

2 Theory

The following Chapter is an overview of the most important physical and mathematical backgrounds that are required in this thesis to treat the topics of OCT, digital holography and holoscopy. It also conveys the formalisms that will be used in the following Chapters. More detailed treatments of the topics presented in this Chapter are found in [48–54].

2.1 The Fourier transform

The Fourier transform plays a very significant and important role in the complete thesis, therefore a summary of its definition and its most important properties is given below. For a more complete overview see e.g. [48, 49, 55].

A complex valued function U of a real position x can also be represented as a related function \tilde{U} of the conjugated spatial frequency k , the latter being called the frequency domain or Fourier-domain representation. Both representations contain all information of the function, but it will be shown throughout this thesis, that physical laws and dependencies take a simpler form using the appropriate one. The Fourier transform $\mathcal{F}[\cdot]$ is the operator relating the functions U and \tilde{U} , defined by

$$\tilde{U}(k) = \mathcal{F}[U(x)] = \int dx U(x) e^{-ikx}. \quad (2.1.1)$$

This transform can be interpreted as a development into orthogonal harmonic functions, namely sines and cosines, and it is a linear operator. The inverse operator

$\mathcal{F}^{-1}[\cdot]$ takes the function \tilde{U} back from frequency space to position space

$$U(x) = \mathcal{F}^{-1}[\tilde{U}(k)] = \frac{1}{2\pi} \int dk \tilde{U}(k) e^{+ikz}.$$

The concept of the Fourier transform and the frequency space representation can easily be generalized to three-dimensional physical fields U , that are a function of position $\mathbf{x} = (x, y, z)$

$$\tilde{U}(\mathbf{k}) = \mathcal{F}[U(\mathbf{x})] = \int d^3x U(\mathbf{x}) e^{-i\mathbf{k} \cdot \mathbf{x}},$$

where $\mathbf{k} = (k_x, k_y, k_z)$ describes the spatial frequencies. A vector dependency of a field can also be described in frequency space along one or two axes, while along the remaining axis it is described in position space, for example by $\tilde{U}_z(x, y, k_z)$ or $\tilde{U}_{xy}(k_x, k_y, z)$. The according generalizations to the Fourier transform operators are straight forward by sequential one-dimensional transforms for each axis.

A list of the most important properties of the Fourier transform along with a list of the transforms of important functions is found in the Appendix A.3.

2.1.1 The Hilbert transform and analytic signals

A complex valued function $U(z)$ is said to be an analytic signal, if its Fourier transform $\tilde{U}(k)$ fulfills the property

$$\tilde{U}(k) = 0, \quad \text{for all } k \leq 0.$$

Assuming a measurement only acquires the real part of a complex valued function $U(z)$, i.e. $\text{Re } U(z)$, the function $U(z)$ can in general not be obtained from the measured data. It follows by using

$$\text{Re } U(z) = \frac{1}{2} [U(z) + U^*(z)],$$

that

$$\begin{aligned} \mathcal{F}[\text{Re } U(z)] &= \frac{1}{2} [\mathcal{F}[U(z)] + \mathcal{F}[U^*(z)]] \\ &= \frac{1}{2} (\tilde{U}(k) + \tilde{U}(-k)), \end{aligned}$$

i.e. the Fourier transform of the acquired data is symmetric with respect to the origin. If the signal $U(z)$ is known to be analytic, it can be obtained by filtering the

negative frequency components and increasing the positive frequency components by a factor two:

$$\tilde{U}(k) = 2\Theta(k)\mathcal{F}[\operatorname{Re} U(z)],$$

where $\Theta(k)$ is the step function (A.1.6). By using $2\Theta(k) = 1 + \operatorname{sgn}(k)$, with $\operatorname{sgn}(\cdot)$ being given by (A.1.7), and writing the multiplication as convolution of their respective position space functions (see convolution theorem (A.3.1)), it follows

$$\tilde{U}(k) = \mathcal{F}[\operatorname{Re} U(z)] + \mathcal{F}\left[\mathcal{F}^{-1}[\operatorname{sgn}(k)] * \operatorname{Re} U(z)\right].$$

With $\mathcal{F}^{-1}[\operatorname{sgn}(k)] = i/(\pi z)$, the analytic signal $U(z)$ can be written in position space as

$$\begin{aligned} U(z) &= \operatorname{Re} U(z) + i \frac{1}{\pi} \left(\frac{1}{z} * \operatorname{Re} U(z) \right) \\ &= \operatorname{Re} U(z) + i \frac{1}{\pi} \int dz' \frac{1}{z - z'} \operatorname{Re} U(z'). \end{aligned}$$

The imaginary part motivates the definition of the Hilbert transform

$$\mathcal{H}[f(z)] = \frac{1}{\pi} \int dz' \frac{f(z')}{z - z'}.$$

The analytic signal of an acquired real signal can be obtained by setting the imaginary part of the signal equal to the Hilbert transform of the acquired real signal.

2.1.2 The discrete Fourier transform (DFT)

In numerical computations the integrals defining the Fourier transform can hardly be computed: signals are not acquired in the entire spatial or frequency domain and they are not obtained arbitrarily dense, i.e. its number of nodes is limited. A discretized version of the Fourier transform, known as the discrete Fourier transform (DFT), is used instead. It takes an N -dimensional complex vector u_n as input (position space vector) and maps it onto an N -dimensional complex vector $\tilde{u}_{n'}$ (frequency space vector). Being a linear transform it can be written as a matrix multiplication, which is defined by

$$\tilde{u}_{n'} = \sum_{n=0}^{N-1} F_{n'n} u_n, \quad \text{with } F_{n'n} = \exp\left(-i \frac{2\pi}{N} n'n\right), \quad (2.1.2)$$

where $F_{n'n}$ denotes the Fourier matrix. Its inverse is obtained by matrix inversion to

$$u_n = \sum_{n'=0}^{N-1} F_{n'n}^{-1} \tilde{u}_{n'}, \quad \text{with } F_{n'n}^{-1} = \frac{1}{N} \exp\left(+i \frac{2\pi}{N} n'n\right).$$

The generalization to a multi-dimensional discrete Fourier transforms is achieved by multilinear mappings using tensors. For example, the Fourier transform $\tilde{u}_{n'm'}$ of a two-dimensional field u_{nm} of size $N \times M$ is given by

$$\begin{aligned} \tilde{u}_{n'm'} &= \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} F_{n'nm'm} u_{nm}, \\ &\text{with } F_{n'nm'm} = F_{n'n} F_{m'm} = \exp\left(-i2\pi \left(\frac{n'n}{N} + \frac{m'm}{M}\right)\right), \end{aligned} \quad (2.1.3)$$

with $F_{n'nm'm}$ being called the two-dimensional Fourier tensor, which is obtained as tensor product of the two one-dimensional Fourier matrices. Its effect is equivalent to transforming each axis individually.

2.1.2.1 Symmetries of the DFT

By extending the scope of the Fourier matrix indices to all integer numbers, symmetries arise:

$$F_{n'n} = F_{(n'+N)n} = F_{n'(n+N)}. \quad (2.1.4)$$

These symmetries often cause confusion in signal processing, as it allows for an alternative definition of the DFT:

$$\begin{aligned} \tilde{u}_{n'} &= \sum_{n=0}^{N/2-1} F_{n'n} u_n + \sum_{n=N/2}^{N-1} F_{n'n} u_n \\ &= \sum_{n=0}^{N/2-1} F_{n'n} u_n + \sum_{n=N/2-N}^{N-1-N} F_{n'n} u_n \\ &= \sum_{n=-N/2}^{N/2-1} F_{n'n} u_n \end{aligned}$$

With this definition, indices of the input and output vectors run from $-N/2$ to $N/2 - 1$. The negative indices are commonly represented as negative frequencies, especially when displaying data in frequency space. The DC ($n' = 0$) component

is centralized in this representation. For most programming languages, that only support indices running from 0 to $N - 1$, frequencies need to be shifted.

2.1.2.2 Relation to the analytic Fourier transform

Let the signal $U(x)$ be known at equidistant points $x = n \cdot \Delta x$, meaning the specific values $u_n = U(n \cdot \Delta x)$ are given for $n = 0 \dots N - 1$. Sampling the function $U(x)$ at these discrete points (see Section 2.2), represented by

$$U(x) = \sum_{n=0}^{N-1} u_n \delta(x - n \cdot \Delta x)$$

and inserting this into the one-dimensional Fourier transform (2.1.1) gives

$$\begin{aligned} \tilde{U}(k) &= \int dz e^{-ikz} \sum_{n=0}^{N-1} u_n \delta(x - n \cdot \Delta x) \\ &= \sum_{n=0}^{N-1} u_n \int dz e^{-ikz} \delta(x - n \Delta x) \\ &= \sum_{n=0}^{N-1} u_n e^{-ikn\Delta x}. \end{aligned}$$

Assuming the outgoing signal is equispaced in the Fourier-domain with spacing Δk , given by $\tilde{u}_{n'} = \tilde{U}(n' \cdot \Delta k)$, one obtains the DFT

$$\tilde{u}_{n'} = \sum_{n=0}^{N-1} e^{-in'n\Delta k\Delta x} u_n. \quad (2.1.5)$$

Comparing (2.1.5) with (2.1.2) gives an important relation between sampling distance of input and output data, when computing the DFT on sample data:

$$\Delta k = \frac{2\pi}{N\Delta x} \quad (2.1.6)$$

If $u_n = 0$ for $n \neq 0, \dots, N - 1$, the DFT of u_n can be computed by the analytic Fourier transform $\tilde{U}(k)$ via

$$\tilde{u}_{n'} = \tilde{U}(n'\Delta k) = \tilde{U}\left(n' \frac{2\pi}{N\Delta x}\right) = \sum_{n=-\infty}^{+\infty} e^{-\frac{2\pi}{N}n'n} u_n = \sum_{n=0}^{N-1} e^{-\frac{2\pi}{N}n'n} u_n.$$

The proof follows from (A.3.8), the Fourier transform of the comb-distribution is again a comb-distribution.

2.1.2.3 The fast Fourier transform (FFT)

When computing the DFT of a given vector, the matrix elements $F_{n'n}$ can be pre-computed to get optimal performance, but complexity of the matrix multiplication remains of the order of $\mathcal{O}(N^2)$. In 1965, James W. Cooley and John Tukey [56] showed a simple way to split the computation of one DFT in two, one of the even elements of the input vector and one of its odd elements. The DFT is split according to

$$\sum_{n=0}^{N-1} F_{n'n} u_n = \sum_{n=0}^{N/2} F_{n'(2n)} u_{(2n)} + \sum_{n=1}^{N/2-1} F_{n'(2n+1)} u_{(2n+1)}.$$

The matrix components can be decomposed to

$$\begin{aligned} F_{n'(2n)} &= \exp\left(-i \frac{2\pi}{N} n'(2n)\right) = \exp\left(-i \frac{2\pi}{N/2} n'n\right) = F_{n'n}^{(N/2)} \\ F_{n'(2n+1)} &= \exp\left(-i \frac{2\pi}{N} n'(2n+1)\right) \\ &= \exp\left(-i \frac{2\pi}{N} n'\right) \exp\left(-i \frac{2\pi}{N/2} n'n\right) \\ &= \exp\left(-i \frac{2\pi}{N} n'\right) F_{n'n}^{(N/2)}, \end{aligned}$$

where $F^{(N/2)}$ denotes the Fourier matrix of size $N/2$. This shows, that one matrix multiplication with $F_{n'n}$ can be replaced with two matrix multiplications with $F_{n'n}^{(N/2)}$. The latter scenario is computationally more efficient and the method can be recursively applied by splitting the matrix multiplication with $F_{n'n}^{(N/2)}$ again in half. Effectively, this results in a reduced complexity of about $\mathcal{O}(N \cdot \log N)$.

In practice, the fast Fourier transforms (FFT) can reduce computation time by several orders of magnitude, compared to the standard DFT. For N that are not a power of two or even prime, suitable algorithms have been demonstrated as well (see e.g. [57]).

2.1.3 Fourier transforms on non-equispaced nodes

Evaluation of FD-OCT data involves an (inverse) Fourier transform of data which were not sampled on an equidistant scale. Instead of $\tilde{U}(k)$ only a related signal $\tilde{U}(t)$ is directly available with t being a monotone function of k . $U(z)$, the Fourier transform of $\tilde{U}(k)$, can be written as an integral transform of $\tilde{U}(t)$ by applying a suitable variable substitution

$$\begin{aligned} U(z) &= \int dk \tilde{U}(k(t)) \exp(-ikz) \\ &= \int dt \frac{dk(t)}{dt} \tilde{U}(k(t)) \exp(-ik(t)z). \end{aligned} \quad (2.1.7)$$

This integral transform is referred to as Fourier transform on non-equispaced data. To perform it, the relation between t and k needs to be known. The inverse Fourier transform on non-equispaced data can be defined accordingly.

2.1.3.1 The non-equispaced DFT (NDFT)

The discretization of (2.1.7) is straight forward. Introducing the signals $\tilde{u}_{n'}^{(k)}$ and $\tilde{u}_{v'}$, discretizations of \tilde{U} equispaced in k and equispaced in t , are given by

$$\tilde{u}_{n'}^{(k)} = \tilde{U}(k_0 + n' \Delta k) \quad \text{and} \quad \tilde{u}_{v'} = \tilde{U}(k(t_0 + v' \Delta t)),$$

respectively. The superscript (k) indicates that this data is linearly in k . A relation between their indices is readily obtained to

$$n'(v') = \frac{k(t_0 + v' \Delta t) - k_0}{\Delta k}.$$

The non-equispaced discrete Fourier transform is adapted from (2.1.2) to

$$u_n = \sum_{v'=0}^{N-1} D_{v'n} \tilde{u}_{v'}, \quad D_{v'n} = \exp\left(-i \frac{2\pi}{N} n'(v') n\right), \quad (2.1.8)$$

where the derivative $dk(t)/dt$ has been neglected. u_n is thus computed by the non-equispaced discrete Fourier transform (NDFT), a multiplication with the non-equispaced Fourier matrix $D_{v'n}$. However, the FFT algorithm requires the equispacing of the input data in n' -space. A fast version of (2.1.8) can therefore not be created and its complexity remains $\mathcal{O}(N^2)$.

In analogy to a standard discrete Fourier transform being a development into orthogonal functions, sines and cosines, the NFFT can be interpreted as the development into sines and cosines, which are chirped by $k(t)$.

2.1.3.2 Interpolation and FFT (iFFT)

In order to improve computation time of an NDFT, the data $\tilde{u}_{v'}$ can be resampled to obtain an approximation of $\tilde{u}_{n'}^{(k)}$ prior to a standard DFT. The resampling step can, for example, be done by linear interpolation. In order to increase the accuracy of the approximation, oversampling by a factor α can be applied by computing the values $\tilde{u}_{n'}^{(k)}$ with their index n running proportional to the wavenumber k in the range 0 to $\alpha N - 1$. The interpolated signal is given by

$$\begin{aligned}
 \tilde{u}_{n'}^{(k)} &\approx \left(\tilde{u}_{\lceil v'(n'/\alpha) \rceil} - \tilde{u}_{\lfloor v'(n'/\alpha) \rfloor} \right) \\
 &\quad \times \left(v' \left(\frac{n'}{\alpha} \right) - \left\lfloor v' \left(\frac{n'}{\alpha} \right) \right\rfloor \right) + \tilde{u}_{\lfloor v'(n'/\alpha) \rfloor} \\
 &= \tilde{u}_{\lceil v(n'/\alpha) \rceil} \underbrace{\left(v \left(\frac{n'}{\alpha} \right) - \left\lfloor v \left(\frac{n'}{\alpha} \right) \right\rfloor \right)}_{\text{weight right point}} \\
 &\quad + \tilde{u}_{\lfloor v(n'/\alpha) \rfloor} \underbrace{\left(1 - \left(v \left(\frac{n'}{\alpha} \right) - \left\lfloor v \left(\frac{n'}{\alpha} \right) \right\rfloor \right) \right)}_{\text{weight left point}}, \\
 &\quad \text{for all } n' \in [0; \alpha N - 1),
 \end{aligned} \tag{2.1.9}$$

where $\lceil \cdot \rceil$ and $\lfloor \cdot \rfloor$ denote the rounding up and down operations, respectively. The N input values $\tilde{u}_{v'}$ are used to create αN values $\tilde{u}_{n'}^{(k)}$, i.e. the length of the subsequent FFT is therefore increased by a factor α . Therefore the time complexity calculating the FFT increases to $\mathcal{O}(\alpha N \cdot \log \alpha N)$. The interpolation (2.1.9) can be rewritten by using $\lceil a \rceil - \lfloor a \rfloor = 1$, $v' - \lceil v' \rceil \leq 0$ and $v' - \lfloor v' \rfloor \geq 0$ as

$$\begin{aligned}
 \tilde{u}_{n'}^{(k)} &\approx \left(1 - \left| v' \left(\frac{n'}{\alpha} \right) - \left\lfloor v' \left(\frac{n'}{\alpha} \right) \right\rfloor \right| \right) \tilde{u}_{\lceil v(n'/\alpha) \rceil} \\
 &\quad + \left(1 - \left| v \left(\frac{n'}{\alpha} \right) - \left\lfloor v \left(\frac{n'}{\alpha} \right) \right\rfloor \right| \right) \tilde{u}_{\lfloor v(n'/\alpha) \rfloor}.
 \end{aligned}$$

By using the triangle-function Δ (A.1.3), it follows

$$\begin{aligned}
 \tilde{u}_{n'}^{(k)} &\approx \Delta\left(v'\left(\frac{n'}{\alpha}\right) - \left\lfloor v'\left(\frac{n'}{\alpha}\right) \right\rfloor\right) \tilde{u}_{\lfloor v'(n'/\alpha) \rfloor} \\
 &\quad + \Delta\left(v'\left(\frac{n'}{\alpha}\right) - \left\lceil v'\left(\frac{n'}{\alpha}\right) \right\rceil\right) \tilde{u}_{\lceil v'(n'/\alpha) \rceil} \\
 &= \sum_{j=\lfloor v'(n'/\alpha) \rfloor}^{\lceil v'(n'/\alpha) \rceil} \Delta\left(v'\left(\frac{n'}{\alpha}\right) - j\right) \tilde{u}_j = \sum_{j=0}^{N-1} \Delta\left(v'\left(\frac{n'}{\alpha}\right) - j\right) \tilde{u}_j,
 \end{aligned}$$

where the last equality indicates, that the triangle function is zero for all data points, except $j = \lfloor v'(n'/\alpha) \rfloor$ and $j = \lceil v'(n'/\alpha) \rceil$. According to this relation, the linear interpolation performs a discrete convolution with the triangle function Δ (Figure 2.1.1a), resulting in an attenuation of the higher frequencies of the signal, which can be removed after the actual FFT by dividing the results by the Fourier transform of the triangle function. According to Section 2.1.2.2 the discrete Fourier transform of the triangle function can be evaluated by computing the analytic continuation to real n' , denoted by $\Delta(k/\alpha)$. The Fourier transform is

$$\begin{aligned}
 \tilde{\Delta}(z) &= \int dk \Delta\left(\frac{k}{\alpha}\right) e^{-ikz} = \int_{-\alpha}^{+\alpha} dk \left(1 - \left|\frac{k}{\alpha}\right|\right) e^{-ikz} \\
 &= \alpha \frac{\sin^2\left(\frac{z\alpha}{2}\right)}{\left(\frac{z\alpha}{2}\right)^2}.
 \end{aligned}$$

This gives the required deconvolution filter

$$\tilde{\Delta}_n = \tilde{\Delta}\left(n \frac{2\pi}{N}\right) = \alpha \frac{\sin^2\left(\frac{\pi \alpha n}{N}\right)}{\left(\frac{\pi \alpha n}{N}\right)^2} = \alpha \cdot \text{sinc}^2\left(\frac{\alpha n}{N}\right),$$

which reverts the fall-off of the high frequencies.

The linear interpolation and the following FFT have approximately a time complexity of $\sim \mathcal{O}(\alpha N)$ and $\sim \mathcal{O}(\alpha N \cdot \log \alpha N)$, respectively. For optimal performance the weighting factors used for the linear interpolation and the values used for the deconvolution after the FFT can be precomputed. This algorithm will be referred to as iFFT α .

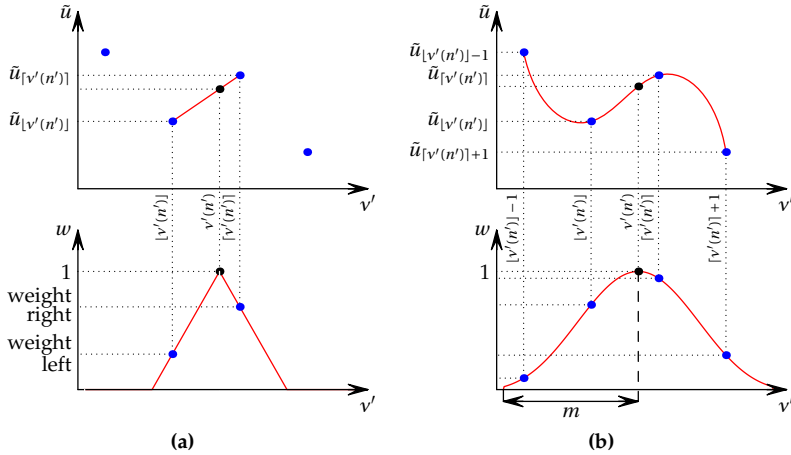


Figure 2.1.1.: Both, the interpolated FFT (a) and the non-equispaced FFT (b), perform a discrete convolution prior to the FFT to achieve interpolation. While the iFFT uses a triangle function as convolution function, the NFFT commonly uses a Kaiser-Bessel window.

2.1.3.3 The non-equispaced FFT (NFFT)

Instead of convolving with a triangle function, which involves only two original data points, an extended window function can be used which is cut off farther away from the data point that needs to be approximated (see Figure 2.1.1b) [58]. This approach is referred to as the non-equispaced fast Fourier transform (NFFT). The cut-off is determined by a parameter m and therefore the algorithm has, in addition to α , another parameter influencing accuracy and speed of the algorithm. Computation is faster if oversampling α and cut-off m are chosen smaller, but accuracy suffers and increased artifacts are observed. The discrete convolution has a complexity of about $\sim \mathcal{O}(m\alpha N)$ and therefore the complete time complexity is approximately $\sim \mathcal{O}(m\alpha N + \alpha N \cdot \log \alpha N)$.

It has been shown, that in most cases a Kaiser-Bessel window (A.1.9) yields good results [59,60] for pre-FFT interpolation. If the window function values for the data points are precomputed completely, the requirement for computing the window function does in practice not increase the time needed for the calculation of the NFFT. This algorithm will be referred to as NFFT α, m .

Holoscopy

Hillmann, D.

2014, XXXVI, 206 p. 58 illus., 1 illus. in color., Softcover

ISBN: 978-3-658-06478-5