Introduction to Signal and Image Processing

Tomas Sauer

Lehrstuhl für Mathematik mit Schwerpunkt Digitale Bildverarbeitung
FORWISS
University of Passau
Innstr. 43
94032 Passau

Version 2.0
Last modifications: 4.7.2022

Chaos is found in greatest abundance whereever order is being sought. It always defeats order, because it is better organized.

T. Pratchett, *Interesting times*

When the epoch of analogue (which was to say also the richness of language, of *analogy*) was giving way to the digital era, the final victory of the numerate over the literate.

S. Rushdie, *Fury*

The most incredible thing about miracles is that they happen.

G. K. Chesterton, *The Innocence of Father Brown*

And it didn't stop being magic just because you found out how it was done.

T. Pratchett, *Wee Free Men*

Tomas Sauer
Chair for Mathematical Image Processing
University of Passau
Innstr. 43
94032 Passau

Contents

1 Image Acquisition

Well, it is easy enough to look wise; it is when a man opens his mouth that the test begins

(P. J. Brebner, *Christopher Quarles College Professor and Master Detective*)

Digital Image Processing is concerned with the processing of digital images. Well, that's just the name of it. In fact, the english term "'image"' is more to the point insofar as an image does not only have to be a picture that digitized by pixellation. This makes image processing more interesting and challenging than just handling pictures. Some classical tasks in image processing can be summarized and classified as follows:

1. **Image Enhancement**: the original image is not of the desired quality, be it due to defects in the acquisition device or due to circumstances or for whatever reasons. Such defects can be noise, over- or underexposure, bad contrast or other structured or unstructured artifacts[1]. The goal is then to derive algorithms that remove or at least reduced these artifacts.

2. **Image Restoration**: parts of the image are corrupted or missing and should be reconstructed from other parts of the image.

3. **Image Compression**: how to store huge images[2] in such a way that the quality is not deteriorated too much. And, by the way, how do we measure quality?

4. **Information Extraction**: our image contains information that has to be extracted automatically. Medical imaging is a classical example since the physicists are not really interested in the image itself but in the diagnosis it implies.

It is clear that answering such questions will require quite some mathematics. Moreover it is to be expected that the methods will depend to some extent on the *nature* of the image under consideration and probably also to the way how the image was created. Therefore it is a good idea to first have a look at some types of images and their ways of acquisition.

[1] Depending on the context an **artifact** can either be a (usually man made) object to be digitized or a (usually unwanted) effect in the image.

[2] In computed tomography industrial images can easily be of the size of up to 1TB or more – in 2022.

1.1 Image Models

When talking about images most people intuitively think of a picture, normally from photography[3]. For digital methods, the images should be transferred to a digital, numerical format, and the standard mathematical model of images is that of a two dimensional rectangular object. These, however, can be modelled in different ways:

1. **Realistic**: An image is an **array**, whose elements are called **pixel** and encode *discrete* color values:

$$B = \begin{bmatrix} b_{jk} & : & \begin{matrix} j = 1, \dots, m \\ k = 1, \dots, n \end{matrix} \end{bmatrix}, \qquad b_{jk} \in \{0, \dots, M\}.$$

   How precisely the colors are encoded, for example as RDG or YUV, cf. (Foley et al., 1990), will be considered a bit later.

2. **Semicontinuous**: Again we consider a rectangular array, but the gray values or color components will be encoded as real numbers:

$$B = \begin{bmatrix} b_{jk} & : & \begin{matrix} j = 1, \dots, m \\ k = 1, \dots, n \end{matrix} \end{bmatrix}, \qquad b_{jk} \in \mathbb{R} \text{ or } b_{jk} \in \mathbb{R}^3.$$

   Often, the values are restricted to the interval $[0, 1]$. Passing from this model to the discrete values above is done by **quatization** of the values which is a relevant topic in signal processing but will *not* be considered in this lecture.

3. **Continuous**: In many modern approaches to image processing, the image is seen as a function from a rectangle or square[4] to $\mathbb{R}$ or $\mathbb{R}^3$ for example, $B : [0, m] \times [0, n] \to \mathbb{R}^3$. One might see this as pixels that are so dense that they form a continuum. This allows us to use methods and techniques from **analysis** which is the foundation of many modern methods for edge detection or denoising. When applying the theory, the concrete image is a *discretization* of the continuous model, which is not simple but where techniques from Numerical Analysis can be used.

Although most real world images are only covered by the discrete model, it makes perfect sense to consider the other two models as moving away from "reality" allows for the use of more powerful mathematical models. Therefore we make the following general definition which is so general that it is almost no definition any more.

Definition 1.1.1 (Image). An **image** B is a mapping from a **domain** D to some set E^d for a suitable d.

[3] Literally $\phi\omega\tau o$-$\gamma\rho\alpha\varphi\epsilon\iota\nu$ means "painting with light".

[4] One you do not have to count pixels, the sizes in x and y direction do not matter so much any more.

Figure 1.1.1: Image *(left)* and color models. From left to right: the **RGB** format, the standard digital image representation, its complementary **CMY** (Cyan, Magenta, Yellow) model used in printers and the **YUC** and **YCbCr** models used in TV encoding systems like PAL and NTSC where now Y stands for the brightness of the pixel and UV and CrCb encode the chroma (= color) content.

Example 1.1.2 (Color models). The most importand case of $d = 3$ in imaging is that of a **color image** where the color is encoded in three different channels. A **color channel** can be the red, the green and the blue component of the image, but there exist lots of different color models that have their justification in various different applications. For example, one can encode the black and white image in one channel and then additional color information in so-called **chroma channels**. Such models are popular in TV transmission, for example. See Fig. 1.1.1 for some examples of color representation and look up (Foley et al., 1990) for more details on color models.

An important aspect in image processing is to understand what the set E in Definition 1.1.1 stands for and what the values in E mean. For example, in digital photgraphy, incoming light rays are transformed by the optical system, passed through a color filter and finally the intensity is measured by a chip, so the value stands for the intensity of a color component of the incoming light. In X-ray computed tomography, on the other hand, the "gray values" stand for the absorption rate of the material which is somewhat related to the density of this material. Therefore an array of size 512×512 with 8 bit values from $0, \ldots, 255$ can mean something totally different in the two situations, hence my require different methods to be processed.

> The methods to be applied to an image may depend on the methods of image aquisition and the underlying image model.

1.2 Photography

The simplest idea of a photographic image generation would be that of a **parallel projection** of objects in the threedimensional ambient space on the "photographic

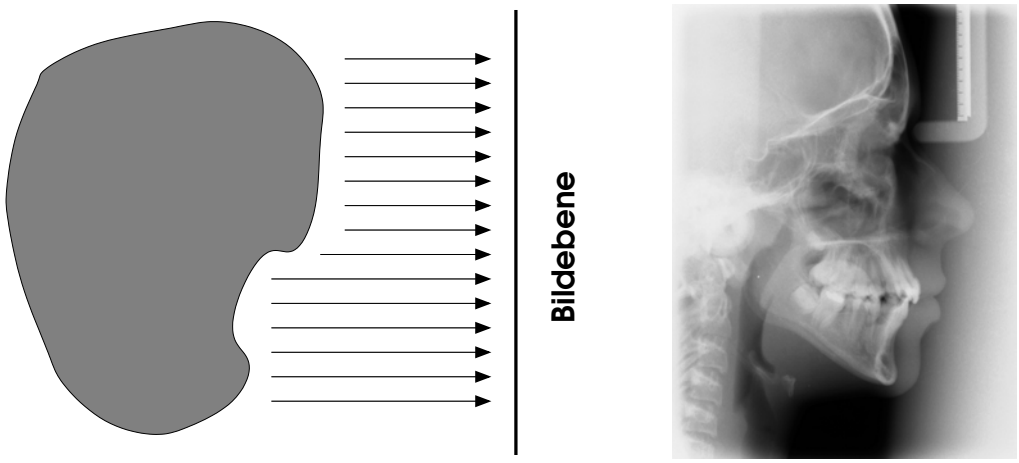


Figure 1.2.1: Parallel projection on the image plane. Such image acquisition methods exist, for example in Ceph X-ray imaging *(left, source: Wikimedia Commons)* where the source is as large as the image and emits parallel rays.

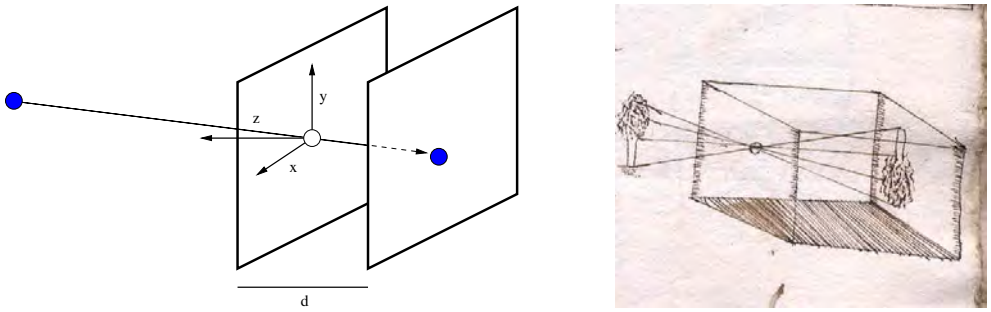


Figure 1.2.2: Central projection in an ideal pinhole camera *(right a schematic drawing from the 17th century, source: Wikipedia)* where the point $\boldsymbol{x}$ is projected on a point $\boldsymbol{x}'$ in the image plane whose distance to the aperture is d.

plate", see Fig. 1.2.1. Parallel projections have no projective distortions and measurements can be taken directly from the image. In optics parallel rays occur whe the object to be imaged is infinitely far away.

1.2.1 Pinholes & Projective Geometry

A somewhat more realistic model of photography is that of the **camera obscura** or **pinhole camera** and is used in the modeling of most camera based images. The mathematical concept behind this imaging is the **central projection**. If we put the origin of our coordinate system into the aperture as in Fig. 1.2.2, then the point

$$\boldsymbol{x} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \in \mathbb{R}^3$$

is mapped to the image point

$$\boldsymbol{x}' = \frac{d}{z} \begin{pmatrix} -x \\ -y \end{pmatrix} \in \mathbb{R}^2 \simeq \mathbb{R}^2 \times \{-d\}$$

in the **image plane**. Since this is an issue in **Projective Geometry**, we switch to **homogeneous coordinates** and embed the point $\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \in \mathbb{R}^3$ cnanonically into the four dimensional plane $\mathbb{R}^3 \times \{1\}$ and then use the equivalence

$$\boldsymbol{x} \simeq \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ 1 \end{pmatrix} \simeq \begin{pmatrix} w\,x_1 \\ w\,x_2 \\ w\,x_3 \\ w \end{pmatrix} \in \mathbb{R}^4, \qquad w \neq 0.$$

Geometrically, we identify all points in the same equivalence class that lie on a straight line through the origin, where the origin itself takes the role of infinity like all points whose last coordinate is zero: $\infty = \mathbb{R}^3 \times \{0\}$.

Conversely, any finite point in $\mathbb{R}^4$, i.e., an point whose last entry is zero, can be identified with a point in $\mathbb{R}^3$ via

$$\mathbb{R}^4 \ni \hat{\boldsymbol{x}} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \mapsto \boldsymbol{x} = \begin{bmatrix} x_1/x_4 \\ x_2/x_4 \\ x_3/x_4 \end{bmatrix} \in \mathbb{R}^3$$

The advantage of this process is that all reasonable geometric transformations in $\mathbb{R}^3$ can now be represented as matrix multiplications:

1. **Translation** by $\boldsymbol{y} \in \mathbb{R}^3$, $T_{\boldsymbol{y}}\boldsymbol{x} = \boldsymbol{x} + \boldsymbol{y}$ is written as

$$T_{\boldsymbol{y}}\hat{\boldsymbol{x}} := \begin{pmatrix} \boldsymbol{I} & \boldsymbol{y} \\ 0^T & 1 \end{pmatrix} \hat{\boldsymbol{x}}.$$

2. **Linear transformation** by a matrix $\boldsymbol{A} \in \mathbb{R}^{3\times 3}$:

$$L_{\boldsymbol{A}}\hat{\boldsymbol{x}} := \begin{pmatrix} \boldsymbol{A} & 0 \\ 0^T & 1 \end{pmatrix} \hat{\boldsymbol{x}}.$$

3. **Affine map** given as matrix multiplication and translation, $L_{\boldsymbol{A},\boldsymbol{y}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{y}$,

$$L_{\boldsymbol{A},\boldsymbol{y}}\hat{\boldsymbol{x}} := \begin{pmatrix} \boldsymbol{A} & \boldsymbol{y} \\ 0^T & 1 \end{pmatrix} \hat{\boldsymbol{x}}.$$

4. **Projection** with factor $d \neq 0$:

$$P_d\hat{\boldsymbol{x}} := \begin{pmatrix} \boldsymbol{I} & 0 \\ -d^{-1}\,\boldsymbol{e}_3^T & 0 \end{pmatrix} \hat{\boldsymbol{x}}.$$

Note that in **cartesian coordinates**, i.e., the "standard" coordinates in $\mathbb{R}^3$, the projection cannot be written as a linear map, i.e., as a matrix multiplication.

Exercise 1.2.1 Does $L_{\boldsymbol{A}}T_{\boldsymbol{y}} = T_{\boldsymbol{y}}L_{\boldsymbol{A}}$ hold? If not, what is the correct formula? ◊

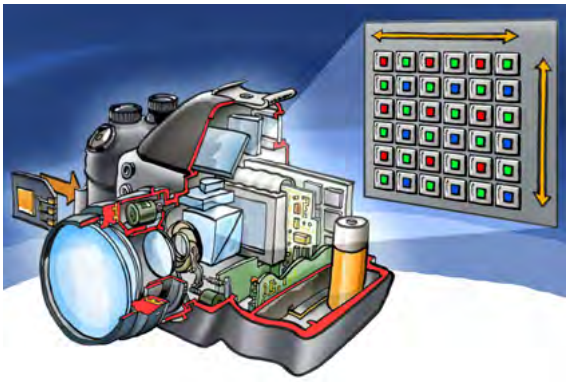


Figure 1.2.3: Digital camera with magnified chip. Each pixel has two green components which is due to that the human eye is more sensitive to green. Source: Peter Welleman @Wikimedia commons

Let us have a closer look at how the projection works to see why this is the math for our pinhole camera. Indeed,

$$\begin{aligned} P_d\hat{\boldsymbol{x}} &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1/d & 0 \end{pmatrix} \begin{pmatrix} wx_1 \\ wx_2 \\ wx_3 \\ w \end{pmatrix} = w \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ -\frac{x_3}{d} \end{pmatrix} = -w\,\frac{x_3}{d} \begin{pmatrix} \frac{d}{x_3}x_1 \\ \frac{d}{x_3}x_2 \\ d \\ 1 \end{pmatrix} \\ &\simeq \begin{pmatrix} -\frac{d}{x_3}x_1 \\ -\frac{d}{x_3}x_2 \\ -d \end{pmatrix} \end{aligned}$$

which is precisely the image point $\boldsymbol{x}$ in the plane $x_3 = -d$, which has the coordinates $-\frac{d}{x_3}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$.

In summary, *all* geometric operations can be realized by simple multiplication with 4×4 matrices, which is also used in so called **kinematic chains** in robotics, cf. (Paul, 1981). This is the reason why such matrix multiplications are integrated in hardware in graphic cards.

This is by far not the end of the story as the pinhole camera is only a very elementary and incomplete model of a camera, totally ignoring the optical part of the camera lens, see Fig. 1.2.3 which induces optical distortions in the image. These distortions are more significant for short focal length while a telfocal length gives a much more homogeneous image, More details about image acquisition in cameras can be found in (Jähne, 2002).

1.2.2 Calibration

The pinhole camera is obviously far from the reality of a real world camera[5] and therefore any camera image contains distortions that are a combination of model

[5]Or, more precisely, noone would buy a pinhole camera nowadays although meanwhile some compensation of lack of optical quality is done by AI in smartphone cameras.

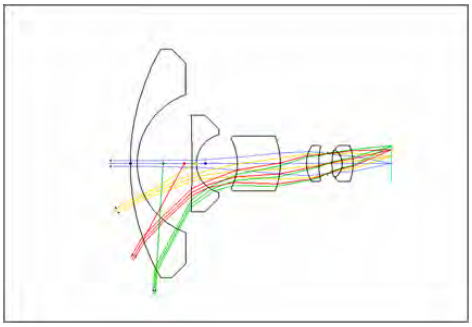



Figure 1.2.4: Schematic representation of a modern camera lense (*left*, source `Jos. Schneider Optische Werke GmbH`) and the famous **fisheye effect** (*right*, source: `Wikimedia Commons`).

errors and imprecisions in manufacturing. These distortions are usually compensated in the image plane. The simplest corrections are based on affine transformations of the form

$$\mathbb{R}^2 \ni \boldsymbol{x} \mapsto \boldsymbol{A}\boldsymbol{x} + \boldsymbol{y}, \qquad \boldsymbol{A} \in \mathbb{R}^{2\times 2},\ \boldsymbol{y} \in \mathbb{R}^2$$

which we can write projectively as $\hat{\boldsymbol{A}}\,\hat{\boldsymbol{x}}$ with

$$\hat{\boldsymbol{A}} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{y} \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{3\times 3}.$$

The translational part $\boldsymbol{y}$ is harmless and "only" fixes the origin in the image plane. The linear part $\boldsymbol{A}$ is decomposed[6] into

$$\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{R} = \boldsymbol{Q}\begin{pmatrix} \alpha & \gamma \\ 0 & \beta \end{pmatrix} = \boldsymbol{Q}\begin{pmatrix} \alpha & 0 \\ 0 & \beta \end{pmatrix}\begin{pmatrix} 1 & \gamma/\alpha \\ 0 & 1 \end{pmatrix} =: \boldsymbol{Q}\boldsymbol{D}\boldsymbol{S}, \qquad \boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I}. \tag{1.2.1}$$

All terms in this formula now have a geometric or optical interpretation:

1. The **orthogonal matrix** $\boldsymbol{Q}$ describes a rotation and maybe a reflection and can be interpreted as a rotation of the camera in the image plane[7] that can be compensated by a proper alignment of the camera.

2. The diagonal **scaling matrix** describes to which extent the x and the y part of the image have to be rescaled, for example to take into account the different resolutions in these directions.

3. The **shear** $\boldsymbol{S}$ results in a nonorthogonality of the image axes, mathematically described as a shearing.

The parameters α, β, γ form (1.2.1) are called **intrinsic parameters** and determining them is known as **intrinsic calibration** of the camera. Fixing the camera pose relative to world coordinates is called **extrinsic calibration**. Image processing libraries like `OpenCV` offer a variety of calibration functions without which optical measurements are impossible.

[6] Here we use the QR **decomposition** from Numerical Linear Algebra: any matrix $A \in \mathbb{R}^{n\times n}$ can be written as $A = QR$ where Q is an orthogonal matrix, i.e., $Q^TQ = QQ^T = I$ and R is an upper triangular matrix, cf. (Golub and van Loan, 1996; Sauer, 2013a).

[7] Do we hold the camera upside down?

Example 1.2.1 (Intrinsic calibration). We decompose

$$\hat{\boldsymbol{A}} = \begin{pmatrix} \boldsymbol{Q}\boldsymbol{R} & \boldsymbol{y} \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{Q} & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{R} & \boldsymbol{Q}^T\boldsymbol{y} \\ 0 & 1 \end{pmatrix} := \begin{pmatrix} \boldsymbol{Q} & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{R} & \boldsymbol{y}' \\ 0 & 1 \end{pmatrix}$$

and suppose that the extrinsic part of the calibration has been done, hence $\boldsymbol{Q}$ is known. Now we take homogeneous points $\hat{\boldsymbol{x}}_j \in \mathbb{R}^4$, $j = 1, \ldots, n$, on a so called **calibration target** in the ambient space which should be mapped to[8]

$$P_d\hat{\boldsymbol{x}}_j = \begin{pmatrix} -\frac{d}{x_{3,j}}x_{1,j} \\ -\frac{d}{x_{3,j}}x_{2,j} \\ 1 \end{pmatrix}, \qquad j = 1, \ldots, n.$$

where the 1 in the last coordinate is due to the fact that we consider the $-d$ plane now as an independent plane. The measured points, however, are $\hat{\boldsymbol{z}}_j$, $j = 1, \ldots, n$, and we must determine $\hat{\boldsymbol{A}}$ in such a way that $\hat{\boldsymbol{A}}\boldsymbol{z}_j \approx P_d\hat{\boldsymbol{x}}_j$. To that end, we compute

$$\hat{\boldsymbol{A}}\boldsymbol{z}_j = \begin{pmatrix} \boldsymbol{Q} & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{R} & \boldsymbol{y}' \\ 0 & 1 \end{pmatrix} \begin{pmatrix} z_{1,j} \\ z_{2,j} \\ 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{Q} & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{R}\boldsymbol{z}_j + \boldsymbol{y}' \\ 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{Q} & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \alpha z_{1,j} + \gamma z_{2,j} + y_1' \\ \beta z_{2,j} + y_2' \\ 1 \end{pmatrix}$$

This means that we must match the vectors

$$\begin{pmatrix} \alpha z_{1,j} + \gamma z_{2,j} + y_1' \\ \beta z_{2,j} + y_2' \\ 1 \end{pmatrix}, \qquad j = 1, \ldots, n,$$

as good as possible to the vectors

$$\begin{pmatrix} \boldsymbol{Q} & 0 \\ 0 & 1 \end{pmatrix} P_d\hat{\boldsymbol{x}}_j =: \begin{pmatrix} \boldsymbol{v}_j \\ 1 \end{pmatrix} = \begin{pmatrix} v_{j,1} \\ v_{j,2} \\ 1 \end{pmatrix}.$$

Their difference is the vector

$$\begin{pmatrix} \alpha z_{1,j} + \gamma z_{2,j} + y_1' - v_{j,1} \\ \beta z_{2,j} + y_2' - v_{j,2} \\ 0 \end{pmatrix}$$

and the fit is perfect if α, β, γ and $\boldsymbol{y}'$ solve the overdetermined linear system

$$\underbrace{\begin{pmatrix} z_{1,1} & 0 & z_{2,1} & 1 & 0 \\ 0 & z_{2,1} & 0 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ z_{1,n} & 0 & z_{2,n} & 1 & 0 \\ 0 & z_{2,n} & 0 & 0 & 1 \end{pmatrix}}_{=:\boldsymbol{B}} \begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \boldsymbol{y}' \end{pmatrix} = \begin{pmatrix} \boldsymbol{v}_1 \\ \vdots \\ \boldsymbol{v}_n \end{pmatrix},$$

[8] The image plane is two dimensional, hence the homogeneous coordinates are three dimensional

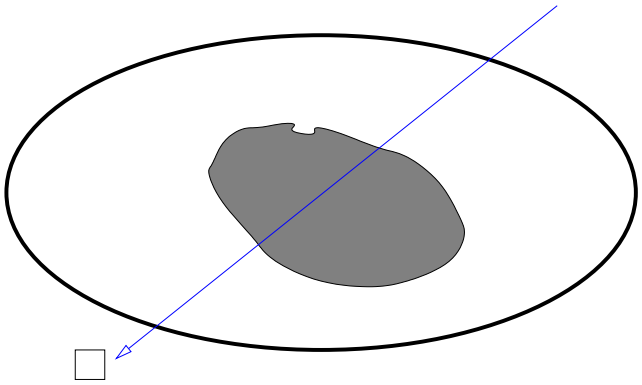


Figure 1.3.1: The beam is sent through the material and the absorbtion is eventually a line integral along this beam.

in the sense that we minimize the euclidean norm of the error which is equivalent to solving

$$\boldsymbol{B}^T\boldsymbol{B}\begin{pmatrix}\alpha\\ \beta\\ \gamma\\ \boldsymbol{y}'\end{pmatrix} = \boldsymbol{B}^T\begin{pmatrix}\boldsymbol{v}_1\\ \vdots\\ \boldsymbol{v}_n\end{pmatrix}.$$

This is a standard task in Numerical Linear Algebra, cf. (Golub and van Loan, 1996).

1.3 Computed Tomography

The **Radon transform** is an **integral transform** of high relevance in many applications from medical imaging to nondestructive testing, since it is the basis of **Computed Tomography**. The main idea is that an X-ray beam is sent through inhomogeneous material, where part of the ray is absorbed and the rest continues its trip through the material. On the other side of the object the remaining energy of the beam is measured, see Fig. 1.3.1. Ignoring physical effects like refraction and beam hardening, we denote the local absorption rate of the material[9] by $f(x)$, $x \in \mathbb{R}^2$, and by $I(x)$ the intensity of the beam at position x. Then we have, according to (Olafsson and Quinto, 2006), for two points x, $x+\delta$ on the ray that

$$I(x+\delta) - I(x) \approx -f(x)\,\delta\,I(x), \qquad \text{d.h.} \qquad I(x+\delta) \approx I(x)\,(1 - f(x)\,\delta)\,.$$

[9]For monochromatic X-rays this is proportional to the density of the material, see (Olafsson and Quinto, 2006, p. 2).

To turn this multiplicative relationship into an additive one, we apply the logarithm to both sides and use a Taylor expansion[10] with respect to δ at $\delta = 0$, yielding

$$\begin{aligned} \log I(x+\delta) &= \log I(x) + \log\left(1 - f(x)\,\delta\right) = \log I(x) - f(x)\,\delta + O\left(\delta^2\right) \\ &\approx \log I(x) - f(x)\,\delta. \end{aligned}$$

Now we decompost the ray from the source x_S to the detector x_D into $N+1$ pieces of length δ and obtain

$$\begin{aligned} \log \frac{I(x_S)}{I(x_D)} &= -\left(\log I(x_D) - \log I(x_S)\right) \\ &= -\sum_{j=0}^{N} \log I(x_S + (j+1)\delta) - \log I(x_S + j\delta) \approx \sum_{j=0}^{n} f(x_S + j\delta)\ \delta, \end{aligned}$$

which is nothing but a **quadrature formula** for the **line integral**

$$\int_{[x_S, x_D]} f(x)\,dx := \|x_D - x_S\| \int_0^1 f\left(\lambda x_S + (1-\lambda) x_D\right)\,d\lambda \tag{1.3.1}$$

normalized such that $\int_{[x_S,x_D]} 1\,dx = \|x_D - x_S\|$.

Remark 1.3.1. The line integral in (1.3.1) has to be taken with care, as it is a d*distributional* definition and an integral over a set of measure zero in $\mathbb{R}^2$. This implies some theoretical subtleties that have to be taken care of when one is interested in mathematical precision and correctness, cf. (Natterer, 1986; Natterer and Wübbeling, 2001).

This is almost the **Radon transform** which has been introduced by J. Radon in 1917 for purely mathematical reasons. Given a function f, the transform computes all possible line integrals of the function. This makes it necessary to paramterize lines in $\mathbb{R}^2$ which can be done by a directional vector y, $\|y\| = 1$, and the signed distance s of the line[11] to the origin. The line is then

$$L = \left\{x \in \mathbb{R}^2 : y^T x = s\right\}.$$

The set $\mathscr{L}$ of all lines $\mathbb{R}^2$ can be identified with $\mathbb{R} \times \mathbb{S}^2$, where $\mathbb{S}^2 = \{y \ : \ \|y\| = 1\}$ is the two dimensional unit ball, i.e., the unit circle.

Exercise 1.3.1 When do two lines $L, L' \in \mathscr{L}$ intersect? ◊

[10] The Taylor expansion of $\log(1 - ax)$ at $x = 0$ is

$$\log(1-ax) = -\sum_{j=1}^{\infty} \left.\frac{a^j}{(1-ay)}\right|_{y=0} x^j = -\sum_{j=1}^{\infty} (ax)^j.$$

[11] Some people insist on calling it a *straight line* though I am not aware of mathematical work that considers non-straight lines, at least not in a euclidean context.

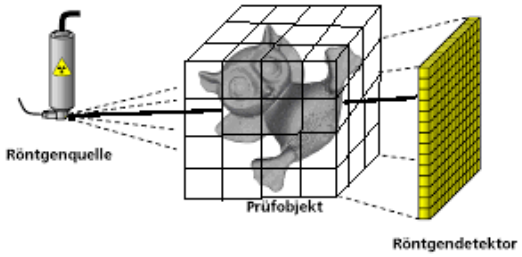


Figure 1.3.2: Schematic representation of an industrial CT scan. The object is rotated to obtain measurements from different angles.

Definition 1.3.2 (Line integral). The **line integral** along a line $L = (s, y) \in \mathscr{L}$ is defined as

$$\int_L f(x)\,dx := \int_{\mathbb{R}} f\left(sy^\perp + ty\right)\,dt, \qquad y^\perp = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}. \tag{1.3.2}$$

Exercise 1.3.2 Show that L passes through the point $s\,y^\perp$. ◊

Definition 1.3.3 (Radon transform). The **Radon transform** R associates to any any function[12] $f : \mathbb{R}^2 \to \mathbb{R}$ the line integral

$$Rf(L) := \int_L f(x)\,dx.$$

For a fixed f, $Rf : \mathscr{L} \to \mathbb{R}$ maps the set of all lines in $\mathbb{R}^2$ to $\mathbb{R}$.

In Computed Tomography an X-ray source emits radition that passes throgh the object and the remaining intensity is recorded, either by a single moving detector, a line detector or even a flat panel, see Fig. 1.3.2 for a schematic representation and Fig. 1.3.3 for a real world scanner. The X-ray beams are recorded on the flat screen and give values of $Rf(L)$ for the lines connecting the source and the pixel on the screen. In contrast to medical CT where usually the soruce and the detector rotate around the patient, the object is rotated in industrial CT.

Since tomography computes the Radon transform along several line, the mathematical problem of reconstruction of the object is

*Compute the **inverse** Radon transform of a given function*

This problem can be approached mathematically *and* numerically. The good news due to Radon is that the function f can be reconstructed from $Rf(\mathscr{L})$ under some mild assumptions on f, but the bad news is that this theoretical result requires the knowledge of $Rf(L)$ for *all* $L \in \mathscr{L}$. And the explicit formula for the inverse Radon transform is highly unstable and numerically infeasible.

On the other hand, any realistic measurement only provides us with $Rf(L_j)$, $j = 1, \ldots, N$, for only a *finite* number of lines which necessitates the development

[12] It is not really "any" function, only those for which the line integrals are well-defined. Here we encounter the mathematical subtleties, as classical function spaces like $L_1(\mathbb{R}^2)$ do not have this property as integrable functions are only defined up to sets of mesaure zero which lines are.

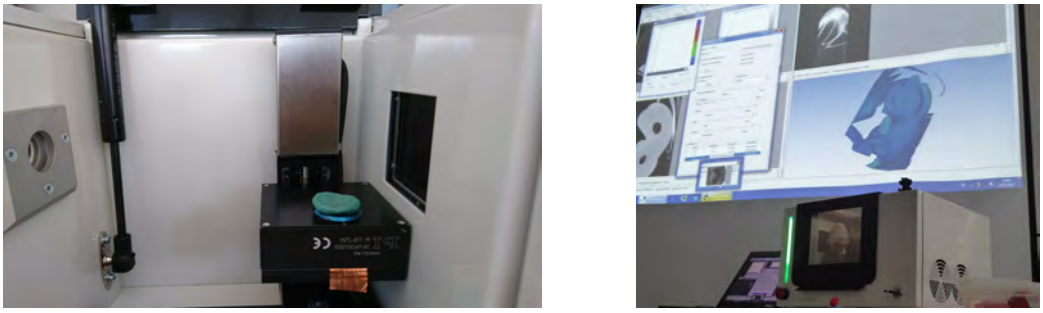


Figure 1.3.3: Inside view of a protable CT device provided by Fraunhofer IIS/EZRT, where one can see the aperture of the X-ray source, the rotation table and the scree *(left, flipped to be aligned with Fig. 1.3.2).* A scan of a "Kinder Überraschung" with the figure inside displayed on the screen in the background *(right).*

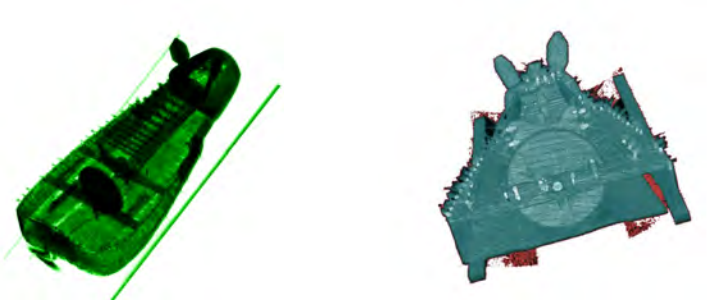


Figure 1.3.4: Scans of a musical intrument (hurdy gurdy) with clearly visible metal artifacts. To make it more interesting, the gray value of the artifacts is close to the density of the wood and they cannot be removed by simple inspection of the gray values.

and application of numerical methods for reconstruction. Of course, such methods exist, and even with a lot of variants suitable to various scan scenarios, but there are still problems caused by the physical realities that usually result in unwanted artifacts in the reconstructions, see

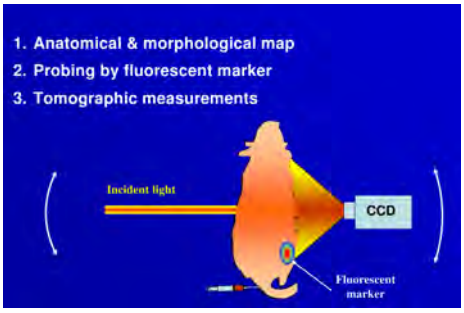
1.4 Indirect Measurement & Inverse Problems

Tomography was already an example of an idirect measurement: the original process does not measure the object of interest, namely the function f, but only the line integrals $Rf(L)$ for some lines L. One has to add a reconstruction process in order to obtain what one is interested in.

Let us formalize that. Instead of measuring a quantity x, we only know $F(x)$ for some function F that is usually not *injective* since the measurement often include a loss of information. In other words, there exist $x \neq x'$ such that $F(x) = F(x)$. The simplest and even most common case of such a problem is an **underdetermined system** of the form

$$Ax = y, \qquad A \in \mathbb{R}^{m\times n}, \quad m < n. \tag{1.4.1}$$

Example 1.4.1 (Optical tomography)**.** In Optical Tomography a fluorescent marker is activated by Laser light, see Fig. 1.4.1, and the emitted light is measured by op-



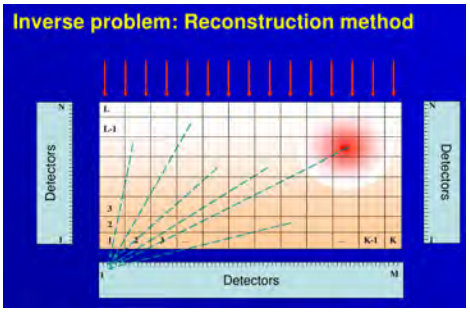


Figure 1.4.1: Optical tomography where a fluorescent marker is activated by a laser and the emitted light is recorded *(left)*. The physical model can be used for the "forward projection" of the light distribution on the detectors *(right)*.

tical sensor surrounding the object. To model the problem, we place a grid on the measured region and model the light distribution between neighboring grid cells as well as between the boundary cells and the detectors. If we denote by $x \in \mathbb{R}^{MN}$ the intensity with in the MN cells of the $M \times N$ grid and by $y \in \mathbb{R}^K$ the values measured at the K detectors, the light distribution can be modeled by a linear system

$$y = Ax, \qquad A \in \mathbb{R}^{K \times MN}.$$

For sufficiently high resolution, this will be underdetermined since the number of cells increases quadratically with the resolution, the number of detectors only linearly.

1.4.1 Solving by Optimizing

To solve such problems, we realized that the system $Ax = b$ from (1.4.1) usually has a lot of solutions, more precisely, an $n-m$ dimensional subspace, which means that we have to choose a solution, ideally a good one, so why not the best? We formulate this requirement as the **optimization problem**

$$\min_x \gamma(x), \qquad Ax = b, \tag{1.4.2}$$

for some **quality functional** $\gamma : \mathbb{R}^n \to \mathbb{R}$, which should at least be bounded from below[13], where we can assume that the bound is zero, hence $\gamma : \mathbb{R}^n \to \mathbb{R}_+$. Classical examples for γ are the p-norms, $\gamma(x) = \|x\|_p$, $1 \le p \le \infty$, or

$$\gamma(x) = \frac{1}{2} x^T B x, \qquad B \text{ strictly positive definite}^{14}. \tag{1.4.3}$$

The problem in (1.4.3) is so classical that we want to have a brief look at how it is solved. To that end, we recall a *necessary* condition for the existence of an extremum of $f : \mathbb{R}^N \to \mathbb{R}$ subject to $g(x) = 0$, $g : \mathbb{R}^n \to \mathbb{R}^m$, namely the existence of **Lagrange multipliers** $\lambda \in \mathbb{R}^m$ for which

$$\nabla f(x) - \nabla g(x)\, \lambda = 0, \qquad \nabla g := \left(\frac{\partial}{\partial x_j} g_k \;:\; \begin{matrix} j = 1, \dots, n \\ k = 1, \dots, m \end{matrix} \right), \tag{1.4.4}$$

[13] Otherwise we cannot guarantee the existence of a minimum.

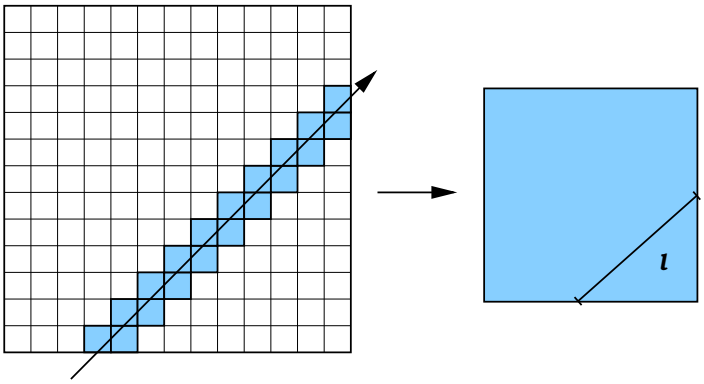


Figure 1.4.2: Tracking a ray through the ROI. Note that the ray meets relatively few grid elements and the length of the intersection is the clipped part of the line *(right)*

holds. You can learn this either in lectures on Anlysis, cf. (Heuser, 1983), or in Optimization, cf. (Sauer, 2013b; Spellucci, 1993). For $f(x) = \frac{1}{2}x^T Bx$ and $g(x) = Ax - b$ we get $\nabla f(x) = Bx$ and $\nabla g(x) = A^T$, hence our solution of (1.4.2) wiht γ as in (1.4.3) is any solution of

$$\begin{aligned} Bx - A^T\lambda &= 0 \\ Ax &= b \end{aligned} \qquad \Leftrightarrow \qquad \begin{pmatrix} B & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} x \\ -\lambda \end{pmatrix} = \begin{pmatrix} 0 \\ b \end{pmatrix}.$$

The convexity of $x \mapsto x^T Bx$ ensures that such a solution exists and therefore we can solve our original problem by solving the square and symmetric[15] $m+n \times m+n$ system

$$Cx = d, \qquad C = \begin{pmatrix} B & A^T \\ A & 0 \end{pmatrix}, \qquad d = \begin{pmatrix} 0 \\ b \end{pmatrix} \tag{1.4.5}$$

and forget about the first n entries of the solution since the values of λ are of no interest.

1.4.2 Tomography once more

In pixellated form, Computed Tomography is also an inverse problem, often know as Algebraic Reconstruction Technique (ART). To that end, we again discretize our region of interest in $\mathbb{R}^2$ which, for simplicity, we assume to be the unit square $D = [0,1]^2$ and decompose it into the mn subrectangles

$$D_{jk} := \left[\frac{j-1}{m}, \frac{j}{m}\right] \times \left[\frac{k-1}{n}, \frac{k}{n}\right], \qquad j = 1, \ldots, m,\ k = 1, \ldots, n,$$

and for each ray passing D we determine which of the subrectangels D_{jk} intersect with this ray and how long this intersection is, see Fig. 1.4.2. This is a simple clipping problem known from computer graphics. Let ℓ_{jk} be the length of the intersections with $\ell_{jk} = 0$ if $\ell \cap D_{jk} = \emptyset$, and denote by p_{jk} the *pixel value*, i.e., the local absorbtion. The the complete absorption along the ray or the discrete

[15] This allow for special and more efficient methods.

approximation for the line integral is given as

$$y = \sum_{j,k=1}^{m,n} \ell_{jk}\, p_{jk}. \tag{1.4.6}$$

To obtain a more familiar way of notation, we vectorize the objects in (1.4.6) as

$$\mathbb{R}^{mn} \ni \boldsymbol{l} := \left(\ell_{n(j-1)+k} \;:\; j,k\right) \qquad \text{and} \qquad \mathbb{R}^{mn} \ni \boldsymbol{p} := \left(p_{n(j-1)+k} \;:\; j,k\right),$$

so that (1.4.6) takes the form $y = \boldsymbol{l}^T \boldsymbol{p}$. For N rays we then get the equations

$$y_j = \boldsymbol{l}_j^T \boldsymbol{p}, \quad j = 1,\dots,N, \qquad \text{hence} \qquad \boldsymbol{y} = \boldsymbol{L}\boldsymbol{p}, \quad \boldsymbol{L} = \begin{pmatrix} \boldsymbol{l}_1^T \\ \vdots \\ \boldsymbol{l}_N^T \end{pmatrix} \in \mathbb{R}^{N\times nm}, \tag{1.4.7}$$

which gives a usually underdetermined system of equations.

Example 1.4.2. Let us consider the simplest possible example, namely the four pixel system

a	b
c	d

and let us shoot two horizontal and two vertical rays through it, giving

$$\begin{aligned} \frac{1}{2}(a+b) &= y_1 \\ \frac{1}{2}(c+d) &= y_2 \\ \frac{1}{2}(a+c) &= y_3 \\ \frac{1}{2}(b+d) &= y_4 \end{aligned}$$

yielding the linear system

$$\underbrace{\frac{1}{2}\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}}_{=:A} \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = y,$$

which is not solvable except when $y_1 + y_2 = y_3 + y_4$. A precise measurement of the rays should have this property, and then we can write $y_4 = y_3 - y_1 - y_2$, hence y_4 is redundant and we can reconstruct a one-parameter family of solutions from the first three equations above. But after a small perturbation of the measurements, there is no choice any more for a, b, c, d that is really consistent with the measurements.

Again this can be solved by minimizing the quadratic error, i.e., by solving

$$\frac{1}{4}A^TAx = A^Ty, \qquad x = \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix},$$

or, explicitly

$$\frac{1}{4}\begin{pmatrix} 2 & 1 & 1 & 0 \\ 1 & 2 & 0 & 1 \\ 1 & 0 & 2 & 1 \\ 0 & 1 & 1 & 2 \end{pmatrix}\begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} y_1 + y_3 \\ y_1 + y_4 \\ y_2 + y_3 \\ y_2 + y_4 \end{pmatrix}$$

that can be shown to have a one dimensional space of solutions from which we can pick, for example, the minimal norm one.

Exercise 1.4.1 Show that the matrices A and A^TA have rank 3 and form the linear system that computes the minimal norm solution. ◊

Exercise 1.4.2 Implement this method in `Matlab/Octave` and test its reconstruction quality. ◊

1.5 Summary

This short and fully incomplete overview over imaging problems shows that there are lost of image acquisition methods that result in completely different image models and also completely different mathematical challenges: in photography one may have to compensate distortions in other imaging modalities we may have to reconstruct inverse problems. Measurements could be corrupted by noise and artifacts and different types of things may have to be detected automatically in images. Therefore, image processing methods will have to be developed depending on the nature and modality of the images and there definitely is no general method or theory that covers everything. This may appear threatening, but it is quite the contrary, it leaves room for creativity, even if it requires a somewhat wider knowledge of methods.

Mathematical Foundations of Signal Processing

2

Reality is software. What does it matter what system it's running on?

(R. Rucker, *Postsingular*)

In the end, there is no choice: the methods use in Image Processing are of a mathematical nature and the more powerful they are, the more substantial is the math. Therefore it does not hurt to meet with some mathematical basics in Signal and Image Processing.

2.1 Signal Spaces

For us a **signal** is a mapping $f : D \to \mathbb{R}$ sein, where $D \subseteq \mathbb{R}^d$ for some d with maybe an emphasis on $d = 1$. Particular cases for D are in the case $d = 1$

1. $D = \mathbb{R}$: an unbounded signal defined on a continuum, no beginning, no end, no gaps.

2. $D = [a, b]$: a time-limited continuum signal. By an affine rescaling, we can always ensure that $a = 0$ and $b = 2\pi$. If, in addition $f(0) = f(2\pi)$, we can extend the signal periodically on $\mathbb{R}$ by setting

$$f(x + 2k\pi) := f(x), \qquad x \in [0, 2\pi], \quad k \in \mathbb{Z}.$$

   In the same way, we can consider any 2π periodic functions as functions on the **torus** $\mathbb{T} = \mathbb{R}/2\pi\mathbb{Z} \simeq [0, 2\pi)$. Periodic functions are "half infite" as they have no boundary again, though they live on a compact domain.

3. $D = \mathbb{Z}$: a disrecte signal, in other words, a biinfinite sequence. We will nevertheless write sequences as functions as this gives some nice analogies.

The above definition of a signal as a function is so general that it is almost totally arbitrary and meaningless. We have to get a bit more restrictive and to that end, will define particular signal spaces of functions that are bounded in some sense.

Definition 2.1.1 (Signal spaces). We denote $L(\mathbb{R})$ all functions $f : \mathbb{R} \to \mathbb{R}$ that are at least locally integrable[1] and by $\ell(\mathbb{Z})$ all real sequences, i.e., functions $f : \mathbb{Z} \to \mathbb{R}$, and define the following spaces:

[1] We will not get specific here on what this means. Whenever we formulate integrals and sums we implicitly assume that the underlying objects are such that the integral or sum is well defined, at least if not stated otherwise. This is slightly imprecise but avoids getting lost in details that are, of course, relevant in principle, but not here.

1. **square summable** functions

$$L_2(\mathbb{R}) := \left\{ f \in L(\mathbb{R}) \ : \ \infty > \|f\|_2 := \left(\int_{\mathbb{R}} |f(t)|^2 \, dt \right)^{1/2} \right\}$$

and sequences

$$\ell_2(\mathbb{Z}) := \left\{ c \in \ell(\mathbb{Z}) \ : \ \infty > \|c\|_2 := \left(\sum_{k \in \mathbb{Z}} |c(k)|^2 \right)^{1/2} \right\};$$

these are often called functions and sequences of finite energy is $\|f\|_2$ in many cases.

2. **absolutely summable** functions and sequences, defined by finiteness of the norms

$$\|f\|_1 := \int_{\mathbb{R}} |f(t)| \, dt \qquad \text{and} \qquad \|c\|_1 := \sum_{k \in \mathbb{Z}} |c(k)| \, .$$

3. **bounded** functions and sequences for which the norms

$$\|f\|_\infty := \sup_{t \in \mathbb{R}} |f(t)| \qquad \text{and} \qquad \|c\|_\infty := \sup_{k \in \mathbb{R}} |c(k)| \, .$$

are finite. But be careful: in case of the continuum, the supremum is an **essential supremum** with possible exceptions on sets of measure zero.

4. **finitely supported** functions and sequences for which there exists $N \in \mathbb{N}$ such that

$$\left. \begin{array}{c} \{t \in \mathbb{R} \ : \ f(t) \neq 0\} \\ \{k \in \mathbb{Z} \ : \ c(k) \neq 0\} \end{array} \right\} \subseteq [-N, N] \, ,$$

again up to sets of measure zero in the continuum case. Such function spaces we write as $L_0(\mathbb{R})$ or $\ell_0(\mathbb{Z})$, since in the latter case $\ell_0(\mathbb{Z})$ is the space of all sequences such that

$$\|c\|_0 := \#\{k \in \mathbb{Z} : c(k) \neq 0\}$$

is finite.

Remark 2.1.2. The spaces $L_1(\mathbb{R})$, $L_2(\mathbb{R})$, $L_\infty(\mathbb{R})$ as well as $\ell_1(\mathbb{Z})$, $\ell_2(\mathbb{Z})$ und $\ell_\infty(\mathbb{Z})$ all form a **Banach space**, i.e., they are **complete**[2], and $L_0(\mathbb{R})$ and $\ell_0(\mathbb{Z})$ are dense in $L_1(\mathbb{R})$ and $L_2(\mathbb{R})$ and $\ell_1(\mathbb{Z})$ and $\ell_2(\mathbb{Z})$ respectively, but not $L_\infty(\mathbb{R})$ bzw. $\ell_\infty(\mathbb{Z})$.

Definition 2.1.3 (δ distribution, Dirac pulse)**.** A particular discrete signal is the **pulse** $\delta \in \ell_0(\mathbb{Z})$, defined as

$$\delta(k) = \delta_{0k} = \begin{cases} 1, & k = 0, \\ 0, & k \neq 0. \end{cases}$$

[2]Recall that this means that any Cauchy sequence has a limit in the space. $\mathbb{R}$ is a Banach space, $\mathbb{Q}$ isn't.

This is often considered as a discrete version of the "Dirac function", a "function" that vanishes everywhere except at the origin where it is so large that

$$\int_{\mathbb{R}} f(x)\,\delta(x)\,dx = f(0)$$

for any "reasonable" f. This however, defines only a **distribution** that should not at all be handled like a function.

To switch between continuous and discrete signal spaces, we define the **sampling operator** $S_h : L(\mathbb{R}) \to \ell(\mathbb{Z})$ with stepsize h as

$$(S_h f)(k) := f(hk), \qquad k \in \mathbb{Z}. \tag{2.1.1}$$

Remark 2.1.4. Formally, the operator S_h makes no sense for arbitrary integrable functions since it is determined by the behavior of f on the set $h\mathbb{Z}$ of measure zero. This can be cured by requiring a certain (piecewise) continuity of f or by setting

$$f(hk) = \lim_{\varepsilon\to 0} \frac{1}{2\varepsilon} \int_{-\varepsilon}^{\varepsilon} f(hk+t)\; dt$$

and making hk a **Lebesgue point**[3] see (Akhieser, 1988).

This already leads to a very interesting and fundamental mathematical problem: for which functions f can we invert the sampling process, i.e., for which functions can f be reconstructed from $S_h f$? Obviously, these are the functions for which the discretization is equivalent to the original function or, in other words, the functions that can be losslessly discretized.

2.2 Fourier

The most fundamental mathematical tool for Signal and Image Processing is without doubt the **Fourier transform** of a function f, especially for $f \in L_2$. We will mainly focus on the formal calculus here and not on all the theoretic details for which we refer to (Katznelson, 1976), for example.

2.2.1 Definition & Calculus

Definition 2.2.1. For $f \in L_1(\mathbb{R})$ we define its **Fourier transform** $\hat{f} : \mathbb{R} \to \mathbb{C}$ as

$$\hat{f}(\xi) := f^{\wedge}(\xi) := \int_{\mathbb{R}} f(t)\, e^{-i\xi t}\, dt, \qquad \xi \in \mathbb{R}, \tag{2.2.1}$$

and the Fourier transform of $c \in \ell_1(\mathbb{Z})$ is the discrete counterpart

$$\hat{c}(\xi) := c^{\wedge}(\xi) := \sum_{k\in\mathbb{Z}} c(k)\, e^{-ik\xi}, \qquad \xi \in \mathbb{R}. \tag{2.2.2}$$

[3] This is just to show that there are subtle problems but also a theory to overcome them.

Remark 2.2.2. 1. In the physical or technical interpretation of signals, the Fourier transform describes the contribution of the respective frequency to the signal.

2. The condition $f \in L_1(\mathbb{R})$ guarantees that $\hat{f}(\xi)$ exists for $\xi \in \mathbb{R}$ in the sense that the value is finite:

$$\left|\hat{f}(\xi)\right| \leq \int_{\mathbb{R}} |f(t)| \underbrace{\left|e^{-i\xi t}\right|}_{=1} dt = \|f\|_1 . \tag{2.2.3}$$

   Note however that $f \in L_1(\mathbb{R})$ is only sufficient, but not necessary for the pointwise existence of the Fourier transform.

3. Sometimes the Fourier transform is equipped with a normalizing factor $(2\pi)^{1/2}$, and we will soon see why. When using different literature, or even worse, libraries, one should check very carefully which normalization is applied, otherwise the constant factor can cause nasty trouble.

4. The Fourier transform can also be defined for more general objects, for example for tempered distributions, cf. (Katznelson, 1976; Yosida, 1965). Moreover, it not only exists on $\mathbb{R}$ or $\mathbb{R}^n$ but on general locally compact abelian groups using the associated Haar measure.

5. The two most important group operations[4] on $\mathbb{R}$ are **translation** and **dilation**, that is, the two operators τ_y and σ_h, defined as

$$\tau_y f = f(\cdot + y) \qquad \text{and} \qquad \sigma_h f = f(h\cdot),$$

   respectively.

Definition 2.2.3 (Convolution). For $f, g \in L(\mathbb{R})$ and $c, d \in \ell(\mathbb{Z})$ we define the **convolution**

$$f * g := \int_{\mathbb{R}} f(\cdot - t)\, g(t)\, dt, \qquad * : L(\mathbb{R}) \times L(\mathbb{R}) \to L(\mathbb{R}), \tag{2.2.4}$$

and

$$c * d := \sum_{k\in\mathbb{Z}} c(\cdot - k)\, d(k), \qquad * : \ell(\mathbb{R}) \times \ell(\mathbb{R}) \to \ell(\mathbb{R}), \tag{2.2.5}$$

as well as

$$c * f := f * c := \sum_{k\in\mathbb{Z}} f(\cdot - k)\, c(k), \qquad * : L(\mathbb{R}) \times \ell(\mathbb{R}) \to L(\mathbb{R}), \tag{2.2.6}$$

whenever the sums or integrals exist. The convolution (2.2.4) is called **continuous** the one in (2.2.5) **discrete** and the one in (2.2.6) **semidiscrete**.

Next, we give and prove the fundamental computational rules in Fourier calculus.

[4] This means groups of operations acting on functions defined on $\mathbb{R}$, and these operations have an inverse of the same type.

Theorem 2.2.4 (Properties of the Fourier transform). *The following statements hold true:*

1. *For $f \in L_1(\mathbb{R})$ and $y \in \mathbb{R}$,*

$$\left(\tau_y f\right)^\wedge(\xi) = e^{iy\xi}\,\hat{f}(\xi), \qquad \xi \in \mathbb{R}. \tag{2.2.7}$$

2. *For $f \in L_1(\mathbb{R})$ and $h > 0$*

$$(\sigma_h f)^\wedge(\xi) = \frac{\hat{f}\left(h^{-1}\xi\right)}{h}, \qquad \xi \in \mathbb{R}. \tag{2.2.8}$$

3. *For $f, g \in L_1(\mathbb{R})$ and $c, d \in \ell_1(\mathbb{Z})$ one has that $f * g \in L_1(\mathbb{R})$ and $c * d \in \ell_1(\mathbb{Z})$, respectively. Moreover, for $\xi \in \mathbb{R}$,*

$$(f * g)^\wedge(\xi) = \hat{f}(\xi)\,\hat{g}(\xi), \qquad \textit{bzw.} \qquad (c * d)^\wedge(\xi) = \hat{c}(\xi)\,\hat{d}(\xi). \tag{2.2.9}$$

4. *For $f \in L_1(\mathbb{R})$ and $c \in \ell_1(\mathbb{Z})$ one has that $f * c \in L_1(\mathbb{R})$ and*

$$(f * c)^\wedge(\xi) = \hat{f}(\xi)\,\hat{c}(\xi), \xi \in \mathbb{R}. \tag{2.2.10}$$

5. *If $f, f' \in L_1(\mathbb{R})$ then*

$$\left(\frac{d}{dx} f\right)^\wedge(\xi) = i\xi\,\hat{f}(\xi), \qquad \xi \in \mathbb{R}. \tag{2.2.11}$$

6. *If $f, xf \in L_1(\mathbb{R})$, then $\hat{f}$ is differentiable and*

$$\frac{d}{d\xi}\hat{f}(\xi) = (-ix\,f)^\wedge(\xi), \qquad \xi \in \mathbb{R}. \tag{2.2.12}$$

7. *If $f, \hat{f} \in L_1(\mathbb{R})$, then*

$$f(x) = \left(\hat{f}\right)^\vee(x) := \frac{1}{2\pi}\int_{\mathbb{R}} \hat{f}(\vartheta)\,e^{ix\vartheta}\,d\vartheta. \tag{2.2.13}$$

Definition 2.2.5. The operation $f \mapsto f^\vee := \frac{1}{2\pi} f^\wedge(-\cdot)$ is called **inverse Fourier transform**[5].

Proof: For 1) we compute

$$\begin{aligned}\left(\tau_y f\right)^\wedge(\xi) &= \int_{\mathbb{R}} f(t+y)\,e^{-i\xi t}\,dt = \int_{\mathbb{R}} f(t)\,e^{-i\xi(t-y)}\,dt = e^{iy\xi}\int_{\mathbb{R}} f(t)\,e^{-i\xi t}\,dt \\ &= e^{iy\xi}\,\hat{f}(\xi),\end{aligned}$$

[5]For obvious reasons.

while 2) is done similarly:

$$(\sigma_h f)^\wedge(\xi) = \int_{\mathbb{R}} f(ht)\, e^{-i\xi t}\, dt = \frac{1}{h} \int_{\mathbb{R}} f(t)\, e^{-i(\xi/h)t}\, dt = \frac{\hat{f}\left(\frac{\xi}{h}\right)}{h}.$$

The first statement, 3) follows from

$$\|f * g\|_1 = \int_{\mathbb{R}} \left| \int_{\mathbb{R}} f(t) g(s-t)\, dt \right| ds \le \int_{\mathbb{R}} \int_{\mathbb{R}} |f(t)g(s)|\, dt\, ds = \|f\|_1\, \|g\|_1$$

and

$$\|c * d\|_1 = \sum_{j\in\mathbb{Z}} \left| \sum_{k\in\mathbb{Z}} c(k)\, d(j-k) \right| \le \sum_{j,k\in\mathbb{Z}} |c(k)\, d(j)| = \|c\|_1\, \|d\|_1,$$

respectively, the second, more interesting part is proved as

$$\begin{aligned}(f * g)^\wedge(\xi) &= \int_{\mathbb{R}} \left(\int_{\mathbb{R}} f(s) g(t-s)\, ds \right) e^{-i\xi t} dt \\ &= \int_{\mathbb{R}} \int_{\mathbb{R}} f(s)\, e^{-i\xi s}\, g(t-s)\, e^{-i\xi(t-s)}\, ds dt = \hat{f}(\xi)\, \hat{g}(\xi),\end{aligned}$$

and

$$\begin{aligned}(c * d)^\wedge(\xi) &= \sum_{j\in\mathbb{Z}} \left(\sum_{k\in\mathbb{Z}} c(k)\, d(j-k) \right) e^{-ij\xi} = \sum_{j,k\in\mathbb{Z}} c(k)\, e^{-ik\xi}\, d(j-k)\, e^{-i(j-k)\xi} \\ &= \hat{f}(\xi)\, \hat{g}(\xi).\end{aligned}$$

4) follows from Exercise 2.2.1 and

$$\begin{aligned}(f * c)^\wedge(\xi) &= \int_{\mathbb{R}} \sum_{k\in\mathbb{Z}} f(t-k)\, c(k) e^{-i\xi t}\, dt = \int_{\mathbb{R}} \sum_{k\in\mathbb{Z}} f(t-k)\, e^{-i\xi(t-k)}\, c(k)\, e^{-ik\xi} \\ &= \hat{f}(\xi)\, \hat{c}(\xi),\end{aligned}$$

oder directly by use of (2.2.7). For 5 we use partial integration[6], to obtain

$$(f')^\wedge(\xi) = \int_{\mathbb{R}} \frac{df}{dt}(t) e^{-i\xi t}\, dt = -\int_{\mathbb{R}} f(t) \frac{d}{dt} e^{-i\xi t}\, dt = i\xi \int_{\mathbb{R}} f(t) e^{-i\xi t}\, dt = i\xi\, \hat{f}(\xi).$$

6) is obtained by computing, for $h > 0$, the difference quotient,

$$\frac{\hat{f}(\xi+h) - \hat{f}(\xi)}{h} = \int_{\mathbb{R}} f(t)\, \frac{e^{-i(\xi+h)t} - e^{-i\xi t}}{h}\, dt = \int_{\mathbb{R}} f(t)\, e^{-i\xi t}\, \frac{e^{-iht} - 1}{h}\, dt;$$

the integral exists since $xf \in L_1(\mathbb{R})$ and

$$\lim_{h\to 0} \frac{e^{-iht} - 1}{h} = \lim_{h\to 0} (-it)\, e^{-iht} = -it$$

[6] This is justified by the fact that for $f \in L_1(\mathbb{R})$ one always has $\lim_{x\to\pm\infty} |f(x)| = 0$ and that compactly supported functions are dense in $L_1(\mathbb{R})$ with respect to the norm $\|\cdot\|_1$.

implies (2.2.12). The proof of 7) is a bit more complex and uses the **Fejer kernel**

$$F_\lambda := \lambda F(\lambda \cdot), \quad \lambda > 0, \qquad F(x) := \frac{1}{2\pi} \int_{-1}^{1} (1 - |t|)\ e^{ixt}\, dt, \quad x \in \mathbb{R},$$

which has the property that for any $f \in L_1(\mathbb{R})$

$$\lim_{\lambda \to \infty} \|f - f * F_\lambda\| = 0, \tag{2.2.14}$$

see (Katznelson, 1976, S. 124–126), hence $f * F_\lambda \to f$ pointwise almost everywhere[7]. Since we consider a limit $\lambda \to \infty$, we can assume without loss of generality that $\lambda > 1$ and get for $x \in \mathbb{R}$ that

$$\begin{aligned}
f * F_\lambda(x) &= \frac{1}{2\pi} \int_{\mathbb{R}} f(t) \left(\lambda \int_{-1}^{1} (1 - |\vartheta|)\, e^{i(x-t)\lambda\vartheta}\, d\vartheta \right) dt \\
&= \frac{1}{2\pi} \int_{\mathbb{R}} f(t) \int_{-\lambda}^{\lambda} \left(1 - \frac{|\vartheta|}{\lambda}\right) e^{i(x-t)\vartheta}\, d\vartheta\, dt \\
&= \frac{1}{2\pi} \int_{-\lambda}^{\lambda} \left(1 - \frac{|\vartheta|}{\lambda}\right) \underbrace{\int_{\mathbb{R}} f(t)\ e^{-it\vartheta}\, dt}_{=\hat{f}(\vartheta)}\ e^{ix\vartheta}\, d\vartheta \\
&= \frac{1}{2\pi} \int_{-\lambda}^{\lambda} \left(1 - \frac{|\vartheta|}{\lambda}\right) \hat{f}(\vartheta)\, e^{ix\vartheta}\, d\vartheta \\
&= \underbrace{\frac{1}{2\pi} \int\limits_{0 \le |\vartheta| \le \sqrt{\lambda}} \overbrace{\left(1 - \frac{|\vartheta|}{\lambda}\right)}^{1 - 1/\sqrt{\lambda} \le \cdot \le 1} \hat{f}(\vartheta)\, e^{ix\vartheta}\, d\vartheta}_{\to \frac{1}{2\pi} \int_{\mathbb{R}} \hat{f}(\vartheta)\, e^{ix\vartheta}\, d\vartheta} + \underbrace{\frac{1}{2\pi} \int\limits_{\sqrt{\lambda} \le |\vartheta| \le \lambda} \left(1 - \frac{|\vartheta|}{\lambda}\right) \hat{f}(\vartheta)\, e^{ix\vartheta}\, d\vartheta}_{\to 0},
\end{aligned}$$

at least in the limit $\lambda \to \infty$. The second integral tends to 0, since it can be estimated by

$$\begin{aligned}
&\left| \int\limits_{\sqrt{\lambda} \le |\vartheta| \le \lambda} \left(1 - \frac{|\vartheta|}{\lambda}\right) \hat{f}(\vartheta)\, e^{ix\vartheta}\, d\vartheta \right| \\
&\quad \le \int\limits_{\sqrt{\lambda} \le |\vartheta| \le \lambda} \underbrace{\left|1 - \frac{|\vartheta|}{\lambda}\right|}_{\le 1} \left|\hat{f}(\vartheta)\right| \underbrace{\left|e^{ix\vartheta}\right|}_{=1} d\vartheta \le \int\limits_{\sqrt{\lambda} \le |\vartheta| \le \lambda} \left|\hat{f}(\vartheta)\right| d\vartheta \le \int\limits_{\sqrt{\lambda} \le |\vartheta|} \left|\hat{f}(\vartheta)\right| d\vartheta,
\end{aligned}$$

which converges to zero for $\hat{f} \in L_1(\mathbb{R})$ as $\lambda \to \infty$. □

Remark 2.2.6. One could prove the inverse Fourier transform in naive way by inserting the terms, exchanging the integrals

$$\hat{f}^\vee(x) = \frac{1}{2\pi} \int_{\mathbb{R}} \int_{\mathbb{R}} f(t) e^{-i\theta t} e^{ix\theta}\, dt d\theta = \frac{1}{2\pi} \int_{\mathbb{R}} f(t) \left(\int_{\mathbb{R}} e^{i\theta(t-x)} d\theta \right) dt$$

[7] At least for a subsequence, see (Forster, 1984, S. 96).

and find out that $\int_{\mathbb{R}} e^{i\theta(t-x)} d\theta$ is the "Dirac delta" $\delta(\cdot - x)$ which is infinite at x and zero otherwise, since, for $x \neq 0$,

$$\int_{\mathbb{R}} e^{i\theta x} d\theta = \sum_{k\in\mathbb{Z}} \underbrace{\int_{2k\pi/x}^{2(k+1)\pi/x} \cos\theta x + i \sin\theta x \, d\theta}_{=0} = 0. \tag{2.2.15}$$

As convincing as (2.2.15) might appear, the argument is **wrong**. With the same "idea" we would get for $h > 0$ that

$$\begin{aligned} \int_{\mathbb{R}} e^{i\theta x} d\theta &= \int_{-h/x}^{h/x} e^{i\theta x} d\theta + \sum_{k=0}^{\infty} \int_{(h+2k\pi)/x}^{(h+2(k+1)\pi)/x} e^{i\theta x} d\theta + \sum_{k=0}^{\infty} \int_{-(h2k\pi)/x}^{-(h+2(k+1)\pi)/x} e^{i\theta x} d\theta \\ &= 2 \sin h, \end{aligned}$$

which would assume any value between -2 and 2. The problem is that the function $e^{i\theta x}$ is simply not integrable and therefore the integral is meaningless. This shows that mathematical details matter and this lecture usually cares, even if the details are not mentioned.

Exercise 2.2.1 Show that for $f \in L_1(\mathbb{R})$ and $c \in \ell_1(\mathbb{Z})$ the inequality

$$\|f * c\|_1 \leq \|f\|_1 \, \|c\|_1$$

holds. ◊

Exercise 2.2.2 Prove without using (2.2.11) that for $f, f' \in L_1(\mathbb{R})$, one has $(f')^\wedge(0) = 0$.
Hint: partial integration. ◊

Exercise 2.2.3 Show for $f \in L_1(\mathbb{R})$ and $h \neq 0$ that

$$(\sigma_h f)^\wedge = \frac{\hat{f}\left(h^{-1}\cdot\right)}{|h|} \tag{2.2.8'}.$$

◊

Example 2.2.7. Let us compute a Fourier transform, namely that of the **cardinal B-spline**, defined as $N_0 = \chi := \chi_{[0,1]}$ and $N_j = \chi * N_{j-1}$, $j \in \mathbb{N}$. Hence, the cardinal B-spline N_j of order j is the $(j+1)$-fold **autoconvolution** of the characteristic function. In particular,

$$\hat{N_0}(\xi) = \hat{\chi}(\xi) = \int_{\mathbb{R}} \chi(t)\, e^{-i\xi t}\, dt = \int_0^1 e^{-i\xi t}\, dt = \left.\frac{e^{-i\xi t}}{-i\xi}\right|_{t=0}^{1} = \frac{1 - e^{-i\xi}}{i\xi},$$

and therefore, by (2.2.9),

$$\hat{N_j}(\xi) = (\hat{\chi}(\xi))^{j+1} = \left(\frac{1 - e^{-i\xi}}{i\xi}\right)^{j+1}.$$

Life can be simple.

Exercise 2.2.4 The **centered B-spline** M_j, $j \in \mathbb{N}_0$, is defined as

$$M_j = \underbrace{\chi_{[-1/2,1/2]} * \cdots * \chi_{[-1/2,1/2]}}_{j+1}.$$

Show that

1. these functions are even: $M_j(-x) = M_j(x)$, $x \in \mathbb{R}$,
2. for $j \in \mathbb{N}_0$ one has

$$\hat{M}_j(\xi) = \left(\frac{\sin \xi/2}{\xi/2}\right)^{j+1}, \qquad \xi \in \mathbb{R}.$$

◊

2.2.2 Continuity & Decay

Next, we note that for $f \in L_1(\mathbb{R})$ and $\xi, \eta \in \mathbb{R}$ we have

$$\left|\hat{f}\,(\xi+\eta) - \hat{f}(\xi)\right| \le \int_{\mathbb{R}} |f(t)| \underbrace{\left|e^{-i\xi t}\right|}_{=1} \left|e^{-i\eta t} - 1\right| dt = \int_{\mathbb{R}} |f(t)| \left|e^{-i\eta t} - 1\right| dt,$$

whose right hand side is independent of ξ. Its limit $\eta \to 0$ is zero since for any $\varepsilon > 0$ there exists $N > 0$, such that

$$\int_{|t|>N} |f(t)|\, dt < \varepsilon,$$

while a sufficiently small value[8] η, the function $\left|e^{-i\eta t} - 1\right|$ can be made as small as we want on $[-N, N]$. We can summarize this finding as follows.

Proposition 2.2.8. *If $f \in L_1(\mathbb{R})$, then $\hat{f} \in C_u(\mathbb{R})$, which is the vector space of uniformly continuous and bounded[9] functions.*

The behaviour of $\hat{f}(\xi)$ for $|\xi| \to \infty$ is described in the next result.

Proposition 2.2.9 (**Riemann–Lebesgue Lemma**). *For $f \in L_1(\mathbb{R})$,*

$$\lim_{\xi \to \pm\infty} \hat{f}(\xi) = 0. \tag{2.2.16}$$

Proof: If, in addition $f' \in L_1(\mathbb{R})$, (2.2.16) is an immediate consequence of (2.2.11) and (2.2.3):

$$\|f'\|_1 \ge \left|(f')^{\wedge}(\xi)\right| = |\xi| \left|\hat{f}(\xi)\right|, \qquad \xi \in \mathbb{R},$$

[8] It is a good exercise to determine that value explicitly.
[9] See (2.2.3).

hence $\left|\hat{f}(\xi)\right| \leq \|f'\|_1 / |\xi| \to 0$ for $|\xi| \to \infty$. For arbitrary $f \in L_1(\mathbb{R})$ and differentiable $g \in L_1(\mathbb{R})$ mit[10] such that $\|f - g\|_1 \leq \varepsilon$, we have

$$\|f - g\|_1 \geq \left|\hat{f}(\xi) - \hat{g}(\xi)\right| \geq \left|\hat{f}(\xi)\right| - |\hat{g}(\xi)|,$$

that is

$$\lim_{|\xi|\to\infty} \left|\hat{f}(\xi)\right| \leq \lim_{|\xi|\to\infty} |\hat{g}(\xi)| + \|f - g\|_1 \leq \varepsilon$$

and since ε can be chosen arbitrarily small, the claim follows. □

2.2.3 Square Integrable & Isometry

So far we have defined the Fourier transform for L_1 functions, so what about other spaces, in particular $L_2(\mathbb{R})$ which plays a fundamental role in Signal Processing. Here we make use of the fact that $L_1(\mathbb{R}) \cap L_p(\mathbb{R})$ is a *dense* subset of $L_p(\mathbb{R})$ for $1 < p < \infty$ and that in the L_2 case we have a particularly nice isometry between functions and their Fourier transform.

Theorem 2.2.10 (**Parseval-Plancherel**). *For $f, g \in L_1(\mathbb{R}) \cap L_2(\mathbb{R})$,*

$$\int_{\mathbb{R}} f(t)\, g(t)\, dt = \frac{1}{2\pi} \int_{\mathbb{R}} \hat{f}(\vartheta)\, \overline{\hat{g}(\vartheta)}\, d\vartheta, \tag{2.2.17}$$

in particular, for $f = g$,

$$\|f\|_2 = \frac{1}{\sqrt{2\pi}} \left\|\hat{f}\right\|_2. \tag{2.2.18}$$

*In other words: up to normalization[11] the Fourier transform is an **isometry** on $L_2(\mathbb{R})$.*

Theorem 2.2.10 allows us to transfer the Fourier transform to $L_2(\mathbb{R})$: given $f \in L_2(\mathbb{R})$, we consider the sequence

$$f_n := \chi_{[-n,n]} \cdot f \quad \in L_1(\mathbb{R}) \cap L_2(\mathbb{R}), \qquad n \in \mathbb{N},$$

which converges to f for $n \to \infty$ in the norm $\|\cdot\|_2$. Since

$$\left\|\hat{f}_{n+k} - \hat{f}_n\right\|_2 = \left\|(f_{n+k} - f)^\wedge\right\|_2 = \sqrt{2\pi}\, \|f_{n+k} - f_n\|_2, \qquad k, n \in \mathbb{N},$$

$\hat{f}_n$ is a Cauchy sequence and converges in the (complete) Banach space $L_2(\mathbb{R})$ to a function, that will be defined as $\hat{f}$.

Proof of Theorem 2.2.10: Defining

$$h(x) = \int_{\mathbb{R}} f(t)\, g\,(t - x)\; dt = (f * g(-\cdot))\,(x), \qquad x \in \mathbb{R},$$

[10] $C_{00}^{\infty}(\mathbb{R})$, the space of infinitely differentiable functions with compact support is already a dens subspace of $L_1(\mathbb{R})$. This is one of the fundamental results in L_p theory.

[11] And the normalization that has this property is

$$\hat{f}(\xi) := \frac{1}{\sqrt{2\pi}} \int_{\mathbb{R}} f(x)\, e^{-ix\xi}\, dx.$$

we get that $h(0) = \int fg$. Moreover,

$$\hat{h}(\xi) = \hat{f}(\xi) \underbrace{(g(-\cdot))^\wedge(\xi)}_{=\overline{\hat{g}(\xi)}} = \hat{f}(\xi)\,\overline{\hat{g}(\xi)}, \qquad \xi \in \mathbb{R}.$$

If now f and g are so "nice" that[12] $\hat{f}, \hat{g} \in L_2(\mathbb{R})$, then (2.2.13) yields that

$$\frac{1}{2\pi}\int_{\mathbb{R}} \hat{f}(\vartheta)\,\overline{\hat{g}(\vartheta)}\,d\vartheta = \frac{1}{2\pi}\int_{\mathbb{R}} \hat{h}(\vartheta)\,e^{i0\theta}\,d\vartheta = h(0) = \int_{\mathbb{R}} f(t)\,g(t)\,dt,$$

which is the **Perseval formula**(2.2.17) from which the **Plancherel identity** (2.2.18) follows immediately. □

2.2.4 Fourier Series & Torus

Next, we have a look at Fourier Analysis on the **torus** $\mathbb{T} = \mathbb{R}/2\pi\mathbb{Z}$ where we will encounter the concept of a **Fourier series** which we will connect to the Fourier transform encountered so far. This will become a fundamental building block for the proof of the Shannon sampling theorem. But let us start with the definition.

Definition 2.2.11. For $f \in L_1(\mathbb{T})$, the **Fourier coefficients** are defined as

$$\hat{f}(k) := f_k := \frac{1}{2\pi}\int_{\mathbb{T}} f(t)\,e^{-ikt}\,dt, \qquad k \in \mathbb{Z},$$

and the **Fourier series** of f is

$$\mathscr{F} f := \sum_{k\in\mathbb{Z}} f_k\,e^{ik\cdot}. \tag{2.2.19}$$

Remark 2.2.12. In general, convergence of the Fourier series is a tricky issue as soon as f is not chosen from $L_1(\mathbb{R})$ or $L_2(\mathbb{R})$.

The trigonometric series in (2.2.19) should be familiar to us: if we define for $f \in L_1(\mathbb{T})$ with Fourier coefficients f_k, $k \in \mathbb{Z}$, the sequence

$$c_f(k) = f_k, \qquad k \in \mathbb{Z},$$

then $c_f \in \ell_1(\mathbb{Z})$ and, according to Definition 2.2.1, in particular (2.2.2)), we have that

$$\mathscr{F} f(\xi) = \hat{c}_f(-\xi), \qquad \xi \in \mathbb{R}. \tag{2.2.20}$$

Since, in addition, for $j, k \in \mathbb{Z}$,

$$\int_{\mathbb{T}} e^{-ijt}e^{ikt}\,dt = \int_{-\pi}^{\pi} e^{i(k-j)t}\,dt = \begin{cases} 2\pi, & j = k, \\ \frac{1}{i(k-j)}e^{i(k-j)t}\Big|_{t=-\pi}^{\pi} = 0, & j \neq k, \end{cases} \tag{2.2.21}$$

we get for $k \in \mathbb{Z}$ that

$$c(k) = \frac{1}{2\pi}\int_{\mathbb{T}} \sum_{j\in\mathbb{Z}} c(j)e^{ijt}\,e^{-ikt}\,dt = \frac{1}{2\pi}\int_{\mathbb{T}} \hat{c}(\theta)\,e^{ik\theta}\,d\theta =: (\hat{c})^\vee(k). \tag{2.2.22}$$

In other words: we already found an inverse Fourier transform here.

Exercise 2.2.5 Show that

[12] This is the case whenever $\hat{f}, \hat{g} \in L_2(\mathbb{R})$ which is the case for differentiable f, g.

1. $f \in L_1(\mathbb{T})$ implies $\hat{f} \in \ell_1(\mathbb{Z})$,
2. $c \in \ell_1(\mathbb{Z})$ auch $\hat{c} \in L_1(\mathbb{T})$.

◊

The following theorem connects Fourier series to the Fourier transform.

Theorem 2.2.13 (**Poisson summation formula**). *For $f \in L_1(\mathbb{R})$, we have*

$$\sum_{k\in\mathbb{Z}} f\left(2k\pi\right) = \frac{1}{2\pi}\sum_{k\in\mathbb{Z}} \hat{f}\left(k\right) \qquad \textit{and} \qquad \sum_{k\in\mathbb{Z}} f(k) = \sum_{k\in\mathbb{Z}} \hat{f}\left(2k\pi\right). \tag{2.2.23}$$

Proof: We "wrap" f into

$$g(x) = \sum_{k\in\mathbb{Z}} f\left(x + 2k\pi\right), \qquad x \in \mathbb{R}, \tag{2.2.24}$$

and remark that $g\left(x + 2\pi\right) = g(x)$, hence g is 2π periodic. Moreover,

$$\begin{aligned} \|g\|_{\mathbb{T},1} &= \int_{\mathbb{T}} |g(t)|\, dt = \int_{\mathbb{T}} \left|\sum_{k\in\mathbb{Z}} f\left(t + 2k\pi\right)\right| dt \le \sum_{k\in\mathbb{Z}} \int_0^{2\pi} \left|f\left(t + 2k\pi\right)\right| dt \\ &= \int_{\mathbb{R}} |f\left(t\right)|\, dt = \|f\|_{\mathbb{R},1} \end{aligned}$$

verifies that $g \in L_1(\mathbb{T})$ and well defined. The Fourier coefficients g_k of g have the form

$$\begin{aligned} g_k &= \frac{1}{2\pi}\int_{\mathbb{T}} g(t)\, e^{-ikt}\, dt = \frac{1}{2\pi}\int_{\mathbb{T}} \sum_{\ell\in\mathbb{Z}} f\left(t + 2\ell\pi\right)\, e^{-ikt}\, dt \\ &= \frac{1}{2\pi}\int_{\mathbb{T}} \sum_{\ell\in\mathbb{Z}} f\left(t + 2\ell\pi\right)\, e^{-ikt} \underbrace{e^{-2i\pi k\ell}}_{=1}\, dt = \frac{1}{2\pi}\int_{\mathbb{T}} \sum_{\ell\in\mathbb{Z}} f\left(t + 2\ell\pi\right)\, e^{-ik\left(t+2\ell\pi\right)}\, dt \\ &= \frac{1}{2\pi}\sum_{\ell\in\mathbb{Z}} \int_{2\ell\pi}^{2(\ell+1)\pi} f\left(t\right)\, e^{-ikt}\, dt = \frac{1}{2\pi}\int_{\mathbb{R}} f(t)\, e^{-ikt}\, dt = \frac{1}{2\pi}\,\hat{f}(k) \end{aligned}$$

and, assuming that the partial sums of the Fourier series would converge[13], we find that

$$\frac{1}{2\pi}\sum_{k\in\mathbb{Z}} \hat{f}(k) = \sum_{k\in\mathbb{Z}} g_k \underbrace{e^{ik0}}_{=1} = g(0) = \sum_{k\in\mathbb{Z}} f\left(0 + 2k\pi\right) = \sum_{k\in\mathbb{Z}} f\left(2k\pi\right),$$

which gives the first identity (2.2.23). This identity and (2.2.8) yield that

$$\sum_{k\in\mathbb{Z}} f(k) = \sum_{k\in\mathbb{Z}} \left(\sigma_{(2\pi)^{-1}} f\right)\left(2k\pi\right) = \frac{1}{2\pi}\sum_{k\in\mathbb{Z}} \left(\sigma_{(2\pi)^{-1}} f\right)^{\wedge}\left(k\right) = \sum_{k\in\mathbb{Z}} \hat{f}\left(2k\pi\right).$$

□

[13] Otherwise the proof gets a bit more tricky and involves a summation method like the Féjer kernel.

2.3 The Sampling Theorem

Now we have all the tools necessary to answer the question posed earlier, namely which functions can be uniquely reconstructed from their samples. We begin with two important concepts.

Definition 2.3.1.

1. A function $f \in L_1(\mathbb{R})$ is called **bandlimited** with **bandwidth** T, or T-bandlimited for short, if
$$\hat{f}(\xi) = 0, \qquad \xi \notin [-T, T].$$
2. The **sinus cardinalis** oder **sinc** function[14] is defined as
$$\operatorname{sinc}(x) := \frac{\sin \pi x}{\pi x}, \qquad x \in \mathbb{R}.$$

Remark 2.3.2. Since
$$\operatorname{sinc} 0 = \lim_{x \to 0} \frac{\sin \pi x}{\pi x} = \lim_{x \to 0} \frac{\pi \cos \pi x}{\pi} = \cos 0 = 1$$
we have that
$$(S_1 \operatorname{sinc})(k) = \operatorname{sinc}(k) = \delta(k), \qquad k \in \mathbb{Z}. \tag{2.3.1}$$
This is why the function is called "cardinal": restricted to the integers $\mathbb{Z}$ it takes the values 0 and 0, more precisely, $\operatorname{sinc}|_{\mathbb{Z}} = \delta$.

The next result, due to the "father" of digital Signal Processing, Claude Shannon tells us that bandlimited function can be reconstructed *exactly* from their discrete samples. This is one of the fundamental theorems in Signal Processing.

Theorem 2.3.3 (**Shannon Sampling Theorem**). *If $f \in L_1(\mathbb{R})$ is a T-bandlimited function and $h < h^* = \frac{\pi}{T}$, then*
$$f = (S_h f * \operatorname{sinc}) \left(h^{-1} \cdot\right) = \sum_{k \in \mathbb{Z}} f(hk) \, \frac{\sin \pi (x/h - k)}{\pi (x/h - k)}. \tag{2.3.2}$$

Remark 2.3.4. The relationship between the bandwidth T of the function and the sampling resolution h is a central and fundamental concept, which justifies a few remarks:

1. In the literature one often finds the statement that the **sampling frequency** $1/h$, often called the **Nyquist frequency**, should be leass than *half* of the maximal frequency T, not T as in Theorem 2.3.3. This is due to the fact that T denotes the *width* of the support interval there, i.e., $\hat{f}$ is supported on $\left[-\frac{T}{2}, \frac{T}{2}\right]$ in this case, cf. (Kammeyer and Kroschel, 1998). Such normalization issue are fairly common and show that definitions should be checked carefully when using different sources.

[14] In the engineering literatur the name **si function** is also common.

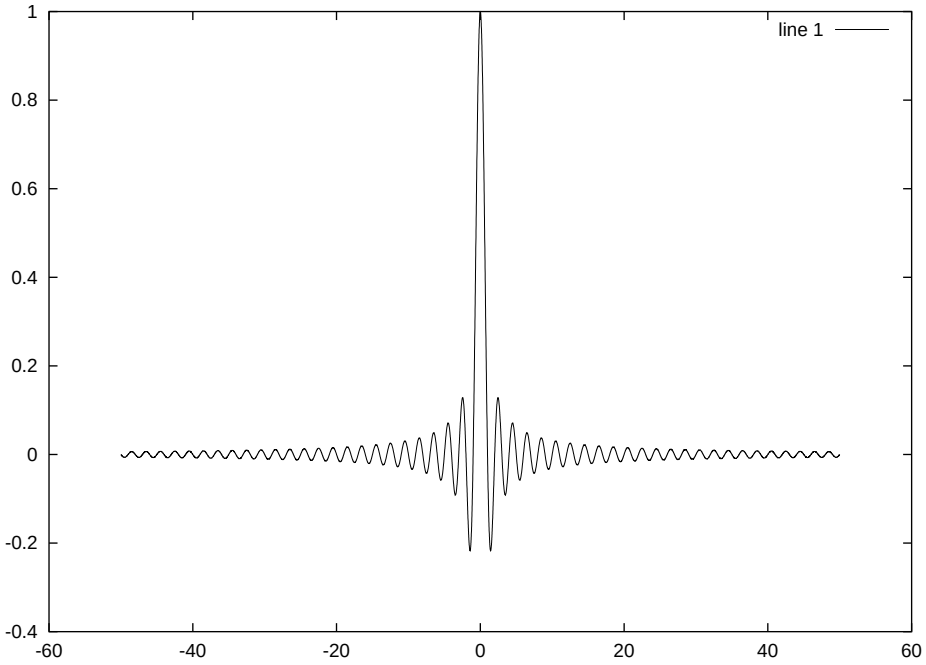


Figure 2.3.1: Plot of the sinc function

2. The sinc function is not really a good way for the *numerical* reconstruction of the signal. It has no finite support and only decays like $\frac{1}{x}$, so that it is not even clear immediately whether it is an L_1 function[15]. This implies that the explicit formula (2.3.2) cannot be recommended in practical applications.

3. In practical applications the sampling rate is often chosen as h^*/k for some integer k which results in a sampling frequency that is the k-fold of what would be necessary. This is called (k-fold) **oversampling**.

4. Many books on Signal Processing from an electrical engineering, for example (Grüningen, 1993) oder (Schüßler, 1992), background provide "proofs" of the sampling theorem that are not reproducible from a strictly mathematical perspective. Often the "Dirac comb", an infinite sum of distributions, is treated as if it were a function. A formally correct proof is the one in (Mallat, 1999), the one here is a modification of the one in (Kammeyer and Kroschel, 1998).

5. The origin of the theorem and the proof is also a story if its own. The first formulation and proof, then in the purely mathematical context of (infinite) interpolation, is due to Whittaker in 1935 (Whittaker, 1935) and was rediscovered[16] by Shannon in 1949 (Shannon, 1949). But Shannon must be credited for realizing its meaning for Signal Processing. And of course, for any Theorem there must be a Russian who discovered it, in this case it is Kotelnikov (Kotelnikov, 1933) even in 1933.

[15] Answer: it is. Exercise: prove that.
[16] Probably independently.

Proof of Theorem 2.3.3: Since f is bandlimited, we have that $\hat{f} \in C_u(\mathbb{R}) \cap L_{00}(\mathbb{R})$, hence also $\hat{f} \in L_1(\mathbb{R})$. For $h > 0$ and $k \in \mathbb{Z}$ the discrete relation (2.2.22) implies that

$$S_h f(k) = \left((S_h f)^\wedge\right)^\vee (k) = \frac{1}{2\pi} \int_{\mathbb{T}} (S_h f)^\wedge (\theta)\, e^{ik\theta}\, d\theta, \tag{2.3.3}$$

as well as

$$\begin{aligned} S_h f(k) &= f(hk) \\ &= \hat{f}^\vee(hk) = \frac{1}{2\pi} \int_{\mathbb{R}} \hat{f}(\theta)\, e^{ihk\theta}\, d\theta = \frac{1}{2\pi} \sum_{j\in\mathbb{Z}} \int_{h^{-1}([-\pi,\pi]+2\pi j)} \hat{f}(\theta)\, e^{ihk\theta}\, d\theta \\ &= \frac{1}{2\pi} \sum_{j\in\mathbb{Z}} h^{-1} \int_{-\pi+2j\pi}^{\pi+2\pi j} \hat{f}\left(h^{-1}\theta\right)\, e^{ik\theta}\, d\theta = \frac{1}{2\pi h} \sum_{j\in\mathbb{Z}} \int_{-\pi}^{\pi} \hat{f}\left(h^{-1}(\theta+2\pi j)\right)\, e^{ik\theta}\, d\theta \\ &= \frac{1}{2\pi h} \int_{-\pi}^{\pi} \left(\sum_{j\in\mathbb{Z}} \hat{f}\left(h^{-1}(\theta+2\pi j)\right) \right) e^{ik\theta} d\theta =: \frac{1}{2\pi} \int_{-\pi}^{\pi} F(\theta)\, e^{ik\theta}, \end{aligned}$$

that is,

$$S_h f(k) = \frac{1}{2\pi} \int_{-\pi}^{\pi} F(\theta)\, e^{ik\theta}, \qquad F = \frac{1}{h} \sum_{j\in\mathbb{Z}} \hat{f}\left(h^{-1}(\cdot + 2\pi j)\right). \tag{2.3.4}$$

The function F satisfies $F \in C(\mathbb{T}) \subset L_1(\mathbb{T})$ since it is obviously 2π periodic and due to f being bandlimited the sum defining $F(\theta)$ has only finitely many nonzero terms for any $\theta \in [0, 2\pi]$. From (2.3.3) and (2.3.4) it therefore follows that

$$0 = \frac{1}{2\pi} \int_{\mathbb{T}} \left((S_h f)^\wedge (\theta) - F(\theta) \right) e^{ik\theta}\, d\theta, \qquad k \in \mathbb{Z},$$

and since the exponential functions $e^{ik\cdot}$, $k \in \mathbb{Z}$, form a complete system of orthonormal functions[17], cf. (Sauer, 2014), we obtain the identity

$$h^{-1} \sum_{j\in\mathbb{Z}} \hat{f}\left(h^{-1}(\theta+2\pi j)\right) = F(\theta) = (S_h f)^\wedge (\theta) = \sum_{j\in\mathbb{Z}} f(hj)\, e^{-ij\theta}, \qquad \theta \in \mathbb{T}. \tag{2.3.5}$$

If h is so small that

$$\begin{aligned} h^{-1}[-\pi,\pi] \supseteq [-T,T] \quad &\Longleftrightarrow \quad [-\pi,\pi] \supseteq [-Th, Th] \quad \Longleftrightarrow \quad Th < \pi \\ &\Longleftrightarrow \quad h < \frac{\pi}{T}, \end{aligned}$$

we get for $j > 0$ and $\theta \in [-\pi,\pi]$ that

$$h^{-1}(\theta + 2\pi j) > \frac{T}{\pi}(-\pi + 2\pi j) \geq T(-1+2j) \geq T,$$

[17] The only function which has all Fourier coefficients equal to zero is the zero function.

and analogously for $j < 0$ that $h^{-1}(\theta + 2\pi j) < -T$. This implies that the sum on the left hand side of (2.3.5) consists only of the term with $j = 0$ and replacing θ by $h\xi$ allows us to conclude that

$$\hat{f}(\xi) = h \sum_{j\in\mathbb{Z}} f(hj)\, e^{-ijh\xi}.$$

Therefore, the assumptions that $T < \pi/h$ and that f is T-bandlimited, yield that

$$\begin{aligned}
f(x) &= \frac{1}{2\pi}\int_{\mathbb{R}} \hat{f}(\theta)\; e^{ix\theta}\; d\theta = \frac{1}{2\pi}\int_{-\pi/h}^{\pi/h} \hat{f}(\theta)\; e^{ix\theta}\; d\theta \\
&= \frac{h}{2\pi}\int_{-\pi/h}^{\pi/h} \sum_{j\in\mathbb{Z}} f(hj)\, e^{i(x-jh)\theta}\, d\theta \\
&= \frac{h}{2\pi}\sum_{j\in\mathbb{Z}} f(hj) \int_{-\pi/h}^{\pi/h} e^{i(x-jh)\theta}\, d\theta = \frac{h}{2\pi}\sum_{j\in\mathbb{Z}} f(hj) \left[\frac{e^{i(x-jh)\theta}}{i(x-jh)}\Big|_{\theta=-\pi/h}^{\pi/h}\right] \\
&= \sum_{j\in\mathbb{Z}} f(hj) \underbrace{\frac{e^{i(x-jh)\pi/h} - e^{-i(x-jh)\pi/h}}{2i}}_{=\sin\pi(x/h-j)} \underbrace{\frac{h}{\pi}\frac{1}{(x-jh)}}_{=(\pi(x/h-j))^{-1}} \\
&= \sum_{j\in\mathbb{Z}} f(hj)\, \frac{\sin\pi\,(x/h-j)}{\pi\,(x/h-j)} = (S_h f * \mathrm{sinc}\,)\,(\cdot/h),
\end{aligned}$$

which gives (2.3.2) and complestes the proof. □

The clou in the proof of Theorem 2.3.3 is to consider the

$$\sum_{k\in\mathbb{Z}} \hat{f}\left(h^{-1}(\theta + 2\pi k)\right) = h\sum_{k\in\mathbb{Z}} f(hk)\; e^{-ik\theta}, \qquad \xi \in \mathbb{T}, \tag{2.3.6}$$

which connects the **periodization** of the Fourier transform of a function with the Fourier transform of the sampled sequence. This identity is independent of whether f is bandlimited or not or whether the sampling is fine enough. If h is so large[18] that the spport of the function $\hat{f}\left(h^{-1}\cdot\right)$ is larger than the interval $[-\pi, \pi]$, then the overlapping parts affect the interior of the support, see Fig. 2.3.2. From this function $\hat{f}$ cannobt be reconstructed any more. But the situation is even worse: the overlay of frequencies that have nothing to do with each other and that are considered modulo 2π, the reconstruction of undersampled signals leads to unwanted effects that are known under the name **aliasing**.

The situation is differen when h is chosen in such a way that the support of $\hat{f}$ is contained *entirely in* $(-\pi, \pi)$ as then there is no overlay and more and 2π periodic Fourier series $(S_h f)^\wedge$ equals the periodization of $\hat{f}$. And what remains in the proof is pure computation.

[18] That is, we missed the Nyquist rate.

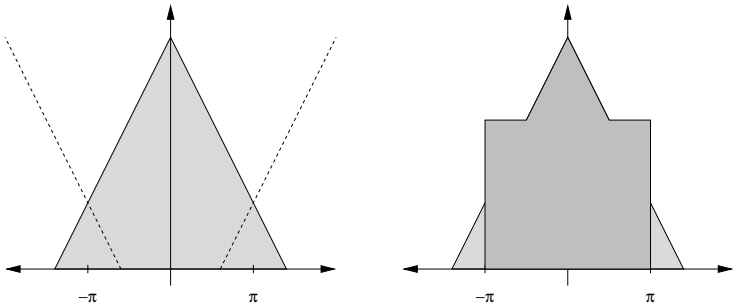


Figure 2.3.2: Periodization of a function whose support is too large. It is impossible to reconstruct the original function *uniquely* from this periodization as there is no unique decomposition of a sum into summands.

2.4 Filters

Next, we consider the fundamental operation in digital Signal Processing, namely the concept of a filter which is used to transform and analyze signals. In full generality, a **filter** is just a mapping from one signal space into another[19]; here, we want to restrict ourselves to the practically most relevant digital filters that map discrete signals to discrete signals.

2.4.1 Definition & Realization

Definition 2.4.1 (Filters and filter types). A **filter** F is an operator that maps $c \in \ell(\mathbb{Z})$ to $Fc \in \ell(\mathbb{Z})$. A filter is called Man nennt so einen Filter

1. **energy preserving** if $F : \ell_2(\mathbb{Z}) \to \ell_2(\mathbb{Z})$ and
$$1 = \|F\|_2 := \sup_{\|c\|_2=1} \|Fc\|_2 .$$
2. **linear filter** if
$$F\left[\alpha c + \beta c'\right] = \alpha\, Fc + \beta\, Fc', \qquad \alpha, \beta \in \mathbb{R}, \quad c, c' \in \ell(\mathbb{Z}).$$
3. **time invariant** if the operator is **stationary**, i.e., if *what* happens is independent of *when* it happens:
$$F\left[c\,(\cdot + k)\right] = \left[Fc\right](\cdot + k)\,, \qquad c \in \ell(\mathbb{Z}), \quad k \in \mathbb{Z}.$$
Using the discrete **translation operator** τ, defined as $\tau c = \tau_1 c = c(\cdot + 1)$, time invariance can be conveniently expressed at the commutative relation
$$\tau F = F\tau \qquad \text{bzw.} \qquad \tau_k F = F\tau_k, \quad \tau_k = \tau^k.$$
4. A filter is called **LTI filter** or just **digital filter**, cf. (Hamming, 1989; Kammeyer and Kroschel, 1998), if it is linear and time invariant.
5. A filter is called **causal** of the result $(Fc)(k)$ ta time k depends only on the values, $c(j)$, $j \le k$, from the past – the filter does not take into account information from the future[20]. A filter that does not consider the past, i.e.,

[19] The two signal spaces may also be the same.
[20] In some way this almost sounds realistic.

where $(Fc)(k)$ depends on $c(j)$, $j \geq k$, is called **anticausal**.

LTI filters are the really nice ones[21] and the only depend on the **impulse response**

$$f := F\delta, \qquad \text{also} \qquad f(k) = [F\delta](k), \quad k \in \mathbb{Z},$$

and even in a very structural way. Since any signal $c \in \ell(\mathbb{Z})$ can be written *formally* as

$$c = \sum_{k\in\mathbb{Z}} c(k)\, \delta(\cdot - k) = \sum_{k\in\mathbb{Z}} c(k)\, \tau_{-k}\delta,$$

linearity and time invariance yield

$$\begin{aligned} Fc &= F\left[\sum_{k\in\mathbb{Z}} c(k)\, \tau_{-k}\delta\right] = \sum_{k\in\mathbb{Z}} c(k)\, F\,[\tau_{-k}\delta] = \sum_{k\in\mathbb{Z}} c(k)\, \tau_{-k}F\delta = \sum_{k\in\mathbb{Z}} c(k)\, \tau_{-k}f \\ &= \sum_{k\in\mathbb{Z}} c(k)\, f\,(\cdot - k) = c * f, \end{aligned}$$

which tells us that an LTI filters always is a *convolution* between the signal and the impulse response. And this immediately tells us due to Theorem 2.2.4 how a filter works in the frequency domain:

$$(Fc)^\wedge(\xi) = (f * c)^\wedge(\xi) = \hat{f}(\xi)\, \hat{c}(\xi). \tag{2.4.1}$$

Here, an LTI filter acts as a simple mulitplication between the Fourier transform of the signal and the Fourier transform of the filter, the so-called **transfer function**. Due to this observation, filters are often designe in the Fourier domain by prescribing their frequency behavior.

Which are the filters that can truely be realized in practice? First of all, they should clearly be causal which mean that for $k < 0$ the pulse δ should give no contribution which means

$$0 = [F\delta](k) = f(k), \qquad k < 0.$$

Noncausal filters can make sense but will have to be realized by buffering some part of the "future" which leads to unavoidable as the result is only available once the future has passed.

Definition 2.4.2. From now on **digital filter** always stands for an LTI filter.

Before we pass to the technical realization, we introduce two more types of filters

Definition 2.4.3 (FIR/IIR)**.** A digital filter is called

1. **FIR filter**[22] if $F\delta \in \ell_0(\mathbb{Z})$, i.e., its impulse response has finite support.
2. *IIR filter*[23] if $F\delta \notin \ell_0(\mathbb{Z})$

[21] And this is the reason why they are used synonymously with "digital filters".
[22] **F**inite **I**mpulse **R**esponse filter.
[23] **I**nfinite **I**mpulse **R**esponse filter.

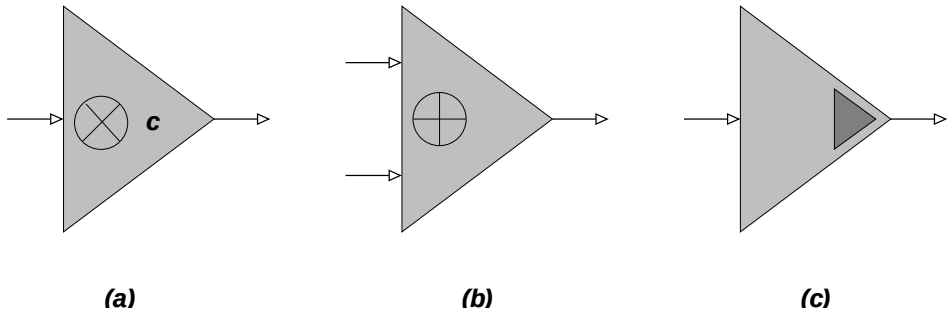


Figure 2.4.1: Symbolic representation of the three building blocks for a digital filter, the multiplier *(a)*, the adder *(b)* and the delay element *(c)*.

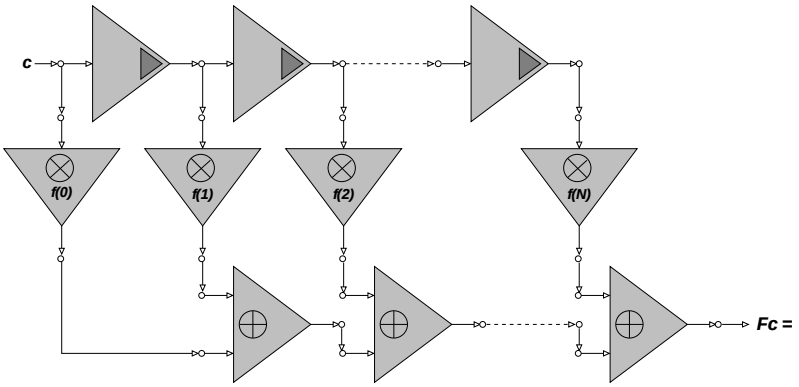


Figure 2.4.2: An FIR filter built as a cascade of the blocks from Fig. 2.4.1. The delay elements take care of the translations, the multipliers of the filter coefficients and adders put everything together.

To realize a digital FIR filter, we need only three components, namely

- a **multiplier** which multiplies a number with a fixed constant c that is encoded into the encoder block,
- an **adder** that adds the two numbers passed to it,
- a **delay** element that saves a number for one time step and then passes it forward into the system.

The symbols for the three elements are depicted in Fig. 2.4.1. Since a delay element, applied to $c \in \ell(\mathbb{Z})$ yields $\tau_{-1}c$ and a chain of k delay elements the signal $\tau_{-k}c$, a causal FIR filter F of the form

$$Fc = f * c = \sum_{k\in\mathbb{Z}} f(k)\, c\,(\cdot - k) = \sum_{k=0}^{N} f(k)\, \tau_{-k}c, \qquad \operatorname{supp} f \subseteq [0, N],$$

can be represented by N of the blocks: the values $\tau_{-k}c$, $k = 0, \ldots, N$, are taken from a chain of $N+1$ delay elements, weighted by multipliers with the values $f(k)$ and then summed up by N adders. The procedure is depicted in Fig. 2.4.2. With the help of M further delay elements, it is possible to realize a a filter whose filter coefficients or **taps** are supported on $[-M, N]$, but the result is delayed by M time units.

Remark 2.4.4 (**Warning**). Any practical FIR filter has a built-in delay related to the size of the support of the filter. This is known as **latency** and can be observed for example in live TV transmissions where the latency in the internal signal processing of the TV sets can cause a difference of a few seconds. Since latency is an issue in real time applications, the length of digital filters is often bounded.

2.4.2 Filter Design

Next we look at a simple example from (Hamming, 1989, Sec. 3.8) that shows how filters are designed in "practice". We calim to be interested in a **symmetric filter**[24] that of course, cannot be causal[25], but it does not matter here. In fact, we will even see that in Image Processing symmetric filters are even quite natural. This FIR filter shall have the property that

$$f(0) = a, \qquad f(\pm 1) = b, \qquad f(\pm 2) = c.$$

Because of the symmetry property $f(k) = f(-k)$, the transfer function is real (valued).

Exercise 2.4.1 Show that if $f \in \ell_0(\mathbb{Z})$ is the real coefficient vector of an FIR filter F, then the transfer function $\hat{f}$ is real if and only if f is symmetric and purely imaginary if f is antisymmetric, i.e., $f(k) = -f(-k)$. ◊

As mentioned before, filter design takes place in the **frequency domain** by requesting properties of the transfer function $\hat{f}$, either seen as a 2π-periodic function on $\mathbb{R}$ or as a function on $\mathbb{T}$. In our example we want to require that the filter fully reproduces low frequencies but blocks high ones. In terms of formulas, this reads as

$$\hat{f}(0) = 1, \qquad \hat{f}(\pi) = 0. \tag{2.4.2}$$

Because of

$$\hat{f}(\xi) = a + b\left(e^{-i\xi} + e^{i\xi}\right) + c\left(e^{-2i\xi} + e^{2i\xi}\right) = a + 2b\cos\xi + 2c\cos 2\xi,$$

(2.4.2) requires that

$$a + 2b + 2c = 1, \qquad \text{und} \qquad a - 2b + 2c = 0,$$

hence $b = \frac{1}{4}$ and $a = \frac{1}{2} - 2c$, yielding

$$\begin{aligned}
\hat{f}(\xi) &= \frac{1}{2} - 2c + \frac{1}{2}\cos\xi + 2c\,\cos 2\xi = \frac{1}{2} - 2c + \frac{1}{2}\cos\xi + 2c\left(2\cos^2\xi - 1\right) \\
&= \frac{1}{2} - 4c + \frac{1}{2}\,\cos\xi + 4c\,\cos^2\xi = 4c\left[\cos^2\xi + \frac{1}{8c}\cos\xi + \frac{1}{8c} - 1\right] \\
&= 4\left(\cos\xi + 1\right)\left(c\,\cos\xi + \frac{1}{8} - c\right).
\end{aligned}$$

The first factor of $\hat{f}$, $\cos\xi + 1$ guarantees the zero at $\xi = \pi$, the second factor degenerates to a constant for $c = 0$, which indicates that the case $c = 0$ will be a special one as then the trigonometric polynomial $\hat{f}$ will have degree 1 and not

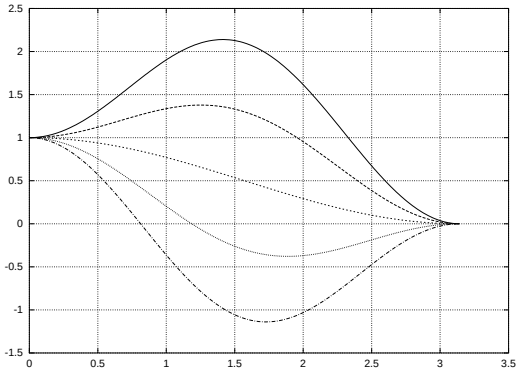


Figure 2.4.3: Examples of transfer functions for $c = -.4, -.2, 0, .2, .4$ (from to down). For $c = 0$, the filter only ranges between 0 and 1 and forms a so-called **sigmoidal function**.

degree 2 as otherwise. Some examples of transfer functions for varying values of c are shown in Fig. 2.4.3.

On the other hand, the desing paramter c can also be chosen in such a way that the resulting filter has further additional properties, for example:

- *preservation of an additional frequency* ω, i.e.,

  $$1 = \hat{f}(\omega) = 4\,(\cos\omega + 1)\left(c\,\cos\omega + \frac{1}{8} - c\right),$$

  hence

  $$c = \left(\frac{1}{4\,(\cos\omega + 1)} - \frac{1}{8}\right)/(\cos\omega - 1) = -\frac{1}{8\,(\cos\omega + 1)},$$

  which is always solvable for $\omega \neq \pi$. Of course, the value $\hat{f}(\omega) = 1$ is impossible for $\omega = \pi$ due to the original requirements (2.4.2).

- *maximal flatness at* $\xi = 0$, which means that as many *derivatives* of $\hat{f}$ as possible vanish at $\xi = 0$, making the filter as similar as possible to a "characteristic function" there. Since

  $$\frac{d}{d\xi}\hat{f}(\xi) = -2b\,\sin\xi - 4c\sin 2\xi$$

  always vanishes at $\xi = 0$, we can require the additional condition

  $$0 = \frac{d^2}{d\xi^2}\hat{f}(0) = -2b\,\cos 0 - 8c\cos 0 \qquad \Rightarrow \qquad c = -\frac{1}{4}\,b = -\frac{1}{16}.$$

- the filter is **balanced**, i.e., it is **antisymmetric** at $\frac{\pi}{2}$:

  $$\hat{f}\left(\frac{\pi}{2} - \xi\right) - \hat{f}\left(\frac{\pi}{2}\right) = \hat{f}\left(\frac{\pi}{2}\right) - \hat{f}\left(\frac{\pi}{2} + \xi\right),$$

[24] That means that $f(k) = f(-k)$ – we will soon see why this is a nice property.
[25] *Exercise:* describe all symmetric causal filters.

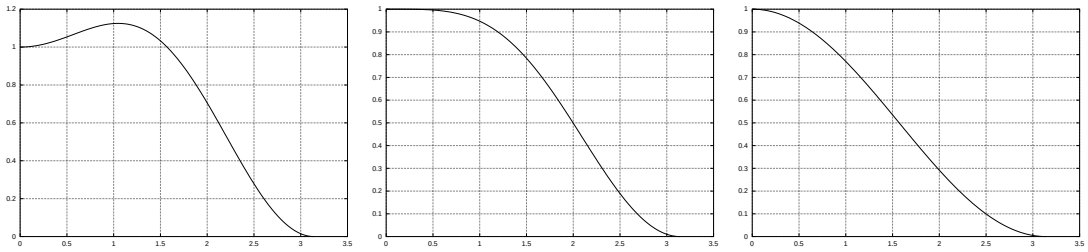


Figure 2.4.4: The three filters for the additional conditions: preservation of the frequency $\frac{\pi}{2}$ *(left*, maximal flatness at 0 *(center)* and balancedness *(right)*.

hence

$$\frac{1}{2}\left[\hat{f}\left(\frac{\pi}{2}-\xi\right)+\hat{f}\left(\frac{\pi}{2}+\xi\right)\right]=\hat{f}\left(\frac{\pi}{2}\right),$$

and in particular for $\xi=\frac{\pi}{2}$,

$$\frac{1}{2}=\hat{f}\left(\frac{\pi}{2}\right)=a+2b\underbrace{\cos\frac{\pi}{2}}_{=0}+2c\underbrace{\cos\pi}_{=-1}\qquad\Rightarrow\qquad a=\frac{1}{2}+2c,$$

which yields together with $a=\frac{1}{2}-2c$ the conditions $a=\frac{1}{2}$ and $c=0$.

The three filters are shown in Fig. 2.4.4.

Of course, this example is very easy, but nevertheless it already shows the general procedure in filter design: the behavior of the filter is described in the *frequency domain* and the coefficients or taps of the filter are chosen in such a way that these properties are met – at least approximately.

Remark 2.4.5 (Filter design & transfer function).

1. The transfer function of a filter, in particual of a nontrivial causal filter, is of the form

$$\hat{f}(\xi)=\sum_{k=-a}^{b}f_k e^{ik\xi}$$

and therefore it normally is a *complex valued* function. Its *real part* defines the weight factor for the respective frequency while the *imaginary part* give the **phase shift** applied to the function by the filter. Since the latter is not really audible[26], one often only prescribes the real part, for example in band pass filters.

2. It may have appeared a little strange already that the frequency range of a filter always ranges between $-\pi$ and π and has no relationship to "real" frequencies measured in **Hertz** (Hz). For example, the range of audible frequencies is usually $10-20000$ Hz and at least audio filters should handle and respect this. The simple solution to this dilemma is the sampling theorem, Theorem 2.3.3: although its Fourier transform is a continuous function, the

[26] At least if it is not varying too much over the frequencies.

filter operates on a *discrete* signal that we can alway assume to be obtained by sampling. And this sampling process has to be so fine, that the Fourier transform of our signal is restricted to the relevant band $[-\pi, \pi]$. And the sampling rate then gives the "real" frequency values for elements of the interval $[-\pi, \pi]$.

The more serious problem, however, is that the transfer functions of FIR filters only form a very small family of functions in $L_2(\mathbb{T})$ or whatever one would like a possible filters. As a reasult, by far not any desired filter can really be represented as an FIR filter.

Definition 2.4.6. A function $f \in C^\infty(\mathbb{T})$ of the form

$$f(x) = \sum_{|k|\le n} f_k\, e^{ikx}, \qquad f_k \in \mathbb{C}, \quad x \in \mathbb{T},$$

is called a **trigonometric polynomial** of degree n.

Exercise 2.4.2 Show that any function of the form

$$f(x) = a_0 + \sum_{k=1}^{n} \left(a_k\, \cos^k x + b_k \sin^k x \right), \qquad a_0, \dots, a_n,\, b_1, \dots, b_n \in \mathbb{C},$$

i.e., any polynomial in $\sin x$ and $\cos x$, is a trigonometric polynomial. ◊

An LTI filter F, represented by its impulse response $f \in \ell(\mathbb{Z})$, is an FIR filter if and only if $f \in \ell_0(\mathbb{Z})$ which in turn is equivalent to its Fourier transform being a trigonomentric polynomial. This is not so bad in principle, sind the trigonometric polynomials are **dense** in most "reasonable" function spaces. This means that for any f and any $\varepsilon > 0$ there exists a trigonometric polynomial f_ε such that

$$\|f - f_\varepsilon\| < \varepsilon.$$

In $L_2(\mathbb{T})$ we even know how to compute the **best approximation** of a transfer function $g = \hat{f}$: take the Fourier series

$$\mathscr{F} g = \sum_{k\in\mathbb{Z}} g_k\, e^{ik\cdot}, \qquad g_k = \frac{1}{2\pi} \int_{\mathbb{T}} g(t)\, e^{-ikt}$$

and form its nth **partial sum**

$$\mathscr{F}_n g := \sum_{|k|\le N} g_k\, e^{ik\cdot} =: \hat{h},$$

to obtain the coefficients of the impulse response h of our filter H. An we even know that $\hat{h}$ is the *unique* trigonometric polynomial of degree n that approximates the treansfer function g best possible with respect to the norm $L_2(\mathbb{T})$. This is what is called a **best approximation**, see (Lorentz, 1966; Sauer, 2017). More about Fourier series can be found for example in (Hardy and Rogosinsky, 1956; Katznelson, 1976; Tolstov, 1962). Unfortunetely, things still are not this easy even if the approximant can be described by elementary means.

Example 2.4.7 (Low pass filter). Now we want to design an almost realistic filter, namely a **low pass** filter that only lets the low frequencies pass and blocks the high one. Let us suppose that the filter lets the lower half of frequencies[27] pass which means that it transfer fuction is

$$\hat{f} = \chi_{[-\pi/2,\pi/2]};$$

this is *not* a trigononmetric polynomial, hence not an FIR filter and therefore cannot be realized practically. To obtain an *approximate* low pass, we determine the Fourier coefficients of $g = \hat{f}$ which can be computed as

$$g_0 = \frac{1}{2\pi}\int_{-\pi/2}^{\pi/2} dt = \frac{1}{2}$$

and, for $k \neq 0$,

$$\begin{aligned} g_k &= \frac{1}{2\pi}\int_{-\pi/2}^{\pi/2} e^{-ikt}dt = \frac{1}{2\pi}\left.\frac{e^{-ikt}}{-ik}\right|_{t=-\pi/2}^{\pi/2} = \frac{1}{2\pi}\frac{e^{-ik\pi/2}-e^{ik\pi/2}}{-ik} = \frac{1}{k\pi}\sin\frac{k}{2}\pi \\ &= \begin{cases} 0, & k \in 2\mathbb{Z}, \\ \frac{(-1)^{(k-1)/2}}{k\pi}, & k \in 2\mathbb{Z}+1, \end{cases} \end{aligned}$$

i.e.,

$$g_{2k+1} = \frac{(-1)^k}{k\pi}, \qquad g_{2k+2} = 0, \qquad k \in \mathbb{N}_0.$$

The partial sum of degree $n = 2m+1$ is then the trigonometric polynomial

$$\begin{aligned} h_n(\xi) &= \frac{1}{2} + \sum_{k=0}^{m} \frac{(-1)^k}{(2k+1)\pi} \underbrace{\left[e^{i(2k+1)\xi} + e^{-i(2k+1)\xi}\right]}_{2\cos(2k+1)\xi} \\ &= \frac{1}{2} + \sum_{k=0}^{m} (-1)^k \frac{2}{(2k+1)\pi}\cos(2k+1)\xi, \end{aligned}$$

which is the explicit form of the best approximating trigonometric polynomial of degree n. However, its quality as an approximation is fairly disappointing, see Fig. 2.4.5.

Fig. 2.4.5 shows the so-called **Gibbs phenomenon**: the partial sums of discontinuous transfer functions *always* come with overshooting effects that do not vanish even with increasing quality of approximation. This makes this type of FIR filters not very well suited for the construction of **band pass** filters whose transfer function is the characteristic function of a subinterval of $[-\pi, \pi]$. In addition, the quality of approximation of discontinuous functions by trogonometric polynomials, the so-called **approximation order**, is very limited, cf. (DeVore and Lorentz, 1993; Lorentz, 1966; Mhaskar and Pai, 2000; Sauer, 2002). As the right hand side image

[27] Which concrete frequencies, in Hz, this concerns, depend once more on the sampling.

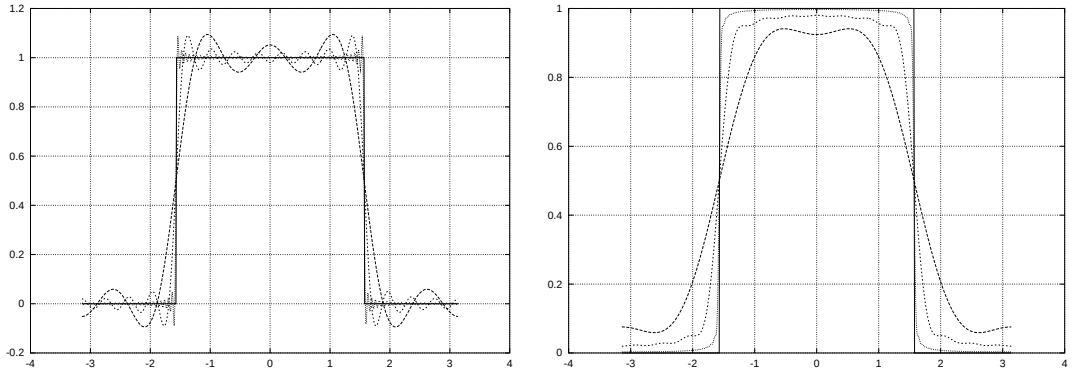


Figure 2.4.5: Approximation of the approximants *(left)* for the low pass filter for the more or less arbitrary values $n = 5, 15, 100$. Note that the "overshooting" peaks at the left and right boundary get only narrower, but not smaller in amplitude. Approximation by *Fejér means (right)* which have a larger error of approximation, but less oscillations and a mor similar shape.

in Fig. 2.4.5 shows, the Gibbs phenomenon can be avoided by chosing an appropriate approximation method like the **Fejér means**[28] which provide *shape preserving* approximation. But this better geometric behavior comes with the price of reduced accuracy or slower convergence for $n \to \infty$.

2.4.3 Filters for Images

Now we consider images in the sense of *pictures*, namely as two dimensional signals that map $\mathbb{R}^2 \to \mathbb{R}$ or $\mathbb{Z}^2 \to \mathbb{R}$, respectively. We start with a formal definition of what we consider here.

Definition 2.4.8 (Signal classe). By $\ell\left(\mathbb{Z}^2\right)$ we denote the set of all signals of the form[29]

$$c = \left(c(\alpha) \ : \ \alpha \in \mathbb{Z}^2\right) = (c(j,k) \ : \ j,k \in \mathbb{Z})$$

and use the norms

$$\|c\|_p := \left(\sum_{\alpha\in\mathbb{Z}^2} |c(\alpha)|^p\right)^{1/p}, \qquad \|c\|_\infty = \sup_{\alpha\in\mathbb{Z}^2} |c(\alpha)|,$$

which are a copy/paste version of what we already know, simply replacing indices by mutltiindices. For functions $f : \mathbb{R}^2 \to \mathbb{R}$ we define the analogous norms as

$$\|f\|_p = \left(\int_{\mathbb{R}^2} |f(t)|^p \ dt\right)^{1/p}, \qquad \|f\|_\infty = \sup_{x\in\mathbb{R}^2} |f(x)|.$$

[28] Instead of the nth partial sum one used the the arithmetic mean of the first $n+1$ partial sums of degree $0, \ldots, n$.

[29] The index $\alpha = (\alpha_1, \alpha_2) \in \mathbb{Z}^2$ used in this formula is called a **multiindex**.

The Fourier transform is also transferred in a very straightforward way to $\mathbb{R}^2$ by setting for $\xi \in \mathbb{R}^2$,

$$\begin{aligned}\hat{f}(\xi) &= \int_{\mathbb{R}^2} f(t) e^{-i\xi^T t}\, dt = \int_{\mathbb{R}^2} f(t) e^{-i(\xi_1 t_1 + \xi_2 t_2)}\, dt = \int_{\mathbb{R}^2} f(t)\, e^{-i\xi_1 t_1}\, e^{-i\xi_2 t_2}\, dt \\ &= \int_{\mathbb{R}} \left(\int_{\mathbb{R}} f(t_1, t_2)\, e^{-i\xi_1 t_1}\, dt_1 \right) e^{-i\xi_2 t_2}\, dt_2 = \int_{\mathbb{R}} \left(f(\cdot, t_2)\right)^{\wedge} (\xi_1)\; e^{-i\xi_2 t_2}\, dt_2.\end{aligned}$$

In other words, we can determine the Fourier transform in a successive way, "variable by variable", as a nested or iterated univariate Fourier transform. And once we have a Fourier transform, the convolution is not far, and again it looks precisely as in the unvivariate case:

$$f * g = \int_{\mathbb{R}^2} f(\cdot - t)\, g(t)\, dt.$$

Exercise 2.4.3 Show the validity of the identity

$$(f * g)^{\wedge} (\xi) = \hat{f}(\xi)\, \hat{g}(\xi)$$

in two variables. ◊

Exercise 2.4.4 Determine for the *bivariate* Fourier transform

- the inverse Fourier transform,
- the Perseval/Plancherel formula,
- the Poissonsche summation formula.

◊

LTI filters are no problem as well as any LTI filter is again defined by its impulse response $f \in \ell\left(\mathbb{Z}^2\right)$ and the filter itself is computed as

$$Fc = f * c = \sum_{\alpha \in \mathbb{Z}^2} f(\cdot - \alpha)\, c(\alpha), \tag{2.4.3}$$

where for any FIR filter $f \in \ell_{00}\left(\mathbb{Z}^2\right)$ we can be assured that (2.4.3) is well defined since the sum is a finite one for any argument. The filter becomes particularly simple if f is a **tensor product**, that is

$$f = f_1 \otimes f_2, \qquad \text{d.h.} \qquad f(\alpha) = f_1(\alpha_1)\; f_2(\alpha_2)\,. \tag{2.4.4}$$

Then,

$$\begin{aligned} f * c &= \sum_{\alpha \in \mathbb{Z}^2} f(\cdot - \alpha)\, c(\alpha) = \sum_{\alpha_1, \alpha_2 \in \mathbb{Z}} f_1\left((\cdot)_1 - \alpha_1\right) f_2\left((\cdot)_2 - \alpha_2\right)\; c(\alpha_1, \alpha_2) \\ &= \sum_{\alpha_1 \in \mathbb{Z}} f_1\left((\cdot)_1 - \alpha_1\right) \sum_{\alpha_2 \in \mathbb{Z}} f_2\left((\cdot)_2 - \alpha_2\right)\; c(\alpha_1, \alpha_2) \\ &= \sum_{\alpha_1 \in \mathbb{Z}} f_1\left((\cdot)_1 - \alpha_1\right)\; \left(f_2 * c\left(\alpha_1, (\cdot)_2\right)\right). \end{aligned} \tag{2.4.5}$$

This already yields a scheme how to apply such a bivariate image filter: for each fixed value α_1 we filter the row $c\,(\alpha_1, \cdot)$ of the signal with the filter f_2, which results in yet another image c' of the form

$$c'(\alpha) = c'(\alpha_1, \alpha_2) = (c(\alpha_1, \cdot) * f_2)\,(\alpha_2),$$

whose columns are the filter by means of f_1. Schematically,

$$\begin{array}{ccccccccccc}
\ddots & \vdots & \vdots & \iddots & & \ddots & \vdots & \vdots & \iddots \\
\dots & c(0,0) & c(0,1) & \dots & \to f_2 \to & \dots & c'(0,0) & c'(0,1) & \dots \\
\dots & c(1,0) & c(1,1) & \dots & \to f_2 \to & \dots & c'(1,0) & c'(1,1) & \dots \\
\iddots & \vdots & \vdots & \ddots & & \iddots & \vdots & \vdots & \ddots \\
 & & & & & & \downarrow & \downarrow & \\
 & & & & & & f_1 & f_1 & \\
 & & & & & & \downarrow & \downarrow & \\
 & & & & & \ddots & \vdots & \vdots & \iddots \\
 & & & & & \dots & (f*c)(0,0) & (f*c)(0,1) & \iddots \\
 & & & & & \dots & (f*c)(1,0) & (f*c)(1,1) & \iddots \\
 & & & & & \iddots & \vdots & \vdots & \ddots
\end{array}$$

If the filter F or his impulse response f have tensor product structure, respectively, then

$$\begin{aligned}
\hat{f}(\xi) &= \sum_{\alpha\in\mathbb{Z}^2} f(\alpha) e^{-i\alpha^T\xi} = \sum_{\alpha\in\mathbb{Z}^2} f(\alpha) e^{-i(\alpha_1\xi_1+\alpha_2\xi_2)} = \sum_{\alpha\in\mathbb{Z}^2} \underbrace{f(\alpha)}_{=f(\alpha_1)\,f(\alpha_2)} e^{-i\alpha_1\xi_1}\, e^{-i\alpha_2\xi_2} \\
&= \left(\sum_{\alpha_1\in\mathbb{Z}} f\,(\alpha_1)\; e^{-i\alpha_1\xi_1}\right)\left(\sum_{\alpha_2\in\mathbb{Z}} f\,(\alpha_2)\; e^{-i\alpha_2\xi_2}\right),
\end{aligned}$$

that is,

$$\hat{f}(\xi) = \hat{f}_1\,(\xi_1)\;\hat{f}_2\,(\xi_2)\,,$$

or

$$(Fc)^\wedge(\xi) = (f * c)^\wedge\,(\xi) = \hat{f}_1(\xi_1)\hat{f}_2(\xi_2)\,\hat{c}(\xi_1,\xi_2). \tag{2.4.6}$$

Or as a brief summary: filtering of images with tensor product filters is particularly easy.

2.4.4 Mean Values and Denoising

In particular in medical imaging a whole multitude of standard filters are common and an accepted standard, cf. (Handels, 2000). We will have a look the most popular and important ones among them. In many cases, the filter is introduced conceptionally in a *continuous* way, i.e. for functions $\phi : \mathbb{R}^2 \to \mathbb{R}$, and then the filter is discretized.

The first such continuous filter is the **mean value filter**, defined as

$$f = \frac{1}{|\Omega|}\chi_\Omega, \qquad \Omega \subset \mathbb{R}^2,$$

where Ω should be a *compact* set as then its volume $|\Omega|$ is well defined. The filtering

$$\begin{aligned} \phi \mapsto f * \phi(x) &= \frac{1}{|\Omega|}\int_{\mathbb{R}^2} \chi_\Omega(t)\phi\,(x-t)\;dt = \frac{1}{|\Omega|}\int_\Omega \phi\,(x-t)\;dt \\ &= \frac{1}{|\Omega|}\int_{x-\Omega} \phi(t)\,dt \end{aligned}$$

associates to each $x \in \mathbb{R}^2$ the mean value of the function over the set $x - \Omega$. The discretization of this filter is as simple as possible, one uses $f = S_h\chi_\Omega$ with a suitable sampling distance h and normalizes the filter by dividing by the number of sampling points in Ω.

Example 2.4.9. For $\Omega = [-1,1]^2$ and $h = 1$, we obtain

$$f = \frac{1}{9}\begin{pmatrix} 1 & 1 & 1 \\ 1 & \boxed{1} & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

where we only depict nonzero entries of the two dimensional impulse response and where the coefficient corresponding to $f(0)$ is maked by a frame. We can easily increase the resolution to obtain averaging filters of the form

$$f = \frac{1}{mn}\begin{pmatrix} 1 & \dots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \dots & 1 \end{pmatrix} \in \mathbb{R}^{m\times n}. \tag{2.4.7}$$

Remark 2.4.10. The matrix form of the impulse response in (2.4.7) can be written as

$$f = 1_m 1_n^T, \qquad 1_n = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \in \mathbb{R}^n,$$

hence, up to a shift $F = \chi_{[0,n-1]} \otimes \chi_{[0,m-1]}$ and the filter is a tensor product of two univariate filters. We are in the particularly simple situation as described above here.

The main application of averaging filter is **denoising**. Noise is an upleasant part of "real world" images, usually caused by measurement or computation[30] which is often expressed in probabilistic terms only. The additive standard model for noisy data is

$$\phi(x) = \psi(x) + \epsilon(x),$$

[30] In Computed Tomography, for example, the image is the result of fairly complex computations that are always affected by roundoff errors that can be seen as computational noise.

where ψ stands for the "true" data and ϵ for the noise. A common, but not always justified[31] is that is has **zero mean** or that its expectation vanishes:

$$E(\epsilon) = \int_{\mathbb{R}^2} \epsilon(x)\, dx = 0.$$

Filtering with a mean value filter then results in

$$F\phi(x) = \frac{1}{|\Omega|} \int_{x+\Omega} \phi(t)\, dt + \frac{1}{|\Omega|} \int_{x+\Omega} \epsilon(t)\, dt,$$

and if the noise has mean value zero, the second term, $\frac{1}{|\Omega|} \int_{x+\Omega} \epsilon(t)\, dt$, should become smaller, provided that the noise is local and that Ω is large enough to cover the support of the noise. On the other hand, the support should be small enough to ensure that $F\psi \sim \psi$, which is possible since

$$\phi(x) = \lim_{h \to 0^+} \frac{1}{h|\Omega|} \int_{x+h\Omega} \phi(t)\, dt, \qquad \phi \in L_1\left(\mathbb{R}^2\right),$$

see (Akhieser, 1988), otherwise the image will be blurred.

Remark 2.4.11. This observation described the main dilemma of filter based denoising: the support of the filter needs to be *small* to avoid blurring and *large* to average out the noise. Since these goals cannot be reached simultaneously, a compromise is necessary that will depend on the nature of the data and the nature of the noise. In practice, this usually means a lot of trial and error.

For a slightly more sophisticated and smoother filter[32] one can use the **Gauß kernel**

$$f(x) = \frac{1}{2\pi\sigma^2}\, e^{-\frac{\|x\|_2^2}{2\sigma^2}}, \qquad \sigma > 0, \tag{2.4.8}$$

with standard deviation σ, yet another parameter that can be chose reasonably or stupidly. To discretize the filter, we can sample f, more precisely $f\chi_{[-N,N]^2}$ to obtain an FIR filter, where the truncation is necessary since the kernel in (2.4.8) has infinite support. Its (purely) discrete counterpart is the **binomial filter**

$$f(j,k) = 2^{-m-n} \binom{m}{j} \binom{n}{k}, \qquad \begin{matrix} j = 0, \ldots, m, \\ k = 0, \ldots, n, \end{matrix}$$

which can be centered for even values of m, n.

Example 2.4.12. The impulse response of the centered binomial filter of dimension $2, 2$ has the form

$$\frac{1}{16} \begin{pmatrix} 1 & 2 & 1 \\ 2 & \boxed{4} & 2 \\ 1 & 2 & 1 \end{pmatrix}, \tag{2.4.9}$$

again with the element with coordinate 0 being marked. The effect of binomial filter is illustrated in Fig.

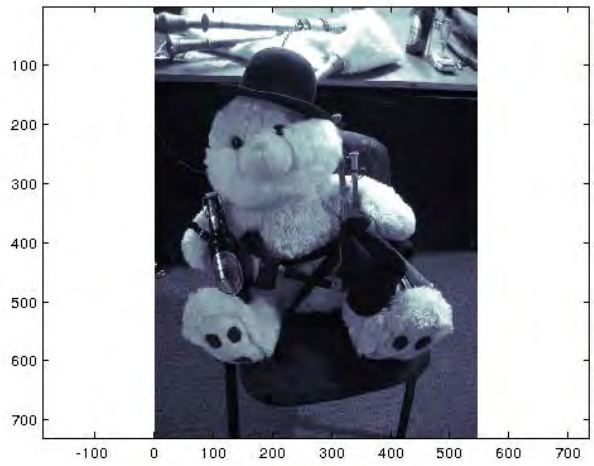


Figure 2.4.6: Test image with edges and fine textural parts that will illustrate the effects of the filters. And it is a *rabbit*, not a *bear*.

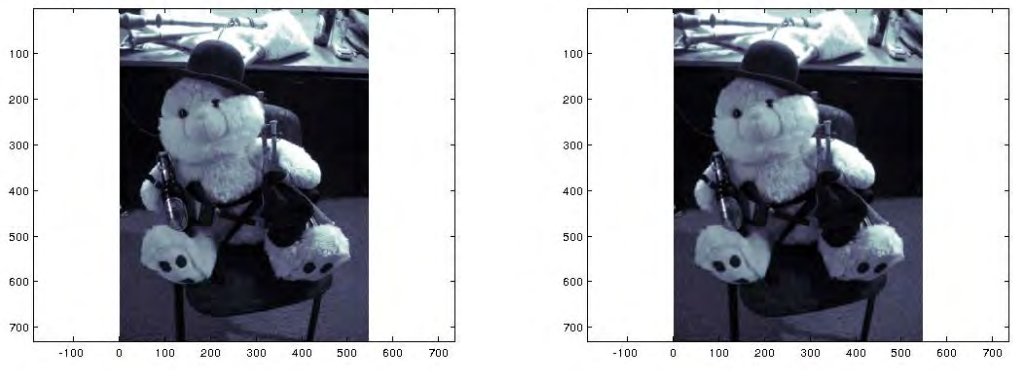


Figure 2.4.7: Application of the $2, 2$ and $4, 4$ binomial filter on the test image of Fig. 2.4.6. The effect of blurring is clearly visible from the disappearance of the fine hair structures.

A general rule of thumb concerning mean value filters, i.e., filters characterized by preservation of constant data, $f * 1 = 1$, is that the blurring is increased by increasing the support of the filter. The effect is a bit mildened by binomial filters since those have a builtin decay of their own. But still the effect is unavoidable.

2.4.5 Derivatives and Edges

The opposite task of denoising is emphasizing and detecting conturs in images which mainly means edges. To that end, a **gradient filter** is applied since edges are points where the difference between neighboring points is large, hence the slope is large which means a large absolute value of the derivative. In two variables, this is the **gradient** of a function ϕ, defined as

$$\nabla\phi = \begin{pmatrix} \dfrac{\partial}{\partial x}\phi(x,y) \\ \dfrac{\partial}{\partial y}\phi(x,y) \end{pmatrix},$$

and usuall discretized as the vector of first order partial differences

$$\nabla c = \begin{pmatrix} \nabla_1 c \\ \nabla_2 c \end{pmatrix} = \begin{pmatrix} c(\cdot+1,\cdot) - c(\cdot,\cdot) \\ c(\cdot,\cdot+1) - c(\cdot,\cdot) \end{pmatrix}. \tag{2.4.10}$$

Remark 2.4.13. The discretization by means of simple *forward differences* is not the best possible way, neither numerically nor geometrically. This is also the case in one variable. He we could consider, for a sampling width of h, the two neighboring points $x \pm h$ and use the first order Taylor approximation

$$f(x \pm h) = f(x) \pm hf'(x) + e(x,h), \qquad \lim_{h\to 0} \frac{e(x,h)}{h} = 0.$$

This leads the the sytem of equations

$$y \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{1}{h} \begin{pmatrix} f(x+h) - f(x) \\ f(x) - f(x-h) \end{pmatrix}$$

for the approximation $y \approx f'(x)$ whose minimal solution[33] is

$$y = \frac{f(x+h) - f(x-h)}{2h}. \tag{2.4.12}$$

[31] In serious denoising, it is important to get a clear idea on the underlying *noise model*, i.e., about properties and distribution of the noise.

[32] This is important on the frequency side! Smooth functions have a rapidly decaying Fourier transform which means that they are well-localized low-pass filters, see Theorem 2.2.4, 5).

[33] In the sense of

$$\min_y \left(y - \frac{f(x+h) - f(x)}{h} \right)^2 + \left(y - \frac{f(x) - f(x-h)}{h} \right)^2. \tag{2.4.11}$$

Hence, a better gradient approximation based on the four axial neighbors is

$$\nabla c = \frac{1}{2}\begin{pmatrix} c(\cdot+1,\cdot) - c(\cdot-1,\cdot) \\ c(\cdot,\cdot+1) - c(\cdot,\cdot-1) \end{pmatrix}.$$

So much for numerics, for simplicity, however, we will continue with the formula from (2.4.10).

Exercise 2.4.5 Verify that (2.4.12) is the solution of (2.4.11). When is the solution unique? ◊

Exercise 2.4.6 Derive the best possible estimate for the discrete gradient at $\alpha \in \mathbb{Z}^s$ based on nine neighbors $c(\beta)$, $\|\beta - \alpha\|_\infty \le 1$. ◊

The components of the discrete difference (2.4.10) are again obtained by filters whose impulse responses are

$$f_1 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & \boxed{-1} & 1 \\ 0 & 0 & 0 \end{pmatrix} \qquad \text{and} \qquad \begin{pmatrix} 0 & 1 & 0 \\ 0 & \boxed{-1} & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

Regardless of their numerical implementation, gradient methods or in general all operations based on differentiation, are very sensitive to noise since the difference quotient has a stepwidth h which amplifies inaccuracies in the data by a factor of h^{-1}. And in the end we want data that is sampled in a very fine way, at least that is what modern sensors (should) provide. To overcome this problem, one first applies denoising to the image before the gradient is used and for this purpose mean value filters are popular. To keep it simple, we consider the averaging operator

$$f := \frac{1}{9}\begin{pmatrix} 1 & 1 & 1 \\ 1 & \boxed{1} & 1 \\ 1 & 1 & 1 \end{pmatrix}.$$

The resulting convolution[34] $\nabla_j * f = f * \nabla_j$, i.e., the impulse response of the components of the resulting operator, is most easily computed by passing to the Fourier transform

$$\begin{aligned}(f * \nabla_1)^\wedge(\xi) = \hat{f}(\xi)\,\hat{\nabla}_1(\xi) &= \left(\frac{1}{9}\sum_{\|\alpha\|_\infty \le 1} e^{i\alpha\xi}\right)\left(e^{i\xi} - 1\right) \\ = \frac{1}{9}\sum_{\|\alpha\|_\infty \le 1} e^{i(\alpha+\epsilon_1)\xi} - \sum_{\|\alpha\|_\infty \le 1} e^{i\alpha\xi} &= \frac{1}{9}\left(e^{2i\xi_1} - e^{-i\xi_1}\right)\left(e^{i\xi_2} + 1 + e^{-i\xi_2}\right),\end{aligned}$$

and the averaged prefilters thus have the impulse responses

$$f_1 = \frac{1}{9}\begin{pmatrix} -1 & 0 & 0 & 1 \\ -1 & \boxed{0} & 0 & 1 \\ -1 & 0 & 0 & 1 \end{pmatrix} \qquad \text{and} \qquad f_2 = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & \boxed{0} & 0 \\ -1 & -1 & -1 \end{pmatrix}.$$

For the application of these filters see Fig. 2.4.8.

[34] I you have not realized yet that the convolution is *commutative*, then prove it.

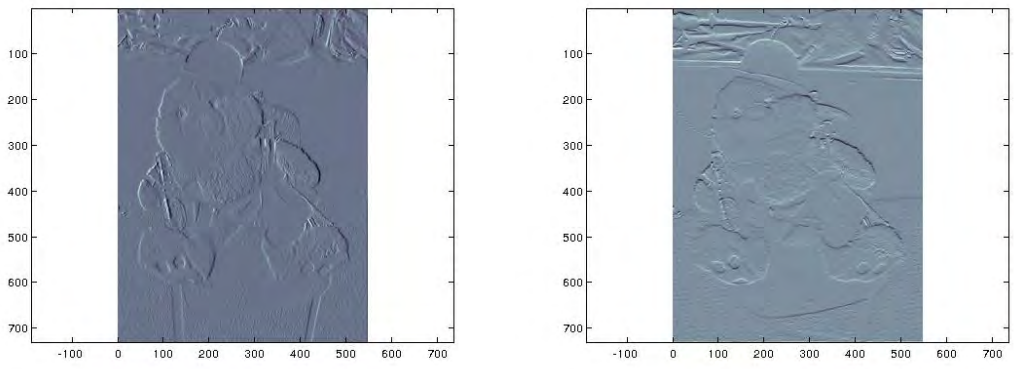


Figure 2.4.8: Application of the two components of the gradient filter, in x-direction *(left)* as well as in y-direction *(right)*. They mainly capture the orthogonal edges as is clearly visible.

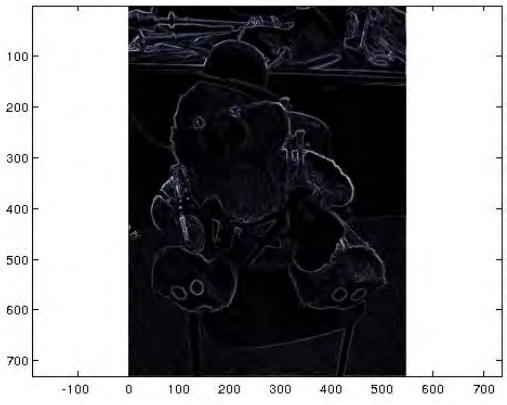


Figure 2.4.9: 1-norm of the gradient for edge detection.

Remark 2.4.14. 1. The above computation shows one of the advantages of using the Fourier transform as it gives us a symbolic calculus for filters, here for their concatenation. It all boils down to adding, multiplying and perhaps factorizing trigonometric polyomials. Later, in Definition 3.4.1, we will meet the z transform that allows us to do the same even more conveniently with (Laurent) polynomials.

2. In particular, $(f * \nabla_1)^\wedge(\xi) = g_1(\xi_1)g_2(\xi_2)$, so that the filter again has a tensor product structure that allows for simple implementation.

Extracting the edges themselves is now easy, we just have to look at the pointwise 1-norm of the smoothed gradient:

$$g = \|f * \nabla c\|_1 = |f * \nabla_1 c| + |f * \nabla_2 c|\,,$$

see Fig. 2.4.9.

As we are talking about derivatives, we can also make us of higher order deriva-

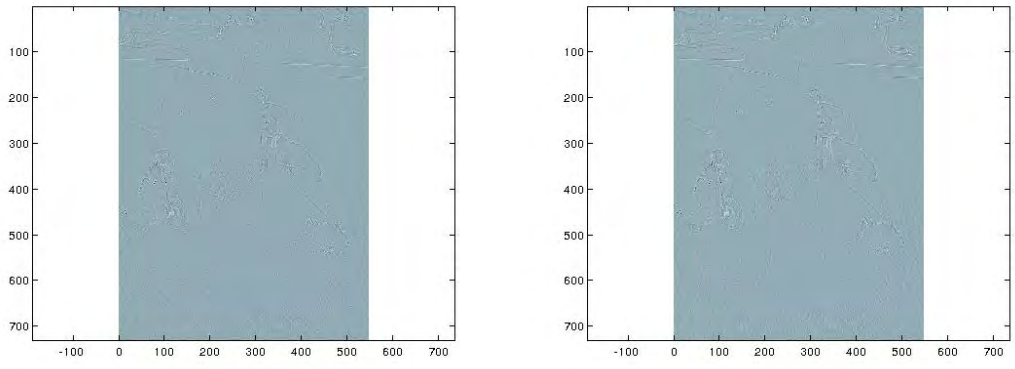


Figure 2.4.10: Direct application of the two Laplace filters from (2.4.13) *(left)* and (2.4.14) *(right)*. There is not much to be recognized due to the amplification of noise which becomes fairly random here. This becomes even more visible in Fig. 2.4.11.

tives and involve the **Laplace operator**

$$\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2},$$

a *second order* differential operator, where the second order derivatives are usually approximated by the second order symmectric differences $c(\cdot + 1) - 2c + c(\cdot - 1)$; this leads to the filter with impulse response

$$f = \begin{pmatrix} 0 & 1 & 0 \\ 1 & \boxed{-4} & 1 \\ 0 & 1 & 0 \end{pmatrix} \tag{2.4.13}$$

as the most simple discretization of the Laplace operator. A variant of this operator which also takes into account diagonal directions yield the impulse response

$$f = \begin{pmatrix} 1 & 1 & 1 \\ 1 & \boxed{-8} & 1 \\ 1 & 1 & 1 \end{pmatrix}. \tag{2.4.14}$$

As pointed out in (Handels, 2000), the Laplace operator is very sensitive to noise which is the reason why it is usally combined with smoothing filters like mean value filters or Gauß filters. Some of these effect are illustrated in Fig.

Exercise 2.4.7 Implement a smoothened version of the Laplace filter and optimize it for good edge detection. ◊

2.4.6 Somewhere in the Middle

The last classic among the image processing filters is the **median filter**, define in the univariate case as

$$Mc(j) = \text{Median}\{c(k) \ : \ k \in j + \Omega\}, \qquad \Omega \subset \mathbb{Z}^2;$$

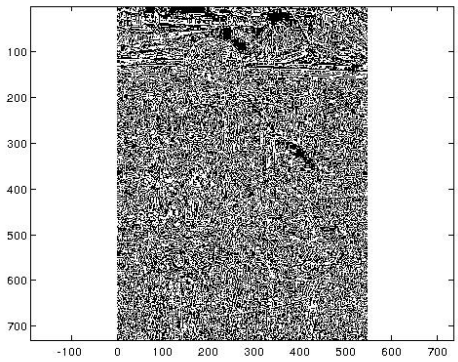


Figure 2.4.11: The noise in the right hand image of Fig. 2.4.10. The values were ranging from -1166 to 605, here only the positive values of the Laplace operator are plotted as black white dots.

to obtain the value of this filter, all elements of $c\,(j+\Omega)$ are sorted with respect to size and then the value in the middle is taken — this is the sloppy definition of the *median*. This filter has some peculiarities, however:

1. it is a nonlinear filter,

2. it has a fairly high computational effort due to the sorting process,

3. it is very robust to outliers.

In addition, the median filter is known to be an edge preserving filter as well. This has to do with the relationship between median and 1-norm as well as the good properties of the total variation which is defined as the 1-norm of the gradient. This, however, is beyond the scope of this lecture, but is a major topic in (Sauer, 2018).

Exercise 2.4.8 Implement a median filter `MedFilt (M,X)` for images where the first Argument `M` is a matrix with binary values $0, 1$ that encodes a discrete characteristic function for Ω. ◊

2.5 The FFT

In this section we consider *the* fundamental algorithm in Signal and Image Processing, namely the **fast Fourier transform**, better known as **FFT**. In the millenium fever of the year 2000 some people created a list of the ten most important and influential mathematical algorithms and the FFT was the unquestioned and undoubted winner. And this despite of the fact that the idea behind is almost incredibly simple. Or maybe because of that.

Since in principle the FFT is only a fast way fo computing the discrete Fourier transform, also known as DFT, we will first consider the DFT and some of its mathematical properties.

2.5.1 The discrete Fourier transform

When considering the Fourier transform of a sequence it may appear strange that, in contrast to the Fourier transform for functions, it maps the seqeunce $c \in \ell(\mathbb{Z})$ to the 2π periodic function $\hat{c} \in C(\mathbb{T})$ that is of a completely different structure. These connection makes perfect sense in the context of *dual groups* but from a naive point of view it is confusing.

But besides that, a continuum of frequency information is hard to process in practice, so that sampling in frequency domain appears almost natural. Due to the 2π periodicity of the Fourier transform $\hat{c}$, this sampling should be chosen with a stepwidth of the form $h = \frac{2\pi}{n}$ for some $n \in \mathbb{N}$. This maps $\hat{c}$ back to the sequence

$$\hat{c}_n = \mathrm{DFT}_n\, c := S_{2\pi/n}\hat{c} = \sum_{k\in\mathbb{Z}} c(k)\, e^{-2\pi i k\,\cdot/n}, \tag{2.5.1}$$

where

$$c, \hat{c}_n \in \ell(\mathbb{Z}).$$

Definition 2.5.1 (DFT). The sequence $\hat{c}_n \in \ell(\mathbb{Z})$ defined in (2.5.1) is called the **discrete Fourier transform** or **DFT** of order n of the sequence $c \in \ell(\mathbb{Z})$.

Since

$$\hat{c}_n\,(\cdot + n) = \sum_{k\in\mathbb{Z}} c(k)\, e^{-2\pi i\, k\,(\cdot/n+1)} = \hat{c}_n,$$

the DFT is a periodic sequence and it suffices to store only a block consisting of n consecutive values, that is, the DFT is alread determined by the values

$$\hat{c}_n(k), \qquad k \in \mathbb{Z}_n = \mathbb{Z}/n\mathbb{Z} \simeq \{0, \dots, n-1\}\,.$$

Remark 2.5.2. Recall that $\mathbb{Z}_n$ is more than only the set $\{0, \dots, n-1\}$, namely the set together with all operations modulo n, which means that addition and multiplication are all well defined. Since we will make explicit use of this fact, it is important to keep this subtlety in mind.

For a periodic or *periodized* sequence $c \in \ell(\mathbb{Z}_m)$ with period of length m, the DFT can conveniently be written in the matrix form

$$\hat{c}_n = V_{n,m}\, c, \qquad V_{n,m} := \left[e^{-2i\pi jk/n} \;:\; \begin{matrix} j \in \mathbb{Z}_n \\ k \in \mathbb{Z}_m \end{matrix} \right] \in \mathbb{C}^{n\times m}, \tag{2.5.2}$$

where in the most common case $m = n$ we simply write V_n. This is a standard form that maps signals of some length to signals of *the same* length or of the same amount of information content. And a short remark on periodization: if $c \in \ell_{00}(\mathbb{Z})$, the sequence can be written in such a way that its support is contained in $\mathbb{Z}_n$ for some n from which we can directly transform c by using V_n.

Example 2.5.3. The complex valued DFT of $S_{2\pi/512}\cos$ on $\mathbb{Z}_{512}$ is depicted in Fig. 2.5.1, where we plot real and imaginary part separately. Since the cosine function is even, its FFT is real, but due to numerical effects the imaginary part is not exactly zero.

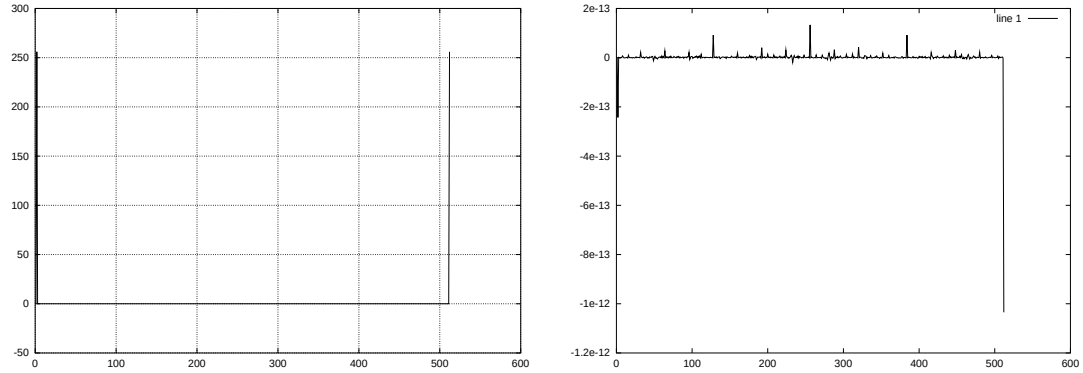


Figure 2.5.1: The discrete Fourier transform $\mathrm{DFT}_{512}\, S_{2\pi/512} \cos$ of a sampled cosine function. The real part of the transform *(left)* has two strong peaks at 1 and 511 which makes perfect sense since $\cos x = \frac{e^{ix}+e^{-ix}}{2}$ and therefore both frequencies ± 1 (modulo 512) have to appear in the DFT. And the imaginary part *(right)* only consists of numerical garbage with an amplitude of 10^{-13}.

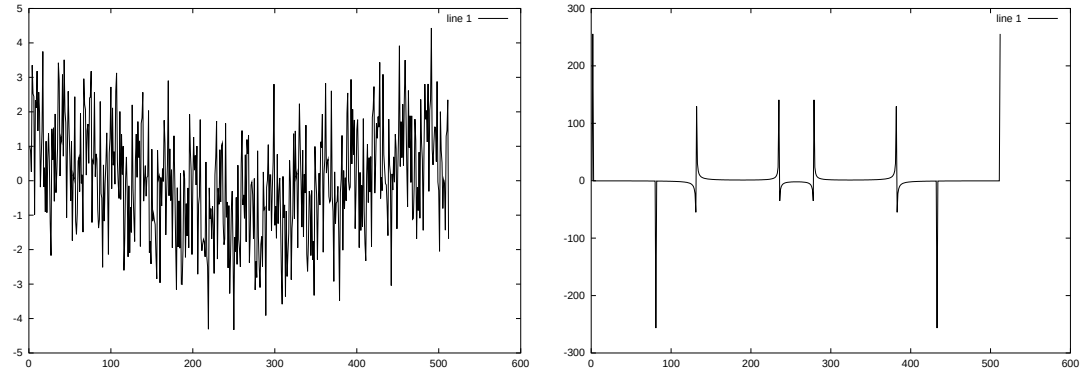


Figure 2.5.2: The function from Example 2.5.4 *(left)* and its DFT *(right)* with real and imaginary part plotted into one coordinate system. The integer frequencies give sharp peaks *either* in the real *or* in the imaginary part, while the non-integer ones lead to blurry spikes in both parts of the spectrum.

In general, sine and cosine functions whose frequencies divide the sampling rate, yield well-localized peaks in the real and imaginary part, while the same functions get massively blurred when they are sampled inappropriately.

Example 2.5.4. We now consider the "speech like" function

$$f(x) = \cos x - \cos 80x + \cos 130.7x + \sin 16x - \sin 277.8x$$

with several "disharmonic" frequency parts and determine $DFT_{512} S_{2\pi/512} f$. The result is depicted in Fig. 2.5.2

In what follows, we assume that $c, \hat{c} \in \ell(\mathbb{Z}_n)$, i.e., that we deal with a length preserving DFT that is described by the matrix V_n. To have a closer look at V_n, it is very useful to recall the notion of the (primitive) nth **root of unity** $\omega = e^{-2\pi i/n}$ which has the property that

$$\omega^n = e^{-2\pi i} = 1 \tag{2.5.3}$$

while $\omega^k \neq 1$ for $k < n$; the latter makes the root *primitive*. Then,

$$V_n = \left[\omega^{jk} \;:\; j,k \in \mathbb{Z}_n\right] = \begin{bmatrix} 1 & 1 & \dots & 1 & 1 \\ 1 & \omega^1 & \dots & \omega^{n-2} & \omega^{n-1} \\ 1 & \omega^2 & \dots & \omega^{2(n-2)} & \omega^{2(n-1)} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & \omega^{n-2} & \dots & \omega^4 & \omega^2 \\ 1 & \omega^{n-1} & \dots & \omega^2 & \omega^1 \end{bmatrix}$$

This matrix has a very simple and explicit inverse[35] that allows us to efficiently compute the inverse DFT.

Lemma 2.5.5 (Inverse DFT). *For $n \in \mathbb{N}$ the **inverse DFT** is given by*

$$V_n^{-1} = \frac{1}{n}\left[e^{2\pi ijk/n} \;:\; j,k \in \mathbb{Z}_n\right] = \frac{1}{n}\left[\omega^{-jk} \;:\; j,k \in \mathbb{Z}_n\right]. \tag{2.5.4}$$

Proof: Denoting the matrix on the right hand side of (2.5.4) by W_n, we observe that

$$(V_n W_n)_{jk} = \frac{1}{n}\sum_{\ell\in\mathbb{Z}_n} \omega^{j\ell}\,\omega^{-\ell k} = \frac{1}{n}\sum_{\ell\in\mathbb{Z}_n}\left(\omega^{j-k}\right)^{\ell}.$$

Hence, the diagonal elements are

$$(V_n W_n)_{jj} = \frac{1}{n}\sum_{\ell\in\mathbb{Z}_n}\left(\omega^{0}\right)^{\ell} = \frac{n}{n} = 1$$

while for the off-diagonal elements with $j \neq k$ we obtain

$$(V_n W_n)_{jk} = \frac{1}{n}\sum_{\ell=0}^{n-1}\left(\omega^{j-k}\right)^{\ell} = \frac{1}{n}\,\frac{\omega^0 - \left(\omega^{j-k}\right)^n}{1-\omega^{j-k}} = \frac{1}{n}\,\frac{1-(\omega^n)^{j-k}}{1-\omega^{j-k}};$$

since $\omega^n = 1$ and $-n < j-k < n$, hence[36] $\omega^{j-k} \neq 1$, we obtain that

$$(V_n W_n)_{jk} = \frac{1}{n}\,\frac{1-1^{j-k}}{1-\omega^{j-k}} = 0$$

hence $V_n W_n = I$ and therefore $W_n = V_n^{-1}$ since V_n is a square matrix[37]. □

Noting that $\omega^{-1} = e^{2\pi i/n} = \overline{\omega}$ is the complex conjugate of ω, the symmetry of V_n yields that

$$V_n^{-1} = \frac{1}{n}\overline{V_n} = \frac{1}{n}\overline{V_n^T} = \frac{1}{n}V_n^H,$$

which we can formulate as follows.

[35] This is not at all common among matrices, so let us celebrate this unusual piece of luck for a moment.

[36] This is the "primitive" part.

[37] Keep in mind that for non-square matrices there is a difference between a *left inverse* and a *right inverse*.

Corollary 2.5.6. *The matrix $n^{-1/2}V_n$ is **unitary**.*

Remark 2.5.7. The distribution of the factor $\frac{1}{n}$ between V_n and V_n^{-1} is unsymmetric and deliberate, and indeed that DFT can also be defined with a factor $1/\sqrt{n}$. This happens in the literature and in toolboxes and makes the constants nicer and yields, as shown in Corollary 2.5.6, to a *unitary* matrix. However, it somewhat destroy the interpretation as sampling of a trigonometric polynomial and introduces another irrational quantity. It is not hard to guess that the constant $1/\sqrt{n}$ plays a role analogous to that of the constant $1/2\pi$ in Fourier transform and inverse Fourier transform. The more analogies the better, but in the DFT context, the factor depends on the *signal length*, and whether the matrix is unitary or "only" satisfies $V_n V_n^H = V_n^H V_n = nI$ is as relevant as the distinction between "orthogonal" and "orthonormal".

Remark 2.5.8. The normalization is relevant when software libraries are used, as once again constants can lead to a lot of confusion there.

For $c \in \ell(\mathbb{Z}_n)$ the DFT $c \mapsto \hat{c}_n \in \ell(\mathbb{Z}_n)$ is a natural operation, if not *the* natural operation. We collect a few properties of $\mathrm{DFT}_n : \ell(\mathbb{Z}_n) \to \ell(\mathbb{Z}_n)$, which form yet another analogy to Theorem 2.2.4. To that end, we need an appropriate notion of convolution which makes use of the periodic structure of $\mathbb{Z}_n$.

Definition 2.5.9 (Cyclic convolution). For $c, d \in \ell(\mathbb{Z}_n)$ the **cyclic convolution** $c * d = c *_n d \in \ell(\mathbb{Z}_n)$ is defined as

$$(c * d)(j) = \sum_{k \in \mathbb{Z}_n} c(k)\, d(j-k), \qquad j \in \mathbb{Z}_n,$$

where $j - k$ is computed in $\mathbb{Z}_n$, hence has to be understood modulo n.

Theorem 2.5.10 (Properties of the DFT). *For $n \in \mathbb{N}$ one has*

1. *DFT_n is an invertible linear map for $\ell(\mathbb{Z}_n)$ to itself with*

$$\|\hat{c}_n\|_2 = \sqrt{n}\|c\|_2, \qquad c \in \ell(\mathbb{Z}_n). \tag{2.5.5}$$

2. *for any $c, d \in \ell(\mathbb{Z}_n)$,*

$$(c *_n d)^\wedge_n = \hat{c}_n\, \hat{d}_n = \left[\hat{c}_n(j)\, \hat{d}_n(j) : j \in \mathbb{Z}_n\right]. \tag{2.5.6}$$

Proof: *1)* Linearity is obvious and invertability follows from Lemma 2.5.5 where the inverse was given explictly. For (2.5.5) we use the unitary invariance of the 2-norm[38] and obtain

$$\|\hat{c}_n\|_2 = \|V_n c\|_2 = \sqrt{n}\left\|\left(n^{-1/2}V_n\right)c\right\|_2 = \sqrt{n}\|c\|_2.$$

[38]Recall that for unitary U, i.e. $U^H U = I$, and any x we have that $\|x\|_2^2 = x^H x = x^H U^H U x = \|Ux\|_2^2$.

2) Since ω is an nth root of unity, i.e., $\omega^n = 1$ and therefore also $\omega^{k+n} = \omega^k$, the sequence $k \mapsto \omega^k$ belongs to $\ell(\mathbb{Z}_n)$. For $j \in \mathbb{Z}_n$, this allows us to conclude that

$$\begin{aligned}(c *_n d)^\wedge_n(j) &= \sum_{\ell\in\mathbb{Z}_n}\left(\sum_{k\in\mathbb{Z}_n} c(k)\,d(\ell-k)\right)\omega^{j\ell} = \sum_{k,\ell\in\mathbb{Z}_n} c(k)\,d(\ell-k)\,\omega^{jk}\omega^{j(\ell-k)}\\ &= \hat{c}_n(j)\,\hat{d}_n(j).\end{aligned}$$

The other half of (2.5.6) is proved analogously. □

Since we also want to deal with images, i.e., two dimensional objects, we should also have a look at the two dimesional DFT of $c \in \ell\left(\mathbb{Z}_n^2\right)$ which can be written as

$$\begin{aligned}\hat{c}(\beta) &= \sum_{\alpha\in\mathbb{Z}_n^2} c(\alpha)\,e^{-2\pi i\alpha^T\beta/n} = \sum_{\alpha_1\in\mathbb{Z}_n}\sum_{\alpha_2\in\mathbb{Z}_n} c(\alpha_1,\alpha_2)\,e^{-2\pi i\alpha_1\beta_1/n}e^{-2\pi i\alpha_2\beta_2/n}\\ &= \sum_{\alpha_1\in\mathbb{Z}_n} e^{-2\pi i\alpha_1\beta_1/n}\sum_{\alpha_2\in\mathbb{Z}_n} c(\alpha_1,\alpha_2)\,e^{-2\pi i\alpha_2\beta_2/n}\\ &= \sum_{\alpha_1\in\mathbb{Z}_n} (c(\alpha_1,\cdot))^\wedge(\beta_2)\;e^{-2\pi i\alpha_1\beta_1/n},\end{aligned}$$

which gives us a practical rule for the two dimensional DFT in the spirit of tensor products:

> For each *row*[39] `c(j,:)` of the matrix `c` one computes the DFT $(c(j,\cdot))^\wedge$ to obtain another matrix (a column of rows) that is then processed columnwise.

Schematically, this looks as follows

$$\begin{array}{|c|c|c|}\hline c(0,0) & \cdots & c(0,n-1)\\ \hline \vdots & \ddots & \vdots\\ \hline c(n-1,0) & \cdots & c(n-1,n-1)\\ \hline\end{array}\;\rightarrow\;\vdots\;\rightarrow\;\begin{array}{|c|c|c|}\hline c'(0,0) & \ldots & c'(0,n-1)\\ \hline \vdots & \ddots & \vdots\\ \hline c'(n-1,0) & \ldots & c'(n-1,n-1)\\ \hline\end{array}\;\begin{array}{c}\\ \\ \\ \downarrow\\ \\ \\ \\ \end{array}\;\begin{array}{|c|c|c|}\hline \hat{c}(0,0) & \ldots & \hat{c}(0,n-1)\\ \hline \vdots & \ddots & \ddots\\ \hline \hat{c}(n-1,0) & \ldots & \hat{c}(n-1,n-1)\\ \hline\end{array} \qquad (2.5.7)$$

Therefore, the performance of a two or higher dimensional DFT clearly depends directly on the performance of a one dimensional DFT and costs as much an $2n$ of them in 2D. However, the horizontal and vertical operations can be performed independently of each and therefore have a lot of potential for parellelization, especially on graphics cards.

2.5.2 Discrete versus Discretized

In many practical cases, for example when working on sound or brain activities, the discrete data to be processed results from a *finite* sampling of a continuous signal, i.e.,

$$c(k) = (S_h f)(k), \qquad k \in \mathbb{Z}_N,$$

[39] We use the notation from `Matlab/octave` here

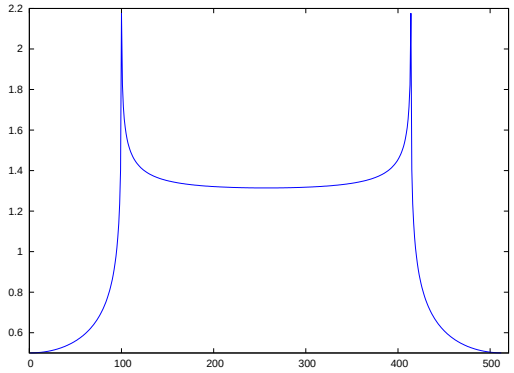


Figure 2.5.3: The DFT of the sinc function from Example 2.5.11, more precisely the modulus, i.e., the complex absolute value of this function. At the boundary we see nasty artifacts that cannot be explained by discretization alone.

where the letter "N" is used to indicate that the number of samples is large. If we compute the DFT $\hat{c}_N$ of this signal c, we compute, from the mathematical point of view, a discretization of the associated trigonometric polynomial

$$\hat{c}(\xi) = \sum_{k\in\mathbb{Z}} c(k)\, e^{-ik\xi} = \sum_{k\in\mathbb{Z}_N} f(hk)\, e^{-ik\xi},$$

on the grid $2\pi\mathbb{Z}_N/N$, that is,

$$\hat{c}_n(j) = \sum_{k\in\mathbb{Z}_N} f(hk)\, e^{-2i\pi jk/N}, \qquad j \in \mathbb{Z}_N.$$

There is no direct connection between this vector and what we intended to compute, namely the discretization of $\hat{f}$, the Fourier transform of f. It should come as no surprise that this leads to artifacts again.

Example 2.5.11. Let us consider the DFT of a sampled sinc-function whose Fourier transform is[40] a characteristic function. We sample this function, in `octave` notation via

```
>> N = 512; c = sinc( 100*pi*(0:N-1)/N );
```

It is deliberate here tht we do not sample at the integers as this would result in a δ sequence. The result of a DFT combined with `fftshift`[41] is depicted in Fig. 2.5.3 and shows that the high frequencies in this *band pass* filter are amplified.

To get closer to the function itself, we incorporate a so-called *quasi interpolant* as an approximation to f based on the the samples. For a function $\phi \in L_{00}(\mathbb{R})$ the **quasi interpolant** is defined as the scaled convolution

$$Q_{h,\phi}c := \phi * c\left(h^{-1}\cdot\right) = \sum_{k\in\mathbb{Z}_N} c(k)\phi\left(h^{-1}\cdot - k\right) = \sum_{k\in\mathbb{Z}_N} f\,(hk)\ \phi\left(h^{-1}\cdot - k\right)$$

[40] Or at least should be.
[41] This is an `octave` function that shifts the zero frequency into the middle of the vector.

definiert. If ϕ is even a cardinal function, defined by $\phi|_{\mathbb{Z}} = \delta$, then $Q_{h,\phi}(hk) = S_h f(k) = f(hk)$, $k \in \mathbb{Z}_N$, and the data is even interpolated, completely quasi-free. If not, one at least hopes to obtain a reasonable approximation.

Example 2.5.12. The most common functions for quasi interpolants are the *cardinal splines* which we know as autoconvolutions of characteristic functions from Example 2.2.7. The interpolatory one among them are those of degree 0 and 1, i.e., χ and $\chi * \chi$, all the other ones a noninterpolatory. The resulting approximation operators are called **Schoenberg operator** and can be found for example in (Sauer, 2002; Sauer, 2007; Schoenberg, 1973).

If we assume[42] that $Q_{h,\phi}c$ *approximates* the function f, meaning that $\left\|f - Q_{h,\phi}c\right\|_1$ is small, then the Fourier transform of $Q_{h,\phi}c$ is also a good approximation of the Fourier transform $\hat{f}$ of f and we can compute the latter from the sampled information as

$$\left(Q_{h,\phi}c\right)^\wedge(\xi) \quad = \quad \left(\sigma_{h^{-1}}(\phi * c)\right)^\wedge(\xi) = h\,(\phi * c)^\wedge(h\xi) = h\,\hat{\phi}(h\xi)\,\hat{c}(h\xi).$$

If we replace ξ by $h^{-1}\xi$, in this identity, and then discretize ξ on the *discrete torus* $\mathbb{T}_N := 2\pi\mathbb{Z}_N/N$, we obtain

$$\hat{f}\left(\frac{2k\pi}{Nh}\right) \simeq \left(Q_{h,\phi}c\right)^\wedge\left(\frac{2k\pi}{Nh}\right) = h\,\hat{\phi}\left(\frac{2k\pi}{N}\right)\hat{c}_N(k), \qquad k \in \mathbb{Z}_N. \tag{2.5.8}$$

This simple formula eventually connects the discrete Fourier transform $\hat{c}_N(k) = (S_h f)^\wedge_N$ of the samples with an *approximate* discretization of the Fourier transform $\hat{f}$ of f and nicely explains the relationship:

1. The **sampling rate** h determines which frequencies of f are really encoded in the DFT $\hat{c}_N$, and the smaller h is, the smaller this frequency range becomes. This, of course, should come as no surprise.

2. The **frequency resolution** as the number of entries of the spectrum that are computer depends on the number N of the discretizations. The larger N is, the more precisely the spectrum is represented and the smaller is the distance between these frequencies. If N is small, on the other hand, more entries will be combined into one. Of course, the computational effort increases with N as well.

3. One cannot decouple these two parameters so easily. Normally the discrete data results from sampling over a substantial range or period of size Nh, so that a high sampling rate will usually be related to a high frequency resolution as well.

4. The normalizing factor h in (2.5.8) is only of interest if we are care for the concrete energy of the frequencies, if we only are for the *distribution* of these values we might as well ignore it.

[42] Otherwise we definitely made a mistake with the quasi interpolant.

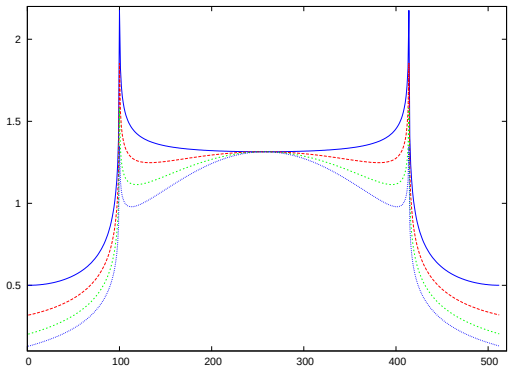


Figure 2.5.4: Filtering of the function from Fig. 2.5.3 with cardinal splines of order $0, 1, 2, 3$. The outliers become smaller with increasing order, however for the price of a bump at the boundary.

5. The distance between $\hat{f}$ and the approximation $\left(Q_{j,\phi} f\right)^{\wedge}$ clearly affects how well these values describe the discretized Fourier transform. Since $\|\hat{g}\|_\infty \leq \|f\|_1$ and since we only sample of $[0, Nh]$, the L_1 quality of approximation

$$\left\|f - Q_{h,\phi} f\right\|_1 := \int_0^{Nh} \left|f(t) - Q_{h,\phi} f(t)\right| \, dt \leq Nh \max_{0 \leq t \leq Nh} \left|f(t) - Q_{h,\phi} f(t)\right|$$

is crucual for the quality of this approximation. For cardinal splines there are estimates of the order Ch^2, i.e., the error decreases quadratically with the sampling rate. sind.

6. If cardinal spline functions are chose as $\phi = \chi * \cdots * \chi$, the correction filter $\hat{\phi}$ is simply a power of the sinc function, but we sample only the first "hill" of this function that decreases faster with increasing order of the spline. In this respect, it results in a stronger damping of unwanted high frequency contributions.

7. If we simply drop or ignore ϕ in (2.5.8), then we replace $\hat{\phi}$ by $\chi_{[0,1]}$, which means that $\phi = \text{sinc}$ and instead of $\hat{f}$ one discretizes the interpolatory reconstruction from the sampling theorem, Theorem 2.3.3. But this is no really good approximation, unless the function is bandlimited, the sampling rate is appropriate and the sampling ranges over all of $\mathbb{Z}$. But this is not the real life.

2.5.3 The Fast Fourier Transform

The most appealing aspect of the DFT, however, is not the fact that it is a consistent extension of the Fourier transform for periodic or periodized functions, but that it can be computed really *fast*. This is due to the *fast Fourier transform* or *FFT*, for short. In this respect, it would be more accurate to call the FFT an FDFT, i.e., a fast discret Fourier transform.

The algorithm itself was (re)discovered by Cooley and Tukey in 1965 in (Cooley and Tukey, 1965), see also (Cooley, 1987; Cooley, 1990), and it works not only

for the DFT, but can actually be applied in quite arbitrary rings and is also used, for example, for the fast multiplication of large integers (Schönhage and Strassen, 1971; Gathen and Gerhard, 1999). All that is needed for the algorithm is a *primitive nth root of unity* like the $\omega = e^{-2i\pi/n}$ in the definition of the DFT.

And the idea behind the FFT is strikingly simple: let us suppose that $n = 2m$ is an even number and remark that

$$\omega^2 = e^{-2\pi i 2/n} = e^{-2\pi i/m} =: \omega_m, \qquad \omega_n := \omega,$$

then we get for any $c \in \ell(\mathbb{Z}_n)$ and $j \in \mathbb{Z}_n$ that

$$\begin{aligned} \hat{c}_n(j) &= \sum_{k\in\mathbb{Z}_n} c(k)\,\omega^{jk} = \sum_{k\in\mathbb{Z}_m} c(2k)\,\omega^{2jk} + \sum_{k\in\mathbb{Z}_m} c(2k+1)\,\omega^{j(2k+1)} \\ &= \sum_{k\in\mathbb{Z}_m} c(2k)\,\omega_m^{jk} + \omega^j \sum_{k\in\mathbb{Z}_m} c(2k+1)\,\omega_m^{jk} \\ &= (c(2\cdot))^\wedge_m(j) + \omega^j\ (c(2\cdot+1))^\wedge_m(j), \end{aligned}$$

which is

$$\hat{c}_n = c^\wedge_n = (c(2\cdot))^\wedge_m + \omega^{\cdot}\,(c(2\cdot+1))^\wedge_m. \tag{2.5.9}$$

The **FFT** now consists of applying this computational rule recursively, which of course requires that $n = 2^k$ for some k, at least in our naive form.

What is now the value of this representation? If we assume that the values $\omega, \ldots, \omega^{n-1}$ are *precomputed* and available in a table[43], the naive version of the DFT as a matrix-vector mulitplication of an $n \times n$ matrix with an n-vector would require $O(n^2)$ operations. So let us check what a computation via (2.5.9) would need and call that number $F(n)$. Then (2.5.9) tells us that for the computation of $\hat{c}_n$ we first have to compute the two DFTs of length $m = n/2$ on the right hand side (cost of $2F(n/2)$), then multiply the second result componentwise with the vector $(\omega^j : j \in \mathbb{Z}_n)$ (cost of n) and the add them componentwise[44] (cost of n). Hence, the total effort in (2.5.9) is $2(F(n/2)+n)$, which leads to the *recurrence relation*

$$F(n) = 2\,(F(n/2) + n)\,. \tag{2.5.10}$$

For a dyadic $n = 2^\ell$ with $\ell \in \mathbb{N}$, we thus get

$$F(n) = 2^k F\left(2^{\ell-k}\right) + k\,2^{\ell+1}, \qquad k = 1, \ldots, \ell, \tag{2.5.11}$$

which follows by induction from the fact that for $k = 1$ (2.5.11) is just (2.5.10) while the inductive step is

$$\begin{aligned} F(n) &= 2^k F\left(2^{\ell-k}\right) + k\,2^{\ell+1} = 2^k\ 2\left(F\left(2^{\ell-k-1}\right) + 2^{\ell-k}\right) + k\,2^{\ell+1} \\ &= 2^{k+1} F\left(2^{\ell-k-1}\right) + 2^{\ell+1} + k\,2^{\ell+1} = 2^{k+1} F\left(2^{\ell-k-1}\right) + (k+1)\,2^{\ell+1}. \end{aligned}$$

[43] They are the same for all $c \in \ell(\mathbb{Z}_n)$ and could be held in some cache; and even if they were not precomputed, the effort for that is $O(n)$.

[44] The vectors are m-periodic and will simply be extended by periodicity.

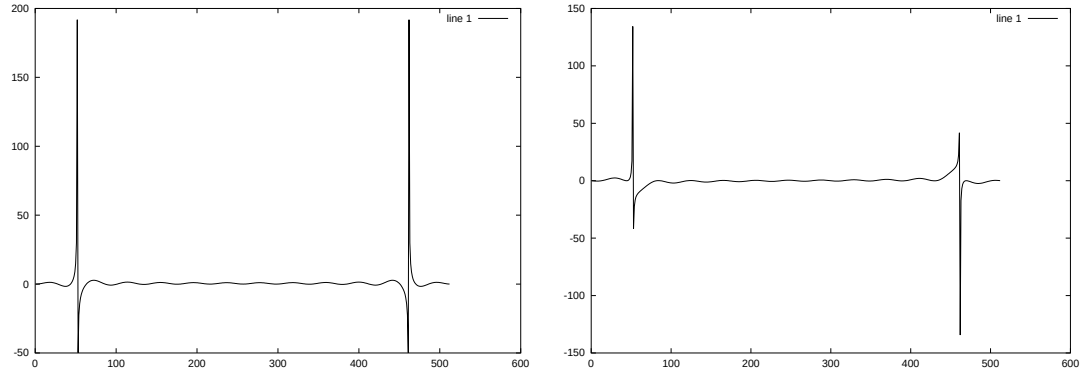


Figure 2.5.5: Real *(left)* and imaginary *(right)* part of the FFT of $S_{2\pi/500}\cos(50\cdot)$, zero-padded to 512 entries. The two main frequencies 50 and 450 are still clearly visible, but the large imaginary parts are quite misleading.

Considering (2.5.11) for the special case $k = \ell = \log_2 n$, this becomes

$$F(n) = \underbrace{2^\ell}_{=n} F(1) + \underbrace{\ell\, 2^{\ell+1}}_{=2n\,\log_2 n} = n\,(2\,\log_2 n + F(1)) = O\,(n\,\log_2 n)\,,$$

which is significantly better than the $O\,(n^2)$ of the naive matrix-vector multiplication. Indeed, $O\,(n\,\log_2 n)$ is a typical asymptotic complexity for methods based on the principle of **divide and conquer**, and is often derived as a consequence of the so-called **master theorem**, (Steger, 2001).

This complexity results holds not only for numbers n that are powers of 2. If ℓ is chosen such that $2^{\ell-1} < n \le 2^\ell$, then we can simply replace n by 2^ℓ and embed the original signal into a larger one, for example by **padding** it, i.e., adding zeros. Then the computational cost

$$\begin{aligned} 2^\ell F(1) + 2\ell\, 2^\ell &\le 2n\,F(1) + 2\log_2(2n)\,2n = 2n\,F(1) + 4n\,(\log_2 n + 1) \\ &= 2n\,(2\log_2 n + F(1) + 2)\,, \end{aligned}$$

is increased by a factor of two which leaves the *asymptotic* complexity still at $O\,(n\,\log_2 n)$.

Example 2.5.13. Of course, padding is just a complexity argument and not something that can be recommended in practice. If, for example, we form $S_{2\pi/500}\cos(50\cdot)$ as a signal in $\ell\,(\mathbb{Z}_{500})$ and pad this sequence by appending 12 zeros to it in order to yield an element of $\ell\,(\mathbb{Z}_{512})$, then the frequencies are blurred significantly, in the real as well as in the imaginary part, see Fig. 2.5.5. And just to make it clear: imaginary frequencies should not even appear in that DFT!

One might assume that the problem with the function in Fig. 2.5.5 would be due to the fact that periodicity is destroyed and that the signal should be extended periodically. But then the period lengths 500 and 512 do not fit, except in the lucky case that the signal is of such high frequency that the periodization meets a full period of the signal.

The FFT that we defined here is known as the **radix**-2 **FFT** since the decompositions are performed on the basis 2 and are based on halving the data. This can

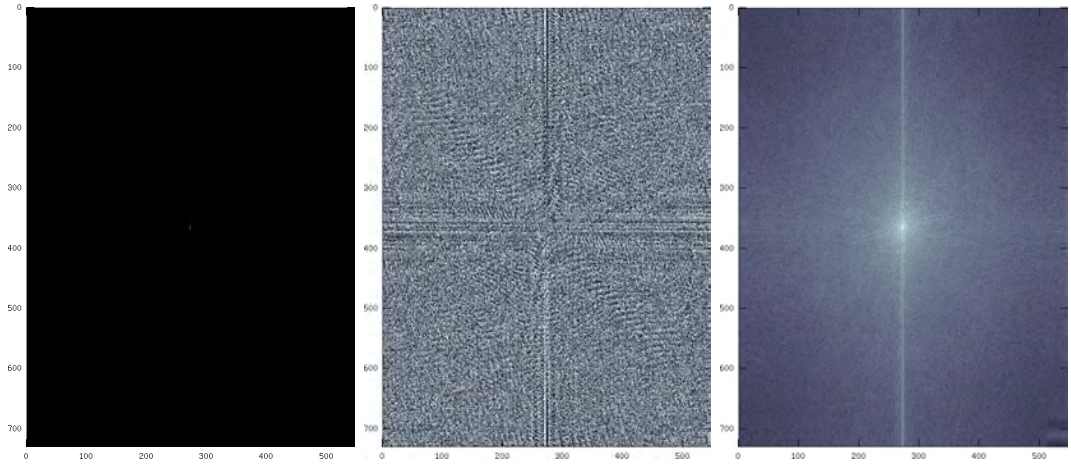


Figure 2.5.6: FFT of our test image from Fig. 2.4.6, showing the absolute value *(left)* and the phase *(center)*. The absolute values vary so strongly that nothing can really be recognized. This is improved by using a **logarithmic representation** of the absolute values *(right)*. And just to make it clear: the phase image has to be considered *cyclically* which means that black is the same as white.

be done with any other basis Die Form der FFT, die wir hier betrachtet haben, ist die sogenannte $p \in \mathbb{N}$ as well, where for $m = n/p$ the analogous decomposition modulo p yields

$$\begin{aligned}\hat{c}_n(j) &= \sum_{k\in\mathbb{Z}_m}\sum_{\ell\in\mathbb{Z}_p} c\,(pk+\ell)\;\omega^{j(pk+\ell)} = \sum_{\ell\in\mathbb{Z}_p}\omega^{j\ell}\left(\sum_{k\in\mathbb{Z}_m} c\,(pk+\ell)\;\omega_p^{jk}\right)\\ &= \sum_{\ell\in\mathbb{Z}_p}\omega^{j\ell}\;\left(c\,(p\cdot+\ell)\right)_m^{\wedge}(j),\end{aligned}$$

which results in the computation of *more* DTFs for *shorter* segments and has a computational complexity of $O\left(n\,\log_p n\right)$ that just differs by a constant once more.

Exercise 2.5.1 Show that the computational cost of the **radix-p FFT** is $O\left(n\,\log_p n\right)$. ◊

Exercise 2.5.2 Derive, on the basis of (2.5.7) an algorithm for a *bivariate* FFT `fft2` and determine its computational complexity. ◊

2.5.4 Fourier and Images

A major reason to apply the FFT in signal processing is the fact that it allows us to compute the cyclic filtering (2.5.6) in $O\left(n\log_2 n\right)$ instead of $O\left(n^2\right)$ by simply transforming both signals with an FFT, then multiplying them componentwise and finally transforming the signal back. Indeed, the transformations cost $O\left(n\log_2 n\right)$ and the multiplication $O(n)$, so that we again end up with $O\left(n\log_2 n\right)$.

The DFT can be applied directly and naively on images, but a little bit of care might be in order as Fig. 2.5.6 shows. But even the logarithmic scale[45] does not

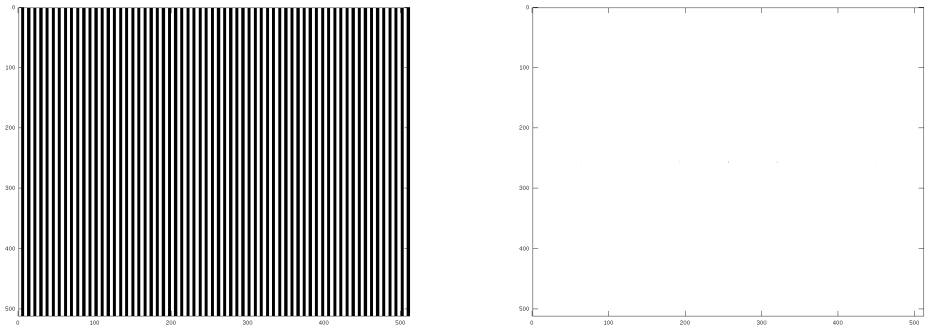


Figure 2.5.7: Stripes without stars *(left)* and their (inverted) DFT *(right)* that is only different from zero at three small, well-localized points that can only be seen in proper resolution.

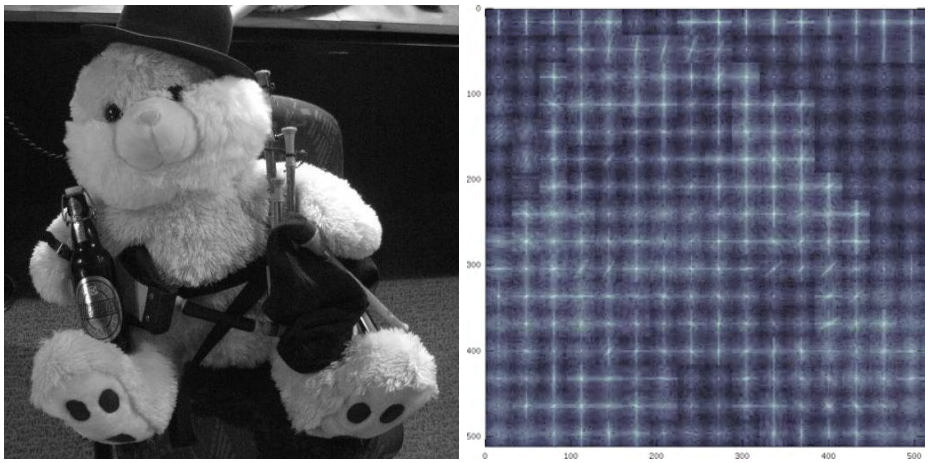


Figure 2.5.8: Image of size 512×512 and the logarithmic DFT of its 16×16 blocks. At locations with contours one can can see crosses if the block FFTs that are even pointing in the same direction as the edges, where there is nothing to be seen, for example on the floow, the DFT is quite diffuse.

really give us valid information about the image. Part of that is due to the fact that the DFT is tailored for *periodic* signals which our image is not. Generating an image that consists of vertical strips by

```
>> X = kron( ones(64), ones(8,1)*[ 1 1 1 1 0 0 0 0 ] );
```

and then transforming it to obtain Fig. Abb:stripes. Here the DFT just shows the constant part, i.e., the sum over all pixel values, in the middle, an the frequency of the stripes in the other two dots. A similar test can be applied to checkerboard patterns.

Exercise 2.5.3 Create different checkerboard images and compute and plot their DFT. ◊

[45]The bright spot in the center of the logarithmic modulus of the FFT shows that the absolute value of the FFT should have a white spot in the center that dominates everything.

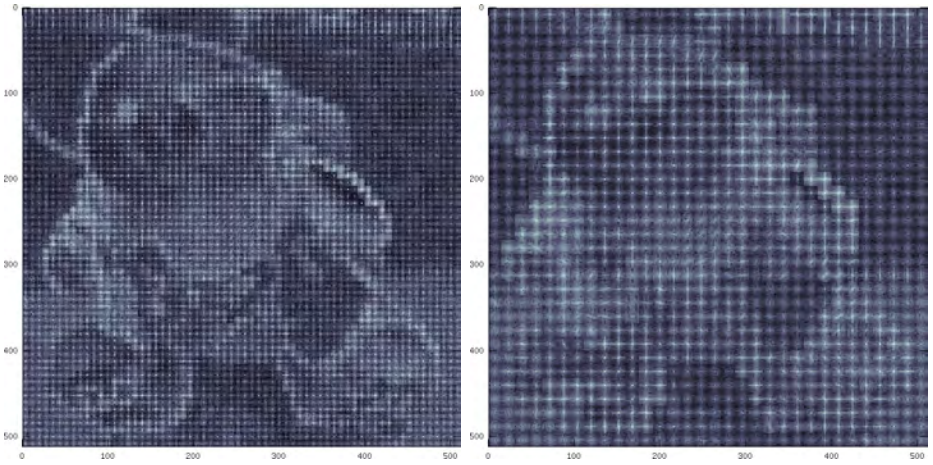


Figure 2.5.9: Block DFT with blocks of size 8 *(left)* and of size 16 *(right)*.

Making the[46] assumption that *locally* images are either constant or or have a somewhat periodic texture, we can try to make use of this assumption. To that end, we decompose an image $C = \big(c(j,k) \,:\, j,k \in \mathbb{Z}_n\big)$ into blocks of size n/m, yielding

$$C = \begin{pmatrix} C_{00} & \dots & C_{0,m-1} \\ \vdots & \ddots & \vdots \\ C_{m-1,0} & \dots & C_{m-1,m-1} \end{pmatrix}, \qquad C_{jk} \in \mathbb{R}^{\frac{n}{m}\times\frac{n}{m}};$$

then, we transform each of these blocks *separately*:

$$\hat{C} \neq \widetilde{C} := \begin{pmatrix} \hat{C}_{00} & \dots & \hat{C}_{0,m-1} \\ \vdots & \ddots & \vdots \\ \hat{C}_{m-1,0} & \dots & \hat{C}_{m-1,m-1} \end{pmatrix}.$$

For different sizes of m, the results are depicted in Fig. 2.5.8 and Fig. 2.5.9. The computation of each block has a cost of $C\ (n/m)^2 \log_2 n/m$ and since we have m^2 blocks in total, the complete cost of the decomposition is $O\left(n^2 \log n\right)$, i.e., the same as a full FFT of the image.

To these *local* DFTs we can now apply a **low pass filter** in a very simple and efficient way by multiplying each of the individual DFTs componentwise with a **mask** $M \in \mathbb{R}^{\frac{n}{m}\times\frac{n}{m}}$ that is concentrated around the center. This can be done with a "hard window" consisting of ones are with a properly normalized binomial filter and leads to

$$C^* := \begin{pmatrix} \hat{C}_{00} \odot M & \dots & \hat{C}_{0,m-1} \odot M \\ \vdots & \ddots & \vdots \\ \hat{C}_{00} \odot M & \dots & \hat{C}_{0,m-1} \odot M \end{pmatrix} = \widetilde{C} \odot (1_{m\times m} \otimes M)\,.$$

This is a good occasion to learn two new matrix products, namely the **Hadamard**

[46] More or less realistic, often more less than more.

produkt

$$A \odot B := \left[a_{jk} b_{jk} \ : \ \begin{matrix} j = 1, \ldots, p \\ k = 1, \ldots, q \end{matrix} \right] \in \mathbb{R}^{p \times q}, \qquad A, B \in \mathbb{R}^{p \times q},$$

and the **Kronecker produkt**[47]

$$A \otimes B = \begin{bmatrix} a_{11} B & \ldots & a_{1q} B \\ \vdots & \ddots & \vdots \\ a_{p1} B & \ldots & a_{pq} B \end{bmatrix} \in \mathbb{R}^{pr \times qs}, \qquad A \in \mathbb{R}^{p \times q}, \quad B \in \mathbb{R}^{r \times s}.$$

Both are implented in `Matlab/Octave` in a very efficient way.

Once we have computed C^*, which can be done by performing n^2/m^2 componentwise multiplications which cost $O(m^2)$ operations each, hence with a total of $O\left(n^2\right)$, we only need to transform this matrix back and obtain a filtered and possibly compressed version of C.

Example 2.5.14. We use the 4×4 binomial filter

$$B_4 = \frac{1}{4} \begin{bmatrix} 1 \\ 2 \\ 2 \\ 1 \end{bmatrix} [1\,2\,2\,1] = \frac{1}{4} \begin{bmatrix} 1 & 2 & 2 & 1 \\ 2 & 4 & 4 & 2 \\ 2 & 4 & 4 & 2 \\ 1 & 2 & 2 & 1 \end{bmatrix}$$

and embed it into an $\frac{n}{m} \times \frac{n}{m}$ matrix whose remaining entries we set to zero. In `octave` this can be done with the following piece of code, where we assume that the image of size 512×512 is stored in `A`:

```
>> bin4 = [ 1 2 2 1 ]'*[ 1 2 2 1 ] / 4;
>> M = zeros( 16 ); M( 7:10,7:10 ) = bin4;
>> B16 = blockDFTshift( A ) .* kron( ones( 32 ), M );
>> A16 = blockIDFTshift( A );
```

The resulting image can be seen in Fig. 2.5.11. Smaller *block artifacts* can (of course) be obtained by using smaller blocks, see Fig. **??**.

Remark 2.5.15. An important practical observation is as follows: the thing with the forward and backward DFT and shifting the zero frequency to the midpoint only works if the block size is a *power of two*, i.e., $n = 2^k$ for some appropriate k. The image size itself only has to be a multiple of n which makes things at least a bit easier.

What did we gain with all this effort except that we have an operation that can be performed efficiently due to the FFT. Let us have a look at Fig 2.5.11 and the block DFT there. There we set all values in each of 16×16 blocks to zero except those that were met by the 4×4 filter. Hence we only kept $\frac{4 \times 4}{16 \times 16} = \frac{1}{16}$ of the information, the image has been *compressed* by means of local low pass filtering.

[47] Which we already encountered as the `kron`–Funktion of `Matlab/Octave`.

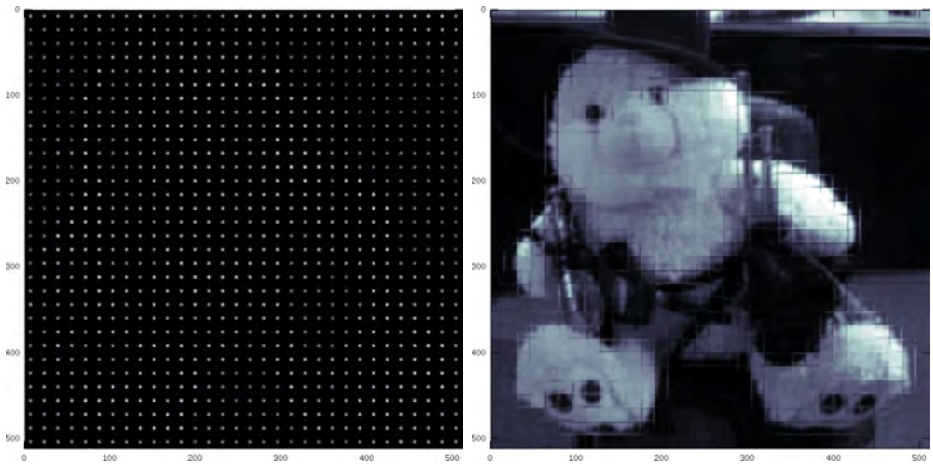


Figure 2.5.10: Truncated block DFT of our standard image *(left)* and its reconstruction by an inverse DFT *(right)*. The image can still be recognized very well despite of the clearly visible block artifacts.

Figure 2.5.11: Reconstruction from 8×8 blocks, using a 4×4 binomial filter *(left)* and the characteristic function $1_{4\times4}$ *(right)*. Can you see a difference?

Figure 2.5.12: Original image *(left)* from Fig. 1.1.1 and its modification with subsampled CbCr component *(right)*. The difference is hardly visible and the edges are still fairly well preserved.

This idea of compressing the low pass content of a transformed image is the fundamental idea behing the image compression standard **JPEG**[48]. However, there is more behind it:

1. JPEG already uses several tricks at image preprocessing. The first step consists of a transformation to the YCbCr color model and subsamples the chroma contents. This is almost invisible to the human eye but already reduces the amount of data to 50% in color images: $\frac{1}{3} + \frac{1}{12} + \frac{1}{12} = \frac{1}{2}$. The effect is illustrated in Fig. 2.5.12.

2. The filtering is not done in the brute force way described above, but by quantization and rounding. That means, the coefficients in the DFT are rounded to finite values and all values *below* a certain threshold are set to zero. This usually includes high frequency content[49]. The value of this threshold determines the **compression rate** of the JPEG compression - the higher it is, the more coefficients are set to zero which improves the compression rate but reduces the image quality, of course.

3. Since computations with complex numbers mean extra effort, JPEG is not using the DFT but the so called discrete cosine transform (DCT) that matches real valued data to real valued data, see Definition 2.5.17.

4. After transformation and quantization, JPEG adds a so called *entropy encoder* that reduces the amount of data even further in a lossless way.

[48]The full documentation of the standard can be found for example at the web address `www.w3.org/Graphics/JPEG/itu-t81.pdf`.

[49]The Fourier transform decays with increasing frequency as we learned in Proposition 2.2.9.

Remark 2.5.16. The transform between the RGB and the YCbCr color model is defined for $0 \le R, G, B \le 1$ as

$$\begin{pmatrix} Y \\ Cb \\ Cr \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}, \qquad -\frac{1}{2} \le Y, Cb, Cr \le \frac{1}{2},$$

and each row sums up to one, which means that it is a weighted combination of the color channels. The inversce of the transformation matrix is easily[50] as

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1 & -0.00092674 & 1.40168676 \\ 1 & -0.34369538 & -0.71416904 \\ 1 & 1.77216042 & 0.00099022 \end{pmatrix} \begin{pmatrix} Y \\ Cb \\ Cr \end{pmatrix}$$

and shows that the Y part, in some sense the "black and white contours", contributes the same to each color channel.

Definition 2.5.17 (DCT). The **discrete Cosine transform** or **DCT** maps a vector $f \in \ell(\mathbb{Z}_n)$ to a linear combination of cosine terms, for example as

$$\mathrm{DCT}_{II} f(j) = \sum_{k \in \mathbb{Z}_n} f(k) \cos \frac{\pi \left(k + \frac{1}{2}\right) j}{n}. \tag{2.5.12}$$

This is known as the **DCT–II**, overall there are four types of DCTs.

Of course, the DFT could be realized naively and directly via the matrix

$$D_n^{II} := \left[\cos \frac{\pi \left(k + \frac{1}{2}\right) j}{n} \ : \ j, k \in \mathbb{Z}_n \right], \qquad \text{d.h.} \qquad \mathrm{DCT}_{II} f = D_n^{II} f,$$

but this would be very inefficient. Since

$$\cos \frac{\pi \left(k + \frac{1}{2}\right) j}{n} \ = \ \cos \frac{2\pi \left(2k+1\right) j}{4n} = 2 \left(e^{\frac{2\pi i (2k+1) j}{4n}} + e^{\frac{-2\pi i (2k+1) j}{4n}} \right),$$

we again only have to pad our signal, that is, we define $d \in \ell(\mathbb{Z}_{4n})$, taking into account that $-j = 4n - j$ in $\mathbb{Z}_{4n}$, as

$$d(2k+1) = d(-2k-1) = c(k), \quad k \in \mathbb{Z}_n, \qquad d(2k) = 0, \quad k \in \mathbb{Z}_{2n},$$

and obtain for $j \in \mathbb{Z}_{4n}$ that

$$\begin{aligned} \hat{d}(j) \ &= \ \sum_{k \in \mathbb{Z}_{4n}} d(k)\, e^{-2\pi i j k/(4n)} = \sum_{k \in \mathbb{Z}_{2n}} d(k)\, e^{2\pi i j k/(4n)} + d(-k)\, e^{-2\pi i j k/(4n)} \\ &= \ \sum_{k \in \mathbb{Z}_n} c(k)\, 2 \left(e^{\frac{2\pi i (2k+1) j}{4n}} + e^{\frac{-2\pi i (2k+1) j}{4n}} \right) = \sum_{k \in \mathbb{Z}_n} c(k) \cos \frac{\pi \left(k + \frac{1}{2}\right) j}{n} \\ &= \ \mathrm{DCT}_{II} c(j), \end{aligned}$$

so that also the DCT can be computed with an effort of $O(n \log_2 n)$. By our standard methods, the DCT can be also extended to the bivariate case and can be implemented via a version of `fft2`.

Exercise 2.5.4 Formulate a 2D-DCT and implement it on the basis of an FFT. ◊

[50] For example using `octave`.

Remark 2.5.18 (DCT). Even if the DCT can be realized through the FFT in principle, the factor 4 or 16 in the two dimensional case is not really nice. Because of that, there are specific methods to compute the DCT directly, and these ideas rely on a matrix matrix formulation like with the DFT: splitting the DFT to obtain the FFT can be written as a **matrix factorization**.

3 Transformations

There is never enough time to do all the nothing you want.

(Calvin and Hobbes, Online 26.8.2018)

Next, we want to consider further *transformations* or *transforms* of signals. As the name indicates, a transformation *transforms* a signal (which includes images, of course) into a different form or even structure. This is mostly due to two reasons:

1. We cannot measure the signal itself but only a transformation of it. The Radon transform from (1.3.2) was a nice example, but also any Photography we take is only a transformation of a three dimensional environment. In this situation, the task is to *invert* the transformation and to reconstruct the original information from it.

2. The transormation allows us to detect or access information that has not been available in the original signal. An example is the frequency content that becomes visible when applying the Fourier transform.

The goal of this chapter is to have a look at some particularly important transformations that are frquently used in signal and image processing.

3.1 The Hough transform

The **Hough transform** was published[1] in 1962 in the patent application (Hough, 1962) and is still very popular in image processing. Mostly, it is used to detect lines in images based on a duality between points and lines. We already know this from the Radon transform where a line in the plane is written as

$$L = \left\{ x \in \mathbb{R}^2 \ : \ v^T x = c \right\}, \qquad \|v\|_2 = 1, \quad c \in \mathbb{R}, \tag{3.1.1}$$

and is thus encoded by the values v and c which we can write as $L = L(v, c)$. There are some degrees of freedom in this encoding since

$$v^T x = c \quad \Leftrightarrow \quad (-v)^T x = (-c), \qquad \text{d.h.} \qquad L(v, c) = L(-v, -c),$$

which means that the encoding is an **even function**. The vectors v on the unit circle can be written as

$$v = v_\theta = \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix}, \qquad \theta \in [-\pi, \pi]$$

[1] The correct wording would be *"disclosed"*.

and eliminating the degree of freedom, we eventually write lines as

$$L = L(\theta, c) := L(v_\theta, c), \qquad \theta \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right), \quad c \in \mathbb{R}.$$

The important part is the reduced interval for θ which makes the representaion for lines a *unique* one. Now, we flip the roles and consider via

$$H(x) = \left\{(\theta, c) \ : \ v_\theta^T x = c\right\} \subset \left[-\frac{\pi}{2}, \frac{\pi}{2}\right) \times \mathbb{R} =: \mathbb{H}, \tag{3.1.2}$$

a parametrization of all lines through a given a point $x \in \mathbb{R}^2$ as a curve in $\mathbb{H}$. If we now consider $H(x)$, $x \in X$, for a (finite) set $X \subset \mathbb{R}^2$ for which many of its elements lie on a line $L(\theta, c)$, then the pair (θ, c) will occur often in $H(X)$. The more points we have on this line, the more often the pair will occur. More precisely, for any choice $(\theta, c) \in \mathbb{H}$ the value

$$n_X(\theta, c) := \#\{x \in X \ : \ (\theta, c) \in H(x)\},$$

counts how many points of X lie on the line $L(\theta, c)$, where of course only values $n_X(\theta, c) \gg 2$ are of interest as two points can always be connected by a line. Since (3.1.2) determines $H(x)$ for each image pixel as indicator curve of a lines through x, the value of the transformation does not depend on color of intensity of this pixel, it just depends on whether it "is there" or not.

Definition 3.1.1.

1. A **binarized** image is an image whose pixels take only the values 0 and 1.
2. The **Hough transform** of a finite[2] binarized image with pixels

$$X = \left\{(x_j, y_j) \in \mathbb{R}^2 \ : \ j = 1, \ldots, N\right\}$$

is defined as

$$(H(X))(\theta, c) = n_X(\theta, c), \qquad (\theta, c) \in \mathbb{H}.$$

Since in general most lines will never be incident with any point of a finite set, $H(X)$ is zero almost everywhere and this yields a rather unstable definition of the transform that would not be robust enoug for practical applications. To overcome that, one decomposes $\mathbb{H}$ into regions $\mathbb{H}_{jk} = \Theta_j \times C_k$, $j, k = 1, \ldots, M, M'$, such that the Θ_j form a partition[3] of $[-\pi/2, \pi/2]$ and the C_k partition a sufficiently large subset of $\mathbb{R}$. Based on the partition, we simply *count* how many of the values $H(x_\ell)$ lie in a certain partition $\mathbb{H}_{jk}$. This can be done in a very simple way by the following procedure:

Algorithm 3.1.2 (Discrete Hough transform).

[2] In principle it is not difficult to extend the Hough transform to continuously defined images.

[3] A **partition** X_j of X is a decomposition of X with the properties $X_j \subset X$, $\bigcup_j X_j = X$ and $X_j^\circ \cap X_k^\circ = \emptyset$, hence a decomposition into subsets whose interiors are disjoint. Sometimes the "interior" is also dropped or trivial, the latter for example whenever X is finite.

Figure 3.1.1: Binarized edges with the threshold set to $\frac{1}{3}$ *(left)*, $\frac{1}{7}$ *(center)* and $\frac{1}{10}$ *(right)* of the maximal value, shown in inverse representation, i.e., detected pixels are shown in black. The dilemma between missing edges and overrepresentation of noise and textures is clearly visible.

1. **Given**: Punkt $x = (x, y)$
2. Set $\boldsymbol{H} = 0_{M \times M'}$
3. For $j = 1, \ldots, M$:
   a) θ_j = midpoint of Θ_j.
   b) Determine an index $1 \leq k \leq M'$ such that
   $$x \cos\theta_j + y \sin\theta_j \in C_k.$$
   c) $H_{jk} \leftarrow H_{jk} + 1$.
4. **Result**: M = discrete/discretized Hough transform

Exercise 3.1.1 Implement the Hough transform from Algorithm 3.1.2. ◊

Fortunately, the Hough transform is already available in `Matlab`[4] and `octave` as a function `houghtf`, so that we can play a little bit with it. Usually, the Hough transform is applied to classify edges in images which should first be processed by an edge filter and then binarized accordingly. This has another important reason: the complexity of the Hough transform depends linearly on the number of pixels to be considered and since edges are usually one dimensional objects, the hope is that an image with N pixels might have about $O\left(\sqrt{N}\right)$.

We once more consider our running example from Fig. 2.4.6. We use gradient filters and their 1-norm to determine the images in Fig. 2.4.9:

```
>> Gx = [ -ones( 3,1) zeros(3) ones( 3,1 ) ]/9; Gy = -Gx';
>> K = abs( filter2( Gx,H ) ) + abs( filter2( Gy,H ) );
>> Kb = K > max( max( K ) ) / 7;
```

[4]However, in an Image Processing toolbox that has to be bought separately.

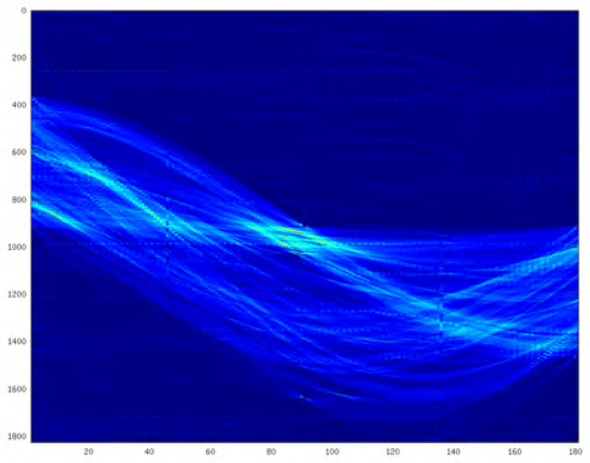


Figure 3.1.2: Hough transform of the edges in Fig. 3.1.1 with the ration $\frac{1}{7}$ threshold. A few of the maxima can be recognized as "bright spots".

The last step in this process that determines `Kb` as some factor of the maximal entry is purely arbitrary. Indeed, the threshold $\frac{1}{7}$ times the maximal value has been chosen experimentally with the goal to have as many lines and a few isolated pixels as possible, see Fig. 3.1.1. Next we use the builtin function `houghtf` to compute the transform from the image:

```
>> [Ht,R] = houghtf( Kb );
```

This function samples the angles with a distance of $1^\circ = \frac{\pi}{180}$, i.e within the range `1:180`, and the c values are choses in the range $[-L, L]$ where L is the length of the diagonal of the image. The sampled values for c are returned in the variable `R`, which makes it a lot easier. The result of this transform can be seen in Fig. 3.1.2. Now, we get the parameters of the most dominant lines from the Hough transform and plot them together with the image:

```
>> [Hr,Hc] = find( Ht > .6*max(max(Ht) ) );
>> imagesc( 1.-Kb ); colormap( bone ); axis equal
>> houghLines( [Hr,Hc],R,H );
```

The result can be seen in Fig. 3.1.3. The method has some fairly obvious advantages and disadvantages:

1. If applied carelessly, the method detects *phantom edges*, in this case the image boundary. This, however, is not a feature of the Hough transform but of the edge detection used to binarize the image: when padding the image with zeros, most of the image boundary has a large gradient and therefore a boundary frame consisting of long lines is generated - the Hough transform just does what it is supposed to do.

2. The line edges are not localized. This can be seen by the diagonal edge which is just defined by the bagpipe chanter and some other edges it accidentially

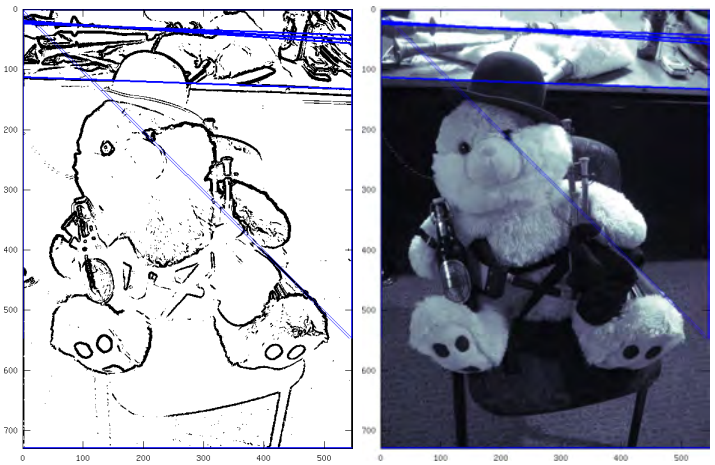


Figure 3.1.3: Line edge detection by means of the Hough transform, in the contour image *(left)* as well as in the image itself *(right)*. Looking carefully the images reveals that the image boundary was detected by the transform which is explained by the fact that it was marked by our edge detection.

meets. Though dominant, the true edge is very short. To find the line edge one first would have to discretize the edge line[5] and then look for all binary pixels that are sufficiently close to it. This is no rocket science, but it also is not for free.

3. On the other hand, this is also an advantage of the method as it can find lines that are partially hidden like the edge of the table behind the hat.

4. Still we face the problem of finding a proper threshold. The 60% of the maximal value were just some choice we made and the level is so hight that for example the lower edge of the table or the legs of the chair are not found. If, on the other hand, we lower the threshold, we may end up with a lot of lines that at some point are too many to still give relevant information. And if we set it to 2, then we get *all* lines through *any* pair of pixels.

The Hough transform is particularly useful if there are lines running through the whole image which are hidden by some local objects of if the image is dominated by a rather small number of lines.

Example 3.1.3 (Orientation of rectangular objects). When dealing with rectangular objects of (ideally) known size, their orientation can be detected rather comfortably by means of the Hough transform, for example in the process of tracking objects on a belt. Once the threshold is calibrated, the orientation can be detected quite easily and efficiently.

[5] A well-known and solved problem in Computer Graphics, cf. (Foley et al., 1990).

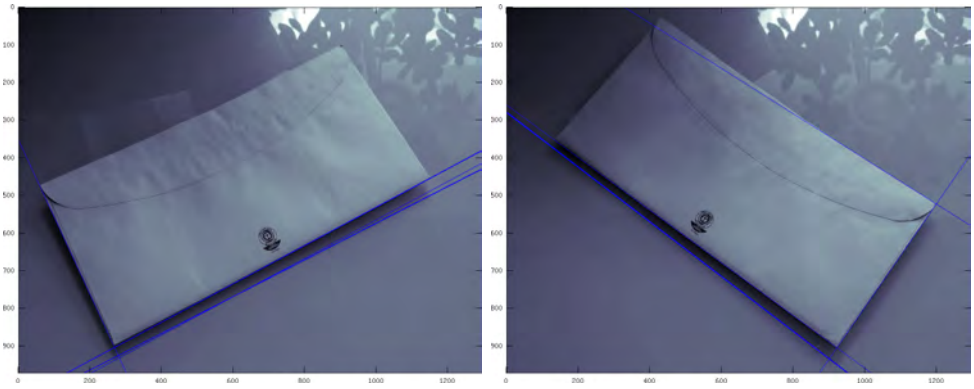


Figure 3.1.4: Envelope with two different orientations. Indeed, the Hough transform recognizes the edges, but also the shadow artifacts. The more complex pattern in the background, however, is ignored.

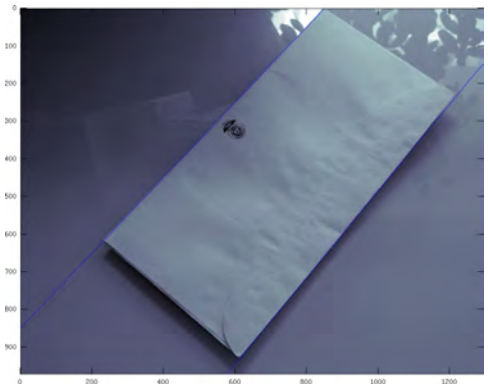


Figure 3.1.5: The envelope once more, this time none of the short sides is detected.

Exercise 3.1.2 Develop an algorithm that detects a rectangular object of your choice in images. You can use any edge detection method and the Hough transform from `Matlab/octave`. ◊

With a little bit of mathematical thinking the Hough transform can be viewed more abstractly and can be generalized to arbitrary implict parametric curves.

Definition 3.1.4. An implicit paramtetric curve with respect to the function $F : \mathbb{H} := \mathbb{R}^s \times \mathbb{P}$ for a given **parameter** $p \in \mathbb{P}$ is the set

$$\left\{x \in \mathbb{R}^s \ : \ 0 = f_p(x) = F(x,p)\right\} \subset \mathbb{R}^s. \tag{3.1.3}$$

The (generalized) **Hough transform** with respect to F associates to $x \in \mathbb{R}^s$ the set of all fitting parameter values

$$H(x) := \{p \in \mathbb{P} \ : \ F(x,p) = 0\} \subset \mathbb{P} \tag{3.1.4}$$

Example 3.1.5. The case of the classical Hough transform is given as[6]

$$\begin{aligned} \mathbb{H} = \mathbb{S}^2 \times \mathbb{R}_+ \quad &\text{and} \quad F(x,p) = v^T x + c, \qquad p = (v,c), \\ \mathbb{H} = \left[-\frac{\pi}{2}, \frac{\pi}{2}\right] \times \mathbb{R} \quad &\text{and} \quad F(x,p) = [\cos\theta, \sin\theta]\, x + c, \qquad p = (\theta, c). \end{aligned}$$

There ist yet another example for a Hough transform in the literature, namely

$$F(x,p) = (x_1 - p_1)^2 + (x_2 - p_2)^2 - p_3^2,$$

which codes the **midpoint** and **radius** of a **circle**. The transform collects all circles that contain a certain point x, which is fairly simple since for a given radius $r = p_3 \in \mathbb{R}_+$ the midpoints of all circles of radius r that contain x lie on circle of radius r with center x. Now, however, one has to figure out where the paramters for circles accumulate in a three dimensional space which is significantly more expensive than the Hough transform for lines. To understand this fact, note that the decomposition of a *cube* into subcubes with edge length h has $O\left(h^{-3}\right)$ parts, while the same decomposition of a square has only $O\left(h^{-2}\right)$ parts. This means that we need many more pixels until on the circles becomes significantly represented in the histogram. Nevertheless, the approach allows, in general, the detection of circles in images and a circle based Hough transform is actually available in `Matlab/octave` and there is no reason to *not* experiment with them.

Further applications of generalized Hough transforms in the sense of Definition 3.1.4 would be possible, but since the complexity grows exponentially in the number of parameters, there is a natural limit to this approach.

3.2 Time-Frequency – Windows & Gabor

It is impossible to practically compute the Fourier transform of a signal of *infinite* length and it is not easy and to determine it for a very long sequence $c \in \ell(\mathbb{Z}_n)$, n very large. Even if $n \log_2 n$ is considered a relatively small growth rate, transforming

[6] $\mathbb{S}^d = \left\{x \in \mathbb{R}^d \ : \ \|x\|_2 = 1\right\}$ stands for the d-dimensional **unit sphere**.

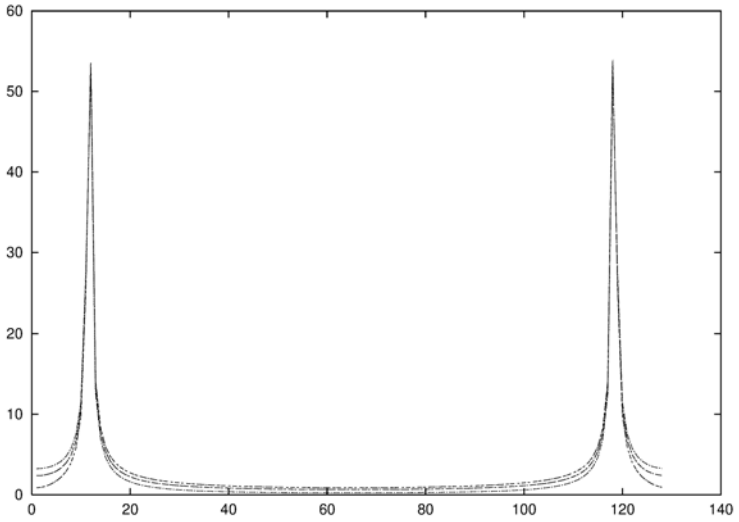


Figure 3.2.1: Absolute values of the FFTs of lentgh 128 with blurred freuquencies despite the fact that that only one frequency (and its mirrored version, we deal with a cosine) should appear. And even worse this effect varies with the invterval; though the variations are not dramatic, they are visible,

a long piece of music[7] used to be quite challenging for computers, and still is for small signal processing chips used in CD players or mp3 players. And even if we can compute it nowadays - what is the information obtained by this transform? It just tells us the distribution of frequencies in the piece but gives no information about melody or distribution of instruments.

3.2.1 The Windowed Fourier Transform

For these and other reasons it is common in signal processing to work not on the full signal but only on snippets of it that are obtained by using a filter[8]. To compute an FFT of length $n = 2^\ell$ we thus consider the **window**

$$\ell(\mathbb{Z}_n) \ni c_k = (Fc)(\cdot + kn) = (f * c)(\cdot + kn), \qquad k \in \mathbb{Z},$$

where the simplest filter would be $f = \delta$ which just cuts the signal into pieces of length n. This approach, however, can lead to problems.

Example 3.2.1. Consider $\cos(64x)$ sampled at the $n = 768 = 3 * 256$ points in $2\pi/768 * \mathbb{Z}_n$ and then transformed by FFT on the 6 intervals of length 128. In Fig. 3.2.1 we can clearly see that the frequency resolution is degraded, the peaks are blurred.

The phenomenon shown in Fig. 3.2.1 is typical for the **windowed Fourier transform** that we consider here: the frequencies leak and spread over a region instead

[7] A way file of an average piece of music is about 40MB, small in 2022, but still substantial in the good old days.

[8] This should not surprise us since almost all signal processing is based on filters.

of being sharply localized. This is known as the **leakage phenenomenon** In trying to understand the reason for this, we look at our naive δ-windows and compute the blocks

$$\begin{aligned}
[c\,(\cdot + 2kn)]_n^\wedge &= \left[\tau_{2kn}\left(c \times \sum_{j\in\mathbb{Z}_n} \tau_j\delta\right)\right]_n^\wedge = \underbrace{\omega^{2kn\cdot}}_{=1}\left(c \times \sum_{j\in\mathbb{Z}_n} \tau_j\delta\right)_n^\wedge \\
&= S_{2\pi/n}\left(c \times \sum_{j\in\mathbb{Z}_n} \tau_j\delta\right)^\wedge = S_{2\pi/n}\left[\hat{c} * \left(\sum_{j\in\mathbb{Z}_n} e^{i\,j\cdot}\right)\right] \\
&= \left[S_{2\pi/n}\hat{c}\right] * \left[\sum_{j\in\mathbb{Z}_n} e^{2\pi i j\cdot/n}\right],
\end{aligned}$$

which shows that even if $\hat{c}$ were *perfectly localized*, the convolution with the exponential sequences *always* would yield the leakage of the frequencies. In other words: this is a principal problem.

Nevertheless, the idea of windowing has a certain appeal: one will also detect the frequencies that appear *locally* in the signal during the time that is covered by the window. And even if this detection is not perfect due to the leakage effect, it may allow us, for example, to recognize the melody in a music recording. And if we look back to the localized DFT that we applied in Fig. 2.5.9, then we actually already did the same already to images. The only modification that we are going to make will be not to use successive, but *overlapping* windos.

3.2.2 The Gabor Transform

The Gabor transform was introduced by Denni Gábor (Gabor, 1946) in 1946 as an explicit technique for the analysis of *sound* data.

Definition 3.2.2 (Gabor transform). A **window function** is any normalized even function $g \in L_2(\mathbb{R})$, i.e., $g(t) = g(-t)$ sowie $\|g\|_2 = 1$. Based on the so called **time frequency atoms**

$$\phi_{u,\xi}(t) = e^{i\xi t}\, g(t-u), \qquad t \in \mathbb{R}, \qquad (u,\xi) \in \mathbb{R}^2, \tag{3.2.1}$$

for g, the **Gabor transform** is defined as

$$Gf(u,\xi) = T_\Phi f(u,\xi) = \int_{\mathbb{R}} f(t)\overline{\phi_{u,\xi}}(t)\,dt = \int_{\mathbb{R}} f(t)\, e^{-i\xi t}\, g(t-u)\,dt, \qquad (u,\xi) \in \mathbb{R}^2. \tag{3.2.2}$$

It is also called the **short time Fourier transform**, or **STFT** for short.

The "short" in STFT is an indiction that the underlying idea here is that g has a small compact support or decays very rapidly and practically has a small support. The Gabor transform yields the so called **spectrogram**

$$|Gf| = |Gf(u,\xi)| \;:\; \mathbb{R}^2 \to \mathbb{R}$$

that tells us which amount of frequency ξ was present at time u – at least approximately. The first observation is that the Gabor transform is a reasonable transform

since any function can be reconstructed from its Gabor transform and therefore no information is lost.

Theorem 3.2.3. *For $f \in L_2(\mathbb{R})$ one has*

$$f(t) = \frac{1}{2\pi} \int_{\mathbb{R}} \int_{\mathbb{R}} Gf\,(u,\xi)\; e^{i\xi t} g(t-u)\, d\xi du \tag{3.2.3}$$

and

$$\int_{\mathbb{R}} |f(t)|^2\; dt = \frac{1}{2\pi} \int_{\mathbb{R}} \int_{\mathbb{R}} |Gf\,(u,\xi)|^2\; d\xi du. \tag{3.2.4}$$

Remark 3.2.4. The Gabor transform works in nice analogy to the Fourier transform:

1. Equation (3.2.3) defines an **inverse Gabor transform** and that even in a very simple way and almost perfect analogy to the Fourier transform.

2. In the same way, (3.2.4) is the counterpiece of the Plancherel indenity (2.2.18) of the Fourier transform. In other words: up to the ubiquitous constant $\frac{1}{2\pi}$, the Gabor transform is an **isometry**.

3. A statement of warning might also be appropriate here: the formulas above hold in the L_2-sense, pointwise identities cannot be derived this way.

Proof: We fix $\xi \in \mathbb{R}$ an consider the functions $g_\xi = e^{i\xi\cdot} g$ and $f_\xi = Gf\,(\cdot,\xi)$. By Definition 3.2.2 and due to the symmetry of g,

$$\begin{aligned} f_\xi(u) &= Gf\,(u,\xi) = \int_{\mathbb{R}} f(t) e^{-i\xi t} \underbrace{g\,(t-u)}_{=g(u-t)}\; dt \\ &= \int_{\mathbb{R}} f(t) e^{-i\xi u}\, e^{i\xi(u-t)} g\,(u-t)\; dt = e^{-i\xi u} \int_{\mathbb{R}} f(t) g_\xi\,(u-t)\; dt, \end{aligned}$$

hence

$$f_\xi(u) = e^{-i\xi u} \left(f * g_\xi\right)(u), \qquad u \in \mathbb{R}, \tag{3.2.5}$$

and also

$$\begin{aligned} &(Gf(\cdot,\xi))^\wedge\,(\omega) = \hat{f}_\xi(\omega) \\ &= \int_{\mathbb{R}} e^{-i\omega u} e^{-i\xi u} \left(f * g_\xi\right)(u)\, du = \left(f * g_\xi\right)^\wedge\,(\xi+\omega) = \hat{f}(\xi+\omega)\, \hat{g}_\xi(\xi+\omega) \\ &= \hat{f}(\xi+\omega) \left(e^{-i\xi\cdot} g_\xi\right)^\wedge\,(\omega) = \hat{f}(\xi+\omega) \left(e^{-i\xi\cdot} e^{i\xi\cdot} g\right)^\wedge\,(\omega), \end{aligned}$$

which can be summarized as

$$(Gf(\cdot,\xi))^\wedge\,(\omega) = \hat{f}(\omega+\xi)\hat{g}(\omega), \qquad \omega \in \mathbb{R}. \tag{3.2.6}$$

Now we apply the Plancherel identity (2.2.17) to (3.2.3) and obtain by substitution of (3.2.6), da"s[9]

$$\begin{aligned}
&\frac{1}{2\pi}\int_{\mathbb{R}}\int_{\mathbb{R}} Gf\,(u,\xi)\; e^{i\xi t} g(t-u)\,d\xi du \\
&= \frac{1}{4\pi^2}\int_{\mathbb{R}} e^{i\xi t}\int_{\mathbb{R}} (Gf(\cdot,\xi))^{\wedge}(\omega)\overline{\underbrace{\big(g(t-\cdot)\big)^{\wedge}(\omega)}_{=g(\cdot-t)}}\,d\omega\,d\xi \\
&= \frac{1}{4\pi^2}\int_{\mathbb{R}} e^{i\xi t}\int_{\mathbb{R}} \hat{f}(\omega+\xi)\hat{g}(\omega)\,\overline{e^{-it\omega}\hat{g}(\omega)}\,d\omega\,d\xi \\
&= \frac{1}{4\pi^2}\int_{\mathbb{R}}\int_{\mathbb{R}} \hat{f}(\omega+\xi)\,|\hat{g}(\omega)|^2\; e^{it(\xi+\omega)}\,d\omega\,d\xi \\
&= \frac{1}{2\pi}\int_{\mathbb{R}} |\hat{g}(\omega)|^2 \underbrace{\frac{1}{2\pi}\int_{\mathbb{R}} \hat{f}(\omega+\xi)e^{it(\omega+\xi)}\,d\xi}_{\left(\hat{f}\right)^{\vee}(t)=f(t)}\,d\omega \\
&= f(t)\,\frac{1}{2\pi}\int_{\mathbb{R}} |\hat{g}(\omega)|^2\,d\omega = f(t)\int_{\mathbb{R}} |g(t)|^2\,dt = f(t)\,\|g\|_2^2 = f(t),
\end{aligned}$$

which proves (3.2.3). For (3.2.4) we also use (3.2.6) to conclude that

$$\begin{aligned}
&\frac{1}{2\pi}\int_{\mathbb{R}}\int_{\mathbb{R}} |Gf\,(u,\xi)|^2\,d\xi du \\
&= \frac{1}{4\pi^2}\int_{\mathbb{R}}\int_{\mathbb{R}} \left|(Gf\,(\cdot,\xi))^{\wedge}(\omega)\right|^2\,d\xi d\omega \\
&= \frac{1}{4\pi^2}\int_{\mathbb{R}} |\hat{g}(\omega)|^2\int_{\mathbb{R}} \left|\hat{f}(\omega+\xi)\right|^2\,d\xi\,d\omega = \frac{1}{4\pi^2}\int_{\mathbb{R}} |\hat{g}(\omega)|^2\,d\omega\int_{\mathbb{R}} \left|\hat{f}(\xi)\right|^2\,d\xi \\
&= \int_{\mathbb{R}} |g(t)|^2\,dt\int_{\mathbb{R}} |f(t)|^2\,dt = \|f\|_2^2\,\|g\|_2^2 = \|f\|_2^2
\end{aligned}$$

which is (3.2.4). □

Remark 3.2.5. The formula (3.2.6) for the Fourier transform $(Gf)^{\wedge}$ of Gf is not only a useful tool in the proof of Theorem 3.2.3, but also the key to an *efficient implementation.* Indeed, taking the inverse Fourier transform of (3.2.6), we find erhalten wir, da"s

$$Gf(u,\xi) = \left(\hat{f}\,(\cdot+\xi)\;\hat{g}\right)^{\vee}(u) \tag{3.2.7}$$

and after sampling f und g we can compute the Fourier transforms by means of the FFT, do a componentwise product and transform it back by yet another FFT, see (FFTW, 2003).

3.2.3 Time-Frequency Analysis

The Gabor transform is a first example of **time-frequency analysis** which tries to take into account both aspects of a signal by providing time localized frequency

[9]Originally this would require $f \in L_1(\mathbb{R})$, but since $L_1 \cap L_2$ is dense in L_2, the problem can be neglected or, better, overcome.

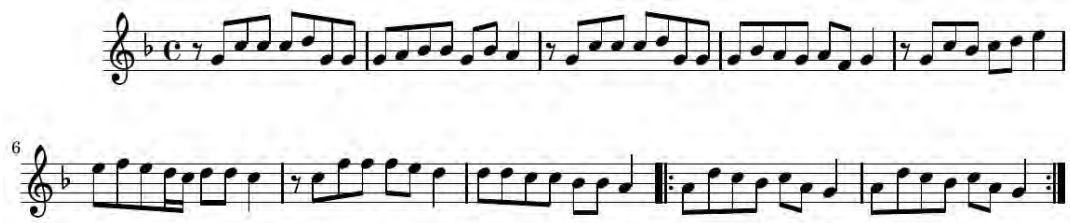


Figure 3.2.2: Time-frequency representation of a musical signal. The position of the dots denotes the pitch of the tone, i.e., its frequency, and the form or type of the note stands for its duration which also fixes the time during which the tone has to sound. Score typesetting by `lilypond`

content. These are indeed the two arguments u, ξ in the Gabor transform. A classical example of time-frequency analysis can be seen in Fig. 3.2.2. The musical score notation there tells us which frequency has to sound a which time and completely describes the signal. To understand better what goes on here, we need a bit more terminology-

Definition 3.2.6 (Time-/frequency localization). The **time localization** of a function $f \in L_2(\mathbb{R})$ is defined as[10]

$$\mu = \mu(f) = \frac{1}{\|f\|_2^2} \int_{\mathbb{R}} t\,|f(t)|^2\,dt, \tag{3.2.8}$$

its **frequency localization** as

$$\hat{\mu} = \hat{\mu}(f) = \frac{1}{\|\hat{f}\|_2^2} \int_{\mathbb{R}} \xi \left|\hat{f}(\xi)\right|^2 d\xi = \frac{1}{2\pi\,\|f\|_2^2} \int_{\mathbb{R}} \xi \left|\hat{f}(\xi)\right|^2 d\xi. \tag{3.2.9}$$

The **time variation** and the **frequency variation** are defined as

$$\sigma^2 = \frac{1}{\|f\|_2^2} \int_{\mathbb{R}} (t-\mu)^2\,|f(t)|^2\,dt, \qquad \hat{\sigma}^2 = \frac{1}{2\pi\,\|f\|_2^2} \int_{\mathbb{R}} (\xi-\hat{\mu})^2 \left|\hat{f}(\xi)\right|^2 d\xi, \tag{3.2.10}$$

respectively.

Remark 3.2.7. It is not obviousl why (3.2.8) is called a "localization", but nevertheless the explanation is not difficult: if $f \sim \delta_x$ for some $x \in \mathbb{R}$, which means that all mass of f is concentrated around the point x, say like $\operatorname{supp} f \subset [x-\varepsilon, x+\varepsilon]$, $\varepsilon > 0$, then

$$\begin{aligned} |\mu - x| &= \frac{1}{\|f\|_2^2} \left| \int_{\mathbb{R}} (t-x)\,|f(t)|^2\,dt \right| \le \frac{1}{\|f\|_2^2} \int_{x-\varepsilon}^{x+\varepsilon} |t-x|\,|f(t)|^2\,dt \\ &\le \frac{\varepsilon}{\|f\|_2^2} \int_{\mathbb{R}} |f(t)|^2\,dt = \varepsilon, \end{aligned}$$

hence $\mu \sim x$. In general μ and $\hat{\mu}$ could by called **barycenter** of f with respect to time and frequency, respectively.

[10] What you see here is also called the **first moment** of the function $|f|^2$.

A function has its (bary)center in time-frequency at the point $(\mu, \hat{\mu})$ and the variations $\sigma, \hat{\sigma}$ tell us how good this localization is. This motivates the following definition.

Definition 3.2.8. The region

$$H(f) := [\mu(f) - \sigma(f), \mu(f) + \sigma(f)] \times [\hat{\mu}(f) - \hat{\sigma}(f), \hat{\mu}(f) + \hat{\sigma}(f)] \subset \mathbb{R}^2 \quad (3.2.11)$$

is called **Heisenberg box** or **Heisenberg rectangle** of f.

The following classical result shows that the Heisenberg boxes cannot be arbitrarily small, more precisely, their *area* is bounded.

Theorem 3.2.9 (**Heisenberg uncertainty relation**). *For $f \in L_2(\mathbb{R})$,*

$$\sigma(f)\, \hat{\sigma}(f) \geq \frac{1}{2}. \quad (3.2.12)$$

In particular $\sigma(f) = 0$ as well as $\hat{\sigma}(f) = 0$ are impossible.

Proof: The proof from (Mallat, 1999), originally due to H. Weyl, makes the slightly stronger assumption that

$$\lim_{t \to \infty} \sqrt{t} f(t) = 0.$$

Since these functions are once more dense in $L_2(\mathbb{R})$, this is no restriction and the result coould be completed to $L_2(\mathbb{R})$ by a limit argument. Replacing f ny $g := e^{-i\hat{\mu}\cdot} f(\cdot + \mu)$, then $\|g\| = \|f\|$ and $\mu(g) = \hat{\mu}(g) = 0$, so that we can also assume that $\mu(f) = \hat{\mu}(g) = 0$.

With some Fourier computations and the the Schwarz inequality we then get

$$\begin{aligned}
\sigma^2(f)\, \hat{\sigma}^2(f) &= \frac{1}{2\pi \|f\|_2^4} \int_{\mathbb{R}} |t\, f(t)|^2 \, dt \int_{\mathbb{R}} \underbrace{\left|\xi \hat{f}(\xi)\right|^2}_{=\left|i\xi \hat{f}(\xi)\right|^2} d\xi \\
&= \frac{1}{2\pi \|f\|_2^4} \int_{\mathbb{R}} |t\, f(t)|^2 \, dt \int_{\mathbb{R}} \left|(f')^{\wedge}(\xi)\right|^2 d\xi = \frac{1}{\|f\|_2^4} \int_{\mathbb{R}} |t\, f(t)|^2 \, dt \int_{\mathbb{R}} |f'(t)|^2 \, dt \\
&\geq \frac{1}{\|f\|_2^4} \left(\int_{\mathbb{R}} |t\, f(t)\, f'(t)| \, dt \right)^2 \geq \frac{1}{\|f\|_2^4} \left(\int_{\mathbb{R}} \frac{t}{2} \left(f^2(t)\right)' \, dt \right)^2 \\
&= \frac{1}{4\|f\|_2^4} \left(\int_{\mathbb{R}} t \left(f^2(t)\right)' \, dt \right)^2 = \frac{1}{4\|f\|_2^4} \left(\left[t f^2(t)\right]_{t=0}^{\infty} - \int_{\mathbb{R}} |f(t)|^2 \, dt \right)^2 \\
&= \frac{1}{4\|f\|_2^4} \left(\int_{\mathbb{R}} |f(t)|^2 \, dt \right)^2 = \frac{1}{4},
\end{aligned}$$

as claimed in (3.2.12). □

Remark 3.2.10 (Heisenberg uncertainty).

1. Die musician version of Theorem 3.2.9 is:

   > One cannot play a jig on the bass pedals of an organ.

   A fast piece of music, and a jig is a fast dance in a 6/8 meter, requires a good *time localization* of the tone, hence a very small value for $\sigma(f)$, due to which $\hat{\sigma} \geq (2\sigma(f))^{-1}$ has to be large. For high frequencies this is much less troublesome that for low ones, since a variation in pitch is always a *relative* modification and not an *absolute* one[11] and hence affects low frequencies much more.

2. This also shows that our tone reception and our potential to produce tones are limited due to principal reasons.

3. Keep in mind that a permanent tone with constant frequency and no beginning and end satisfies $\sigma(f) = \infty$ which indeed yields that $\hat{\sigma}(f) = 0$. This is the perfect frequency localization of the Fourier transform, but it requires knowledge of the signal over infinite time and is not applicable in practice.

4. For inequalities like (3.2.12) it is good to know in which situations equality occurs as the functions for which this happens are the ones of *optimal* time-frequency localization. For Heisenberg uncertainty they can be given explicitly, namely, all functions of the form

   $$f(t) = a\, e^{i\omega t - b\,(t-u)^2} = a\, e^{i\omega t}\, e^{-b(t-u)^2}, \qquad u, \omega \in \mathbb{R},\ a, b \in \mathbb{C}. \tag{3.2.13}$$

   Functions of that type are called **modulated Gaussian** as the leading term $e^{i\omega t}$ stands for a **phase shift**, also called **modulation**.

3.3 Wavelets

A modern and powerful method to perform time-/frequency analysis is the *continuous wavelet transform* which we will consider now. A little bit of care is worthwhile since they are introduced and treated in slightly different ways in the standard literature, for example (Daubechies, 1992; Holschneider, 1995; Louis et al., 1998; Mallat, 1999). By the way: a somewhat journalistic but very accesible introduction to the topic can be be found in (Hubbard, 1996).

Definition 3.3.1 (Wavelets & admissibility). A complex valued[12] function $\psi \in L_2(\mathbb{R})$ is called **wavelet**[13], if it has mean value zero, i.e, if

$$\int_{\mathbb{R}} \psi(t)\, dt = 0. \tag{3.3.1}$$

[11] For example, a deviation of *10 cent* which is usually accepted in tuning means that two frequencies ω, ω' satisfy $2^{-1/120}\omega \leq \omega' \leq 2^{1/120}\omega$, or $\omega' \in \left[2^{-1/120}, 2^{1/120}\right]\omega$, respectively, and the width of this tolerance interval obviously depends ω.

[12] It makes sense to admit complex valued functions. On the one hand, they can be easily handled by treating the real and imaginary part separately and, on the other hand, the most famous and most frequently used wavelet *is* a complex valued one.

[13] Originally **ondelette** (French) which also means “small wave”

A wavelet is called **normalized** if $\|\psi\|_2 = 1$ and **admissible** if

$$C_\psi := \int_{\mathbb{R}} \frac{\left|\hat{\psi}(\xi)\right|^2}{|\xi|}\, d\xi < \infty. \tag{3.3.2}$$

The name "wavelet" for a function with zero mean value is due to the inuition that some part of the function must be below the x-axis, some part above, hence the function has a certain wavelike shape.

The **admissibility condition** (3.3.2) in particular requires that da"s

$$0 = \hat{\psi}(0) = \int_{\mathbb{R}} \psi(t)\, dt,$$

hence any admissible function is a wavelet. This is why admissibility is sometimes part of the wavelet definition, otherwise one distinguishes between a wavelet and an **admissible wavelet**. If ψ is a *real wavelet*, i.e., $\psi : \mathbb{R} \to \mathbb{R}$, then

$$\hat{\psi}(-\xi) = \int_{\mathbb{R}} \psi(t)\, e^{i\xi t}\, dt = \int_{\mathbb{R}} \overline{\psi(t)\, e^{-i\xi t}}\, dt = \overline{\hat{\psi}(\xi)}$$

and (3.3.2) slightly simplifies into

$$\begin{aligned} \infty \;&>\; \int_{\mathbb{R}} \frac{\left|\hat{\psi}(\xi)\right|^2}{|\xi|}\, d\xi = \int_0^\infty \frac{\left|\hat{\psi}(\xi)\right|^2}{|\xi|}\, d\xi + \int_0^\infty \frac{\left|\hat{\psi}(-\xi)\right|^2}{|-\xi|}\, d\xi \\ &=\; 2\int_0^\infty \frac{\left|\hat{\psi}(\xi)\right|^2}{\xi}\, d\xi, \end{aligned}$$

hence

$$C_\psi = \int_0^\infty \frac{\left|\hat{\psi}(\xi)\right|^2}{\xi}\, d\xi < \infty. \tag{3.3.3}$$

Definition 3.3.2 (Wavelet transform). For a normalized wavelet ψ and $f \in L_2(\mathbb{R})$ the **wavelet transform** is defined as

$$W_\psi f(u,s) := \int_{\mathbb{R}} f(t)\, \frac{1}{\sqrt{|s|}} \overline{\psi\left(\frac{t-u}{s}\right)}\, dt, \qquad (u,s) \in \Gamma = \mathbb{R} \times \mathbb{R}. \tag{3.3.4}$$

The term $1/\sqrt{|s|}$ is there to ensure that also the *scaled* wavelet

$$\psi_s := \frac{1}{\sqrt{|s|}} \psi\left(s^{-1}\cdot\right), \qquad s \in \mathbb{R}, \tag{3.3.5}$$

is normalized:

$$\|\psi_s\|_2 = \frac{1}{|s|} \int_{\mathbb{R}} |\psi(t/s)|^2\, dt = \int_{\mathbb{R}} |\psi(t)|^2\, dt = \|\psi\|_2 = 1.$$

But let us first have a look at classical examples for wavelets.

Example 3.3.3 (Classical wavelets).

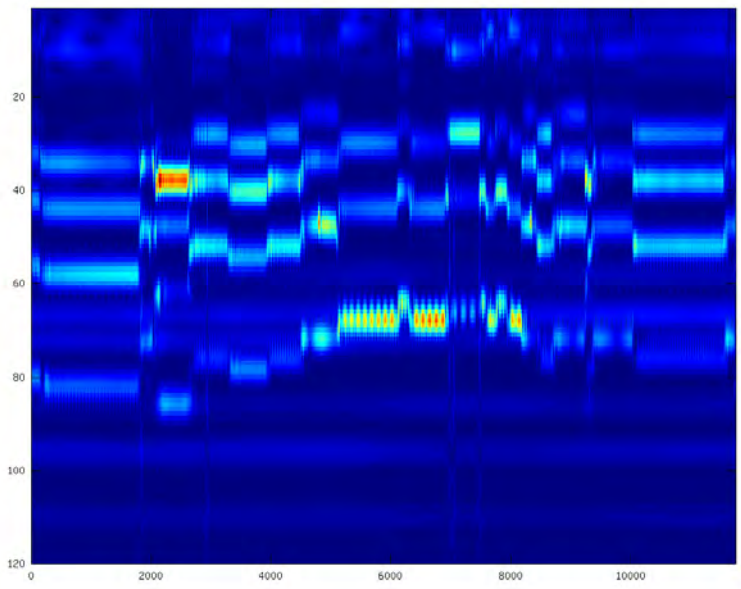


Figure 3.3.1: **Scalogram** of a fragment of the melody from Fig. 3.2.2, with a sampling frequency of 8000Hz. The scalogram plots the color coded modulus $\left|W_\psi f(u,s)\right|$ relative to u and s. The wavelet used here is the *Morlet wavelet* from Example 3.3.3

1. The **Haar wavelet** is the discontinuous function

$$\psi := \chi(\cdot + 1) - \chi = \begin{cases} 1, & x \in [-1, 0), \\ -1, & x \in (0, 1], \\ 0, & \text{sonst}, \end{cases} \tag{3.3.6}$$

   where $\chi = \chi_{[0,1]}$. Since $\|\psi\|_2 = \sqrt{2}$, it is not normalized[14], but it is the only "classical" wavelet with *compact support.*

2. The **mexican hat wavelet** is defined as

$$\psi(t) := \left(1 - t^2\right) e^{-t^2/2} = -\frac{d^2}{dt^2} e^{-t^2/2}, \qquad t \in \mathbb{R}, \tag{3.3.7}$$

   and has no compact support, but decays exponentially for $x \to \infty$ which is responsible for its good time localization. Its Fourier transform has the form

$$\hat{\psi}(\xi) = \sqrt{2\pi}\, \xi^2\, e^{-\xi^2/2}$$

   and thus practically coincides with the wavelet itself.

3. The **Morlet wavelet**, also known as **Morlet's Gaussian wavelet**, see (Holschneider, 1995), is the complex sibling of the Morlet wavelet and combibes its decay rate with a **phase modulation**:

$$\psi(t) = e^{i\omega t}\, e^{-t^2/2}, \qquad t \in \mathbb{R}, \qquad \omega \in \mathbb{R}_+. \tag{3.3.8}$$

[14] But of course it is not difficult to normalize it.

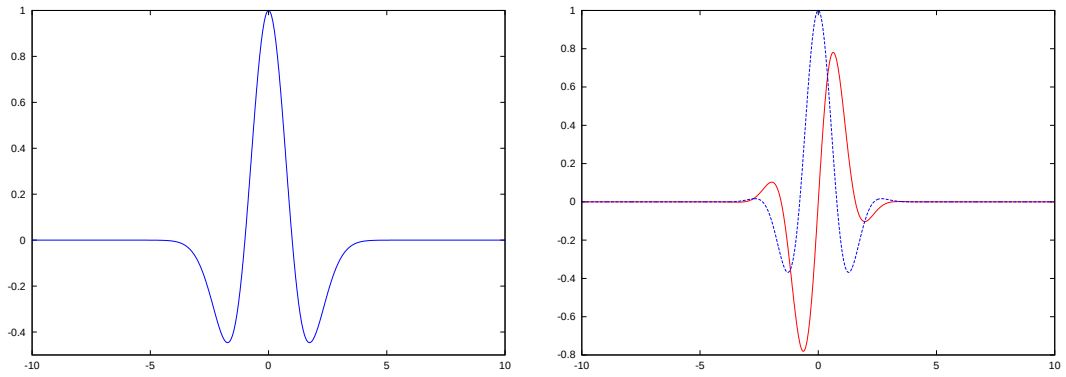


Figure 3.3.2: The mexican hat wavelet *(left)* adn the Morlet wavelet *(right)* with real and imaginary part plotted separately.

The frequency ω is a shape parameter for the wavelet to control how "wiggly" it is. Strictly speaking, the Morlet wavelet as given in (3.3.8) is *not* a wavelet since $\int \psi \neq 0$, but this can be easily cured by an appropriate correction term, cf. (Mallat, 2009).

Remark 3.3.4. That the mexican hat wavelet is formed from the function $e^{-(\cdot)^2/2}$ is no accident, of course. If we recall Remark 3.2.10, then we realize that the function is of the form (3.2.13) and thus provides optimal time-/frequency resolution. The second derivative in (3.3.7) has a different reason.

Exercise 3.3.1 Compute the Fourier transform of the Morlet wavelet and of the mexican hat wavelet. ♢

The reason why admissibility has been introduced is in the following theorem: provided that we use an admissible wavelet, the wavelet transform is invertible.

Theorem 3.3.5 (Inverse wavelet transform). *For a normalized admissible wavelet ψ and $f \in L_2(\mathbb{R})$ we have the* ***inverse wavelet transform***

$$f(t) = \frac{1}{C_\psi} \int_{\mathbb{R}} \int_{\mathbb{R}} W_\psi f(u,s) \, \frac{1}{\sqrt{|s|}} \psi\left(\frac{t-u}{s}\right) du \frac{ds}{s^2} \tag{3.3.9}$$

Proof: First, we realize that the Fourier transform of the complex conjugate of a function takes the form

$$\left(\overline{f}\right)^\wedge(\xi) \;=\; \int_{\mathbb{R}} \overline{f(t)} e^{-i\xi t} \, dt = \overline{\int_{\mathbb{R}} f(t) \, e^{i\xi t} \, dt} = \overline{\hat{f}(-\xi)}.$$

In the first step of our proof we determine, like in the proof of Theorem 3.2.3, the Fourier transform of the wavelet transform as

$$\begin{aligned} \left(W_\psi f(\cdot, s)\right)^\wedge(\xi) &= \left(\int_{\mathbb{R}} f(t) \, \overline{\psi_s(t-\cdot)} \, dt \right)^\wedge(\xi) \\ &= \left(f * \overline{\psi_s(-\cdot)} \right)^\wedge(\xi) = \hat{f}(\xi) \, \overline{\hat{\psi}_s(\xi)}, \end{aligned}$$

hence,

$$\left(W_\psi f(\cdot, s)\right)^\wedge(\xi) = \sqrt{s}\,\hat{f}(\xi)\,\overline{\hat{\psi}(s\xi)}, \qquad \xi \in \mathbb{R}, \quad s \in \mathbb{R}, \tag{3.3.10}$$

and consequently, for $s \in \mathbb{R}$,

$$\begin{aligned}
&\int_{\mathbb{R}} W_\psi f(u,s)\,\frac{1}{\sqrt{s}}\psi\left(\frac{t-u}{s}\right)\,du \\
&= \frac{1}{2\pi}\int_{\mathbb{R}} \left(W_\psi f(\cdot,s)\right)^\wedge(\xi)\;\left(\psi_s\,(t-\cdot)\right)^\wedge(\xi)\,d\xi \\
&= \frac{1}{2\pi}\int_{\mathbb{R}} |s|\,\hat{f}(\xi)\,\hat{\psi}(s\xi)e^{i\xi t}\,\overline{\hat{\psi}(s\xi)}d\xi = \frac{|s|}{2\pi}\int_{\mathbb{R}} e^{i\xi t}\,\hat{f}(\xi)\,\left|\hat{\psi}(s\xi)\right|^2\,d\xi.
\end{aligned}$$

Changing the order of integration and performing a change of variables $\omega = s\xi$, we thus get for the full integral that

$$\begin{aligned}
&\frac{1}{C_\psi}\int_{\mathbb{R}}\int_{\mathbb{R}} W_\psi f(u,s)\,\frac{1}{\sqrt{s}}\psi\left(\frac{t-u}{s}\right)\,du\frac{ds}{s^2} \\
&= \frac{1}{2\pi\,C_\psi}\int_{\mathbb{R}} \hat{f}(\xi)e^{i\xi t}\int_{\mathbb{R}}\frac{\left|\hat{\psi}(s\xi)\right|^2}{|s|}\,ds\,d\xi = \frac{1}{2\pi\,C_\psi}\underbrace{\int_{\mathbb{R}}\hat{f}(\xi)e^{i\xi t}\,d\xi}_{=\left(\hat{f}\right)^\vee(t)=f(t)}\underbrace{\int_{\mathbb{R}}\frac{\left|\hat{\psi}(\omega)\right|^2}{|\omega|}\,d\omega}_{=C_\psi} \\
&= f(t),
\end{aligned}$$

as claimed. □

Taking into account that for a *real valued* ψ one has $\left|\hat{\psi}(-\xi)\right|^2 = \left|\hat{\psi}(\xi)\right|^2$, the same method yields a somewhat simpler inversion formula for a real wavelet ψ, cf. (Mallat, 1999).

Corollary 3.3.6. *For a real wavelet ψ and $f \in L_2(\mathbb{R})$,*

$$f(t) = \frac{1}{C_\psi}\int_0^\infty\int_{\mathbb{R}} W_\psi f(u,s)\,\frac{1}{\sqrt{s}}\psi\left(\frac{t-u}{s}\right)\,du\frac{ds}{s^2}, \qquad C_\psi = \int_0^\infty \frac{\left|\hat{\psi}(\xi)\right|^2}{\xi}\,d\xi. \tag{3.3.11}$$

Remark 3.3.7 (Wavelet transform & inversion formula).

1. Due to the way we formulated and proved it, the inversion formula (3.3.9) only holds in the L_2 sense and *not* pointwise; for that purpose we would need more refined arguments. And if we look carefully at the proof, then we see that we quite carelessly changed the order of integration and almost divided by zero. More careful (and correct) versions of the inversion formula with appropriate proofs can be found for example in (Daubechies, 1992) or (Louis et al., 1998).

2. From (3.3.10) one can see that the wavelet transform is *redundant* as for $s, s' \in \mathbb{R}_+$ and $\xi \in \mathbb{R}$ we obtain

$$\hat{f}(\xi) = \frac{\left(W_\psi f\,(\cdot,s)\right)^\wedge(\xi)}{\sqrt{s}\hat{\psi}(s\xi)} = \frac{\left(W_\psi f\,(\cdot,s')\right)^\wedge(\xi)}{\sqrt{s'}\hat{\psi}(s'\xi)},$$

that is,

$$\left(W_\psi f\,(\cdot, s)\right)^\wedge(\xi) = \sqrt{\frac{s}{s'}}\,\frac{\hat{\psi}\,(s\xi)}{\hat{\psi}\,(s'\xi)}\,\left(W_\psi f\,(\cdot, s')\right)^\wedge(\xi), \tag{3.3.12}$$

which is valid as long as $\hat{\psi}(s'\xi) \neq 0$. In other words:

> Knowing the wavelet transform for *a single* scale s means to know it for practically and s.

3. From an intuitive point of view this redundancy makes sense: why should a one dimensional signal f yield a fully two dimensional transformation? This would make it some space filling curve like the *Peano curve* (Gelbaum and Olmstedt, 1964) and not very exciting from a practical point of view.

4. The appearance of the $1/s^2$ term in the inverse wavelet transform is a little surprising in the beginning but one could take the point of view that it just makes the proof work. But of course, this is only half of the truth and there are fundamental reasons why is has to be exactly this term, see (Grossmann et al., 1985). All this has to do with abstract Harmonic Analysis, integration on locally compact abelian groups, dual groups and the realization of the affine group. And though this is fairly abstract stuff[15], it is extremely useful and the necessary basis to *really* understand what is going on here.

To better understand the time-frequency behavior of wavelet, we next have a look at the Heisenberg boxes of the wavelet transform. To that end, we have to consider the **time-frequency atoms**

$$\psi_{u,s} = \psi_s(\cdot - u) = \psi\left(\frac{\cdot - u}{s}\right), \qquad (u, s) \in \mathbb{R} \times \mathbb{R}_+.$$

Since for arbitrary $f \in L_2(\mathbb{R})$ and $u \in \mathbb{R}$,

$$\mu\left(f(\cdot - u)\right) = \int_{\mathbb{R}} t\,|f(t-u)|^2\,dt = \int_{\mathbb{R}} (t+u)\,|f(t)|^2\,dt = \mu(f) + u$$

we can always assume that after a proper shift ψ is a **centered wavelet**, i.e., that $\mu(\psi) = 0$. Centered wavelets are also important from a practical point of view since only then $W_\psi f(u, s)$ connects to $f(u)$, otherwise there could be a shift in time between the signal and the wavelet transform. The frequency localization $\hat{\mu}(\psi)$, on the other hand, is a constant that describes the inherent oscillation of the wavelet.

Proposition 3.3.8. *The Heisenberg box $H\left(\psi_{u,s}\right)$, $u \in \mathbb{R}$, $s \in \mathbb{R}_+$, of a centered wavelet ψ has the midpoints $\left(u, s^{-1}\hat{\mu}(\psi)\right)$ and the edge lengths $s\sigma(\psi)$ as well as $\hat{\sigma}(\psi)/s$.*

[15] And it takes half a semester to learn it.

Proof: Purely computational:

$$\begin{aligned}\mu\left(\psi_{u,s}\right) &= \int_{\mathbb{R}} t\left|\frac{1}{\sqrt{s}}\psi\left(\frac{t-u}{s}\right)\right|^2 dt = \frac{1}{s}\int_{\mathbb{R}}(t+u)\left|\psi\left(\frac{t}{s}\right)\right|^2 dt \\ &= \int_{\mathbb{R}} st\left|\psi(t)\right|^2 dt + u\int_{\mathbb{R}}\left|\psi(t)\right|^2 dt = s\mu(\psi)+u = u,\end{aligned}$$

as well as

$$\begin{aligned}\hat{\mu}\left(\psi_{u,s}\right) &= \int_{\mathbb{R}} \xi\left|\hat{\psi}_{u,s}\right|^2 d\xi = \int_{\mathbb{R}} \xi\left|e^{i\xi u}\,\hat{\psi}_s\right|^2 d\xi = \int_{\mathbb{R}} \xi\left|\sqrt{s}\hat{\psi}\left(s\xi\right)\right|^2 d\xi \\ &= \int_{\mathbb{R}} s\xi\left|\hat{\psi}\left(s\xi\right)\right|^2 d\xi = \frac{1}{s}\int_{\mathbb{R}}\xi\left|\hat{\psi}\left(\xi\right)\right|^2 d\xi = \frac{\hat{\mu}(\psi)}{s}.\end{aligned}$$

For the variances we get in essentially the same way that

$$\begin{aligned}\sigma^2\left(\psi_{u,s}\right) &= \int_{\mathbb{R}}(t-\mu_t)^2\left|\frac{1}{\sqrt{s}}\psi\left(\frac{t-u}{s}\right)\right|^2 dt \\ &= \int_{\mathbb{R}}(t-u)^2\left|\frac{1}{\sqrt{s}}\psi\left(\frac{t-u}{s}\right)\right|^2 dt = \frac{1}{s}\int_{\mathbb{R}} t^2\left|\psi\left(\frac{t}{s}\right)\right|^2 dt \\ &= \int_{\mathbb{R}}(st)^2\left|\psi\left(t\right)\right|^2 dt = s^2\,\sigma^2\left(\psi\right)\end{aligned}$$

and

$$\begin{aligned}\hat{\sigma}^2\left(\psi_{u,v}\right) &= \int_{\mathbb{R}}\left(\xi-s^{-1}\mu_\xi\right)^2\left|\hat{\psi}_{u,s}(\xi)\right|^2 d\xi = s\int_{\mathbb{R}}\left(\xi-s^{-1}\mu_\xi\right)^2\left|\hat{\psi}(s\xi)\right|^2 d\xi \\ &= \int_{\mathbb{R}}\left(\frac{\xi-\mu_\xi}{s}\right)^2\left|\hat{\psi}(\xi)\right|^2 d\xi = \frac{1}{s^2}\int_{\mathbb{R}}\left(\xi-\mu_\xi\right)^2\left|\hat{\psi}(\xi)\right|^2 d\xi = \frac{\hat{\sigma}^2(\psi)}{s^2}.\end{aligned}$$

□

Corollary 3.3.9. *All Heisenberg boxes of the wavelet transform cover the same area.*

Proof:

$$s\,\sigma(\psi)\times\frac{\hat{\sigma}(\psi)}{s} = \sigma(\psi)\,\hat{\sigma}(\psi),$$

which is precisely the term appearing in the uncertainty relation (3.2.12). □

Exercise 3.3.2 Show that the Heisenberg boxes for the atoms $\phi_{u,\xi}$ of the Gabor transform (3.2.2) are of the form

$$H_{u,\xi} = \left[u-\sigma(\phi), u+\sigma(\phi)\right]\times\left[\xi-\hat{\sigma}(\phi), \xi+\hat{\sigma}(\phi)\right]. \tag{3.3.13}$$

◊

Remark 3.3.10. Proposition 3.3.8 and (3.3.13) show the fundamental difference between the Gabor transform and the wavelet transform: while the Gabor transform works with a *constant* time and frequency resolution, the wavelet transform uses a *relative* resolution in time and frequency. In regions of high frequency, i.e.,

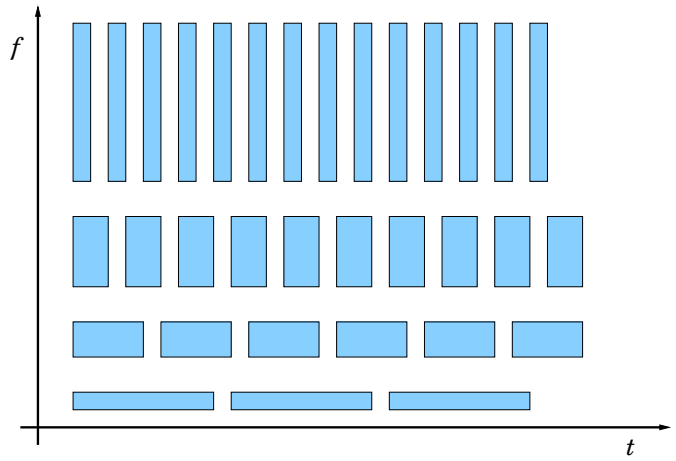


Figure 3.3.3: Schematic representation of the Heisenberg boxes for a wavelet transform. Higher values of the y axis correspond to higher frequencies which are approximated by $1/s$. High frequency boxes become narrow and high, low frequency boxes wide and flat.

for a *small* scale parameter s, the time resolution is higher while the absolute precision of the frequency decays. In regions of low frequency, it is the opposite, here the frequency is met more precisely while the signal cannot be located that well in time. This way of analysis is in accordance with our perception of sound, in fact, a tone show sound for a few periods to be recognized as such.

The nature of the Heisenberg boxes is depicted in Fig. 3.3.3. To obtain a uniform covering of the plane with such rectangley, it is natural to choose the scales *geometrically*, i.e., as

$$s_j = s_0\,\sigma^j, \qquad j \in \mathbb{Z}_M, \qquad \sigma > 1, \tag{3.3.14}$$

with an **initial scale** s_0 and **progression** σ. This actually has a lot of advantages that can even be justified and explaine numerically, cf. (Klein, 2011). Another nice remark is that the choice (3.3.14) of frequencies also corresponds with the *tempered scale* in setting musical intervals.

There remains the question of how to choose s_0, σ and M, for which we can at least give some rules of thumb:

1. The lowest scale s_0 corresponds via $\xi \sim 1/s_0$ to the *highest* frequency contained in the signal. Here the Shannon Sampling Theorem gives a lower bound for the frequency in terms of the sampling frequency or the Nyquist frequency ξ^*, respectively. This leads to an upper bound $s_0 \le 1/\xi^*$ for s_0.

2. The highest scale $s_{M-1} = s_0\sigma^{M-1}$ should still remain so small that the essential part[16] of the wavelet lies *within* the sampled region since otherwise the

[16]Keep in mind that most wavelets do not have compact support, but decay fast, so that some *essential support* Ω can be defined by requiring

$$\int_\Omega |\psi(t)|\; dt \ge \rho\|\psi\|_1$$

for some $\rho \in (1-\varepsilon, 1)$, $\epsilon > 0$.

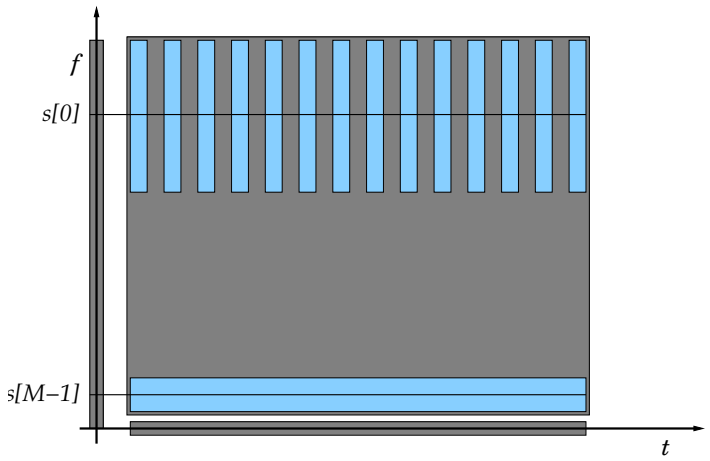


Figure 3.3.4: Schematic representation of the scale restrictions. The widest Heisenberg box on the bottom should still lie within the sampling region, the narrow boxes on top, on the other hand, still lie within the frequency range determined by the sampling frequency.

value of the wavelet transform (3.3.4) would depend significanlty on unknown values of f.

3. The rest is an interaction between σ and M and mainly affects the computational effort. Once the minimal and maximal scale are fixed, one can either choose the number M of scales and obtain σ or vice versa.

The meaning of the rules of thumb is depicted in Fig. 3.3.4. The way how the signal is sampled yields a time-frequency window for reasonable scales. The time window is determined by the sampling region $[t_0, t_{N-1}]$, the frequency window by the sampling theorem from the sampling step width h. And whenever a Heisenberg box is *not* contained in this frame, it is affected by phantom components that are not contained in the sampled signal.

3.3.1 Implementation of an FWT

Having settled the theory, we now head into another, more practical direction and look how the wavelet transform can be computed *numerically* in an efficient way. To this end, we first remark that the function to be transformed usually is not available explicitly, but in *sampled* form

$$f\left(t_j\right), \qquad t_0 < \cdots < t_N, \qquad N \in \mathbb{N},$$

and that there are only *finitely many* samples. In many situations

1. the sampling positions t_j are **equidistant**, that is, $t_j = t_0 + jh$, $h > 0$,

2. the number N of samples is not fixed a priori,

3. the sampling points can neither be modified nor influenced.

A naive way to compute the wavelet transform could be to use these points t_j and to approximate the integral by a **quadrature formula**

$$W_\psi(s,u) \sim \sum_{j=0}^{N} w_j\, f\left(t_j\right)\, \frac{1}{\sqrt{s}}\overline{\psi\left(\frac{t_j-u}{s}\right)}, \qquad w_j > 0, \tag{3.3.15}$$

see (Isaacson and Keller, 1966; Gautschi, 1997; Sauer, 2000a). In the simplest case is the **rectangle rule** where $w_j = t_{j+1} - t_j$, $j = 0, \ldots, N-1$, and appropriate w_N, for equidistant knots we may even take $w_j = h$, $j = 0, \ldots, N$. What happens in this case is just an integration of the piecewise constant function whose values at the knots are defined by the values of the function to be integrated. For fixed scale s we then have, for any u, a computational effort of order $O(N)$ and if we use this to determine the wavelet transform at $u_j = t_j$, $j = 0, \ldots, N$, we have a total effort of $O\left(N^2\right)$ per scale.

Moreover, we have to evaluate the wavelet ψ at the positions $\frac{t_j - t_k}{s}$, $j, k = 0, \ldots, N$. This is not so bad on the one hand since the wavelets are known *explicitly* in many cases, see Example 3.3.3, but it is at least reasonable to center the wavelet around the origin. A more tricky side effect is that this type of evaluation restricts the range of scales: good wavelets are usually *well localized* in time *and* frequency which means that

$$\lim_{x\to\pm\infty} \psi(x) = 0, \qquad \lim_{\xi\to\pm\infty} \psi(\xi) = 0,$$

and this rate of decay can be an exponential one, see again Example 3.3.3. If now, for such a wavelet, the scale s is small enough, then the quadrature formula (3.3.15) takes the undesired form

$$W_\psi\left(s,t_j\right) \sim w_j\, f\left(t_j\right)\, \frac{1}{\sqrt{s}}\overline{\psi\left(0\right)}. \tag{3.3.16}$$

Nevertheless, the implementation is simple if we use the convolution structure and turn it into a product of Fourier transforms. To do so, we simply use our previous computations, substitute (2.5.8) into (3.3.10) and obtain

$$\left(W_\psi f(s,\cdot)\right)^\wedge\left(\frac{2\pi k}{N}\right) \sim \left(W_\psi f_\varphi(s,\cdot)\right)^\wedge\left(\frac{2\pi k}{N}\right) = \sqrt{s}\, h\, \hat{f}_h(k)\, \hat{\varphi}\left(\frac{2\pi k}{h\,N}\right) \hat{\psi}\left(\frac{2\pi k s}{N}\right),$$

or

$$\mathrm{DFT}\left(\sigma_h W_\psi f(s,\cdot)\right) \sim \sqrt{s}\, h\, \hat{f}_h(k)\, \hat{\varphi}\left(\frac{2\pi k}{h\,N}\right) \hat{\psi}\left(\frac{2\pi k s}{N}\right), \tag{3.3.17}$$

respectively. Then an inverse DFT or inverse FFT transforms (3.3.17) into the computational rule

$$\left[W_\psi\left(s,t_j\right)\ :\ j \in \mathbb{Z}_N\right] \leftarrow \sqrt{s} h\ \mathrm{IDFT}\left[\hat{f}_h(k)\, \hat{\varphi}\left(\frac{2\pi k}{h\,N}\right) \hat{\psi}\left(\frac{2\pi k s}{N}\right)\ :\ k \in \mathbb{Z}_N\right], \tag{3.3.18}$$

that could be called a **fast wavelet transform** or **FWT** if this name would not be already reserved for some different concept in the context of filter banks.

Remark 3.3.11 (Fast computation of the continuous wavelet transform). The fast continuous wavelet transform, **FCWT**, has some interesting properties:

1. The computation of the wavelet transform only costs $O(N \log N)$ per scale and the values $\hat{\phi}$ of the quasi interpolation filter are independent of f and s and can thus be precomputed and stored in a table.

2. The computations for individual scales are independent of each other and thus can be parallelized very easily and efficiently. In particular, it is possible to perform the computation on a GPU or an FPGA which allows for realization in real time.

3. The Fourier transform $\hat{\psi}$ of the wavelet has to be sampled for each s, but for most relevant wavelets the Fourier transform is even known explicitly. Often wavelets are even designed in the frequency domain just like we did in the filter design in Section 2.4.2.

3.3.2 The Inverse Transform and its Catches

Once we can realized the wavelet transform in an efficient way, we may also try to implement the inverse transform. To that end, we have one more look at the formula (3.3.9),

$$f = \frac{1}{C_\psi} \int_{\mathbb{R}} \underbrace{\int_{\mathbb{R}} W_\psi f(s,u) \frac{1}{\sqrt{|s|}} \psi\left(\frac{\cdot - u}{s}\right) du}_{= W_\psi f(s,\cdot) * \psi_s = \left(\hat{W}_\psi(s,\cdot)\, \hat{\psi}_s\right)^\vee} \frac{ds}{s^2}$$

and compute it once more by means of an FFT. All that is left, is the integral $\int \frac{ds}{s^2}$, for which we have no choice but to incorporate a quadrature formula. The knots of this formula are the scales s_j for which we computed the wavelet transform, since these are the values for which we assume that we know it at these discrete locations

$$W_\psi f\left(s_j, t_k\right), \qquad j \in \mathbb{Z}_M,\ k \in \mathbb{Z}_N. \tag{3.3.19}$$

This inversion, first considered in (Domes, 2007), see (Sauer, 2011), determines a matrix [17] in the sense of (3.3.17) as

$$F := \sqrt{s} h \left[\mathrm{IDFT}\left(\left(W_\psi f\left(s_j, \cdot\right)\right)^\wedge_h \hat{\varphi}\left(\frac{2\pi \cdot}{h N}\right) \hat{\psi}\left(\frac{2\pi \cdot s_j}{h N}\right) \right)(k) \ : \ \begin{matrix} j \in \mathbb{Z}_M \\ k \in \mathbb{Z}_N \end{matrix} \right].$$

These are the values to be integrated with respect to s, i.e., summed up with respect to the s_j, which finally yields the values of the function:

$$f(t_k) = \sum_{j \in \mathbb{Z}_M} w_j F_{jk} = w^T F.$$

[17] Or a vector of vectors.

The *weights* w_j of the quadrature formula can be determined for example by a composite rectangle rule as

$$w_j = \int_{s_j}^{s_{j+1}} \frac{ds}{s^2}, \qquad j \in \mathbb{Z}_M,$$

for an appropriate s_M that is chosen such that the constant function is reproduced by the formual. Now our natural choice konstante Funktionen exakt reproduziert. Jetzt (3.3.14) turns out to be useful as it yields

$$w_j = \int_{s_0\sigma^j}^{s_0\sigma^{j+1}} \frac{ds}{s^2} = -\frac{1}{s}\Big|_{s_j}^{s_j\sigma} = \frac{1}{s_j}\left(1-\sigma^{-1}\right),$$

that is,

$$w = \frac{\sigma-1}{\sigma}\left[s_j^{-1} \;:\; j \in \mathbb{Z}_M\right]. \tag{3.3.20}$$

Quadrature formulas of higher order can be adapted analogously.

Exercise 3.3.3 Compute w for the composite trapezoidal rule. ◊

If we look carefully at (3.3.10) we see that our inversion fomula appears to be overly complicated since (3.3.10) can be refomulated as

$$\hat{f}(\xi) = \frac{\left(W_\psi f(s,\cdot)\right)^\wedge(\xi)}{\sqrt{s}\hat{\psi}(s\xi)}, \qquad \xi \in \mathbb{R}\setminus\{0\}, \quad s \in \mathbb{R}_+,$$

due to which we can reconstruct $\hat{f}$ and thus f (almost) from a *single* scale $s > 0$ of the wavelet transform as long as $\hat{\psi}(\xi) \neq 0$ holds for any $\xi \neq 0$. Starting with (3.3.17) this leads to the simple inversion formula

$$\hat{f}_h = \left[\frac{\left(W_\psi f(s,\cdot)\right)^\wedge\left(\frac{2\pi k}{N}\right)}{\sqrt{s}\,h\,\hat{\varphi}\left(\frac{2\pi k}{hN}\right)\hat{\psi}\left(\frac{2\pi ks}{hN}\right)} \;:\; k \in \mathbb{Z}_N\right],$$

which, however, is in no way practical. It is possible, however, to show that local effect on the wavelet side are also local on the signal side, cf. (Sauer, 2011). Nevertheless, there is one important observation:

> The wavelet transform is highly redundant since almost all information on f is already obtained in a single scale of $W_\psi f$.

In principle, the inverse wavelet transform (3.3.9) can be applied to *any* function in the two variables s and u angewendet and yields, as a result, a function in a single variable. If we then apply the wavelet transform to this univariate functions, we obtain another bivariate function that needs *not* be one we started with. In other words,

$$W_\psi^{-1} W_\psi = I, \qquad W_\psi W_\psi^{-1} \neq I. \tag{3.3.21}$$

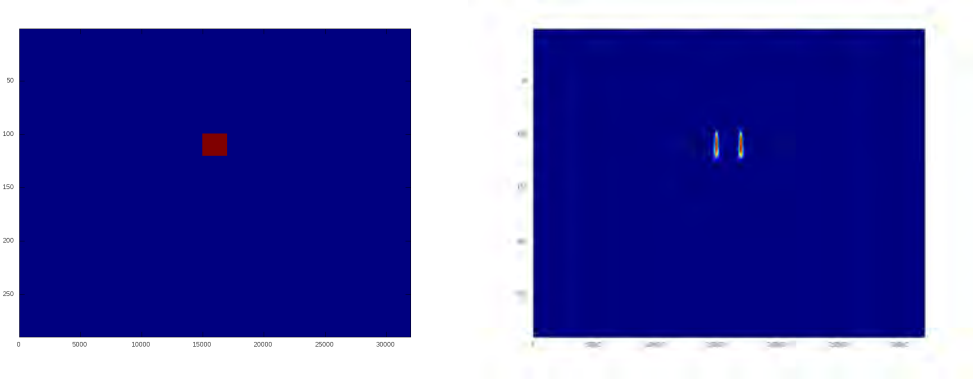


Figure 3.3.5: A two dimensional signal *(left)* and its related wavelet transform *(right)*. Obviously, the signals are different.

Let us have a closer look at this effect by remarking that the wavelet transform $W_\psi f$ as a bivariate function in s, u must satisfy the **compatibility conditions**

$$\frac{\left(W_\psi f(s,\cdot)\right)^\wedge(\xi)}{\left(W_\psi f(s',\cdot)\right)^\wedge(\xi)} = \sqrt{\frac{s}{s'}}\frac{\hat{\psi}(s\xi)}{\hat{\psi}(s'\xi)}, \qquad s,s' \in \mathbb{R}_+, \quad \xi \in \mathbb{R}. \tag{3.3.22}$$

Definition 3.3.12.

1. A bivariate function $g : \mathbb{R}_+ \times \mathbb{R} \to \mathbb{C}$ is called ψ-**compatible**, if it satisfies (3.3.22).
2. Two functions $g, g' : \mathbb{R}\times\mathbb{R} \to \mathbb{C}$ are called **related** if $W_\psi^{-1} g = W_\psi^{-1} g'$.

The wavelet transform of a function f is trivially ψ-compatible as that is how we built this property. On the other hand hte wavelet transforms are the only π-compatible functions and related to any function.

Lemma 3.3.13. *For any $g : \mathbb{R}\times\mathbb{R} \to \mathbb{C}$ there exists a ψ-compatible function g', related to g, which is a wavelet transform.*

Proof: Setting $g' = W_\psi W_\psi^{-1} g$, we get

$$W_\psi^{-1}(g - g') = W_\psi^{-1} g - \underbrace{W_\psi^{-1} g W_\psi}_{=I} W_\psi^{-1} g = W_\psi^{-1} g - W_\psi^{-1} g = 0.$$

Therefore, g and g' are related and g' ist obviously a wavelet transform. □

This explains the ambiguity of the inverse wavelet transform: if we form equivalence classes modulo W_ψ^{-1} in $L_2(\mathbb{R}\times\mathbb{R})$ by identifying related functions, then by Lemma 3.3.13 each such equivalence class contains a unique wavelet transform: $W_\psi f$ and $W_\psi f'$ being in the same equivalence class means $0 = W_\psi^{-1}\left(W_\psi f - W_\psi f'\right) = f - f'$. The operator $W_\psi W_\psi^{-1}$ is then the projector on this equivalence class and the representing wavelet transform is the only element in this equivalence class that is reproduced by $W_\psi W_\psi^{-1}$.

3.3.3 Examples: Music and Edges

Music analysis and **audio analysis** in general deals with tones ore sequences of tone which is defined locally as follows.

Definition 3.3.14. A **tone**[18] is a periodic function, i.e., a function such that $f\left(\cdot + \omega^{-1}\right) = f$ for some $\omega > 0$, called the **frequency** of the tone which is usually given in **Hertz**, where $1\text{Hz} = 1\text{s}^{-1}$.

From analysis, we know a representation method for periodic functions, namely the Fourier series. If we set the period length from $1/\omega$ to 2π, we get the following result.

Proposition 3.3.15. *Each 2π periodic function f can be uniquely described by its **Fourier series***

$$\frac{a_0}{2} + \sum_{k=1}^{\infty} a_k \cos(k\cdot) + \sum_{k=1}^{\infty} b_k \sin(k\cdot) \tag{3.3.23}$$

where

$$a_k = \frac{1}{\pi}\int_0^{2\pi} f(t)\cos kt\,dt, \qquad b_k = \frac{1}{\pi}\int_0^{2\pi} f(t)\sin kt\,dt. \tag{3.3.24}$$

Remark 3.3.16. Proposition 3.3.15 has to be taken with a bit of care.

1. All **Fourier coefficients** of f are well defined by (3.3.24) for reasonable functions f, and they are unique in the sense that two different functions have different Fourier series, but there is *no* general statement concerning convergence of these series and even if it converges, it is not clear that the limit of the series coincides with f.

2. The answer on convergence is actually negative: as shownby DuBois–Reymond in 1873, there are even *continuous* functions whose Fourier series diverges at least at one point, cf. (Sauer, 2017).

3. On the other hand, things are not so bad since the Fourier series of nice functions converge *almost everywhere*, but this requires a bit more of mathematical effort, see (Hardy and Rogosinsky, 1956).

Our Definition 3.3.14 for a tone is not very realistic as it means that a tone would have to sound constantly over infinite time, even without any change in loudness. Let us make some more realistig assumptions:

1. The duration of a tone is *significantly larger* than the inverse frequency $1/\omega$ needed for a single period of the tone which means that at least we face a periodic signal over a certain period of time.

2. The **amplitude** or **loudness** of the tone remains constant during this period.

[18] Be aware that precussion instruments usually do not generate a tone, but *noise*.

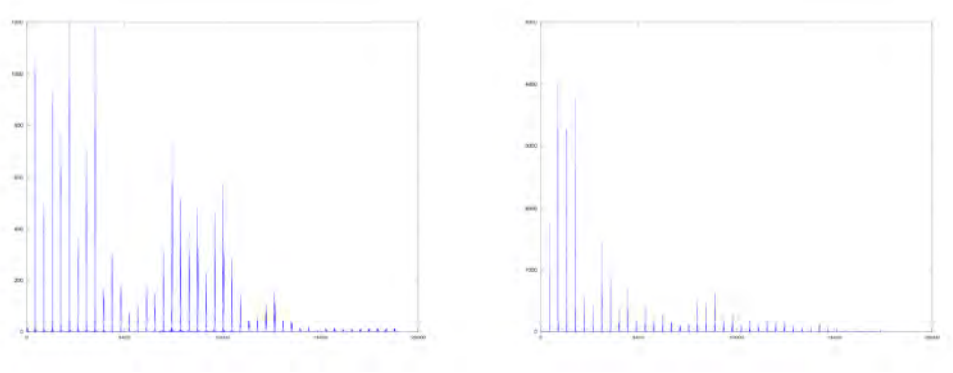


Figure 3.3.6: Spectra of two woodwind instruments for the same tone (a, 440Hz). The higher share of high partial tones on the left shows that this instument has a "sharper" sound

If we assume that these two conditions are valid, which still excludes string instruments where the amplitude decays exponentially, but at least includes woodwind instruments, then we can give the acoustic or musical interpretation of the Fourier series:

> *Each tone can be decomposed into **partial tones** whose frequencies are integer multiples of the **base frequency**. The Fourier coefficients for these tones describe the **spectrum** of the tone and therefore its **timbre**.*

The effect of the spectral fingerprint can be seen in Fig. 3.3.7. In fact, the spectrum is used for a lot of applications, for example also to recognize speech and voices.

All this only works for tones of infinite duration, in the moment where for example an instrument chances the frquency, even the length of the periodization changes. There is a concept of *instantaneous frequency*, cf. (Mallat, 1999), but a good and pragmatic approach is to use time-frequency analysis, for example by means of the Gabor transform or a wavelet transform. With the latter, one should use a more "musical" wavelet that is similar to a modulated tone. Here, in fact the Morlet wavelet is a good choice.

The difference between pure frequency analysis and time-frequency analysis can be illustrated nicely by yet anothter acoustic phenomenon, known as **beats**, which are the acoustic realization of the **addition theorem**

$$\cos\omega + \cos\omega' = 2\,\cos\frac{\omega+\omega'}{2}\,\cos\frac{\omega-\omega'}{2}, \tag{3.3.25}$$

of the cosine. Interpreting both side of (3.3.25) acoustically or musically, the left hand side stands for two simultaneous tones with the frequencies ω and ω', while the right hand side is a tone of *average frequency* Frequenz $(\omega+\omega')/2$ to which an **amplitude modulation** with the *difference frequency* $(\omega-\omega')/2$ is applied. Depending on the transform the one uses, one may obtain either side of equation (3.3.25), see Fig. **??**. With a Gabor transform the influence of the window size can be seen even as a transition between the two interpretations. If the window is very large, the Gabor transform is like a Fourier transform and the left hand side of (3.3.25) is

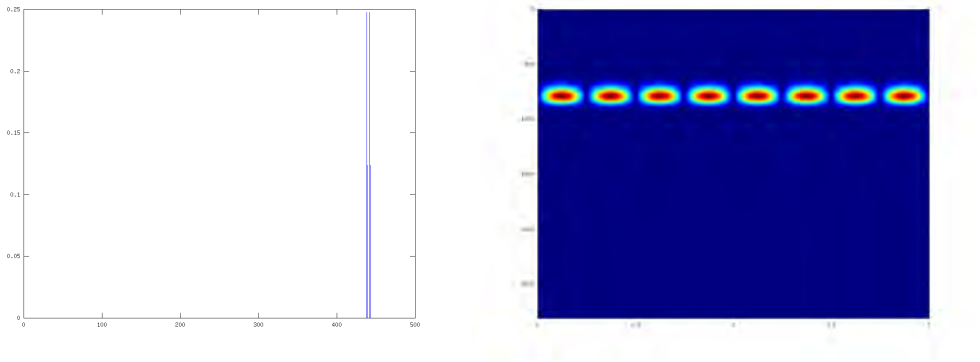


Figure 3.3.7: Spectrum and wavelet transform of the expression from (3.3.25). The spectrum *(left)* reproduces the left hand side, i.e., two localized frequencies while the wavelet transform shows an amplitude modulated and blurred tone *(right)*.

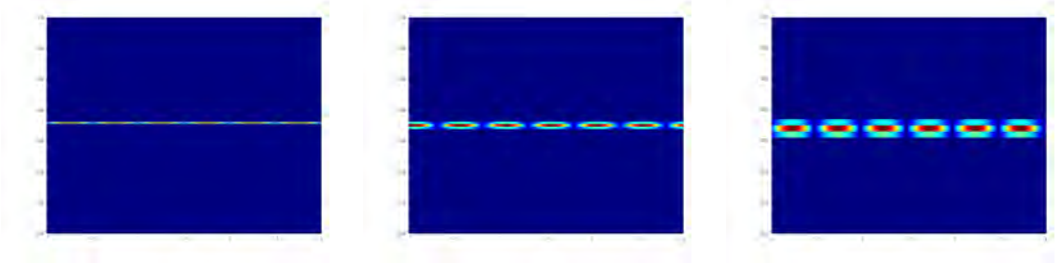


Figure 3.3.8: Three Gabor transform with varying window size from large *(left)* to small *(right)*.

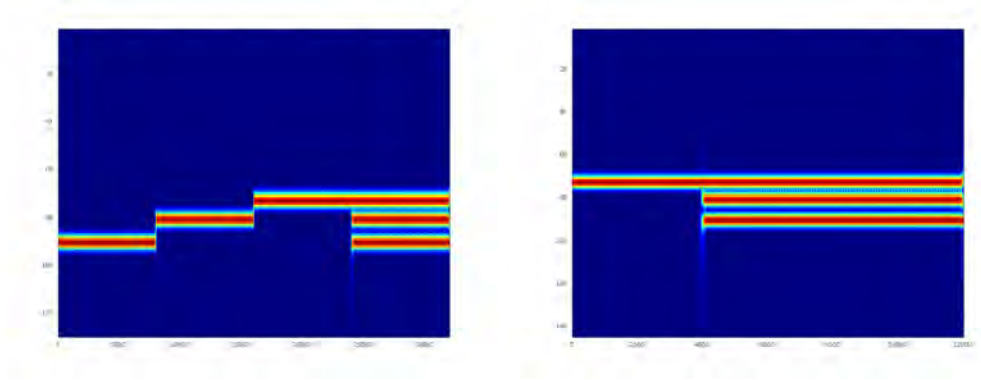


Figure 3.3.9: A simple melody, name a chord, played first successively and then simultaneously *(left)* plus a zoom into the simultaneous chord *(right)*.

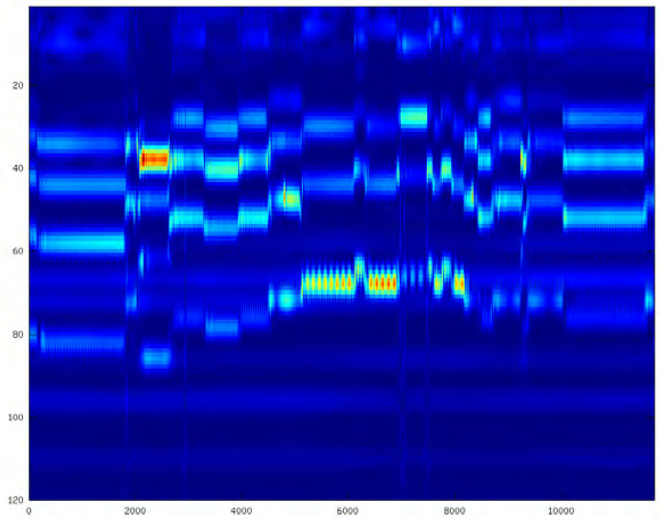


Figure 3.3.10: Wavelet transform of a bit of real music played on a drone instrument.

active, if the window gets smaller, the frequency blurs and the beat behaves like the right hand side. Also note that the phases of the beat changes in the two rightmost pictures while its frequency remains constant.

Another bit of of time-frequency can be seen when looking a simple chord as in Fig. 3.3.9. The three individual tones are clearly separated and the width of the strips is due to the Heisenberg uncertainty principle, but the simultaneous chord blurs the highest and the lowest tone. This is yet another unavoidable aspect of the wavelet transform:

> The higher the frequency, the higher the time resolution, but for the prices of a lower frequency resolution.

As a summary, let us recall that wavelets always provide a *relative* time and frequency precision while the Gabor transform works is *absolute* precision.

Finally, a little snippet of a piece of real music on a real drone instrument.

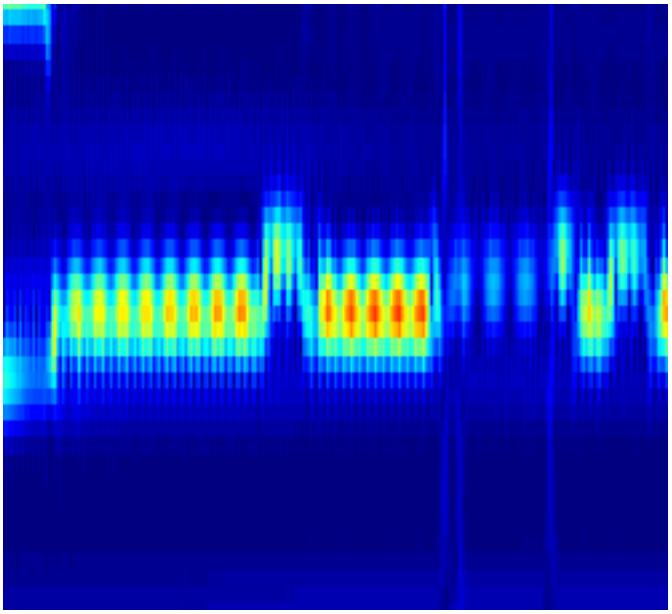


Figure 3.3.11: Snippet of Fig 3.3.10, showing how one partial tone of the major sixth meets with a partial of the drone and thus gets amplified.

The **drone** produces a constant tone which, together with its partial tones, can be recognized from the horizontal blue stripes in the wavelt transform. Also the melody and its partials can be recognized as well as the fact that some partials of the melody instrument meet with partials of the drone which gives them a peculiar sound or timbre, see Fig. 3.3.12. This happens only if the instrument has a *pure tuning*, which would get us into music theory and distract us too far from the main content of this lecture. But there is nice literature cf. (Barbour, 1951; Benson, 2007) on very different levels.

After this little excursion to the world of music, we want to look at another application, where wavelets offer a real advantage, namely the detection of local regularity. To that end, recall that, due to the $(i\xi)^k$ in the Fourier transform of a kth derivative, the Fourier transform

$$\hat{g}(\xi) = (i\xi)^{-k}\,\hat{f}(\xi), \qquad f = g^{(k)},$$

of a differentiable function g decays the faster for $\xi \to \pm\infty$, the larger the smoothness k of the function g is. In other words, the (global) smoothness of the function relates to the decay rate of its Fourier transform. But since the Fourier transform is global, this would just depend on the least smooth position of the function.

Wavelets allow provide a much more detailed analysis as the following result due to Jaffard shows, cf. (Mallat, 1999). The formulation is copied from another lecture, (Sauer, 2008).

Theorem 3.3.17. *Let ψ a wavelet with n vanishing moment and rapidly decaying derivatives of order $\le n$.*

1. *If $f \in L_2(\mathbb{R})$ is Lipschitz continuous[19] of order $\alpha < n$ at $x \in \mathbb{R}$, then there exists a*

[19] A function f is called **Lipschitz continuous** of order α at x if $|f(x') - f(x)| \le C|x' - x|^\alpha$ for some constant C and all x' from a neighborhood of x. It is some type of *controled continuity*.

constant C, such that

$$\left|W_\psi f(u,s)\right| \le C\, s^{\alpha+\frac{1}{2}} \left(1 + \left|\frac{u-x}{s}\right|^{\alpha}\right), \qquad (u,s) \in \Gamma = \mathbb{R}\times\mathbb{R}_+. \tag{3.3.26}$$

2. *If, conversely, $\alpha \notin \mathbb{N}$ and there exist C as well as $\alpha' < \alpha$ such that*

$$\left|W_\psi f(u,s)\right| \le C\, s^{\alpha+\frac{1}{2}} \left(1 + \left|\frac{u-x}{s}\right|^{\alpha'}\right), \qquad (u,s) \in \Gamma, \tag{3.3.27}$$

then f is Lipschitz continuous of order α at x.

The intuition of Theorem 3.3.17 is that *local regularity* of a function can be characterized by the decay of the local wavelet coefficients, and this can be used to find singularities as these are positions of *low* local regularity, as depicted in Fig. **??**.

The restriction to the wavelet is that it has n **vanishing moments** which means that

$$\int_{\mathbb{R}} t^k\, \psi(t)\, dt = 0, \qquad k = 0, \dots, n-1.$$

This restriction is very mild, one vanishing moment is even the definition of a wavelet. There exist wavelets with any given number of vanishing moment since a simple partial integration shows that whenever ψ is a differentiable wavelet with fast decay[20] and n vanishing moments, then its derivative ψ' is a wavelet with $n+1$ vanishing moments:

$$\int_{\mathbb{R}} t^{n+1}\, \psi'(t)\, dt = \lim_{t\to\infty} \underbrace{(\cdot)^{n+1}\psi\Big|_{-t}^{t}}_{\to 0} - \underbrace{\int_{\mathbb{R}} t^n\, \psi(t)\, dt}_{=0} = 0.$$

In fact, this is the construction principle for the mexican hat wavelet (3.3.7): applying the second derivative to optimal time frequency resolution function gives it two vanishing moments and makes it a good wavelet.

Exercise 3.3.4 How many vanishing moments does the Haar wavelet have? ◊

3.4 Filterbanks

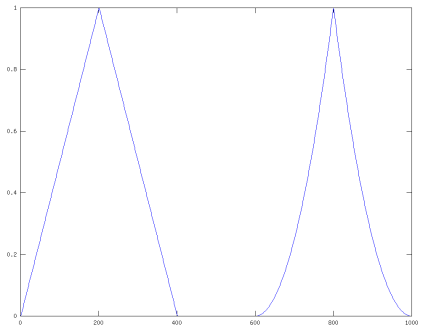
We now introduce another, fully discrete concept of wavelets that will be based completely on the filters that we encountered in Section 2.4. To handle them mathematically, the following concept turns out to be useful.

Definition 3.4.1 (z-transform). The z-**transform** of a discrete signal $c \in \ell(\mathbb{Z})$ is the formal *formal* **Laurent series**[21]

$$c^*(z) := \sum_{k\in\mathbb{Z}} c(k)\, z^{-k}, \qquad z \in \mathbb{C}_\times = \mathbb{C}\setminus\{0\}. \tag{3.4.1}$$

[20] ψ should decay faster than any polynomial can grow, i.e. $t^k\psi(t) \to 0$ for $t \to \pm\infty$ and any k.

[21] If you feel uncertain with the infinite series assume the signal to be finitely supported, that is sufficient here.



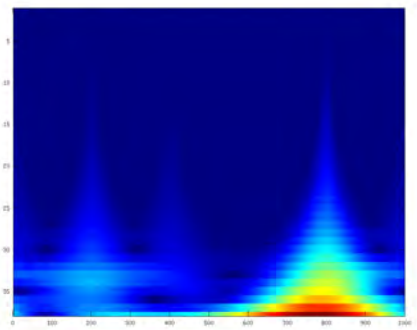


Figure 3.3.12: A simple test function *(left)* and its wavelet transform *(right)*. The decay of the wavelets ceofficients and the cusps in the functions, i.e., the discontinuities in the derivative, match nicely.

The z-transform is closely related to our Fourier transform of a discrete signal signal since for $\theta \in \mathbb{T}$,

$$c^*\left(e^{i\theta}\right) = \sum_{k\in\mathbb{Z}} c(k)\, e^{-ik\theta} = \hat{c}(\theta), \tag{3.4.2}$$

and we can simply jump between the two. In particular,

$$(c * d)^*(z) = c^*(z)\, d^*(z). \tag{3.4.3}$$

Exercise 3.4.1 Prove (3.4.3). ◊

The idea of **subband coding** which is used especially in data compression, consists of decomposition a signal into so-called **subbands** and to encode each of these subsignals independently, but of course in such a way that the signal can be reconstructed from them.

Example 3.4.2. The most intuitive idea for a subband decomposition would be to mimic an *equalizer* and to use

$$\chi^{\vee}_{[t_j,t_{j+1}]}, \qquad 0 = t_0 < t_1 < \cdots < t_{n-1} < t_n = 2\pi.$$

This sounds very much in the spirit of High-Fi, but the realization would be tricky as the filters would be relatively long and therefore affected by severy latency.

Definition 3.4.3 (Up- and downsampling). For $n \geq 2$, we call the operator $\downarrow_n$ that associates to $c \in \ell(\mathbb{Z})$ the signal

$$\downarrow_n c = c(n\cdot)$$

downsampling operator, while the **upsampling** operator $\uparrow_n$ is defined as

$$\uparrow_n c(j) = \begin{cases} c(j/n), & j \in n\mathbb{Z}, \\ 0, & j \in \mathbb{Z} \setminus n\mathbb{Z}. \end{cases}$$

Obviously, $\downarrow_n\uparrow_n = I \neq \uparrow_n\downarrow_n$. With the help of downsampling we can indeed decompose a signal c into the subbands

$$c_j = \downarrow_n \tau_j c, \qquad j \in \mathbb{Z}_n,$$

and recombine them via

$$c = \sum_{j\in\mathbb{Z}_n} \tau_{-j} \uparrow_n c_j,$$

which works due to the cure formula

$$I = \sum_{j\in\mathbb{Z}_n} \tau_{-j} \uparrow_n\downarrow_n \tau_j. \tag{3.4.4}$$

The decomposition process gives

$$\begin{array}{cccccccccc}
\dots & c(-1) & \boxed{c(0)} & c(1) & \dots & c(n-1) & \boxed{c(n)} & c(n+1) & \dots & \to c_0 \\
\dots & c(0) & c(1) & c(2) & \dots & c(n) & c(n+1) & c(n+2) & \dots & \to c_1 \\
 & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & & \\
\dots & c(n-2) & \boxed{c(n-1)} & c(n) & \dots & c(2n-2) & \boxed{c(2n-1)} & c(2n) & \dots & \to c_{n-1}
\end{array}$$

so that

$$c_j = c\,(n\cdot + j)\,, \qquad j \in \mathbb{Z}_n,$$

which can be combined via upsampling and translation into

$$\begin{array}{cccccccccc}
c_0 \to & 0 & c(0) & 0 & \dots & 0 & c(n) & 0 & \dots & \\
c_1 \to & 0 & 0 & c(1) & \dots & 0 & 0 & c(n+1) & \dots & \\
 & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & \\
c_{n-1} \to & c(-1) & 0 & 0 & \dots & c(n-1) & 0 & 0 & \dots & \\
\hline
 & c(-1) & c(0) & c(1) & \dots & c(n-1) & c(n) & c(n+1) & \dots & \to c
\end{array}$$

which proves (3.4.4). In **subband coding** we replace the shifts in the process above by general filters F_j, $j \in \mathbb{Z}_n$, and the downsample the filtered signal, i.e.,

$$c_j = \downarrow_n F_j\, c, \qquad j \in \mathbb{Z}_n. \tag{3.4.5}$$

Schematically, this process can be depicted as

$$c \to \odot \begin{array}{ccccc} \nearrow & \boxed{F_0} & \to & \boxed{\downarrow_n} & \to\ c_0 \\ & \vdots & & \vdots & \vdots \\ \searrow & \boxed{F_{n-1}} & \to & \boxed{\downarrow_n} & \to\ c_{n-1} \end{array} \tag{3.4.6}$$

and the decomposition is called the **analysis filterbank** based on the filters $F = \big(F_j : j \in \mathbb{Z}_n\big)$. Formally, the filterbank maps $\ell(\mathbb{Z})$ to $\ell^n(\mathbb{Z})$. Because of the downsampling, each subband c_j of c contains only the nth part of the information of $F_j c$, hence also one nth of the information in c, provided the filters are reasonable. This is the motivation to choose the number of filters equal to the downsampling factor. Such a filterbank is called **critically sampled**, otherwise one speaks of **oversampled** or **undersampled** filterbanks.

Example 3.4.4. The simplest filterbank is the one with $F_j = \tau_j$, $j \in \mathbb{Z}_n$, and just decomposes a signal modulo n.

Next, we want to give a mathematical description of the filterbank under the usual assumption that each F_j is an FIR filter with impulse response f_j, hence $F_j c = f_j * c$. This suggests the following notion.

Definition 3.4.5 (Modulation matrix). The **modulation matrix** $M(z)$ for the filterbank

$$F = \left(F_j \ : \ j \in \mathbb{Z}_n\right)$$

is defined as

$$\begin{aligned} M(z) &:= \frac{1}{n}\left(f_j^*\left(e^{2\pi ik/n}\, z\right) \ : \ j,k \in \mathbb{Z}_n\right) \\ &= \frac{1}{n}\begin{pmatrix} f_0^*(z) & f_0^*\left(e^{2\pi i/n} z\right) & \dots & f_0^*\left(e^{2\pi i(n-1)/n} z\right) \\ \vdots & \vdots & \ddots & \vdots \\ f_{n-1}^*(z) & f_{n-1}^*\left(e^{2\pi i/n} z\right) & \dots & f_{n-1}^*\left(e^{2\pi i(n-1)/n} z\right) \end{pmatrix}. \end{aligned} \tag{3.4.7}$$

It once more takes a particularly simple form for $n = 2$ where

$$M(z) = \begin{pmatrix} f_0^*(z) & f_0^*(-z) \\ f_1^*(z) & f_1^*(-z) \end{pmatrix}.$$

In general, the factors $e^{2\pi ij/n}$ in front of the variable z are nth roots of unity and thus play the role of generalized signs. The modulation matrix, on the other hand, is the mathematical tool to describe the action of a filterbank.

Theorem 3.4.6. *For the filterbank from* (3.4.6) *we have*

$$\left(c_j^*\left(z^n\right) \ : \ j \in \mathbb{Z}_n\right) = M(z)\ \left(c^*\left(e^{2\pi ij/n} z\right) \ : \ j \in \mathbb{Z}_n\right), \tag{3.4.8}$$

that is,

$$\begin{pmatrix} c_0^*\left(z^n\right) \\ \vdots \\ c_{n-1}^*\left(z^n\right) \end{pmatrix} = M(z) \begin{pmatrix} c^*(z) \\ c^*\left(e^{2\pi i/n} z\right) \\ \vdots \\ c^*\left(e^{2\pi i(n-1)/n} z\right) \end{pmatrix}. \tag{3.4.9}$$

Exercise 3.4.2 Determine the modulation matrix for the translation filters $F_j = \tau_j$, $j \in \mathbb{Z}_n$. ◇

Definition 3.4.7 (Polyphase vector). The vector

$$\left(c^*\left(e^{2\pi ij/n} z\right) \ : \ j \in \mathbb{Z}_n\right) \tag{3.4.10}$$

is called the **polyphase vector** of the signal $c \in \ell(\mathbb{Z})$.

For $n = 2$, (3.4.9) becomes

$$\begin{pmatrix} c_0^*\left(z^2\right) \\ c_1^*\left(z^2\right) \end{pmatrix} = \begin{pmatrix} f_0^*(z) & f_0^*(-z) \\ f_1^*(z) & f_1^*(-z) \end{pmatrix} \begin{pmatrix} c^*(z) \\ c^*(-z) \end{pmatrix}.$$

Let us begin to tackle the proof of Theorem 3.4.6 by first determining what up- and downsampling does to the z-transform.

Lemma 3.4.8. *For $n \in \mathbb{N}$,*

$$(\downarrow_n c)^*(z^n) = \frac{1}{n} \sum_{k \in \mathbb{Z}_n} c^*\left(e^{2\pi i k/n} z\right), \qquad (\uparrow_n c)^*(z) = c^*(z^n). \tag{3.4.11}$$

Proof: Since

$$\frac{1}{n} \sum_{k \in \mathbb{Z}_n} e^{-2\pi i j k/n} = \begin{cases} 1, & j \in n\mathbb{Z}, \\ 0, & j \notin n\mathbb{Z}, \end{cases}$$

see Lemma 2.5.5, the left hand identity in (3.4.11) follows via

$$\begin{aligned} (\downarrow_n c)^*(z^n) &= \sum_{j \in \mathbb{Z}} (\downarrow_n c)(j)\, z^{-nj} = \sum_{j \in \mathbb{Z}} c(nj)\, z^{-nj} = \sum_{j \in \mathbb{Z}} c(j)\, z^{-j} \left(\frac{1}{n} \sum_{k \in \mathbb{Z}_n} e^{-2\pi i j k/n} \right) \\ &= \frac{1}{n} \sum_{j \in \mathbb{Z}} c(j) \sum_{k \in \mathbb{Z}_n} \left(e^{2\pi i k/n} z\right)^{-j} = \frac{1}{n} \sum_{k \in \mathbb{Z}_n} c^*\left(e^{2\pi i k/n} z\right), \end{aligned}$$

The right hand side (3.4.11) is a consequence of the simple computation

$$(\uparrow_n c)^*(z) = \sum_{j \in \mathbb{Z}} c(j)\, z^{-nj} = c^*(z^n).$$

□

Also (3.4.11) is simpler for $n = 2$: since $e^{i\pi} = -1$,

$$(\downarrow_2 c)^*\left(z^2\right) = \frac{1}{2}\left(c^*(z) + c^*(-z)\right), \qquad (\uparrow_2 c)^*(z) = c^*\left(z^2\right). \tag{3.4.12}$$

There is another interpretation of (3.4.11) for which we write $c^*(z)$ as

$$\begin{aligned} c^*(z) &= \sum_{j \in \mathbb{Z}} c(j) z^{-j} = \sum_{k \in \mathbb{Z}_n} \sum_{j \in \mathbb{Z}} c(nj+k)\, z^{-nj-k} = \sum_{k \in \mathbb{Z}_n} z^{-k} \sum_{j \in \mathbb{Z}} (\uparrow_n \tau_k c)(j)\, z^{-nj} \\ &= \sum_{k \in \mathbb{Z}_n} z^{-k}\, (\uparrow_n \tau_k c)^*(z^n) =: \sum_{k \in \mathbb{Z}_n} z^{-k}\, \widetilde{c}_k^*(z^n), \end{aligned}$$

where $\widetilde{c}_k$ is the signal obtained from the simple translation filterbank, sometimes also called the **lazy filterbank**. If we now substitute $z = e^{2\pi i j/n} z$, then

$$c^*\left(e^{2\pi i j/n} z\right) = \sum_{k \in \mathbb{Z}_n} e^{-2\pi i j k/n} z^{-k}\, \widetilde{c}_k^*(z^n)$$

and consequently

$$\begin{aligned}\frac{1}{n}\sum_{j\in\mathbb{Z}_n} c^*\left(e^{2\pi ij/n}z\right) &= \frac{1}{n}\sum_{j\in\mathbb{Z}_n}\sum_{k\in\mathbb{Z}_n} e^{-2\pi ijk/n} z^{-k}\,\widetilde{c}_k^*\,(z^n)\\ &= \sum_{k\in\mathbb{Z}_n}\underbrace{\left(\frac{1}{n}\sum_{j\in\mathbb{Z}_n} e^{-2\pi ijk/n}\right)}_{=\delta_{0k}} z^{-k}\,\widetilde{c}_k^*\,(z^n) = \widetilde{c}_0^*\,(z^n)\,.\end{aligned}$$

Multiplying by $e^{2\pi ik/n}$ then finally gives

$$z^{-k}\,\widetilde{c}_k^*\,(z^n) = e^{2\pi ik/n}\,\frac{1}{n}\sum_{j\in\mathbb{Z}_n} c^*\left(e^{2\pi ij/n}z\right), \qquad k\in\mathbb{Z}_n. \tag{3.4.13}$$

Exercise 3.4.3 What is (3.4.13) for $n = 2$? ◊

Proof of Theorem 3.4.6: Since $c_j = F_j c = f_j * c$, we get for $j \in \mathbb{Z}_n$,

$$\begin{aligned}c_j^*\,(z^n) &= \left(\downarrow_n \left(f_j * c\right)\right)^*(z^n) = \frac{1}{n}\sum_{k\in\mathbb{Z}_n}\left(f_j * c\right)^*\left(e^{2\pi ik/n}z\right)\\ &= \frac{1}{n}\sum_{k\in\mathbb{Z}_n} f_j^*\left(e^{2\pi ik/n}z\right)\; c^*\left(e^{2\pi ik/n}z\right).\end{aligned}$$

from which (3.4.9) follows by passing to matrix-vector notation. □

Since it does not really make if difference whether we consider the z-transform on $\mathbb{C}$ or the Fourier transform on the unit circle, we can also formulate the modulation matrix for theFourier transform of the signal. Substituting $z = e^{i\xi/n}$ into (3.4.9), it then follows that

$$\begin{pmatrix}\hat{c}_0(\xi)\\ \vdots\\ \hat{c}_{n-1}(\xi)\end{pmatrix} = \begin{pmatrix}c_0^*\,(z^n)\\ \vdots\\ c_{n-1}^*\,(z^n)\end{pmatrix} = M(z)\,\left(c^*\left(e^{2\pi ij/n}\,z\right)\;:\; j\in\mathbb{Z}_n\right)$$

$$= M\left(e^{i\xi/n}\right)\left(c^*\left(e^{2\pi ij/n}\,e^{i\xi/n}\right)\;:\; j\in\mathbb{Z}_n\right) = M\left(e^{i\xi/n}\right)\left(c^*\left(e^{i(\xi+2j\pi)/n}\right)\;:\; j\in\mathbb{Z}_n\right)$$

$$= \begin{pmatrix} f_0^*\left(e^{i\xi/n}\right) & f_0^*\left(e^{i(\xi+2\pi)/n}\right) & \dots & f_0^*\left(e^{i(\xi+2(n-1)\pi)/n}\right)\\ \vdots & \vdots & \ddots & \vdots\\ f_{n-1}^*\left(e^{i\xi/n}\right) & f_{n-1}^*\left(e^{i(\xi+2\pi)/n}\right) & \dots & f_{n-1}^*\left(e^{i(\xi+2(n-1)\pi)/n}\right)\end{pmatrix}\begin{pmatrix} c^*\left(e^{i\xi/n}\right)\\ c^*\left(e^{i(\xi+2\pi)/n}\right)\\ \vdots\\ c^*\left(e^{i(\xi+2(n-1)\pi)/n}\right)\end{pmatrix}$$

$$= \begin{pmatrix} \hat{f}_0\left(\frac{\xi}{n}\right) & \hat{f}_0\left(\frac{\xi}{n}+2\pi\frac{1}{n}\right) & \dots & \hat{f}_0\left(\frac{\xi}{n}+2\pi\frac{n-1}{n}\right)\\ \vdots & \vdots & \ddots & \vdots\\ \hat{f}_{n-1}\left(\frac{\xi}{n}\right) & \hat{f}_{n-1}\left(\frac{\xi}{n}+2\pi\frac{1}{n}\right) & \dots & \hat{f}_{n-1}\left(\frac{\xi}{n}+2\pi\frac{n-1}{n}\right)\end{pmatrix}\begin{pmatrix}\hat{c}\left(\frac{\xi}{n}\right)\\ \hat{c}\left(\frac{\xi}{n}+2\pi\frac{1}{n}\right)\\ \vdots\\ \hat{c}\left(\frac{\xi}{n}+2\pi\frac{n-1}{n}\right)\end{pmatrix},$$

hence

$$\left(\hat{c}_j(\xi) \ : \ j \in \mathbb{Z}_n\right) = \left(\hat{f}_j\left(\tfrac{\xi+2k\pi}{n}\right) \ : \ j,k \in \mathbb{Z}_n\right)\left(\hat{c}\left(\tfrac{\xi+2j\pi}{n}\right) \ : \ j \in \mathbb{Z}_n\right), \tag{3.4.14}$$

where the matrix

$$\left(\hat{f}_j\left(\tfrac{\xi+2\pi k}{n}\right) \ : \ j,k \in \mathbb{Z}_n\right)$$

is also called **polyphase matrix** for the filterbank.

Finally, we also want to recombine the subbands c_j, $j \in \mathbb{Z}_n$, into a signal c by going the opposite way of first upsampling, the filtering and finally summing up the results:

$$c' = \sum_{j\in\mathbb{Z}_n} G_j \uparrow_n c_j. \tag{3.4.15}$$

This leads to the **synthesis filterbank**

$$\begin{array}{ccccccc} c_0 & \to & \boxed{\uparrow_n} & \to & \boxed{G_0} & \searrow & \\ \vdots & & \vdots & & \vdots & & \oplus \to c \\ c_{n-1} & \to & \boxed{\uparrow_n} & \to & \boxed{G_{n-1}} & \nearrow & \end{array} \tag{3.4.16}$$

that can be expressed in the calculus of z-transforms as

$$c^*(z) = \sum_{j\in\mathbb{Z}_n} \left(G_j \uparrow_n c_j\right)^*(z) = \sum_{j\in\mathbb{Z}_n} g_j^*(z)\left(\uparrow_n c_j\right)^*(z) = \sum_{j\in\mathbb{Z}_n} g_j^*(z)\, c_j^*(z^n). \tag{3.4.17}$$

To return to the modulation matrix, we recall that its input data was the vector $\left(c^*\left(e^{2\pi ij/n}z\right) \ : \ j \in \mathbb{Z}_n\right)$ which is obtained by replacing z in (3.4.17) by $e^{2\pi ij/n}z$, $j \in \mathbb{Z}_n$. Taking into account that $\left(e^{2\pi ij/n}z\right)^n = z^n$, we can use the matrix-vector form

$$\begin{pmatrix} c^*(z) \\ \vdots \\ c^*\left(e^{2\pi i(n-1)/n}z\right) \end{pmatrix} = \underbrace{\begin{pmatrix} g_0^*(z) & \cdots & g_{n-1}^*(z) \\ \vdots & \ddots & \vdots \\ g_0^*\left(e^{2\pi i(n-1)/n}z\right) & \cdots & g_{n-1}^*\left(e^{2\pi i(n-1)/n}z\right) \end{pmatrix}}_{=:\widetilde{M}(z)} \begin{pmatrix} c_0^*(z^n) \\ \vdots \\ c_{n-1}^*(z^n) \end{pmatrix}, \tag{3.4.18}$$

or

$$\left(c^*\left(e^{2\pi ij/n}z\right) \ : \ j \in \mathbb{Z}_n\right) = \widetilde{M}(z)\left(c_j^*(z^n) \ : \ j \in \mathbb{Z}_n\right), \tag{3.4.19}$$

respectively. In the end, we consider the full system concerning of the analysis and the synthesis part, and this is the **filterbank**

$$\begin{array}{ccccccccccc} & & \nearrow & \boxed{F_0} & \to & \boxed{\downarrow_n} & \to & c_0 & \to & \boxed{\uparrow_n} & \to & \boxed{G_0} & \searrow & & \\ c & \to & \odot & \vdots & & \vdots & & \vdots & & \vdots & & \vdots & & \oplus & \to c' \\ & & \searrow & \boxed{F_{n-1}} & \to & \boxed{\downarrow_n} & \to & c_{n-1} & \to & \boxed{\uparrow_n} & \to & \boxed{G_{n-1}} & \nearrow & & \end{array} \tag{3.4.20}$$

A natural requirement for such a filterbank is that what is put in should be obtained as a result.

Definition 3.4.9 (Perfect reconstruction). The filterbank (F, G) provides **perfect reconstruction** if $c' = c$ holds in (3.4.20) for all input signals c.

Remark 3.4.10.

1. By means of the modulation matrix, perfect reconstruction can be described very elegantly by substituting (3.4.8) into (3.4.19) which shows that perfect reconstruction is equivalent to

$$\left(c^*\left(e^{2\pi ij/n}z\right) \;:\; j \in \mathbb{Z}_n\right) = \widetilde{M}(z)\, M(z)\, \left(c^*\left(e^{2\pi ij/n}z\right) \;:\; j \in \mathbb{Z}_n\right). \qquad (3.4.21)$$

   Hence, a filterbank provides perfect reconstruction whenever

$$\widetilde{M}(z)\, M(z) = I.$$

2. A more generous approach allows a *time delay* in the perfect reconstruction and only requests that $c' = \tau_j c$ for some $j \in \mathbb{Z}$. Due to

$$(\tau_k c)^*(z) = z^{-k}\, c^*(z),$$

   we obtain in this case that

$$\left(e^{-2\pi ijk/n}z^{-k}c^*\left(e^{2\pi ij/n}z\right) \;:\; j \in \mathbb{Z}_n\right) = \widetilde{M}(z)\, M(z)\, \left(c^*\left(e^{2\pi ij/n}z\right) \;:\; j \in \mathbb{Z}_n\right),$$

   which holds whenever

$$\widetilde{M}(z)\, M(z) = \operatorname{diag}\left(e^{-2\pi ijk/n}z^{-k} \;:\; j \in \mathbb{Z}_n\right).$$

3. A straightforward computation yields

$$\widetilde{M}(z)\, M(z) = \left(\sum_{\ell\in\mathbb{Z}} g_\ell^*\left(e^{2\pi ij/n}\, z\right) f_\ell^*\left(e^{2\pi ik/n}\, z\right) \;:\; j,k \in \mathbb{Z}_n\right).$$

As we have seen in the above remark, perfect reconstruction follows from $\widetilde{M}\, M = I$, just subtitute into (3.4.21). The converse also holds true, but it is not so obvious.

Theorem 3.4.11 (Perfect reconstruction). *A filterbank (F, G) provides perfect reconstruction if and only if $\widetilde{M}(z)\, M(z) = I$, $z \in \mathbb{C}$.*

Proof: The direction "$\Leftarrow$" has been verified in Remark 3.4.10. For the converse, we consider the equivalent form

$$0 = \left(I - \widetilde{M}(z)\, M(z)\right)\, \left(c^*\left(e^{2\pi ij/n}z\right) \;:\; j \in \mathbb{Z}_n\right)$$

of (3.4.21) and set $c = \tau_k\delta$, $k \in \mathbb{Z}_n$. Since this implies that $c^*(z) = z^k$, hence

$$\left(c^*\left(e^{2\pi ij/n}z\right) \;:\; j \in \mathbb{Z}_n\right) = z^k\, \left(e^{2\pi ijk/n} \;:\; j \in \mathbb{Z}_n\right),$$

and since $z^k \neq 0$ on $\mathbb{T}$, we obtain that

$$0 = \left(I - \widetilde{M}(z)\, M(z)\right)\, \left(e^{2\pi ijk/n} \;:\; j \in \mathbb{Z}_n\right).$$

The vectors $\left(e^{2\pi ijk/n} \;:\; j \in \mathbb{Z}_n\right)$, $k \in \mathbb{Z}_n$, are *linearly independent* as columns of the nonsingular inverse DFT from Lemma 2.5.5. But this requests that $I - \widetilde{M}(z)\, M(z)$ equals zero for all $z \in \mathbb{T}$, hence also for all $z \in \mathbb{C}$. □

3.5 Subdivision, Functions & Wavelets

Having characterized the "good" filterbanks, i.e., those with perfect reconcstruction, we will now work with *cascades* of filterbanks which we will relate to decompositions of functions. Since all our *computational* operations will be of a purely discrete nature, we obtain a relationship between the discrete and the continuous world this way.

For the sake of simplicity, we consider the most common case $n = 2$, where our analysis filterbank is of the form

$$c \;\to\; \odot \;\begin{array}{l} \nearrow \;\boxed{F_0}\; \to \;\boxed{\downarrow_2}\; \to \; c_0 \\ \searrow \;\boxed{F_1}\; \to \;\boxed{\downarrow_2}\; \to \; c_1 \end{array}$$

The new idea now is to plug the components c_0 and/or c_1 into *the same* analysis filterbank and thus get a further decomposition of the original signal c as

$$c \;\to\; \odot \;\begin{array}{l} \nearrow \;\boxed{F_0}\; \to \;\boxed{\downarrow_2}\; \to \; c_0 \to \odot \;\begin{array}{l} \nearrow \;\boxed{F_0}\; \to \;\boxed{\downarrow_2}\; \to c_{00} \\ \searrow \;\boxed{F_1}\; \to \;\boxed{\downarrow_2}\; \to c_{01} \end{array} \\ \searrow \;\boxed{F_1}\; \to \;\boxed{\downarrow_2}\; \to \; c_1 \to \odot \;\begin{array}{l} \nearrow \;\boxed{F_0}\; \to \;\boxed{\downarrow_2}\; \to c_{10} \\ \searrow \;\boxed{F_1}\; \to \;\boxed{\downarrow_2}\; \to c_{11} \end{array} \end{array} \tag{3.5.1}$$

and so on. This way we obtain a tree structure of signals, known as **wavelet packages**. In **wavelet analysis**, however, one only further decomposes the signals components that result from the *low pass filter*, hence, we iterate as

$$c \;\to\; \odot \;\begin{array}{l} \nearrow \;\boxed{F_0}\; \to \;\boxed{\downarrow_2}\; \to \; c_0 \to \odot \;\begin{array}{l} \nearrow \;\boxed{F_0}\; \to \;\boxed{\downarrow_2}\; \to c_{00} \\ \searrow \;\boxed{F_1}\; \to \;\boxed{\downarrow_2}\; \to c_{01} \end{array} \\ \searrow \;\boxed{F_1}\; \to \;\boxed{\downarrow_2}\; \to \; c_1 \end{array} \tag{3.5.2}$$

and so on. In principle, this can be done with arbitrary perfect reconstruction filter banks, but the wavelet analysis only works reasonably if F_0 is a **low pass filter**, i.e., $F_0 1 = 1$, and F_1 is a **high pass filter**, i.e., $F_1 1 = 0$. Of course, we will have to specify and justify this later.

After r steps of the cascade we obtain the output signals

$$c_j^r \in \ell(\mathbb{Z}), \qquad j = 0, \ldots, 2^r - 1,$$

where we can see the cascade that had been applied directly from the binary representation of the index j. More precisely, if

$$j = \sum_{k=0}^{r-1} \epsilon_k \, 2^k =: \epsilon_{r-1} \cdots \epsilon_0, \qquad \epsilon_k \in \{0, 1\},$$

then

$$c_j^r = \downarrow_2 \; F_{\epsilon_0} \cdots \downarrow_2 \; F_{\epsilon_{r-1}} \, c.$$

Conversely, the synthesis from such a decomposition is realized by cascading the synthesis filterbank in the following way:

$$\begin{array}{llllll} c_{00} \to & \boxed{\uparrow_2} \to & \boxed{G_0} & \searrow & & \\ c_{01} \to & \boxed{\uparrow_2} \to & \boxed{G_1} & \nearrow & \oplus \to c_0 \to & \boxed{\uparrow_2} \to \boxed{G_0} \quad \searrow \\ & & & & & \qquad\qquad \oplus \to c \\ c_{10} \to & \boxed{\uparrow_2} \to & \boxed{G_0} & \searrow & & \\ c_{11} \to & \boxed{\uparrow_2} \to & \boxed{G_1} & \nearrow & \oplus \to c_1 \to & \boxed{\uparrow_2} \to \boxed{G_1} \quad \nearrow \end{array} \tag{3.5.3}$$

The synthesis cascade now allows for a shift of perspective by forgetting the underlying analysis cascaed for a moment and to see it as a method to generate functions of function approximations from simple discrete data, which is the idea of **subdivision**.

Imagine that we take a signal $x \in \ell(\mathbb{Z})$, feed it into c_0^r (with a *vector* index 0 here) and then consider the signal $c_r = c(x, r) \in \ell(\mathbb{Z})$ that we obtain this way. By definition,

$$c_r = (G_0 \uparrow_2)^r \; x,$$

hence, by Lemma 3.4.8,

$$\begin{aligned} c_r^*(z) &= \left[(G_0 \uparrow_2)^r \; x\right]^* (z) = g_0^*(z) \; \left[\uparrow_2 \; (G_0 \uparrow_2)^{r-1} \; x\right]^* (z) \\ &= g_0^*(z) \; \left[(G_0 \uparrow_2)^{r-1} \; x\right]^* \left(z^2\right) = g_0^*(z) \, c_{r-1}^* \left(z^2\right), \end{aligned}$$

and thus by iteration,

$$c_r^*(z) = g_0^*(z) \cdots g_0^* \left(z^{2^{r-1}}\right) \; x^* \left(z^{2^r}\right) = \left[\prod_{j=0}^{r-1} g_0^* \left(z^{2^j}\right)\right] \; x^* \left(z^{2^r}\right). \tag{3.5.4}$$

The upsampling process can be seen as inserting zeros between the components of the original signal, and from this perspective, our signal $c = (G_0 \uparrow_2)^r \, x$ will contain the 2^r-fold amount of data relative to x. This can be seen by choosing x finitely supported, say with N nonzero entries. Then the support size of c will be $O\left(2^r N\right)$, where some overlap due to the support size of g_0 will have to be considered for the constant. Moreover,

$$S_G x := G_0 \uparrow_2 x = g_0 * (\uparrow_2 x) = \sum_{k \in \mathbb{Z}} g_0 \left(\cdot - 2k\right) \; x(k)$$

and therefore

$$\tau_2 S_G = S_G \tau \qquad \text{bzw.} \qquad \tau_{2^r} S_G^r = S_G \tau,$$

because of which the signal $S_G^r x$ should be seen as a discrete function with abscissae $2^{-r} k$, $k \in \mathbb{Z}$: if, for example, x is a k-periodic signal, i.e, $x \in \ell(\mathbb{Z}_k)$m then $S_G^r x$ is $2^r k$ periodic, provided all convolutions in the filterbank have been done in $\mathbb{Z}_k$; otherwise it holds locally. And finally we note that since S_G is a linearer Operator and any signal x can be formally written as

$$x = \sum_{k \in \mathbb{Z}} x(k) \, \tau_k \, \delta, \qquad \delta(j) = \delta_{j0}, \quad j \in \mathbb{Z},$$

we have

$$c_r^*(z) = \sum_{k\in\mathbb{Z}} z^k\, x(k) \left[\prod_{j=0}^{r-1} g_0^*\left(z^{2^j}\right)\right] \underbrace{\delta^*\left(z^{2^r}\right)}_{=1},$$

and, in the spirit of impulse response, it is sufficient to restrict oneself to $x = \delta$.

This dicret function c, more prescisely, the discrete function $c\,(2^{-r}\cdot) \in \ell\,(2^{-r}\mathbb{Z})$, defined at **dyadic points** $2^{-r}\mathbb{Z}$ of order r, shall now be investigated in terms of the Fourier transform, i.e., we consider

$$\varphi_r(\xi) = [c_r\,(2^{-r}\cdot)]^\wedge\,(\xi), \qquad r \in \mathbb{N}_0.$$

To define this function in a formally correct way, we recall the relation (3.4.2) between the z-transform and the Fourier transform to obtain a sequence φ_r, $r \in \mathbb{N}_0$ of trigonometric polynomials which can be written as

$$\varphi_r(\xi) = 2^{-r}\,\hat{c}\,(\xi/2^r) = 2^{-r} c^*\left(e^{i\xi/2^r}\right) = \prod_{j=0}^{r-1} \frac{1}{2}\, g_0^*\left(e^{i\xi/2^{r-j}}\right) = \prod_{j=1}^{r} \frac{1}{2}\, g_0^*\left(e^{i\xi/2^j}\right),$$

and which allows us to define a *limit function*

$$\varphi(\xi) := \varphi_\infty(\xi) := \lim_{r\to\infty} \varphi_r(\xi) = \prod_{j=1}^{\infty} \frac{1}{2}\, g_0^*\left(e^{i2^{-j}\xi}\right), \tag{3.5.5}$$

provided, of course, that the infinite product converges. Then the limit φ has a remarkable property, namely,

$$\begin{aligned}\varphi(\xi) &= \frac{1}{2}\, g_0^*\left(e^{i\xi/2}\right) \prod_{j=2}^{\infty} \frac{1}{2}\, g_0^*\left(e^{i2^{-j}\xi}\right) = \frac{1}{2}\, g_0^*\left(e^{i\xi/2}\right) \underbrace{\prod_{j=1}^{\infty} \frac{1}{2}\, g_0^*\left(e^{i2^{-j}(\xi/2)}\right)}_{=\varphi(\xi/2)} \\ &= \frac{1}{2}\, g_0^*\left(e^{i\xi/2}\right)\, \varphi\left(\frac{\xi}{2}\right). \end{aligned} \tag{3.5.6}$$

If we assume that $\varphi \in L_1(\mathbb{R})$, we can apply an inverse Fourier transform and obtain a uniformly conntinuous function $\phi = \varphi^\vee$.

Remark 3.5.1. All the ϕ_r are composed from *trigonometric polynomials*, hence reasonably defined on $2^r\mathbb{T}$ due to the dilation involved in the definition, the limit function must be an L_1 function on all of $\mathbb{R}$ as the limit. This is one of the subtleties that make the complete proofs a little bit more complex even if the main ideas are fairly natural and straightforward.

With $\varphi = \hat{\phi}$, (3.5.6) becomes

$$\hat{\phi}(\xi) = \frac{1}{2}\, g_0^*\left(e^{i\xi/2}\right)\, \hat{\phi}\left(\frac{\xi}{2}\right) = \frac{1}{2}\hat{g}_0\left(\frac{\xi}{2}\right)\, \hat{\phi}\left(\frac{\xi}{2}\right) = [(g_0 * \phi)\,(2\cdot)]^\wedge\,(\xi),$$

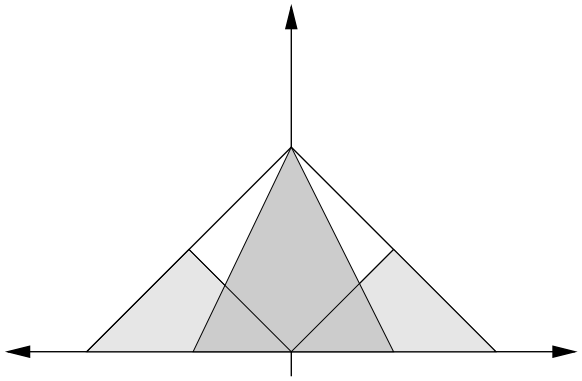


Figure 3.5.1: The piecewise linear **hat function** and its representation via the refinement equation (3.5.7). Just sump up the "narrow" triangles to obtain the "wide" one.

hence,

$$\phi = (g_0 * \phi)\ (2\cdot) = \sum_{k\in\mathbb{Z}} g_0(k)\,\phi\,(2\cdot - k)\,. \tag{3.5.7}$$

This *fundamental* identity is called **refinement equation** or **two scale relation** and means that the function can be represented as combination of its squeezed copies, see Fig. 3.5.1. This is *not at all* a common property of functions, quite the contrary, the functions that satisfy (3.5.7) are precisely those that result from a convergent subdivision process.

But "convergent" is the word. Since we want the limit to exist, we need criteria for the convergence of the infinite product in (3.5.5). As the sequence in an infinte series has to converge to zero, the factors in an infinite product have to converge to 1 which leads to the necessary condition

$$1 = \frac{1}{2}\lim_{r\to\infty} g_0^*\left(e^{i2^{-r}\xi}\right) = \frac{1}{2}\,g_0^*(1) = \frac{1}{2}\,\hat{g}_0(0). \tag{3.5.8}$$

This however means that, properly normalized, G_0 is a lowpass filter and reproduces constant signal, and this is also the reason why in wavelet analysis we only cascade on lowpass parts of the signal.

A *sufficient* condition for the existence of a continuous function ϕ that satisfies the refinement equation (3.5.7) is given in the following result due to Daubechies (Daubechies, 1988), which is taken together with its proof from (Vetterli and Kovačević, 1995).

Proposition 3.5.2 (Existence of continuous refinable functions). *If g_0^* can be written as*[22]

$$g_0^*(z) = \left(\frac{1+z}{2}\right)^k q(z) \qquad \textit{where} \qquad \max_{z\in\mathbb{T}} |q(z)| < 2^k \quad \textit{and} \quad q(1) = 2, \tag{3.5.9}$$

then there exists a continuous *solution of* (3.5.7).

[22] Recall the general assumption that G_0 is an FIR filter, hence $g_0 \in \ell_0(\mathbb{Z})$ and thus g_0^* is a Laurent polynomial.

Proof: We will show that under the assumption (3.5.9) the infinite product (3.5.5) converges (pointwise) and yields L_1-function whose inverse Fourier transform has to exist and is the uniformly continuous function ϕ.

We first decompost the product into

$$\prod_{j=1}^{\infty} \frac{1}{2}\hat{g}_0\left(2^{-r}\xi\right) = \prod_{j=1}^{\infty} \left(\frac{1+e^{i2^{-r}\xi}}{2}\right)^k \prod_{j=1}^{\infty} \frac{1}{2}\hat{q}\left(2^{-r}\xi\right) \tag{3.5.10}$$

and treat the factors on the right hand side separately. Since

$$\begin{aligned} \frac{1-e^{i\xi}}{\xi} &= \frac{1+e^{i\xi/2}}{2}\,\frac{1-e^{i\xi/2}}{\xi/2} = \frac{1+e^{i\xi/2}}{2}\,\frac{1+e^{i\xi/4}}{2}\,\frac{1-e^{i\xi/4}}{\xi/4} \\ &= \cdots = \underbrace{\frac{1-e^{i2^{-r}\xi}}{2^{-r}\xi}}_{\to i} \prod_{r=1}^{N} \frac{1+e^{i2^{-r}\xi}}{2}, \end{aligned}$$

hence

$$\prod_{r=1}^{\infty} \frac{1+e^{i2^{-r}\xi}}{2} = \frac{1-e^{i\xi}}{i\,\xi},$$

the first product has the value

$$\left(\frac{1-e^{i\xi}}{i\,\xi}\right)^k = \left(e^{i\xi/2}\,\frac{e^{-i\xi/2}-e^{i\xi/2}}{\xi}\right)^k = (-i)^k e^{ik\xi/2}\left(\frac{\sin\xi/2}{\xi/2}\right)^k,$$

which is bounded in modulus by $\le C_1\,(1+|\xi|)^{-k}$ for an appropriate constant $C_1 > 0$.

For the second factor in (3.5.10), we use the abbreviation $h = \frac{1}{2}q$. Since Da $h(1) = 1$, there exists a constant $C_2 > 0$ such that for $|\xi| \le 1$ the estimate[23] $\left|h\left(e^{i\xi}\right)\right| \le 1 + C_2|\xi| \le e^{C_2|\xi|}$ holds, hence we have, for $|\xi| \le 1$,

$$\prod_{j=1}^{\infty} \left|h\left(e^{i2^{-r}\xi}\right)\right| \le \prod_{j=1}^{\infty} e^{C_2 2^{-r}|\xi|} = \exp\left(\sum_{j=1}^{\infty} \frac{C_2|\xi|}{2^r}\right) = e^{C_2|\xi|} \le e^{C_2}. \tag{3.5.11}$$

For arbitrary $\xi \in \mathbb{R}$ we now choose $n \in \mathbb{N}$ such that $2^{n-1} \le |\xi| < 2^n$, and use the following decomposition together with (3.5.11):

$$\begin{aligned} \prod_{j=1}^{\infty} \left|h\left(e^{i2^{-r}\xi}\right)\right| &= \prod_{j=1}^{n} \left|h\left(e^{i2^{-r}\xi}\right)\right| \prod_{j=n+1}^{\infty} \left|h\left(e^{i2^{-r}\xi}\right)\right| \\ &= \prod_{j=1}^{n} \left|h\left(e^{i2^{-r}\xi}\right)\right| \prod_{j=1}^{\infty} \left|h\left(e^{i2^{-r}(\xi/2^n)}\right)\right| \le \prod_{j=1}^{n} \left|h\left(e^{i2^{-r}\xi}\right)\right| \; e^{C_2} \le B^n\, e^{C_2}, \end{aligned}$$

[23] This may appear slightly magic at first, but this fairly standard trick is only based on the quite simple observation that the functions $1 + cx$ und e^{cx} have the same value for $x = 0$, but the derivatives c und $c\,e^{cx} \le c$ are different.

where

$$B := \max_{z\in\mathbb{T}} |h(z)| \le 2^{k-1-\varepsilon} \qquad \text{for } \varepsilon > 0.$$

Consequently,

$$B^n \le 2^{n(k-1-\varepsilon)} \le \underbrace{\left(2\,2^{n-1}\right)}_{\le 2\,|\xi|\le 1+|\xi|}{}^{k-1-\varepsilon} \le 2^k\ (1+|\xi|)^{k-1-\varepsilon}$$

and therefore

$$\prod_{j=1}^{\infty} \left| g_0^*\left(e^{i2^{-r}\xi}\right)\right| \le C_1\ (1+|\xi|)^{-k}\ e^{C_2 k}\ 2^k\ (1+|\xi|)^{k-1-\varepsilon} \le C_3\,(1+|\xi|)^{-1-\varepsilon}\,.$$

Hence the product is convergent everywhere[24], the resulting function belongs to $L_1(\mathbb{R})$ and admits an inverse Fourier transform. □

Exercise 3.5.1 Show: Whenever ϕ is a nontrivial solution of the refinement equation (3.5.7), one has $\hat{\phi}(0) \neq 0$ and $\hat{g}(0) = 2$. ◊

Remark 3.5.3. The requirement (3.5.2) concerns f_0 and not really the sequence g_0 that is relevant for the refinement equation (3.5.7). Since we can replace z by z^{-1} in (3.5.2), however, it is completely irrelevant whether the sufficient condition is formulated in terms of f_0 or g_0.

So, let ϕ denote the[25] solution of the refinement equation (3.5.7), then we already know three possibilities to to construct this function, even numerically: konstruierten k"onnen:

1. via the Fourier transform

$$\phi = \left(\prod_{j=1}^{\infty} \frac{1}{2}\,\hat{g}_0\left(-\frac{\cdot}{2}\right)\right)^{\vee}; \tag{3.5.12}$$

   this values could be computed at integer points and then an FFT could be applied.

2. via the **cascade scheme**

$$\phi = \lim_{j\to\infty} T_G^j\,\psi, \qquad T_G\psi := (g_0 * \psi)\,(2\cdot)\,, \tag{3.5.13}$$

   with a reasonably chosen start function ψ. This methdod constructs a sequence of functions that convergs to a *fixpoint* of the **transfer operator** T_G. And yes, all this has a lot do with the famous Banach Fixpoint Theorem and contractions.

[24] If you want to do it fully correct, you have to invoke *dominated convergence* here.

[25] The "the" is not so trivial! Not every refinement equation that permits a solution also has *unique* solution. The reasonable ones do, but this is a different story to be told somewhere else.

3. via the **subdivision scheme**

$$\phi = \lim_{j\to\infty} S_G^j \delta, \qquad \lim_{j\to\infty} \sup_{k\in\mathbb{Z}} |\phi\left(2^{-r}k\right) - S_G\delta(k)| = 0, \tag{3.5.14}$$

where now the limit function is determined discretely at a denser and desner set of points.

Making use of the linearity of all the operators involved in the process, we can identify limit functions of subdivision as convolutions.

Corollary 3.5.4. *If the subdivision scheme converges in the sense (3.5.14), then we have for any initial data c that*

$$S_G^r c = \phi * c = \sum_{k\in\mathbb{Z}} c(k)\, \phi\left(\cdot - k\right).$$

Based on what we did so far, the wavelets will only be a matter of *interpretation* of the filterbank operations. Having ϕ at hand, a signal $c \in \ell(\mathbb{Z})$ will not be seen as a discrete function on $\mathbb{Z}$ but as *coefficients* of the function

$$f_c := c * \phi = \sum_{k\in\mathbb{Z}} c(k)\, \phi(\cdot - k), \tag{3.5.15}$$

hence, we associate to the sequence the limit function of the subdivision scheme. The simplest examples for such approximations are the piecewise constant or piecewise linear function $\phi = \chi_{[0,1]}$ or $\phi = \chi_{[0,1]} * \chi_{[0,1]}$, respectively. The latter are the hat functions of Fig. 3.5.1. Since $\phi = T_G\phi$,

$$\begin{aligned} f_c &= \sum_{j\in\mathbb{Z}} c(j)\, \phi(\cdot - j) = \sum_{j\in\mathbb{Z}} c(j)\ \left(T_G\phi\right)(\cdot - j) = \sum_{j\in\mathbb{Z}} c(j)\ \sum_{k\in\mathbb{Z}} g_0(k)\, \phi\left(2\cdot - 2j - k\right) \\ &= \sum_{j\in\mathbb{Z}}\sum_{k\in\mathbb{Z}} c(j)\, g_0(k)\, \phi\left(2\cdot - 2j - k\right) = \sum_{k\in\mathbb{Z}} \underbrace{\sum_{j\in\mathbb{Z}} g_0\left(k - 2j\right)\ c(j)}_{=S_G c(k)}\, \phi\left(2\cdot - k\right) \\ &= \left(S_G c * \phi\right)(2\cdot), \end{aligned}$$

$S_G c$ corresponds to the coefficients of *the same* function f_c with respect to the squeezed function $\phi(2\cdot)$. And this brings us to wavelets: the input and output signal c of the perfect reconstruction filterbank

$$c \to \odot \begin{array}{c} \nearrow \\ \searrow \end{array} \begin{array}{c} \boxed{F_0} \\ \boxed{F_1} \end{array} \to \begin{array}{c} \boxed{\downarrow_2} \\ \boxed{\downarrow_2} \end{array} \to \begin{array}{c} c_0 \\ c_1 \end{array} \to \begin{array}{c} \boxed{\uparrow_2} \\ \boxed{\uparrow_2} \end{array} \to \begin{array}{c} \boxed{G_0} \\ \boxed{G_1} \end{array} \begin{array}{c} \searrow \\ \nearrow \end{array} \oplus \to c \tag{3.5.16}$$

will be interpreted as the coefficients of a function $f = c * \phi(2\cdot)$ belonging to the space

$$V_1 = \text{span}\ \{\phi\left(2\cdot - k\right)\ :\ k \in \mathbb{Z}\}.$$

The space is generated by dilated copies fo the refinable **scaling function** ϕ, shifted by $k/2$, $k \in \mathbb{Z}$. The refinement equation (3.5.7) can also be as an explicit representation of ϕ as a function in V_1 since V_1 is **shift invariant**, that is $f \in V_1$ implies $f(\cdot - k) \in V_1$, $k \in \mathbb{Z}$, we get that

$$V_1 \supseteq V_0 := \text{span } \{\phi(\cdot - k) \ : \ k \in \mathbb{Z}\}. \tag{3.5.17}$$

Settinh

$$V_j = \text{span } \left\{\phi\left(2^j \cdot - k\right) \ : \ k \in \mathbb{Z}\right\},$$

the refinement equation (3.5.7) yields

$$V_0 \subseteq V_1 \subseteq V_2 \subseteq \cdots$$

This means that the spaces V_j, $j \in \mathbb{N}$, or even $j \in \mathbb{Z}$, form a **Multiresolution Analysis** or **MRA**, a concept introduced by Mallat, see, for exmple (Daubechies, 1992; Louis et al., 1998; Mallat, 1989; Mallat, 1999; Vetterli and Kovačević, 1995). The (minimal) properties of an MRA are:

1. a *nested* scale of function spaces $V_0 \subseteq V_1 \subseteq \cdots$
2. shift invariance of the spaces V_j,
3. a two scale relation $f \in V_j \Rightarrow f(2\cdot) \in V_{j+1}$.

Usually there are further conditions on an MRA, like being subspaces of $L_2(\mathbb{R})$ and being generated by a so-called *Riez basis*, but we stick to the core points here as we are going to derive it diretly from the filterbank.

Indeed, in our filterbank (3.5.16) we decomposed the signal c or, equivalently[26], the associated function $c * \phi(2\cdot)$ into the subbands c_0 and c_1 that have to be interpreted accordingly. Due to our construction, the operation $c_0 = \downarrow_2 F_0\, c$, i.e., the determination of $c_0 * \phi$, is a **projection** from V_1 to V_0 if and only if it satisfies

$$c * \phi(2\cdot) \in V_0 \qquad \Leftrightarrow \qquad c_1 = 0. \tag{3.5.18}$$

But (3.5.18) simply follows from the fact that the modulation matrix M can be inverted: if there were two representations c_0, c_1 and $\widetilde{c}_0, \widetilde{c}_1$ such that

$$G_0 \uparrow_2 c_0 + G_1 \uparrow_2 c_1 = G_0 \uparrow_2 \widetilde{c}_0 + G_1 \uparrow_2 \widetilde{c}_1,$$

their z-transforms satisfy

$$0 = \widetilde{M}(z) \left(\begin{pmatrix} c_0^*(z^2) \\ c_1^*(z^2) \end{pmatrix} - \begin{pmatrix} \widetilde{c}_0^*(z^2) \\ \widetilde{c}_1^*(z^2) \end{pmatrix} \right) \qquad \Rightarrow \qquad \begin{pmatrix} c_0^*(z^2) \\ c_1^*(z^2) \end{pmatrix} = \begin{pmatrix} \widetilde{c}_0^*(z^2) \\ \widetilde{c}_1^*(z^2) \end{pmatrix}.$$

Since

$$V_0 \ni f = c * \phi = (S_G c * \phi)(2\cdot)$$

the combination $c_0 = S_G c$ and $c_1 = 0$ is exactly this *unique* way to represent $f \in V_0$, which proves (3.5.18).

[26] At least if the relation between c and $c * \phi$ is bijective or in some way controlable. This is the Riez basis issue, by the way.

Definition 3.5.5. The **wavelet** for the scaling function ϕ is the function with respect to the synthesis part of a perfect reconstruction filterbank is

$$\psi = \sum_{k\in\mathbb{Z}} g_1(k)\,\phi\,(2\cdot - k)\,. \tag{3.5.19}$$

Remark 3.5.6. If $n \geq 2$ one has to use the $n-1$ wavelets,

$$\psi_j = \sum_{k\in\mathbb{Z}} g_j(k)\,\phi\,(2\cdot - k)\,, \qquad j = 1,\ldots,n-1. \tag{3.5.20}$$

The rest of the extension is straightforward.

Lemma 3.5.7. *If the filterbank has perfect reconstruction, then for any $c \in \ell(\mathbb{Z})$,*

$$c * \phi(2\cdot) = c_0 * \phi + c_1 * \psi. \tag{3.5.21}$$

Proof: Perfect reconstruction means

$$c = G_0\ \uparrow_2\ c_0 + G_1\ \uparrow_2\ c_1 = \sum_{j=0}^{1}\sum_{k\in\mathbb{Z}} g_j\,(\cdot - 2k)\,c_j(k)$$

and therefore

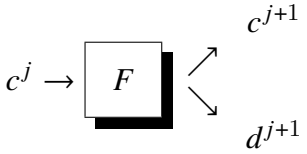
$$\begin{aligned}
c * \phi\,(2\cdot) &= \sum_{\ell\in\mathbb{Z}} c(\ell)\,\phi\,(2\cdot - \ell)\\
&= \sum_{\ell\in\mathbb{Z}}\sum_{j=0}^{1}\sum_{k\in\mathbb{Z}} g_j\,(\ell - 2k)\,c_j(k)\,\phi\,(2\cdot - \ell)\\
&= \sum_{k\in\mathbb{Z}}\sum_{\ell\in\mathbb{Z}}\sum_{j=0}^{1} g_j\,(\ell)\,c_j(k)\,\phi\,(2\cdot - (\ell + 2k))\\
&= \sum_{k\in\mathbb{Z}} c_0(k)\underbrace{\sum_{\ell\in\mathbb{Z}} g_0(\ell)\,\phi\,(2(\cdot - k) - \ell)}_{=\phi(\cdot - k)} + \sum_{k\in\mathbb{Z}} c_1(k)\underbrace{\sum_{\ell\in\mathbb{Z}} g_0(\ell)\,\phi\,(2(\cdot - k) - \ell)}_{=\psi(\cdot - k)}\\
&= c_0 * \phi + c_1 * \psi.
\end{aligned}$$

□

It is almost too simple to be true, but this is already the fundamental point in practically any type of wavelet decomposition of functions, just up to some iteration. To better understand what we really are doing, we write the decomposition of the filterbank in a slightly different way. To that end, we set

$$c^0 := c, \qquad c^{j+1} :=\downarrow F_0 c^j, \qquad d^{j+1} :=\downarrow F_0 c^j, \tag{3.5.22}$$

or, schematically,



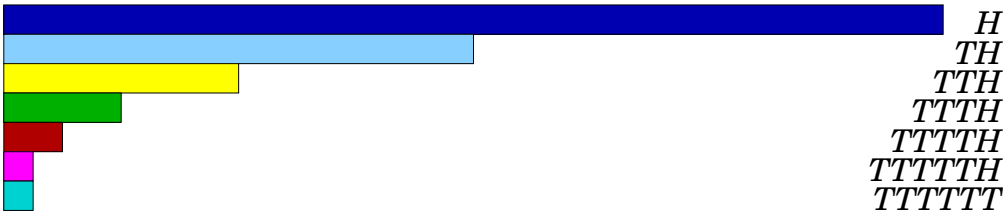


Figure 3.5.2: The decomposition *pyramid*, shifted to the left and put upside down. H and T indicate the application of high- and low-pass filters (German: *"Hochpass"* and *"Tiefpass"*.

and then *iterate* this procedure on the low-pass part of the signal:

$$\begin{array}{ccccccccc} c = c^0 & \to & c^1 & \to & c^2 & \to \cdots \to & c^n \\ & \searrow & & \searrow & & & \searrow \\ & & d^1 & & d^2 & & d^n \end{array} \qquad c \simeq \left(c^n, d^1, \ldots, d^n\right).$$

If the filterbank provides perfect reconstruction, we can reconstruct c from the decomposition by simply inverting the process:

$$\begin{array}{ccccccccc} c^n & \to & c^{n-1} & \to & \cdots \to & c^1 & \to & c^0 = c. \\ & \nearrow & & \nearrow & & & \nearrow \\ d^n & & d^{n-1} & & & d^1 \end{array}$$

This way any perfect reconstruction filterbank defines an invertable transformation

$$c \leftrightarrow \left(c^n, d^1, \ldots, d^n\right) \tag{3.5.23}$$

which is known as **discrete wavelet transform**, **DFT** or **pyramid scheme**. The latter name is due to the fact that d^j contains essentially half of the information of c^{j-1} and then the data can be arranged in a pyramid-like fashion, see Fig. 3.5.2.

Theorem 3.5.8 (**Wavelet decomposition** of functions). *If the filterbank provides perfect reconstruction, then*

$$c * \phi\left(2^n \cdot\right) = c^n * \phi + \sum_{j=1}^{n} d^j * \psi\left(2^{n-j} \cdot\right). \tag{3.5.24}$$

Proof: With all our previous work, the proof is surprisingly simple. We just use (3.5.21) and get for any $x \in \mathbb{R}$ that

$$\begin{aligned} c * \phi\left(2^n x\right) &= c * \phi\left(2\left(2^{n-1}x\right)\right) = c^1 * \phi\left(2^{n-1}x\right) + d^1 * \psi\left(2^{n-1}x\right) \\ &= c^2 * \phi\left(2^{n-2}x\right) + d^2 * \psi\left(2^{n-1}x\right) + d^1 * \psi\left(2^{n-1}x\right) \\ &= \cdots = c^n * \phi(x) + \sum_{j=1}^{n} d^j * \psi\left(2^{n-j}x\right). \end{aligned}$$

which is just (3.5.24). □

Remark 3.5.9 (Interpretation of the DWT). The interesting point is the *interpretation* of Theorem 3.5.8:

1. Although the decomposition in (3.5.24) is a decomposition in terms of *functions*, all *computations* are performed exclusively on the coefficients and thus are fully discrete. Even better, we only have to apply filterbanks that can be implemented efficiently, in a parallel way and, if needed, on specialized hardware.

2. Once more, the function $c * \phi\,(2^n \cdot)$ on the left hand side of (3.5.24) is a quasi interpolant like in the Shannon Sampling Theorem. In our case we use copies of the scaling function ϕ that are dilated by a factor 2^n, hence are massively localized, and shifted by $k/2^n$. This corresponds to a fine sampling of a given function.

3. This highly detailed function is then decomposed in the resolution levels $2^{n-1}, 2^{n-2}, \ldots, 1$, which consist of less squeezed copies of ϕ with a lower resolution. Hence, we consciously pass from a high level of detail to coarser and coarser representations of the signal.

4. The *detail information* of the signal is then recorded in the **wavelet coefficients** d^j and the details are finer if the associated index j is smaller.

3.6 Applications

This was quite a bit of theory and the valid question in any applied science[27] is: what can it be used for? Indeed it can be used for a lot of things if we recall the fundamental principle of the DWT:

> The wavelet coefficients are generated by *local* high-pass filters

Here, "local" means that the filters have only finite support and that therefore only the entries $c^0(k)$, $|j-k| \le N$, contribute to $d^1(j)$. Continuing the pyramid, we obtain that $d^r(j)$ only depends on $c^0(k)$, $|j - k/2^r| \le N$. On the other hand, "high-pass" means that the wavelet coefficients are *small* if c is almost constant in some regions, hence is described almost completely by the coarser signal or, in other words, contains fewer additional information. This gives us the hope that such (almost) piecewise signals can be be compressed with small loss by applying the decomposition (3.5.24). This can be formulated *quantitatively* like for example in the following that we neither can[28] nor want to[29] to prove.

[27] And image processing is an applied science.
[28] This would require a substantial amount of additional theory which would be nice, but only for a specialized lecture.
[29] It would lead to far away from image processing.

Theorem 3.6.1. *Under certain conditions on ϕ[30] and ψ[31] there exists a constant $C > 0$ such that for any $n+1$ times differentiable f one has*

$$\left|d^k(j)\right| \leq C 2^{-k(n+1)} \left\|f^{(n+1)}\right\|_2, \qquad j \in \mathbb{Z}. \tag{3.6.1}$$

There also exists a local *version of this result.*

Theorem 3.6.1 tells us that whenever f has the maximal smoothness $n+1$ everywhere, then the wavelet coefficients d^k all decay of the order $2^{-k(n+1)}$. If, on the other hand, f is less smooth, for example only differentiabl of order n', $n' < n$, then the requirements on ϕ and ψ are still satisfied and we can apply the theorem once more to obtain a decay rate $2^{-k(n'+1)}$, hence the smoothness is related in a well-defined way with the rate of decay. To summarize:

> The smoother the function, the higher the decay rate.

Even if a simple converse of (3.6.1) does not exist since *dyadic singularities*, i.e., singularities at the dyadic points $2^{-k}\mathbb{Z}$, cannot be detected, we can approximate the smoothness of f in some neighborhood of $x \in \mathbb{R}$ by

$$\lim_{k\to\infty} -k^{-1} \log_2 \left|d^k\left(\lfloor 2^k x \rfloor\right)\right|;$$

the decay rate of the coefficients can be used for smoothness detection!

And this brings us back to our applications in Image and Signal Processing since the interesting parts of images, the **features**, usually are defined by lesser smoothness than in the "nice" or "boring" components of the image. This makes the wavelet decomposition some sort of Swiss Army Knife in Image and Signal Processing since it offers several advances simultaneously:

1. in compression one forms a wavelet decomposition and only keeps a certain fraction of the components, namely those of largest modulus,
2. in feature detection, one looks for locations where *all* corresponding wavelet coefficients $d^k\left(2^{-k}j\right)$ are significant,
3. in denoising one sets all small wavelet coefficients to zero.

Example 3.6.2 (Features). Let us have a look a the edges of a simple hat function, see Fig. 3.6.1. We can see very clearly that the normalized wavelet coefficients "point" to singularity and that their absolute value increases the more "point" that corner is. This should actually come as no surprise since the angles are related to the curvature, hence the second derivative.

Example 3.6.2 is actually quite impressive but academic and artificial, to be honest. Real world signals are not that simple and easy but contain a little bit of everything.

[30] The conditions are compact support and the so-called **Strang-Fix conditions** that ensure that all polynomials of degree n can be written in the form $c * \phi$ for some appropriate signal $c \in \ell(\mathbb{Z})$

[31] Compact support and — just as a warning — the support size is a factor of the constant C in 3.6.1, hence FIR is a fundamental requirement again. Moreoever, we need $n+1$ vanishing moments which is actually closely tied to the Strang–Fix conditions on ϕ.

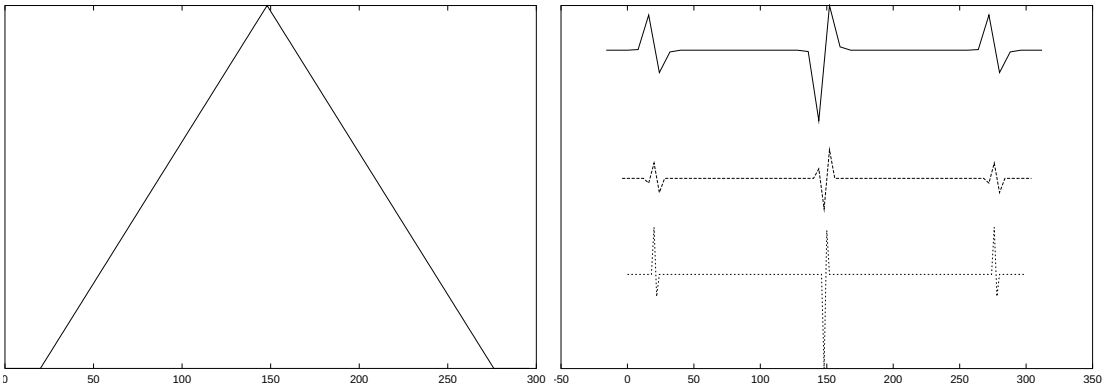


Figure 3.6.1: A simple piecewise linear functions *(left)* and its wavelet coefficients *(right)* with respect to Strang-Fix conditions of sufficiently high order.

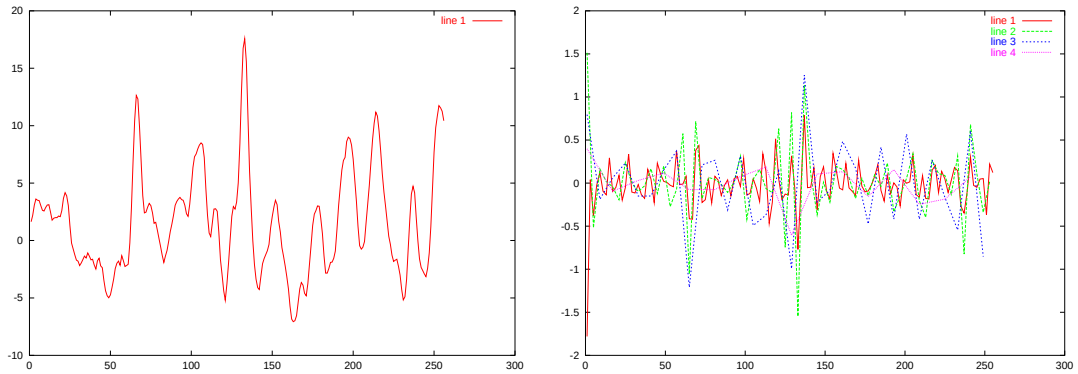


Figure 3.6.2: EEG measurement of brain activity *(left)* and the associated wavelet coefficients *(right)*.

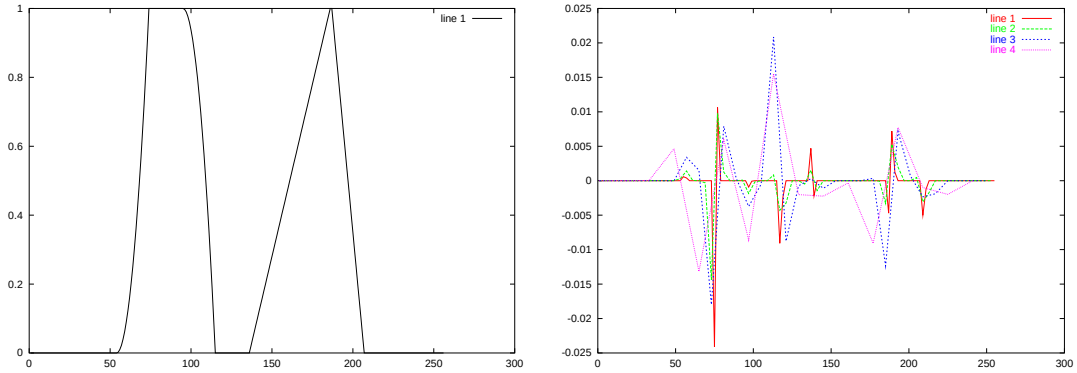


Figure 3.6.3: A simple test signal *(left)* and its wavelet coefficients *(right)*. It can be clearly seen that the localization of the wavelet coefficients is far from perfect.

Example 3.6.3. Fig. 3.6.2 shows a part of an EEG (electroencephalography) signal einem EEG–Signal und die zugeh"origen Waveletkoeffizienten. Man sieht doch, da"s in diesem Fall die “Spitzen” wesentlich unsch"arfer sind und die “Ecken” und “Kanten” nicht so scharf charakterisieren.

So far we have not considered the effect of the filter length on the decomposition. We could bypass that by using the philosophy of the DWT and embed finitely supported signals c with[32] $\operatorname{supp} c \subset [0, N-1]$ into $\ell(\mathbb{Z}_N)$ and then use periodicity all the way. This, however, is not without problems since c may or may not have a periodic structure and we may well end up with periodization artifacts, especially when doing feature detection. Avoiding peridicity, we can simply compute the support of $g * c$, assuming that the filter g is supported on the interval $[0, n]$ and the signal c on the interval $[0, N]$. The result of the convolution,

$$g * c(j) = \sum_{k\in\mathbb{Z}} g(j-k)\, c(k) = \sum_{k=0}^{N} g(j-k)\, c(k)$$

is nonzero if $j - k \in [0, n]$, hence $j \in k + [0, n]$ or $j \in [0, N + n]$. The filter “blurs” the signal across its boundary and this effect becomes the more prominent and intensive the longer the filter is.

Example 3.6.4 (Artifacts). Here we briefly discuss two examples of artifacts that can occur in the real world application of wavelets.

1. In Fig. 3.6.3 we see a “test signal” with breakpoints of different orders of regularity. The wavelet coefficients still point to these singularities, but they are much less focused. Morevoer, we see significant wavelet coefficients on the boundary that are *pure articfacts*, see Fig. 3.6.4.

2. In general, the artifacts cannot be prevented by *any* type of signal extension, whichever clever way one intends to choose them. Here we used zero padding, but also periodization is no general solution, as for example Fig. 3.6.5 showed quite drastically.

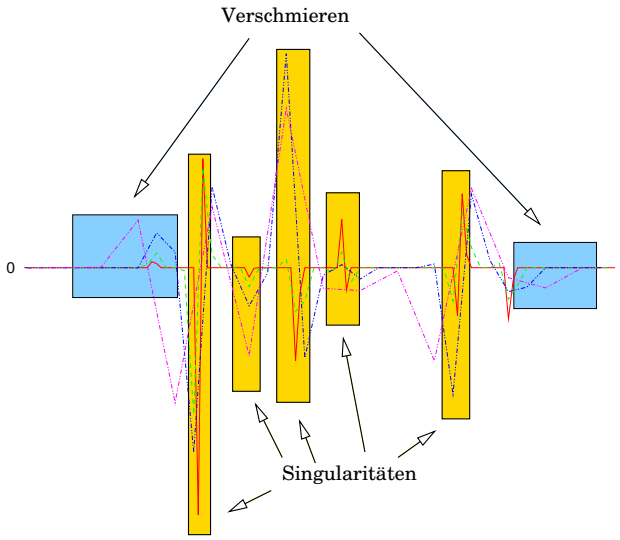


Figure 3.6.4: Explanation of the wavelet coefficients in Fig. 3.6.3. The interior wavelet coefficients are blurred or "leak" like in the windowed Fourier transform, on the boundary we have extension artifacts due to the support of the filter.

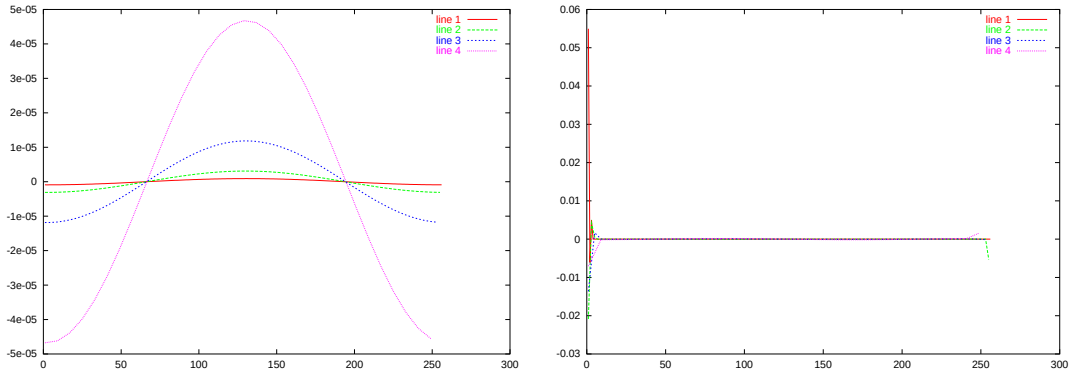


Figure 3.6.5: Wavelet coefficients of periodic extensions of the sine function, where once the period is chosen accordingly to that of the sine *(left)*, and once in the "wrong way" *(right)*. Note the scaling of the axes! Hence, without *a priori* knowledge of the signal, periodic extensions can fail utterly.

The main tool in our application of wavelets for signal processing purposes, especially in denoising and compression, is the concept of *thresholding*.

Definition 3.6.5. Given a constant $\theta \in \mathbb{R}_+$, a **thresholding** operator $T_\theta : \ell(\mathbb{Z}) \to \ell(\mathbb{Z})$ maps small values to zero via

1. (**hard thresholding**)

$$T_\theta^h c(k) = \begin{cases} c(k), & |c(k)| \geq \theta, \\ 0, & |c(k)| < \theta, \end{cases} \qquad k \in \mathbb{Z}. \tag{3.6.2}$$

2. (**soft thresholding**)

$$T_\theta^s c(k) = \operatorname{sgn} c(k)\ (|c(k)| - \theta)_+ = \begin{cases} c(k) - \theta, & c(k) \geq \theta, \\ 0, & |c(k)| \leq \theta, \\ c(k) + \theta, & c(k) < -\theta, \end{cases} \qquad k \in \mathbb{Z}. \tag{3.6.3}$$

One advantage of soft thresholding or **shrinkage** compared to hard thresholding is that the mapping

$$c \mapsto \operatorname{sgn} c\ (|c| - \theta)_+$$

is continuous, but for the price that the whole signal is reduced by the constant θ which results in a loss of *contrast*. The real advantage, however, is that soft thresholding solves a minimization problem.

Proposition 3.6.6. *For $c \in \ell_{00}(\mathbb{Z})$ one has*

$$T_\theta^s c = \operatorname*{argmin}_x \|c - x\|_2^2 + 2\theta\ \|x\|_1 \tag{3.6.4}$$

Proof: Assuming that c vanishes outside $[0, N]$, we can immediately conclude that so also does the solution c^* of (3.6.4) as otherwise we would just have unnecessary terms that increase both terms in (3.6.4). Hence, we can rewrtie the functional to be minimized as

$$\lambda_c(x) = \sum_{j=0}^{N} (c(j) - x(j))^2 + 2\theta \sum_{j=0}^{N} |x(j)| = \sum_{j=0}^{N} \Big((c(j) - x(j))^2 + 2\theta\, |x(j)| \Big),$$

and this is a *decoupled* optimization problem that can be solved *separately* for any pair $c(j)$ and $x(j)$, $j = 0, \ldots, N$. So we only have to find out how to find the solution

$$c^* = \min_x (c - x)^2 + 2\,\theta\,|x| \tag{3.6.5}$$

of the scalar problem. The first observation is that x has to have the same sign as c as otherwise $-x$ would give a smaller value in (3.6.5). In particular, $c = 0$ implies $x = 0$. Hence, for $c \geq 0$ we have to minimize the function

$$(c - x)^2 + 2\,\theta\,|x| = (c - x)^2 + 2\,\theta\,x$$

[32] This can always be ensured by a proper shift.

with respect to $x \in [0, \infty)$, and the minimum is either assumed for $x = 0$ or for the value x that solves

$$0 = \frac{d}{dx}\left((c-x)^2 + 2\,\theta\,x\right) = 2(x - c + \theta) \qquad \Leftrightarrow \qquad x = c - \theta.$$

Here, $x > 0$ can happen only for $c > \theta > 0$, but then the solution is the global minimum. For $c \leq \theta$, on the the other hand, the minimal solution is $x = 0$. An analogous argument for $c < 0$ yields that $x = 0$ or $x = c + \theta$ and all together we obtain (3.6.3). □

Remark 3.6.7. There is much more behind the concept of shrinkage than we can explain here. Convex optimization problems like the one in (3.6.5) play a fundamental role in modern image processing and convex optimization based on subgradients is part of nowaday's image processing toolbox, cf. (Chambolle et al., 2010).

Another important degree of freedom is how to choose the parameter θ that can either be a fixed number independent of c, but which should better be determined depending on $c \in \ell_0(\mathbb{Z})$

- as $\lambda \|c\|_\infty$, $0 < \lambda < 1$, hence as a fraction of the maximal value occuring in c,
- as a fraction of the average of moduli,

$$\theta = \lambda \frac{\|c\|_1}{\#c} = \frac{\lambda}{\#c} \sum_{k \in \mathbb{Z}} |c(k)|\,, \qquad \#c := \#\left\{k \;:\; c(k) \neq 0\right\}.$$

- as a relative **median** for $0 < \lambda < 1$:

$$\theta = |c(k)| \qquad \text{such that} \qquad \#\left\{j \;:\; |c(j)| \leq |c(k)|\right\} \sim \lambda\,\#c.$$

All the approaches erase coefficients whose size is small relative to the data c, where the median is the most robust approach but needs most computational effort.

Having the thresholding operator at hand, we can finally sketch the applications where we first compute the decomposition

$$c \mapsto \left(d^1, \ldots, d^n, c^n\right),$$

to which we appy thresholding in the following way:

Edge detection: depending on the order n of the feature, we determine $\hat{d}^k = T_\theta 2^{kn} d^k$ for some relatively large θ and look at the distribution of the surviving coefficients. If they accumulate somewhere, then this is an indicator for a singularity according to Theorem 3.6.1.

Compression: here we for $\hat{d}^k = T_\theta d_k$ as well as $\hat{c}^k = T_\theta c_n$, where θ controls the compression rate and therefore the quality of the result. Aftwards, we only store the nonzero coefficients of $\left(\hat{d}^1, \ldots, \hat{d}^n, c^n\right)$ or use an *entropy encoder* that efficiently encodes the zeros.

Denoising: we again compute decomposition and (soft) threshold followed by a reconstruction step. It can be shown that this wavelet shrinkage with respect to Haar wavelets is an approximation for the so-called *TV regularization*, meanwhile the standard in denoising.

The extension to images requires a fairly modest effort when using the tensor product as we already did with the Fourier transform.

Definition 3.6.8. Let ϕ_1, ϕ_2 be scaling functions with wavelets ψ_1, ψ_2, the the bivariate scaling function is defined as

$$\phi(x,y) = \phi(x)\,\phi(y), \qquad (x,y) \in \mathbb{R}^2.$$

The associated wavelets[33] are

$$\psi_1(x,y) = \phi(x)\,\psi(y), \qquad \psi_2(x,y) = \psi(x)\,\phi(y), \qquad \psi_3(x,y) = \psi(x)\,\psi(y).$$

The bivariat function ϕ generates an MRA we need refinability and a decomposition. Assuming that we have the refinement equation

$$\phi_j = a_j * \phi\,(2\cdot) = \sum_{k\in\mathbb{Z}} a_j(k)\,\phi\,(2\cdot - k)\,, \qquad j = 1,2,$$

it follows in a purely forma way that

$$\begin{aligned}\phi &= \phi(x)\,\phi(y) = \left(\sum_{j\in\mathbb{Z}} a_1(j)\phi_1(2x-j)\right)\left(\sum_{j\in\mathbb{Z}} a_1(k)\phi_2(2y-k)\right)\\ &= \sum_{(j,k)\in\mathbb{Z}^2} a_1(j)a_2(k)\,\phi_1(2x-j)\phi_2(2y-k) = \sum_{\alpha\in\mathbb{Z}^2} (a_1\otimes a_2)\,(\alpha)\phi(2\cdot-\alpha),\end{aligned}$$

and the same trick can be applied to the decomposition formula

$$\phi_j\,(2\cdot) = c'_j * \phi_j + d'_j * \psi_j, \qquad j = 1,2,$$

to yield the following result.

Theorem 3.6.9. *The function ϕ is refinable,*

$$\phi = \sum_{\alpha\in\mathbb{Z}^2} (a_1\otimes a_2)\,(\alpha)\phi\,(2\cdot-\alpha)\,, \tag{3.6.6}$$

and we have

$$\phi\,(2\cdot) = c * \phi + \sum_{j=1}^{3} d_j * \psi_j, \tag{3.6.7}$$

where

$$c = c'_1\otimes c'_2, \qquad d_1 = c'_1\otimes d'_2, \qquad d_2 = d'_1\otimes c'_2, \qquad d_3 = d'_1\otimes d'_2. \tag{3.6.8}$$

[33] Yes, there are several of them.

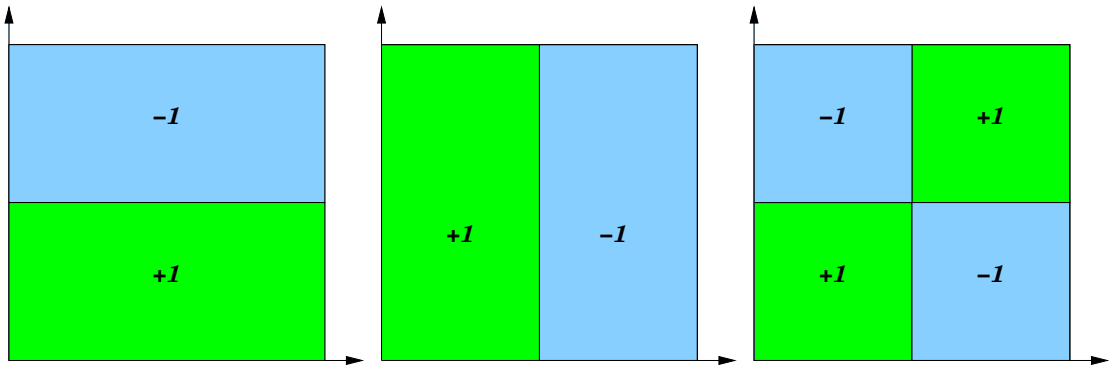


Figure 3.6.6: The three tensor product Haar wavelets: scaling in x and wavelet in y *(left)*, wavelet in x and scaling in y *(center)* and wavelet in both variables *(right)*

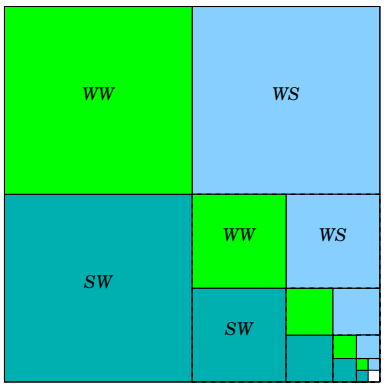


Figure 3.6.7: The wavelet decomposition for images: one obtains three parts for the wavelet components and one coarser low-pass part. Like in one-dimensional signals, the latter one is the decomposed further.

Exercise 3.6.1 Prove Theorem 3.6.9. ◊

Note that now we have to deal with *three* wavelets, where two of them are tensor products of scaling function and wavelet and one is a tensor product of the two wavelets.

Example 3.6.10. The simplest case is that of tensor product Haar wavelets based on $\phi_1 = \phi_2 = \chi_{[0,1]}$ and $\psi_1 = \psi_2 = \chi_{[0,\frac{1}{2}]} - \chi_{[\frac{1}{2},1]}$. The sign distribution of the resulting functions can be seen in Fig. 3.6.6 zu sehen.

The three wavelets in the bivariate case fulfill different tasks: Ψ_1 detects edges parallel to the x-axis, Ψ_2 those parallel to the y-axis, while Ψ_3 seems to care for diagonal ones. The latter is not really true, but sufficient for the intuition. The decomposition of an image is shown schematically in Fig. 3.6.7. The implementation is again done with tensor product filters that can be implemented easily by "cascading" univariate filters as shown before.

For application purposes we decompose our standard image via wavelets, but now we store the scaling part in the *top left* quarter of the image. The decompositions are shown in Fig. 3.6.8. For the wavelet decomposition we used the so-called Daubechies wavelets, where the "4" stands for the number of taps, i.e., nonzero

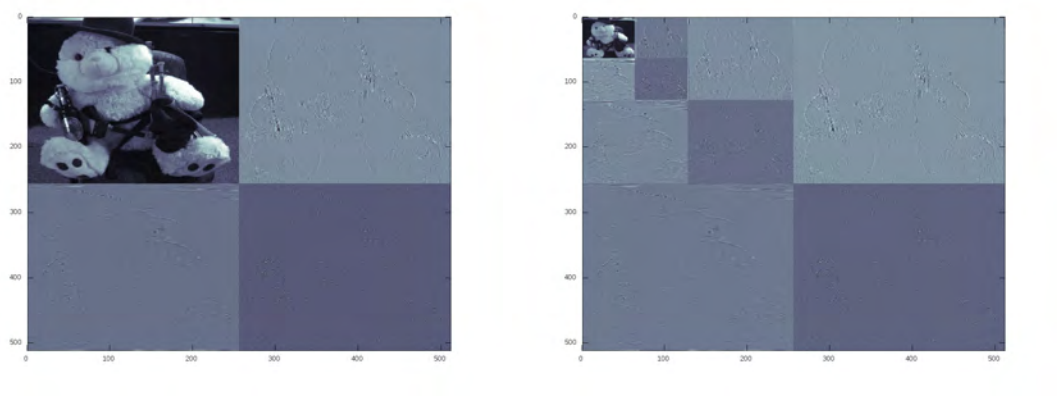


Figure 3.6.8: One *(left)* and three *(right)* levels of wavelet decomposition where the detection of differently directed edges can be seen quite clearly. Moreover, we note that the wavelet parts of the signal are *sparse*, i.e., the contain only very few nonzero entries which are all points with the dominating grey.

filter entries. These wavelets and the decomposition are part of the JPEG2000 standard where instead of a DCT one uses wavelet decompositions of periodized blocks of size $2^n \times 2^n$. And a look at the image shows us why: even without thresholding the wavelet coefficients are already quite sparse.

The Daubechies wavelets belong to a particular class of filter banks, namely the so called **quadrature mirror filters** that are defined by $G(z) = F\left(z^{-1}\right)$, that is, the reconstruction filters are only flipped copies of the analysis filters. Obviously, one has to store fewer coefficients for these filters; for more information see (Daubechies, 1992; Vetterli and Kovačević, 1995).

Further Aspects of Imaging

4

[...] an old Russian proverb which means, roughly, that when the polar bear excrement interferes with the fan belts, the machinery overheats.

(R. Shea, *The Illuminatus! Trilogy*)

Real world data is, unfortunately, not so easy as we would like it to be. In fact, in many applications, for example in Computed Tomography, see Section 1.2.2, or more general Section 1.4, we are interested in a signal f, but all we can measure is

$$g = Tf + \epsilon \tag{4.0.1}$$

where T is a known and hopefully linear operator and ϵ some (random) **noise**. In this chapter we give a very short and superficial introduction to this issue, in general this is almost a science by itself.

4.1 Approximation of Random Signals

Here we change the perspective a bit and look at *stochastic* signals which are modeled as a **random process** which is some sort of a rule to generate random variables, given by density functions. A very good introduction to this subject can be found in Als erstes besch"aftigen wir uns mit (Peebles, 1980).

Definition 4.1.1. A **random process** is a function[1] $f : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$, written as $f(s,t)$ that associates to each $s \in \mathbb{R}$ a **density function** $f(s,\cdot)$ which is called the **realization** of the random process. The **expectation** of the realization is

$$\mathbb{E}_s(f) = \int_{\mathbb{R}} f(s,t)\, dt,$$

the **variance**

$$\mathbb{V}_s(f) = \int_{\mathbb{R}} (f(s,t) - \mathbb{E}_s(f))^2 \, dt$$

and the **standard deviation** is $\sigma_s(f) = \mathbb{V}_s(f)^{1/2}$.

The discrete **random signal** f is then a sequence

$$f(j) = f(s_j, \cdot), \qquad j \in \mathbb{Z}, \tag{4.1.1}$$

[1] *Warning:* we are not considering all details here and will stay vague. It is about the idea, not the (doubtlessly interesting) math behind it.

i.e., each measurement or sample is a realization of the random process and we can only speak about the expectation and the variance of these measurements. Except, of course, then $\sigma_{s_j}(f) = 0$, then the signal is *deterministic.* Assuming that the signal has zero mean, $\mathbb{E}\left(f_j\right) = 0$, the **covariance** of the measurements is the matrix

$$\boldsymbol{K}(f) = \left(\mathbb{E}\left(f(j)\, f(k)\right) \;:\; j,k \in \mathbb{Z}\right).$$

Our first goal is to approximate the random signal by finite information in such a way that the *expected* error becomes as small as possible; since expectation and variance are related to integrals, this will become some L_2 theory.

To that end, let the sequences g_j, $j \in \mathbb{N}_0$, be an *orthonormal basis* of $\ell_2(\mathbb{Z})$ with respect to the inner product

$$\langle f, g\rangle = \sum_{j\in\mathbb{Z}} f(j)\, g(j).$$

Then the **projection** to the subspace spanned by $g_0, \ldots, g_m$ is

$$f_m := \sum_{j=0}^{m} \left\langle f, g_j\right\rangle g_j = \sum_{j=0}^{m}\sum_{k\in\mathbb{Z}} f(k)\, g_j(k)\, g_j.$$

And one fundamental question in random signal processing is

> *How to choose $g_0, \ldots, g_m$ in such a way that the **expected** error becomes minimal?*

In other words, we have to minimize

$$\varepsilon_m := \mathbb{E}\left(\|f - f_m\|^2\right) = \mathbb{E}\left(\left\|\sum_{j=m+1}^{\infty} \left\langle f, g_j\right\rangle g_j\right\|^2\right) = \sum_{j=m+1}^{\infty} \mathbb{E}\left(\left\langle f, g_j\right\rangle^2\right)$$

by finding a proper orthonormal system g_j of signals. Then we can approximate f in a deterministic way as

$$f = \sum_{j=0}^{\infty} \left\langle f, g_j\right\rangle g_j.$$

Since for an arbitrary $x \in \ell_2(\mathbb{Z})$ we have

$$\begin{aligned}
\mathbb{E}\left(\langle f, x\rangle^2\right) &= \mathbb{E}\left(\left(\sum_{j\in\mathbb{Z}} f(j)\, x(j)\right)^2\right) = \mathbb{E}\left(\sum_{j,k\in\mathbb{Z}} f(j)\, f(k)\, x(j)\, x(k)\right) \\
&= \mathbb{E}\left(x^T \left(f(j)\, f(k) : j,k \in \mathbb{Z}\right) x\right) = x^T \left(\mathbb{E}\left(f(j)\, f(k)\right) : j,k \in \mathbb{Z}\right) x \\
&= x^T \boldsymbol{K}(f)\, x = \langle \boldsymbol{K}(f)\, x, x\rangle,
\end{aligned}$$

we can express the expected error as

$$\varepsilon_m = \sum_{j=m+1}^{\infty} \left\langle \boldsymbol{K}(f)\, g_j, g_j\right\rangle \tag{4.1.2}$$

Exercise 4.1.1 Show that the covariance matrix $\boldsymbol{K}(f)$ is positive semidefinite. ◊

If the signal f is finitely supported, say on $[0, N]$, then there exists an orthonormal basis of eigenvectors of the symmetric and positive semidefinite *matrix* $\boldsymbol{K}(f) \in \mathbb{R}^{N+1\times N+1}$. Any such basis[2] is called a **Karhunen-Loève basis** and for each element of this basis we have that

$$\left\langle \boldsymbol{K}(f)\, g_j, g_j \right\rangle = \left\langle \lambda_j\, g_j, g_j \right\rangle = \lambda_j \underbrace{\left\| g_j \right\|^2}_{=1} = \lambda_j,$$

hence the error depends only of the associated eigenvalue.

Theorem 4.1.2. *If f is supported on $[0, N]$, then a basis $G = \left\{ g_j \;:\; j = 0, \ldots, N \right\} \subset \ell_2(\mathbb{Z})$, minimizes the error ε_m if and only if G is a Karhunen-Loève basis with*

$$\left\langle \boldsymbol{K}(f) g_j, g_j \right\rangle \geq \left\langle \boldsymbol{K}(f) g_{j+1}, g_{j+1} \right\rangle. \tag{4.1.3}$$

Definition 4.1.3. The eigenvalues for the largest eigenvalues of $\boldsymbol{K}(f)$ are called the **principal components** of f.

Proof: For any orthononormal set $h_j \in \ell(\mathbb{Z}_{N+1})$, $j = 0, \ldots, N$, we define the orthogonal matrix $\boldsymbol{H} = \left(f_j(k); j, k = 0, \ldots, n \right)$ and note that

$$\begin{aligned} \sum_{j=0}^{N} \lambda_j &= \operatorname{trace} \boldsymbol{K}(f) = \operatorname{trace} \left(\boldsymbol{H}^T \boldsymbol{K}(f) \boldsymbol{H} \right) \\ &= \sum_{j=0}^{N} \left(\begin{pmatrix} h_0^T \\ \vdots \\ h_N^T \end{pmatrix} \boldsymbol{K}(f) \left(h_1, \ldots, h_N \right) \right)_{jj} = \sum_{j=0}^{N} \left\langle \boldsymbol{K}(f)\, h_j, h_j \right\rangle, \end{aligned}$$

so that a basis minimizes ε_m if and only if it *maximizes*

$$\sum_{j=0}^{N} \lambda_j - \varepsilon_m = \sum_{j=0}^{N} \left\langle \boldsymbol{K}(f)\, h_j, h_j \right\rangle - \underbrace{\sum_{j=m+1}^{N} \left\langle \boldsymbol{K}(f)\, h_j, h_j \right\rangle}_{=\varepsilon_m} = \sum_{j=0}^{m} \left\langle \boldsymbol{K}(f)\, h_j, h_j \right\rangle.$$

If G is a Karhunen-Loève basis and H another orthonormal basis that we can write in terms of G as

$$h_j = \sum_{k=1}^{N} \left\langle h_j, g_k \right\rangle g_k,$$

then

$$\begin{aligned} \left\langle \boldsymbol{K}(f)\, h_j, h_j \right\rangle &= \sum_{k,\ell=1}^{N} \left\langle h_j, g_k \right\rangle \left\langle h_j, g_\ell \right\rangle \left\langle \boldsymbol{K}(f)\, g_k, g_\ell \right\rangle = \sum_{k,\ell=0}^{N} \left\langle h_j, g_k \right\rangle \left\langle h_j, g_\ell \right\rangle \lambda_k \left\langle g_k, g_\ell \right\rangle \\ &= \sum_{k=0}^{N} \left\langle h_j, g_k \right\rangle^2 \lambda_k, \end{aligned}$$

[2] For uniqueness, the eigenvalues would all have to be different.

and also

$$\sum_{j=0}^{m} \left\langle \boldsymbol{K}(f)\, h_j, h_j \right\rangle \;=\; \sum_{j=0}^{m} \sum_{k=0}^{N} \left\langle h_j, g_k \right\rangle^2 \lambda_k = \sum_{k=0}^{N} \underbrace{\left(\sum_{j=0}^{m} \left\langle h_j, g_k \right\rangle^2 \right)}_{=:q_k \le 1} \lambda_k,$$

hence,

$$\sum_{k=0}^{N} q_k \;=\; \sum_{k=0}^{N} \sum_{j=0}^{m} \left\langle h_j, g_k \right\rangle^2 = \sum_{j=0}^{m} \underbrace{\sum_{k=0}^{N} \left\langle h_j, g_k \right\rangle^2}_{=\|h_j\|^2=1} = m+1.$$

Consequently,

$$\begin{aligned}
&\sum_{j=0}^{m} \left\langle \boldsymbol{K}(f)\, h_j, h_j \right\rangle - \sum_{j=0}^{m} \left\langle \boldsymbol{K}(f)\, g_j, g_j \right\rangle = \sum_{j=0}^{m} \left\langle \boldsymbol{K}(f)\, h_j, h_j \right\rangle - \sum_{j=0}^{m} \lambda_j \\
&= \sum_{k=0}^{N} q_k\, \lambda_k - \sum_{j=0}^{m} \lambda_j = \sum_{k=0}^{N} q_k\, \lambda_k - \sum_{j=0}^{m} \lambda_j + \lambda_m \underbrace{\left(m+1 - \sum_{k=0}^{N} q_k \right)}_{=0} \\
&= \sum_{j=0}^{m} \left(q_j - 1\right) \lambda_j + \sum_{j=m+1}^{N} \lambda_j\, q_j + \lambda_m \sum_{j=0}^{m} \left(1 - q_j\right) - \lambda_m \sum_{j=m+1}^{N} q_j \\
&= \sum_{j=0}^{m} \underbrace{\left(\lambda_j - \lambda_m\right)}_{\ge 0} \underbrace{\left(q_j - 1\right)}_{\le 0} + \sum_{j=m+1}^{N} q_j \underbrace{\left(\lambda_j - \lambda_m\right)}_{\le 0} \le 0
\end{aligned}$$

with equality if and only if[3] $N = m$ und $q_j = 1$, $j = 1, \ldots, N$. In other words: the Karhunen-Loève basis is always better than other orthogonal bases and the best result is obtained if the basis is ordered such that $\lambda_1 \ge \lambda_2 \ge \cdots \ge \lambda_N$. □

We cannot consider statistical image processing in depth here. However, a good Karhunen-Loève basis can build a bridge and even the DCT can be interpreted in this sense and allows for an interpretation of JPEG compression in terms of optimal decompositions. However, this is beyond the scope of this lecture.

[3] At least in the case that $\boldsymbol{K}(f)$ is *strictly* positive definite, but eigenvectors for the zero eigenvalue play no role in information retrieval, so this is no restriction.

References 5

A Literatur

Akhieser, N. I. (1988). *Lectures on Integral Transforms*, volume 70 of *Translations of Mathematical Monographs*. AMS.

Barbour, J. M. (1951). *Tuning and Temperament. A Historical Survey*. Michigan State Press. Dover reprint 2004.

Benson, D. J. (2007). *Music. A Mathematical Offering*. Cambridge University Press.

Björck, A. (1996). *Numerical Methods for Least Squares Problems*. SIAM.

Chambolle, A., Caselles, V., Cremers, D., Novaga, M., and Pock, T. (2010). An introduction to total variationfor Image Analysis. In Fournasier, M., editor, *Theoretical Foundations and Numerical Methods for Sparse Recovery*, volume 9 of *Radon Series Comp. Appl. Math*, pages 263–340. De Gruyter.

Cooley, J. W. (1987). The re–discovery of the Fast Fourier Transform. *Mikrochimica Acta*, 3:33–45.

Cooley, J. W. (1990). How the FFT gained acceptance. In Nash, S. G., editor, *A History of Scientific Computing*, pages 133–140. ACM–Press and Addison–Wesley.

Cooley, J. W. and Tukey, J. W. (1965). An algorithm for machine calculation of complex Fourier series. *Math. Comp.*, 19:297–301.

Craven, P. and Wahba, G. (1979). Smoothing noisy data with spline functions: estimating the correct degree of smoothing by the method of generalized cross-validation. *Numer. Math.*, 31:377–403.

Daubechies, I. (1988). Orthonormal bases of compactly supported wavelets. *Commun. on Pure and Appl. Math.*, 41:909–996.

Daubechies, I. (1992). *Ten Lectures on Wavelets*, volume 61 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. SIAM.

DeVore, R. A. and Lorentz, G. G. (1993). *Constructive Approximation*, volume 303 of *Grundlehren der mathematischen Wissenschaften*. Springer.

Domes, J. (2007). Schnelle inverse Wavelettransformation. Zulassungsarbeit zum ersten Staatsexamen, Justus–Liebig–Universität Gießen.

FFTW (2003). `FFTW` – the Fastest Fourier Transform in the West. `http://www.fftw.org`.

Foley, J., van Dam, A., Feiner, S., and Hughes, J. (1990). *Computer Graphics*. Addison Wesley, 2nd edition.

Forster, O. (1984). *Analysis 3. Integralrechung im $\mathbb{R}^n$ mit Anwendungen.* Vieweg, 3. edition.

Gabor, D. (1946). Theory of communication. *J. IEEE*, 93:429–457.

Gathen, J. v. z. and Gerhard, J. (1999). *Modern Computer Algebra.* Cambridge University Press.

Gautschi, W. (1997). *Numerical Analysis. An Introduction.* Birkhäuser.

Gelbaum, B. R. and Olmstedt, J. M. H. (1964). *Counterexamples in Analysis.* Holden–Day. Dover reprint 2003.

Golub, G., Heath, M., and Wahba, G. (1979). Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21:215–223.

Golub, G. and van Loan, C. F. (1996). *Matrix Computations.* The Johns Hopkins University Press, 3rd edition.

Grossmann, A., Morlet, J., and Paul, T. (1985). Transforms associated to square integrable group representations. i.general results. *J. Math. Phys.*, 26:2473–2479.

Grüningen, D. C. v. (1993). *Digitale Signalverarbeitung.* VDE Verlag, AT Verlag.

Hamming, R. W. (1989). *Digital Filters.* Prentice–Hall. Republished by Dover Publications, 1998.

Handels, H. (2000). *Medizinische Bildverarbeitung.* B. G. Teubner.

Hardy, G. H. and Rogosinsky, W. W. (1956). *Fourier Series.* Cambridge University Press, 3. edition. Republished by Dover Publications, 1999.

Heuser, H. (1983). *Lehrbuch der Analysis. Teil 2.* B. G. Teubner, 2. edition.

Holschneider, M. (1995). *Wavelets: an analysis tool.* Clarendon Press, Oxford.

Horn, R. A. and Johnson, C. R. (1985). *Matrix Analysis.* Cambridge University Press.

Hough, P. V. C. (1962). Method and means for recognizing complex patterns. US Patent 3069654.

Hubbard, B. B. (1996). *The world according to wavelets.* A.K. Peters.

Isaacson, E. and Keller, H. B. (1966). *Analysis of Numerical Methods.* John Wiley & Sons.

Jähne, B. (2002). *Digitale Bildverarbeitung.* Springer.

Kammeyer, K. D. and Kroschel, K. (1998). *Digitale Signalverarbeitung.* Teubner Studienbücher Elektrotechnik. B. G. Teubner, Stuttgart.

Katznelson, Y. (1976). *An Introduction to Harmonic Analysis.* Dover Books on advanced Mathematics. Dover Publications, 2. edition.

Klein, A. (2011). *Zur Numerik kontinuierlicher Wavelet- und Matrixwavelet–Transformationen.* PhD thesis, Justus–Liebig–Universität Gießen.

Kotelnikov (1933). On the carrying capacity of the "ether" and wire in telecommunications. In *First All Union Conference of Communications I, zd. Red. Upr. Svyazi RKKA, Moscov.* In Russian.

Lorentz, G. G. (1966). *Approximation of Functions.* Chelsea Publishing Company.

Louis, A. K., Maaß, P., and Rieder, A. (1998). *Wavelets.* B. G. Teubner, 2. edition.

Mallat, S. (1989). Multiresolution approximations and wavelet orthonormal bases of $L^2(\mathbb{R})$. *Trans. Amer. Math. Soc.*, 315:69–87.

Mallat, S. (1999). *A Wavelet Tour of Signal Processing.* Academic Press, 2. edition.

Mallat, S. (2009). *A Wavelet Tour of Signal Processing: The Sparse Way.* Academic Press, 3rd edition.

Mhaskar, H. M. and Pai, D. V. (2000). *Fundamentals of Approximation Theory.* Narosa Publishing House.

Natterer, F. (1986). *The Mathematics of Computerized Tomography.* John Wiley & Sons.

Natterer, F. and Wübbeling, F. (2001). *Mathematical Methods in Image Reconstruction.* SIAM.

Nocedal, J. and Wright, S. J. (1999). *Numerical Optimization.* Springer Series in Operations Research. Springer.

Olafsson, G. and Quinto, E. T., editors (2006). *The Radon Transform, Inverse Problems, and Tomography*, volume 65 of *Proceedings of Symposia in Applied Mathematics.* AMS.

Osher, S., Burger, M., Goldfarb, D., Xu, J., and Yin, W. (2004). An iterative regularization method for total variation based image restoration. Technical report, UCLA.

Paul, R. P. (1981). *Robot Manipulators.* MIT Press.

Peebles, P. Z. (1980). *Probability, Random Variables and Random Signal Principles.* McGraw–Hill.

Sauer, T. (2000a). Numerische Mathematik I. Vorlesungsskript, Friedrich–Alexander–Universität Erlangen–Nürnberg, Justus–Liebig–Universität Gießen. `http://www.math.uni-giessen.de/tomas.sauer`.

Sauer, T. (2000b). Numerische Mathematik II. Vorlesungsskript, Friedrich–Alexander–Universität Erlangen–Nürnberg, Justus–Liebig–Universität Gießen. `http://www.math.uni-giessen.de/tomas.sauer`.

Sauer, T. (2002). Approximationstheorie. Vorlesungsskript, Justus–Liebig–Universität Gießen. Online verfügbar, Lehrstuhlseite.

Sauer, T. (2007). Splinekurven und –flächen in Theorie und Anwendung. Vorlesungsskript, Friedrich–Alexander–Universität Erlangen–Nürnberg, Justus–Liebig–Universität Gießen. `http://www.math.uni-giessen.de/tomas.sauer`.

Sauer, T. (2008). Integraltransformationen. Vorlesungsskript, Justus–Liebig–Universität Gießen. `http://www.math.uni-giessen.de/tomas.sauer`.

Sauer, T. (2011). Time–frequency analysis, wavelets and why things (can) go wrong. *Human Cognitive Neurophysiology*, 4:38–64.

Sauer, T. (2013a). Einführung in die Numerische Mathematik. Vorlesungsskript, Universität Passau.

Sauer, T. (2013b). Optimierung. Vorlesungsskript, Universität Passau.

Sauer, T. (2014). Analysis 1. Vorlesungsskript, Universität Passau.

Sauer, T. (2017). Constructive Approximation. Lecture notes, University of Passau.

Sauer, T. (2018). Advanced Imaging. Lecture notes, University of Passau.

Schoenberg, I. J. (1973). *Cardinal Spline Interpolation*, volume 12 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. SIAM.

Schönhage, A. and Strassen, V. (1971). Schnelle Multiplikation großer Zahlen. *Computing*, 7:281–292.

Schüßler, H. W. (1992). *Digitale Signalverarbeitung*. Springer, 3. edition.

Shannon, C. E. (1949). Communications in the presence of noise. *Proc. of the IRE*, 37:10–21.

Spellucci, P. (1993). *Numerische Verfahren der nichtlinearen Optimierung*. Internationale Schriftenreihe zu Numerischen Mathematik. Birkhäuser.

Steger, A. (2001). *Diskrete Strukturen 1. Kombinatorik – Graphentheorie – Algebra*. Springer.

Tolstov, G. P. (1962). *Fourier Series*. Prentice–Hall. Republished by Dover Publications, 1972.

Vetterli, M. and Kovačević, J. (1995). *Wavelets and Subband Coding*. Prentice Hall.

Whittaker, J. (1935). *Interpolatory function theory*, volume 33 of *Cambridge Tracts in Math. and Math. Physics*.

Yosida, K. (1965). *Functional Analysis*. Grundlehren der mathematischen Wissenschaften. Springer–Verlag.

Index 5