline{\ker(A)}$ be given and let $g = \lim_{n\to\infty} g_n$, $g_n \in \ker(A)$ for all $n \geq 1$. Then, since $A$ is continuous, we have

$$Ag = A\left(\lim_{n\to\infty} g_n\right) = \lim_{n\to\infty} Ag_n = 0,$$

and therefore $g \in \ker(A)$. Thus, $\ker(A)$ is closed and a subspace. □

**Theorem A.2.2** *Let $A$ and $A^*$ be bounded linear operators on $\mathcal{H}$. Then*

$$\operatorname{ran}(A^*)^{\perp} = \ker(A),$$

*and as a consequence*

$$\mathcal{H} = \ker(A) \oplus \overline{\operatorname{ran}(A^*)}.$$

**Proof:** Since $A$ is bounded we conclude from Theorem A.2.1 that $\ker(A)$ is a subspace. Suppose $g \in \mathcal{H}$ is given. Then from

$$\langle f, A^*g \rangle = \langle Af, g \rangle = 0 \quad \text{for all } f \in \operatorname{ran}(A^*)^{\perp},$$

it follows that $\operatorname{ran}(A^*)^{\perp} = \{f \in \mathcal{H} : Af = 0\} = \ker(A)$. □

## A.3 Random processes

The theory of random processes is a branch of probability theory and probability theory is a special area of the branch of mathematics known as measure theory. Probability theory and measure theory both concentrate on functions that assign real numbers to certain subsets of an arbitrary set according to certain rules. These set functions can be viewed as measures of the sizes or weights of the sets. Both probability theory and measure theory consider only non-negative real-valued set functions. The value assigned by the function to a set is called the probability or the measure of the set, respectively.

The simplest situation where probability can be considered involves a random experiment with a finite number of outcomes. Let $\Omega = \{\omega_n\}_{n=1}^{N}$ denote the set of all outcomes. To each outcome $\omega_n$ we can assign a probability. Every subset $A$ of $\Omega$ in this case also has a well-defined probability which is equal to the sum of the probabilities of the outcomes contained in $A$. If the number of outcomes in $\Omega$ is countably infinite, the situation is quite similar to the finite case. However, if $\Omega$ contains an uncountable number of points – for example, if we spin a fair wheel and the outcome is known to be equally likely to be any number between 0 and 1 – then the probability that any particular point will occur is zero because there is an uncountable infinity of possible points, none more likely to occur than any others. Hence knowing only that the probability of each and every point is zero, we would be hard pressed to make any meaningful inferences about the probability of the outcome being between $\frac{1}{2}$ and $\frac{3}{4}$.

The difficulty in this example shows that, in general, it is necessary to consider probabilities to be defined on subsets of $\Omega$, rather than on points of $\Omega$. A subset $A$ on which a probability is defined is called an *event.*

Let $\Omega$ be an arbitrary set and let $\mathcal{A}$ be a $\sigma$-algebra[1] denoting the class of all events. The pair $(\Omega, \mathcal{A})$ is called a *measurable space.* The name "measurable space" reflects the fact that we can assign a non-unique measure to such a space. We can make this notion more precise with the following definition.

**Definition A.3.1 (measure)** *A set function $\mu$ defined on the sets of a $\sigma$-algebra $\mathcal{A}$ is called a measure if it satisfies the following properties:*

1. $\mu(A) \geq 0$ *for* $A \in \mathcal{A}$,
2. $\mu(\bigcup_{n=1}^{N} A_n) = \sum_{n=1}^{N} \mu(A_n)$ *whenever* $A_1, \ldots, A_N$ *are finitely or denumerable infinitely many disjunct sets of $\mathcal{A}$ (completely additive),*
3. *Whenever* $(A_n)_n$ *is a sequence of sets in* $\mathcal{A}$ *such that* $A_1 \supset A_2 \supset \cdots \supset A_n$, $\bigcap_{n=1}^{\infty} A_n = \emptyset$ *then* $\lim_{n\to\infty} \mu(A_n) = 0$.

If we add the constraint $\mu(\Omega) = 1$, the measure becomes a *probability measure* and $(\Omega, \mathcal{A}, \mu)$ becomes a *probability measure space* or *probability space.*

**Definition A.3.2 (probability space)** *A probability space is a triplet $(\Omega, \mathcal{A}, \mathcal{P})$ where $\Omega$ is a non-empty set, $\mathcal{A}$ is a $\sigma$-algebra of subsets of $\Omega$ and $\mathcal{P}$ is a probability measure defined on $(\Omega, \mathcal{A})$. The set $\Omega$ is called the sample space.*

In information theory literature, what we call a probability space is sometimes called an *ensemble.*

A random variable is a mapping from some sample space into a set. In this thesis we only consider real-value random variables, i.e., the first space is the sample space portion of a probability space and the second space is (a subset of) the real line. A random variable is perhaps best thought of as a measurement on a probability space, i.e., for each sample point $\omega$ the random variable produces some value, denoted functionally as $f(\omega)$. $\omega$ can be viewed as the result of some experiment and $f(\omega)$ as the result of a measurement made on the experiment.

We now provide a precise mathematical definition of a random variable and then make some comments on the reasoning behind the definition.

[1] A $\sigma$-algebra is a collection of subsets of an arbitrary set that is closed under complementation and countable union.

Given the real line $\mathbb{R}$, the *Borel $\sigma$-algebra* is defined as the $\sigma$-algebra generated by all open intervals of the form $(a, b)$ on $\mathbb{R}$. The members of the Borel algebra are called *Borel sets*. We shall denote the Borel $\sigma$-algebra of the real line by $\mathcal{B}$.

**Definition A.3.3 (measurable mapping)** *Let $(\Omega_1, \mathcal{A}_1)$ and $(\Omega_2, \mathcal{A}_2)$ be two measurable spaces and let $f$ be a function with domain $\Omega_1$ and range in $\Omega_2$. The function $f$ is said to be a measurable function or measurable mapping of $(\Omega_1, \mathcal{A}_1)$ into $(\Omega_2, \mathcal{A}_2)$ if for every set $A$ in $\Omega_2$, the inverse image of $A$ under $f$,*

$$f^{-1}(A) = \{\omega : f(\omega) \in A\},$$

*is in $\Omega_1$.*

**Definition A.3.4 (random variable)** *Let $(\Omega, \mathcal{A}, \mathcal{P})$ be a probability space. A (real) random variable is a measurable mapping $(\Omega, \mathcal{A}) \mapsto (\mathbb{R}, \mathcal{B})$.*

Given a probability space $(\Omega, \mathcal{A}, \mathcal{P})$ and a random variable $f : \Omega \mapsto \mathbb{R}$, we need a precise definition of the probability $P(f(\omega) \in B)$, or $P(B)$ for short, for any $B \in \mathcal{B}$, the Borel $\sigma$-algebra or event space of the real line. Since the probability measure $\mathcal{P}$ is actually on the original measurable space $(\Omega, \mathcal{A})$ and since $f$ assumes a value in $B$ if and only if $\omega \in \Omega$ is chosen so that $f(\omega) \in B$, the desired probability is clearly

$$P(B) = \mathcal{P}(\{\omega : f(\omega) \in B\}) = \mathcal{P}(f^{-1}(B)).$$

In other words, the probability that a random variable $f$ takes on a value in a Borel set $B$ is the probability of the inverse image of the Borel set $B$ under the random variable $f$.

It is clear that the foregoing natural definition of the probability of an output event of a random variable only makes sense if the probability $\mathcal{P}(f^{-1}(B))$ makes sense, i.e., if the subset $f^{-1}(B)$ of $\Omega$ corresponding to the output event $B$ is itself an event, in this case an input event or member of the event space $\mathcal{A}$ of the original sample space. For this reason a random variable is defined as a measurable function, rather than an arbitrary function $f : \Omega \mapsto \mathbb{R}$.

Mathematically, a random variable is a measurable function defined on the sample space of a probability space $(\Omega, \mathcal{A}, \mathcal{P})$. The amount of randomness is described by $P$, i.e., by a probability measure on an event space of the real line. When doing computations involving random variables, it is usually simpler to

concentrate on the probability space $(\Gamma, \mathcal{B}, P)$ rather than on the original underlying probability space $(\Omega, \mathcal{A}, \mathcal{P})$, where $\Gamma$ is the set of all possible values the random variable can take on. The set $\Gamma$ is often called the alphabet of the random variable. For calculations involving the output values of a random variable, then the probability space $(\Gamma, \mathcal{B}, P)$ is more direct to deal with.

So far we have used lower case letters to denote random variables. In the following, however, we also often use upper case letters, like $U$, $V$, $X$ and $Y$, for random variables.

**Definition A.3.5 (distribution function)** *Let $X_1, \ldots, X_n$ be random variables defined on a common probability space $(\Omega, \mathcal{A}, \mathcal{P})$ and let $X = (X_1, \ldots, X_n)$. Also, let $x_1, \ldots, x_n \in \mathbb{R}$ and $x = (x_1, \ldots, x_n)$. The function $F$, defined by*

$$F(x) = \mathcal{P}(\{\omega : X_j(\omega) \leq x_j, \ j = 1, \ldots, n\}),$$

*is called the (joint) distribution function of $X$.*

For each $x \in \mathbb{R}^n$, $F(x)$ gives the probability that the sample values $X_j(\omega)$ are less than or equal to $x_j$.

**Definition A.3.6 (independence)** *Let $X_1, \ldots, X_n$ be random variables defined on a common probability space $(\Omega, \mathcal{A}, \mathcal{P})$ and let $F_i = \mathcal{P}(\{\omega : X_i(\omega) \leq x_i\})$ denote the distribution function of $X_i$. Then $X_1, \ldots, X_n$ are called independent if and only if*

$$F(x) = \prod_{i=1}^{n} F_i(x_i).$$

Where $F$ is absolutely continuous there exists a non-negative function $p$, called a (joint) *probability density function*, such that

$$P(B) = \int_B p(x)dx.$$

Where $X_1, \ldots, X_n$ are discrete random variables, i.e. take at most countably many values, $F$ is a step function and there exists a non-negative function $p$, called a *probability mass function*, such that

$$P(B) = \sum_{x \in B} p(x).$$

In the most general case the distribution function is a composite of a step function, an absolutely continuous part and a continuous function whose points of increase

all belong to a Lebesgue-null set [139, p. 178]. We have limited ourselves to cases where $F$ consists solely of a step function and an absolutely continuous part. Although these random variables cannot be described by either a probability density function or a probability mass function we can describe them by a combination of a discrete distribution and a continuous distribution. To avoid unnecessary writing, however, we will use the description in terms of a probability density function only and assume that these functions may contain impulse functions.

**Definition A.3.7 (conditional probability density)** *If* $x_m, \ldots, x_n$ *are such that* $p(x_m, \ldots, x_n) > 0$ *and*

$$F(x_1, \ldots, x_{m-1} | x_m, \ldots, x_n) = \mathcal{P}(\{\omega : X_i(\omega) \leq x_i \ | X_j(\omega) = x_j, \ i = 1, \ldots, m-1, \ j = m, \ldots, n\}),$$

*then*

$$p(x_1, \ldots, x_{m-1} | x_m, \ldots, x_n) = \frac{p(x_1, \ldots, x_n)}{p(x_m, \ldots, x_n)},$$

*is called the conditional probability density.*

A random variable which turns out to be very useful is the *mutual information* $i(x_i; x_j)$ between $X_i(\omega)$ having a value $x_i$ and $X_j(\omega)$ having a value $x_j$. The mutual information is defined as the information provided about $X_i(\omega) = x_i$ by the occurrence of $X_j(\omega) = x_j$.

**Definition A.3.8 (mutual information)** *Let* $X_i$ *and* $X_j$ *be real random variables defined on a common probability space* $(\Omega, \mathcal{A}, \mathcal{P})$. *The mutual information between* $X_i(\omega) = x_i$ *and* $X_j(\omega) = x_j$ *is defined as*

$$i(x_i; x_j) = \log \frac{p(x_i, x_j)}{p(x_i) p(x_j)} \tag{A.2}$$

Similarly, we can define the *conditional mutual information* between $X_i(\omega) = x_i$ and $X_j(\omega) = x_j$ given $X_k(\omega) = x_k$.

**Definition A.3.9 (conditional mutual information)** *Let* $X_i$, $X_j$ *and* $X_k$ *be real random variables defined on a common probability space* $(\Omega, \mathcal{A}, \mathcal{P})$. *The conditional mutual information between* $X_i(\omega) = x_i$ *and* $X_j(\omega) = x_j$, *given* $X_k(\omega) = x_k$, *is defined as*

$$i(x_i; x_j | x_k) = \log \frac{p(x_i, x_j | x_k)}{p(x_i | x_k) p(x_j | x_k)}. \tag{A.3}$$

The base of the logarithm determines the numerical scale used to measure information. The most common bases are 2 and $e$, for which the numerical values of (A.2) and (A.3) are called the number of *bits* and *nats*, respectively. It should be noted that both (A.2) and (A.3) are symmetric in the sense that $i(x_i; x_j) = i(x_j; x_i)$ and $i(x_i; x_j|x_k) = i(x_j; x_i|x_k)$.

**Definition A.3.10 (expectation)** *The expectation (mean, average, first moment) of a random variable $X$ is given by*

$$EX = \int_{-\infty}^{\infty} xp(x)dx.$$

With this, the average value of mutual information, denoted by $I(x_i; x_j)$, is given by

$$I(x_i; x_j) = \int\int p(x_i, x_j) \log \frac{p(x_i, x_j)}{p(x_i)p(x_j)} dx_i dx_j.$$

The *average mutual information* $I(x_i; x_j)$ represents the mean amount of information that knowledge of the value assumed by $X_i$ supplies about the value assumed by $X_j$, or vice versa. It is easy to show (see also [46, p. 24]) that the average mutual information satisfies $I(x_i; x_j) \geq 0$ with equality if and only if $x_i$ and $x_j$ are independent. In [140] it is shown that the average mutual information can also be well defined where $X_j = f(X_i)$, which implies that the joint probability density $p(x_i, x_j)$ is an impulse function concentrated on the set $\{(x_i, f(x_i)) : x_i \in \mathbb{R}\}$.

The *average conditional mutual information*, denoted by $I(x_i; x_j|x_k)$, is given by

$$I(x_i; x_j|x_k) = \int\int\int p(x_i, x_j, x_k) \log \frac{p(x_i, x_j|x_k)}{p(x_i|x_k)p(x_j|x_k)} dx_i dx_j dx_k. \quad \text{(A.4)}$$

The average conditional mutual information satisfies $I(x_i; x_j|x_k) \geq 0$ with equality if and only if for each $X_k(\omega) = x_k$, $X_i$ and $X_j$ are independent. The situation in which $I(x_i; x_j|x_k) = 0$ is very important and can be visualized as a pair of cascaded channels. Let $X_i$ be the input of the first channel, let $X_k$ be both the output of the first channel and the input of the second channel and let $X_j$ be the output of the second channel. The situation $I(x_i; x_j|x_k) = 0$ means that $p(x_i|x_j x_k) = p(x_i|x_k)$ for all $x_i, x_j, x_k$ for which $p(x_j, x_k) > 0$, so that the output of the second channel depends only on the input of the second channel. Hence, the condition $I(x_i; x_j|x_k) = 0$ states that there is no "hidden" channel

passing information about $X_i$ to $X_j$. In this case the random variables $X_i, X_k, X_j$ are said to *form a Markov chain* in that order (denoted by $X_i \to X_k \to X_j$). We will make this notion more explicit in the following definition.

**Definition A.3.11 (Markov chain)** *Let $X_1, \ldots, X_n$ be real random variables defined on a common probability space $(\Omega, \mathcal{A}, \mathcal{P})$. If the conditional distribution of $X_n$ depends only on $X_{n-1}$ and is conditionally independent of $X_i$, $i \leq n-2$, then the random variables $X_1, \ldots, X_n$ are said to form a Markov chain in that order (denoted by $X_1 \to X_2 \to \cdots \to X_n$). Specifically, $X_1, \ldots, X_n$ form a Markov chain $X_1 \to \cdots \to X_n$ if the joint probability density function can be expressed as*

$$p(x_1, \ldots, x_n) = p(x_1)p(x_2|x_1) \cdots p(x_n|x_{n-1}).$$

Note that $X_i \to X_j \to X_l$ implies $X_l \to X_j \to X_i$. The following two theorems are very important.

**Theorem A.3.1 (chain rule for information)** *Let $X_1, \ldots, X_n$ be jointly distributed random variables defined on a common probability space $(\Omega, \mathcal{A}, \mathcal{P})$. Then*

$$I(x_1, \ldots, x_{n-1}; x_n) = \sum_{j=1}^{n-1} I(x_j; x_n|x_{j-1}, \ldots, x_1).$$

**Proof:** See [141, p. 22] □

**Theorem A.3.2 (data processing inequality)** *Let $X_1, \ldots, X_n$ be jointly distributed random variables defined on a common probability space $(\Omega, \mathcal{A}, \mathcal{P})$. If $X_i \to X_j \to X_l$, then $I(x_i; x_j) = I(x_i; x_l) + I(x_i; x_j|x_l) \geq I(x_i; x_l)$.*

**Proof:** See [141, p. 32] □

The result of Theorem A.3.2 has the important interpretation that any processing on $X_j$, whether it is deterministic or stochastic, can never increase the average mutual information.

**Definition A.3.12 (random process)** *A random process $\{X_t, t \in T\}$ is a family of random variables, indexed by a real parameter $t$ and defined on a common probability space $(\Omega, \mathcal{A}, \mathcal{P})$. By definition, for each $t$, $X_t$ is a $\mathcal{P}$-measurable function. For each $\omega \in \Omega$, $X_t(\omega)$ defines a function of $t$ and is called a sample function or a realization of the process.*

**Remark A.3.1** *In time-series analysis the index (or parameter) set $T$ is a set of time points, very often $\{0, \pm 1, \pm 2, \ldots\}$, $[0, \infty)$ or $(-\infty, \infty)$, but random processes in which $T$ is not a subset of $\mathbb{R}$ are also important. Here, however, the index set $T$ will always be a subset of $\mathbb{R}$, in particular $T = \mathbb{Z}$.*

**Definition A.3.13 (discrete source)** *A discrete source is a random process $\{X_n, n \in T\}$ defined on a probability space $(\Omega, \mathcal{A}, \mathcal{P})$ where $T \subset \mathbb{Z}$.*

Thus, a discrete source selects a point $\omega \in \Omega$ and then produces the sample sequence $\{X_n(\omega), n \in T\}$.

**Definition A.3.14 (second-order random process)** *A second-order random process is a one-parameter family of random variables satisfying $E|X_t|^2 < \infty$ for all $t \in T$.*

Consider a probability space $(\Omega, \mathcal{A}, \mathcal{P})$ and the collection $\mathcal{H}_X$ of all second-order random variables $X$ defined on $\Omega$. The space $\mathcal{H}_X$ becomes a Hilbert space if we define the norm $\|X\|$ for any $X \in \mathcal{H}_X$ and inner product $\langle X, Y \rangle$ for any $X, Y \in \mathcal{H}_X$ by

$$\|X\| = \left(E|X|^2\right)^{\frac{1}{2}},$$

$$\langle X, Y \rangle = EX\overline{Y}.$$

**Definition A.3.15 (covariance function)** *If $\{X_t, t \in T\}$ is a second-order random process, then the covariance function $R$ of $\{X_t\}$ is defined by*

$$R(t, s) = E(X_t - EX_t)\overline{(X_s - EX_s)}, \quad t, s \in T,$$

*and the correlation function $r$ of $\{X_t\}$ is defined by*

$$r(t, s) = EX_t\overline{X_s}, \quad t, s \in T.$$

**Definition A.3.16 (uncorrelated)** *A second-order random process $\{X_t, t \in T\}$ is said to be uncorrelated if*

$$E(X_t - EX_t)\overline{(X_s - EX_s)} = 0 \text{ for all } t, s \in T, \ t \neq s.$$

*The second-order random processes $\{X_t, t \in T\}$, $\{Y_t, t \in T\}$ are said to be mutually uncorrelated if*

$$E(X_t - EX_t)\overline{(Y_s - EY_s)} = 0 \text{ for all } t, s \in T.$$

As far as second-order properties are concerned, we are primarily interested in linear operations on processes. Since the output of a linear operation on $\{X_t, t \in T\}$ is equal to the sum of the outputs of the same operation on $X_t - EX_t$ and $EX_t$ separately, we can always assume a zero mean with little loss of generality. In the following we always assume that the mean is zero, so that the correlation function and the covariance function are the same.

**Definition A.3.17 (wide-sense stationary process)** *A process* $\{X_t, t \in T\}$ *is said to be a wide-sense stationary process (w.s.s. process) if*

1. $E|X_t|^2 < \infty$ *for all* $t \in T$,
2. $EX_t = \mu$ *is a constant for all* $t \in T$,
3. $EX_t\overline{X_s} = R(t-s)$ *for all* $s, t \in T$.

A process $\{X_t, t \in T\}$ is said to be *strictly stationary* if the distributions remain the same with the passage of time, i.e., the multivariate distributions of the random variables $X_{t_1+h}, \ldots, X_{t_n+h}$ are independent of $h$. If a real Gaussian process is stationary in the wide-sense then the process is strictly stationary because the parameters that determine a Gaussian distribution are the mean and the covariances. A strictly stationary process is stationary in the wide-sense if it is a second-order process.

**Theorem A.3.3** *The covariance function* $R$ *of a w.s.s. process* $\{X_n, n \in \mathbb{Z}\}$ *is non-negative definite, i.e.,*

$$\sum_{m,n=1}^{N} R(m-n)\alpha_m\overline{\alpha_n} \geq 0,$$

*for every set of complex numbers* $\alpha_1, \ldots, \alpha_N$. *Conversely, any function* $R$ *satisfying this condition is the covariance function of a w.s.s. process.*

**Proof:** See also [138, p. 473] □

The following theorem describes the non-negative definite functions as Fourier transforms and is usually referred to as Bochner's theorem.

**Theorem A.3.4 (Bochner)** *A function* $R(h)$, $h \in \mathbb{Z}$, *is the covariance function of a w.s.s. process* $\{X_n, n \in \mathbb{Z}\}$ *if and only if it is of the form*

$$R(h) = \frac{1}{2\pi}\int_{-\pi}^{\pi} e^{i\xi h} dF(\xi),$$

*where $F$ (defined for $|\xi| \leq \pi$) is bounded and monotone non-decreasing with $F(\pi) - F(-\pi) = 2\pi R(0)$.*

**Proof:** See also [138, p. 474]. □

**Remark A.3.2** *The function $F$ is called a spectral-distribution function. Since $F$ is bounded and monotone non-decreasing we have $F \in L^2(-\pi, \pi)$ so that there exists a non-negative function $S$, such that*

$$F(\lambda) = \int_{-\pi}^{\lambda} S(\xi) d\xi.$$

*Naturally, $S$ is called the spectral-density function.*

# *Appendix B*

# *Proof of Theorem 2.4.1*

In this appendix we prove Theorem 2.4.1. Before this, however, we will first derive some results which we shall use in the proof.

**Lemma B.1.1** *Let $X$, $Y$ and $Z$ be jointly distributed random variables defined on a common probability space $(\Omega, \mathcal{A}, \mathcal{P})$. $I(x; z|y) = 0$ if and only if $p(x|yz) = p(x|y)$ for all $x$, $y$, $z$ for which $p(y, z) > 0$.*

**Proof:** Obviously, from (A.4), we have $I(x; z|y) = 0$ if and only if $p(x, z|y) = p(x|y)p(z|y)$. Hence, $I(x; z|y) = 0$ if and only if $p(x|yz) = p(x|y)$ for all $x$, $y$, $z$ for which $p(y, z) > 0$, as required. □

Hence, Lemma B.1.1 shows that we have $I(x; z|y) = 0$ if and only if $X \rightarrow Y \rightarrow Z$.

**Theorem B.1.5** *Let $X$, $Y$ and $Z$ be jointly distributed random variables defined on a common probability space $(\Omega, \mathcal{A}, \mathcal{P})$ and let $f$ be a measurable function.*

1. *If $Z = f(Y)$, then $I(x; y|z) = 0$ if and only if $y \equiv f(y) \pmod{X}$ for all $y \in \Gamma_y$.*
2. *If $Y = f(X)$, then $I(y; z|x) = 0$ if and only if $x \equiv f(x) \pmod{Z}$ for all $x \in \Gamma_x$.*

**Proof:**

1. From Theorem A.3.2 we have $I(x; y) = I(x; z) + I(x; y|z) \geq I(x; z)$ with equality if and only if $I(x; y|z) = 0$ and thus if and only if $p(x|yz) =$

$p(x|z)$ for all $x, y, z$ for which $p(y, z) > 0$ by Lemma B.1.1. Moreover, since $X \to Y \to Z$ we also have $p(x|yz) = p(x|y)$ for all $x, y, z$ for which $p(y, z) > 0$. Hence, $p(x|y) = p(x|z)$ for all $x, y, z$ for which $p(y, z) > 0$. This condition is satisfied if and only if $y \equiv f(y) \pmod{X}$ for all $y \in \Gamma_y$.

2. From Theorem A.3.2 we have $I(y; z) = I(x; z) + I(y; z|x) \geq I(x; z)$ with equality if and only if $I(y; z|x) = 0$ and thus if and only if $p(z|xy) = p(z|x)$ for all $x, y, z$ for which $p(x, y) > 0$. Similarly to what was said in assertion 1, we conclude that this condition is satisfied if and only if $x \equiv f(x) \pmod{Z}$ for all $x \in \Gamma_x$. □

**Theorem B.1.6** *Let $X, Y$ and $Z$ be jointly distributed random variables defined on a common probability space $(\Omega, \mathcal{A}, \mathcal{P})$ and let $f$ be a measurable function.*

1. *Let $Z = f(Y)$. Then $y \equiv f(y) \pmod{X}$ for all $y \in \Gamma_y$ if and only if $\forall y_1, y_2 \in \Gamma_y : f(y_1) = f(y_2) \Rightarrow y_1 \equiv y_2 \pmod{X}$.*

2. *Let $Y = f(X)$. Then $x \equiv f(x) \pmod{Z}$ for all $x \in \Gamma_x$ if and only if $\forall x_1, x_2 \in \Gamma_x : f(x_1) = f(x_2) \Rightarrow x_1 \equiv x_2 \pmod{Z}$.*

**Proof:**

1. First assume that $y \equiv f(y) \pmod{X}$ for all $y \in \Gamma_y$. Let $y_1, y_2 \in \Gamma_y$. Using the equivalence relation in Lemma 2.4.1, we conclude from $f(y_1) = f(y_2)$ that $y_1 \equiv f(y_1) \equiv f(y_2) \equiv y_2 \pmod{X}$.

   To prove the converse, assume that $y_1, y_2 \in \Gamma_y : y_1 \equiv f(y_1) \pmod{X}$, $y_2 \not\equiv f(y_2) \pmod{X}$. We then conclude from $f(y_1) = f(y_2)$ that $y_1 \equiv f(y_1) \equiv f(y_2) \not\equiv y_2 \pmod{X}$.

2. Similar to the proof of assertion 1. □

**Theorem 2.4.1** *Let $X, Y$ and $Z$ be jointly distributed random variables defined on a common probability space $(\Omega, \mathcal{A}, \mathcal{P})$ and let $f$ be a measurable function.*

1. *If $Z = f(Y)$, then $I(x; y|z) = 0$ if and only if $\forall y_1, y_2 \in \Gamma_y : f(y_1) = f(y_2) \Rightarrow y_1 \equiv y_2 \pmod{X}$.*

2. *If $Y = f(X)$, then $I(z; y|x) = 0$ if and only if $\forall x_1, x_2 \in \Gamma_x : f(x_1) = f(x_2) \Rightarrow x_1 \equiv x_2 \pmod{Z}$.*

**Proof:** Follows on trivially from Theorems B.1.5 and B.1.6. □

# *Appendix C*

# *Proof of Theorem 2.5.1*

In this appendix we prove Theorem 2.5.1. Before proving the main theorem, however, we first derive some results we need for the proof. To begin with, we will derive some properties and manipulations of the average mutual information. After this, we derive a result about distortions based on single-letter distortion measures.

**Lemma C.1.2** *Let $X_1, \ldots, X_n$ be jointly distributed random variables defined on a common probability space $(\Omega, \mathcal{A}, \mathcal{P})$. For any $m \leq n$ we have*

$$I(x_1, \ldots, x_{n-1}; x_n) \geq I(x_1, \ldots, x_{m-1}; x_n),$$

*with equality if and only if $I(x_m, \ldots, x_{n-1}; x_n | x_1, \ldots, x_{m-1}) = 0$.*

**Proof:** Using the chain rule for information (Theorem A.3.1), we have

$$\begin{aligned} I(x_1, \ldots, x_{n-1}; x_n) &= \\ &= I(x_1, \ldots, x_{m-1}; x_n) + I(x_m, \ldots, x_{n-1}; x_n | x_1, \ldots, x_{m-1}) \\ &\geq I(x_1, \ldots, x_{m-1}; x_n), \end{aligned}$$

with equality if and only if $I(x_m, \ldots, x_{n-1}; x_n | x_1, \ldots, x_{m-1}) = 0$. □

**Lemma C.1.3** *Let $X_1, \ldots, X_n$ and $Y_1, \ldots, Y_n$ be jointly distributed random variables defined on a common probability space $(\Omega, \mathcal{A}, \mathcal{P})$ and let $X_1, \ldots, X_n$ be independent. Then*

$$I(x; y) = \sum_{i=1}^{n} I(x_i; y_i), \tag{C.1}$$

*if and only if $p(x|y) = \prod_{i=1}^{n} p(x_i|y_i)$.*

**Proof:** First of all, note that since $X_1, \ldots, X_n$ are independent, we have $p(x) = \prod_{i=1}^n p(x_i)$ and

$$\begin{aligned}\sum_{i=1}^n I(x_i; y_i) &= \sum_{i=1}^n \int\int p(x_i, y_i) \log \frac{p(x_i, y_i)}{p(x_i)p(y_i)} dx_i dy_i \\ &= \sum_{i=1}^n \int\int p(x, y) \log \frac{p(x_i, y_i)}{p(x_i)p(y_i)} dxdy \\ &= \int\int p(x, y) \log \left( \prod_{i=1}^n \frac{p(x_i, y_i)}{p(x_i)p(y_i)} \right) dxdy.\end{aligned}$$

Consider only those $x, y$ for which $p(x, y) > 0$. In that case we have

$$\begin{aligned}\sum_{i=1}^n I(x_i; y_i) - I(x; y) &= \\ &= \int\int p(x, y) \log \left( \frac{p(x)p(y)}{p(x, y)} \prod_{i=1}^n \frac{p(x_i, y_i)}{p(x_i)p(y_i)} \right) dxdy \\ &= \int\int p(x, y) \log \left( \frac{\prod_{i=1}^n p(x_i|y_i)}{p(x|y)} \right) dxdy. \qquad \text{(C.2)}\end{aligned}$$

At this point we employ the well-known inequality $\ln x \leq x - 1$ with equality if and only if $x = 1$ in order to deduce from (C.2) that

$$\begin{aligned}\sum_{i=1}^n I(x_i; y_i) - I(x; y) &\leq \\ &\leq \log e \int\int p(x, y) \left( \frac{\prod_{i=1}^n p(x_i|y_i)}{p(x|y)} - 1 \right) dxdy \qquad \text{(C.3)} \\ &= \log e \int\int \left( p(y) \prod_{i=1}^n p(x_i|y_i) - p(x, y) \right) dxdy \\ &\leq 0, \qquad \text{(C.4)}\end{aligned}$$

since the integration is over those $(x, y)$ pairs for which $p(x, y) > 0$. We have equality in (C.3) if and only if $\prod_{i=1}^n p(x_i|y_i) = p(x|y)$ for all $x, y : p(x, y) > 0$ and equality in (C.4) if and only if $\prod_{i=1}^n p(x_i|y_i) = 0$ when $p(x|y) = 0$. Thus, both inequalities are satisfied with equality, and consequently we have (C.1) if and only if $\prod_{i=1}^n p(x_i|y_i) = p(x|y)$. This completes the proof. □

**Lemma C.1.4** *Let $X_1, \ldots, X_n$ and $Y_1, \ldots, Y_n$ be jointly distributed random variables defined on a common probability space $(\Omega, \mathcal{A}, \mathcal{P})$ and let $X_1, \ldots, X_n$ be independent. If $p(y|x) = \prod_{i=1}^n p(y_i|x_i)$, then*

$$p(x|y) = \prod_{i=1}^n p(x_i|y_i).$$

**Proof:** Since $X_1, \ldots, X_n$ are independent we have $p(x) = \prod_{i=1}^n p(x_i)$. Therefore, we have

$$\begin{aligned} p(x, y) &= p(x)p(y|x) \\ &= \prod_{i=1}^n p(x_i)p(y_i|x_i) \\ &= \prod_{i=1}^n p(x_i, y_i). \end{aligned} \tag{C.5}$$

Integration of (C.5) over $x$ yields $p(y) = \prod_{i=1}^n p(y_i)$ so that $Y_1, \ldots, Y_n$ are independent as well. Moreover, we conclude

$$\begin{aligned} p(x|y) &= \frac{p(x, y)}{p(y)} \\ &= \prod_{i=1}^n \frac{p(x_i, y_i)}{p(y_i)} \\ &= \prod_{i=1}^n p(x_i|y_i) \end{aligned}$$

which completes the proof. □

**Lemma C.1.5** *Let $\rho_N(x, y)$ be a single-letter distortion measure defined by (2.1), and $x = (x(n))_{n=1}^N$, $y = (y(n))_{n=1}^N$. Then*

$$d(x, y) = \frac{1}{N} \sum_{n=1}^N d(x(n), y(n)).$$

**Proof:**

$$\begin{aligned} d(x,y) &= \int\int p(x,y)\rho_N(x,y)dxdy \\ &= \frac{1}{N}\sum_{n=1}^{N}\int\int p(x,y)\rho(x(n),y(n))dxdy \\ &= \frac{1}{N}\sum_{n=1}^{N}\int\int p(x(n),y(n))\rho(x(n),y(n))dx(n)dy(n) \\ &= \frac{1}{N}\sum_{n=1}^{N} d(x(n),y(n)). \end{aligned}$$

□

In words, a single-letter distortion measure gives rise to a distortion which itself is the sum of distortions of the sample values. We are now ready to prove the main theorem.

**Theorem 2.5.1** *Let $T_a$ and $T_s$ be information-preserving signal transforms such that $u = T_a x$ and $y = T_s v$. If $x, y$ are $N$-element sequences and $u, v$ are $M$-element sequences, and $R_x(D)$ and $R'_u(D)$ denote the rate-distortion function of the source process and the transformed process where the filter bank channels are coded separately, respectively, then $R'_{u,M}(D) \geq \mu^{-1} R_{x,N}(D)$, $\mu = \frac{M}{N}$, with equality if and only if*

$$r(u) = \prod_{k=1}^{K} r(u_k). \tag{C.6}$$

**Proof:** Since $T_a$ and $T_s$ are information-preserving transforms, we conclude from Theorem 2.4.4 that $R_{u,M}(D) = \mu^{-1} R_{x,N}(D)$ so that it is sufficient to show that $R'_{u,M}(D) \geq R_{u,M}(D)$ if and only if $r(u) = \prod_{k=1}^{K} r(u_k)$. First assume that (C.6) holds so that $I(u_k; u_1 \cdots u_{k-1}) = 0$ for all $k = 1, \ldots, K$. To show that $R'_{u,M}(D) = R_{u,M}(D)$ we will show that the infimum of $I(u; v)$ on $\mathcal{R}_D$ is achieved on $\mathcal{R}'_D$. Using the chain rule for information we conclude that

$$\begin{aligned} I(u;v) &= \sum_{k=1}^{K} I(u_k; v|u_1 \cdots u_{k-1}) \\ &= \sum_{k=1}^{K} (I(u_k; vu_1 \cdots u_{k-1}) - I(u_k; u_1 \cdots u_{k-1})) \end{aligned}$$

$$
\begin{aligned}
&= \sum_{k=1}^{K} I(u_k; vu_1 \cdots u_{k-1}) \\
&\geq \sum_{k=1}^{K} I(u_k; v_k), \qquad \text{(C.7)}
\end{aligned}
$$

from Lemma C.1.2. Let $R_{u_i}(D_i)$ denote the rate-distortion function of the random process corresponding to the sample sequence $u_i$. Using the convexity of $R_{u,M}$, Lemma C.1.5 and (C.7), we conclude that

$$
\begin{aligned}
\frac{1}{M} I(u; v) &\geq \frac{1}{K} \sum_{k=1}^{K} \frac{K}{M} I(u_k; v_k) \qquad \text{(C.8)} \\
&\geq \frac{1}{K} \sum_{k=1}^{K} R_{u_k, \frac{M}{K}}(D_k) \\
&\geq R_{u,M}\left( \frac{1}{K} \sum_{k=1}^{K} D_k) \right) \\
&= R_{u,M}(D).
\end{aligned}
$$

Hence, a necessary condition for achieving the infimum of $I(u; v)$ on $\mathcal{R}_D$ is that we have equality in (C.8). Using Lemmas C.1.3 and C.1.4, we conclude that this is achieved by choosing $r(v|u) = \prod_{k=1}^{K} r(v_k|u_k)$. In that case we have

$$
\begin{aligned}
r(u, v) &= r(u)r(v|u) \\
&= \prod_{k=1}^{K} r(u_k)r(v_k|u_k) \\
&= \prod_{k=1}^{K} r(u_k, v_k).
\end{aligned}
$$

Consequently, we have $r(v) = \prod_{k=1}^{K} r(v_k)$ so that $I(v_k; v_1 \cdots v_{k-1}) = 0$ for all $k = 1, \ldots, K$. To show that $r(v_k, u|u_k) = r(v_k|u_k)r(u|u_k)$ for all $k = 1, \ldots, K$, again using the chain rule for information, we conclude that

$$
\begin{aligned}
I(u; v) &= \sum_{k=1}^{K} I(u; v_k|v_1 \cdots v_{k-1}) \\
&= \sum_{k=1}^{K} (I(v_k; uv_1 \cdots v_{k-1}) - I(v_k; v_1 \cdots v_{k-1}))
\end{aligned}
$$

$$
\begin{aligned}
&= \sum_{k=1}^{K} I(v_k; uv_1 \cdots v_{k-1}) \\
&= \sum_{k=1}^{K} (I(u_k; v_k) + I(v_k; uv_1 \cdots v_{k-1}|u_k)) \\
&= \sum_{k=1}^{K} I(u_k; v_k),
\end{aligned}
$$

so that $0 = I(v_k; uv_1 \cdots v_{k-1}|u_k) \geq I(v_k; u|u_k) \geq 0$ for all $k$. This is satisfied if and only if $r(v_k, u|u_k) = r(v_k|u_k)r(u|u_k)$ for all $k = 1, \ldots, K$, so that the infimum of $I(u; v)$ on $\mathcal{R}_D$ is achieved on $\mathcal{R}'_D$.

Conversely, let us assume that $R'_{u,M}(D) = R_{u,M}(D)$ and thus $I(v_k; uv_1 \cdots v_{k-1}|u_k) = 0$ for all $k$. In that case we have

$$
\begin{aligned}
I(u; v) &= \sum_{k=1}^{K} I(u; v_k|v_1 \cdots v_{k-1}) \\
&= \sum_{k=1}^{K} (I(v_k; uv_1 \cdots v_{k-1}) - I(v_k; v_1 \cdots v_{k-1})) \\
&= \sum_{k=i}^{K} (I(u_k; v_k) + I(v_k; uv_1 \cdots v_{k-1}|u_k) - I(v_k; v_1 \cdots v_{k-1})) \\
&= \sum_{k=1}^{K} (I(u_k; v_k) - I(v_k; v_1 \cdots v_{k-1})) \\
&\leq \sum_{k=1}^{K} I(u_k; v_k). \qquad \text{(C.9)}
\end{aligned}
$$

Therefore, using the definition of $R_{u,M}(D)$, we conclude that

$$
\begin{aligned}
R_{u,M}(D) &\leq \frac{1}{M} I(u; v) \\
&\leq \frac{1}{M} \sum_{k=1}^{K} I(u_k; v_k). \qquad \text{(C.10)}
\end{aligned}
$$

Since we assumed that $R'_{u,M}(D) = R_{u,M}(D)$ we have equality in (C.10) and thus in (C.9) so that $I(v_k; v_1 \cdots v_{k-1}) = 0$ for all $k = 1, \ldots, K$, or equivalently $r(v) = \prod_{k=1}^{K} r(v_k)$. Therefore, equality in (C.9) also implies $r(v|u) = \prod_{k=1}^{K} r(v_k|u_k)$ by

Lemma C.1.3 and we conclude that

$$\begin{aligned} r(u) &= \frac{r(u,v)}{r(v|u)} \\ &= r(u|v)\prod_{k=1}^{K}\frac{r(v_k)}{r(v_k|u_k)} \\ &= \prod_{k=1}^{K} r(u_k)\frac{r(u|v)}{\prod_{k=1}^{K} r(u_k|v_k)}. \end{aligned}$$

Note however that $r(u)$ is independent of $D$. Therefore, for all $D > 0$ and all $u$ and $v$ we have

$$r(u|v) = c(u)\prod_{k=1}^{K} r(u_k|v_k),$$

where

$$c(u) = \frac{r(u)}{\prod_{k=1}^{K} r(u_k)},$$

is independent of $D$. To complete the proof we show that $c(u) = 1$. Thus we fix $u$. Since $\rho(u, v)$ discriminates well in the sense that it vanishes only for $v = u$, we get that, as $D \downarrow 0$,

$$r(u|v) \to \delta(u - v),$$

$$r(u_k|v_k) \to \delta(u_k - v_k),$$

in the sense of probabilities. Now since

$$\delta(u - v) = \prod_{k=1}^{K} \delta(u_k - v_k),$$

it follows that $c(u) = 1$, which completes the proof. □

# Appendix D

# The condition of a set of linear equations

In this appendix we review the notion of the *condition* of a set of linear equations and the *condition number* which quantifies the sensitivity or numerical stability of a set of linear equations. We shall rely on this discussion to study the numerical behaviour of various realizations.

Let us consider the computing of

$$y = Ax, \tag{D.1}$$

where $A \in \mathbb{C}^{n\times n}$, $x \in \mathbb{C}^n$ and $A$ is non-singular. The condition of (D.1) measures, by definition, the worst relative variation of $y$ with respect to the corresponding allowable variation of $A$ and $x$. Let the measure we take be the norm for vectors, and the related operator norm for matrices, and let $A$ and $x$ be subject to arbitrary (but infinitesimal) variations, such that $y + dy = (A + dA)(x + dx)$. Thus, neglecting second order effects, we have

$$dy = Adx + (dA)x, \tag{D.2}$$

and, therefore,

$$\|dy\| \leq \|A\|\|dx\| + \|dA\|\|x\|.$$

Let

$$A = U\Sigma V^* = [\eta_1 \cdots \eta_n] \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_n \end{bmatrix} \begin{bmatrix} \zeta_1^* \\ \vdots \\ \zeta_n^* \end{bmatrix},$$

be the singular value decomposition (SVD) of $A$. $U$ and $V$ are $n \times n$ unitary matrices and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n > 0$ are the singular values. The inverse $A^{-1}$ of $A$ is, then,

$$A^{-1} = V\Sigma^{-1}U^* = [\zeta_1 \cdots \zeta_n]\,\Sigma^{-1}\begin{bmatrix} \eta_1^* \\ \vdots \\ \eta_n^* \end{bmatrix}.$$

If we write (D.2) as $dy = A\left(dx + A^{-1}(dA)x\right)$, $\|dy\|$ is maximum when $dx + A^{-1}(dA)x$ is in the direction of $\zeta_1$, since the singular vector $\zeta_1$ corresponds to the largest singular value of $A$, namely $\sigma_1$. Moreover, from $x = A^{-1}y$ it follows that

$$\|x\| \leq \|A^{-1}\|\|y\|,$$

with equality when $y$ is in the direction of $\eta_n$ ($x$ in the direction $\zeta_n$) since the singular vector $\eta_n$ corresponds to the largest singular value of $A^{-1}$, namely $\sigma_n^{-1}$. It follows that

$$\frac{\|dy\|}{\|y\|} \leq \|A^{-1}\|\|A\|\left(\frac{\|dA\|}{\|A\|} + \frac{\|dx\|}{\|x\|}\right), \tag{D.3}$$

with equality as soon as the two directional conditions are met. We conclude that

$$\frac{\|dy\|/\|y\|}{\|dA\|/\|A\| + \|dx\|/\|x\|}, \tag{D.4}$$

is bounded by $\|A^{-1}\|\|A\|$, and that the bound may be obtained under certain conditions of $A, x, dA$ and $dx$. For this reason

$$\kappa(A) = \|A^{-1}\|\|A\|,$$

is called the *condition number* of the system of linear equations. Note that $\kappa(\cdot)$ depends on the underlying norm. When the norm is to be stressed, we use a subscript. Thus, for the 2-norm, $\kappa(A)$ is given by

$$\kappa_2(A) = \frac{\sigma_1}{\sigma_n}.$$

We have the following properties.

**Corollary D.1.1** *The following statements hold for a given norm:*

1. $\kappa(A) \geq 1$.

2. $\kappa(A_1 A_2 \cdots A_n) \leq \kappa(A_1)\kappa(A_2)\cdots\kappa(A_n)$.

3. $\kappa(\alpha A) = \kappa(A)$ *for any non-zero* $\alpha \in \mathbb{C}$.

**Proof:**

1. $\kappa(A) = \|A^{-1}\|\|A\| \geq \|A^{-1}A\| = \|I\| \geq 1$.

2. $\kappa(A_1 A_2 \cdots A_n) = \|A_n^{-1} \cdots A_1^{-1}\|\|A_1 \cdots A_n\| \leq \|A_n^{-1}\| \cdots \|A_1^{-1}\|\|A_1\| \cdots \|A_n\| = \|A_1^{-1}\|\|A_1\| \cdots \|A_n^{-1}\|\|A_n\| = \kappa(A_1)\kappa(A_2)\cdots\kappa(A_n)$.

3. Since $\|\alpha A\| = |\alpha|\|A\|$ and $\|(\alpha A)^{-1}\| = |\alpha|^{-1}\|A^{-1}\|$ we conclude that $\kappa(\alpha A) = |\alpha|^{-1}|\alpha|\|A^{-1}\|\|A\| = \|A^{-1}\|\|A\| = \kappa(A)$, as required.

□

When the matrix-vector multiplication $y = Ax$ is implemented, inequality (D.3) maintains that round-off noise comes from two sources. One is the actual error in $dA$ and/or $dx$. The other is the *natural sensitivity* of the problem, which is measured by $\kappa(A)$. Note that $\kappa(A)$ may be very large even for matrices with reasonable eigenvalues. For example, consider the matrix

$$A = \begin{bmatrix} 1 & 0 \\ 10 & 1 \end{bmatrix}. \tag{D.5}$$

Although the eigenvalues are all equal to one, the condition number is approximately equal to 100 since $\sigma_1 \approx 10$ and $\sigma_2 \approx 0.1$. The difficulty with these types of matrices is that a "large" off-diagonal entry in $A$ means an equally large entry in $A^{-1}$ – contrary to the general expectation that $A^{-1}$ would get smaller as $A$ gets bigger. If $\kappa(A)$ is large, then $A$ is said to be *ill-conditioned*. Note that this is a norm dependent property. Matrices with small condition numbers are said to be *well-conditioned*. In the 2-norm, unitary matrices are perfectly conditioned in that $\kappa_2(A) = 1$ if $A$ is unitary. We conclude that unitary maps are optimal in terms of numerical stability.

It is interesting to see what happens if $A$ itself is a product of matrices, i.e. $A = A_n \cdots A_1$. We have the following result.

**Theorem D.1.7** *Let* $y = Ax$ *with* $A = A_n \cdots A_1 x$. *Then*

$$\frac{\|dy\|}{\|y\|} \leq \sum_{i=1}^{n} \prod_{j=i}^{n} \kappa(A_j) \frac{\|dA_i\|}{\|A_i\|} + \prod_{j=1}^{n} \kappa(A_j) \frac{\|dx\|}{\|x\|}.$$

**Proof:** The proof is by induction. The case $n = 1$ yields (D.3). Let $n > 1$ and $y^{(n+1)} = A_{n+1}Ax$ so that $dy^{(n+1)} = dA_{n+1}(Ax) + A_{n+1}d(Ax) = dA_{n+1}y + A_{n+1}dy$. Using the induction hypothesis we conclude that

$$\begin{aligned}
\frac{\|dy^{(n+1)}\|}{\|y^{(n+1)}\|} &\leq \kappa(A_{n+1})\left(\frac{\|dy\|}{\|y\|} + \frac{\|dA_{n+1}\|}{\|A_{n+1}\|}\right) \\
&\leq \kappa(A_{n+1})\left(\sum_{i=1}^{n}\prod_{j=i}^{n}\kappa(A_j)\frac{\|dA_i\|}{\|A_i\|} + \prod_{j=1}^{n}\kappa(A_j)\frac{\|dx\|}{\|x\|}\right. \\
&\qquad \left. + \frac{\|dA_{n+1}\|}{\|A_{n+1}\|}\right) \\
&= \sum_{i=1}^{n+1}\prod_{j=i}^{n+1}\kappa(A_j)\frac{\|dA_i\|}{\|A_i\|} + \prod_{j=1}^{n+1}\kappa(A_j)\frac{\|dx\|}{\|x\|}.
\end{aligned}$$

□

Theorem D.1.7 and Corollary D.1.1, assertion 2, show that even when $A$ itself is well-conditioned, the quantity (D.4) can become arbitrarily large, depending on the factorization $A = A_n \cdots A_1$. However, if $A_j$ is unitary for all $j = 1, \ldots, n$, we have $\kappa_2(A) = \kappa_2(A_n) \cdots \kappa_2(A_1) = 1$ and we conclude that unitary factorizations of $A$ are optimal in terms of numerical stability.

# *Samenvatting*

De laatste decennia worden beelden steeds vaker digitaal gerepresenteerd. Digitale representatie van beelden heeft een aantal belangrijke voordelen boven analoge representatie, zoals flexibiliteit, robuustheid en geschiktheid voor een verscheidenheid aan signaalprocessingstechnieken. Het digitaal representeren van beelden heeft ook een nadeel, namenlijk dat de opslag of transmissie van deze signalen snelle interfaces en een respectievelijk grote opslagcapaciteit of transmissiebandbreedte vereist. Een mogelijkheid om het aantal bits te reduceren, en daarmee de opslag- of transmissiekosten te verlagen, is het toepassen van *datacompressie*.

Transformatiecodering is een van de meest efficiënte methoden voor datacompressie van gecorreleerde signalen. Het gebruik van een tijd-discrete signaaltransformatie voor de werkelijke codering (d.w.z. voor de kwantisatie en de afbeelding op binaire codewoorden) kan een significante reductie van de codercomplexiteit opleveren. Een geschikte signaaltransformatie leidt tot signaalrepresentaties die eenvoudig te coderen zijn. Dit proefschrift beschrijft ontwerp, applicatie en realisatie van tijd-discrete signaaltransformaties voor datacompressie, in het bijzonder datacompressie van beelden.

In hoofdstuk 2 laten we zien dat een tijd-discrete signaaltransformatie beschouwd kan worden als een $(K, L, m)$ filterbank, waarbij $K$ het aantal filterkanalen, $L$ de filterlengte en $m$ de decimatiefactor voorstelt. Het plaatsen van tijd-discrete signaaltransformaties in dit meer algemene framewerk betekent dus dat het ontwerp van signaaltransformaties geformuleerd kan worden als zijnde het vinden van een geschikte filterbankarchitectuur (keuze van $K$, $L$ en $m$) en het kiezen van geschikte filterresponsies. Om de invloed van verschillende keuzes voor $K$, $L$ en $m$ op de codeerefficiëntie te onderzoeken, maken we gebruik van

rate-distortietheorie.

In hoofstuk 3 concentreren we ons op het werkelijke ontwerp van de filterbank. Hier bespreken we welke factoren, aanvullend op de codeerefficiëntie, de keuze van een architectuur en filterresponsies beïnvloeden. Deze factoren zijn, bijvoorbeeld, perceptuele kwaliteit en implementatie-aspecten. Gebaseerd op deze discussie stellen we een lijst samen van gewenste beperkingen en gebruiken deze om een geschikte filterbankarchitectuur te kiezen. Bovendien laten we zien hoe geschikte filterresponsies ontworpen kunnen worden, d.w.z. filterresponsies die aan de gestelde beperkingen voldoen. We tonen aan dat de klasse van zogenaamde *overlappende orthogonale transformaties* (LOT's) een geschikte kandidaat is voor onze applicatie van beeldcodering.

In hoofdstuk 4 bespreken we de applicatie van overlappende transformatiecodering van X-ray cardio-angiografische beeldsequenties. Met verliesvrije codeertechnieken kan de bitrate van deze X-ray beeldsequenties met een factor 2,5 – 3,5 gereduceerd worden. Het doel van het werk dat hier beschreven wordt is om reductiefactoren in de ordegrootte van 8 – 16 te verkrijgen. Om dit te bewerkstelligen maken wij gebruik van niet-verliesvrije codeertechnieken, in het bijzonder de technieken die gebaseerd zijn op overlappende transformatiecodering met LOT's. We beginnen met het bespreken van X-ray angiografie. Deze specifieke applicatie legt enige additionele eisen op aan de parametersetting van de filterbank. Vervolgens bespreken we het gehele transformatiecodeersysteem in meer detail en beschrijven resultaten die verkregen zijn met een software-implementatie van deze methode. Dit codeerschema is door Philips ingebracht in de standaardisatiediscussie van niet-verliesvrije datacompressie-algoritmen, georganiseerd door de ACR-NEMA commissie welke wordt ondersteund door de National Electrical Manufacturers Association (NEMA) en de American College of Radiology (ACR). We laten zien dat LOT's in staat zijn om X-ray beelden effectief te decorreleren, hetgeen een efficiënte codering van het getransformeerde signaal vereenvoudigd. We tonen ook aan dat LOT's bijzonder geschikt zijn om de compressie aan te passen aan het menselijk visueel systeem en aan eventuele post-processing, zoals beeldopscherping.

We eindigen dit proefschrift met de realisatie van tijd-discrete signaaltransformaties. Dit wordt gedaan in hoofdstuk 5. Het ontwerp van realisaties, in het bijzonder van realisaties die geschikt zijn voor *very large scale integration* (VLSI) technologie, bestaat uit het bepalen van een geschikt algoritme om de signaaltransformatie te berekenen en het afbeelden van dit algoritme op een architectuur. Wij laten zien dat, door *zowel* algoritme *als* architectuurontwerp gezamenlijk te beschouwen, wij numeriek robuuste en implementatie-efficiënte realisaties van

filterbanken kunnen ontwerpen. Deze realisaties kunnen volledig programmeerbaar gemaakt worden in de zin dat de architectuur gebruikt kan worden voor willekeurige signaaltransformaties, inclusief de discrete cosinustransformatie, de LOT of discrete wavelettransformaties.

# *Acknowledgements*

I would like to thank a number of people who have contributed to this thesis.

First of all I would like to thank Marcel Breeuwer for his guidance and encouragement during my research. He has been closely involved in the work described in this thesis.

I am greatly indebted to the management of the Philips Research Laboratories in Eindhoven, and to the head of the digital signal processing group, Peter van Otterloo, in particular, under whose supervision I carried out my research. I would like to thank them for their support and for providing the excellent research environment that enabled me to prepare this thesis. I would also like to thank all the countless friendly people and colleagues I met at Philips, who together create the right scientific atmosphere which leads to exciting research. I am especially indebted to my room-mate, former fellow-student and, above all, dear friend, Rik Theunis.

I would like to thank Prof. Jan Biemond and Prof. Ed Deprettere of Delft University of Technology for their constructive remarks and their guidance during the process of completing this thesis. I would like to thank Gerben Hekstra at Delft University of Technology and Zeng Zhiqiang at Tsinghua University, Beijing, for their contribution and the fruitful discussion of several aspects of realization and VLSI-implementation design and Paul Zwart of Philips Medical Systems for providing me with the JPEG coded sequences. I would also like to thank Stan Baggen, Rob Beuker, Marcel Breeuwer, Henk Hollmann, Guido Janssen, René Klein Gunnewiek, Gertjan Keesman, Ton Kalker, Rik Theunis and Ludo Tolhuizen, all at Philips Research, and Frans Willems of Eindhoven University of Technology for reviewing and commenting on parts of the text.

Last, but not least, I am very grateful to the people who gave me moral sup-

port. Firstly, my parents who always encouraged me to do more than just enough and who gave me the opportunity to study at Delft University of Technology. Secondly, my wife Karin for her support in general which was invaluable in enabling me to complete this thesis.

# *Curriculum vitae*

Richard Heusdens was born in Alkmaar, the Netherlands, in 1965. In 1984 he obtained his HAVO (upper general secondary education) diploma from the Jan Arentz College in Alkmaar. In 1988 he obtained his HTS (institute of technology) diploma from the HTS Alkmaar. Next, he enrolled in the Department of Electrical Engineering of Delft University of Technology, the Netherlands, from which he received his M.Sc. degree in 1992.

In the spring of 1992 he joined the digital signal processing group at the Philips Research Laboratories in Eindhoven, the Netherlands. He has worked on various topics in the field of signal processing, such as image/video compression and VLSI architectures for image-processing algorithms. The research discussed in this thesis was carried out at the Philips Research Laboratories. Richard Heusdens is currently studying various topics in the field of audio/speech coding and wireless communication.

# *Glossary of symbols*

## General

| | |
|---|---|
| $T_a$, $T_s$ | analysis, synthesis operator |
| $h_k$, $f_k$ | analysis, synthesis filter |
| $\hat{h}_k$, $\hat{f}_k$ | Fourier-transform of $h_k$, $f_k$ |
| $H(z)$, $F(z)$ | $\mathcal{Z}$-transform of $h_k$, $f_k$ |
| $K$ | number of filter bank channels |
| $L$ | filter length |
| $m$ | decimation (down sample) factor |

## Chapter 2: Rate-distortion performance of discrete-time signal transforms

| | |
|---|---|
| $(\Omega, \mathcal{A}, \mathcal{P})$ | probability space |
| $\{X_n, n \in \mathbb{Z}\}$ | discrete source (random process) |
| $\{U_n, n \in \mathbb{Z}\}$ | transformed source process |
| $\{V_n, n \in \mathbb{Z}\}$ | decoded transformed process |
| $\{Y_n, n \in \mathbb{Z}\}$ | reproduced process |
| $x$ | sample sequence of $\{X_n, n \in \mathbb{Z}\}$ |
| $u$ | sample sequence of $\{U_n, n \in \mathbb{Z}\}$ |
| $v$ | sample sequence of $\{V_n, n \in \mathbb{Z}\}$ |
| $y$ | sample sequence of $\{Y_n, n \in \mathbb{Z}\}$ |
| $M$ | row dimension of $T_a$ |
| $N$ | column dimension of $T_a$ |
| $R_x$ | rate-distortion function of the source |
| $R_u$ | rate-distortion function of the transformed source |

| | |
|---|---|
| $R'_u$ | rate-distortion function of the transformed source where the filter bank channels are coded separately |
| $E$ | expectation operator |
| $S$ | spectral-density function |

**Chapter 3: Design of lapped orthogonal transforms**

| | |
|---|---|
| $A, S$ | matrix with rows being the analysis, synthesis filters |
| $\mathcal{A}, \mathcal{S}$ | set of analysis, synthesis filters |
| $A_e, A_o$ | even, odd-symmetric parts of constructing filters |
| $B_a$ | transformation matrix for shaping $A_o$ |

**Chapter 4: Application to X-ray cardio-angiograms**

| | |
|---|---|
| $Q$ | quantizer |
| $C$ | lossless coder |
| $x_r, x_e$ | raw, enhanced data |
| $h_{k,l}$ | two-dimensional analysis filter |
| $h_{enh}$ | enhancement filter |

**Chapter 5: Realizations for discrete-time signal transforms**

| | |
|---|---|
| $A, S$ | matrix with rows being the analysis, synthesis filters |
| $R, Q$ | lower-triangular, unitary factor of the RQ-factorization |
| $L$ | reduced lower-triangular matrix, i.e., $R = [L \;\; O]$ |
| $X$ | augmented analysis filter matrix, i.e., $X = [P \;\; A]$ |
| $F_I, \ldots, F_{IV}$ | type I, ...,type IV $\mu$-rotation |
| $\mathcal{F}$ | set of feasible angles |
| $A_e, A_o$ | even, odd-symmetric parts of constructing filters |
| $B_a$ | transformation matrix for shaping $A_o$ |
| $U$ | composite matrix $U = [A_e^* \;\; A_o^*]^*$ |
| off$(X)$ | off-diagonal energy |
| $(p, q)$ | rotation indices |
| $(\theta, \gamma)$ | rotation angle and argument of (complex) sine/cosine |
| $\Delta_{p,q}^{(i)}$ | increase in diagonal energy caused by a rotation in the $(p, q)$ plane at iteration $i$ |

# *Index*

Bold face page numbers are used to indicate pages with important information about the entry, e.g., the formal definition of mathematical terms or a detailed explanation, while page numbers in normal type indicate a secondary discussion or a textual reference.