DIPLOMARBEIT

# Developing a Fully Scanner-integrated Reconstruction Pipeline for Magnetic Resonance Spectroscopic Imaging (MRSI) at 7 T

zur Erlangung des akademischen Grades

**Diplom-Ingenieur**

im Rahmen des Studiums

**Biomedical Engineering: Medical Physics and Imaging**

eingereicht von

**B.Sc. Jan Luxemburg**
Matrikelnummer 11936026

ausgeführt am Atominstitut der Technischen Universität Wien
und Exzellenzzentrum für Hochfeld-Magnetresonanz der Medizinischen Universität Wien

Betreuung:
Em.Univ.-Prof. Dipl.-Ing. Dr.techn. Gerald Badurek
Assoc.-Prof. Dipl.-Ing. Wolfgang Bogner, PhD

Wien, 07.06.2022

(Verfasser) (Betreuer)

# Deutsche Kurzfassung

Zur Unterstützung der Forschungsgruppe im Hochfeld-MR-Zentrum in Wien am AKH und der Medizinischen Universität wurde eine vollständig Scanner-integrierte Rekonstruktionspipeline für eine schnelle Ganzhirn-magnetresonanzspektroskopische Bildgebung (MRSI)-Sequenz entwickelt. Der Entwicklungsprozess wurde durch bereitgestellte Offline-Rekonstruktionspipelines geleitet, die als Vorlage dienten. Die Vorteile einer Online-Implementierung umfassen einen flüssigen Ablauf des Bildgebungsprozesses, eine höhere Zeiteffizienz und wären auch für die Übertragung in klinische Umgebungen relevant.

Drei unterschiedliche Aspekte der Datenverarbeitungsschritte wurden in separaten Abschnitten dieser Arbeit behandelt. Erstens, die konzeptionellen Aspekte wurden im Abschnitt *Datenverarbeitungskonzepte in MRSI (Data-processing concepts in MRSI)* beschrieben. Zweitens wurden die mathematischen Überlegungen zur durchgeführten Rekonstruktion im Abschnitt über *Datenrekonstruktionsverfahren (Data-reconstruction procedures)* dargelegt. Drittens wurden die Details zur Programmierung in den Abschnitten *Implementierung der kartesischen Rekonstruktion (Implementation of the Cartesian reconstruction)* und *Implementierung der konzentrischen Ringtrajektorie (CRT)-Rekonstruktion (Implementation of the CRT reconstruction)* beschrieben.

Die resultierenden online, i.e., am Scanner, berechneten Bilder wurden dann im Abschnitt *Vergleich von Online- und Offline-Rekonstruktion (Comparison of online and offline reconstruction)* mit den Offline-Ergebnissen verglichen. Das Ergebnis dieser Diplomarbeit war eine vollautomatische Online-Rekonstruktionspipeline. Bilder, FIDs und Spektren der einzelnen Datenverarbeitungsschritte zeigten eine insgesamt gute Qualität mit einigem Verbesserungspotenzial. Die Qualität der Ergebnisse der *vollständigen Rekonstruktion (complete reconstruction)*, i.e., aller Datenverarbeitungsschritte zusammengenommen, muss unbedingt weiter gesteigert werden, da durch die Kombination der Datenoperationen auch deren Abweichungen vom Standard kombiniert wurden und eine akzeptable Schwelle überschritten haben.

# Abstract

To support the research group in the High-Field MR Centre in Vienna at the AKH and the Medical University, a fully scanner-integrated reconstruction pipeline for a fast whole-brain Magnetic Resonance Spectroscopic Imaging (MRSI) sequence was developed. The development process was guided by provided offline reconstruction pipelines, which served as a standard. The advantages of an online implementation comprise a smoother workflow of the imaging process, higher time efficiency, and would be also relevant for translation into clinical settings.

Three different aspects of the data-processing steps were discussed in separate Sections of this thesis. First, the conceptual aspects were described in the Section on *Data-processing concepts in MRSI*. Second, the mathematical considerations of the implemented reconstruction were stated in the Section on *Data-reconstruction procedures*. Third, the details about the programming were reviewed in the Sections on *Implementation of the Cartesian reconstruction* and *Implementation of the concentric ring trajectory (CRT) reconstruction*.

The resulting online, i.e., on the scanner, calculated images were then compared to the offline results in the Section on *Comparison of online and offline reconstruction*. The result of this thesis was a fully automatic online reconstruction pipeline. Images, FIDs, and spectra from the individual data-processing steps showed an overall good quality with some room for improvement. The quality of the outputs of the *complete reconstruction*, i.e., all the data-processing steps combined, definitely needs to be further increased, as by combination of the data operations, also their deviations from the standard combined, and exceeded an acceptable threshold.

# Contents

## Acknowledgments

I am deeply thankful to my supervisor, Mr. Prof. Gerald Badurek, who guided me through the whole course of my diploma studies with endless patience, advice, and support. It was always a pleasure to work with him and I learned a lot from him. He was an ever-growing source of inspiration for me.

I wish to express my thanks to the leader of the research group in the High-Field MR Centre in Vienna at the AKH and the Medical University, Mr. Prof. Wolfgang Bogner. His generous attitude in all affairs related to my diploma thesis, professional, as well as personal, stimulated me to a significant improvement in my scientific career.

I am also very thankful to my colleague and dear friend Mr. Bernhard Strasser, PhD. He was always there to go through all the technical details, regarding the reconstruction algorithms, with me. Mr. Fabian Niess, PhD helped me greatly at the very beginning of my work in the group. His introduction into the topic provided my with a stable foundation on which to built this thesis. Great thanks also belong to Mrs. Mary McAllister for her friendly approach and diligence, with which she supported me to improve the stylistic aspects of the thesis.

Last, but not least, my gratitude goes to my lovely girlfriend, Sara, my father, mother, sister, grandparents and the rest of the family and friends for their patience, love, and motivating me on this joyful journey.

# 1 Introduction

The *Introduction* Chapter is divided into five sections. In the first Section the *Motivation* for the whole thesis is stated. This is done by defining the starting point of the work, the overall goal, and the path taken to achieve the goal. Also, the benefits of the chosen solution are narrated.

In the second Section on *Physical background of Nuclear Magnetic Resonance*, the reader is introduced to the topic by explaining the relevant properties of atomic nuclei, and their interrelatedness, such as nuclear spin, magnetic moment, gyromagnetic ratio, and Larmor precession in a magnetic field. Based on that, the basic way to induce and record signals is described. Further considerations are categorized into, on the one hand, the *Classical picture* (i.e., macroscopic magnetization and its temporal evolution and relaxation processes), and on the other, the *Quantum-mechanical picture* (i.e., quantization of magnetic energy levels and their population, chemicals shift, and scalar coupling).

The creation of images from the signals is explained in the Section on *Basics of Magnetic Resonance Imaging* with the localization concepts of *slice selection*, *frequency encoding*, and *phase encoding*. Furthermore, the spectroscopic dimension, and localization of the spectra, is introduced, together with the most important metabolites. Besides the basic localization techniques, the advanced concept of *spatial-spectral encoding*, applied in the RF-pulse sequence utilized in this thesis, is demonstrated. Also, imaging and spectral artefacts, including their causes and elimination strategies, are accounted for.

In the fourth Section on *Acquisition features in MRSI*, the necessary elements during measurement are pointed out: *water* and *lipid suppression*, *motion correction*, and *scanner instability handling*. Those aspects can be either included during the acquisition process, or later in the data processing, as is illustrated in the next section.

Finally, in the fifth Section on *Data-processing concepts in MRSI*, all of the necessary, in-data operations are described. Those operations start with raw data and arrive at metabolic maps. Those include the, in this thesis implemented, *spatial-spectral reconstruction*, in-data *water signal handling*, *lipid suppression*, *motion correction*, *scanner instability handling*, and *spectral quantification*.

## 1.1 Motivation

As part of this diploma thesis, the reconstruction part of a fast whole-brain MRSI sequence (using *spatial-spectral* encoding on a *concentric ring trajectory*) was implemented directly on a 7 T scanner, using C++. The code has already been implemented in MATLAB in three different versions. Thus, the following versions were available at the beginning of this diploma thesis: a standard pipeline for patient measurements, an independent prototype pipeline from Boston, and an independent prototype pipeline from Vienna.

The best elements from the three versions were brought together and the code was transferred to C++ according to the Image Calculation Environment (ICE) from Siemens. This software solution had to be fast and reliable. The following data operations were implemented: *rearrangement of the data including temporal interleaves*; *spatial discrete Fourier transformation*; *field-of-view shift*; *frequency-offset correction*; and *density compensation*. Each of those data-processing steps was implemented in a single *functor*, i.e., the fundamental data-processing unit of ICE. After successful implementation of each step, the C++ reconstruction algorithms were tested experimentally and compared to a standard. The best implementation variant was selected based on the following quality criteria: reconstruction speed and image artefacts.

Each *functor* performed only a small, but, independent and complete, data-processing step. The modular structure of the developed program enabled to use the algorithms with a great degree of flexibility. Thus, the created solution does not demand a thorough knowledge on the side of the user, and can be used and worked on selectively. The reconstruction was implemented only for the 2D case. Later, it will be extended for the third dimension.

## 1.2 Physical background of Nuclear Magnetic Resonance

Nuclear Magnetic Resonance (NMR) is a tool that uses a strong external magnetic field to study the nuclear spin properties of various atoms and molecules [1]. It has been shown that $^1H$ atomic nuclei (i.e., protons) in a magnetic field can interact with it. To induce magnetic interactions, radio-frequency (RF) electro-magnetic pulses tuned to a specific frequency according to the field strength of the external field are used. After applying the pulse, different hydrogen nuclei of one molecule start to resonate at different frequencies depending on their chemical environment, thereby making it possible to identify different molecules. The ability of NMR to identify molecules is utilized in spectroscopy methods. Many other nuclei show the same behaviour and can be used as well for NMR resonance experiments [1].

NMR can be explained either by semi-classical physics, where the involved magnetic fields are considered as classical electromagnetic entities and quantum physics enters only via the quantization of energy and angular momenta, or by fully quantum-mechanical description. The semi-classical picture can not be applied to describe all of the NMR features, such as scalar coupling. Nevertheless, it is especially helpful to understand the basic concepts because of its intuitive descriptions.

As NMR is a nuclear phenomenon, only the nuclear properties of atoms will be considered and not those of the electronic shell, for now. All nuclei consist of protons and neutrons. These are commonly termed nucleons. The nucleons possess a spin angular momentum (i.e., nuclear spin or just spin here) as their intrinsic property, which derives from the spins of quarks comprising the nucleons. Both protons and neutrons have an intrinsic spin quantum number of $\frac{1}{2}$. However, within a nucleus the nucleons may have also an additional non-zero orbital angular momentum. Therefore the notation nuclear *spin*, $\vec{I}$. Two other quantum numbers are necessary to describe the components of $\vec{I}$. First, the absolute value of the nuclear spin, $\sqrt{I(I+1)}\hbar$, with $\hbar$ the reduced Planck's constant. Second, its component $I_z = m\hbar$ along the arbitrarily chosen quantization axis (here z-axis), where $m = -I, -I+1, ..., I-1$, and $I$ is called *azimuthal* or *magnetic* quantum number. The spin quantum number, $\vec{I}$, often named simply *nuclear spin*, depends on the nuclear composition of protons and neutrons, mainly, whether their numbers are even or odd. There is one special case in which the nuclei have no nuclear spin, i.e., $I = 0$. This is when the nucleus has an even number of protons and an even number of neutrons. In all other cases, the nuclei have a non-zero spin [1]. Semi-classical theory and quantum theory for the description of NMR both have in common that they are based on the nuclear spin, $\vec{I}$. A non-zero nuclear spin causes an according magnetic moment

$$\vec{\mu} = \gamma \vec{I} \tag{1.1}$$

where $\gamma$ is the so-called gyromagnetic ratio that is a specific constant for each type of nucleus. As the excitation pulses and encoding mechanisms in a scanner sequence are also nucleus-specific, in this thesis, the $^1H$ nucleus, a proton, is of the most importance because the used RF-pulse sequence (described in the *Materials and Methods* Section) is tuned to it. Together with some of the other important nuclei for NMR, the gyromagnetic ratios and other nuclear properties are depicted below:

| Nucleus | Spin | $\gamma$ in MHz/T | Natural abundance in % |
|---|---|---|---|
| $^1H$ | 1/2 | 42.576 | 99.99 |
| $^2H$ | 1 | 6.54 | 0.02 |
| $^{13}C$ | 1/2 | 10.71 | 1.07 |
| $^{23}Na$ | 3/2 | 11.26 | 100 |
| $^{31}P$ | 1/2 | 17.24 | 100 |

Table 1.1: Spin quantum number, gyromagnetic ratio and natural abundance of selected nuclei [1]

From Table 1.1, it can be seen that the gyromagnetic ratio of a proton is the highest. The natural abundance influences the signal intensity in NMR spectroscopy [2].

Due to the quantization of the nuclear spin, the magnetic energy, $E_m$, is split into discrete energy levels, as described by Eq. 1.2.

$$E_m = -\hbar\gamma m B_0 \tag{1.2}$$

In Eq. 1.2, $B_0$ stands for the field strength of the external magnetic field. Thus, the gyromagnetic ratio $\gamma$ is found in both semi-classical and quantum theoretical descriptions of NMR.

The difference between the two theories is that the classical theory describes NMR on a large scale using a macroscopic magnetization $\vec{M}$ (or net magnetization) which is built up by a vector addition of the individual nuclear magnetic moments $\vec{\mu}$:

$$\vec{M} = \sum_i \vec{\mu}_i \tag{1.3}$$

In contrast to classical theory, quantum theory deals with the phenomenon on an elementary scale, attributing the magnetic changes of the sample to the spin state of the nucleus and the transition between spin states (i.e., spin flip). The spin flip process displays the switching between discrete magnetic energy levels. When a spin transits from a lower to a higher energy level, an RF quantum of a frequency, $\nu$, is absorbed corresponding to the energy difference between the two levels. In the relaxation process, the opposite occurs, as the nuclear spin transits back to the lower energy level by emitting a photon of the same frequency $\nu$. The equation for calculating the frequency of the absorbed or the emitted quantum of the electro-magnetic field is derived in the Subsection on *Energy levels*.

In semi-classical description, after the excitation RF pulse, the net magnetization relaxes, (i.e., realigns with the main magnetic field). The signal of this relaxation is recorded by a coil. The quantum-theoretical equivalent of the evolution of the net magnetization vector is the absorption and emission of photons by which the nuclear spins flip and the nuclei transit between the discrete energy levels. Caused by emission of the energy difference, a signal in the RF coil is received and recorded in the form of a Free Induction Decay (FID) or an echo. In this process, the energy-level-population numbers change over time according to the Boltzmann statistics.

The two underlying theories are two different tools that describe the same process. Most of the NMR phenomena are described more intuitively using the classical picture. Therefore, we will begin with purely classical explanations and then introduce quantum-mechanical interpretations. There is an alternative quantum mechanical analogy for most classical phenomena. But, some behaviours can be fully accounted for only by quantum mechanics.

### 1.2.1 Classical picture

The basis of the classical picture builds the angular momentum, $\vec{L}$, which is a rotational analogy to the linear momentum. It characterises a cyclical movement about a central axis. The rotation is caused by torque, which is the rate of change of the angular momentum. To calculate the angular momentum, only the tangential component, $\vec{p}_{tang}$, of the momentum is relevant and is then multiplied by the particle's radius, $r$, Eq. 1.4. The angular momentum can also be calculated by multiplying the magnitude of the momentum, $\vec{p}$, by the momentum lever arm, $a$, Eq. 1.5 [3].

$$\vec{L} = r \cdot \vec{p}_{tang} \tag{1.4}$$

$$\vec{L} = a \cdot \vec{p} \tag{1.5}$$

Classically, the magnetic moment $\vec{\mu}$ arises from the spin angular momentum of a nucleus $\vec{L}$ from Eqs. 1.4 or 1.5, times a factor, $\gamma$, Eq. 1.6. All nuclei with a non-zero spin therefore possess a magnetic moment:

$$\vec{\mu} = \gamma \vec{L} \tag{1.6}$$

where $\gamma$ is the gyromagnetic ratio, an isotope-specific property. The gyromagnetic ratio can be determined as the quotient of a particle's charge, $e$, to its twofold mass, $m$:

$$\gamma = \frac{e}{2m} \tag{1.7}$$

The concept of the angular momentum and magnetic moment also applies to subatomic particles, such as protons, which is relevant for our considerations. According to classical theory, angular momentum, and therefore, the spin as well, can attain any value. The spin can be intuitively visualized as a rotation of a spherical particle around its axis. The angular momentum of a proton is synonymous with the nuclear spin of a hydrogen nucleus, as it consists of only one proton.

Based on the magnetic moment, a proton behaves as a magnetic dipole with a north and south pole, spatial orientation, and magnitude. The angular momentum is linked to the nuclear magnetic moment because nucleons (i.e., protons and neutrons) consist of charged quarks which possess their own spins [4]. When there is no magnetic force acting on the magnetic moment, its orientation is random. In an ensemble of several protons, the orientation of their magnetic moments is collectively canceled out, resulting in a zero net magnetization of the sample.

According to the classical description of NMR developed by Felix Bloch, the magnetic moments of nuclear spins can be added. This vector addition leads to a macroscopic magnetization vector, $\vec{M}$, formalized in Eq. 1.3. This classical model is useful for the illustration of basic NMR concepts discussed in the Section on *Basics of Magnetic Resonance Imaging*.

Essential for NMR is the presence of an external magnetic field, $\vec{B}_0$. This magnetic field is static and homogeneous over a certain volume in which the sample is placed. As the sample consists of magnetic moments, it interacts with the field $\vec{B}_0$. The magnetic interaction between the outer field and the nuclear magnetic moments exert a torque, $\vec{T}$, on the nuclear spins as a change of the angular momentum in time, see Eq. 1.8. The resulting torque is perpendicular to the direction of both counterparts, see Eq. 1.9:

$$\vec{T} = \frac{d\vec{L}}{dt} \tag{1.8}$$

$$\vec{T} = \vec{\mu} \times \vec{B}_0 \tag{1.9}$$

Under the action of the torque, nuclear spins behave in the external magnetic field as bar magnets. They gradually align their magnetic dipole moment with the direction of the external field. At first, spins swing over the equal orientation axis when initially coming into contact with the field, and then, swing back and forth again (decreasing in amplitude) according to their natural frequency and eventually stabilizing in the middle. The nuclear magnetic moments are forced to change their orientation according to the field. Most of the spins are still oriented randomly, but a certain portion of the spins is oriented parallel to $\vec{B}_0$. A small portion of the magnetic moments aligns anti-parallel to $\vec{B}_0$, as depicted in Fig. 1.1 [1].

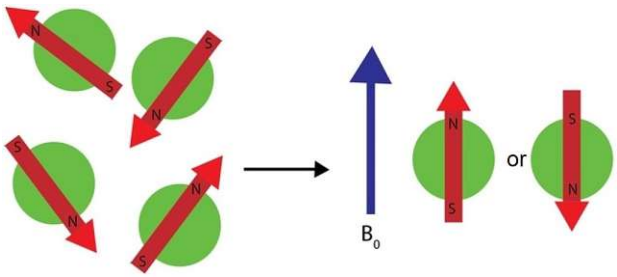


Figure 1.1: Random orientation of nuclear magnetic moments without the presence of an external magnetic field $\vec{B}_0$ and the parallel and anti-parallel orientation inside $\vec{B}_0$ [1].

In summary, there is a slight bias toward a parallel orientation with $\vec{B}_0$. This results in the net magnetization, $\vec{M}$, depicted in Fig. 1.2.

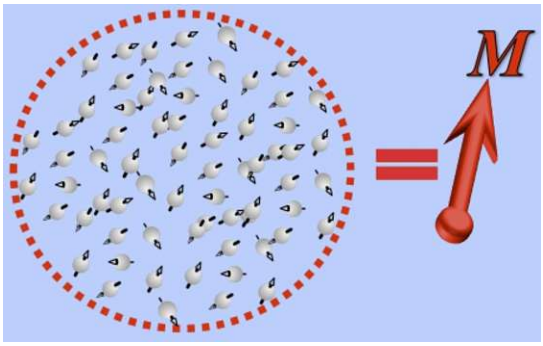


Figure 1.2: Addition of the magnetic moments of a single nuclei to one macroscopic magnetization vector [5].

Apart from the tendency of the spins to align with the field $\vec{B}_0$, they also start to precess around the external field. This is because the exerted torque changes their angular momentum perpendicularly to the field's orientation. This process is known as the Larmor precession with the Larmor frequency, $\omega_0$:

$$\omega_0 = \gamma B_0 \tag{1.10}$$

whereby the Larmor frequency depends on the gyromagnetic ratio $\gamma$ and the amplitude of the external field $\vec{B}_0$.

### 1.2.2 Bloch equations

The Bloch equations are a classical tool with which to quantify the movement of the net magnetization vector, $\vec{M}$, in the three spatial directions. Therefore, there are three equations written in the form of first-time derivatives of the macroscopic magnetization, $M$. Those equations deal with the excitation RF pulses and the relaxation processes. However, they do not include the scalar coupling effects, which can be described only by quantum theory.

The equations are known in two different forms depending on the reference coordinate system (i.e., on the position of the observer). First, the equations are written in the *laboratory* frame of reference. This coordinate system is anchored in the experiment room with regard to its position and orientation. In the simplest case, when the relaxation processes are not considered, the components of $\vec{M}$ are described as follows:

$$\begin{aligned}
\frac{dM_x(t)}{dt} &= \gamma[M_y(t)B_0 - M_z(t)B_{1y}] \\
\frac{dM_y(t)}{dt} &= \gamma[M_z(t)B_{1x} - M_x(t)B_0] \\
\frac{dM_z(t)}{dt} &= \gamma[M_x(t)B_{1y} - M_y(t)B_{1x}]
\end{aligned} \tag{1.11}$$

The magnetization vector, $\vec{M}$, is represented, in Eqs. 1.11, by is components, $(M_x, M_y, M_z)$, as is also the RF magnetic field $\vec{B}_1 = (B_{1x}, B_{1y})$ (i.e., the excitation pulses). To add the relaxation processes to our consideration, Eqs. 1.11 must be extended with an additional term that is different for each of the three components. Bloch equations, in the laboratory frame of reference, including also the relaxation processes, are given in Eqs. 1.12.

$$\frac{dM_x(t)}{dt} = \gamma[M_y(t)B_0 - M_z(t)B_{1y}] - \frac{M_x(t)}{T2} \tag{1.12}$$

$$\frac{dM_y(t)}{dt} = \gamma[M_z(t)B_{1x} - M_x(t)B_0] - \frac{M_y(t)}{T2}$$

$$\frac{dM_z(t)}{dt} = \gamma[M_x(t)B_{1y} - M_y(t)B_{1x}] - \frac{M_z(t) - M_0}{T1}$$

In Eqs. 1.12, $T1$ is the spin-lattice relaxation constant and $T2$ characterises the spin-spin relaxation.

The other possibility to describe the Bloch equations is in the *rotating* frame of reference. Here, the reference coordinate system rotates around the main magnetic field, $\vec{B}_0$, with a frequency, $\omega_0$. The frequency, $\omega_0$, is identical to the frequency of the RF field, $\vec{B}_1$. After coordinate system conversion into the rotating frame, the expressions for the macroscopic magnetization $M'$ change to Eqs. 1.13.

$$\frac{dM'_x(t)}{dt} = (\omega_0 - \omega)M'_y(t) - \frac{M'_x(t)}{T2} \tag{1.13}$$

$$\frac{dM'_y(t)}{dt} = -(\omega_0 - \omega)M'_x(t) + \gamma B_1 M'_z(t) - \frac{M'_y(t)}{T2}$$

$$\frac{dM'_z(t)}{dt} = -\gamma B_1 M'_y(t) - \frac{M'_z(t) - M_0}{T1}$$

In contrast to the *laboratory* frame of reference, in the *rotating* frame, the RF field, $\vec{B}_1$, is oriented parallel to the $x'$ axis. The frequency offset of the main magnetic field $\vec{B}_0$ of $\omega_0 = \gamma B_0$ was reduced to $(\omega_0 - \omega)$, where $\omega$ is the actual frequency considering the effects of magnetic field inhomogeneities. This reduced frequency-offset term equals zero in the on-resonance condition. This means that the transverse magnetization, $M'_x(t)$ and $M'_y(t)$, ceases its precession in the rotating frame.

The rotating frame of reference is a useful tool with which to follow the magnetization vector because it virtually removes the Larmor precession, which makes the movement easier, but does not lose any of its important aspects. Therefore, one can focus on the rotations caused by the RF pulse [6].

### 1.2.3 Classical phenomena

One of the most pronounced classical phenomena is relaxation. The term relaxation explains the causes of the temporal evolution of the net magnetization, $\vec{M}$, back to its thermal equilibrium after the excitation pulse, $\vec{B}_1$, was applied, as stated in the Bloch equations. There are two main types of relaxation. The first is the so-called T1 relaxation and the other the T2 relaxation, which is more practically characterized by the so-called T2* relaxation. The physical cause of the relaxation processes is mainly the dipole-dipole interaction acting through space which differentiates it from the scalar coupling, discussed in the Subsection on *Quantum-mechanical phenomena*, based on chemical bonds.

The magnetization vector is flipped by the excitation RF pulse from the z-axis into the transverse xy-plane. This creates a non-equilibrium state. The spins tend to re-align with $\vec{B}_0$ as the RF field is turned off after the pulse. The equilibrium state is characterized by a net magnetization vector along the z-axis with no transverse component. Regaining the equilibrium state is described by T1 (i.e., *spin-lattice*) relaxation, in Fig. 1.3. After five T1 time intervals, the system recovers equilibrium again. Therefore, the T1 constant describes how fast the system rebounds back to thermal equilibrium. For protons, the *spin-lattice* relaxation time is usually longer at higher field strengths.

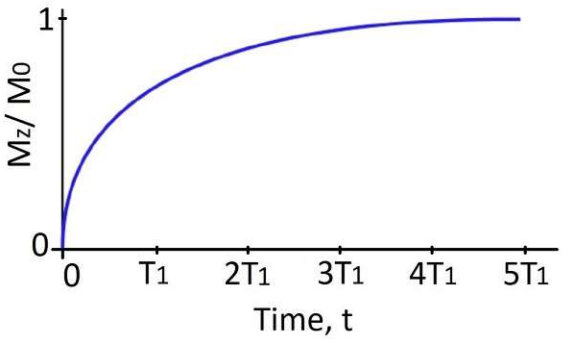


Figure 1.3: Diagram of the T1 relaxation process showing the exponential recovery of the longitudinal magnetization after an excitation pulse [1].

The recovery of the longitudinal magnetization, $M_z$, according to Eq. 1.14, shows an exponential growth toward the thermal equilibrium magnetization, $M_0$ [1]:

$$M_z(t) = M_z(0) \cdot (1 - e^{-\frac{t}{T1}}) \tag{1.14}$$

The effects of T1 relaxation are projected into virtually all MR experiments. There are some methods how to eliminate the effects of the reappearance of the longitudinal magnetization. The simplest way is to make the repetition time (TR) five times longer than the T1 constant, which enables full regeneration of the thermal equilibrium before the next excitation so that the slow *spin-lattice* relaxation does not affect the measurement substantially. On one hand, this technique delivers a decreased SNR per unit of measurement time, but it is a simple technique used in MR spectroscopy to eliminate the T1 relaxation effects. In contrast to MR spectroscopy, in MR imaging, a shorter TR is chosen and the pronounced effects of T1 relaxation are utilized to create contrast as different tissues possess different T1 constants.

On the contrary, the T2 relaxation constant expresses how fast the transverse magnetization disappears. This phenomenon, known as the *spin-spin* relaxation, is depicted in Fig. 1.4 and generally happens faster than the *spin-lattice* relaxation.

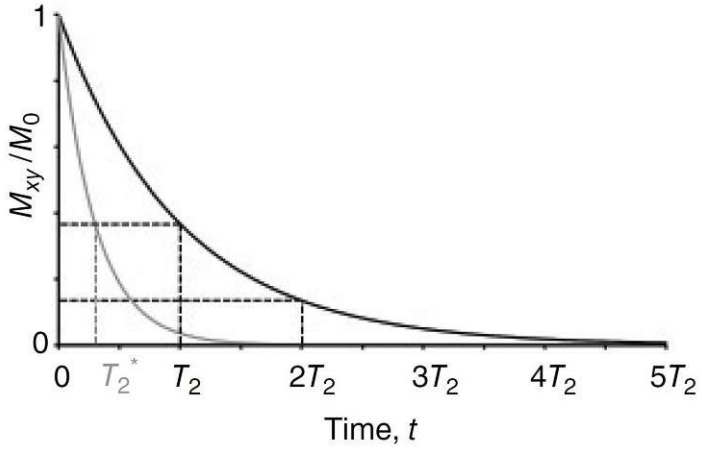


Figure 1.4: Diagram of the T2 (black) and the T2* (grey) relaxation processes showing the exponential loss of the transverse magnetization after an excitation pulse [6].

The T2 relaxation is caused by the fact that the sample is subject to some random field fluctuations related to Brownian motion. Thus, after the excitation, the Larmor frequencies of the

individual spins start to differ from each other. According to classical theory, this is due to the perturbations of the local magnetic fields, which, in turn, influence the orientation of the nuclear spins. This leads to a decrease in the transverse magnetization. After some time, the spins are asynchronous, which corresponds to a total loss of the transverse magnetization. The spin synchrony refers to phase coherence. NMR is characterized by a relatively long-lasting phase coherence, which leads to long life-times of detectable magnetization in the transverse plane, as T2 is very short in solid material, but very long in liquids, which are compositionally much more similar to soft tissues. This has, as a consequence, the invention of a broad range of sequences where RF pulses are combined in all possible ways. In contrast to the *spin-lattice* relaxation constant, the *spin-spin* constant is, in theory, seen as independent of the magnetic field strength, but the apparent change is not. In real T2 measurements, the constant indeed decreases with an increasing $B_0$. The temporal evolution of the transverse magnetization is described by Eq. 1.15.

$$M_{xy}(t) = M_{xy}(0) \cdot e^{-\frac{t}{T2}} \tag{1.15}$$

The disappearance of the transverse magnetization in the form of an exponential decay that reaches zero magnetization in the xy-plane after five T2 relaxation times is clearly shown. As the T2 relaxation is caused by random processes, the signal cannot be corrected for the effects of T2 relaxation.

Actually, the loss of the transverse magnetization is faster than the T2 relaxation and occurs according to T2* relaxation processes, also depicted in Fig. 1.4. This even faster relaxation derives from the fact that, in the sample, there are neighbouring volumes of different magnetic susceptibility. The most pronounced effects are caused on the boundary between air-filled nasal cavities or cranial bones and the liquid in the brain. This leads to inhomogeneities of the magnetic field, which are consequently characterized by faster de-phasing of the spins and a decrease in the magnitude of the transverse magnetization. This ultimately leads to an overall loss of signal. In contrast to the T2 relaxation, the T2* relaxation has non-changing sources. Those can be classified as systematic, and therefore also accounted for in the recovery of some of the lost signal.

As the inhomogeneities of the magnetic field lead to spin de-phasing via T2* relaxation, the signal decreases. Nevertheless, the effects of the T2* relaxation can be eliminated with the spin-echo sequence. This sequence is displayed in Fig. 1.5.

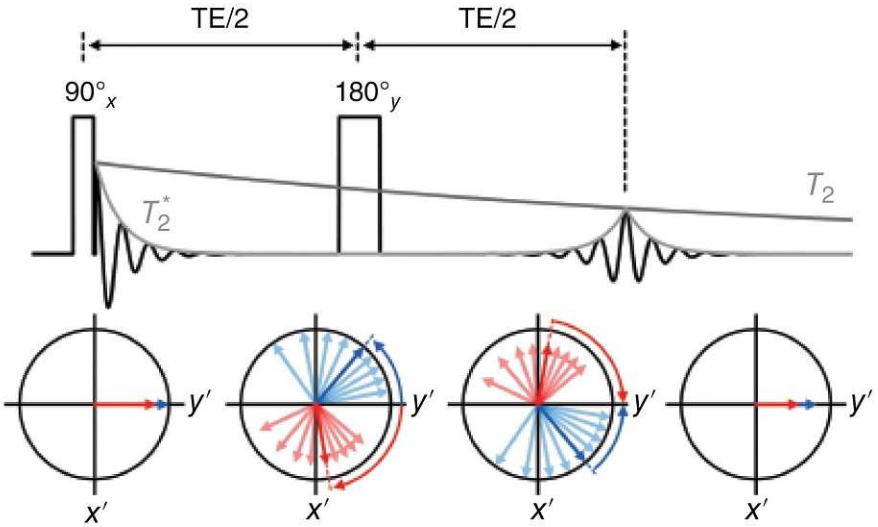


Figure 1.5: Time order of the RF pulses in the spin-echo sequence illustrating the relaxation times T2 and T2* and the repetition time. The influence of the pulses on the transverse magnetization and spin de-phasing is also displayed [6].

The spin-echo sequence starts with a 90° excitation pulse that rotates the longitudinal magnetization into the xy-plane (in Fig. 1.5 the first x'y'-diagram from left). The signal then quickly decays according to T2*. Because of the field inhomogeneities, some spins have acquired more phase (e.g., red arrows) and some less phase (e.g., blue arrows), second diagram from left. After half of the echo time (TE), a 180° refocusing pulse is applied. This pulse inverts the accumulated phases such that during the second half of TE, the spins still accumulate phase, but in the opposite direction, which leads to re-focusing (third diagram from left) and the echo is formed (fourth diagram from left). At this time point, the effects of the T2* relaxation are eliminated.

The interplay of all the three relaxation processes influences the specific form of the recorded signal, the FID. The word *Induction* in the FID term refers to the evolution of the spin phases as they *induce* a signal in the receiver coil. For the basic FID formation, three steps are required. First, it is necessary for the main magnetic field, $\vec{B}_0$, to reach a total non-zero net magnetization. Second, this magnetization is flipped via an RF pulse into the transverse plane, where it starts to precess. Third, the precessing magnetization induces the FID signal that is recorded.

The FID is the information carrier in NMR experiments. Up to this point, we have explored the basic information content, which will be further expanded for *chemical shifts* and *scalar coupling* in the Subsection on *Quantum-mechanical phenomena*, and for *encoding* in the Section on *Basics of Magnetic Resonance Imaging*.

### 1.2.4 Quantum-mechanical picture

As NMR is a spectroscopy method, it acts via electro-magnetic waves to force spins to transit between energy levels [NMR Introduction]. The nucleons inside an atomic nucleus possess angular momentum on their own as in classical theory, but, in quantum mechanics, the orbital angular momenta, $l$, are quantized. The orbital angular momentum, $l$, refers to the mechanical moment of the specific nucleon as it moves inside the nucleus. The possible values for $l$ depend on the principal quantum number $n$, which stands for the occupied energy level. The principal quantum number attains only positive integer values, where the smallest value means the lowest energy level and the highest value the highest energy level. The values for $l$ span from 0 to $(n-1)$ in discrete integer steps. For example, when $n = 3$, the orbital angular momentum can have any of the following values [7]:

$$n = 3 \rightarrow l = 0, 1, 2$$

When the orbital angular momenta of all nucleons in one nucleus are summed, the calculation results in the total orbital angular momentum of the whole nucleus, $L$. The total orbital momentum $L$ is also quantized and takes on values, according to Eq. 1.16.

$$L = \hbar(l(l+1))^{1/2} \tag{1.16}$$

Therefore, the total orbital momentum is given in units of $\hbar$, where $\hbar$ is the reduced Planck's constant, $h/2\pi$, with $h$ being the Planck's constant. The nuclear magnetic moment, $P_m$, arises from the nuclear orbital angular momentum, $L$, as stated in Eq. 1.17.

$$P_m = \frac{e}{2m}L \tag{1.17}$$

The gyromagnetic ratio, $\gamma$, is defined by dividing the magnetic moment, $P_m$, by the orbital momentum, $L$:

$$\gamma = \frac{P_m}{L} = \frac{e}{2m} \tag{1.18}$$

Eq. 1.18 gives the same result as Eq. 1.7. Nuclei with even atomic and neutron numbers have a zero magnetic moment. In nuclei with odd atomic and neutron numbers, the magnetic moment is determined from the excess nucleons [8].

### 1.2.5 Energy levels

From the quantum-mechanical point of view, the state of a nuclear spin is defined by its energy, $E_m$, it has in a magnetic field, $B$, as was already stated in Eq. 1.2:

$$E_m = -\hbar\gamma_S m_S B$$

In this particular form of the equation, $m_S$ signifies the direction of the spin angular momentum quantum number of a given nucleus, $S$. The orientation of the spins is parallel with the main magnetic field and can be either in the same direction as $\vec{B}_0$, where the nuclei are at the lower magnetic energy level. The opposite is true when the nuclear magnetic moment aligns in the opposite direction to the external field and is at the upper energy level. The number of the magnetic energy levels, $n_E$, depends on the quantum number, $m_S$, as there are $2m_S + 1$ energy levels for each $m_S$. For example, the hydrogen nucleus with $m_H = 1/2$ splits into

$$n_E = 2m_H + 1 = 2 \cdot 1/2 + 1 = 2$$

energy levels when placed inside a magnetic field. The transition between energy levels corresponds to the energy quantum $\Delta E$, a photon, with a specific frequency $\nu$:

$$\Delta E = h\nu \tag{1.19}$$

The frequency expressed from Eq. 1.19 is then given as:

$$\nu = \frac{\Delta E}{h} \tag{1.20}$$

From Eq. 1.21, we know that the energy difference between two energy levels in an external magnetic field is:

$$\Delta E = \hbar\gamma B_0 \tag{1.21}$$

This is because the energy levels are one $m_H$ apart. Inserting Eq. 1.21 into 1.20 results in:

$$\nu = \frac{\hbar\gamma B_0}{h} = \frac{\gamma B_0}{2\pi} \tag{1.22}$$

According to Eq. 1.22, the frequency of the emitted or absorbed quantum $\nu$ is defined in terms of the external magnetic field strength $B_0$ and the gyromagnetic ratio $\gamma$.

### 1.2.6 Quantum-mechanical phenomena

Phenomena describable by quantum theory will come into focus here. First among the quantum phenomena is the Zeeman effect, or Zeeman splitting, which is depicted in Fig. 1.6, on the example of a spin-1/2 hydrogen nucleus.

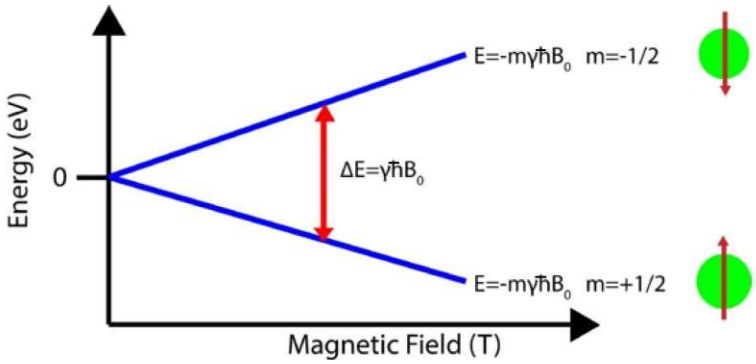


Figure 1.6: Splitting of the energy levels of a hydrogen nucleus depending on $B_0$ [1].

As illustrated in Fig. 1.6, the splitting of the energy levels occurs only in the presence of a magnetic field. The energy difference between the energy levels is linearly proportional to the field strength of the applied magnetic field. Therefore, it follows that, when there is no magnetic field, there is no splitting of the energy levels and the spins are in a so-called degenerate state. The degenerate state of the spins is the quantum equivalent of the classical zero net magnetization vector. Furthermore, the energy gap is dependent on the gyromagnetic ratio, which varies with the isotope of interest. As stated above, the energy gap corresponds to the energy quantum of photons of a specific frequency, $\nu$, which lies in the RF range, expressed in Eq. 1.24. The RF quantum is provided by an RF pulse for a transition from the lower to the upper level. This transition is depicted in Fig. 1.7.

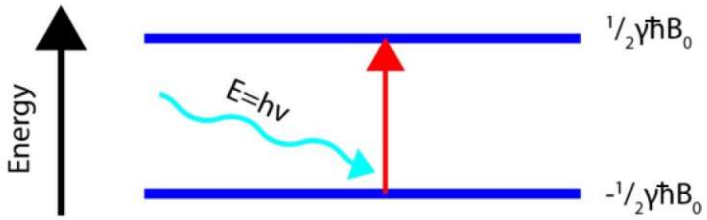


Figure 1.7: Transition from the lower nuclear magnetic energy level to the upper level (i.e., spin flip) via energy absorption from a photon of frequency, $\nu$ [1].

This frequency, for in vivo NMR, typically lies in the range between tens of MHz and the lower hundreds of MHz, whereby the highest $\nu$ is for protons at 7 T with 297 MHz. Those nuclei whose spin $m_S$ is oriented in the same direction as the magnetic field (in Fig. 1.6, i.e., $m_H = +1/2$) are in the lower energy state and can transit to the upper state via absorption of an RF quantum. On the other hand, those nuclei whose spin points in the opposite direction to the external field (i.e., $m_H = -1/2$) are in the higher energy state and can relax into the lower energy state via an emission of an RF photon [1].

The relative spin population of the energy levels is described by the Boltzmann distribution and depends on the energy difference, $\Delta E$, between the levels and the temperature, $T$:

$$\frac{N_+}{N_-} = e^{-\frac{\Delta E}{kT}} \tag{1.23}$$

In Eg. 1.23, $N_+$ is the population of the upper energy level, and $N_-$ is the lower level population. The connection between the energy difference and the temperature is established via the Boltzmann constant, $k = 1.380649 \cdot 10^{-23} J/K$. The energy difference can be calculated as follows for the example of carbon-13:

$$\Delta E = \hbar \gamma_{^{13}C} B_0 \tag{1.24}$$

$$\Delta E_{7T} = 1.0546 \cdot 10^{-34} \frac{J}{Hz} \cdot 10.7084 \frac{MHz}{T} \cdot 7T$$

$$\Delta E_{7T} = 7.9052 \cdot 10^{-27} J$$

Using the specific gyromagnetic ratio, $\gamma_{^{13}C}$, of carbon-13, it can be seen that the energy levels are relatively close to each other, thus, the thermal energy of the environment can easily cause the lower energy spins to flip into the upper level. The NMR signal is proportional to the difference between $N_-$ and $N_+$ because it is built up by the difference between the emitted energy (i.e., from spins which flip down) and the absorbed energy (i.e., of spins that flip up).

By inserting the calculated $\Delta E_{7T}$ into Eq. 1.23 and considering a temperature range between 0.001 and 1,000 K, we receive the following diagram:

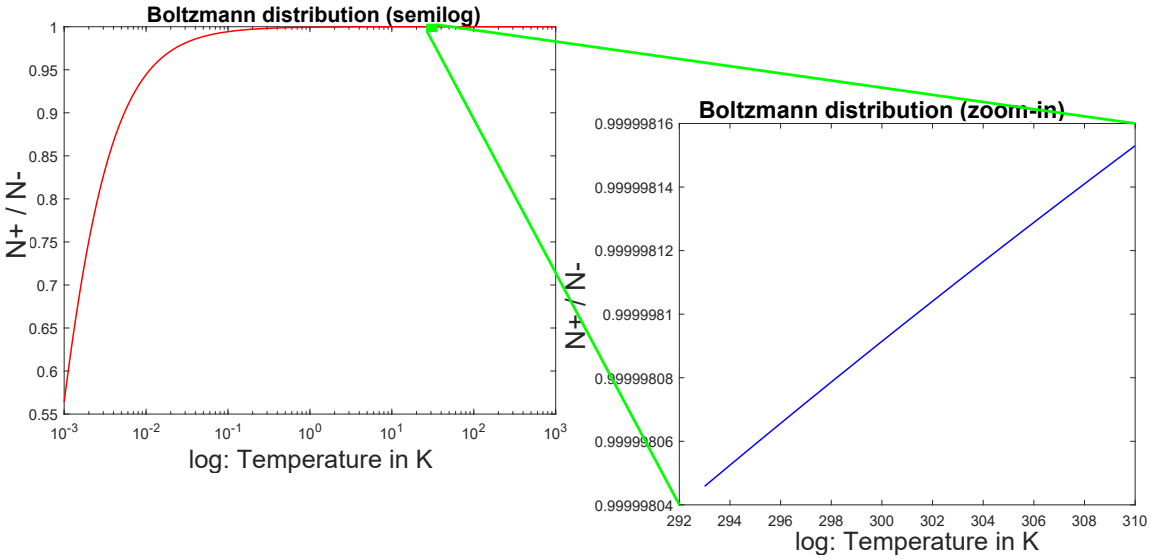


Figure 1.8: Semilog plot of the Boltzmann distribution at 7 T, spanning a temperature range from 0.001 to 1,000 K (red curve) with a zoom-in (green marking) into the body temperature range as a linear plot (blue curve).

The boundary values 0 and 1 of the population ratio mean that, when the ratio equals 0, there are no spins in the upper energy level. As the ratio approaches one, the number of spins in the higher state increases, as can be seen in Fig. 1.8, until the point where the ratio reaches the value of 1, which stands for an equal population of the two levels. As the population ratio first increases rapidly from 0.55 and then approaches the value of 1 slowly, the red curve in Fig. 1.8 was plotted in a semilog mode with a logarithmic temperature scale ranging from 0.001 to 1,000 K. When we look at the same nucleus with identical parameters, but in a 1.5 T scanner instead of a 7 T, the energy difference amounts to:

$$\Delta E_{1.5T} = 1.0546 \cdot 10^{-34} \frac{J}{Hz} \cdot 10.7084 \frac{MHz}{T} \cdot 1.5T \tag{1.25}$$

$$\Delta E_{1.5T} = 1.6940 \cdot 10^{-27} J$$

From the comparison of $\Delta E_{7T}$ and $\Delta E_{1.5T}$, it is obvious that the Zeeman splitting increases linearly with increasing field strength $B_0$. This leads to a change in the population numbers. When we consider an illustrative sample consisting of two million carbon-13 nuclei that are either in the higher, $N_+$, or the lower energy state, $N_-$, we can conclude that

$$N = N_+ + N_- = 10,000,000 \ , \tag{1.26}$$

which can be written as

$$N_+ = 10,000,000 - N_- \ .$$

From the data used for the diagram in Fig. 1.8, we can extract the ratio of the energy level populations for a characteristic temperature in clinical NMR of 36 °C:

$$\frac{N_+}{N_-} = 0.99999814 \tag{1.27}$$

After inserting the modified Eq. 1.26 into 1.27, we arrive at:

$$\frac{2,000,000 - N_-}{N_-} = 0.99999814 \tag{1.28}$$

Solving Eq. 1.28 for $N_-$ gives:

$$N_- = 5,000,005$$

Then, we can easily calculate that, under this condition there are 4,999,995 spins in the upper energy level. This in turn tells us that a sample of ten million spins of carbon-13 nuclei at 36 °C in a 7 T scanner shows a difference in the energy level populations of ten spins. This translates into a difference of one spin per million, which signifies the typical sensitivity of NMR - on the order of parts per million (ppm).

When we perform the same calculation in a 1.5 T field, we arrive at energy level populations of $N_+ = 4,999,999$ and $N_- = 5,000,001$. From the results at different field strengths and identical temperatures, it is apparent that there is a higher difference in energy level populations at stronger fields. Simultaneously, there are more spins in the lower energy level in a stronger magnetic field, i.e., oriented in the same direction as $\vec{B}_0$. A pictorial representation of the described behaviour is displayed in Fig. 1.9.

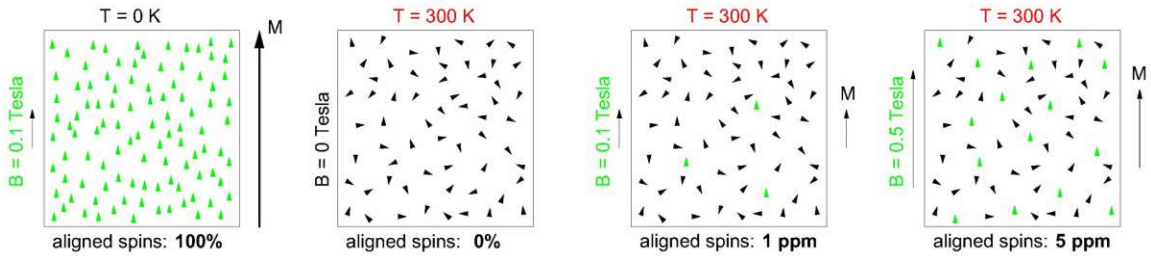


Figure 1.9: Influence of magnetic field strength and temperature on spin alignment [4].

It can be seen that, at absolute zero temperature, all spins would be aligned along the main field, $\vec{B}_0$, as was mathematically demonstrated in Fig. 1.8, albeit at a much smaller field strength of $B_0 = 0.1$ T. At room temperature (i.e., 300 K), the spins are oriented randomly when there is no external magnetic field. As Fig. 1.9 also relies on classical theory using the net magnetization, $\vec{M}$, it shows that, in the second sub-picture from left, $\vec{M}$ equals zero. When the magnetic field strength slightly rises to 0.1 T, one spin in a million aligns with $\vec{B}_0$. By increasing $\vec{B}_0$ to 0.5 T, four more spins align with it.

The relaxation processes from the quantum-mechanical point of view are caused by fluctuations in the local magnetic fields as in classical theory. Those fluctuations then result in spin state transitions. The dynamics of the local fields can be linked to probabilities of the spin flips. Those probabilities are determined by the amplitudes of the transverse local field fluctuations at resonance frequency [6].

The signal in NMR is influenced not only by the population of the nuclear energy levels, transitions and the energy difference between them, but also by the proximity of the nucleus, and the electron shell. This property makes it possible to identify the molecular composition of the sample and is referred to as *nuclear shielding* or *chemical shift*. The *nuclear shielding* arises from the fact that, in a sample, there are not sole nuclei but whole atoms consisting of the nucleus and the electron shell. As electrons are charged particles moving around the nucleus, they generate magnetic fields. Those local magnetic fields interact with the external magnetic field, $\vec{B}_0$. In this way,

the nuclei are shielded to some degree from the external magnetic field, and thus, the designation *nuclear shielding*. The shielding effect influences the nuclear magnetic energy levels and causes the difference between energy levels to slightly decrease, as is illustrated in Fig. 1.10.

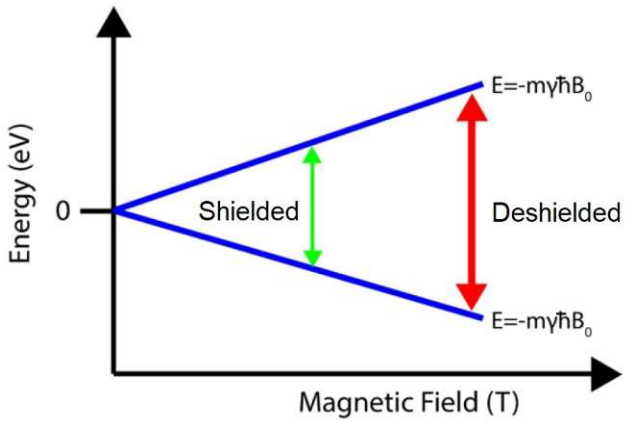


Figure 1.10: Illustrative picture of the difference in the splitting of nuclear magnetic energy levels caused by surrounding atomic shell electrons [1].

The new difference between the energy levels corresponds to a quantum of a different frequency. Therefore, the frequency of the RF pulse must be adapted to enable the spin flip process. As the same nuclei (e.g., hydrogen) inside one molecule are surrounded by different electronic environments, differences in the NMR signal from different chemical groups can be detected. Due to this effect, different chemical substances can be recognized, which is why this effect is termed *chemical shift*.

To be able to differentiate chemical components, the time signal (i.e., FID) must be converted into a spectrum. This is done via a mathematical algorithm termed a Fourier transformation (FT), particularly the fast FT (FFT). This will be discussed in more detail in a later Section on *Data processing concepts in MRSI* [1].

Together with the *chemical shifts*, another quantum-mechanical phenomenon, the *scalar coupling*, enables differentiation of various metabolites in the sample because different molecules show different behaviours with respect to those phenomena. The resulting technique from this interplay is known as NMR spectroscopy (NMRS or simply MRS).

The *chemical shift* discussed earlier is mediated through space, caused by the dipole coupling and holds information about the chemical environment of nuclei. The dipole interactions result in the relaxation processes. The *scalar coupling* is a quantum effect mediated not through space but through chemical bonds [6].

Nuclei in the sample are mostly not separated, but bonded to molecules via chemical bonds. The chemical bonds are mediated by atomic electrons. In a typical organic covalent bond, there are two electrons, one from each of the partnering nuclei. The spin orientation of the binding electron with respect to the nuclear spin orientation influences the energy level. This change in energy levels causes the resonances, transitions between energy levels, to change. Scalar coupling is the interaction between spins of the binding electrons and the spins of the bonded nuclei that leads to the *splitting of resonances*. The *scalar coupling* splits a sole resonance, $\nu_1$, into two, one lower, $\nu_{1-}$, and one higher, $\nu_{1+}$. The spins of the two binding electrons must be oriented anti-parallel to each other (i.e., one up and the other down), according to Pauli's exclusion principle. This physical law determines the relative spin orientation between the nucleus and the binding electron. The situation, where the electron spin is anti-parallel to the nuclear spin, is energetically more favourable compared to the state, where the electron spin and the nuclear spin point to the same direction [6].

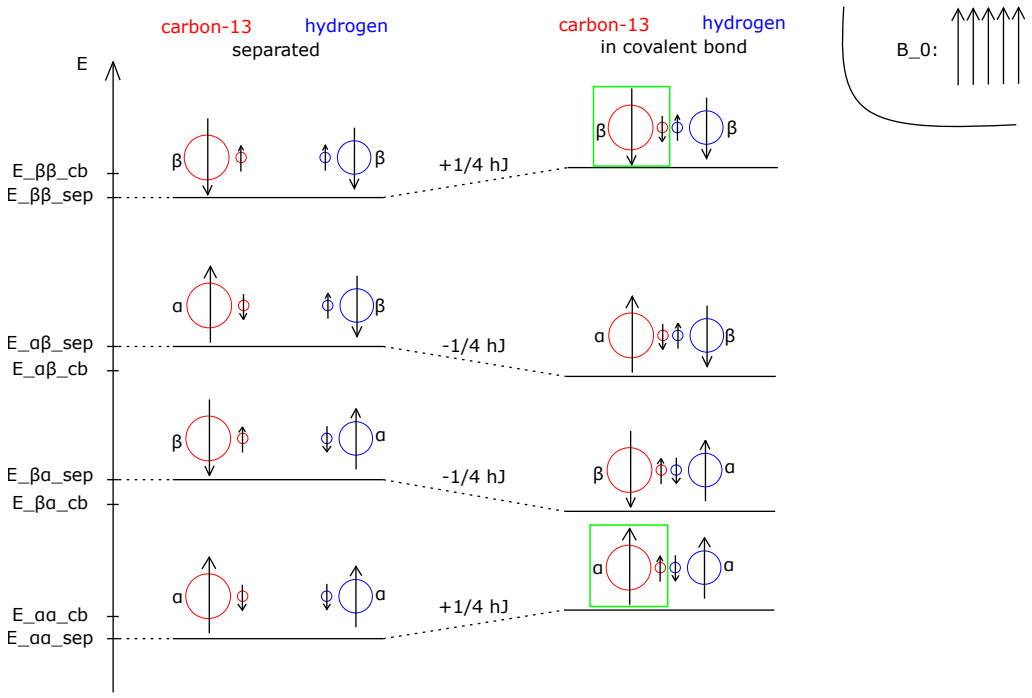


Figure 1.11: Energy level diagram of a two-spin-system: carbon-13 and hydrogen, separated (left) and in a covalent bond (right), visualizing the concept of *scalar coupling* and *resonance splitting*. The smallest circles symbolize the electrons entering the bond. Inspired by [6].

In Fig. 1.11, the index $\beta$ designates the higher energy state and $\alpha$ the lower one. We first consider the carbon-13 and the hydrogen nuclei separately (bottom index *sep*). There are two possible energy level transitions for the carbon-13 nucleus, and likewise for the hydrogen. The possible carbon transitions are:

$$E_{\beta\beta,sep} \rightarrow E_{\alpha\beta,sep}$$

and

$$E_{\beta\alpha,sep} \rightarrow E_{\alpha\alpha,sep}$$

Those transitions are characterized by the spin flip of the carbon while the spin of the hydrogen nucleus remains unchanged. The energy values of the respective levels are:

$$E_{\beta\beta,sep} = +\frac{1}{2}h\nu_C + \frac{1}{2}h\nu_H \tag{1.29}$$

$$E_{\alpha\beta,sep} = -\frac{1}{2}h\nu_C + \frac{1}{2}h\nu_H$$

$$E_{\beta\alpha,sep} = +\frac{1}{2}h\nu_C - \frac{1}{2}h\nu_H$$

$$E_{\alpha\alpha,sep} = -\frac{1}{2}h\nu_C - \frac{1}{2}h\nu_H$$

From Eqs. 1.29, it is obvious that the energy of a given nucleus is given by one half of the Planck's constant, $h$, and the specific frequency, $\nu_C$ or $\nu_H$. The energy is positive when the nuclear

spin is anti-parallel to the direction of the main field $\vec{B}_0$ displayed in the upper right corner of Fig. 1.11, and negative when parallel. The energy of the two nuclei together simply amounts to the sum of the energies of the two separate nuclei.

The energy differences, $\Delta E$, between the levels of the carbon deliver the resonances, given in Eqs. 1.30 and 1.31.

$$\Delta E_{C1,sep} = E_{\beta\beta,sep} - E_{\alpha\beta,sep} \tag{1.30}$$

$$\Delta E_{C1,sep} = +\frac{1}{2}h\nu_C + \frac{1}{2}h\nu_H + \frac{1}{2}h\nu_C - \frac{1}{2}h\nu_H$$

$$\Delta E_{C1,sep} = h\nu_C$$

and

$$\Delta E_{C2,sep} = E_{\beta\alpha,sep} - E_{\alpha\alpha,sep} \tag{1.31}$$

$$\Delta E_{C2,sep} = +\frac{1}{2}h\nu_C - \frac{1}{2}h\nu_H + \frac{1}{2}h\nu_C + \frac{1}{2}h\nu_H$$

$$\Delta E_{C2,sep} = h\nu_C$$

We can clearly see that $\Delta E_{C1,sep} = \Delta E_{C2,sep} = \Delta E_{C,sep} = h\nu_C$, meaning that the two transitions result in the same resonance. Similar arguments hold true for hydrogen, which also results in only one resonance for the two transitions, but with the frequency $\nu_H$ instead of $\nu_C$.

This is different from the case on the right side of Fig. 1.11, where the carbon and the hydrogen atoms are covalently attached. In the covalent bond, the two electrons build a pair. For paired electrons, the Pauli's exclusion principle forbids the two electrons to be in an identical quantum state. Therefore, one of the electron spins must be up and the other down. In the case of the $\beta\beta$ and the $\alpha\alpha$ state, this forces one of the electron spins to be parallel to the nuclear spin, thus entering an energetically less favourable state (i.e., a state of higher energy). The two cases of the parallel nuclear-electron orientations are highlighted in Fig. 1.11 by a green rectangle. The increase in energy of the two energy levels equals $\frac{1}{4}hJ$, where $J$ is the *coupling constant*, which gives the phenomenon of *scalar coupling* also the name *J coupling*.

In the case of the $\alpha\beta$ and $\beta\alpha$ states, the situation is different. Both of the nuclear-electron spin pairs are in an anti-parallel arrangement, leading to an energy-level decrease of $\frac{1}{4}hJ$ in both cases. This makes those states energetically more favourable, resulting in energy values of the chemically bonded (signified by a lower index *cd*) carbon to hydrogen, formalized in Eqs. 1.32:

$$E_{\beta\beta,cb} = E_{\beta\beta,sep} + \frac{1}{4}hJ \tag{1.32}$$

$$E_{\alpha\beta,cb} = E_{\alpha\beta,sep} - \frac{1}{4}hJ$$

$$E_{\beta\alpha,cb} = E_{\beta\alpha,sep} - \frac{1}{4}hJ$$

$$E_{\alpha\alpha,cb} = E_{\alpha\alpha,sep} + \frac{1}{4}hJ$$

The shift in energy levels of $\frac{1}{4}hJ$ leads to the *splitting of resonances*, both for the carbon and the hydrogen. Let us focus now on only the spin flip of the carbon nucleus, which is seen in transitions formalized in Eqs. 1.33.

$$E_{\beta\beta,cb} \rightarrow E_{\alpha\beta,cb} \tag{1.33}$$

and

$$E_{\beta\alpha,cb} \rightarrow E_{\alpha\alpha,cb}$$

The differences between those energy levels are calculated in Eqs. 1.34 and 1.35.

$$\Delta E_{C1,cb} = E_{\beta\beta,cb} - E_{\alpha\beta,cb} \tag{1.34}$$

$$\Delta E_{C1,cb} = E_{\beta\beta,sep} + \frac{1}{4}hJ - E_{\alpha\beta,sep} + \frac{1}{4}hJ$$

$$\Delta E_{C1,cb} = h\nu_C + \frac{1}{2}hJ$$

and

$$\Delta E_{C2,cb} = E_{\beta\alpha,cb} - E_{\alpha\alpha,cb} \tag{1.35}$$

$$\Delta E_{C2,cb} = E_{\beta\alpha,sep} - \frac{1}{4}hJ - E_{\alpha\alpha,sep} - \frac{1}{4}hJ$$

$$\Delta E_{C2,cb} = h\nu_C - \frac{1}{2}hJ$$

For the two possible carbon transitions, there are two distinct resonances that lead to *resonance splitting*. The same holds true for the hydrogen resonances, which are $\Delta E_{H1,cb} = h\nu_H + \frac{1}{2}hJ$ and $\Delta E_{H2,cb} = h\nu_H - \frac{1}{2}hJ$. This splitting of resonances is observed in the spectra of Fig. 1.12.

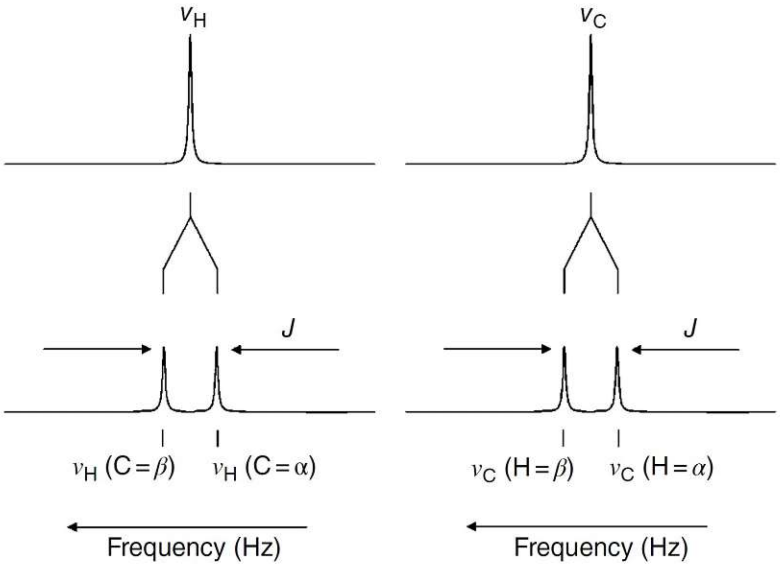


Figure 1.12: Spectra of the carbon and hydrogen nuclei separately (upper part) and in a chemical bond (lower part), showing *resonance splitting* due to *scalar coupling* [6].

In this simple case, the two carbon resonances are in the spectrum exactly one $J$ apart from each other, as follows from the results of $\Delta E_{C1,cb}$ and $\Delta E_{C2,cb}$. The same holds true for hydrogen.

As the primary focus of this diploma thesis is not the phenomenon of *scalar coupling*, the consideration of the simplest case, a two-spin-system over a single bond, provides a basic understanding.

Let us only briefly state that the resulting spectra from this effect can be of two sorts: first-order spectra and second-order spectra. Those two orders of spectra depend on whether there is a heteronuclear (as is the case with the previous considerations that lead to a weakly-coupled spin-system) or a homo-nuclear interaction (e.g., carbon-carbon bonding termed a strongly-coupled spin-system) underlying the process. Furthermore, it is differentiated between *scalar coupling* over one bond, as in our example, and also over two and three bonds. The effects over four bonds can be mostly neglected, as the coupling constant $J$ decreases rapidly with the number of bonds [6].

## 1.3 Basics of Magnetic Resonance Imaging

The basics of Magnetic Resonance Imaging (MRI) are the NMR processes that lead to the creation of the signal and the localization of the spatial origin of the signal, enabling the reconstruction of images. Both processes can be done in numerous ways. In this introduction, a few basic combinations of the two MRI components will be described. Generally, the signal (i.e., FIDs and spin-echoes) is induced via RF pulses and the localization is achieved by means of magnetic field gradients, $G$, additional to the main magnetic field in which the gradients can also be used to bring up additional signals in the form of gradient-echoes [9]. The temporal arrangement of the pulses and gradients, as well as the thereby created signals, are presented in the form of sequence diagrams. First, only the signal-inducing parts of the sequences will be discussed in more detail, followed by standard localization techniques that use a special matrix for systematic data storage, the *k-space*, and finally arriving at localized sequences.

The RF pulses are emitted by special coils termed RF transmit coils. Those are of three basic forms: surface; volume; and array coils. Each of the coil types has its own use. When the RF coil is used to excite the spins by means of RF pulses, it is working in the *transmit* mode. The other mode the RF coil can work in is the *receive* mode. The RF coil also receives the signal that was induced by the previous excitation. During the recording of the signal, an analog-to-digital converter discretely samples the measured electrical voltage in the format of complex numbers.

The most basic MRI sequence consists of only one 90° pulse (i.e., *flip angle* of 90° or excitation) and localization gradients. With this sequence, localized FIDs are measured. Another possibility to obtain MRI measurements is by inducing an echo after the FID and measuring the echo signal instead [6]. The production of an echo can be achieved in two major ways - either by a refocusing 180° pulse after the excitation, which gives rise to a spin-echo or by an alternating gradient, that results in a gradient-echo [9]. The spin-echo sequence is depicted in Fig. 1.13.

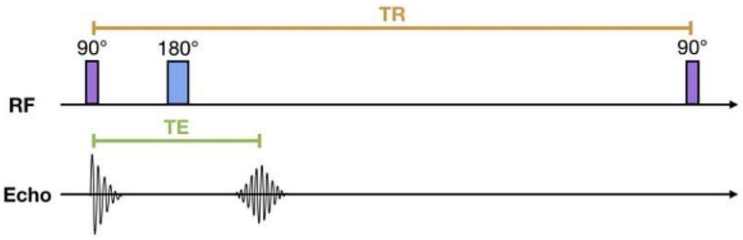


Figure 1.13: The RF pulses of a conventional spin-echo sequence with the induced signals and the timing of the sequence in the form of $TE$ and repetition time ($TR$) [10].

In Fig. 1.13, a refocusing pulse is applied $TE/2$ after the excitation. As was already briefly touched upon the previous section, the first excitation pulse flips the *net magnetization* into the xy-plane, inducing voltage in the receiver coil that decays according to the relaxation processes (i.e., as the spins de-phase), resulting in the FID. Then, the 180° pulse causes the remaining in-plane magnetization to point in the opposite direction, thus inverting the sign of the phase

accumulation. The spins that gain positive phase, as well as spins that accumulate negative phase, start to approach synchrony, which leads to a rising signal as more and more spins are in-phase. When all spins are synchronized, the echo peaks at $TE$ and then decays. After one $TR$, this whole procedure is repeated, as depicted in Fig. 1.13. This is the basic, non-localized version of the spin-echo sequence.

The difference between the FID and the echo signal is, that at the time of the excitation pulse, the spins are synchronized due to the net magnetization created through the main magnetic field, and, as they de-synchronize, the decaying signal is the FID. On the other hand, the echo is recorded when a 180° pulse is applied at $TE/2$ after the excitation pulse.

The other way to produce an echo is utilized in the gradient-echo sequence, which consists of one excitation pulse with a *flip angle* $\alpha$ and two gradient pulses, depicted in Fig. 1.14. The two gradient pulses are of an opposite sign. The goal of the first gradient pulse is to generate a *phase shift*. The second gradient pulse is applied simultaneously with the recording of the signal, and thus, also called the readout gradient. Because of the opposite sign of the second gradient, the phase difference gradually decreases, resulting in the rise of the echo peak again when all spins are in-phase and then decreasing as the spins de-phase.

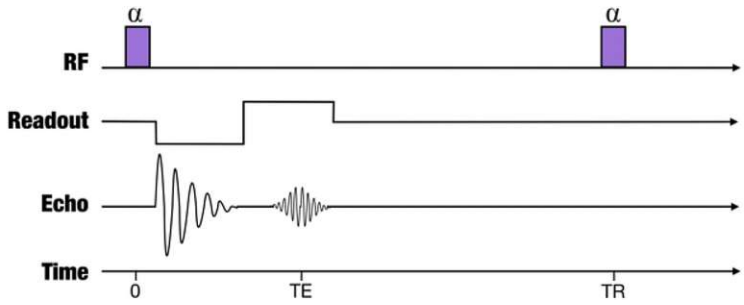


Figure 1.14: The RF pulses of a gradient-echo sequence with the induced signals and timing [9].

The signal localization can be accomplished via either traditional *phase encoding* or *spatio-spectral encoding* (SSE). Both of these approaches use magnetic field gradients that are created by special gradient coils inserted into the gantry of the scanner. The simplest gradient system consists of three coils, each of which is reserved for one spatial direction. Thus, there are the x-, y-, and z-coils. The x-coil creates a magnetic field that changes in strength along the x-direction, the y- and z-coil do the same in the other two directions, as is displayed in Fig. 1.15 [6].

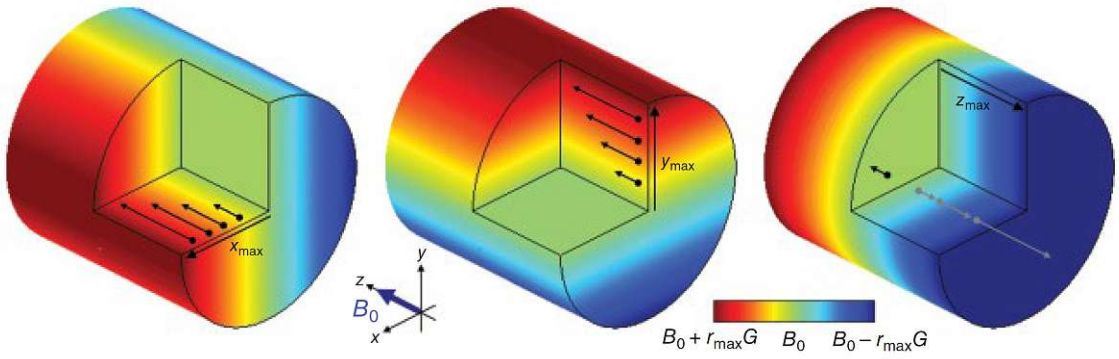


Figure 1.15: Magnetic field gradients along the x-, y-, and z-directions shown separately, created by the x, y, and z-gradient coils, respectively [6].

Be aware that in Fig. 1.15, as in all MR experiments, the main field, $\vec{B}_0$, always points to the z-direction, regardless of the applied gradient. The gradients generate a change only in the magnetic field strength, but do not change the overall direction of the field. The main magnetic field and the gradients combine and the final field distribution is a result of their vector addition. One important property of the gradients is that they reduce $B_0$ on one side and increase it on the other side [6].

For spatial encoding, the gradients and their timing, with respect to excitation pulses and readout of the signal, are essential. The conventional localization strategy is to encode the signal in the three spatial dimensions with three separate gradients in which each of those gradients covers one of the spatial axes. In the z-direction, the signal is often localized via *slice selection*. After the *slice selection*, the signal still needs to be localized in the x- and y-directions (i.e., in the xy-plane, in-plane) to create an image of the slice. In one of those directions, the spatial information is usually added by *frequency encoding* and in the other by *phase encoding*. This is the conventional strategy. Nevertheless, it is also possible to localize the signal in-plane by *frequency encoding*, with two gradients by varying their magnetic field strength in orthogonal directions [6] or with various *k-space* trajectories, as will be discussed in a later section.

As mentioned above, the data recorded in MRI is stored in *k-space*. The applied gradients are directly related to the position in this data matrix. By varying the gradients, the whole *k-space* is filled along a trajectory. The trajectories vary for different gradients. A typical *phase encoded k-space* is depicted in Fig. 1.16.

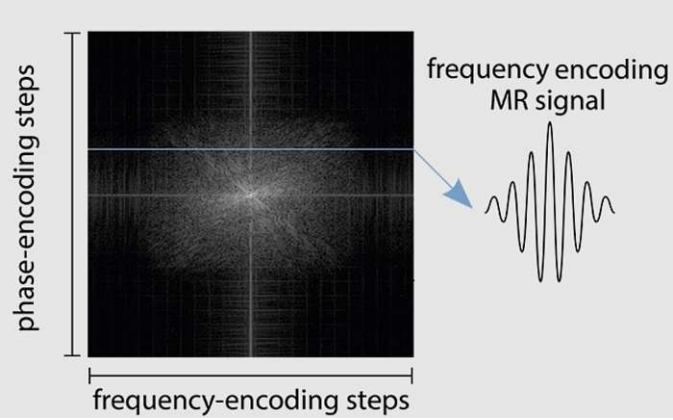


Figure 1.16: Recorded raw MRI data with a detail of one *k-space* line highlighted in blue [11].

The *frequency encoding* is on the horizontal axis and the *phase encoding* is on the vertical axis. The axis descriptions are apt, as the signal is stored depending on the encoded frequency, respectively phase, which is determined in the acquisition process by the applied gradients and their timing. The highlighted blue line in Fig. 1.16 holds information about one *phase-encoding* step, which usually corresponds to one signal readout. Along the horizontal lines, the data of the transverse magnetization are stored in the form of an FID or an echo [12].

What exactly is measured is the complex-valued voltage sampled from the receiver coil as a function of time. Depending on the sequence, either one or more slices are acquired. In case different slices are *slice-encoded*, the measurement is called *multi-slice 2D*, whereas when *phase encoding* is used, it is referred to as *3D*. When acquiring more than one slice, the two-dimensional *k-spaces* are stacked atop each other in the z-direction.

The arrangement of the data is such that the details of the image with low signal intensities at the periphery of the *k-space* are represented (i.e., grey and black regions), mostly sharp edges termed high spatial frequencies because the contours switch very rapidly from one form to another. Opposed to that, the middle of the data matrix contains the bulk of the signal (i.e., white), which

represents low spatial frequencies. Therefore, it is said that *k-space* represents the spatial frequencies of the image, and thus, is also denoted as *spatial frequency space*[12].

To come from the *k-space* to a basic image, a spatial FT is performed in all measured space directions [12].

One pixel in *k-space* contains information about each of the image pixels and vice versa, one image pixel has contributions from all the *k-space* pixels, whereas one *k-space* pixel contributes to each of the image pixels differently [12]. This is obvious from the definition of the FT in Eq. 1.36.

$$S_{image,n} = \sum^{m} S_{kspace,m} \cdot e^{-2\pi i <\vec{k}_m, \vec{r}_n>} \tag{1.36}$$

The signal, in *k-space* or image depending on the lower index, is designated by $S$. The indexing to loop over all *k-space* points is $m$, whereas $n$ loops over all image points, $i$ symbolizes the imaginary unit, and the angle brackets represent a vector product of the two position vectors. This formula was used in this diploma thesis to implement the *spatial FT*, especially the *discrete FT* (*DFT*) for the *concentric ring* trajectory (CRT).

When acquiring images in a three-dimensional space, the signal must be localized in all three spatial directions (x, y, and z). In the z-direction, the signal is localized through *slice selection* and, in-plane, it is done via *frequency encoding* and *phase encoding*. The *spatial frequency space* has a time-related *spatial frequency* variable $k(t)$ assigned to it in Eq. 1.37 [6].

$$k(t) = \gamma \int_0^t G(t')dt' \tag{1.37}$$

The *k-space* variable is composed of two parts, $k_{frequency}$ and $k_{phase}$ describing coordinates on the two axes. With this definition, the measured signal is formalized in Eq. 1.38.

$$M_{xy}(k(t)) = \int_{-\infty}^{+\infty} M_0(r) \cdot e^{ik(t)r} dr \tag{1.38}$$

From Eq. 1.36, the FT relationship between the recorded signal, $M_{xy}(k(t))$, and the spatial spin density distribution, $M_0(r)$ (i.e., the image), is evident. The signals are stored in *k-space* as discrete data points sampled at a given time interval, $\Delta t$, called the *dwell time*. The calculation of FOV from the *dwell time* is stated in Eq. 1.39.

$$FOV = \frac{1}{\Delta t \cdot G} \tag{1.39}$$

In Eq. 1.39, $G$ is the gradient strength. Also, the *spectral bandwidth* can be determined from the *dwell time* as the inverse of it. Eq. 1.39 enables a determination of the FOV in the sampling direction, which is the *frequency encoding* direction for standard *phase encoding* [6].

With the *spatial frequency* variable, the movement in *k-space* according to applied gradients is described. In the middle of the *k-space*, both in-plane gradients are zero, and therefore, $k_{frequency}$ and $k_{phase}$ equal zero, which designates the coordinate origin. To start to fill the *k-space* from the bottom line, a maximum negative $G_{phase}$ is applied, displacing the *k-space* coordinate to the lowest line. Simultaneously, the de-phasing part of the $G_{frequency}$ is applied to select the right-most $k_{frequency}$ coordinate. In this manner, the desired position in *k-space* can be reached and the acquisition can start.

### 1.3.1 Slice selection

For *slice selection*, the gradient is applied during excitation, which causes a positional dependence of the Larmor frequencies over the object along the z-direction. The Larmor frequency, along the z-direction in a 0.2 $m$ long object, when a linear gradient is applied, spreads a range given in Eqs. 1.40.

$$\nu(z) = \frac{\gamma}{2\pi}B_0 + \frac{\gamma}{2\pi}zG_{slice} \tag{1.40}$$

$$\nu(-0.1m) = \frac{\gamma}{2\pi}B_0 - 0.1m\frac{\gamma}{2\pi}G_{slice}$$

over

$$\nu(0) = \frac{\gamma}{2\pi}B_0$$

to

$$\nu(0.1m) = \frac{\gamma}{2\pi}B_0 + 0.1m\frac{\gamma}{2\pi}G_{slice}$$

In this set of equations, $\nu(z)$ is the resonance frequency at position $z$ and $G_{slice}$ is the *slice selection* gradient given in units $T/m$. As spins at different z-positions undergo Larmor precession at different frequencies, only a portion of them can be excited with a selective RF pulse. Since only spins are excited whose Larmor frequency is contained in the frequency range of the RF pulse, it is possible to excite spins only in one slice. It is essential that the *slice selection* gradient is applied simultaneously with the selective excitation pulse [6].

The position of the slice is determined by the strength of the gradient and by the transmitter frequency of the RF pulse, $\nu_{RF}$. Eq. 1.40 is used to determine the needed $\nu_{RF}$ to excite just the one layer of the sample at a given distance from the isocenter. For this, it is also necessary to determine the gradient strength, $G_{slice}$, which is calculated as a quotient of the $RFbandwidth$ and desired $slicethickness$ [6], see Eq. 1.41.

$$G_{slice} = \frac{RFbandwidth}{slicethickness} \tag{1.41}$$

When the resulting gradient strength with its two factors is then kept constant, the position of the slice can be varied only by changing the transmitter frequency [6].

The slice thickness is also determined by the strength of the gradient and by the bandwidth of the RF pulse. The bandwidth is the frequency range transmitted in a given time with the unit Hz. It holds true that, with an increasing bandwidth of the RF pulse, the minimal thickness of the slice increases and vice versa [13]. The thinner the slice (i.e., better resolution in the z-direction) the stronger the gradient must be, because, with a steeper gradient, the resonance frequencies across the sample span a greater range according to Eq. 1.40, and therefore make it possible to select a thinner slice [6].

As MRI is a complex and very rich technique, it is also possible to select multiple slices at one time with some additional precautionary steps. This is one of many possibilities for accelerated or fast imaging because more slices are measured during one $TR$. We will not go into more detail on this in this diploma thesis as it is not the focus of this work. It is only interesting for our considerations that the slice profiles are not perfectly rectangular, meaning that there is a partial excitation and signal contamination from the adjacent slices [6].

### 1.3.2 Frequency encoding

The gradient for *frequency encoding*, $G_{freq}$, is played out during the signal acquisition. Thus, the spins attain different Larmor frequencies at different spatial positions along the direction of $G_{freq}$. The variation of the Larmor frequency is linear for linear gradients, thereby holding spatial information. This process, as are all other spatial encoding processes, is governed by Eq. 1.40. To localize the signal, the $G_{freq}$ must be started immediately after the excitation and end together with the acquisition. In this manner, the different Larmor frequencies along the frequency encoding direction (i.e., x or y) are recorded. To obtain the spatial information from the detected signal, an FT along this direction is performed and gives the spin density in the direction of the applied gradient. An important parameter for measurement is the *acquisition dwell time*, which must be

chosen according to the Nyquist sampling theorem and determines the range of detected frequencies, the *readout bandwidth*. The aim of this is to avoid *spatial aliasing* (i.e., fold-in of peripheral image stripes in the under-sampled direction).

The Nyquist theorem states that all frequencies contained in the signal are sampled accurately when the highest frequency is sampled at least twice per period (i.e., $2\pi$). As the *dwell time* determines the *spectral bandwidth*, the Nyquist theorem postulates that the frequency of the signal is measured correctly when it lies inside the *spectral bandwidth*. Sampled signals containing higher frequencies will not enable detection of those frequencies and the frequency analysis (i.e., the FT) will be compromised, the special term for this is *aliasing*. The bandwidth will be discussed in more detail in the Subsection on *Spectral fitting and quantification*. Dividing the frequency range, the *spectral bandwidth*, by the gradient strength results in a position range, the FOV [6].

As spatial assignment of the signal in only one of the in-plane directions is insufficient, an additional spatial encoding is necessary to obtain an image of a 3D object. There are more possibilities to achieve this. One possibility is to also use a *frequency encoding* gradient, but one in the other in-plane direction than the first *frequency encoding* gradient. Then, to reconstruct the spatial information from this acquisition technique, a filtered back-projection is used. But, the majority of MRI acquisitions use a different encoding approach [6].

### 1.3.3 Phase encoding

The standard approach to encode the other spatial in-plane direction is *phase encoding*. In this approach, the phase of the signal is influenced by the encoding gradient. The gradient is applied only before the acquisition period, and therefore, does not influence the recorded resonance frequencies, only the phases. The resonance frequencies during the $G_{phase}$ are indeed influenced, but, as this gradient is turned off before the acquisition period begins, the previous changes in the frequency manifest themselves in the signal as phase shifts, which are recorded and contain the spatial information of the other spatial in-plane direction. The concept of *phase encoding*, together with *slice selection* and *frequency encoding*, is depicted in Fig. 1.17, for the example of a gradient-echo sequence.

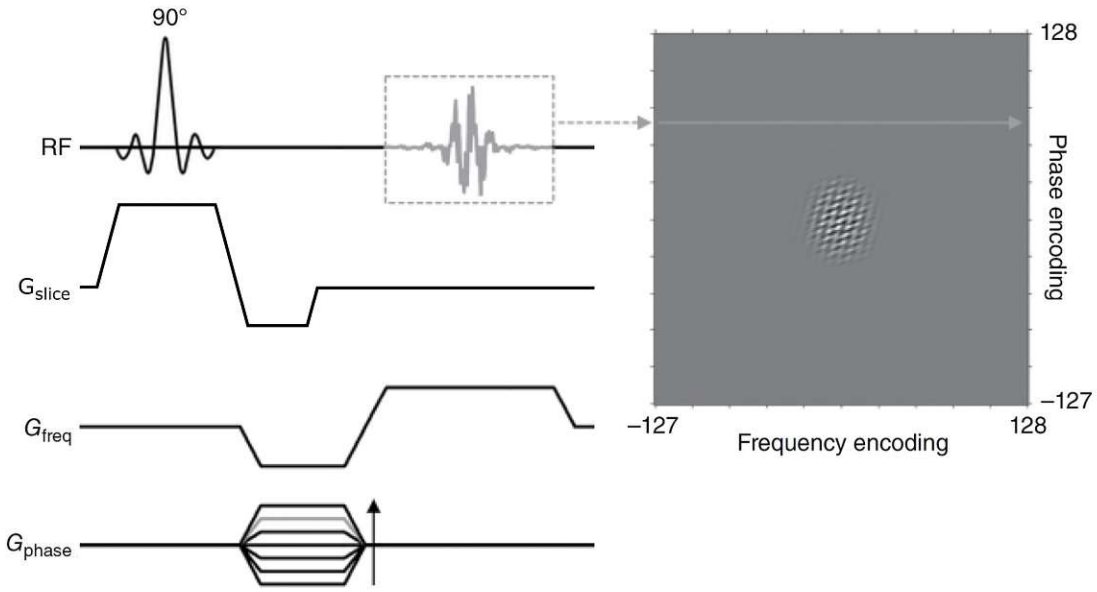


Figure 1.17: Display of the RF pulse, *slice selection*, *frequency* and *phase encoding* gradients of a gradient-echo sequence also showing how the measured data is stored in *k-space*. Modified from [6].

In a gradient-echo sequence, the first gradient pulse is mostly half the area of the second one. The goal of the first gradient pulse is to prepare the spatial encoding of the in-plane magnetization via generation of a *phase shift* [6].

But, in the presence of another $G_{phase}$, the part of the reverse-signed frequency gradient no longer cancels out. The additional phase generates an indirect frequency. The remaining accumulated frequency carries independent information of the one coded by the *frequency-encoding* gradient. The phase gradient, in increments of distinct *phase-encoding* steps, is stored in one horizontal line of the *k-space* [6].

To gather the complete information for all points on the phase axis, the amplitude of the phase gradient is gradually changed. One *phase encoding* line in the *k-space* is traditionally sampled for one excitation, one $TR$. In terms of the acquired phase of the spins, it holds true that the more phase acquired, the farther the spins are from the isocenter in the direction of $G_{phase}$ and vice versa. The signal that acquired zero phase is found in the *k-space* center. The acquired phase with respect to the applied gradient $G_{phase}$ represents a frequency that is directly proportional to the spatial distribution of the spins. This is the same encoding property as was described by the *frequency encoding*, thus, the same procedure is performed to arrive at the image, namely the FT. The FT is performed along both in-plane directions.

Traditional MRI sequences combine the three presented techniques (i.e., *slice selection*, *frequency*, and *phase encoding*) to localize the signal in all spatial dimensions, thereby enabling reconstruction of images upon spatial FT in which the *slice selection* gradients are applied simultaneously with the RF pulses. All of the principles of spatial localization in MRI are the same for MR Spectroscopy (MRS) and MR Spectroscopic Imaging (MRSI) [6].

### 1.3.4 Magnetic Resonance Spectroscopy

The name spectroscopy in MRS comes from the word spectrum. As explained above, spectrum is the Fourier-transformed time signal that displays the resonance frequencies contained in the sample as well as their amplitudes. The unit on the horizontal axis in a spectrum is given in parts per million (ppm) and on the vertical axis is the signal amplitude of a given frequency. Why frequencies in Hz are converted into units ppm will be explained in a later Subsection on *Spectral fitting and quantification*. For spectra, it is characteristic that the areas below the peaks are directly proportional to the number of nuclei in a specific nuclear environment [1], with the condition that only those frequencies sampled according to the Nyquist criterion are represented correctly.

As MRI, MRS also relies on the principles of NMR to produce signals. The signals are recorded the same way as in MRI. As the brain is the anatomical region of interest in numerous clinical MRS studies, and also the central piece of this diploma thesis, the CRT sequence, and the reconstruction thereof, is designated for brain metabolic imaging. Magnetic Resonance Spectroscopy is, just like MRI, a non-invasive technique. In the clinical routine, MRS should not be used alone, but always together with MRI as many diagnoses can be made just on the basis of images (even without metabolic information).

The informational content of MRS is not based solely on the proton density, as such, but is more specific in terms of recording different proton resonances depending on their chemical environment. Therefore, with MRS, the differentiation of specific molecules, and the protons by which they are bound, is possible. Because of that, MRS has found wide acceptance in clinical diagnostics, as it enables more precise characterization of tissues and various changes thereof.

What makes this method even more powerful is the fact that it can be a quantitative technique that provides information about the concentrations of metabolites, for example, in the brain. This picture of the metabolites helps to identify disorders. Some of the most prominent metabolites, their abbreviations, positions in the spectrum, as well as what they indicate, are summarized in Table 1.2 [14].

| Metabolite | Abbreviation | Resonance [ppm] | Indicator for |
|---|---|---|---|
| N-acetyl aspartate | NAA | 2.02 | neurons |
| Choline | Cho | 3.2 | membranes |
| Creatine | Cr | 3.0 | energy |
| Lactate | Lac | 1.33 (doublet) | oxygenation |
| Lipids | - | 0.9 - 1.5 | necrosis |
| Myo-inositol | mI | 3.56 | osmolyte |
| Glutamine, Glutamate | Glu, Gln | 2.2 - 2.4 | ammonia |

Table 1.2: Overview of some of the important MRS metabolites with their abbreviations, resonances, and diagnostic use [14].

Depending on the metabolites of interest, two important parameters, which were already introduced in a previous section, must be set properly in all MRS sequences: TE and TR. Short echo times should be selected for spectroscopy of short-T2 metabolites ,such as mI, Glu, or Gln. Short means, in this context, in a range from 20 to 35 ms. On the other hand, metabolites in the upper part of Table 1.2 (i.e., NAA, Cho, Cr, and Lac) must be excited with longer TEs between 135 and 270 ms [14]. Furthermore, the amplitude of the respective resonances in the spectrum is not constant for different TEs and TRs as depicted in Fig. 1.18.

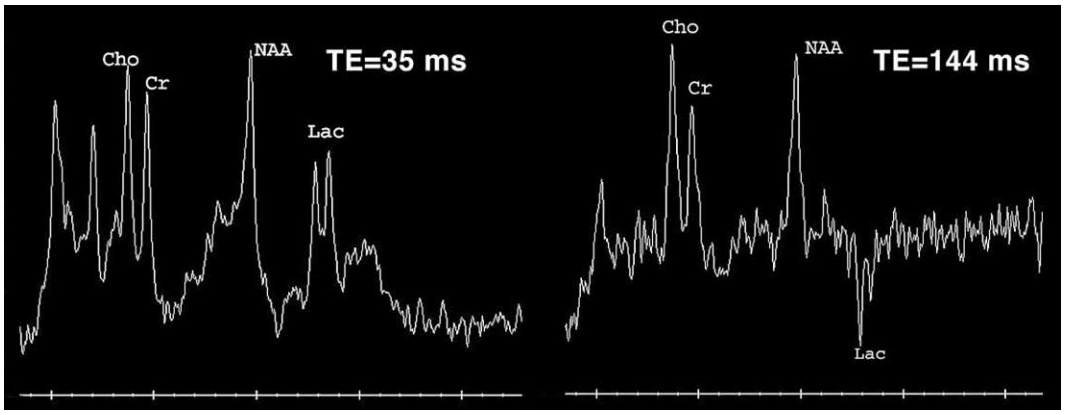


Figure 1.18: Two MR spectra of the major metabolites showing the differences in amplitude and form of the peaks between excitation with short (left) and long TE (right) [14].

The amplitude of the metabolites also depends on the localization sequence used. Here are a few details on the major MRS metabolites given in Table 1.2.

The most pronounced peak of all is that of NAA. When a decrease in its concentration is detected in the spectrum, it points to neuronal damage [14].

The peak of Cho actually comprises not only choline, but also glycerophosphocholine, phosphatidylcholine, and phosphocholine. Those are some of the building blocks of cell membranes and thus, Cho is a good indicator of membrane metabolism [14].

The creatine peak as well contains the concentrations of another metabolite. In this case, it is phosphocreatine. Both metabolites present a marker for energy metabolism of the brain [14].

There are some unusual qualities in the appearance of Lac in the spectrum. As can be seen in Fig. 1.18, it is a so-called doublet because it consists of two peaks. This doublet stands for short TEs, as well as for long TEs, above the baseline just as all the other peaks, but for medium TEs (e.g., 144 ms) the Lac resonances are below the baseline. The lactate even undergoes J-evolution of the signal. In physiological conditions, this metabolite is below the level of detectability. It is only pronounced when processes with an insufficient oxygen supply are initiated, such as anaerobic glycolysis (i.e., breakdown of sugars) [14].

When there is an increase of lipids observed in the spectrum, it may indicate necrotic, meaning dead, areas of tumours. It is of great importance to place voxels for lipid observation carefully, e.g., not directly next to the base of the skull, where there are many physiological lipids that would contaminate the signal [14].

The marker for osmolytes and astrocytes is mI. Osmolytes have a considerable impact on cellular biochemical processes, such as protein folding or protein-ligand interactions [15]. This metabolite is best investigated with short TEs. When this marker increases, it may point to one of the following diagnoses: Alzheimer's disease; frontotemporal dementia; or HIV infection [14].

The metabolites Glu and Gln appear as multiple resonances in the range stated in Table 1.2 for short TEs. A rise in the concentration of Glu and Gln may be a sign of hyperammonemia (i.e., metabolic condition characterized by high levels of $NH_3$ [16]) in medical conditions such as hepatic encephalopathy [14].

The spectroscopy technique is not limited to signal acquisition from hydrogen, but can be also applied to other nuclei. Those are called ex-nuclei and some are $^{13}C$, $^{15}N$, $^{19}F$, $^{23}Na$, and $^{31}P$. The sequence must be customized according to the nucleus of interest. Mainly, the RF transmit frequency has to be adjusted to the Larmor frequency, $\omega_0$, of the nucleus. The ex-nuclei's Larmor frequencies can be calculated according to Eq. 1.10 by inserting the corresponding gyromagnetic ratio $\gamma$ given in Table 1.1 and the field strength of the main magnetic field $B_0$. As this diploma thesis is about hydrogen imaging and spectroscopy, there will be no further considerations about ex-nuclei techniques. One of the advantages of hydrogen MRS over other nuclei is its high absolute sensitivity to magnetic manipulation, which enables a better spatial resolution [14].

The main difference between MRS and MRI is that MRI does not encode the spectral dimension. MRI utilizes simply the sum signal per voxel, including water. In MRS, the water signal must be suppressed because water in in vivo experiments has concentrations approximately four orders of magnitude higher than the metabolites of interest. This is achieved with special pulses. According to the characteristics of the volume from which the signal is recorded, MRS techniques can be divided into two major categories: single-voxel spectroscopy (SVS) and Magnetic Resonance Spectroscopic Imaging (MRSI) [14]. A special subcategory of MRSI is chemical shift imaging (CSI), which relies only on *phase encoding*.

As indicated in the name, SVS is the most basic MRS technique, as only one voxel (i.e., three dimensional or volume pixel) is excited and provides the signal. Because SVS covers only a small volume of tissue, it is very time-efficient. Another advantage of SVS is that it immediately provides signal intensity values unlike the other MRS categories, which need some signal processing to arrive at the intensities [14].

When we consider MRS measurement sequences, especially for proton SVS, we realize that the hardware requirements are the same as for standard MRI. Thus, MRS can be performed on any clinical scanner. Special sequences are only necessary to capture the localized spectra. We will go through two basic sequences. First, the point-resolved spectroscopy sequence (PRESS), which relies on a double spin-echo. Second, the stimulated echo acquisition mode (STEAM), which excites the echo by a total of three pulses. Both of those techniques focus on the informational content of the echo rather than that of the FID. But, the echo is acquired in a more FID-like manner, as only the second half of the echo, starting with the maximum and lasting throughout the decay-duration, is recorded. To achieve sufficient SNR, the main magnetic field strength of the scanner should be 1.5 T or above, which results in a greater energy gap between the nuclear energy levels and leads to a better spectral resolution (i.e., improved chemical specificity) [17].

It is interesting to pay attention to one of the first methods to study the chemical shifts of organic compounds using MR. This early technique excited the sample with different frequencies in numerous experiments and recorded the induced signals. Without further processing, the obtained time signals were studied to extract chemical shift information. MRS studies were performed in this time-consuming way until the revolutionary discovery of Ernst and Anderson. They devised the idea to apply the FT to the time signal coming from the excitation with a single frequency and arrive at the spectrum. The application of FT made it unnecessary to sweep through a range of different resonance frequencies because all the signal components could be analyzed after only one

experiment [18].

For modern sequences, it is essential to be able to select an arbitrary volume of interest (VOI) and obtain signal only from there as precisely as possible. This may be done using a surface coil that excites only the superficial part of the sample, and thus, the signal comes only from near the coil, which, of course, enables the selection of a VOI only from this limited region. To place the voxel anywhere in the sample, another type of coil must be used that enables excitation of the whole object, the volume coil. Then, the selection of the voxel location can be realized through linear field gradients combined with RF pulses of a specific bandwidth, exactly the same principle as for *slice selection* in MRI. Another possibility for spatial localization would be to saturate the longitudinal magnetization of nuclei everywhere else but in the VOI. However, there is less signal contamination when exciting only the desired voxel [17]. The two sequences used for the volume-selective MRS were mentioned earlier and will be described in more detail now.

There are three RF pulses in the PRESS sequence, depicted in Fig. 1.19. The first pulse is a 90° degree pulse applied simultaneously with a *slice selection* gradient, $G_z$. The second and the third pulse are both 180° pulses. One of them serves the purpose of localization in the x-direction accompanied by $G_x$, and the other along the y-axis, localized via a gradient along the same direction. The principle of the gradual volume selection is visualized on the right side of Fig. 1.19. Both refocusing pulses are applied in the *frequency encoding* mode [17].

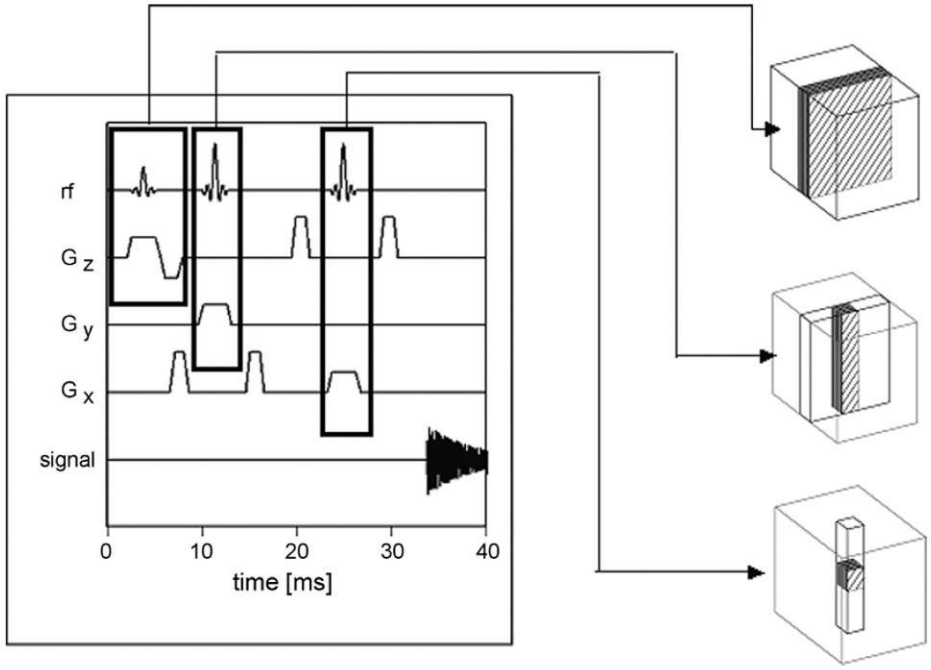


Figure 1.19: The RF pulses and magnetic field gradients of the PRESS sequence visualizing the gradual selection of the VOI [17].

The working principle of spatial determination is the following. First, only one slice is excited due to the $G_z$ gradient turning the magnetization vector into the xy-plane. The negative part of $G_z$ is needed to balance the phase shifts caused by the excitation pulse. Before the first refocusing pulse, the magnetic field gradient in the x-direction is prepared, thus, refocusing spins only in a selected column on a desired x-position, see Fig. 1.19. This pulse is applied at time TE1/2 so that an echo peaks at TE1. This echo is not sampled, as the localization process is not yet finished. The signal is recorded only from the second echo, which is created by the second refocusing pulse at time TE1 + TE2/2 so that the second echo has its maximum at TE1 + TE2. The second

echo is the information carrier of this sequence in which only the second half is evaluated. This differentiates it from MRI where the whole echo is used. The duration of the utilized echo in MRS is usually on the order of hundreds of ms. This is necessary to identify minor frequency differences (i.e., chemical shifts or metabolites). The desired effect of both 180° pulses is achieved because they refocus only spins at the intersection of the planes at the selected x- and y-positions, thus, generating signal that comes only from the VOI. The dimensions of the voxel are determined by the same parameters as the slice thickness discussed in the previous section (i.e., gradient strength and RF pulse bandwidth) [17].

The other localization method for SVS is the STEAM sequence, where there are also three RF pulses, but this time, all of them are ideally 90° pulses, illustrated in Fig. 1.20. The localization is implemented according to the same principles as in the PRESS sequence. The three excitations following each other cause a stimulated echo resulting in a more intense signal.

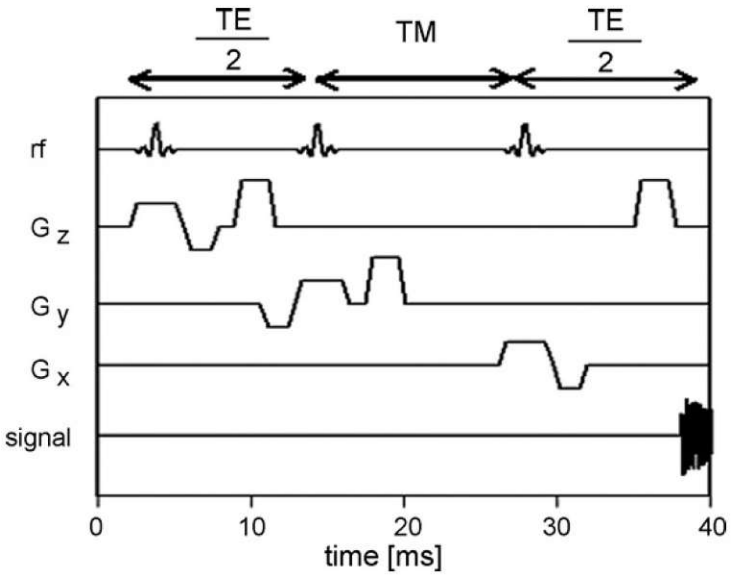


Figure 1.20: STEAM pulse sequence displaying the temporal arrangement of the excitations and magnetic field gradients [17].

The first z-slice excitation rotates the net magnetization into the xy-plane. After that, the spins start to de-phase according to the T2 relaxation processes. At TE/2 (see Fig. 1.20), the second excitation is applied, which rotates the previously created transverse magnetization into the zy-plane, in this case. The direction of the z-component is opposite $\vec{B}_0$. The still remaining transverse magnetization experiences further de-phasing due to relaxation (mainly T2 and T2*) and spoiler gradients until it is completely diminished. Only the anti-parallel z magnetization remains because it is influenced solely by the slower T1 processes. After TE/2 + TM (TM is on the order of units of ms, i.e., very short) the third excitation is applied together with $G_x$, flipping the remaining magnetization back into the xy-plane, where it was after the first excitation. After this, re-phasing takes place, and, at TE + TM, the spins gain phase coherence that manifests as an echo in the receiver coil. A precise localization in this sequence would not be possible without the spoiler gradients, which, again, work against signal contamination. What is different from the spoilers in the PRESS sequence is that, in the STEAM sequence, there is also a pre-phasing gradient (in the $G_z$ line right before the second excitation), the effect of which comes together with the last spoiler gradient in the same direction, ensuring that the magnetization of interest is not also de-phased. Again, to avoid signals from outside the VOI, there is yet another spoiler gradient in the $G_y$ line right after the slice selection in the corresponding plane. An advantage of the STEAM sequence is that a short TE can be chosen than that in PRESS, making it the method of choice for short TE

measurements [17].

PRESS as well as STEAM are subject to various artefacts, and thus, further measures inside the sequence are taken to achieve desirable quality of the spectra. [17]. General remarks on the most important kinds of artefacts will be reviewed in the Subsection on *Imaging and spectral artefacts*. Here, only considerations specific to those sequences are briefly stated. In the PRESS sequence, there may be signal contamination from outside the VOI. This is because on the borders of the selected slice the second 180° pulse may have an effect as if there was a 90° pulse applied at the border region. Therefore, the signal from those additional parts of the sample may also be recorded.

The solution for this is proper gradient spoiling, which causes the nuclei outside the VOI to de-phase more rapidly, and therefore, prevents the nuclei from contributing to the signal. Those gradients were shown in Fig. 1.19, with the two high and narrow gradients in the $G_z$ line applied before and after the second refocusing pulse. The second pair of the spoiler gradients is in the $G_x$ line before and after the first 180° pulse.

### 1.3.5 Magnetic Resonance Spectroscopic Imaging

In addition to the single-voxel techniques, there is chemical shift imaging, which collects data not only from one but also from several voxels. Those voxels are selected according to diagnostic interest. The measured voxels in CSI can lie either in a 1D column, a 2D plane, or can cover a whole 3D volume. In MRSI, the worlds of MRI and MRS come together. The form of *k-space* in MRSI is the combination of both. Coming from MRI, *k-space* is filled with discrete points. In MRS, a time signal is assigned to each *k-space* point, which is later converted into a spectrum. Thus, in the MRSI data matrix, there is a time signal measured at each position. This can be imagined as an additional time or spectral dimension.

This volumetric technique records a larger tissue region than SVS so the acquisition accordingly takes longer. In contrast to SVS in CSI, some post-processing of the data is required to arrive at the signal intensities and metabolite concentrations. Metabolite ratios are also calculated. Those tasks are usually performed automatically by special algorithms, explained in more detail in the Subsection on *Spectral fitting and quantification* [14].

MRSI can be localized in two major ways - either by *phase* or by *spatial-spectral encoding*. To induce the signal, any method can be utilized, e.g., the PRESS sequence described in the previous section [19].

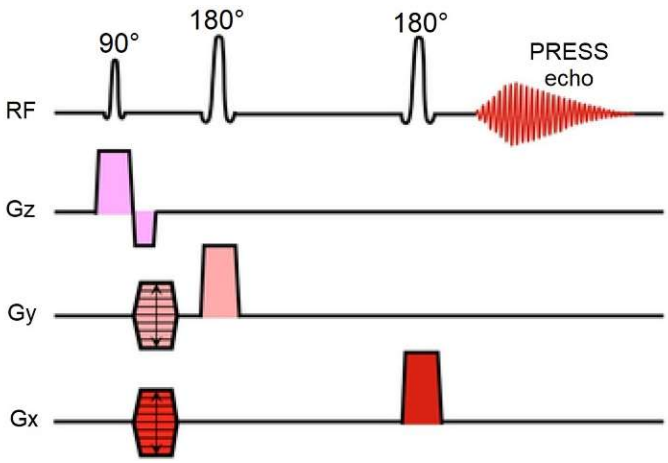


Figure 1.21: Excitation and localization utilizing the PRESS sequence with additional *phase encoding* in the x- and y-direction, rendering it a single-slice CSI sequence [19].

The modified PRESS sequence in a CSI method samples spectra not only from a single-voxel, but also from a whole slice that is built up by many voxels. The various *phase encoding* steps are

indicated in Fig. 1.21 by the stepped gradients, $G_x$ and $G_y$, both of which are placed before the first refocusing pulse. This sequence can be used, for example, for proton brain MRSI [19]. The spoiler and saturation gradients are left out on purpose in Fig. 1.21, but, when running this sequence, they are in place as was explained in the SVS PRESS.

An advantage of CSI over SVS is that a larger area can be measured and at the same time, a more precise examination is provided because of the higher spatial resolution. But, there are also disadvantages to CSI, such as greater time demands, less field homogeneity over the area of interest, lower SNR, lower quality of the resulting localized spectra caused, e.g., by higher signal contamination [19].

The most complex and most time consuming acquisition method is MRSI with three-dimensional coverage. There, a continuous volume is divided into individual voxels enabling assessment of the chemical composition of each of them, and thus, making it possible to create *metabolic maps* in 2D or in 3D. More *metabolic maps* will be discussed later in the *quantification* subsection. Volumetric MRSI requires a considerably larger amount of time for the acquisition compared to SVS and 2D CSI, as well as for the post-processing [14]. Regarding the longer experiment times it is interesting to mention the typical TR value, which can be two seconds, causing the total measurement time to expand beyond acceptable limits for coverage of larger volumes when using standard methods, such as CSI. That is the reason for a whole array of MRSI acceleration methods termed fast MRSI.

### 1.3.6 Spatial-spectral encoded MRSI

The *spatial-spectral encoded* (SSE) sequences achieve shorter scanning times, as the encoding of the spatial and spectral dimension happens simultaneously during one TR with the help of varying readout gradients. Compared to *phase encoding*, the SSE MRSI leads to better spatial resolution as well as shorter scan times [20].

In SSE, it is possible to sample the *k-space* on various trajectories, virtually with any arbitrary trajectory that is ideally intrinsically closed. The standard trajectory is *Cartesian* in which equidistant parallel lines are measured, as is the case in echo-planar spectroscopic imaging (EPSI). Some of the more exotic ones are rosettes or *concentric ring trajectory* (CRT). Each of these trajectories has some advantages and disadvantages. The quality of a given trajectory can be assessed based on its *k-space* density and the necessary gradient re-winders. For quality assessment, the measured *k-space* density is compared to the desired density, which can be, for example, uniform (i.e., everywhere the same), or in the form of a Hamming filter (i.e., high in the middle and decaying outward). The gradient re-winders are tracks in the *k-space* that create a closed path so that many repetitions of the same trajectory can be performed to acquire the time signal necessary for spectra. More details will be discussed in a later Section on *Different k-space trajectories of SSE*.

To fill all needed *k-space* points, the sequence-specific trajectory must be applied many times in such a way that the coverage is Nyquist-conformed. For example, in EPSI this means that multiple lines have to be recorded. For CRT, there have to be smaller circles in the middle of *k-space* and subsequent circles with an increasing diameter until the edges of the desired *k-space* are reached [21]. In this Section, the EPSI sequence will be discussed. In a later section, the CRT sequence, developed by my colleagues and mentors Bernhard Strasser and Lukas Hingerl, and which forms the core of this diploma thesis, will be discussed in detail. The current form of the offline reconstruction for CRT was also created by these colleagues.

All of the points on one trajectory cannot be measured at once, but the trajectory has to be filled gradually by varying the gradients (e.g., in a sine wave-line manner for CRT). The data is sampled in such a way that, at the end of the excitation, the gradients reach a desired starting point in *k-space* and the FID is sampled after the excitation. The varying gradients are played out, performing a *k-space* sampling along the trajectory. To sample the time signal at each point in *k-space* according to the Nyquist criterion, a minimal time difference must be abided between each two successive time signal data points on each position in *k-space*. In an ideal situation, the sampling of the given trajectory would occur in an instant, measuring all of the first time signal points and would be prepared again at the start of the trajectory to start sampling all the second FID data points

within the desired *spectral dwell time*. A minimum *spectral dwell time* is necessary for a Nyquist-sampled measurement of all frequencies contained in the signal. But, a circumnavigation of the trajectory cannot happen in an instant, but is rather performed gradually. Thus, it is sometimes the case that, by larger or longer *k-space* trajectories, the gradient limitations, in particular the slew rate, do not allow repeating the trajectory fast enough to adhere to the minimum *spectral dwell time*. The solution for this is so-called *temporal interleaving* in which not each, but only each second or third data point of the signal is measured in one TR. Thus, the gradients can take two or three *spectral dwell times* to complete one circumnavigation. Each *temporal interleave* demands its own excitation, and therefore, one TR. The measurement of the correct time points is achieved through a time-shifted sampling. When the measurement is complete, one of the first data-processing steps is to put the two or three *temporal interleaves* together so that the consecutive time points are properly ordered. But, as the signals of the different *temporal interleaves* do not come from the same excitation, their joining can cause artefacts as there are always imperfections, e.g., in the RF pulse or movement. To keep those artefacts within acceptable limits, there is usually a maximum of three *temporal interleaves*[21]. More on this topic will be discussed in the respective *data-processing* section.

The first *spatial-spectral encoding* sequence that will be introduced here is EPSI. The core of the EPSI sequence is the oscillating readout gradient, resembling the concept of *frequency encoding*. The sampling trajectory in this sequence is a rectilinear one. Although the SNR per unit volume and unit time is comparable between EPSI and conventional MRSI, there is a major improvement in spatial resolution and SNR for various metabolites [20].

First, in the EPSI sequence, one spatial slice is excited conventionally as described above. Then, one horizontal line is selected by the y-gradient. The line is then measured many times to sample the time signal at all its points. This is done in a back and forth fashion many hundreds of times, achieved by the specific form of $G_x$, depicted in Fig. 1.22. This technique is referred to as *spatial-spectral encoding* because the applied gradients sample *k-space* positions and also the FID time points at the same time [21].

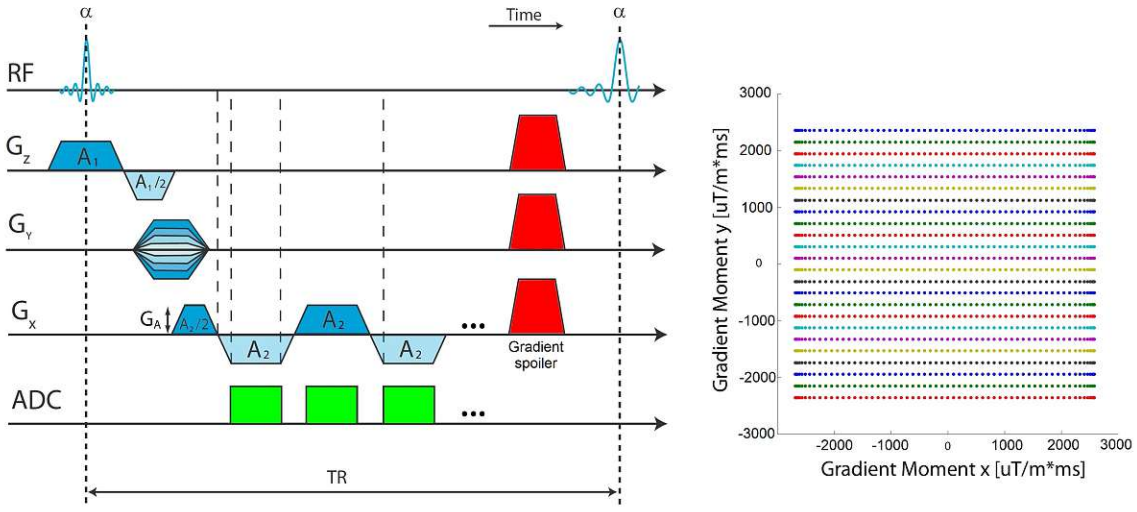


Figure 1.22: The sequence diagram of EPSI utilizing the principles of *slice selection* via $G_z$ and RF, *phase encoding* via $G_y$, and *spatial-spectral encoding* via $G_x$ (left). The *k-space* trajectories for complete sampling (right) [21].

The sampling occurs only when the amplitude of $G_x$ is constant, symbolized by the green boxes. When one line has been completely measured, the rest of the magnetization is spoiled and the next line is selected in the traditional *phase encoding* mode (i.e., by stepwise ramping up the strength

of $G_y$) [21]. The area below the x- and y-gradients is termed the *gradient moment* (GM). There is a necessity for a certain GM to obtain the desired FOV with a number of voxels, $N$, in that dimension (e.g., in the x-direction), which is determined by Eq. 1.42.

$$GM_x = \gamma \frac{N_x}{FOV} \tag{1.42}$$

The conversion of GM into *k-space* coordinates and other conversions will be discussed in a later section.

### 1.3.7 Imaging and spectral artefacts

Artefacts in images and in spectra are an important topic for MR physicists and diagnostics. An artefact can be defined as a feature in the image or spectrum that is not present in the original object [22]. As there are many different artefacts, we will consider only the most prominent of them and introduce their causes and elimination strategies in two tables dedicated to imaging and spectral artefacts separately, starting with imaging, in Table 1.3.

| Artefact | Cause | Elimination strategies in MRI |
|---|---|---|
| Motion | Patient: physiological and arbitrary movements | Patient immobilization, gating, shorter sample dimension *phase encoded* |
| Flow | Patient: blood brings spins aligned with $\vec{B}_0$ into the VOI | Increasing the TE |
| Metal | Patient: different magnetic susceptibility | Sample orientation, small voxels, fast imaging sequences, higher sampling frequency |
| Chemical shift | Signal processing: interface of different chemical environments (e.g., water and fat) | Swap *phase* and *frequency encoding* gradients, compare the shifts |
| Partial volume | Signal processing: signal from small objects is averaged over the whole voxel | Smaller voxel size |
| Wrap around | Signal processing: FOV is smaller than imaged object, under-sampling | Filter out frequencies over the Nyquist limit in *frequency encoding*, more *phase encoding* steps |
| Gibbs ringing | Signal processing: under-sampling of high spatial frequencies | Increase matrix size, *k-space* filtering |
| $\vec{B}_0$ inhomogeneity | Inhomogeneous resonance frequency distribution in sample | Shimming |
| Gradient inhomogeneity | Imperfections in the gradient-generated fields | *Phase encoding* gradient along shorter object dimension, FOV reduction |

Table 1.3: Overview of some of the most prominent MRI artefacts, their causes and elimination strategies [22].

Motion artefacts appear as ghost images or diffuse image noise. These artefacts affect mainly the *phase encoding* direction, as physiological movements (e.g., heart beat) have a similar frequency (i.e., seconds up to minutes until all horizontal lines in *k-space* are collected in a conventional *Cartesian* sampling) and do not touch the *frequency encoding* direction because the time scale there is much shorter, on the order of milliseconds. Thus, periodic physiological movements cause ghost images and non-repetitive movements lead to diffuse image noise [22].

The flow artefact has the form of an altered intravascular signal, and erroneous localization of the signal source can also occur [22].

The presence of metals inside the sample causes distortions of $\vec{B}_0$ by local magnetic fields. This is because of the significant difference in magnetic susceptibility at the interface between metal and tissue. Metal artefacts may result in regional signal loss, high signal in the periphery, and image distortions [22]. In addition, spectral resolution is reduced (i.e., chemical specificity is lowered) by this artefact.

The chemical shift artefact is especially pronounced at high field-strengths, where the fat image is shifted with respect to the water image, which also affects all other frequency-shifted metabolites alike. The image shift between the different signal sources manifests because of the different resonance frequencies (fat has a lower frequency). This artefact affects the *slice selection* direction due to the frequency-selective excitation, which cannot be perfect when the two most abundant substances (i.e., water and fat) precess at different frequencies [23].

Partial volume artefacts are caused by other voxels contributing their signal to the actual target voxel. This inevitably leads to a loss of detail in images. The effect of this artefact can be mitigated by decreasing the nominal voxel dimensions [22].

When outer parts of the image are mirrored by the other side of the image, that is a typical sign of the wrap-around artefact, which emerges when the FOV selected is too narrow. The regions that would not fit into it are displayed in the wrong place [22].

The Gibbs ringing artefact is caused by under-sampling of high-detail features in images (i.e., high spatial frequencies at the *k-space* periphery). The typical manifestation is regularly spaced bright and dark bands slowly fading in brightness with distance [22].

To compensate for $\vec{B}_0$ inhomogeneities, a procedure termed *shimming* is used, which will be described in a later Section on *Acquisition features in MRSI*. The external field inhomogeneity brings about distortions in the spatial localization as well as in signal intensity [22].

Unwanted changes in the voxel width result from gradient field imperfections in the *phase encoding* direction and lead to spatial distortions. The inhomogeneity of gradient fields might be inflicted by malfunctioning coils or a faulty current source. This artefact can be reduced by decreasing the gradient strength [22].

In addition to the artefacts affecting MRI, there are the MRS artefacts. An overview of those is presented in Table 1.4.

| Artefact | Cause | Elimination strategies in MRS |
|---|---|---|
| Poor shimming | Field inhomogeneity leading to different resonance frequencies | Manual shimming after the automated shimming |
| Incomplete water suppression | Water suppression fails | Three CHESS pulses: flip the water magnetization into the xy-plane and spoil it |
| *Chemical shift displacement* | Same as *chemical shift* in MRI artefacts | Stronger gradients, RF pulses with higher bandwidth |
| Multi-voxel spectral contamination | Signals are assigned to the wrong voxels | Increase the matrix size, digital filtering |

Table 1.4: Overview of MRS artefacts, their causes and possible elimination strategies [24].

Artefacts in the spectrum arise from poor $\vec{B}_0$ shimming. They appear as overly broad, short, and not sharply separated spectral lines.

When the water suppression is inadequate, the metabolite signals are covered in background noise coming from the remaining water signal as the water concentration is much higher than that of the metabolites in living samples [24]. The resonance frequency of water has its peak at 4.7 ppm, so it does not lie directly over the metabolites as they are spread between 1 and 4 ppm. But, as the concentration of water, and thus, also its magnetization, is higher than the metabolite concentrations, the base of the water peak can be so wide that it reaches into the metabolite part of the spectrum. The solution for this issue is the proper use of water suppression pulses. This can be achieved, for example, by the use of *chemical shift selective* (CHESS) pulses. But, as the water suppression is mostly not perfect, the main goal in dealing with the water signal is to press its wide base below the level of noise. Furthermore, to receive high-quality spectra, the widths of the peaks (i.e., line widths) should be small. The line widths are determined by the T2 processes together with the $B_0$ field homogeneity inside the VOI. To support the field homogeneity, *shimming* is used [17]. Also, by magnetic field maps obtained with MRI measurements, the $B_0$ and $B_1$ fields can be created in a more homogeneous manner, thus increasing the quality of spectroscopy data [6].

The *chemical shift displacement* artefact is the biggest concern in SVS, whereas the *multi-voxel spectral contamination* artefact is the biggest problem in MRSI. The *chemical shift displacement* artefact is caused by the different resonance frequencies of protons according to their chemical environment during the application of the localization gradient and the frequency-selective RF pulse. Based on the frequency difference, $\Delta\omega$, a spatial displacement, $\Delta x$, is created [17].

$$\Delta x = \frac{\Delta\omega G}{\gamma} \tag{1.43}$$

The spatial shift is a function of the frequency difference and the strength of the applied gradient. This displacement applies during the spatial determination in all directions, thereby resulting in a diagonal position shift of the VOI. To choose the frequency of RF pulses in MRS, usually the resonance frequency of NAA or another metabolite, leads to no *chemical shift displacement* of the chosen metabolite. For an exact localization of the other metabolites, all would have to be treated individually, as the displacement is different for each. A solution to minimize the effect of this artefact is to increase the gradient strength, which means also that the selective RF pulses need to have a larger bandwidth [17].

Regarding the *multi-voxel spectral contamination* artefact, the quality of the elimination of signal coming from outside the VOI can be assessed by the strength of the contaminating fat signals. The signal contamination from outside the VOI is caused by the not-quite rectangular shape of the slice profile. It is hardly possible to create perfectly sharp excitation edges, as the duration of the RF pulse is limited. This results in a decreased excitation near the borders of the slice and behind the borders in an additional minor excitation, as symbolized by the shape of the RF pulse (e.g., in Figs. 1.21 and 1.22). There may also be another small excitation further away from the slice.

Numerous MRS studies have been conducted on the brain, as this organ is easily accessible to this technique and the number of motion artefacts is limited [14]. This is also because there are no macroscopic physiological movements in the brain as there are, for example, in the abdominal area.

There are also some challenges connected with MRS. One of these is the pronounced difference of the magnetic susceptibilities between brain, fat, bone, and air. This makes it difficult to perform MRS in tissue adjacent to regions with great susceptibility variations because magnetic field inhomogeneity arises and artefacts follow. Spectra are very sensitive to this, which makes it hard to obtain them in the vicinity of, for example, the skull base or paranasal sinuses. To overcome this challenge, single-voxel spectroscopy is used in those regions because it performs more robustly with regard to magnetic susceptibility artefacts. This is because it is easier to focus $\vec{B}_0$ shimming on a small volume than obtain equally good results over the much larger volume needed for MRSI. Other possible solutions are to suppress the outer volumes or to create saturation bands inside the FOV and thereby reduce those artefacts [14].

### 1.3.8 High-field Magnetic Resonance

Scanners with a nominal magnetic field of 3 T are considered high-field MR. Human scanners with a main field of 7 T and above are referred to as ultra-high field. The development of hardware and applications at high-field strengths brought great promise. The biggest and most obvious advantage of the high- and ultra-high field scanners is a greater splitting of the energy levels, which led to improvements in SNR, especially in the SNR per unit time. This can be used either to increase the spatial resolution or to shorten the acquisition time [25]. When the EPSI sequence is implemented on a high-field scanner, there is an increase in SNR per unit volume and time, which scales linearly with the nominal field strength. The use of EPSI at 7 T enables, e.g., the separate quantification of J-coupled metabolites (i.e., Glu and Gln) [20].

It was possible for researchers from the High-Field MR Centre in Vienna [25] to realize the potential of high-field when they created an ultra-resolution MRSI sequence with a matrix size of 128×128 and voxel dimensions of 1.7 mmm × 1.7 mmm × 8 mmm. This was achieved by a combination of 200ms TR, optimized FID acquisition, shortened water suppression, parallel imaging, and pulse-cascaded Hadamard encoding, all of which can be read about in detail in the cited paper [25].

One of the main technical challenges is to achieve reasonable $B_0$ and $B_1$ field homogeneity because the inhomogeneities decrease spectral quality, complicate water and lipid suppression, and cause larger *chemical shift displacements*. The reason the field homogeneity is worse at 7 T than at 3 T is that the splitting of energy levels is directly proportional to the field strength, and therefore, at the higher field, $\Delta\omega$ from Eq. 1.43 increases and makes the *chemical shift displacements* more pronounced and causes more issues than on a 3 T scanner. The field inhomogeneities may be greatly improved by higher-order shims. Also, long measurement times may be an issue. It may be challenging to avoid lipid contamination, but this can be dealt with in post-processing [25].

Other challenges at high magnetic field strengths are restrictions of specific absorption rates (i.e., how much RF power is deposited per unit mass in W/kg causing heating of tissue) or shorter T2 time constants. From the previously introduced sequences, the RF power limitations benefit, for example, the STEAM sequence over PRESS because STEAM does not contain any refocusing pulses that may cause problems in terms of peak RF power [17]. Due to shorter T2s, longer echo times are unsuitable, and thus, shorter ones need to be applied. Another way to overcome shorter T2s is to acquire FIDs instead of echoes [25]. The acquisition of FIDs further increases the SNR/t as its amplitude is higher than that of an echo [21].

### 1.3.9 Alternative medical imaging techniques

As MR is not the only medical imaging technique it is important to know about other diagnostic possibilities and their typical uses. Therefore, the following imaging modalities will be briefly reviewed and compared to MR: X-ray; computed tomography (CT); ultrasound; and positron emission tomography (PET). In clinical practise, it is essential to apply the most appropriate method for an accurate diagnosis and selection of the best treatment. Either one diagnostic method or several can be used. All of the methods described are non-invasive and painless for the patient [26].

In X-ray imaging, electromagnetic waves of a frequency range between $10^{16}$ and $10^{19}$ Hz [27] are used to penetrate tissue and the attenuated intensities are measured on the other side of the sample. From those data, the image is easily calculated. Nevertheless, the applied frequencies in this imaging modality fall under ionizing radiation and frequent exposure to X-rays increases risk of harm. The image quality is sufficient for accurate diagnosis of anatomical structures such as bones. X-ray imaging is the fastest imaging technique [26].

CT is based on the same technology as X-ray imaging, but the sample is not measured from only one viewpoint, as is the case in X-rays. A series of X-ray images is taken from systematically arranged angles that must be processed afterward. By this procedure, CT images contain a higher amount of information than plain X-ray images. Therefore, CT is suitable for more applications,

such as imaging of internal organs, as well as bones, soft tissues, and blood vessels [26].

Ultrasound imaging, also termed sonography, is unique in the way that it enables seeing the internal workings of bodily structures in real time. This is because ultrasound is based on the principle of reflection of high-frequency sound waves. Therefore, ultrasound gives information about the movement of organs and blood flowing through vessels. An important advantage of ultrasound over X-ray imaging and CT is that there is no ionizing radiation applied, making it a suitable technique for examinations of pregnant women and other radiation-sensitive groups. It is important to note at this point that MR also does not use ionizing radiation [26].

As the name implies, in PET, there are positrons used for imaging. Those come from an administered radiopharmaceutical, which is a positron emitter. The injected substance is usually glucose, marked with radioactive atoms. The sugar is easily absorbed and spread out in the body, taking up to one hour, and afterward, also naturally removed from the body. The freed positrons consequently annihilate nearby electrons and produce two photons flying apart in the exact opposite direction. Then, similarly to CT, the attenuated signal intensities are captured on detectors, and transformed into the information needed for image assembly. The PET diagnostic tool enables the examination of complex systemic diseases because it reflects processes on the cellular level. It enables viewing of the functioning of the brain, heart, blood flow, oxygen use, metabolism, or infection [28].

Although much has been explicated about MR in this thesis, not much has been described yet about the course of the patient examination. MR is able to acquire images of high quality, especially of soft tissues. But, there are a few things that may cause patient discomfort. First, the space inside the scanner is quite limited. Second, when the sequence is running, the application of the gradients can get quite loud. Third, the time of the medical intervention can be quite long for some diagnostic tasks. That is why there is much ongoing innovation regarding the acceleration of the acquisition process. It is also interesting to mention here that, with a so-called functional MRI (fMRI), it is possible to measure several metabolic processes, as is the case in PET. But, fMRI is not the topic of this diploma thesis, so it will not be discussed here in any further detail.

## 1.4 Acquisition features in MRSI

The most important of the acquisition modifications used in MRSI are *water* and *lipid suppression*, *motion correction*, and *scanner instability handling*. The artefacts that cause issues can be also dealt with after the acquisition in the data-processing stage. These other techniques will be explained in detail in the Section on *Data-processing concepts in MRSI*.

### 1.4.1 Water suppression

In this Subsection, *in-sequence water suppression* (WS) will be discussed. The rationale for the necessity of WS in MRS as well as in MRSI was explained in an earlier subsection on *spectral artefacts*. Here, first, the above-mentioned method using three CHESS pulses will be described. Then, another in-sequence water suppression technique, recommended by Tkáč et al. [29] in their experts' consensus, will be described - the *variable pulse power and optimized relaxation delays* (VAPOR) technique.

Based on already established knowledge, the water suppression procedure is ruled by the T1 relaxation process, so that, during the excitation of desired metabolites, there is only a minimal number of hydrogen nuclei in water molecules aligned with $\vec{B}_0$. As the CHESS pulses must excite only the selected spins, the pulse duration has to be relatively long (i.e., 20 to 30 ms) to provide a narrow enough bandwidth. The flip angle $\alpha$ of each of the three pulses, depicted in Fig. 1.23, might be different depending on factors such as timing, local T1 relaxation constant, and excitation effects. Because this is a non-trivial procedure, iterative optimizations of $\alpha$ are performed until a satisfactory level of the residual water signal is achieved [30].

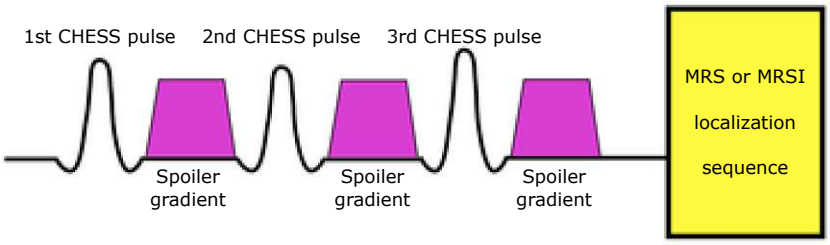


Figure 1.23: Schematic sequence diagram of the selective excitations and consecutive spoiling gradients for water suppression according to the CHESS method [30].

In the CHESS method, the water suppression is applied before the localization part of the sequence. As was already stated, CHESS first flips the water magnetization into the xy-plane where it is subsequently spoiled. The spoiling is done via a gradient that desynchronizes the spins. There are usually three CHESS pulses necessary for a sufficient water suppression [30].

The other WS technique presented here is VAPOR, which was originally designed for ultra-high-field (i.e., 9.4 T) rodent brain spectroscopy, a specific of which is the usual use of surface coils. In the original version, seven pulses were applied. As there is a relatively broad range of water T1 constants ranging from tissue to cerebrospinal fluid, an optimized design of the relative flip angles was necessary, including the timing of the pulses. Through careful design, a decrease in sensitivity for $B_1+$ (i.e., RF transmission) inhomogeneities can be achieved. In addition to that, in the free intervals, *outer-volume suppression* (OVS) blocks for lipids can be inserted. Those will be reviewed in the Subsection devoted to *Lipid suppression*.

Subsequently, the VAPOR technique was also adapted for human brain applications at 7 T. Changes were made in the design of the pulses and also in their timing to create enough space for OVS blocks and to guarantee desired frequency selectivity, achieved by extending the pulse duration. Furthermore, one extra pulse was added. In this manner, the intensity of the remaining water signal was suppressed considerably below the amplitudes of the major metabolite. It is interesting to remark that the VAPOR WS can be combined with STEAM-localized acquisition. Also, an additional CHESS pulse can be added to enhance the WS robustness [29].

### 1.4.2 Lipid suppression

For the lipid suppression in MRSI, there are recent recommendations including but not limited to *volume pre-selection*, OVS, and *selective lipid suppression*. There were three special techniques recommended that require extra devoted hardware and which are based on dynamic shimming, special gradient systems, and crusher coils. Those special techniques will not be reviewed in this thesis as they are beyond the scope of this work. All of the above-mentioned techniques are focused on the suppression of subcutaneous lipids in human MRS and MRSI experiments [29].

The process of *lipid suppression* is a very important step in MR spectroscopy, as lipid contamination in all proton spectroscopy approaches decreases the data quality considerably [29].

To guarantee a reliable *lipid suppression*, SVS-localization precision is essential. The desired level of *lipid suppression* can be achieved, for example, by applying the STEAM technique supported by OVS. The effectiveness of *lipid suppression* is dictated mainly by the PSF and movement. It can be performed either in-sequence or in data processing, as is the case for WS [29]. In this Section, only the relevant acquisition modulations will be described.

*Outer-volume suppression* is one of the most common *lipid suppression* techniques for multi-voxel spectroscopic imaging. OVS uses pre-saturation (i.e., selective excitations and subsequent spoiling gradients) of the tissues containing lipids with a contamination potential. To increase selectivity, the pass bands are kept constant and the pulse durations are lengthened. The presaturation is done in so-called *slabs*, which are thick spatial slices positioned in such a way that they cover the volume

around the brain as closely as possible. An example of the OVS-slab placement for a volumetric hydrogen MRSI of the brain is depicted in Fig. 1.24 [29].

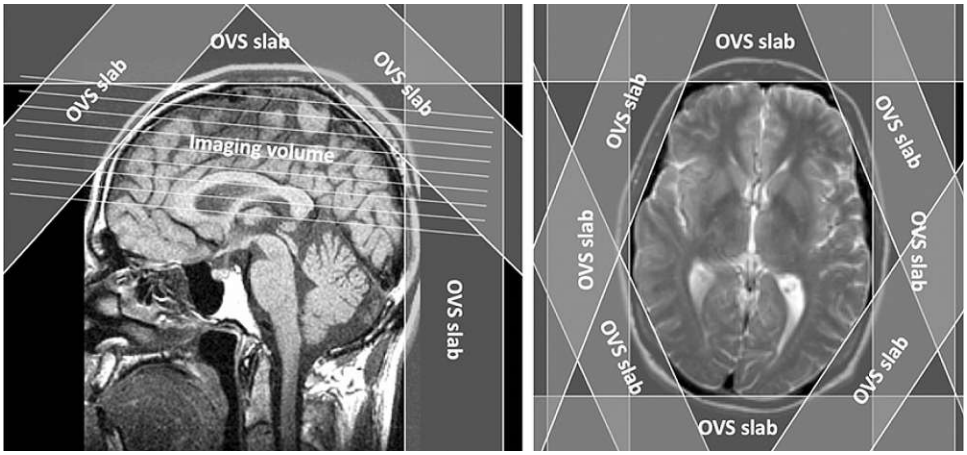


Figure 1.24: Positioning of the OVS slabs for a whole-brain hydrogen MRSI measurement [29].

To eliminate the subcutaneous-lipid contamination as much as possible, there may be up to 16 OVS slabs applied in all spatial directions. As mentioned above, for time-effectiveness purposes, the OVS is applied in an alternating manner with WS pulses [29].

### 1.4.3 Motion correction

Aside from the water and lipid suppression, *motion correction* and *scanner instability handling* are also necessary to acquire high-quality MRSI data. Those corrections are important because high $\vec{B}_0$ homogeneity proves essential for artefact-free data. Special attention must be paid to these issues mainly because of long scan times that increase the occurrence of movement and scanner instabilities. Movement during data acquisition leads to deviations in localization and shimming [31]. The artefacts caused by patient motion are termed *ghosting* or *smearing*, which may appear as in Fig. 1.25 [32]. Only the techniques applied during the acquisition will be described in this Section.

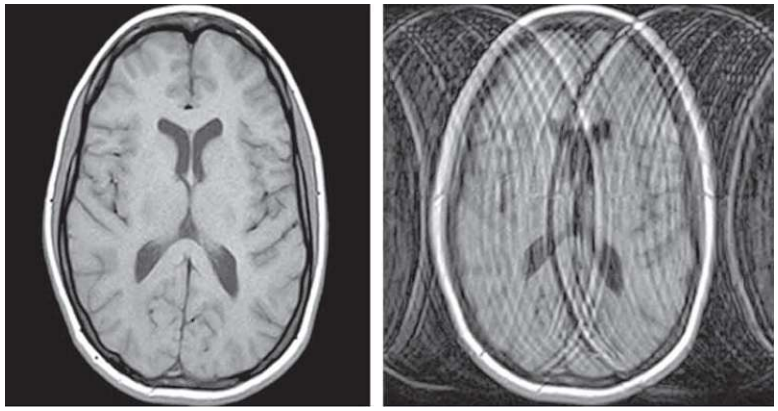

Figure 1.25: Illustrative image of the motion-related *ghosting* artefact on the right and a clear version of the same image on the left. This artefact typically manifests in the *phase encoding* direction, in this case along the x-axis [32].

As displayed in Fig. 1.25, motion damages the signal typically in the *phase encoding* direction, which was the x-direction in this case. The motion affects the *k-space* with respect to the magnitude and phase of the signal. Deviations in magnitude are caused by displacement of the measured voxel at the moment when an RF pulse is applied, and phase errors are created after the excitation by movements in the transverse plane [32].

The most effective technique with which to reduce the *ghosting* artefact is, first, the application of sequences with *short acquisition times*, simply because this decreases the amount of time in which the motion may occur. Those sequences became available with MR hardware improvements, such as stronger gradients, higher *slew rates* and higher $B_0$ field-strengths [32], all of which were utilized in this diploma thesis. It is also possible to accelerate the acquisition by various MR methods (e.g., different SSE *k-space* trajectories such as CRT). For further control of those undesired effects, the following strategies can be applied. A very effective and the most trivial one is *subject immobilization*, as was already mentioned in the subsection on *imaging artefacts*, also very helpful in spectroscopy.

And, finally, there are two data-driven approaches: *retrospective* and *prospective correction*. The *prospective motion correction* will be described in this Section as it is applied during the acquisition, and the *retrospective* variant will be reviewed in a later Section on *Data-processing concepts in MSRI* [31].

*Prospective motion correction* can be applied in real-time utilizing either an *internal* or *external reference* for motion tracking. With both of these techniques, localization corrections are achievable, but field homogeneity (i.e., shimming) can be improved only by *internal navigators*. In response to the observed deviations, the sequence frequencies and shims can be changed, usually once per TR. The benefit of the correction is stability of the MRS data and improved quantification, which enables measurement of even low-concentration metabolites. With *prospective motion correction*, gating of the acquisition is also possible, stopping the acquisition when there are severe movements expected and allowing it otherwise [31].

The *internal navigators* are extra sequence elements that assess the field distribution inside the volume of interest. They use the measured signal to draw conclusions about movement. These sequences can correct for the patient's position and also for $B_0$ fluctuations. The pulses of *navigator sequences* have to be implemented in accordance with the rest of the sequence. In this manner, the sequence duration might be prolonged. The *motion correction* is then technically achieved by appropriately changing the excitation and gradient encoding in real-time according to the tracking data. The *navigators* usually apply pulses with very low $\alpha$, selective to water and thereby minimizing their influence on metabolite signals [31].

*Prospective motion correction* via *external tracking* can use, e.g., a camera and marker system with the marker placed stably on the patient's forehead for brain MRSI. In this case, the motion tracking is independent of the sequence, so the corrections can be accomplished much more often than with *navigator sequences*. But, the detected movements still must be projected into the sequence execution to take effect [31].

However, it is possible to combine the best properties of the *internal* and the *external* tracing methods in a so-called *hybrid system* [31].

### 1.4.4 Scanner-instability handling

The last category of artefacts and their in-acquisition mitigation strategies discussed here are those caused by scanner instability. Artefacts caused by magnetic field distortions are differentiated according to the main MR-hardware components, which can cause $\vec{B}_0$ and *RF inhomogeneity*. Also, Eddy currents resulting from gradient application will be briefly reviewed [32]. The *gradient field non-linearities* will be discussed in the relevant *data-processing* section.

In the *main field inhomogeneities* (i.e., errors in the geometry of $\vec{B}_0$), the field distribution is not the same everywhere inside the gantry. The signal is then disturbed, which leads to errors in the final image. Although the scanner manufacturers strive for perfectly homogeneous $B_0$, in reality, there are always some field deviations. The major part of those deviations is caused by the

inability of the hardware to produce the nominal value of the field at the periphery of the bore. The main field fluctuations are also not predictable [32]. For this, measurements are taken with the sample inside the scanner, striving for a homogeneous magnetic field as a result. The deviations from a homogeneous field distribution are then compensated by additional magnetic fields emitted from *shim coils*. This procedure is mostly done automatically and is designed to minimize the T2* relaxation effects, prolonging the T2* constant toward the T2 constant [22].

The *inhomogeneities in the RF pulses* do not lead to distortions of the main field. But, the $B_1+$ *fluctuations* cause signal deviations (e.g., because of inhomogeneous lipid suppression), which are more severe at field strengths of 3 T and above because of the higher Larmor frequencies. Therefore, the excitation frequencies must be higher and more inhomogeneity might emerge. The $B_1+$ *inhomogeneities* are caused by imperfections in the coil quality. Those errors can also result from standing waves, especially when one of the dimensions of the object is bigger than the RF-pulse wavelength used, which would be the case for frequencies higher than 300 MHz. One strategy to reduce those error sources is to control the RF transmission elements separately, called *RF shimming*. The *RF shimming* enables the production of more precise excitations [32].

A phenomenon caused primarily by strong gradients (e.g., in diffusion-weighted spectroscopic imaging because of their high amplitude and long duration) is termed *Eddy currents*. *Eddy currents* lead to *magnetic field errors*. When a gradient is applied, the overall magnetic field changes, which, in turn, sets into motion electrical currents inside tissues. From those currents (i.e., *Eddy currents*), magnetic fields of small dimensions emerge. Those new currents further influence the main field. One strategy to handle the *Eddy currents* is the use of actively shielded gradient coils [32].

## 1.5 Data-processing concepts in MRSI

The series of the *data-processing steps* starts with the raw data. Raw data is processed to arrive ultimately at images and metabolic maps. An outline of the main data-processing steps is given in Fig. 1.26.

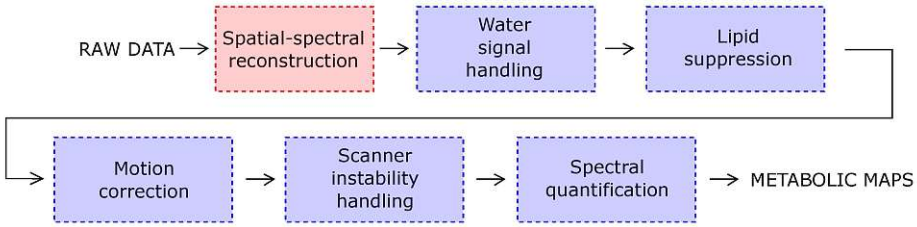


Figure 1.26: Overview of the major data-processing steps.

Only the *spatial-spectral reconstruction* highlighted in red, in Fig. 1.26, was implemented in this thesis. Nevertheless, the blue highlighted operations on the data are also essential to create MRSI outputs of good quality.

When the raw data has been recorded, it goes through the *reconstruction pipeline*. At the end of the reconstruction, there are the corrected time signals and preliminary spectroscopic images can be plotted using the first FID points to show proton density images.

All of the following steps will be described in the respective Subsection: *Water signal handling*; *Lipid suppression*; *Motion correction*; *Scanner instability handling*; and *Spectral quantification*.

### 1.5.1 Spatial-spectral reconstruction

The *spatial-spectral reconstruction* part of the data processing was the main focus of the work for this diploma thesis. The *conceptual description* of the data processing will be given in this Section. In a later Section on *Data-reconstruction procedures*, the respective *mathematical equations* will

be given. The *programming aspects* will be explained in the Sections on *Implementation of the Cartesian reconstruction* and *Implementation of the CRT reconstruction*.

The reconstruction consisted of six main steps, which are depicted in Fig. 1.27 in the order they were actually performed.


Figure 1.27: Data processing steps of the *spatial-spectral reconstruction*.

In the sequence used, there were, in principle, two different data subsets measured: calibration data and MRSI data. The calibration data was used, for example, for the coil combination, which was not implemented in the framework of this thesis. Only the MRSI data was processed for the *spatial-spectral reconstruction*. Thus, we had to ensure that only the desired data went through the pipeline, accomplished in the *Select data* task.

The selected data then had to be sorted from the measured arrangement into a systematic *k-space*. The goal was to sort the data from the measured scans into one data object consisting of the following dimensions: rings (i.e., radius, $k_r$); points on the ring (i.e., angle, $k_\varphi$); and FID points (i.e., time, $k_t$). As in SSE, the acquisition order is one repetition of the *k-space* trajectory after the other. The FID points from all the circumnavigations had to be re-assembled into FIDs in the same way as if they were measured with an SVS technique. On rings with two or three TIs, it had to be taken into account that, in one scan, only each second or third FID point was sampled, forcing an interleaved FID assembly on those circles. An illustration of a ring sampled with two TIs is given in Fig. 1.28.

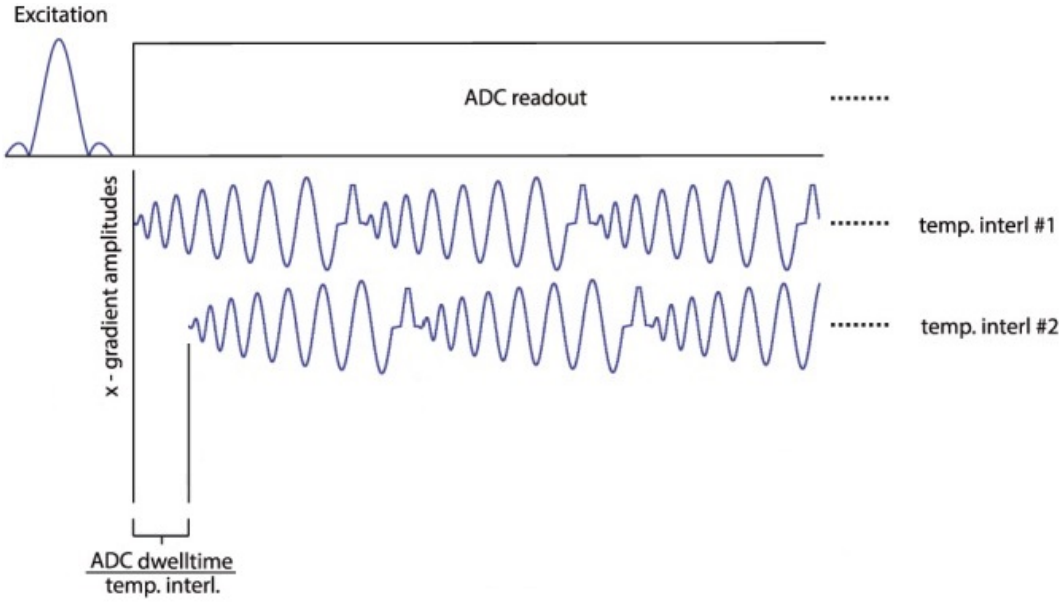


Figure 1.28: Illustrative sampling of the FID time signal with two temporal interleaves [33].

During TI number 1, only each second FID point is measured whereas the sampling starts with the first FID point in Fig. 1.28. To acquire all the missing FID points in between, TI number 2 (i.e., the lower x-gradient line) is sampled. When sampling with two TIs, it is essential that the x-gradient is played out with a time shift of a half *ADC dwell time*, $t_{ADC}$, whereas the *ADC dwell time* is the time passed between measuring the current and the subsequent point on the ring

(i.e., for CRT sampling) in the same TI. The general formula for calculating the time shift of the x-gradient between different TIs is given in Fig. 1.28. When $t_{ADC}$ is multiplied by the number of points on the ring, it gives the time to measure one FID point, the *spectral dwell time*. The *spectral dwell time* equals the time difference between two consecutive FID points and, as was stated above, the inverse of that builds the SBW.

When the data has been sorted correctly, all subsequent operations can be conveniently performed. There were three operations performed in the *k-space* domain (i.e., *FOV shift*, *frequency-offset correction*, and *density compensation*). And, one operation converting the edited *k-space* data into the *image domain*, the *spatial DFT*.

The aim of the next data operation was to correct for the *shift of the FOV* relative to the magnet isocenter. To acquire the parameters necessary for calculating the shift, first, the *localizer* positioning sequence was launched. Using the *localizer* image, the FOV was centered via translation and rotation. The applied shifts were stored in the raw data header. As all subsequent measurements were taken in a non-centered manner, the FOV of the images had to be shifted during the *spatial-spectral reconstruction*.

As already mentioned, when directly recording the FID, the gradients need time to reach the signal position in the *data matrix*. All the points on the ring trajectory cannot be measured at once. The gradients need some time to come from one point on the circle to the next while the data is sampled. This induced a first-order phase error on the data, resulting in different phase shifts of different spectral peaks. To obtain artefact-free data, all points on the ring in one circumnavigation would have to be measured in an instant, which is not possible. Therefore, it is necessary to perform the *frequency-offset correction*, which compensates for the fact that, with an increasing $\varphi$ in the $k_\varphi$ direction, there is an acquisition delay and frequency shift. Those shifts are first determined, calculated back, and applied to the data.

In fact, it does not matter whether one performs the *DFT* or Fast FT (FFT) to arrive in the image domain, because, in any case, the *density compensation* of the non-uniformly sampled data must be done [34]. For the FFT, the data must first be *density-compensated*, interpolated onto a *Cartesian* grid, and then the FFT can be calculated. In this diploma thesis, the *DFT* was implemented for which the data also had to be first *density-compensated*, and directly after that, the *DFT* could be performed.

To derive an interpretable image from the measured *k-space*, an FT in all measured spatial directions (i.e., in three dimensions for more than one spatial slice) must be performed. What the FT essentially does is to take as input any measured signal and transform it into its frequency and amplitude components. As was mentioned above, the input here for the *DFT* is also called *spatial frequency*, thus, after the FT, the contained spatial frequencies become apparent and thereby form an image. In the case of a *Cartesian k-space*, an FFT can be performed, which is a highly optimized algorithm. But, here, the *DFT* was applied because no re-gridding of the CRT data was necessary. As the main purpose of this thesis was to create a reconstruction pipeline for the *concentric ring trajectory* inclusive of the *DFT*, first the *DFT* was implemented for an easier *Cartesian* trajectory for preparatory and verification purposes. The *Cartesian* version of the *DFT* was implemented in two different ways with regard to access to the data. Both methods, with their advantages and disadvantages, are explained in the Section *Implementation of the Cartesian trajectory*.

### 1.5.2 Water signal handling

After the *spatial-spectral reconstruction*, the *water signal handling* is performed. As the WS is not the only method to deal with the water signal in localized proton spectroscopy, there are methods that do not apply any WS pulses in the sequence, but also acquire the water together with the metabolite signals and are rooted only in the processing of the "contaminated" data. The reason to perform (imaging) spectroscopy without WS is first to shorten the sequence duration and also to simplify its implementation by leaving out the WS part. Also, the measured full water signal can be used as a reference for quantification, phase correction, and frequency or amplitude variations [29].

This approach has not been used since the beginnings of proton MRS because the available dynamic range for detection was limited, which is no longer an obstacle when appropriate analog-to-digital converters (ADC) are used [29].

Nevertheless, there are still some issues that need to be addressed. One consideration are the *side bands* of the water signal in case of any signal modulation during acquisition (e.g., by switching of slice or spoiler gradients leading to water frequency modulation). Those *side band* artefacts are problematic because they spread a wide spectral range, and thus, overlap the metabolites [29].

The properties of the *side lobes* are utilized to solve this problem. Mainly, the fact that their amplitude is directly proportional to the strength and sign of the gradients used and that the water signal is in-phase with the signals of interest. This can be exploited by metabolite cycling (MC). In MC, the *chemical-shift* selective RF pulses position the metabolite signals either in the up- or down-field of the spectrum, relative to the water signal. This cycling of the metabolites is done in an alternating fashion in which the water signal always remains in place. Then, in the data processing, the two variants of the measured signal are added and subtracted, which separates the metabolite signals from the water signal and its side lobes [29].

### 1.5.3 Lipid suppression

Tkac et al. recommended *lipid-suppression* methods utilizing the three following components simultaneously: *k-space filtering*; *high spatial resolution*; and *lipid regularization* acting in the data-processing domain [29].

The *k-space filtering* in the more complex three-component method does nothing else than a multiplication of the raw data by a weighting function. In this manner, the PSF can be adjusted according to specific needs. For the weighting function, mostly low-pass filters are applied. One example is the Hamming function, $H$, Eq. 1.44, dependent on the data-point index, $n$, and the total number of the *k-space* points, $N$, combined inside a cosine function [29].

$$H(n) = 0.54 + 0.46 \cdot cos(2\pi \frac{n}{N}) \tag{1.44}$$

In general, low-pass filters minimize the side lobes of the PSF, and thus, help to prevent spreading of unwanted lipid signals, termed *voxel bleeding*. The Hamming function suppresses the peripheral side lobes most effectively (i.e., up to 0 dB in subplot IV in Fig. 1.29) and the side lobes near the central bright spot less effectively [29].

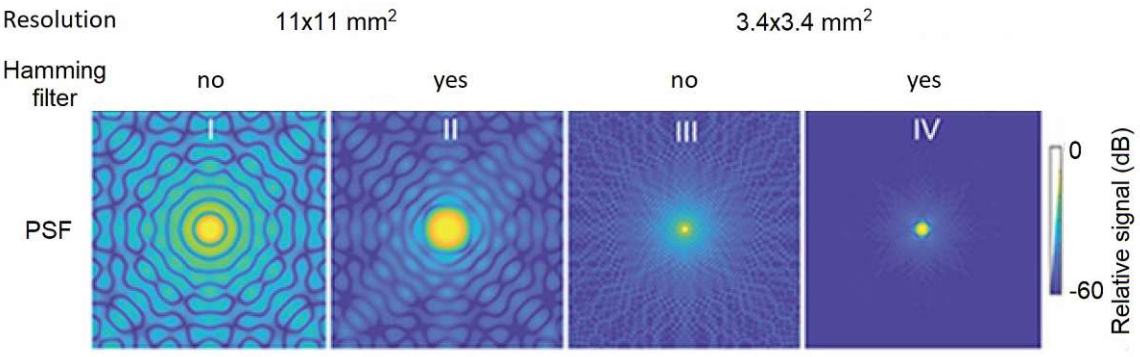


Figure 1.29: Shape of the 2D-PSF in relation to the application of the Hamming filter and the spatial resolution. The amplitude of the PSF-plots is given on a decibel (i.e., attenuation) scale related to the height of the central peak [29].

An improvement of the PSF (i.e., fewer side lobes and a narrower central bright-spot) can be accomplished by combining the application of a Hamming filter with high spatial resolution, as

depicted in Fig. 1.29.

When comparing plots II and IV in Fig. 1.29, it is evident that, by increasing the resolution (i.e., from $11 \times 11\ mm^2$ to $3.4 \times 3.4\ mm^2$), the PSF significantly improves after the effects of spatial *k-space filtering*. This shows that not only in-sequence *lipid suppression* methods help to avoid lipid contamination from the extra-cranial space [29].

As was mentioned above, the quality of the resulting *metabolic maps* can be further increased by including *lipid regularization* in the data processing, which is, in essence, a minimization problem of the scalar product of two vectors (i.e., increasing their orthogonality). One of the vectors is formed by lipids and the other by the metabolite spectrum. To keep the reconstructed data similar to the measured *k-space*, there is a data consistency term, the minimization of which, ensures just that [29].

### 1.5.4 Motion correction

In this Section, only *retrospective motion correction* (i.e., performed on the already acquired data) will be reviewed, as its *prospective* counterpart has already been described in a previous section.

The *retrospective motion correction* methods are generally easier to implement than the *prospective*, but are not able to perfectly calculate back the motion. In SVS with more transients, the measurements affected by motion can be identified and dismissed. On the remaining data, a procedure called *spectral alignment* is performed, which corrects for frequency and phase changes. In this way, the quality of the spectra is somewhat improved, but one problem still remains, the spectral averaging from different locations inside the sample. Also, major $B_0$ inhomogeneities cause spectral distortion that cannot be corrected by those means. Therefore, the spectral averaging with the use of *retrospective motion correction* cannot be perfect [31].

### 1.5.5 Scanner-instability handling

The scanner instability discussed in this Section concerns the gradient system. *Non-linearities in the gradient fields* result from the finite size of the respective coils. But, unlike the $\vec{B}_0$ inhomogeneities, the gradient deviations are predictable, can be corrected, and thus, the localization errors eliminated. But, gradient field errors also lead to a decrease in spatial resolution that cannot be recovered [32].

### 1.5.6 Spectral fitting and quantification

In this Section, a unit conversion of the spectral x-axis will first be described, followed by a description of some spectral properties, such as *spectral bandwidth*, *spectral dwell time*, number of data points, and how they are interconnected. The topic of *averages* will be described next. After that, the role of *filtering* with respect to *spectral fitting* will be mentioned. Then, parametric models, and especially, *linear combination model fitting*, will be reviewed for SVS and MRSI. The process will continue with the creation of the *metabolic maps* and conclude with SNR considerations.

When the *spatial-spectral reconstruction* is complete, the process of *spectral fitting and quantification* is next, because the spectra must first be created from the time signal (i.e., FID in this case) by an FFT. The spectrum in unit Hz is centered around the main metabolite, the resonance frequency of which is subtracted from each data point on the frequency axis of the spectrum. The range of the spectrum is then given by the spectral bandwidth (SBW). The SBW is determined by the *spectral dwell time*, $t_{spectral}$, and the number of data points in one scan, $ns$, according to Eq. 1.45.

$$SBW = \frac{1}{t_{spectral} ns} \tag{1.45}$$

The next step is the conversion of the horizontal axis of the spectral diagram from Hz to ppm (i.e., parts per million). The resonance frequencies given in Hz are linearly dependent on the magnetic field strength, thus, the resonance of a given metabolite is different when measured on 3T

or a 7T machine. The main advantage of the units ppm is that the peak positions in the spectrum are independent of the magnetic field strength, $B_0$, and therefore, the spectra can be compared across different scanners. The physical quantity, given in ppm, is the *chemical shift*, $\delta$, calculated as follows:

$$\delta = \frac{\nu_i - \gamma_0}{\gamma_0} \cdot 10^6 \tag{1.46}$$

In Eq. 1.46, $\nu_i$ is the resonance frequency in Hz of each of the data points indexed with $i$, and $\gamma_0$ stands for the Larmor frequency of the reference nucleus. The reference nucleus is usually the chemical substance tetramethylsilane (TMS), which is used for its ideal properties such as its strong and sharp resonance, and an ideal chemical shift, not interfering with other metabolites [35].

Sometimes, because the exact same signal is acquired multiple times, the individual signal sets are referred to as *averages*. Considering the averages and processing them is only relevant for 2D acquisitions where a good signal-to-noise-ratio (i.e., SNR) is achieved by multiple amounts of data. Averages are usually not measured in 3D acquisitions because the measurement times would be unacceptably long, as the measurement time scales linearly with the number of averages. The other reason for not measuring averages in 3D is that, with a 3D experiment, the amount of data is already sufficient for SNR. The same holds true for multi-channel acquisitions (i.e., measurements with an array of receiver coils, e.g., 32 channels). To process the *averages*, they are usually simply added.

In SVS, the pre-processed data (i.e., corrected for *Eddy-current* effects, *motion*, *frequency* and *phase* drifts, *residual water* and *lipids*, and edited by the RF coil combination, spectral FT, signal averaging) are then ready for analysis with the goal of estimating the metabolite-signal intensity. This can be accomplished in one of the following ways: *parametric fitting*; peak fitting; modeling of macromolecule and baseline signals; or peak integration. Here, only the *parametric fitting* will be described as it is one of the most popular analysis techniques [36].

But, before the *fitting model* can be applied, the spectra must go through the *filtering* process. The filters are performed on both the spectroscopic and the noise data. A classical filter is the Hamming function, which was described above. Another part of the *filtering* is the *frequency-offset correction*, the concept of which has already been described above. As a result of this correction, the baseline of the spectrum is curved. This issue can be handled either by *parametric fitting* by subtracting an uneven baseline that might negatively influence the fitting, or when the first-order phase errors are not corrected and the same errors are applied on spectra in the *fitting process* [21].

In the data processing of volumetric brain MRSI measurements, there have been significant improvements recently that have increased the quality, which may be relevant for routine clinical practise, and thus, enable brain MRSI to eventually become a standard imaging modality as is SVS currently. The establishment of this method in clinics has great potential for adding valuable information relevant for healthcare. One of the significant technological developments is a wider availability of scanner with 3 T fields and above. Further developments include multi-channel receiver coils, accelerated encoding methods, and novel techniques to arrive at the metabolic maps [37].

As was stated above, the *fitting* analysis method is one of the parametric fitting techniques and will be described in the following because of its frequent use. In this technique, the contributions of individual metabolites to the whole spectrum are estimated. Based on those estimates, the spectrum can be split, as depicted in Fig. 1.30. The single metabolite spectra are individually referred to as *basis spectra* and, altogether, as the *basis set*. The full spectral contribution of each metabolite is determined in phantoms or by numerical simulations. When the *basis spectra* are used, the number of necessary model functions to represent the whole spectrum is reduced, resulting in less parameters to fit. The *fitting* is a minimization problem typically solved via non-linear least-squares analysis. In this method, the basis spectra are inserted to fit the acquired spectrum. To achieve this, the amplitudes and frequencies of the basis set are adjusted. To increase the precision of the fit, global values are also changed (e.g., spectral phase or linewidth).

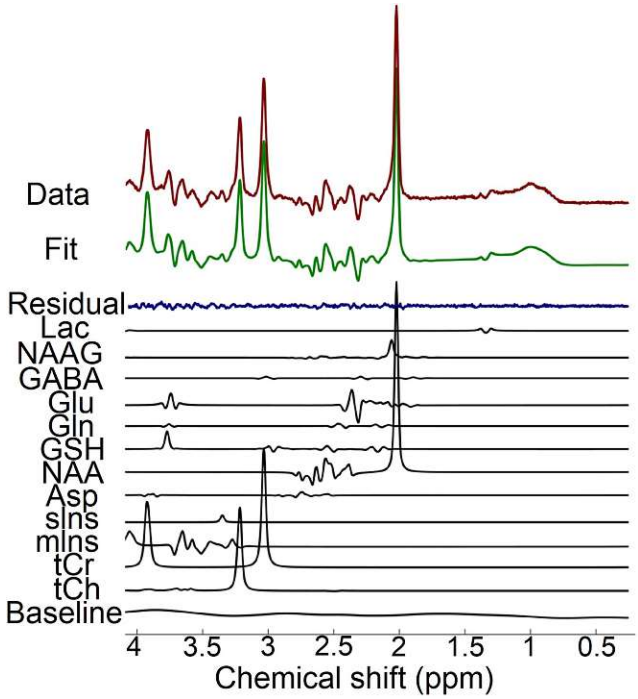


Figure 1.30: *Parametric model fitting* from a 3T human brain spectrum obtained with a PRESS sequence showing the acquired data, overall fit, fit residual, and individual metabolite components [36].

In Fig. 1.30, the difference between the data (red) and the fit (green) indicates the goodness of the fit or residual (blue). The *fitting* method is recommended by Near and colleagues for in vivo proton MRS studies because it is able to analyse dense spectroscopic data (e.g., short-TE human brain scans). With this model, it is also possible to include macromolecular and baseline components. Baseline components that are not modeled may cause uncertainties of the fit [36].

The *spectral fitting* method is also relevant for processing in MRSI. With this method, large amounts of spectroscopic data can be analyzed automatically. The modeling process is usually performed in an iterative manner and prior knowledge about metabolite spectra is involved. The prior spatial information is used, for example, to exclude local outliers. Another optimization possibility is to set spatial constraints or to choose starting values of iterative algorithms. As stated above phase and frequency corrections before the fitting improve the results of the spectral analysis [37].

Typical data operations subsequent to the *parametric fitting model* are the calculation of *metabolic maps*, time and spectral SNR, and also determination of the individual *tissue contributions* [21]. There are more steps, but, for this thesis, it is sufficient to mention only the above.

When the spectral fitting is completed, the data are prepared to be reliably quantified. In this process, the intensity values are converted into concentration units. For this, some kind of reference is necessary to guarantee quality conditions for interpretation and inter-patient and inter-group comparisons. Either an internal or an external reference can be used. Possible internal references include the use of tissue water or metabolic ratios. On the other hand, for external referencing, signal sources from outside the body are used and also the ERETIC (i.e., electric reference to access in vivo concentrations) method [36]. The final steps of the data analysis is the *normalization* of the signals and the following *quantification*. The *normalization* is necessary due to field and acquisition differences that must be eliminated to ensure comparability. The field corrections compensate for transmit/ receive sensitivity and the different acquisition methods cause deviations in the response amplitude. This can be implemented via co-registered proton-density MRI or water reference MRSI.

When tissue water is used as a reference, it provides the advantage of exact co-registration and can be also utilized in other data processing steps in which the volume fraction can be obtained from high-resolution MRI segmentation. To eliminate all side effects and arrive at correct quantities, the reference should be corrected for water content, *partial volume effects*, and water and metabolite relaxation constants [37].

The creation of *metabolic maps* starts with the normalized data. Several types can be produced, such as metabolic ration maps, interpolated maps, maps with excluded outliers, and others. Also, further synthetic maps can be produced. Those maps are created by a linear combination of the metabolic maps, the essence of which is to analyse all metabolites by computing an orthonormal discriminant vector. More on this procedure can be found in the dissertation of Bernhard Strasser. The results can be used, for example, for the assessment of differences between a healthy control group and oncology patients [21].

After the *metabolic maps*, the two types (i.e., *time* and *spectral*) of SNR are calculated. For the calculation, both of those methods utilize the pre-processing noise data. The *temporal* SNR uses as a signal the *parametric-fitted* concentration of the total NAA (tNAA). As noise is the standard deviation of the time domain, noise is used and scaled to compensate for previous scaling. The *spectral* SNR is determined by performing a scaled FT on the noise data and, as the signal, the spectral plots are obtained. The reader is invited to consult [21] for further details.

Another way of *quantification* is to use the calculation of *tissue contributions*. In this approach, the *tissue contributions* are essential for gathering the absolute concentrations (i.e., in units mol/kg). In the brain, there are the following main tissues: gray matter; white matter; and cerebrospinal fluid. The different contributions of those tissues are determined for each voxel. The metabolic concentrations relative to water can be determined from MRSI measurements. It is then possible to convert the relative concentrations into absolute ones by determining the concentration of water because the water concentration is different for each of the three major brain tissues. This is done via segmentation of images and an automated tissue recognition. After that, the PSF of the data is applied to the segments and the absolute quantification can follow [21].

# 2 Materials and Methods

In this Section, all of the hardware, software techniques, and applied data-processing methods will be described, beginning with the hardware components of the scanner used, and how it was set up for taking measurements. This will be followed by detailed explanations of the SSE *k-space* trajectories, in particular, the CRT encoding method of choice in this thesis. Next, how the *offline reconstruction* for this trajectory was implemented by a colleague prior to the start of this thesis will be detailed. Then, the tools used to implement the *online reconstruction*, the ICE *framework*, will be described. Then, follow up on a previous Subsection on *Spatial-spectral reconstruction* that detailed the conceptual side of the applied data operations, the mathematical descriptions will be provided. Finally, there will be a short subsection about the display of reconstructed data.

## 2.1 Magnetic Resonance scanner hardware

In the High-Field MR Centre in Vienna at the AKH and the Medical University, there are two 3T systems and one 7T system available. All these machines are devoted to high-field MR research and the manufacturer of all of these is Siemens (Erlangen, Germany). For each of the three scanners, many coils are available, including volume and surface coils. Furthermore, the measurements and development can be performed on ex-nuclei such as $^{13}C$, $^{23}Na$, or $^{31}P$ [38].

To implement the tasks of this thesis, only the 7T machine was used. The full name of the machine is *7 Tesla Siemens Magnetom*, which is depicted in Fig. 2.1. It is able to perform whole-body measurements and is housed in a new building on the medical campus. This scanner is equipped with a multi-channel receiver coil, with 32 elements in which each of those coil elements records the signal individually from a slightly different position [38].

Figure 2.1: The 7 Tesla Siemens Magnetom scanner [39].

The desired magnetic field distributions inside this scanner are created by a strong micro-gradient system, *7T SC72CD*, with a total gradient strength of 70 mT/m. The nominal slew rate of the gradient system is 200 mT/m/s. In this thesis, a head-volume coil with 32 receive channels from the company Nova Medical (Wilmington, USA) was utilized [25]. The receiver channels are

independent and each of them generates a partial image [40], which are then fused together during the data processing (i.e., a coil combination that was not implemented in this thesis; more about coil combination can be read in [21]).

Other technical specifications of the scanner are a bore size of 0.6 m, a magnet length of 2.7 m, and a total system weight around 25 tons. The system supports passive as well as active *shimming*. The cooling of the superconductor magnet creating the main field is ensured by a zero helium boil-off technology [40].

On the *7T Magnetom* scanner, the following hardware (HW) units are involved in the imaging process. The first unit is the whole set of scanner coils, designated as *Scanner* in Fig. 2.2. Those coils take care of the main, gradient, and excitation fields, as well as of the signal-receive process. The three remaining HW units with the details (i.e., number of central processing units [CPUs], operating system [OS], and number of logical memory addresses) are given in Table 2.1 [41].

| HW unit | CPUs | OS | Memory addresses |
|---|---|---|---|
| MPCU | 1 | VxWorks | 32-bit |
| MRIR | multi | Linux | 64-bit |
| Host | 2 | Windows XP | 32-bit |

Table 2.1: Computing units of the MR system with number of CPUs, OS, and number of memory addresses [41].

All of the HW components are connected via an ethernet cable with a rate of data transfer of 1 Gbit/s. The communication on this network is dictated by the transmission control protocol and the internet protocol (TCP/ IP) [41]. The diagram of how the image reconstruction is embedded into the MR system is depicted in Fig. 2.2.

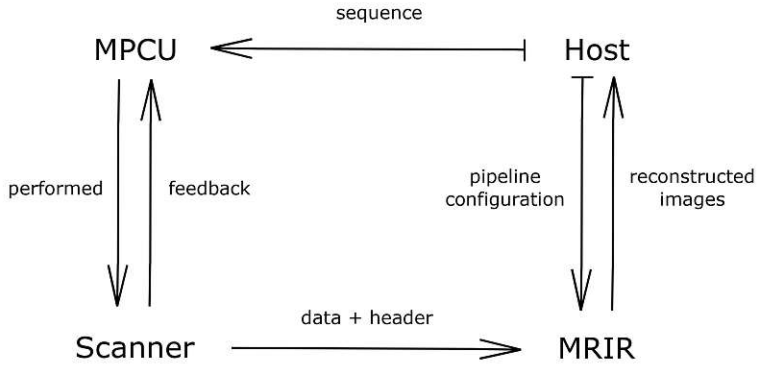


Figure 2.2: Diagram of the image reconstruction embedding into the MR system relying on the described HW components. Edited from [41].

The whole imaging process (i.e., the acquisition as well as the reconstruction) was initiated by the Host. First, the sequence was transferred to the MPCU from where it was performed by the scanner. During the acquisition, there was a constant feedback loop running between the MPCU and the scanner. As the acquisition started to generate data, it was moved together with the measurement data header to the MR imager (MRIR). The MRIR unit received instructions on the configuration of the reconstruction pipeline from the Host. Finally, the reconstructed images were sent from the MRIR to a database on the Host [41].

## 2.2 Experimental set-up

In the framework of this thesis, numerous experiments were performed on the 7 T scanner. When significant progress was made on the implementation of the image reconstruction, the program was always compiled and tested. In the development process, the functionalities were continuously tested in a virtual box on a laptop, but, to guarantee an absolutely correct execution, experiments directly on the scanner were necessary.

The experiments were mostly done on phantoms. Phantoms are devices often used in biomedical research as substitutes for human tissues. They are used to verify that imaging systems and methods are working as expected. Based on phantom tests, it is certain that the medical images can be relied on and that the quantification results are correct. Further phantom use-cases are image comparison between different machines, comparison of images from a single machine at different time points, and for scanner calibrations. Overall, the accuracy of the diagnosis of disease, brain injury, cancer, implants, and other issues is ensured, with the ultimate goal of better patient care [42].

The phantoms are manufactured from a variety of materials: plastic; salt solutions; silicones; epoxy; polyurethane foams; carbon powder; water; mineral oils; and others. The materials are chosen based on properties such as density, stretching strength, hardness, availability, cost, and many more to mimic the response of living tissues as closely as possible [42].

In our case, the phantoms were used to acquire data on the 7 T scanner. The data was then reconstructed in two different pipelines implemented in two different programming environments and also programming languages. The results were then compared in which conformity was expected as one pipeline served as a standard (i.e., the *offline reconstruction pipeline*) and the newly developed pipeline should achieve the same quality (i.e., the *online pipeline*). The advantages of phantoms are that they are available at any time, are standardized, and are not subject to movement. The specific phantom used for our experiments was the *Spherical Phantom 18 blue* containing 0.011 g of *Macrolex blue* per liter (i.e., mineral oil) from Siemens, depicted in Fig. 2.3. This phantom possessed two resonances and was non-conductive. In the selection process of the phantom, the occurrence of standing-wave artefacts was considered (i.e., applied wavelengths must not equal or be near the phantom diameter).


Figure 2.3: *Spherical Phantom 18 blue* from Siemens used for all phantom measurements in this thesis.

The procedure for phantom experiments included the following. It was extremely important, before entering the scanner room, that all ferromagnetic objects be removed and left in the an-

techamber. The patient table was then pulled out of the scanner completely and the transfer and receive head-coil was put in place. The coil connectors were then easily plugged in. The connectors and plugs were numbered for correct matching. After that, it was ensured that the connectors were positioned upright. Next, supporting pads for the phantom were put inside the coil so that the phantom could be stably positioned and fixated. A cross on the phantom was then aligned with a laser cross on the scanner to set a zero position. After that, the patient table was slowly moved into the magnet until the system signaled the correct position on its display. With this last step, everything was prepared and the magnet room was left.

On a console (i.e., Host) in the pre-room, the phantom-identification data were first filled out and the *localizer* sequence started. The *localizer* image was centered and the shifts stored in the data header. Then, the CRT sequence was started in which more runs with different matrix sizes were measured.

When the measurement was complete, the raw data, including headers, was exported via TWIX software to an external drive for development purposes. Ultimately, when the whole *online reconstruction pipeline* was successfully implemented it ran directly on the scanner simultaneously with the acquisition process. Then, the resultant images, in format .ima (i.e., digital imaging and communications in medicine, DICOM), were downloaded onto an external computer where they could be displayed and analyzed. The *displaying of the data* will be discussed in a later subsection.

To restore the magnet room to the initial condition, the patient table was slid out and the pads and the phantom were removed. To unplug the connectors of the inserted coil, there were fuses that had to be released first.

When the last phantom experiments delivered accurate results, an in vivo volunteer-measurement was performed. This was done to prove the principle of the developed reconstruction pipeline on in vivo data. The reconstructed images were then further processed by the algorithms of the High-Field MR Centre in Vienna to arrive at the final metabolic maps.

## 2.3 Different k-space trajectories of SSE

The following *k-space* trajectories used for *spatial-spectral encoding*, as well as their, properties will be reviewed: EPSI; *rosettes*; and *concentric rings*. All of these encode the 2D xy-plane via SSE and the spatial encoding of the z-direction is done via classical *slice selection*. In Figs. 2.4 to 2.6, three different *k-space* trajectories are displayed. In those plots, there is always one trajectory highlighted in pink. This pink trajectory is edited and exercised systematically at different positions, depicted in blue, until the whole *k-space* is sampled. Each of the trajectories (e.g., the pink one) is repeated many hundreds of times after one excitation to collect data along the spectral dimension. The diagrams also depict the gradient forms and slew rates as functions of time. In those two time functions, the blue line always describes the x- and the red line the y-component. On the right side of each of those figures, the achieved sampling density is shown.

Figs. 2.4 - 2.6 explore the relationship between the three subplots (i.e., *k-space* trajectory, gradients, and *slew rate*). In the sequence, gradients, $G$, are applied in the corresponding direction (i.e., x or y) to localize the signal. The gradients are mathematically noted as functions of time, $G_x(t)$ and $G_y(t)$. To determine the position in *k-space*, the gradients must be first integrated over time to give the so-called gradient moment, $GM$. The gradient moments are likewise functions of time given with respect to the spatial direction of the gradient, formalized in Eq. 2.1.

$$GM_x(t) = \int G_x(t)dt \tag{2.1}$$

The x-gradient, $G_x(t)$, is given in units T/m and time in s, therefore, the resulting $GM$ unit is Ts/m. When $GM$ is known, the *k-space* coordinates $k_x$ and $k_y$ can be calculated with Eq. 2.2.

$$k_x(t) = \gamma GM_x(t) \tag{2.2}$$

The unit of the gyromagnetic ratio, $\gamma$, of a specific nucleus is $1/(\mathrm{Ts})$, which gives the *k-space* unit $1/\mathrm{m}$. Based on Eq. 2.2, it can be said that *k-space* is nothing but a differently scaled gradient moment. All of the above calculated quantities are calculated for the y-component in the exact same way.

It is interesting to note that the reverse operation to Eq. 2.1 also holds true, where the gradient is calculated as a time derivative of the gradient moment:

$$G_x(t) = \frac{dGM_x(t)}{dt}$$

When a time derivative of the gradient is determined, it results in the *slew rate*, $s$:

$$s_x(t) = \frac{dG_x(t)}{dt}$$

When we consider the *k-space* coordinates that are practically equivalent to $GM$, the evolution of the position in the *data matrix* over time can be described by $G$ and the rate of change by $s$.

The sequence diagram of EPSI has already been described in Fig. 1.22. In this case, only the x-direction is localized by SSE and the y-direction by traditional *phase encoding*, therefore, in the gradient and slew rate diagram in Fig. 2.4, only the blue line varies in time. The gradient form is as rectangular as possible according to the slew rate limits. The sampling (i.e., ADC) takes place only during the phases of constant gradient strength. The parts of the diagram with tilted gradient strength come from the acceleration and deceleration process (i.e., slew rate). The achieved sampling density has moderate values at the x-extremes and in the center.

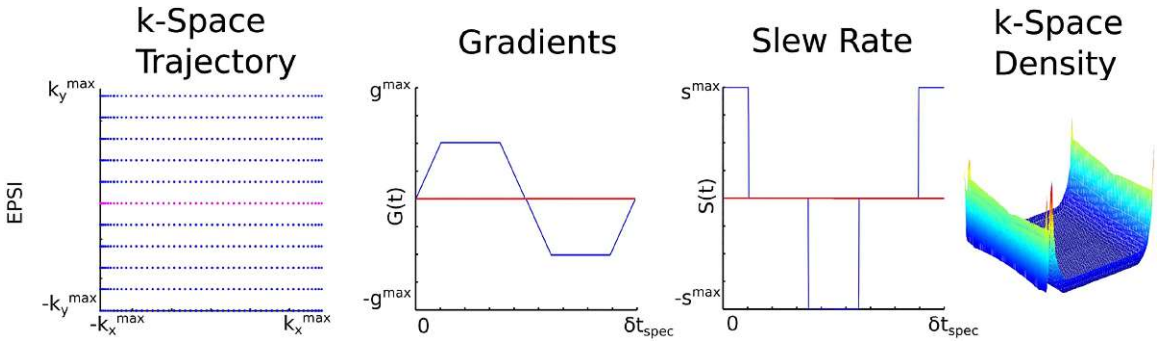


Figure 2.4: *Cartesian* EPSI *k-space* trajectories with gradient and slew rate diagram and measured *k-space* density [31].

The advantage of the EPSI sequence over clinical standard proton MRSI protocols, which represent a reference for acceleration comparison of all SSE trajectories, is an acceleration factor of 20 or higher. An advantage over other SSE trajectories is a constant *k-space* weighting due to sampling along a *Cartesian* trajectory. But, EPSI is connected with some losses of the SNR/t, just as all the other trajectories presented here. Either the SBW or the spatial resolution is limited, as is also the case in all the remaining trajectories. Furthermore, this sequence is inherently very gradient-demanding because the slew rate should be as high as possible and should also change in this manner to minimize the amount of useless data. The EPSI sequence should be implemented on scanners with a maximum field strength of 7 T [31].

An example of the filling of *k-space* by *rosettes* is depicted in Fig. 2.5 in which the basis trajectory, in pink, is a ring. The *k-space* trajectory, in Fig. 2.5, is just one version of rosettes, the simplest one (i.e., eccentric rings). Rosettes can be also parameterized in such a way that they look, e.g., like spirals. To create the circular shape of the trajectory, the gradients must have a phase-shifted sine-wave form. The slew rates are again phase-shifted sine waves. The positioning of the single rings differs in the initial value of the gradient moment (i.e., the integrated area below

the gradient curve). When the gradient is zero, the sampling occurs in the center of the *k-space* and, as the gradient strength increases in amplitude, the higher spatial frequencies are sampled. This principle holds true for both the in-plane directions. For the depicted *rosette* trajectory, the sampling density is very high in the center due to the intersection of all the rings.

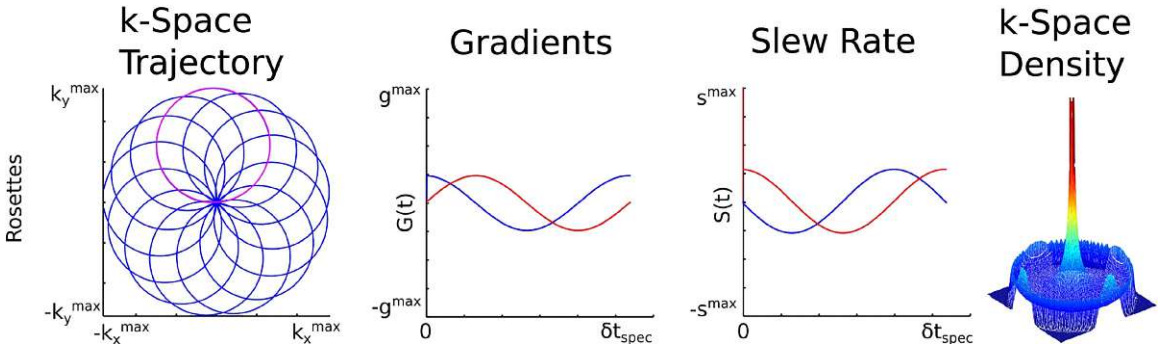


Figure 2.5: *Non-cartesian rosette k-space* trajectories [31].

The advantage of this *rosette* trajectory is that the measurement can be taken either at high speed or more slowly, but with less stress on the gradient coils. This acquisition technique is recommended for ultra-high fields of 7 T and above [31].

The *equidistant* CRT sampling is very similar to the *rosettes*. What differentiates the two techniques is the centering of the rings. With *rosettes*, the center of each ring is at a different position, but for CRT, the *spatial-frequency space* center is the middlepoint of all the rings, as the designation indicates this with the word *concentric*. To sample rings with an identical center and an increasing radius, the amplitude of both gradients is increased incrementally while the relative phase of the two gradients is kept constant, unlike with *rosettes*, where the amplitude was always the same, but the relative phase of the gradient sine waves changed. The slew rate for CRT is likewise sine-shaped, shown in Fig. 2.6. For the outer rings, the slew rate is mostly insufficient to sample according to the Nyquist criterion, therefore, *temporal interleaves* are applied as has been described above.

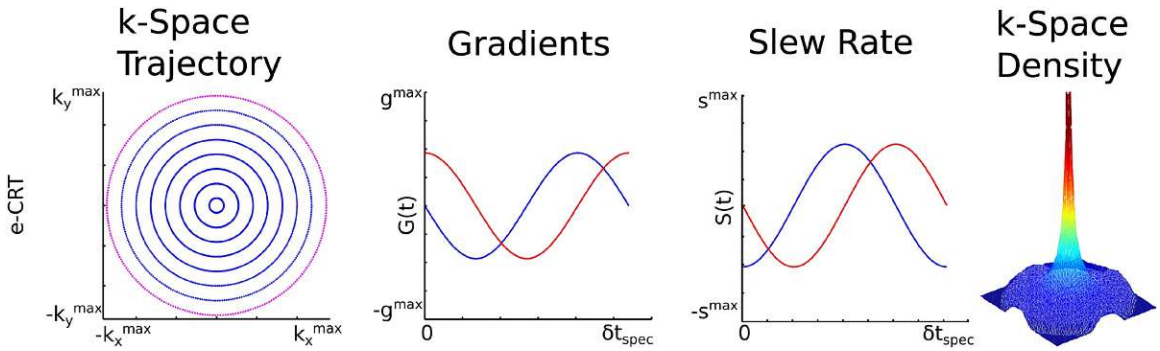


Figure 2.6: *Non-Cartesian concentric ring k-space* trajectories [31].

The CRT sequence was implemented by our MR Centre colleagues in two versions, *OldADC* and *NewADC*. Those versions differed from each other in the number of points on a ring. In *OldADC*, the number of points per circle was constant for all rings. In this version, the *k-space* density is very high at the center, as depicted in Fig. 2.6. But, this effect can be partially mitigated by measuring less points on the inner rings. This is possible because the Nyquist criterion is never violated there. With the increasing ring radius, $k_r$, the number of points per ring, $N\varphi(r)$, also increases in the

*NewADC* version. Both the versions have pros and cons. On the plus side in *NewADC*, less data is acquired, by a factor of 1.5 to 2, which is caused by the lower number of points. Therefore, the reconstruction needs less time for processing. However, the programming implementation was more complicated because of the variable number of data points per ring, and therefore, it was necessary to use *cells* in MATLAB, which, in turn, increased the computational demands. Also, the *density compensation* was different due to the changed sampling density because that is dependent on how fast or how often *k-space* is traversed along a trajectory for a specific area. The *density compensation* was not as strong as in *OldADC*. In summary, there was no significant difference between the two versions apart from the different $N\varphi(r)$. Therefore, only the *OldADC* version of the sequence was used in this diploma thesis.

The measurement of *concentric rings* allows high acceleration factors, but is very gradient-demanding, up to the limits where TIs must be used. Appropriate field strengths for this method are 3 T and above [31].

When comparing the different SSE trajectories, one of the most important issues is how fast the *k-space* can be filled with the basic, pink trajectory (i.e., how many excitations or TRs). The number of trajectories is given by the spatial Nyquist condition (i.e., the maximum distance between the samples in *spatial frequency space*). For example, for an 8×8 matrix, the EPSI technique demands eight TRs because the eight lines, but the CRT satisfies the same matrix size with only four TRs, as only four rings are needed. For spirals and rosettes, it is not trivial to assess, as it depends on the exact implementation. In general, it is crucial how well the basic trajectory can cover the sampling space.

With respect to SNR efficiency, a given trajectory is more effective the more the sampling density resembles the shape of the target filter (e.g., Hamming function). The time efficiency is assessed based on how many data points have to be discarded (e.g., data acquired during gradient re-winders because the same trajectory is repeated very often). Furthermore, the re-winder data also complicate the reconstruction.

The applied gradients are not perfect and have some inaccuracies, as stated in the section on *artefacts*. It is also good to know how robust a trajectory is against inhomogeneities (e.g., in the gradient fields). This issue is not easy to answer, but the researchers at the High-Field Centre in Vienna have experimental proof that the CRT performs better on this task than the others. The actual form of the gradients can be assessed via special sequences or via field probes. The field probes measure the spatial and temporal qualities of the magnetic fields, and thus, help to determine the differences between the played out and the desired gradients. This is then used to calibrate the scanner and enable higher resolution. The field probe can consist, for example, of 12 small measuring units evenly distributed over the inner volume of the scanner.

## 2.4 Established offline reconstruction pipeline

The *offline reconstruction pipeline* was the starting point of this thesis. It was developed by Bernhard Strasser. The programming language was MATLAB, short for Matrix Laboratory. The mode of this reconstruction was *offline*, which means it had to be performed on a remote computer. To run this reconstruction, the data had to be first acquired, subsequently downloaded from the Host to a drive or directly to another computer, and finally reconstructed there. Nevertheless, this procedure was firmly established before the beginning of this diploma thesis, and thus, its outputs were used as a reference-standard for the newly developed *online reconstruction*.

The *offline reconstruction* was embedded only on one computer. All its software components were stored in one file directory, as MATLAB demands. Inside this directory, approximately 30 separate MATLAB source-code files (i.e., file format .m) were stored, whereas most of those files were, in fact, custom-created functions. Those files were cross-linked via function calls. The outline of the function-call succession is depicted in Fig. 2.7 where only the relevant functions or .m files are mentioned.

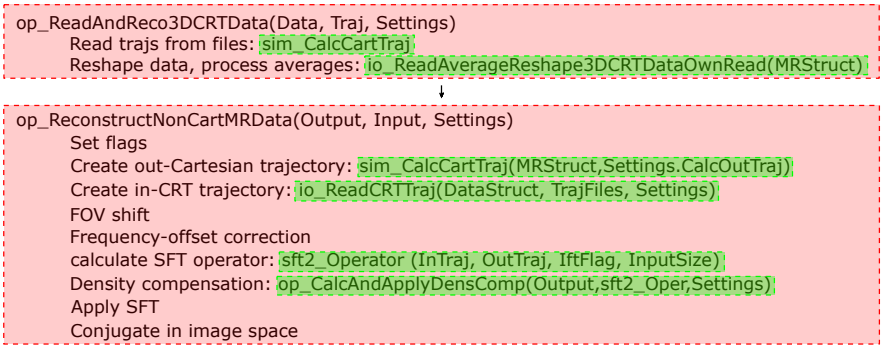


Figure 2.7: Overview of the main steps of the *offline reconstruction pipeline*. The two main .m-files are highlighted in red, and, in green, separate functions called from the superordinate functions.

The two big red rectangles in Fig. 2.7 symbolize two major .m files. All functions were executed in the displayed order, from top to bottom. From each of those functions, the execution of several other functions was triggered, highlighted in green. After performing all their tasks, the code execution continued to the next step inside the big red rectangle.

The reconstruction was triggered from the file *op-ReadAndReco3DCRTData* with the following parameters: *Data*; *Traj* (i.e., trajectory); and *Settings*. Most of its parameters are self-explanatory in which the *Settings* included *flags* determining which data-processing steps were to be performed. Inside the main triggering file, first, the function *sim-CalcCartTraj* was called, which read in the *Cartesian* trajectory. When this function was complete, the program returned to the main file, reshaped the data, and processed the averages in the *io-ReadAverageReshape3DCRTData-OwnRead* function.

When the first red part was complete, the *op-ReconstructNonCartMRData* was called. Inside this function, all the main reconstruction steps were performed or triggered to be performed by separate functions. As a first step, the *flags* were set. After that, the CRT *k-space* and *Cartesian* image trajectories were created. First of the data operations was the *FOV shift*, which was always executed by default (i.e., there was no *flag* to turn it off). On the shifted data, the *frequency-offset correction* was performed. Then followed the calculation of the *DFT* (i.e., *slow FT, SFT*, in Fig. 2.7) operator using the previously computed trajectories as inputs. Then, the *density compensation* was performed in an extra function. At this point, all the *k-space* operations had been applied and the data could be finally converted into the *image domain* by applying the pre-calculated SFT. At the end, the image data were conjugated (i.e., signs of the imaginary part of the complex-valued data were changed) for correct frequency display.

To execute only selected parts of the reconstruction, *flags* and commenting out of code parts were used. In the development process of the *online reconstruction*, only individual data-processing elements were performed. There were *flags* available for turning on and off the *frequency-offset correction* and *density compensation*. The *FOV shift* functionality had no pre-implemented means to turn this on and off, as it was written in a part of the code that was always executed. Thus, to switch it off, the respective part of the code was simply set into a comment (i.e., non-executable part of code).

## 2.5 Tools for the online reconstruction

The set of tools used for the *online reconstruction pipeline* is called the Image Computation Environment (ICE) and it was provided by Siemens (Erlangen, Germany). The ICE comprises all processes and programs that are connected to the data acquisition and reconstruction. A part of the ICE is the ICE *framework* that provides the developing environment for ICE *programs*. The ICE *framework* comes with pre-defined objects and functions that are implemented in the ICE

*library*. The cornerstones of the reconstruction are *pipe services* and *functors* [41].

The programming language used within the ICE *framework* is C++, which allows a wide range of functionality and provides good control over the implementation (e.g., memory management) and performance.

### 2.5.1 ICE program

The ICE *program* represents the reconstruction part of a sequence and describes the *pipeline*. Its dedicated file type is .ipr (i.e., ICE *program*). This file contains the names of the ICE *configurators* to be used and also the applicable dynamic link libraries (i.e., the names of the .dll files). The configuration of the pipeline happens on the Host (see Fig. 2.2) and its output is a file with the suffix .IRIS. By opening the .IRIS file in a special program called *XBuilder*, the composition of the pipeline with all its ICE *pipe services* and *functors* can be viewed. This was often used at the development stage to check for the correct succession of the data operations.

The *pipe services* work as containers for *functors* and manage incoming and outgoing events. When multi-threading (i.e., simultaneous execution of several operations) is in place, data synchronization mechanisms are necessary. The *pipe services* ensure that the incoming data are processed in the right order. This is achieved at the transition between two *pipe services*, in particular, where all the data are synchronized (i.e., the data exit the *pipe service* in the exact same order in which they came in) [41].

The reconstruction as such is then executed in a process called *MrIrisContainer.exe*, which runs on the imager (i.e., MRIR in Fig. 2.2). The instructions are brought from the Host to the MRIR via the above-mentioned ethernet connection [41].

Thus, first, to create an ICE *program*, the .ipr file is written where the *configurators* (i.e., one or more default and one custom made) are activated. Second, the *configurator* is implemented to describe how the pipeline should be assembled from *pipe services* and the therein contained *functors*. Third, the data-processing algorithms are implemented in the *functors* [41]. To be able to compile the ICE *program* successfully, two further file types must be implemented, a .trs file and an object map.

The .trs file states all the source code files (i.e., file type .cpp) to be used. Those include the *configurator*, all *functor* source code files, and the object map file. Also applied ICE *libraries* and custom-built libraries are stated in the .trs file. In the object map file, which is formally a .cpp file, all the used source-code header files (i.e., suffix .h) of the *configurator* and *functors* are included and an object is declared for each of them.

### 2.5.2 ICE configurator

The ICE *configurator* sets up the reconstruction pipeline as a dynamic link library and runs only on Windows OS. There are default *configurators* provided by the ICE *framework* that create a standard pipeline [41]. To implement the CRT reconstruction, a new *configurator* was created to edit the default pipeline, and new *functors* were created and added into the standard pipeline.

As stated above, the ICE *configurator* comprises a header, an .h file, and a source code, a .cpp file. In the header file, the *configurator* is defined, necessary elements of the ICE architecture are included, and a class of the *configurator* type is declared. The *configurator* class includes the *Compose* method, which is of an *IResult* data type, a boolean value.

The *configurator* source file implements the *Compose* method. In this method, all configurations of the pipeline are performed. Those include removing of unwanted *functors*, finding existing *pipe services* and *functors*, setting their properties, inserting custom-made *functors* into the already existing or newly created *pipe services*, and linking them to the correct *functors* that are already in place. In this manner, all the pipeline-manipulating functions are a part of the ICE *framework*.

In all the source codes, so-called *ICE-OUTs* were used to generate outputs during the reconstruction. Those outputs were used, for example, to indicate errors at specific spots, which improved the debugging process and code-execution times could be displayed via those outputs. Also, the current

position in the reconstruction was signaled by this means. The advantage of using the *ICE-OUTs* was that, with the programmer-defined message, a time stamp and the source file were also printed [41].

### 2.5.3 ICE functor

The ICE *functor* is an element of the image calculation pipeline. The *functors* receive data via events, perform reconstruction procedures on them, and dispatch the processed data again via events to subsequent *functors* [41]. There were two basic types of *functors* used for the reconstruction, *scan* and *image functors*.

The raw data entered the pipeline in the same way it was measured, in the form of scans or ADCs. The dedicated functor type for their processing is the *scan functor*. This *functor* type is entered only by one scan at a time. Thus, to process all the scans, the *functor* is executed repeatedly, once for each scan. To differentiate the scans from one another, they were marked with identifying indices. One of those indices was, for example, the *evaluation info mask* that identified the type of scan. Only the *ONLINE* scans (i.e., the spectroscopic data, not the calibration data) were used in this thesis. Also, a *line index* was used to designate different ring radii, $k_r$. Finally, the ICE *dimension indices* were also used to store information about the scan index on a given ring, current TI, maximum TIs on that ring, and also, the number of $k_\varphi$ points on the ring.

When the scan-by-scan processing was finished, all the scans were collected by the ICE *SpecLooper functor* and a single *image object* was created that contained all of the measured data. This multi-dimensional data object was constructed according to the predefined amount of dimensions and their extent. The *functors* operating on those data objects are called *image functors*. Each of those *functors* was entered by this whole object and the programmer could work with the whole measured data set at the same time, which was the reason for implementing most of the *functors* in this type.

Every *functor*, regardless of its type, had *properties* assigned to it. A *functor property* is an input parameter that can be edited outside the *functor*, mostly in the *configurator*. This was a very useful feature of the ICE *framework* and was used often. Those *properties* had to be declared in the *functor*-class definition via a *property map* method. Also, a *get/ set* method was used to make those *properties* accessible.

As was the case for the *configurator*, each of the ICE *functors* was implemented in two files, a header file and a source code. Both the files had a clearly defined structure and mandatory elements and methods. In the header file, the following actions were performed: comments were written describing the *functor's* operation and a *functor* class was defined with all the *functor properties*, *get/ set* methods, constructor, destructor, and method declarations. *Scan functor* methods were *BeginInit*, *EndInit*, *FirstCall*, *ComputeScan*, and *endOfJob*. The methods of the *image functors* were identical with only one exception, a *ComputeImage* method rather than a *ComputeScan*. Each of those methods was a member of the *functor* class.

All those methods had to be implemented in the *functor's* .cpp file, whereby most of the methods did virtually nothing and almost all the functionality was performed by the *Compute* methods, either on scans or images. There were three parameters of the *Compute* methods. An *access specifier*, *IceAs*, which was a pointer on the data, header, *MdhProxy*, for scans that synchronized the sequence with the data acquisition, and *MiniHeader* for images, both of which were pointers on the data header. The third parameter was a control structure *ScanControl* or *ImageControl* for the two *functor* types, respectively, which were likewise passed on as pointers. The measurement data header contained, for example, loop counters and *orientation data* [41]. The *slice orientation data* (SODA) will be of importance in the Section on ICE *objects, dimensions, and access specifiers*.

The main procedure to write our own *functors* was the following. First, it had to be decided in which *functor* method the functionality must be performed. Second, all necessary ICE *objects* were created, followed by the ICE *access specifiers* pointing to those *objects*. Then, the ICE *functions* or custom-made algorithms, or both, in combination, were performed. When all data operations of that *functor* had been finalized, the data, the header, and the control parameters were sent to the successor [41].

To familiarize ourselves with the data processing inside the pre-implemented ICE *functors*, only their accessible header files were studied. The source codes of the ICE *functors* were not available to the programmer. Nevertheless, in the header files, there was always a brief description of how the functor operated. Those descriptions included only a rough outline, and sometimes, a trial-and-error approach, in connection with reverse engineering, was necessary to determine, what, exactly the *functor* did.

### 2.5.4 ICE objects, dimensions, and access specifiers

For the implementation of the *functors*, the ICE *framework* provides the following well-defined objects: ICE *objects* and *access specifiers*. The ICE *objects* serve the purpose of storing data. They are designed as smart pointers, which means that they self-destruct when they are no longer referenced.

The ICE *object* was created inside the *functor's* .cpp file in two steps. First, it was declared and then initialized using a firmly defined form for both. Through the initialization, ICE *dimensions* were stated in a desired order including their extent (i.e., number of elements). It is important to note that assigning the ICE *object* to a header permitted the sending of pixel data. The maximal extent of the *objects* was limited technically only by the size of the computer's RAM [41]. The *objects* normally contained the measured and processed data, but they could be also created artificially and used, e.g., to define data operators. The operators created in such way could be then easily applied to the data.

The *objects* consisted of ICE *Blocks*, which are contiguous data-storage areas in the computer memory. Each of the *blocks* had its own set of SODAs assigned to it [41]. The *blocks* were made up of at least one ICE *dimension*. Examples of those *dimensions* are COL, used to address the FID points, SEG for $k_\varphi$ points on rings, LIN for ring radii, PAR for different z-partitions, CHA for different receiver-coil elements, and SET for averages. There were, in total, sixteen *dimensions* available but the cited were the most important and intensively used ones in this thesis.

The smallest data unit inside the ICE *framework* was a *pixel*, which stored the actual value, mostly of a complex-float data type. Each of the *pixels* was stored at a precise location in the memory and its value could be thereby retrieved, as will be shortly discussed. The *pixels* possessed one of the two *pixel properties* that were tightly connected to the ICE *dimensions* and determined the memory layout and thereby the access possibilities. The *pixel properties* were FIX (i.e., fixated) and VAR (i.e., variable). One of the two *pixel properties* was rigidly connected to each of the *dimensions* and could not be changed by the user. The FIX *pixel property* organized the data of those FIX-*dimensions* into one ICE *Block*. Conversely, each element of a VAR-*dimension* was stored in a different *block*, and therefore, also in a different area in memory. From all the above-mentioned *dimensions* (i.e., COL, SEG, LIN, PAR, CHA, and SET) only the PAR dimension was of the *pixel property* VAR and all the others were FIX. That means that each partition was stored in a different area in memory and all the other dimensions in one partition were located in one contiguous memory area.

This allowed incremental logic to be used for systematic data access, through direct memory addressing. This approach offered significantly faster data handling, but it was crucial to carefully adhere to the correct order of data access. Therefore, it was important to know that the succession of the ICE *dimensions* in the memory was determined by the exact order in which the *dimensions* were stated in the initialization of the ICE *access specifier*.

The ICE *access specifier* enabled access to the data of the *objects*. One *access specifier* was linked to only one *object* while one *object* could have several *access specifiers* associated with it. The *access specifier* could either cover the whole range of the *object's* data or just a defined part of it that was stated during its initialization. During the initialization of the *access specifier*, the boundaries and the validity of the respective *object* were checked [41]. The *access specifier* was also created in two identical steps just like the *object* (i.e., declaration and initialization).

There were several methods in which to handle the *access specifiers*. After an *access specifier* was created it could be re-initialized or modified, in which the re-initialization affected the whole *access*

*specifier* and the modification performed changes only on the mentioned *dimensions*. Furthermore, the length of any of the *dimensions* could be retrieved. The most essential method applicable to the *access specifier* was *calcSplObjStartAddr*. This method calculated the starting address of the first data *pixel* of a given *access specifier*. This method was used frequently for direct access to the data. The *access specifiers* made it easy to apply pre-defined ICE *functions* to the data. Those were useful algorithms contained in the ICE *library*. The structure that enabled the *functions* to be performed on *access specifiers* is called an ICE *wrapper* with a firmly defined structure, shown in Listing 2.1.

```
Ice.function(access specifier/ specifiers);
```

Listing 2.1: General definition of an ICE *wrapper* enabling the use of predefined *functions*.

The ICE *wrapper* applied the specified *function* on the given *access specifier* or *specifiers*. The number of *access specifiers* demanded as input for the *wrapper* depended on the interface of the *function*. The *wrapper* could be applied only to data lying in one contiguous area in memory. Therefore, to apply one *wrapper* to several partitions, which were of the *pixel property* VAR and located in different areas in memory, it had to be called the same number of times as there were partitions. During the call of each ICE *wrapper*, a calculation of the SODA was performed. This ensured proper handling of data positions and orientations and correct labeling of images [41]. The SODA is a set of slice-specific parameters. It described all the data lying in one ICE *Block*. Some of the stored SODA parameters were the following: slice position, slice normal, slice thickness, pixel spacing, and whether the given parameter set was valid or not, in which all of the vectors were based on the patient sagittal-coronal-transversal coordinate system. The SODAs were extracted and calculated from the data header automatically by the ICE *framework*.

When a new *object* was created (e.g., for some data-processing operator), the SODAs were created but not filled so the parameter set was invalid. But, objects coming from previous functors carried valid SODA information with them, which could be transferred to the newly created object, rendering it valid. As this procedure was not supervised by the ICE *framework*, the programmer had to know what he or she was doing, otherwise, reconstruction errors could emerge. When predefined *functions* were used, the SODAs were automatically passed to the resulting *objects* as part of each *function's* SODA calculation [41].

### 2.5.5 Running an ICE program on the scanner

When the ICE *program* was completed, there were certain steps necessary to run it. As shown in Table 2.1 (see Subsection 2.1), the imager was running on Linux and the Host on Windows OS. Therefore, it was necessary to create binaries for both the systems. Binaries are files that can be directly installed. To create those, the ICE *program*, as well as the *configurator* and *functor* files, were all jointly compiled. The resulting binaries were a .dll and an .evp (i.e., type library) for the Windows Host and a .so library for the Linux imager, as illustrated in Fig. 2.8.

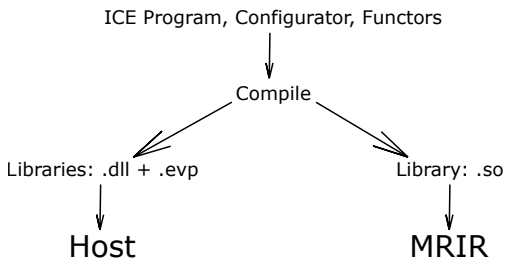


Figure 2.8: Series of steps to run the ICE *program*. Edited from [41].

The final libraries were then copied from the compiling computer into their destined location (i.e., either on the Host or MRIR). With that, the *program* was in place and ready to be run on the scanner.

It is also interesting to mention the most important details of the compiling process. This was done on a remote computer where the *program* was developed. All the testing and compiling was done in a virtual box (VB) where all the ICE infrastructure was installed. In this VB, there were two virtual machines (VM), one with Windows, named IDEA, and the other with Linux OS, called MARS. The compiling was triggered from a command prompt on the IDEA VM. To create all the binaries (i.e., .dll, .evp, and .so) the MARS also had to be turned on, as it participated in the compiling process.

### 2.5.6 Data files

The following three data-file types were used in the reconstruction process. The first file type, the raw data, which was the result of the acquisition on scanner, was the .dat file. The .dat file also included the measurement header.

When the reconstruction was performed online, on the scanner, either during the acquisition or after it (i.e., retro reconstruction) an .ima file was the output in both cases. This file type is also called DICOM as was mentioned earlier. Regarding the third file type, the ICE reconstruction could also be performed offline, on a remote computer in the VB, which was mostly the case during the development stage. Thus, the resulting data file was a .spe file.

How the images were plotted from those two reconstructed data files will be discussed in the Section on *Displaying the data*.

## 2.6 Data-reconstruction procedures

In this Section, the fundamental mathematical formalisms of all implemented data-processing steps will be written down and described. The first two subsections will describe data pre-processing (i.e., data selection and rearrangement), and then three subsections will explain data editing in *k-space*, and, ultimately, the conversion of the data into *image space*.

### 2.6.1 Selection of scans

The selection of only the desired scans was accomplished in the following way. The discrimination of the data subsets was possible according to the header parameter *evaluation info mask*. The sequence recorded the following subsets of the data: *ONLINE*, *NOISEADJ*, and *PATREF*, in which only the *ONLINE* data was used for the reconstruction in this thesis. The current subset of the data was determined in the scan *functor*. The selection of the scans was achieved via a succession of logical conditions. The *NOISEADJ* and *PATREF* scans were immediately discarded. Only the *ONLINE* scans were kept, and some operations were performed on them.

With those scans came the header information on the coordinates of the first point on each ring. The $k_x$ and $k_y$ coordinates were comma value, but, because the interface did not allow storing of comma values, they were systematically converted into integers in the acquisition and, for the reconstruction, they had to be calculated back into the original values. There was also a range limitation for the numbers, it was not possible to store negative values in the acquisition, which was also specially handled in the reconstruction.

The negative values from the acquisition were written into the header as larger than 60,000. This was used in the reconstruction to recognize negative numbers. When this condition was fulfilled, the operation in Eq. 2.3 was used for the pre-comma y-coordinate, $y_{pre-comma}$, using the ICE *header parameter* 2, $H_{p2}$.

$$y_{pre-comma} = H_{p2} - 65,536 \tag{2.3}$$

When the post-comma part of the y-coordinate was negative, it was stored in the sequence as larger than 9,999. This number was again used as a condition for the recognition of negative post-comma numbers. With Eq. 2.4, the exact value of the negative post-comma number was determined using the value stored in the third *header parameter*. This procedure was necessary for numbers with a zero pre-comma part.

$$y_{post-comma} = H_{p3} - 65,536 \tag{2.4}$$

When the recognition and calculation of the negative numbers were complete the pre- and post-comma parts of each of the two coordinates (i.e., x and y) could be put together according to:

$$y = y_{pre-comma} + x_{post-comma}/10,000 \tag{2.5}$$

The procedure for the x-coordinates was equivalent to that of the y-coordinates. How those coordinates were forwarded to the successor *functors*, together with other essential header parameters, will be described in the Section on *Implementation of the CRT reconstruction*.

### 2.6.2 Data rearrangement and temporal interleaving

The rearrangement of the data, including the *temporal interleaving*, occurred in the first *image functor* according to the equations given in this Subsection. The sorting of the measured *ONLINE* scans into a systematic *k-space* was done based on two main equations that determined the indices of the point on ring $k_\varphi$ and FID point $k_t$.

The index $k_\varphi$ was calculated by Eq. 2.6 based on a data-point index, $i_{dp}$, which counted through all the data points (i.e., pixels) of all scans on one ring, the number of FID points (i.e., number of time data points) $Nt$, the number of points on ring, $N\varphi$, the current TI of each of the scans, $currTI$, and the maximum number of TIs on each ring, $maxTI$.

$$k_\varphi = (i_{dp} - \frac{Nt \cdot N\varphi \cdot currTI}{maxTI})\%N\varphi \tag{2.6}$$

The $currTI$ equaled 0 for TI1, 1 for TI2, and 2 for TI3. The % symbol in Eq. 2.6 is the modulo operator, which gives the residual of an integer division.

The FID point index was determined using Eq. 2.7 with the same input values as for Eq. 2.6.

$$k_t = (i_{dp} - \frac{Nt \cdot N\varphi \cdot currTI}{maxTI}) \cdot \frac{maxTI}{N\varphi} + currTI \tag{2.7}$$

Both equations were derived in this thesis based on the observation of the data structure and the conversion relationship between the measured scans and the systematic *k-space*.

The sorted data was then sent to the successor together with the previously calculated $k_x$ and $k_y$ coordinates of the first points and the $maxTI$ on each ring.

### 2.6.3 Field-of-view shift

To achieve a shift of the FOV in the *image domain*, a linear phase must be multiplied on the signal in *k-space*. But first, a conversion between coordinate systems is necessary, from the scanner to the patient coordinate system, as those mostly do not coincide with each other. They are shifted relative to each other as a result of translation, and tilted, as a result of rotation.

First, the position vector of the scanner, $\vec{p}_{scanner}$, was assembled from the header scanner-position-parameters $p_{scanner,cor}$, $p_{scanner,sag}$, and $p_{scanner,tra}$. Next, the normal vector of the scanner, $\vec{n}_{scanner}$, was also created from header information. The patient normal vector, $\vec{n}_{patient}$, was defined as a unit vector in the transversal direction. Then, the cross-product vector, $\vec{c}$, between $\vec{n}_{scanner}$ and $\vec{n}_{patient}$ was calculated. The resulting $\vec{c}$ was thus perpendicular to both vectors. The vector $\vec{c}$ was then extended into a matrix $C$ according to Eq. 2.8.

$$C = \begin{pmatrix} 0 & -c_0 & c_1 \\ c_2 & 0 & -c_0 \\ -c_1 & c_0 & 0 \end{pmatrix} \tag{2.8}$$

The matrix $C$ was then squared according to linear-algebra rules, giving $C_{squared}$. Also, the dot product, $d$, of $\vec{n}_{scanner}$ and $\vec{n}_{patient}$ was calculated. The last necessary element was the identity matrix, $I$. All the above-prepared elements were then used to calculate the rotational matrix, $R$, in Eq. 2.9.

$$R = I + C + d \cdot C_{squared} \tag{2.9}$$

Using the matrix $R$, the missing patient position vector, $\vec{p}_{patient}$, was calculated.

To be able to determine the *FOV shift* operator, the x- and y-coordinates of all points on all rings in *k-space* were necessary. The *Cartesian* coordinates, $k_x$ and $k_y$, of the first point on each ring, which were forwarded from the *data rearrangement* step, were taken and converted into the polar coordinates, $k_r$ and $k_\varphi$, by Equations 2.10 and 2.11.

$$k_r = \sqrt{k_x^2 + k_y^2} \tag{2.10}$$

$$k_\varphi = atan2(k_y, k_x) \tag{2.11}$$

To determine all the points on each ring, an angle increment, $\Delta\varphi$, was determined from $N\varphi$:

$$\Delta\varphi = \frac{2\pi}{N\varphi}$$

By incrementally adding $\Delta\varphi$ to the first points on each ring, all points on all rings were calculated, in polar coordinates. The polar points were then converted back into *Cartesian* coordinates, because this coordinate system was used for further calculations, via Eqs. 2.12.

$$k_x = k_r \cdot cos(k_\varphi) \tag{2.12}$$

$$k_y = k_r \cdot sin(k_\varphi)$$

To calculate the *FOV shift* operators, two factors were necessary: the number of lines in the image, $Nl$, and the gyromagnetic ratio, $\gamma$. The entries of the *FOV shift* operator, in the x-direction, $O_{FOV,x}$, were found by Eq. 2.13.

$$O_{FOV,x} = e^{2\pi i \cdot Nl \cdot \gamma \cdot k_x \cdot (-\vec{p}_{patient,sag})} \tag{2.13}$$

In Eq. 2.13 and 2.14, $i$ is the imaginary unit. The operator in the y-direction resulted from Eq. 2.14.

$$O_{FOV,y} = e^{2\pi i \cdot Nl \cdot \gamma \cdot k_y \cdot \vec{p}_{patient,cor}} \tag{2.14}$$

The two operators were then multiplied resulting in an *FOV shift* operator in both the in-plane directions, $O_{FOV}$, Eq. 2.15.

$$O_{FOV} = O_{FOV,x} \cdot O_{FOV,y} \tag{2.15}$$

Finally, the application of the operator to the *k-space* data, $S(\vec{k})$, was done again via an element-wise multiplication, see Eq. 2.16.

$$S'(\vec{k}) = O_{FOV} \cdot S(\vec{k}) \tag{2.16}$$

This resulted in the shifted data, $S'(\vec{k})$.

### 2.6.4 Frequency-offset correction

The *frequency-offset correction* operator was determined by the accumulated phase, and therefore, depended on the FID point, the point on ring, and on the number of TIs on the respective ring.

To prepare the relevant components for the creation of the operator, the maximum TI values on each ring were retrieved into a vector, $maxTI(k_r)$.

Next, the time offset vector, $\vec{t}(k_\varphi)$, was prepared. This was only dependent on the $k_\varphi$ coordinate. Therefore, its value was simply set to 1 for the first point on the ring, 2 for the second point on the ring and so on.

After that, the frequency-offset vector, $\vec{f}(k_t)$, as a function of the FID point, $k_t$, was calculated using the number of FID points, $Nt$, the number of points on ring, $N\varphi$, and the $maxTI(k_r)$, Eq. 2.17.

$$\vec{f}(k_t) = (\frac{k_t}{Nt} + 0.5 \cdot (\frac{1}{Nt} - 1)) \cdot \frac{maxTI(k_r)}{N\varphi} \tag{2.17}$$

The two offset vectors, $\vec{t}(k_\varphi)$ and $\vec{f}(k_t)$, were later combined, therefore, they had to be extended for the other missing dimension. The time offset vector, $\vec{t}(k_\varphi)$, was extended by the time dimension $k_t$ into a 2D matrix, $T(k_\varphi, k_t)$, in which the time offset remained dependent only on the position on the ring. The point-on-ring dimension, $k_\varphi$, had to be added to the frequency-offset vector, $\vec{f}(k_t)$, which resulted in a 2D matrix, $F(k_\varphi, k_t)$, still varying only with the time.

With those two 2D matrices, the entries $x(k_\varphi, k_t)$ of the *frequency-offset correction* operator, $O_{corr}$, could be calculated according to Eq. 2.18.

$$x(k_\varphi, k_t) = e^{-2\pi i \cdot T(k_\varphi, k_t) \cdot F(k_\varphi, k_t)} \tag{2.18}$$

The resulting entries were conjugated to deliver correct operator values, $O_{corr}$, Eq. 2.19.

$$O_{corr} = \bar{x}(k_\varphi, k_t) \tag{2.19}$$

The resultant operator had to be applied on the FIDs in the spectral domain, not in the time in which they were located. Thus, a FFT was first performed on each FID, transforming it into the spectral domain.

As in the CRT sequence, FIDs and not echoes were recorded, a *fft shift* in the spectral dimension was necessary. The *fft shift* did practically nothing else except flipping the first half of the data points in a spectrum with the second half. In this manner, the zero-frequency component was shifted to the center of the spectrum [43].

Now, the *frequency-offset* operator could be applied on the data via an element-by-element multiplication resulting in corrected data in the spectral domain. To transform the FIDs back into the time domain, as further data operations were still to be performed in that domain, the *fft shift* was applied to bring the data into the same order as they were before the first *fft shift*. Finally, an inverse FFT was performed on each FID, transforming it into the desired time domain so the data could be sent to the next step.

### 2.6.5 Density compensation

When sampling *k-space* along a *Cartesian* trajectory, the measurement of one line lasts one TR. In contrast to that, when CRT sampling is applied, during one TR, one ring is sampled regardless of its radius, whereas different rings have different radii, and therefore, cover different *k-space* areas. Because of this, the sampling density is non-uniform, as shown in Fig. 2.6 and it must be compensated by a *density compensation* function, $\xi$. For determining $\xi$, first, the density function, $\rho$, was determined.

The $\rho$ of *k-space* points on individual rings was determined. This was done by multiplying the number of points on the ring, $N\varphi$, by the time difference between adjacent $k_\varphi$, $\Delta t$, and divided by the surface area assigned to the respective ring, $A_r$, Eq. 2.20.

$$\rho = \frac{N\varphi\Delta t}{A_r} \tag{2.20}$$

Eq. 2.20 could be simplified by substituting $\Delta t$ with 1 because it was constant throughout the whole sequence. Furthermore, for $N\varphi$, likewise, the constant 1 could be inserted because in the *OldADC* version of the CRT sequence, which was used, the number of points per ring was also invariable. After those simplifications, Eq. 2.20 was reduced to Eq. 2.21.

$$\rho = \frac{1}{A_r} \tag{2.21}$$

From this expression $\xi$ could be derived with Eq. 2.22.

$$\xi = \frac{1}{\rho} \tag{2.22}$$

Resulting into Eq. 2.23 for the *density compensation* function.

$$\xi(r) = A_r \tag{2.23}$$

The *density compensation* areas, $A_r$, for three innermost rings are depicted in Fig. 2.9.

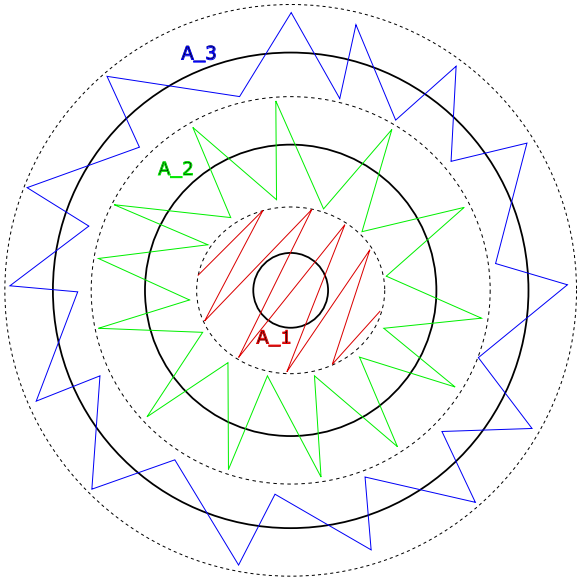


Figure 2.9: Calculation scheme of *density compensation* areas for the *concentric ring* trajectory in k-space.

To calculate those areas, the first points on each ring were retrieved and scaled by the gyromagnetic ratio and FOV. The ring radii, drawn as full black lines in Fig. 2.9, were then determined using Eq. 2.10. Next, the ring radii were normalized by the radius of the largest ring.

After that, the radii of the density compensation areas (i.e., $A_1$, $A_2$, and $A_3$ in Fig. 2.9) were calculated. The radii of the *density compensation* areas, the dashed lines, were determined as distances between the common ring center and radii exactly between two adjacent rings.

The areas, $A_r$, were determined according to Fig. 2.9. The area, $A_1$, of the inner-most ring (red) was calculated as a full circle and all the other areas (i.e., $A_2$ and $A_3$ in this example) were calculated in a donut-like shape.

To apply $\xi(r)$, it was multiplied on the *k-space* data, $S(\vec{k})$, see Eq. 2.24.

$$S'(\vec{k}) = A_r \cdot S(\vec{k}) \tag{2.24}$$

The result of Eq. 2.24 was the *density compensated* data, $S'(\vec{k})$.

### 2.6.6 Fourier transformation

The FT implemented in this thesis was the *spatial DFT* for transforming the data from *k-space* into the *image domain*. For the CRT reconstruction, only the in-plane *DFT* was implemented, as this pipeline was able to handle only 2D data sets (i.e., with only one partition). Nevertheless, the *Cartesian* reconstruction was implemented for volumetric data. There, the in-plane *DFT* was performed in one step and the remaining spatial transformation in the z-direction was performed after that, whereas the principles of the *DFT* are identical regardless of the spatial direction. Those principles will be reviewed here on the example of the xy-*DFT* for CRT.

The definition of the FT, already given in Eq. 1.36, shows all the needed elements:

$$S_{image,n} = \sum^{m} S_{kspace,m} \cdot e^{-2\pi i < \vec{k}_m, \vec{r}_n >}$$

In this equation, $n$ is an index in the *image domain*, and $m$ is a *k-space* index. The definition shows that a given image pixel is calculated as a sum over all *k-space* pixels weighted by the exponential term dependent on the signal-origin, $\vec{k}_m$, and signal-destination, $\vec{r}_n$, vectors. The *image* position vector, $\vec{r}_n$, resembled coordinates of a full square stretching between $\vec{r}_{min}$ and $\vec{r}_{max}$ in both spatial directions (i.e., x and y):

$$\vec{r}_{min} = (-\frac{N}{2} + 0.5)f \tag{2.25}$$

$$\vec{r}_{max} = (\frac{N}{2} - 0.5)f$$

In Eqs. 2.25, $N$ symbolizes the number of pixels in an image line (i.e., the matrix size). All the image coordinates were multiplied by a factor, $f$, which was composed of the gyromagnetic ratio and the FOV. The image positions between *min* and *max* were created in equidistant steps. The *k-space* position vector, $\vec{k}_m$, was calculated in the same way as described in the Section on *FOV shift*.

When both position vectors have been established, the exponential term in Eq. 1.36 (i.e., the discretization) was pre-calculated for all $m$ and $n$ values. This pre-calculation was not necessary, however useful, since the results of it were used in the subsequent code repetitively. Thus, the pre-calculation was performed for optimization purposes. The operation in the cited equation was performed for each FID point separately (i.e., for each element in the time dimension). Thus, the complete dataset was transformed into the *image domain*.

The calculation of the *k-space* trajectory for the *DFT* in the *offline reconstruction pipeline* differed from the *online* version. In the *offline reconstruction*, the whole gradient functions, $G_x(t)$ and $G_y(t)$, were discretely integrated, by a rectangular method, resulting in gradient-moment functions, $GM_x(t)$ and $GM_y(t)$, which were then converted to *k-space* coordinates, according to Eq. 2.2.

On the other hand, in the *online reconstruction*, only the first point on each ring was calculated via the discrete integration from the discrete-valued gradients, $G_x(t)$ and $G_y(t)$, at time points t = 0. The gradients were also integrated according to the rectangular method to give the gradient moments. The gradient moments were then converted into *k-space* coordinates in the same way as in the *offline reconstruction*. All remaining points on the circular trajectory were calculated by converting the first $k_x$ and $k_y$ coordinates into polar coordinates, $k_r$ and $k_\varphi$, from this, then, via an angle increment, $\Delta\varphi$, all the other points on that ring were calculated in polar coordinates, and finally convert back into $k_x$ and $k_y$.

## 2.7 Displaying the data

Whether the data was processed on a PC (i.e., offline) or directly on the scanner (i.e., online) determined the format of the output file where the reconstructed data was stored. Each of those two file types were structured in a different way and had to be read out in a different way. MATLAB scripts were used to display the processed data from both reconstructions. In the case of *offline* reconstruction, the output was a .spe file. Those files were displayed using a single MATALB script. From the *online* reconstruction emerged an .ima file (i.e., DICOM). To display the data from this file, it had to be opened by a whole set of MATLAB scripts.

### 2.7.1 Offline reconstruction

In the *established offline reconstruction* pipeline developed by Bernhard Strasser in MATLAB, the data was displayed directly after the processing, as it was still found in the MATLAB script where it was reconstructed. In the ICE *offline reconstruction*, the data was processed in the IDEA virtual box. From there, the data was copied onto the physical computer where it was read into MATLAB using a script from Bernhard Strasser.

First, the name of the .spe file was written into a variable. Then, the file was opened and read out into a data variable according to its dimensions (i.e., FID points and the matrix size). After that, the file could be closed. The data in the variable was reshaped and the dimensions were separated. Then, the real and imaginary components of the complex-valued data had to be squeezed together because they were stored in the .spe file separately. After that, the data was ready to be used for plotting and further evaluations. The following plots were viewed to check the quality of the results: absolute, real, and imaginary images after the application of selected data-processing steps.

At the beginning, the data was plotted only after performing the *DFT*. As more and more data operations were implemented, the data was also displayed for *FOV shift*, *frequency-offset correction*, and *density compensation*, in which the *DFT* was always applied to transform the data into *image space*. No data quality checks were done on *k-space* data after the applied processing steps. The data were plotted in the form of subplots for better overview and comparability. The MATLAB function used for plotting was *imagesc*, which resulted in color-scaled images from blue to yellow.

### 2.7.2 Online reconstruction

The output file type of the *online reconstruction* was DICOM. The DICOM file had to be viewed differently than the .spe file. To view this, a set of MATLAB scripts previously written by colleagues was used. The read-in of the reconstructed data was triggered from a script called *ReadDicomsScript* in which the data was also then available for display and analysis. From this script, the *io-ReadAndReshapeSiemensData* function was called, which recognized the software version of the reconstruction and the used sequence. From this function, the actual data-reading script was called *read-csi-dicom*. The *read-csi-dicom* function first retrieved the dimensions of the data from the header. Then, the data could be read from the .ima file into a variable. Finally, the data was ordered into a neat *data object* according to its dimensions. This data object, together with the header, was then propagated back into the triggering function.

## 2.8 Implementation of the Cartesian reconstruction

In this Section, the *Cartesian reconstruction pipeline* will be described, from the programming point of view, based on the concepts and mathematical equations introduced above. This reconstruction can process multi-slice data sets, as well as multi-channel acquisition, in which the *coil combination* was performed in one of the ICE *functors*. The only *functor* that was implemented for the *Cartesian* trajectory was the *spatial discrete Fourier transformation*.

### 2.8.1 Default ICE pipeline

The *pipe services* and therein contained *functors* of the original ICE reconstruction are depicted in Fig. 2.10.

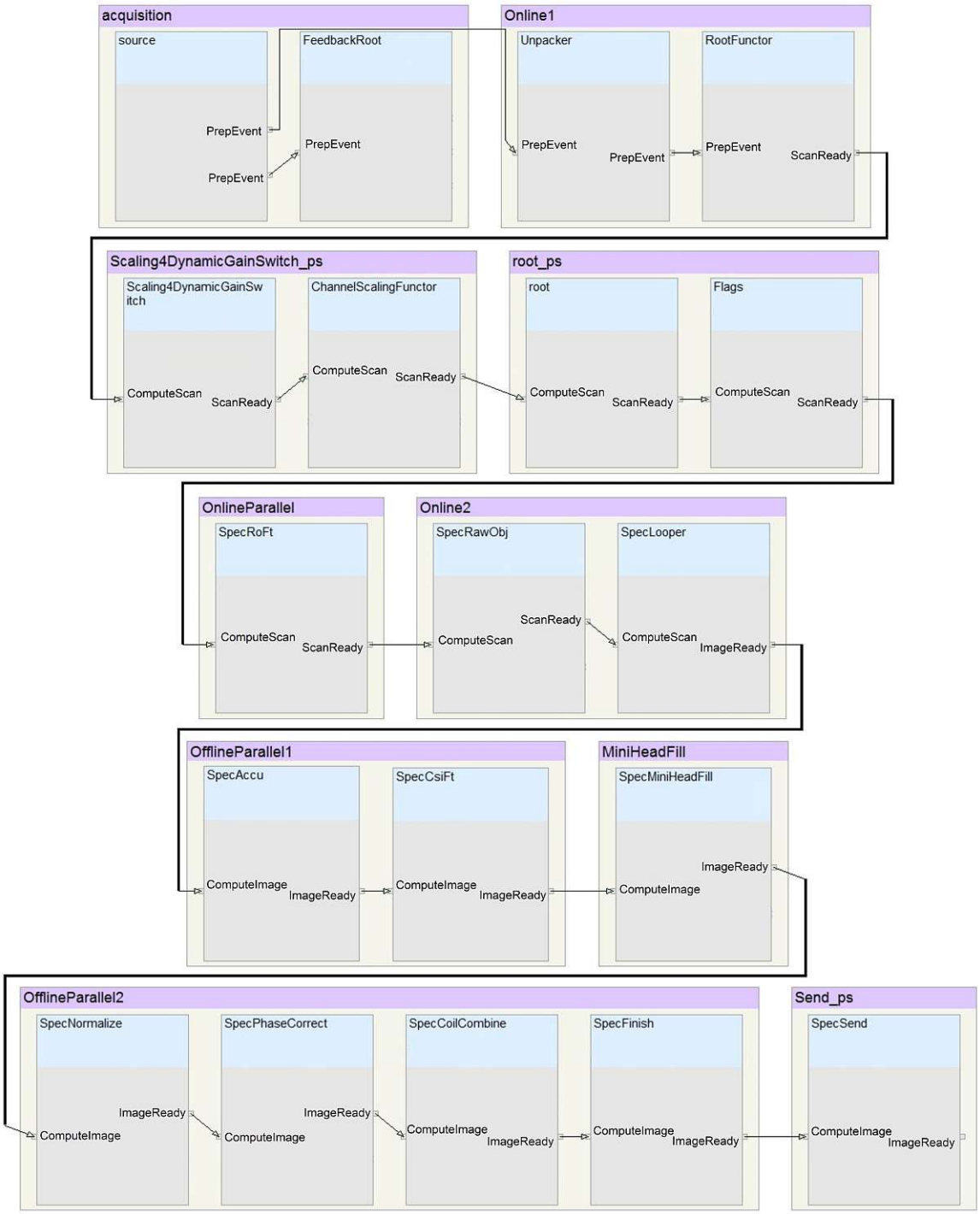


Figure 2.10: Scheme of the default ICE *pipeline* for the *Cartesian reconstruction* with its *pipe services*, *functors*, events, and connections between them.

The names of the *pipe services* are written above the group of *functors*, on a violet background. On the blue background in Fig. 2.10, the *functor* names are shown. The incoming and outgoing *functor events* are displayed on a gray background, inside each *functor*. The diverse *events* of the first four *functors* were replaced by *PrepEvent* (i.e., preparative *event*), for simplification and elimination of unnecessary details. The non-simplified image of the *Cartesian* default pipeline can be viewed in the supplementary file to this thesis, on page 2, where also the threading (i.e., how many operation threads were performed simultaneously, depending on the number of logical cores) of the individual *pipe services* is depicted.

This standard pipeline can process data, measured exclusively along a *Cartesian* trajectory. Recall that at the end of each *pipe service*, not each *functor*, the data was synchronized. This, together with the *threading*, determined the grouping of the *functors*.

*Functors*, interesting for our purposes, were the following: *Flags*; *SpecRawObj*; *SpecLooper*; *SpecMiniHeadFill*; *SpecFinish*; and *SpecSend*, as all of them could be also used for the CRT *reconstruction* pipeline, and will be, therefore, described in the following subsection. Those *functors* were responsible for the structural workings of the program. In contrast to the structural, the functional tasks were performed by *SpecCsiFt* (i.e., *spatial Fourier transformation*), *SpecPhaseCorrect* (i.e., *frequency-offset correction*), and *SpecCoilCombine*. Except for the *SpecCoilCombine functor*, which performed the *coil combination*, equivalents for the other two *functors* were implemented for the CRT *reconstruction*, as will be explained later.

The part of the pipeline in Fig. 2.10, built by *scan functors*, starts with *RootFunctor* and goes until *SpecLooper*, where the transition to *image functors* was made. The rest of the pipeline consisted of *image functors*. The type of each *functor* can be recognized from its incoming and outgoing events: *ComputeScan* and *ScanReady* signifying *scan functors*, and *ComputeImage* and *ImageReady* an *image functor*.

### 2.8.2 Modified pipeline

The original pipeline, in Fig. 2.10, was edited in only one aspect. The ICE *Fourier transformation* was replaced by a custom-built *functor*, *CartesianFT*, which performed the same function. Only the adjusted part of the pipeline is depicted in Fig. 2.11. This figure shows only the simplified version of the pipeline. The full version can be viewed in the supplementary material, on page 3.

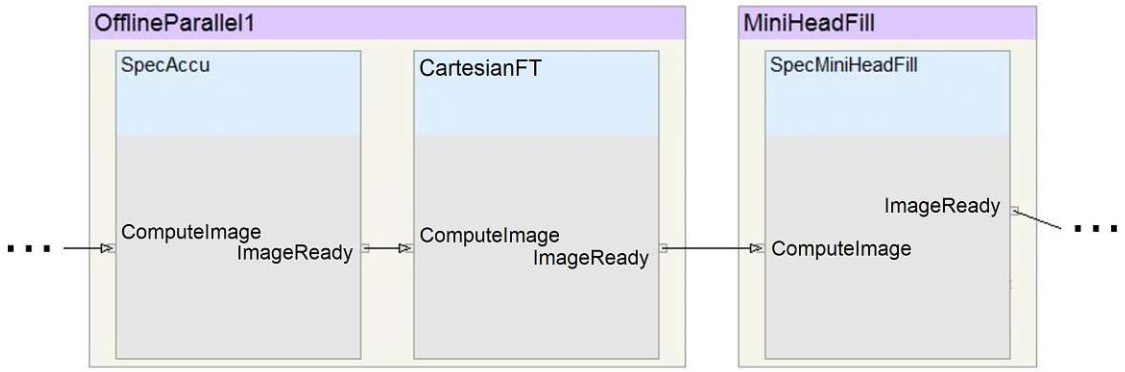


Figure 2.11: Part of the *Cartesian reconstruction* pipeline, which was modified from the default, with all the other *pipe services* and *functors* in an identical arrangement as in Fig. 2.10.

In the following subsection, the implementation of the custom-made *Fourier transformation* will be reviewed.

### 2.8.3 Fourier transformation

As explained above, for training and verification purposes, the *spatial Fourier transformation functor* was first implemented for the *Cartesian* trajectory. This was done in the *CartesianFT functor*, which replaced *SpecCsiFt*. The results from the original pipeline served as the ground-truth, to which the results of the own (i.e., custom-made) *functor* were compared.

The own version was implemented using two different solutions. The main difference between the two solutions was the data-access strategy. In *solution 1*, the data was accessed via *pixel* addresses that were directly incremented in the computer memory. On the other hand, in *solution 2*, the *access specifiers* were modified to arrive at the desired pixels and the address of each *pixel* was retrieved using the ICE *function, calcSplObjStartAddr()*.

#### Solution 1

The *spatial DFT* implemented according to *solution 1* will be described here, in its most elementary form, i.e., only for 2D spectroscopic data sets, measured with a single receiver element. This algorithm was also implemented to handle multi-channel (i.e., multiple receiver-coil elements) acquisitions with multiple slices, but, for the sake of simplicity, it will not be explained in full, here. The extension for the other *dimensions* was guided by the same principles as in the basic variant.

Prior to the code in Listing 2.2, some preparations were necessary. First, the coordinates of all the measured data points (i.e., the input *Cartesian* trajectory), $k_x$ and $k_y$, were calculated. Second, the *image* points, $r_x$ and $r_y$, were determined. Next, the discretization, $discret_{xy}$, was pre-calculated, for the reasons stated above, and stored in a 2D array.

```
1  for (int r_xy = 0; r_xy < (Nx·Ny); r_xy++) {
2    for (int k_t = 0; k_t < Nt; k_t++) {
3      for (int k_xy = 0; k_xy < (Nx·Ny − 1); k_xy++) {
4        *outputPixel = *outputPixel + *srcPixel · discret_xy[r_xy][k_xy];
5        srcPixel += Nt;
6      }
7      srcPixel = srcPixel − (Nx·Ny − 1) · Nt;
8      srcPixel++;
9      *outputPixel = conj(*outputPixel);
10     outputPixel++;
11   }
12   srcPixel = (std::complex<float>*)srcAs.calcSplObjStartAddr();
13 }
```

Listing 2.2: Algorithm of the basic variant of the Cartesian *spatial discrete Fourier transformation, solution 1*.

A graphic, using colors to illustrate the succession of memory access in the *k-space*, as well as in the *image object*, is depicted in Fig. 2.12.

The different memory increments are color-coded as follows: green (i.e., ++ or +1) shifted the memory access to the directly following address; red (i.e., $+N_t$) skipped all the vector size points and enabled access to the next *k-space* position, in which $k_x$ was incremented first, and when all x-coordinates were done, automatically shifted to the next column (i.e., next $k_y$ and the first $k_x$). The blue line (i.e., -($N_x * N_y$ - 1) · $N_t$) returned to the starting in-plane position of the current vector size point, in the *data matrix*.

When all in-plane (i.e., $k_x$ and $k_y$) and all $k_t$ data points were processed, the memory access was set back to start, the orange address increment, line 12.

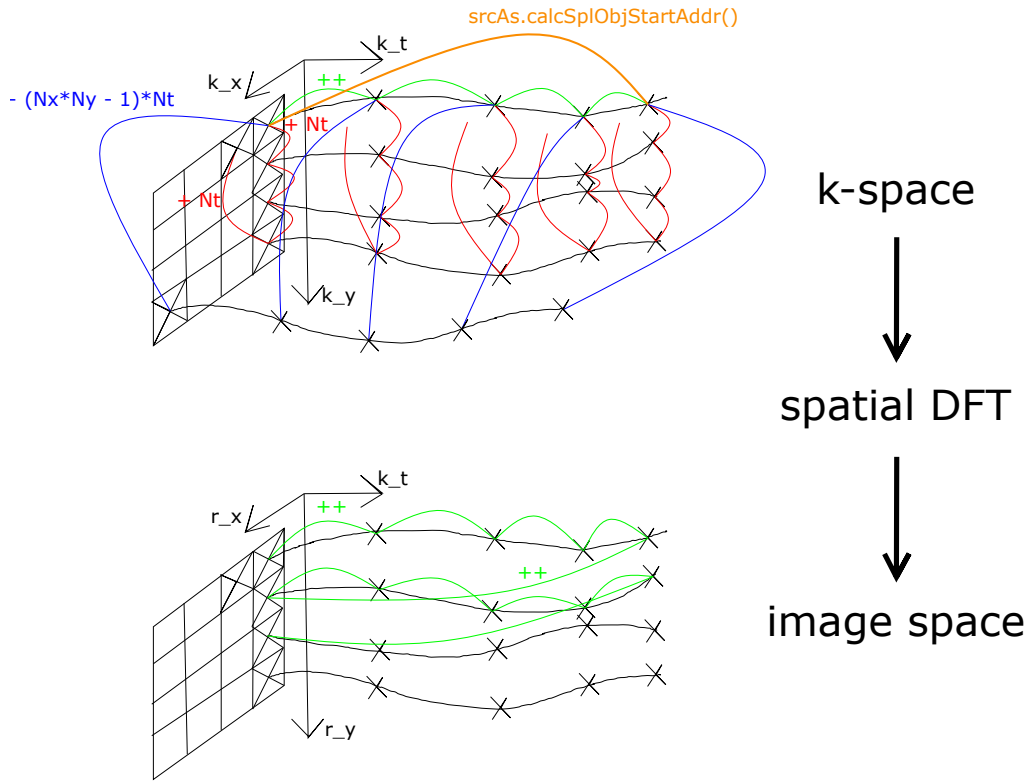


Figure 2.12: Illustration of the data access and the memory address increments, in the *k-space* and *image object*.

The data access of the *image object* was different from its *k-space* counterpart. With a simple increment (i.e., ++), in green, all memory addresses were accessed, one after the other, filling the *image object* with the accumulated values, line 9.

As can be seen in Listing 2.2, the *k-space* contributions to the current *image pixel* were accumulated on the *image-pixel* address, *outputPixel*, see line 4 in Listing 2.2. The successive addressing of all in-plane *k-space pixels* was achieved via the increment in line 5, by the red memory-address increment in Fig. 2.12. When the contributions of all *k-space* points were accumulated in the *image* point, the current address in the *data matrix* was set back to indices $k_x = 0$ and $k_y = 0$, by the blue address increment, line 7 in Listing 2.2. To switch to the next FID point, the *source pixel*, *srcPixel*, was incremented by one, the green address increment in the upper part of Fig. 2.12, line 8. Before the algorithm moved to the next *image pixel* the accumulated *image* value was conjugated. After that, the *image pixel* address was incremented by one, the green line in the lower part of Fig. 2.12.

When all FID points on one *image* position were processed, the *k-space pixel* address was shifted to the starting position, the orange address increment, line 12, to perform the same set of operations for all other *image* positions.

Note that, in Listing 2.2, the calculation of the starting pixel position of the *source access specifier*, *srcAs*, was determined only once for each ring, line 12, and that the *access specifier* was not modified at all in *solution 1*, which significantly decreased the computational demand and accelerated the reconstruction, with respect to *solution 2*.

### Solution 2

On the other hand, *solution 2* did not directly work with the memory-address increments. Instead, the *access specifiers* were modified, dimension by dimension, as the loops were progressively opened

and the memory addresses were retrieved from it afterward. The outline of *solution 2* is shown in Listing 2.3. This solution will be described also for only the simplest scenario, covering all in-plane pixels and the spectral *dimension* to it. The implementation for multiple channels and slices will not be depicted, but was implemented in this thesis. The preparatory steps prior to the code in Listing 2.3 were the same as for *solution 1*.

```
1  for (int kt = 0; kt < Nt; kt++) {
2    srcAs.modify (COL, kt, 1, 1);
3    outAs.modify (COL, kt, 1, 1);
4    for (int rx = 0; rx < Nx; rx++) {
5      outAs.modify (LIN, rx, 1, 1);
6      for (int ry = 0; ry < Ny; ry++) {
7        outAs.modify (SEG, ry, 1, 1);
8        for (int kx = 0; kx < Nx; kx++) {
9          srcAs.modify (LIN, kx, 1, 1);
10         for (int ky = 0; ky < Ny; ky++) {
11           srcAs.modify (SEG, ky, 1, 1);
12           srcPixel=(std::complex<float>*)srcAs.calcSplObjStartAddr();
13           dummy_xy = dummy_xy + *srcPixel · discret_xy[rx][ry][kx][ky];
14         }
15       }
16       outPixel=(std::complex<float>*)outAs.calcSplObjStartAddr();
17       *outPixel = conj(dummy_xy);
18       dummy_xy = 0.0;
19     }
20   }
21 }
```

Listing 2.3: Implementation of the *spatial DFT* for the *Cartesian* trajectory, solution 2.

In contrast to *solution 1*, in *solution 2*, the *source*, as well as the *output access specifier*, was modified, in all of its *dimensions*, to arrive at the desired memory address, which was then retrieved for each *k-space pixel*, line 12 in Listing 2.3. This was accomplished, by the call of the *calcSplObjStartAddr function*, much more often than in the *first solution*, as the *source pixel* address was retrieved exactly $Nt \cdot Nx^2 \cdot Ny^2$ times, compared to $Nx \cdot Ny$ times in *solution 1*. This had, as a consequence, a considerably higher computational demand, which was also caused by the steady modifications of both *access specifiers*.

When the address of the *source pixel* was retrieved, its contribution to the *image pixel* was calculated and subsequently accumulated into an auxiliary variable, $dummy_{xy}$.

When all the *k-space* contributions were summed in the auxiliary variable, the address of the destined *image pixel* was extracted, line 16, and the accumulated value was conjugated and written to the memory address, line 17. After that, it was essential to reset the auxiliary variable, and thereby, prepare it for the next *image* position.

The *output access specifier*, filled with the calculated *image* data, was sent, together with the *data header* and the *control structure*, to the next *functor*.

## 2.9 Implementation of the CRT reconstruction

The six *functors*, mathematically described in the Section on *Data-reconstruction procedures*, will be reviewed here with respect to their programming aspects. For data access, only *solution 2*, which modified the *access specifiers*, was utilized for the implementation of the CRT reconstruction. *Solution 2* was explained in detail in the previous section on *DFT* for a *Cartesian* trajectory. This solution was selected for its robustness and easier scalability, when compared to *solution 1*.

In the final pipeline for the *concentric ring* trajectory, the data was not conjugated in the

beginning (i.e., before the first user-created *functor*), as this was done in one of the ICE *functors*. The data was conjugated after the last user *functor*, which was the *spatial DFT*. This was also the case for the *Cartesian* reconstruction.

The development of the *online reconstruction* stimulated a few formal changes in the CRT sequence. Those changes occurred only in the header parameters, which were necessary as inputs for the reconstruction, or made the reconstruction more robust. An example of this was the number of data *pixels* per measured ADC, or the vector size (i.e., number of data *pixels* per assembled FID). Those and other parameters were added into the output header of the sequence to develop the reconstruction with as much flexibility as possible, without any hard-coded parameters. The sequence was edited by a colleague. When the changes were made, new raw data, including the header, were measured. The reconstruction was then accordingly adjusted.

As was mentioned earlier, for debugging purposes, an *offline ICE reconstruction* was performed, at first, to make sure that all changes were implemented correctly. When the reconstruction was working on a PC, it was then tested on the scanner.

### 2.9.1 Default ICE pipeline

The simplified ICE default pipeline, used for the CRT reconstruction, is depicted in Fig. 2.13. For the full version of this pipeline, consult the supplementary file, on page 4. This pipeline was created automatically by the ICE *configurator IrisDefaultConfigurator*, from the ICE *library IrisConfigurators*. The CRT default pipeline included a considerably lower number of elements than the default pipeline for the *Cartesian* trajectory, because ICE was not able to process data measured on a *non-Cartesian* trajectory, such as the CRT, and therefore, the processing chain was interrupted after the *root functor*. Nevertheless, this pipeline was sufficient as a starting point and was significantly modified to achieve the desired results. All its major modifications will be described in the next subsection.

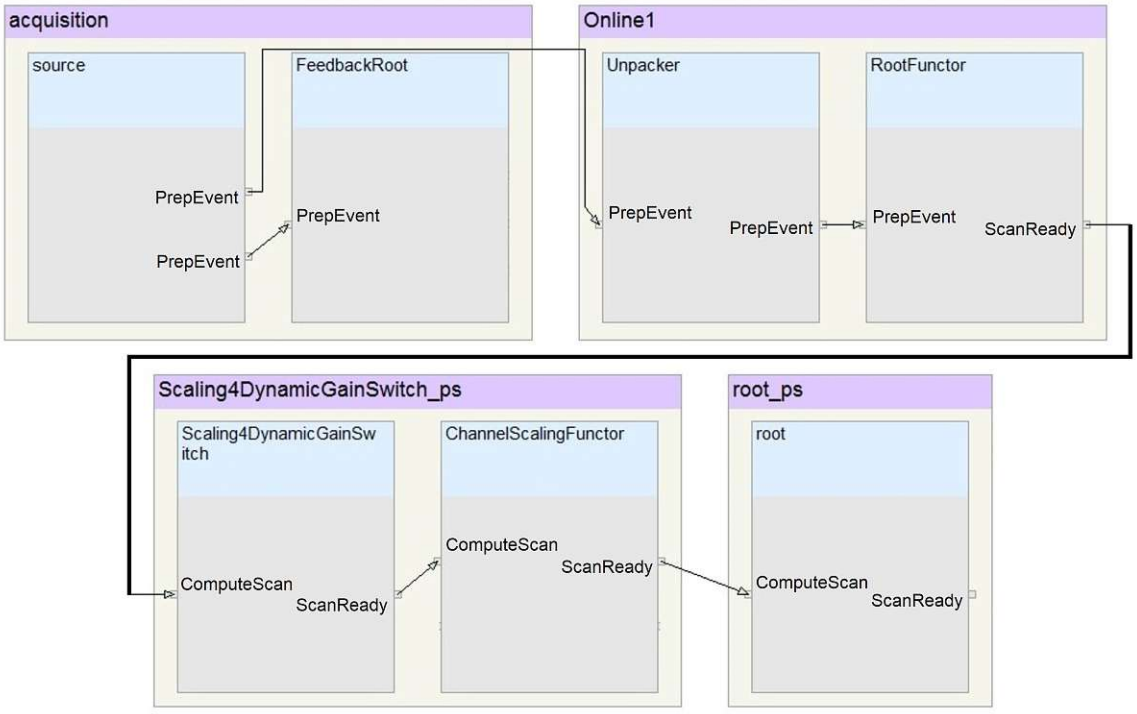


Figure 2.13: Diagram of the default ICE *pipeline* for the CRT reconstruction showing its *pipe services* and *functors*.

### 2.9.2 Modified pipeline

To implement all of the desired functionality, another ICE *configurator* was joined to the *program* and custom-created *functors* were added. The simplified form of the final CRT *reconstruction* pipeline is shown in Fig. 2.14. The non-simplified version is depicted on page 5 of the supplementary material.

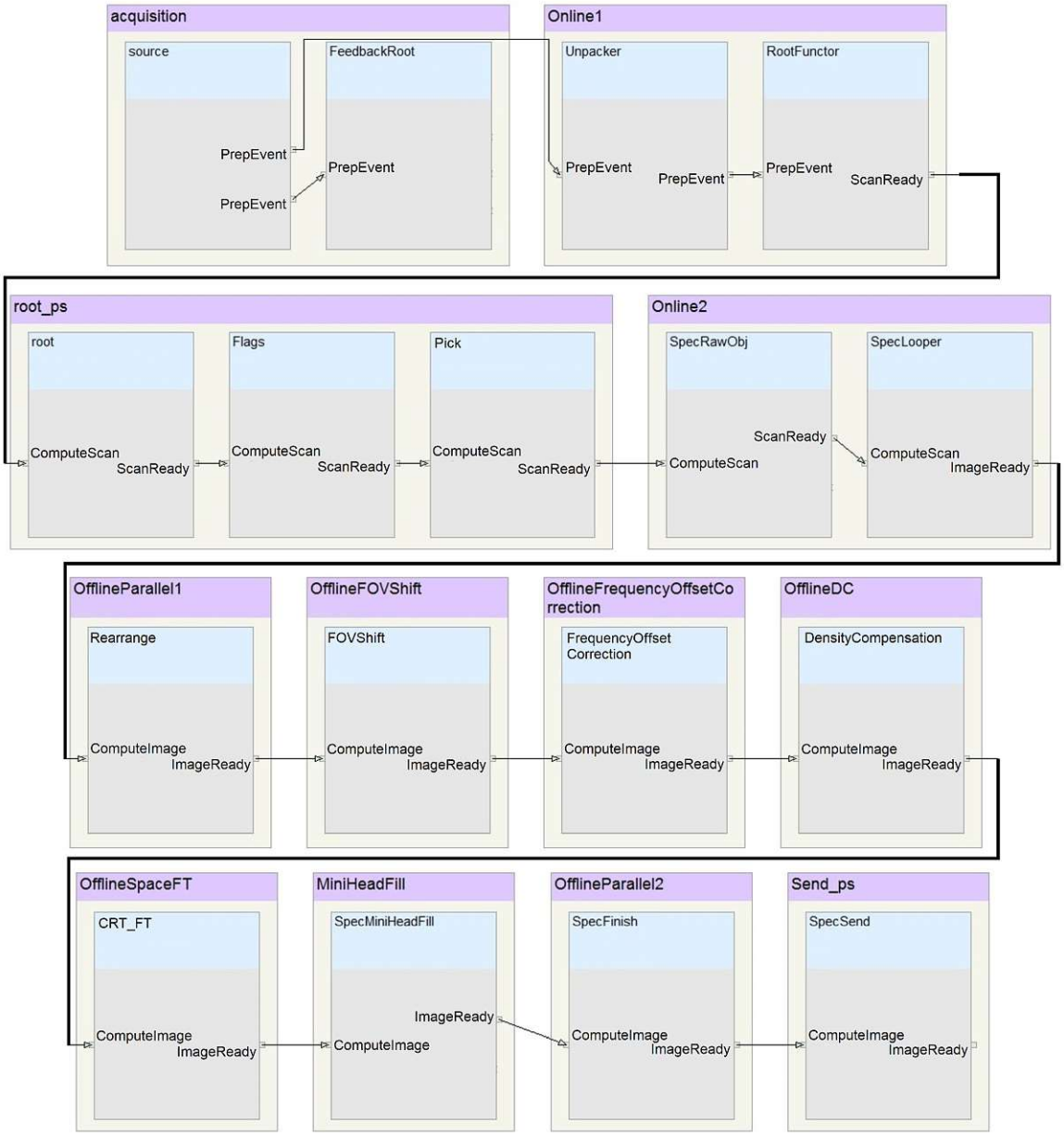


Figure 2.14: Configuration of the modified pipeline for the CRT reconstruction.

As can be seen from the comparison of Fig. 2.13 to 2.14, the following default *functors* were kept: *source*; *FeedbackRoot*; *Unpacker*; *RootFunctor*; and *root*. Both *functors* in the *Scaling4DynamicGainSwitch-ps pipe service* were removed, as they scaled only the measured values. The *functors source* and *FeedbackRoot* were a part of the acquisition process, bringing the data into the reconstruction pipeline. The *Unpacker functor*, together with *RootFunctor*, from the *pipe service Online1*,

instantiated the following structures: *data header*; *data object*; slice orientation data; *ScanControl* (i.e., a *control object* specific to *scan functors*); and an *access specifier* linked to the ICE *object*. Finally, the *root functor* provided compatibility with all the subsequent pipeline elements.

To arrive at the modified pipeline, in Fig. 2.14, from the default version, in Fig. 2.13, the ICE *configurator IceSpectroFConfigurator*, from the ICE *library IceSpectroF*, was added. This was done in the .ipr file. This *configurator* added the following ICE *functors* into the pipeline: *Flags*; *SpecRoFt*; *SpecRawObj*; *SpecLooper*; *SpecAccu*; *SpecCsiFt*; *SpecMiniHeadFill*; *SpecFinish*; and *SpecSend*.

A custom-made *CRT configurator*, from the *CRT library*, was also added in the .ipr file. All the actions described from here, until the end of this Subsection, were implemented in the *CRT configurator*. First, unwanted ICE *functors* were removed from the pipeline. Those were: *Scaling4DynmicGainSwitch*; *ChannelScalingFunctor*; *SpecRoFt*; *SpecAccu*; and *SpecCsiFt*. The reason for removing those *functors* was that their functionality was not compatible with the CRT reconstruction. One example of a compatibility issue was the *SpecCsiFt functor*, which implemented the *spatial FT* for a *Cartesian* trajectory and was unable to handle the CRT data. Therefore, it had to be removed.

The *Flags functor* detected and stored signaling information from the header (e.g., timestamps of the first and the last scan in a slice). This information was later used for ICE's housekeeping. After *Flags*, the *Pick functor* was inserted, which did the scan selection and processed the coordinates of the first points on each ring. Next, for each repetition, a *raw data object* was created, by the *SpecRawObj functor*. The different repetitions were stored in the AVE *dimension*. The created *objects* were then exported to be used by the subsequent *functors*. The *dimensions* of the *raw data objects* were determined by incoming boundaries, forwarded as *functor* properties: *RawCol* (i.e., number of elements in the COL *dimension* of the *raw data object*); *RawLin* (i.e., the same but for the LIN *dimension*); and *RawSeg*; etc. To change the extent of the data *objects*, those properties were adjusted.

The transition in the pipeline from *scan* to *image functors* was done inside the *SpecLooper*, which is demonstrated in Fig. 2.14 on its incoming, *ComputeScan*, and outgoing, *ImageReady*, event. When the pipeline call-behaviour was altered to the *image functor* modus, the rest of the custom-made *functors* could be performed. Those inserted *functors* were: *Rearrange*; *FOVShift*; *FrequencyOffsetCorrection*; *DensityCompensation*; and *CRT-FT*. The programming of those *functors* will be described in the next subsections.

After the chain of custom-made *functors*, the ICE *functor*, *SpecMiniHeadFill*, removed some standard parameters and, in return, added some spectroscopy parameters. The *functor SpecFinish* converted the data, from the scanner into the DICOM coordinate system, mainly by placing the *dimensions* COL and LIN on the first and second place. This operation was necessary for the ICE *Send* functionality. Finally, the data was sent to the Host by the *SpecSend functor* and the whole *functor* chain was terminated.

The *properties* of the ICE *functors*, *SpecRawObj* and *SpecLooper*, had to be changed to perform desired actions, and guarantee compatibility. In the *SpecRawObj functor*, five of its properties were changed. The number of elements along the COL *dimension* was changed by the *RawCol property*. The extent of this *dimension* was edited to store all the ADC points, plus six additional data pixels, in which two of the six data pixels were used to propagate the x- and y-coordinates of the first points on each ring. The remaining four were used to store: current-ADC index; current-TI index; number of $k_\varphi$ points; and the number of maximum TIs on a given ring. The LIN *dimension* encoded the individual rings, $k_r$, and the number of elements of this *dimension* was set by the *RawLin property*. For compatibility, also, the *PhaseResolution property* was set to the same value as *RawLin*. The *dimension* SEG was used to store all the measured ADCs per one ring. The number of ADCs per ring was conveyed through the *RawSeg* property, which had to be synchronized with the property *ReadResolution*.

The edited *functor property* in the *SpecLooper functor* was *NLin*, where the number of rings was stored and was set to the same value as *RawLin* and *ReadResolution*, of the *SpecRawObj functor*. All those *functor properties* were set automatically using the header information.

To switch the execution of the own *functors* on and off, a boolean *property*, *PowerButton*, was implemented. Thus, in the own *configurator*, the *functors FOVShift*, *FrequencyOffsetCorrection*, and *DensityCompensation* could be easily set *true* to perform the data-processing operations, and *false* to forward the unchanged input data. The data operations, in *functors Pick*, *Rearrange* and *CRT-FT*, were always performed.

### 2.9.3 Selection of scans

For the *selection of scans*, first, it had to be recognized to which data-subset (i.e., calibration or MRSI data) the current scan belonged. This was accomplished through a series of logical conditions, based on the *evaluation info masks* retrieved from the *header*. When a calibration scan was recognized, all other operations on the current scan were skipped and the next scan came in.

When an MRSI scan was registered, the four parameters to calculate the x- and y-coordinates on the first point on each ring (i.e., $x_{pre-comma}$, $x_{post-comma}$, $y_{pre-comma}$, and $y_{post-comma}$) were retrieved from the header. The conversion of the stored integer values was done according to Equations 2.3 - 2.5.

Next, for each scan, an extra *scan object* was created with six additional data pixels to store the calculated x- and y-coordinates and the other four parameters, described earlier. After that, an *access specifier* to this *object* was created. To render the *object* valid, slice orientation data were copied from the incoming *object*. This was the standard procedure for creating ICE *object*, *access specifier*, and SODAs, which was used in every *functor*.

Another procedure, used in every *functor* was handling of the *pixel* data (i.e., reading and writing from and into ICE *objects*). This procedure is very important because the whole reconstruction was based on it. First, the *pixel*, *pPixel*, was defined, as can be seen in Listing 2.4.

```
std::complex<float>* pPixel;
```

Listing 2.4: Declaration of a *pixel* object, *pPixel*, which was used to read out and overwrite the data.

The star in Listing 2.4 designated that it was formally a pointer to a data address in the computer memory. The data address was retrieved via the command in Listing 2.5.

```
pPixel = (std::complex<float>*) accessSpecifier.calcSplObjStartAddr();
```

Listing 2.5: Initialization of the previously defined *pixel* object by applying a predefined ICE *function* to an *access specifier*.

The initialization was done using the *calcSplObjStartAddr() function* and applying it to the desired *access specifier*. As this *function* retrieved only the address of the starting *pixel*, the *access specifier* had to be precisely modified, before this call. An example of how the COL *dimension* (i.e., individual data points) of the *access specifier* was modified to access the data at a desired location, 100, is shown in Listing 2.6.

```
accessSpecifier.modify(COL, 100, 1, 1);
```

Listing 2.6: Modification of *access specifier* via an ICE *function*, specifically of the *pixel dimension* of the data, setting the access to position 100.

The parameters of the *modify function* must be set in a precise order: *dimension* - *access start* - *access length* - *access increment*. For a controlled data access, the parameters *length* and *increment* were always set to one. With this, the data *pixel* was ready to be used. When the *pixel* object was written with a star in front of it, the actual value on the memory address was accessed, however, when written without the star, the address itself was used.

As the *Pick functor* was the last one before the assembly of an *image object*, containing all the *scans*, some measures had to be taken for the object to be created correctly. Most of this was already done in the sequence, by entering the right header parameters. Yet, one important point remained, to set the SEG index. The SEG index was used by the following *SpecRawObj functor* to number the measured ADCs on each ring. For this purpose, the following line of code was used.

```
aMdh.setCseg(currIda + adcPerCircle / currIdd * currIdb);
```

Listing 2.7: Setting the ADC index as a parameter of an ICE *function*.

The *setCseg method* set the SEG index, according to the given parameter. This value was calculated as is depicted in Listing 2.7, from *currIda* (i.e., current ADC index, which was restarted for each TI), *adcPerCircle* (i.e., total number of ADCs per ring), *currIdd* (i.e., current maximum TI), and *currIdb* (i.e., current TI). After that, a series of finalizing steps was taken, which was the same for all the other custom-made *functors*. First, the x- and y-coordinates of the first points on each ring, together with the current maximum TI index, were written after the data of the newly created *object*. Second, the *access specifier* was modified to comprise the whole data *object*. This was necessary to forward all the data. As in the *functor*, the *access specifiers* were steadily modified with an *access length* of one, starting at various locations. Third, the total number of elements in each *dimension* of the data *object* was set to the right value, in the respective *control object*. Last, but not least, the data *object*, together with the *header* and the *control object*, were sent to the next *functor*.

### 2.9.4 Data rearrangement and temporal interleaving

The *Rearrange functor* was the first *image functor* in the CRT pipeline. All the following *functors* were of the *image* type, as well, as is depicted in Fig. 2.14. The input from the predecessor was a two-dimensional data *object*, consisting of ADCs and the therein contained data points, in the same order, in which the data was acquired. This *functor* rearranged the measured ADCs into a systematic *k-space*, with the *dimensions*: LIN (i.e., rings, $k_r$); SEG (i.e., points on ring, $k_\varphi$); and COL (i.e., FID points, $k_{t,out}$). The rearrangement also considered *temporal interleaving*. This was achieved mainly by determining the indices, $k_\varphi$ and $k_{t,out}$, from the running data-point index, $i_{dp}$, using Equations 2.6 and 2.7 from a previous section. The $k_r$ index was already known from the header. The algorithm outline is shown in Listing 2.8.

```
1  for (int k_r = 0; k_r < Nr; k_r++) {
2    srcAs.modify(LIN, k_r, 1, 1);
3    outAs.modify(LIN, k_r, 1, 1);
4    i_dp = 0;
5    for (int k_ADC = 0; k_ADC < N_ADC; k_ADC++) {
6      srcAs.modify(SEG, k_ADC, 1, 1);
7      //currentTI, currentMaxTI
8      for (int k_t,src = 0; k_t,src < samplesInScan; k_t,src++) {
9        srcAs.modify(COL, k_t,src, 1, 1);
10       srcPixel = (std::complex<float>*) srcAs.calcSplObjStartAddr();
11       k_φ = (i_dp - (Nt_dst * N_φ * currentTI) / currentMaxTI);
12       k_t,out = Eq. 3.7
13       outAs.modify(SEG, k_φ, 1, 1);
14       outAs.modify(COL, k_t,out, 1, 1);
15       outPixel = (std::complex<float>*)outAs.calcSplObjStartAddr();
16       *outPixel = *srcPixel;
17       i_dp++;
18     }
19   }
20 }
```

Listing 2.8: Implementation of the CRT-data rearrangement, including *temporal interleaves*. In line 7, the *currentTI* and *currentMaxTI* indices were retrieved. In line 12, the calculation of $k_{t,out}$ is replaced by an equation reference because of limited space.

Notice that some dimensions of the *source*, *srcAs*, and *output access specifier*, *outAs*, were modified with the same index, while other dimensions with different indices. The reason for this was the different structure of the two underlying *data objects* and their *access specifiers*. The *access*

*specifiers* were modified such, that the data was always read out and stored in the desired memory address, only. The readout from *source* and overwrite of the *output pixel* is depicted in line 16, Listing 2.8.

The data point index, $i_{dp}$, was incremented after performing all the commands of the innermost loop, line 17. What is also interesting to point out, is the place where $i_{dp}$ was reset. This was done in line 4, directly inside the loop over the rings, before any other loop was started. That means that this index ran through all the data points, i.e., all $k_{ADC}$ and $k_{t,src}$, of each ring, and was set back to zero after that to perform the same for all rings.

The rearrangement of the data was implemented using the steadily incremented $i_{dp}$ because, with the derived equations for $k_\varphi$ and $k_{t,out}$, it was the most efficient and robust way to handle this.

The nesting of the counted loops was chosen for the implementation of the *data rearrangement*, and also of all other *functors*, because the exact number of the data points, as well as the number of elements in each of the objects' dimensions was known.

### 2.9.5 Field-of-view shift

In the .h file of the *FOV shift functor*, the following *properties* were defined: FOV in the x- and y-direction; individual elements of the scanner-position; and the scanner-normal vector. All those parameters were extracted from the *header* and their *get/ set methods* were declared.

Then, in the .cpp file, as at the beginning of each other *functor*, an output *object* with its related *access specifier* and SODA was created. The *FOV-shift* values were loaded into the programming environment and the elements of $\vec{p}_{scanner}$ and $\vec{n}_{scanner}$ were assembled into vectors. The cross product, $\vec{c}$, was calculated manually, using vector-element multiplication. The $\vec{c}$ was then extended into a square, 2D cross-product matrix, $C$, according to equations in the corresponding *Data-reconstruction procedures* Section. After that, the matrix $C$ was squared and the dot product, $d$, of $\vec{n}_{scanner}$ and $\vec{n}_{patient}$ was calculated.

An identity matrix, $I$, was then initialized and the rotational matrix, $R$, could be determined, as all its building blocks, stated in Eq. 2.9, were prepared. The calculation of $R$ was implemented with two nested for-loops, Listing 2.9.

```
1 for (int rw = 0; rw < 3; rw++) {
2   for (int cl = 0; cl < 3; cl++) {
3     R[rw][cl] = I[rw][cl] + C[rw][cl] + d · C_squared[rw][cl];
4   }
5 }
```

Listing 2.9: Calculation of the rotational matrix, $R$, from pre-calculated components: $I$; $C$; $d$; and $C_{squared}$.

The abbreviation *rw* and *cl* in Listing 2.9 stand for *rows* and *columns*. With matrix $R$, the unknown patient-position vector, $\vec{p}_{patient}$, could be found, by multiplying the rotational matrix with the scanner-position vector. After that, two *objects* for the *FOV shift* operators were declared with the *dimensions* $k_r$ and $k_\varphi$. The spectral dimension, $k_t$, of these operators was not defined because the *FOV shift* was the same for all the FID points.

Next, all points on all rings were calculated from the stored first points on each ring. The resulting *Cartesian* coordinates were stored in an array, with *dimension* lengths: $Nr$; $N\varphi$; and 2. Thus, the coordinates were indexed by $k_r$ and $k_\varphi$, but, were stored as x- and y-values, which could be retrieved from the third dimension of length 2 (i.e., index 0 for x and index 1 for y). This indexing was chosen for compatibility purposes with the operators.

With that, all ingredients were in place and the *FOV shift operators* could be assembled, according to Eq. 2.13 and 2.14, Listing 2.10.

```
1 for (int k_r = 0; k_r < Nr; k_r++) {
2   fovXAs.modify(LIN, k_r, 1, 1);
```

```
3    fovYAs.modify(LIN, k_r, 1, 1);
4    for (int k_φ = 0; k_φ < Nφ; k_φ++) {
5      fovXAs.modify(SEG, k_φ, 1, 1);
6      fovXPixel = (std::complex<float>*) fovXAs.calcSplObjStartAddr();
7      *fovXPixel = Eq. 3.12
8      fovYAs.modify(SEG, k_φ, 1, 1);
9      fovYPixel = (std::complex<float>*) fovYAs.calcSplObjStartAddr();
10     *fovYPixel = Eq. 3.13
11   }
12 }
```

Listing 2.10: Calculation of the *FOV shift* operators, separately for x- and y-directions. In line XY and YZ, the code was replaced by equation references due to limited space.

The *access specifiers* were first modified. Then, the *pixel* addresses were retrieved. Third, the desired values were written in the right places, lines 7 and 10 in Listing 2.10. Notice that the operators in the x- and y-direction were filled separately. This clear separation was chosen for easier error identification, in the debugging process. A common *FOV-shift* operator, in both in-plane directions simultaneously, could be implemented as well, as the two directions were combined later anyway, as shown in Listing 2.11. The two operators were multiplied with each other, according to Eq. 2.15 and the result was stored in another *access specifier*, $fovAs$.

```
Ice.Mul(fovAs, fovXAs, fovYAs);
```

Listing 2.11: Create of an *FOV shift* operator in both in-plane directions.

For this purpose, an ICE *wrapper*, with the *Mul function* (i.e., multiplication), was utilized, in which the interface of *Mul* performed an element-wise multiplication of $fovXAs$ with $fovYAs$. The resultant operator, $fovAs$, was then applied to the data, as stated in Eq. 2.16, again via an ICE multiplication:

```
Ice.Mul(outputAs, srcAs, fovAs);
```

Listing 2.12: Applying the *FOV shift* operator, in the form of an *access specifier* on the input data, *srcAs*, and storing the result in the *output access specifier*.

In Listing 2.12, the *srcAs* contained forwarded data from the previous *functor*. The result of this multiplication was stored in *outputAs*. In this manner, the function of this *functor* was fulfilled.

To send all the data to the successor *functor*, the *output access specifier*, *data header*, and the *image-control object* the following line of code, in Listing 2.13, was called.

```
ImageReady(outputAs, dataHeader, imageControl);
```

Listing 2.13: Sending the data from one *image functor* to the next, including the *header* and *control object*.

This step was similar for all *functors*, as it was the standard way to propagate the data through the pipeline. The event, used to send the data, was *ImageReady*, as for all other *image functors*, depicted, e.g., in Fig. 2.14.

### 2.9.6 Frequency-offset correction

In the *frequency-offset correction functor*, first, *output*, *intermediate*, and *frequency-offset correction objects* and related structures were created. Then, a vector, holding the maximum numbers of TIs per ring, $maxTI(k_r)$, was retrieved from behind the data. The data processing of this *functor* occurred in a ring-by-ring fashion, which was achieved by an inclusion in one big loop, over all rings.

First, the time, $\vec{t}(k_\varphi)$, and the *frequency-offset* vector, $\vec{f}(k_t)$, according to Eq. 2.17, were initialized. Second, the vectors, $\vec{t}(k_\varphi)$ and $\vec{f}(k_t)$, were extended into 2D arrays by adding the other missing *dimension* into $T(k_\varphi, k_t)$ and $F(k_\varphi, k_t)$, shown in Listings 2.14 and 2.15, respectively.

```
1 for (int k_φ = 0; k_φ < Nφ; k_φ++) {
2   for (int k_t = 0; k_t < Nt; k_t++) {
3     T[k_φ][k_t] = t⃗[k_φ];
4   }
5 }
```

Listing 2.14: Assembly of the *time-offset* array from the *time-offset* vector, which was given as a linearly growing function of the FID points.

Two for-loops were used to simply insert the vector, $\vec{t}(k_\varphi)$, on all positions to create the 2D array, $T(k_\varphi, k_t)$. The matrix, $F(k_\varphi, k_t)$, was implemented in much the same way. But, the difference was, as shown in Listing 2.14, that the outer loop ran over all points on ring, $k_\varphi$, and the inner one over FID points, $k_t$, which was the exact opposite of Listing 2.15, where the order of the loops was the other way round.

```
1 for (int k_t = 0; k_t < Nt; k_t++) {
2   for (int k_φ = 0; k_φ < Nφ; k_φ++) {
3     F[k_φ][k_t] = f⃗[k_t];
4   }
5 }
```

Listing 2.15: Initialization of the *frequency-offset* array by assigning the *frequency-offset* vector to all $k_\varphi$ locations (i.e., points on ring).

The two pieces of code were implemented this way because, in each of the vectors, a different *dimension* was missing.

When the time and frequency arrays were prepared, the entries of the *frequency-offset correction* operator were calculated according to Eq. 2.18, including the conjugation in Eq. 2.19. This was done in two nested loops going through all $k_\varphi$ and $k_t$. Next, an FFT was performed on all the FIDs of the current ring to convert them into spectra, see Listing 2.16.

```
Ice.FT(intermediateAs, srcAs, FT_Normal);
```

Listing 2.16: Using an ICE *wrapper* to perform a normal (i.e., forward), fast Fourier transformation on the FIDs converting them into spectra.

The ICE *FT function* was used to perform a forward FFT on the data (i.e., srcAs) and the result was written in the *intermediate access specifier*. There had to be no explicit *fft shift*, either before or after the spectral FT, because it was done automatically inside ICE *FT*. Finally, the *frequency-offset correction* operator was applied on the spectra (i.e., *intermediateAs*), in the standard way by an ICE multiplication of the operator with the data. When the data was corrected, an inverse FT was performed to transform the data back to the time domain. This procedure is shown in Listing 2.17.

```
\emph{Ice.FT(outputAs, intermediateAs, FT_Inverse);}
```

Listing 2.17: Using an ICE *wrapper* to transform the data from the frequency domain back to the time domain via an inverse (i.e., backward), fast Fourier transformation.

After this step, the major loop, repeated for each ring, was terminated, and the whole process was repeated for the next ring until all rings were processed. The finalizing steps were identical to those of the previous *functors*.

### 2.9.7 Density compensation

In the *density compensation functor*, after the initial steps, the coordinates of the first point on each ring were read out from behind the data and scaled by the gyromagnetic ratio and FOV. From those values, the ring radii, $r_r$, were determined and normalized with the radius of the outermost ring. From $r_r$, the *density compensation* radii, $r_c$, were calculated, as is shown in Listing 2.18.

```
for (int k_r = 0; k_r < (Nr - 1); k_r++) {
  r_c[k_r] = (r_r[k_r + 1] + r_r[k_r]) / 2;
}
```

Listing 2.18: Implementation of the *density compensation* radii, $r_c$, from the ring radii, $r_r$. All but the last *density compensation* radius were calculated, as the last one had to be determined differently.

By this piece of code, all $r_c$ were determined except for the radius of the largest ring. The remaining radius was calculated according to Fig. 2.9 (i.e., the blue area, $A_3$). When all $r_c$ were determined, the resulting *density compensation* areas could be calculated, see Listing 2.19.

```
for (int k_r = 1; k_r < Nr; k_r++) {
  A_r[k_r] = π · (r_c^2[k_r] - r_c^2[k_r - 1]);
}
```

Listing 2.19: Calculation of all but the innermost *density compensation* area.

The implementation in Listing 2.19 gave all $A_r$, except the first one (i.e., of the innermost ring) because the first *density compensation* area was calculated separately, as the full innermost ring. Then, the *density compensation* operator was applied to the data.

### 2.9.8 Fourier transformation

The implementation of the *Fourier transformation* for the *concentric ring trajectory* is inserted in Listing 2.20.

```
1  for (int k_t = 0; k_t < Nt; k_t++) {
2    srcAs.modify (COL, k_t, 1, 1);
3    outAs.modify (COL, k_t, 1, 1);
4    for (int r_x = 0; r_x < Nx; r_x++) {
5      outAs.modify (LIN, r_x, 1, 1);
6      for (int r_y = 0; r_y < Ny; r_y++) {
7        outAs.modify (SEG, r_y, 1, 1);
8        for (int k_r = 0; k_r < Nr; k_r++) {
9          srcAs.modify (LIN, k_r, 1, 1);
10         for (int k_φ = 0; k_φ < Nφ; k_φ++) {
11           srcAs.modify (SEG, k_φ, 1, 1);
12           srcPixel=(std::complex<float>*) srcAs.calcSplObjStartAddr();
13           dummy_xy = dummy_xy + *srcPixel · discret_xy[r_x][r_y][k_r][k_φ];
14         }
15       }
16       outputPixel=(std::complex<float>*) outAs.calcSplObjStartAddr();
17       *outputPixel = conj(dummy_xy);
18       dummy_xy = 0.0;
19     }
20   }
21 }
```

Listing 2.20: Implementation of the *spatial discrete Fourier transformation* for concentric rings.

As the implementation for the *concentric ring trajectory* was very similar to solution 2 of the *Cartesian DFT*, only the differences will be discussed in this Subsection.

The only difference between Listing 2.3 and 2.20 was that, instead of the $k_x$ and $k_y$ for-loops, in Listing 2.2 (i.e., *Cartesian* trajectory), the ring-specific loops went through indices $k_r$ and $k_\varphi$. Also, indexing of the *discretization* relied on $k_r$ and $k_\varphi$, instead of $k_x$ and $k_y$. In all other aspects the two codes were identical.

## 2.10 Evaluation tools

The evaluation tools include spectroscopic images from the first FID points of real and imaginary parts of the complex-valued signal, ICE-MATLAB-quotient images, FIDs, and spectra.

All the above mentioned quality-assessment tools were applied on data from phantom measurements, on a 16×16 matrix, which was measured with eight *k-space* rings. In addition, the results of the *complete reconstruction*, on the 16×16 matrix, were compared to the same object, but on a 64×64 matrix (i.e., higher spatial resolution). FIDs and spectra from the larger matrix (i.e., 64×64 voxels) were evaluated using an additional statistical measure, *cross-correlation* maps.

For plotting the outcomes in the *Results* Chapter, the *k-space* trajectory in the *offline reconstruction* was calculated the same way as in the *online pipeline* and replaced the trajectory calculated via discrete integration along the whole gradient functions. By doing so, there were no differences anymore in the data processing between the two pipelines.

### 2.10.1 Tools for image evaluation

For evaluation purposes, a circular mask was placed over the images. The main reason for this was that the image results outside the central circular area were not valid, as there were *aliasing artefacts*. By laying a circular mask over the image, those artefacts were removed. The diameter of the mask corresponded with the matrix size. The values of the invalid pixels were set to NaN (i.e., not a number), in MATLAB, thus, they did not distort the value scales displayed next to the images.

The quotient images, used to compare ICE to MATLAB images, were calculated via a voxel-by-voxel division of ICE by MATLAB image data (i.e., ICE ./ MATLAB). Outliers in the quotient images seemed to derive mostly from the fact that the MATLAB image had, at that location, values very close to zero, which, when divided by, resulted in very large quotient values. Those outliers were irrelevant for the quality of the outcome. Instead, the focus was on variations between ICE and MATLAB. To quantify the deviations in the quotient images, *normalized image contrast*, $c$, was used.

$$c = (1 - \frac{Q_{min}}{Q_{max}}) \cdot 100\% \tag{2.26}$$

In Eq. 2.26, $Q_{min}$ stands for the lowest quotient value and $Q_{max}$ for the highest. The values $Q_{min}$ and $Q_{max}$ were determined only from the relevant image area (i.e., inside the phantom, where good quality data were expected). The outcome was multiplied by 100 to arrive at the *normalized image contrast* in percent.

Low $c$ values were desirable because they proved a high degree of conformity between results of the *online* and *offline reconstruction pipeline*, with $c = 1\%$ being a very high conformity and $c = 10\%$ and above requiring further improvements.

### 2.10.2 Tools for evaluation of FIDs and spectra

The instruments used to compare the FIDs and spectra were same: visual appearance; mean; standard deviation; and *cross-correlation*.

The plotted FIDs and spectra from the 16×16 matrix, from a representative voxel position in the middle of the image, $r_x = 8$ and $r_y = 8$, were depicted in the *Results* Chapter. FIDs and spectra from the 64×64 matrix were also shown from the central voxel (i.e., $r_x = 32$ and $r_y = 32$). This location was chosen for all FID and spectrum plots. All FIDs and spectra show the absolute values of the complex-valued signals. Also, means and standard deviations were inserted into each FID and spectrum diagram, in the upper right corner. *Cross-correlation* coefficients between the two resulting signals were written into figure captions. On the horizontal axis of the FID plots were the FID points, in units of *spectral dwell time*, $t_{spectral}$, and on the vertical axis the signal amplitude.

The spectra were created in MATLAB via a *spectral Fourier transformation* and an *fft shift* of the FID signals.

The tool to compare FIDs and spectra, resulting from the two different reconstruction pipelines, was the *cross-correlation*. *Cross-correlation* is a standard instrument of stochastic analysis to compare two time series. Its calculation results in a single value, which represents similarity of the two signals. The *cross-correlation* coefficient values range from -1, for no similarity at all, to +1, when the two signals are identical. This analysis instrument made it possible to assess the quality of the obtained results from the *online* pipeline, compared to the reference. The *cross-correlation* coefficient, $\kappa$, between two signals, $S_i$ and $S_j$, was calculated according to:

$$\kappa_{ij} = \frac{\gamma_{ij}}{\sqrt{\sigma_i^2 \cdot \sigma_j^2}} \tag{2.27}$$

In Eq. 2.27, $\gamma_{ij}$ is the covariance of the two signals and $\sigma^2$ are the individual signal variances, further specified with the signal indices [44].

# 3 Results

In the first Subsection, comparative data representations between the ICE (i.e., the *online reconstruction pipeline*) and MATLAB (i.e., the offline, established standard pipeline) results, are depicted. The individual data-processing steps are displayed separately to assess the quality of the data after each of the steps. Also, all the steps combined (i.e., *complete reconstruction*) are shown in a separate Subsection to show the cumulative effects of all the data operations. To see, how each specific data operation influenced the quality of the outcome (i.e., image, FID, and spectrum), results from a reconstruction only after performing the *spatial Fourier transformation* are inserted as well. In the second Subsection, runtimes of all *functors* are briefly described. The *functor* with a limiting runtime, i.e., the *spatial DFT*, is presented in more detail.

## 3.1 Comparison of online and offline reconstruction

The results, after applying only the *spatial DFT*, are shown as first. After that, resulting data representations from the individual data operations are depicted and described, in separate Subsections. The results in Subsections *Fourier transformation*, *Frequency-offset correction*, *Density compensation*, and *Complete reconstruction* are plotted from an acquisition, where there was no *FOV shift* applied (i.e., the phantom was placed directly in the isocenter). To demonstrate the *FOV shift* effects and the quality of its implementation in the *online reconstruction*, the results in the *FOV shift* Subsection are shown from a different acquisition than the rest. In this separate acquisition, the phantom was laid in the isocenter, but then, a given *FOV shift* was applied. Thereby, the FOV was shifted and the object (i.e., phantom) visually moved away from the center of the image. This procedure is, of course, not standard, as the *FOV shift* serves the exact opposite purpose, but, this was done solely for verification purposes, as it is not easy, in the given experimental setting, to position the phantom off-center.

In the first four Subsections, the results are depicted on a 16×16 matrix. In the fifth Subsection, *Complete reconstruction*, they are be shown on the 16×16, as well as on a 64×64 matrix, to see the differences in the reconstructed data of the same object, imaged with a higher, and for practical applications more relevant, spatial resolution. The scaling in images, FIDs, and spectra was often different between ICE and MATLAB, but, this was of less importance for the quality because the scaling does not influence the contrast, which is the most important image feature.

### 3.1.1 Fourier transformation

The resulting images, FIDs, and spectra from both pipelines, after performing only the *DFT*, are depicted in Figs. 3.1 - 3.4. In Fig. 3.1, a real-part image from the *online* (left), *offline reconstruction* (middle), and their quotient are depicted.

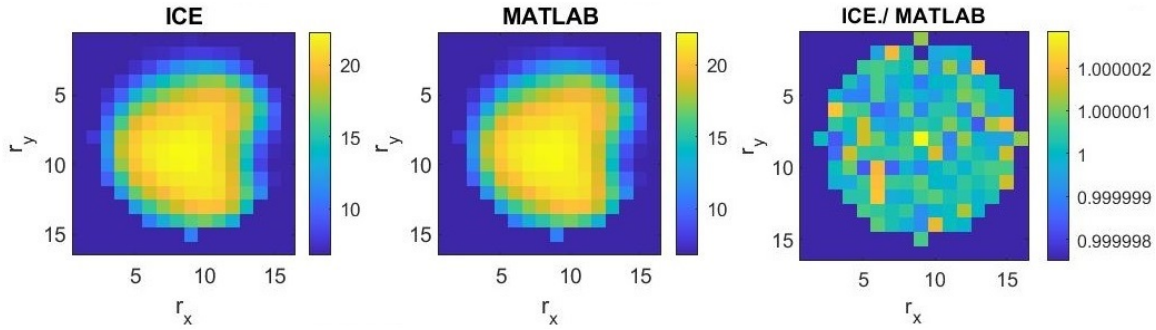


Figure 3.1: Real-part images from both CRT reconstruction pipelines, only after performing the *spatial DFT* and the quotient image (right) with a very low *normalized contrast*, $c = 0.0004\%$.

In Fig. 3.1, the ICE and MATLAB images appear identical. In the quotient image (i.e., ICE ./ MATLAB), it can be seen on the scale, next to the image, that the range of the quotient values is very closely distributed around one. Furthermore, the quotient image shows merely random variations and no patterns, which indicates a correct implementation of the *spatial DFT*, also including the scaling, in this case. The almost negligible random variations were caused only by different machine precision used by the two pipelines. Next, in Fig. 3.2, the same set of plots is shown, but for the imaginary part of the first FID points.

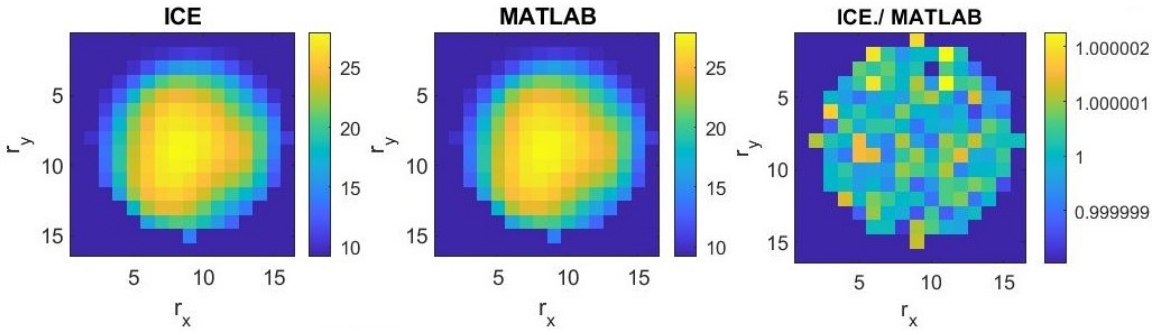


Figure 3.2: Identical to images in Fig. 3.1, only showing the imaginary part, instead of the real part with a *normalized contrast* in the quotient image of $c = 0.0003\%$, which proves a very high conformity between the results of the two reconstructions.

In the lower part of Fig. 3.2, also, very narrow and merely random variations between the results of the two reconstructions are shown.

As all of the acquisitions in this thesis were MRSI, using the *OldADC* CRT sequence, including the spectral dimension, the FIDs are plotted in Fig. 3.3.

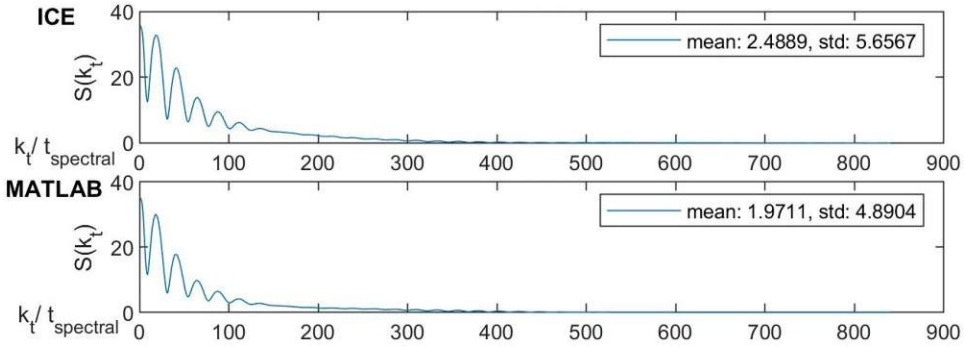


Figure 3.3: The time signal (i.e., FID) from *online* (above) as well as *offline reconstruction pipeline* (bellow) is shown, only after the *spatial DFT* has been applied with a very high *cross-correlation* of the FIDs between ICE and MATLAB, $\kappa = 0.9996$.

The FIDs from ICE and MATLAB appear very similar. The statistical measure for similarity, in this case, was the *cross-correlation*. In Fig. 3.3, the *cross-correlation* coefficient $\kappa = 0.9996$ is very close to one, which implies a great degree of similarity with the standard.

In Fig. 3.4, the spectra are depicted, and again, compared via *cross-correlation*. On the horizontal axis is shown the frequency in units of ppm and on the vertical axis, the amplitude of the absolute signal.

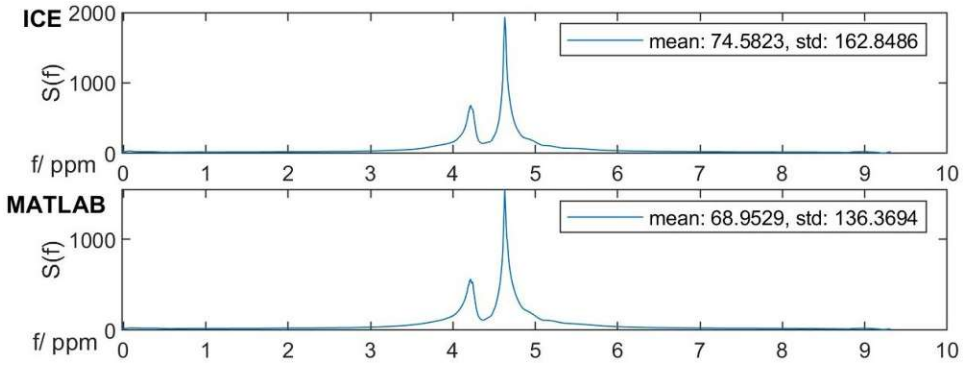


Figure 3.4: Frequency domain results from *online* (above) and *offline reconstruction pipeline* (bellow), only after performing the *spatial DFT* with a very high *cross-correlation* between the ICE and MATLAB spectra $\kappa = 0.9997$.

The two plots in Fig. 3.4 look identical, with a very high *cross-correlation* of 0.9997. The two resonance peaks are clearly visible, as was expected for the phantom used, which was described in the *Materials and Methods* Chapter.

### 3.1.2 Field-of-view shift

The results of the online, as well as the offline, *field-of-view shift* are presented in this Subsection. The data was acquired with the phantom in the isocenter. The *FOV* was then *shifted* in both in-plane directions, in this case right and anterior (down). This *FOV shift* is depicted in Figs. 3.5 and 3.6. As before, the ICE, MATLAB, images are displayed in one figure together with the corresponding (i.e., real- or imaginary-part) image only after performing the *spatial DFT* to demonstrate the effects of the current data operation. The real- and imaginary-part quotient images are shown together, in a separate figure.

This way of displaying the outcomes was also used for the *frequency-offset correction*, *density compensation*, and *complete reconstruction* (on the 16×16 matrix).

Real-part images are depicted in Fig. 3.5.

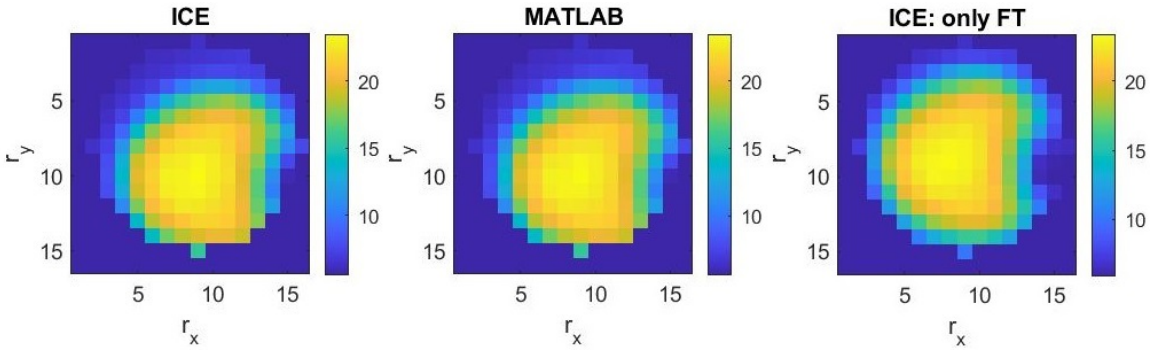


Figure 3.5: Effects of the *field-of-view shift* on the real-part image.

The ICE and MATLAB images look very alike. The effect of the *FOV shift* can be seen, when the ICE (left) and MATLAB (middle) images, in Fig. 3.5, are compared to the non-shifted image (right).

Imaginary-part images are shown in Fig. 3.6.

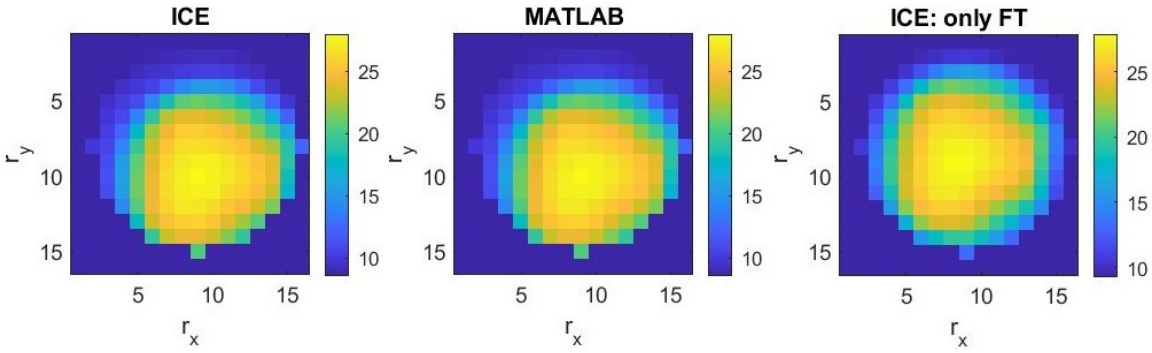


Figure 3.6: Effects of the *field-of-view shift* on the imaginary-part images.

As can be seen in Fig. 3.6, the effect of the *field-of-view shift* was the same on the imaginary-part, as was on the real-part image.

Both inter-pipeline quotient images, after the *field-of-view shift*, are shown in Fig. 3.7.

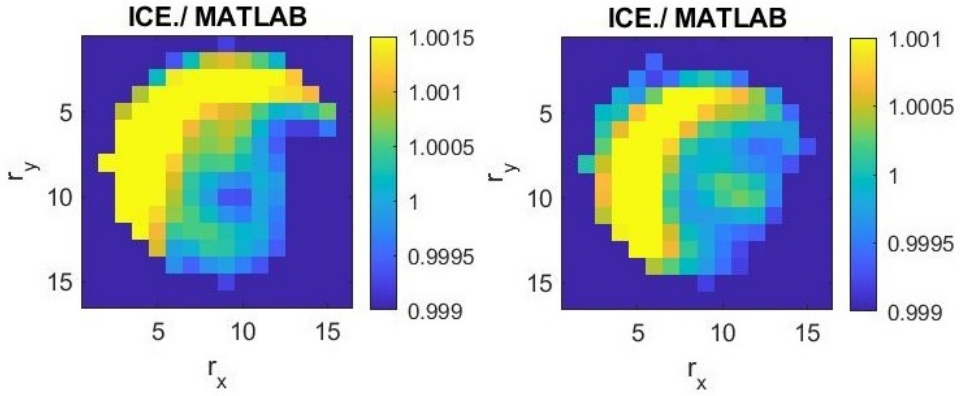


Figure 3.7: Real- (left) and imaginary-part (right) quotient images between the ICE and MATLAB results of the *field-of-view shift* with a very low *normalized contrast*, identical to both quotient images, of $c = 0.2\%$, proving a great degree of inter-pipeline conformity. The yellow outliers in the quotient images are found outside the phantom, and therefore, have no influence on the image quality.

The similarity of the ICE and MATLAB images was confirmed by the resulting quotient image, in Fig. 3.7 with a very low *normalized contrast* of $c = 0.2\%$, signaling a high quality of the online implementation. In the quotient image, the yellow voxels in the upper left part of the circular mask are irrelevant for the image quality, since, after the *FOV shift*, the object was visually shifted to the lower right. Thereby, the higher quotient values do not lie inside the phantom anymore.

FIDs from the both pipelines, which performed the *field-of-view shift*, as well as an FID only after the *spatial DFT*, are depicted in Fig. 3.8.

As the amplitude of the signal was identical in the images from both pipelines, it was also identical in the FIDs. Thereby, the means and standard deviations were comparable and they display very similar values in ICE and MATLAB, as can be seen in the upper right part of each FID plot, in Fig. 3.8. A very high inter-pipeline conformity was achieved with a *cross-correlation* of $\kappa = 0.99997$.

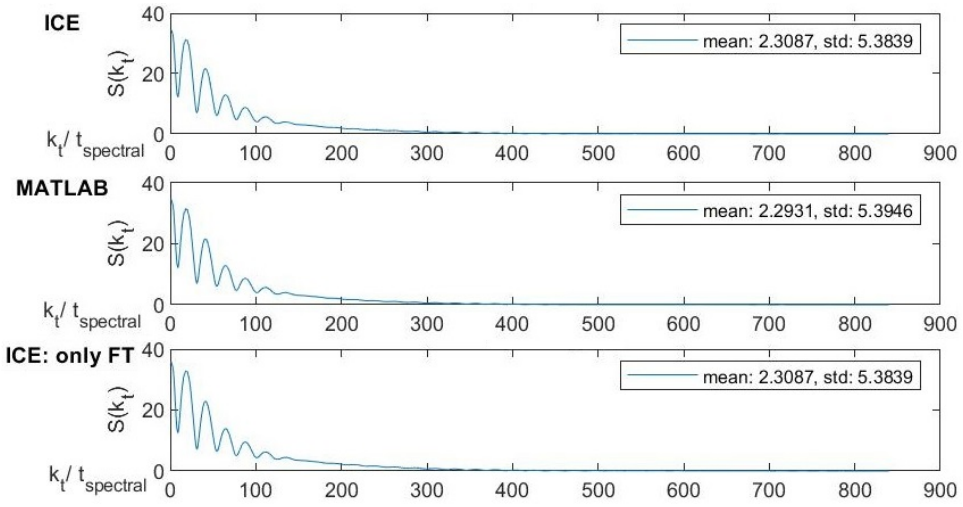


Figure 3.8: Effects of *field-of-view shift* on the time domain of the *online* and *offline reconstruction* with a very high inter-pipeline *cross-correlation* $\kappa = 0.99997$.

The *field-of-view-shift* did not effect the form of the FID in any way, which was the expected correct outcome.

Corresponding spectra, to the FIDs in Fig. 3.8, are presented in Fig. 3.9.

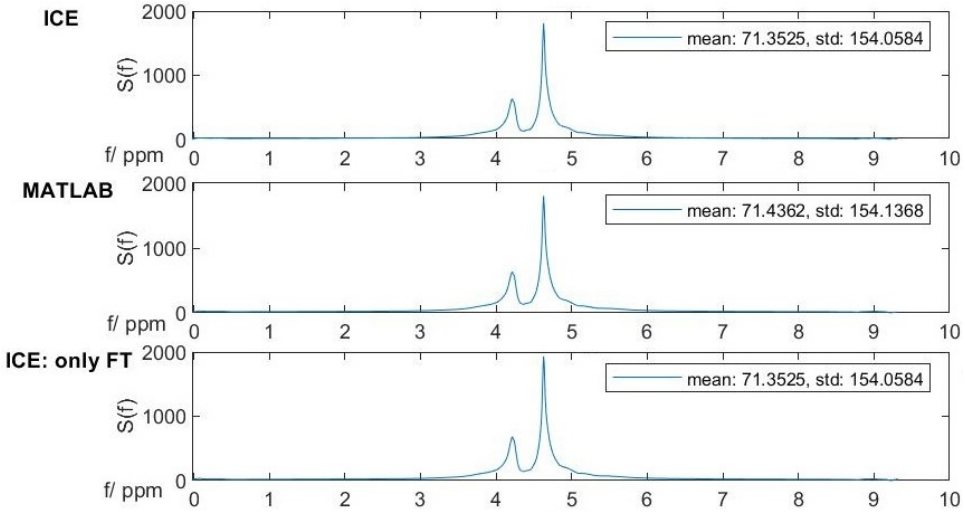


Figure 3.9: Effects of *field-of-view shift* on the frequency domain from *online* and *offline reconstruction pipeline* with a very high *cross-correlation* $\kappa = 0.99995$.

Spectra from ICE and MATLAB, after the *field-of-view shift* have the same form, as well as mean and standard deviation values. The *cross-correlation* coefficient was therefore very close to one, 0.99995.

Just as it was expected and confirmed for the FIDs, the *field-of-view shift* only shifted the voxel signals in respect to the *FOV*, but, the spectra were not affected. With this, it was verified that this data operation is working properly.

### 3.1.3 Frequency-offset correction

The graphical representations of the *frequency-offset correction* results are displayed in this Subsection, and are the same as in the Subsection on *Field-of-view shift*.

First, the real-part images are shown in Fig. 3.10.

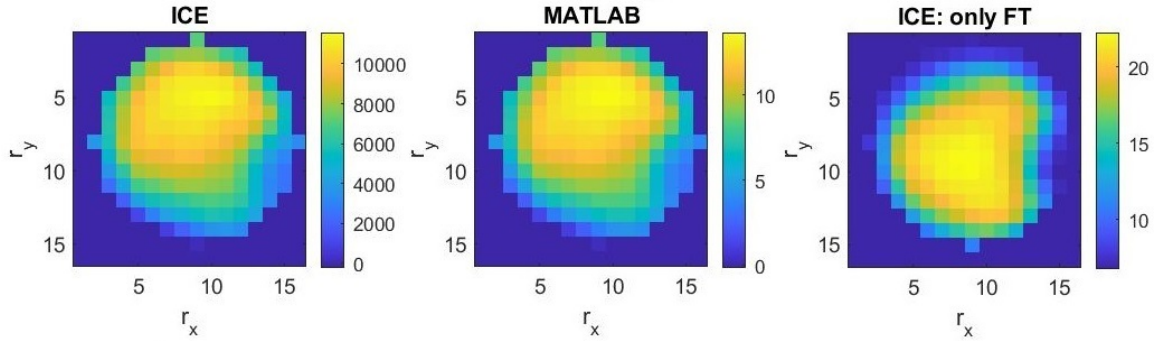


Figure 3.10: Effects of the *frequency-offset correction* on the real-part image from the *online* and *offline reconstruction*.

When comparing the ICE (left) and MATLAB (middle) images in Fig. 3.10, they appear identical, in terms of image contrast. Yet, the scaling of the two images is different. The highest signal in the ICE image is three orders of magnitude higher than in the MATLAB version. This is not problematic because mainly the contrast matters, as was mentioned earlier. Furhtermore, it can be seen in Fig. 3.10 that, when the *frequency-offset correction* was applied, the image was less blurred, as the high signal was located more in the upper part of the image and not everywhere around the central area of the image, as was the case when only the *spatial DFT* was applied (right).

In Fig. 3.11, the imaginary part of the *frequency-offset-corrected* images are depicted.

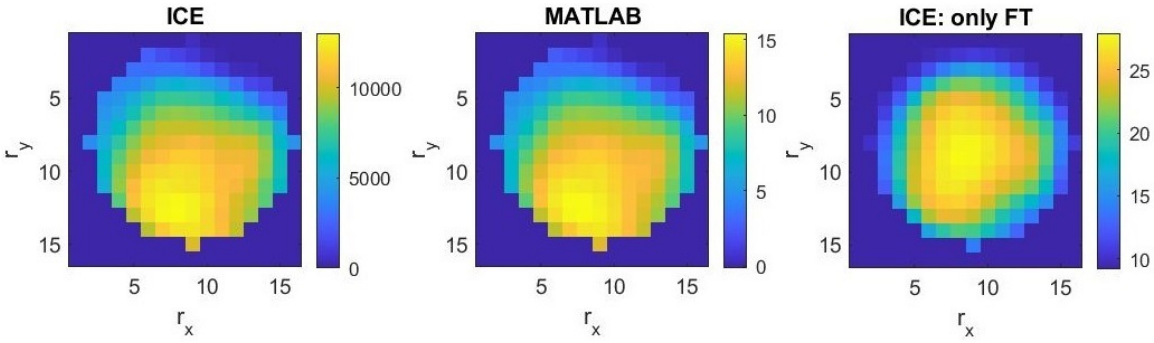


Figure 3.11: Effects of the *frequency-offset correction* on the imaginary-part images.

The images from the two pipelines seem again identical. The reduction of the signal blurring upon the *frequency-offset correction* can be seen, when comparing the *corrected* ICE image (left), in Fig. 3.11, to the image only after the *spatial DFT* (right).

In Fig. 3.12, quotient images, comparing the two (i.e., ICE and MATLAB) *frequency-offset correction* pipelines, are depicted.

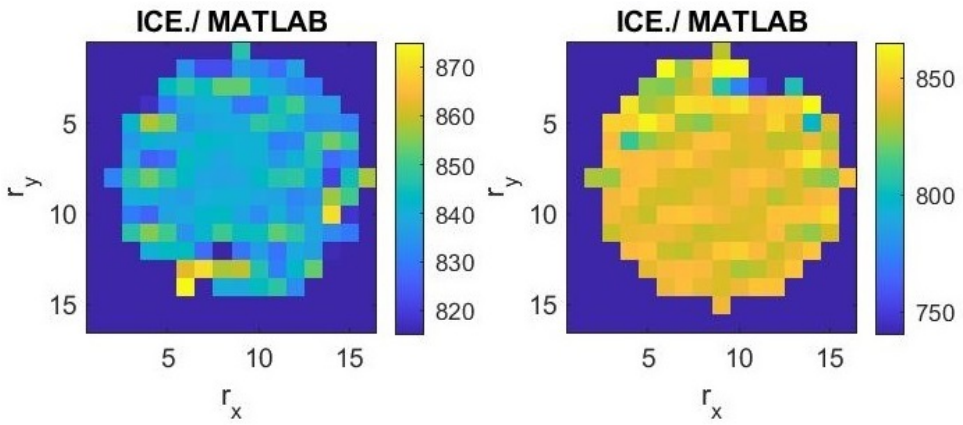


Figure 3.12: Real- (left) and imaginary-part (right) quotient images between the ICE and MATLAB results of the *frequency-offset correction* displaying deviations between the results from *online* and *offline reconstruction* with a *normalized contrast* in the real-part quotient image of $c = 4.7\%$, and in the imaginary-part quotient image of $c = 4.8\%$, which shows conformity between the two pipelines within acceptable limits.

As depicted in Fig. 3.12, there were some deviations in the quotient images. The *normalized contrast*, $c$, in the real-part quotient image reached a value of 4.7%, signaling a need for further improvements in the precision of the results from the *online reconstruction*. The same holds true for the imaginary-part quotient image. The outliers in the quotient images are irrelevant for the image quality because, first, mainly the image contrast matters, and second, they lie outside the phantom, where significant deviations were expected.

In Fig. 3.13, there are FIDs after performing the *frequency-offset correction* from the ICE and MATLAB pipelines, as well as the FID coming from the ICE pipeline only after the *DFT*.

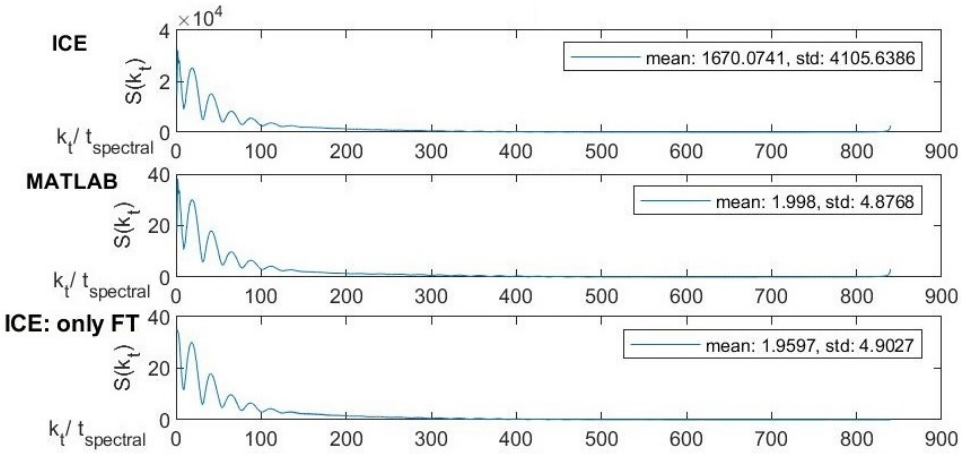


Figure 3.13: Effects of *frequency-offset correction* on the time domain of the *online* and *offline reconstruction* with a very high inter-pipeline *cross-correlation* $\kappa = 0.9998$.

In Fig. 3.13, the FID from ICE copies the form of the MATLAB standard and differs only in the amplitude of the signal. When is the ICE image compared to the non-corrected FID, there is no difference, except for different scaling. The *cross-correlation* coefficient was calculated for the *corrected* FIDs from the two different pipelines, i.e., ICE and MATLAB, having a value very close to one, proving a strong correlation, and therefore good quality of the *online reconstruction*.

From the three FIDs in Fig. 3.13, spectra were created and are depicted in Fig. 3.14.

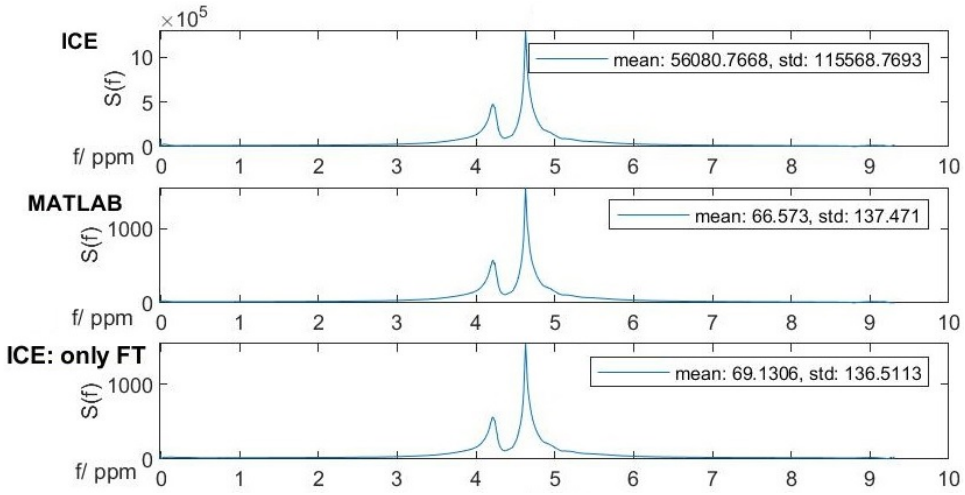


Figure 3.14: Effects of *frequency-offset correction* on the frequency domain from *online* and *offline reconstruction pipeline* with a very high *cross-correlation* $\kappa = 0.9997$.

After this data-processing step, the spectra from both pipelines successfully maintained the two resonances. The ICE spectrum shows no deviation from the MATLAB version. The change made through the *frequency-offset correction* is, also here, nothing else than a increase in the signal amplitude, in the *online result*.

### 3.1.4 Density compensation

The ICE and MATLAB real-part images after the *density compensation* are displayed in Fig. 3.15. Also the ICE image with no additional data processing, i.e., only after performing the *spatial DFT*, is shown to illustrate the effects of the *density compensation*.

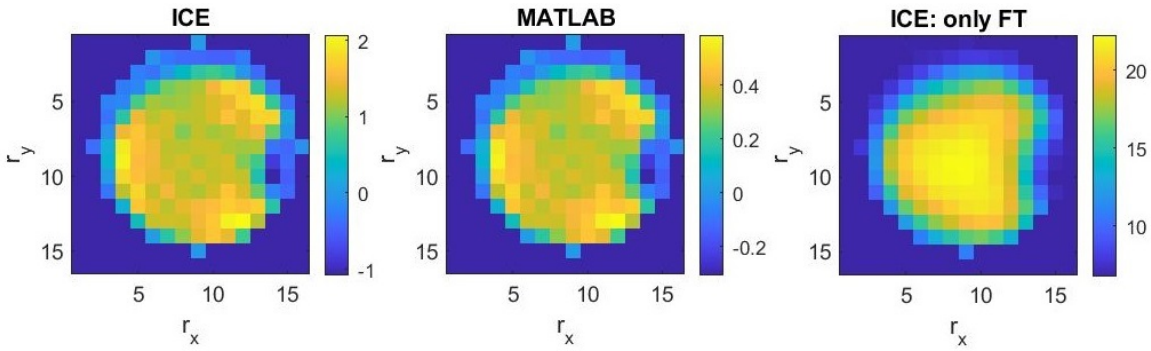


Figure 3.15: Effects of the *density compensation* on real-part images.

In Fig. 3.15, the ICE and MATLAB seems visually identical. They are more closely via the quotient image in Fig. 3.17.

What the *density compensation* essentially does, is an optimization of the spatial response function. In case of the *OldADC* CRT sequence, this resulted into stronger weighting of the high spatial frequencies, which removed signal blurring and brought up more details in the images. This is apparent from the comparison of the *density compensated* ICE image (left), in Fig. 3.15, and ICE image only after the *spatial DFT* (right).

In Fig. 3.16, the imaginary-part images are depicted.

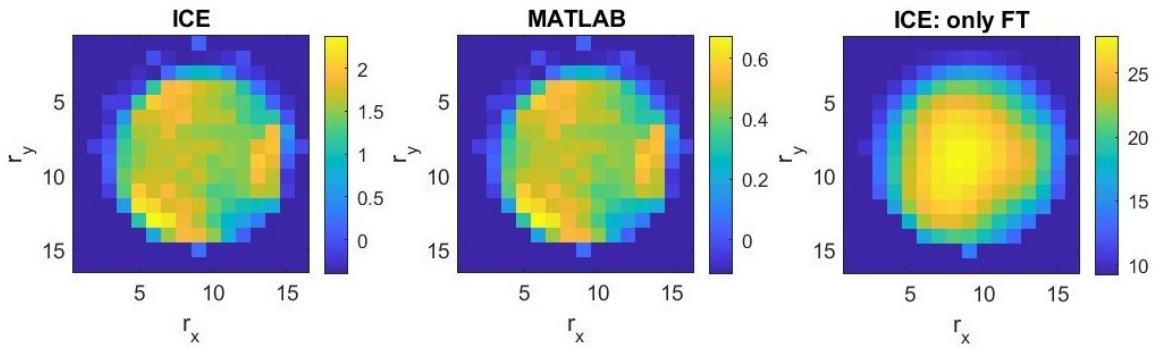


Figure 3.16: Identical to Fig. 3.15, only showing the imaginary, instead of the real, part of the complex-valued signal.

As for the real part, the effect of the *density compensation* is a clear highlighting of image details, as there is not just a bulk of signal over the whole center of the image, as can be seen in the rightmost subplot, in Fig. 3.16, but a distinction between regions with lower and higher signal intensities is evident.

In Fig. 3.17, the real- (left) and imaginary-part (right) quotient images were inserted.

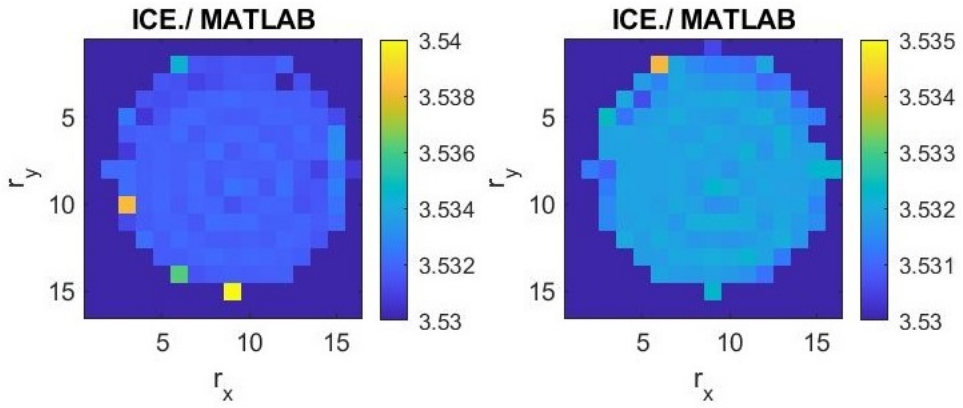


Figure 3.17: Inter-pipeline comparison via real- (left) and imaginary-part (right) quotient images between the ICE and MATLAB results of the *density compensation* with a very low *normalized contrast* in the real-part plot of $c = 0.2\%$ and an even lower $c$ of 0.1% in the imaginary-part plot. These *contrast* values show an overall high conformity between the results of the two pipelines.

As can be seen in Fig. 3.17, in the real-part quotient image, the range of values on its scale is very narrow (i.e., from 3.53 to 3.54) signaling a high quality of the online implementation. The comparison between the ICE and MATLAB image in Fig. 3.16 (i.e., the quotient image) delivers the same outcome as for the real part, with an even narrower range of the quotient values, from 3.53 to 3.535, underlined by a very low *normalized contrast* of $c = 0.1\%$.

Fig. 3.18 shows the absolute value of the FIDs from a voxel in the center of the image. The ICE FID, i.e., from the *online reconstruction*, is identical to the *offline* version, except for the high frequency superimposing the FID in the red-highlighted areas. Nevertheless, the *cross-correlation* coefficient between the *compensated* ICE and MATLAB time signal remained high with the exact value of $\kappa = 0.9801$.

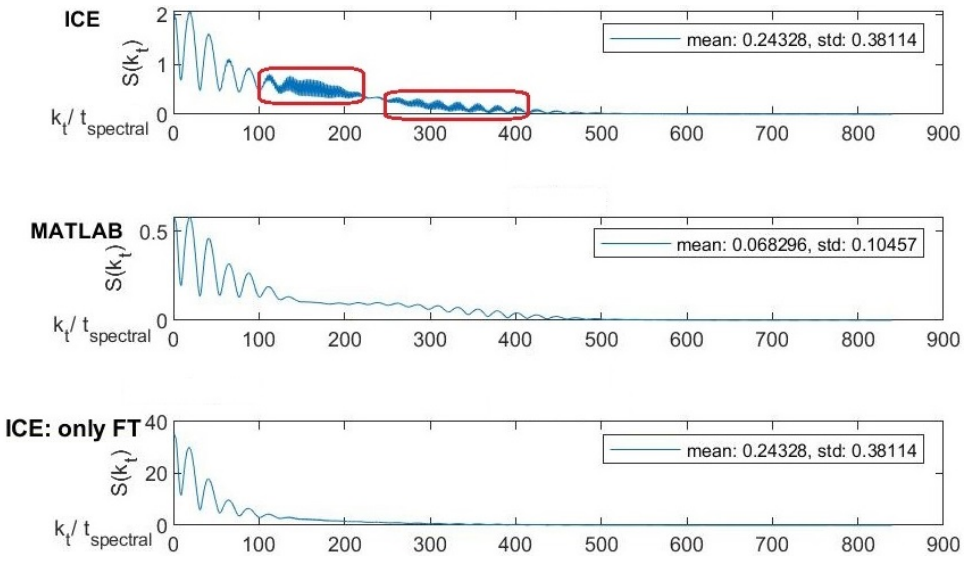


Figure 3.18: Effects of *density compensation* on the time domain showing a slower signal decay when compared to the FID only after the *spatial DFT*. In the FID from ICE there is a high-frequency artefacts on top of the FID. The affected areas are highlighted by red rectangles. *Cross-correlation* of the FIDs between ICE and MATLAB $\kappa = 0.9801$.

In Fig. 3.18, when comparing the MATLAB to the ICE plot only after the *spatial DFT*, in the region between 100 and 400 *spectral dwell times*, $t_{spectral}$, it can be clearly seen that the *density-compensated* FID decayed slower. The same is the case in the ICE plot, but harder to recognize because of the overlying high-frequency artefact.

The *density-compensated* spectra are shown in Fig. 3.19.

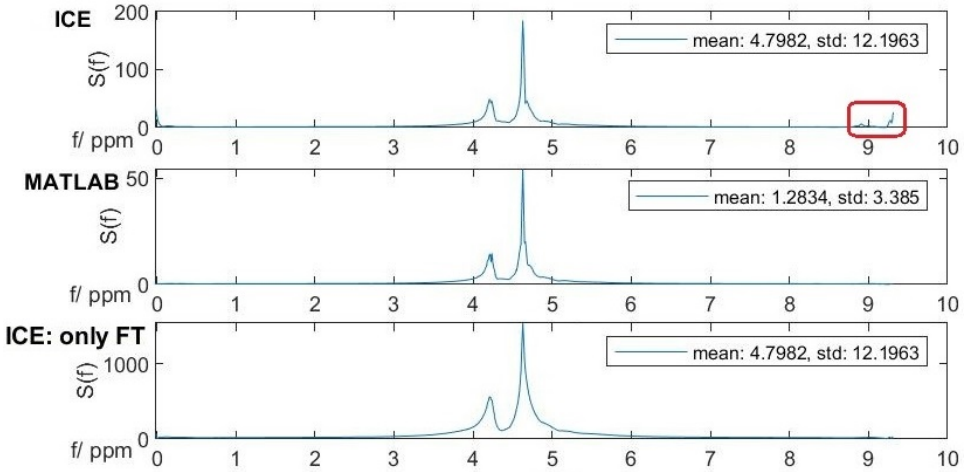


Figure 3.19: Effects of *density compensation* on the frequency domain, displayed for the *online* and *offline reconstruction* with a high *cross-correlation* $\kappa = 0.9737$. The high-frequency artefact in the ICE manifests at the left end of the spectrum.

The high-frequency artefact in the ICE plot in Fig. 3.18, is visible in the corresponding spectrum in Fig. 3.19, as a small peak at a high frequency, above 9 ppm. Otherwise, the two spectra do not show any pronounced differences, resulting into a high $\kappa$ of 0.9737.

### 3.1.5 Complete reconstruction

In this Subsection, first, the results of the *complete reconstruction* are shown in the same way as in the previous Subsections, on a 16×16 matrix.

Second, a similar set of results (i.e., real and imaginary part images, FIDs, and spectra) were plotted on a 64×64 matrix (i.e., higher spatial resolution). Again, *online* and *offline reconstruction* were compared. Instead of reviewing the effects of the data operations, the effects of increasing the spatial resolution were shown, i.e., a comparison of the results between the two different matrix sizes by displaying an image from ICE on the 16×16 matrix in the rightmost subplot in the figures. Also, an FID and spectrum at the same image location (i.e., on the 16×16 matrix the voxel at $r_x = 8$ and $r_y = 8$, equivalent to the $r_x = 32$ and $r_y = 32$ voxel on the 64×64 matrix) of the smaller matrix were inserted below the FIDs in Fig. 3.28, and spectra in Fig. 3.30.

Third, *cross-correlation* maps were depicted for FIDs and spectra on the 64×64 matrix.

Images on the 16×16 matrix, built up by the real part of the first FID points, are shown in Fig. 3.20.

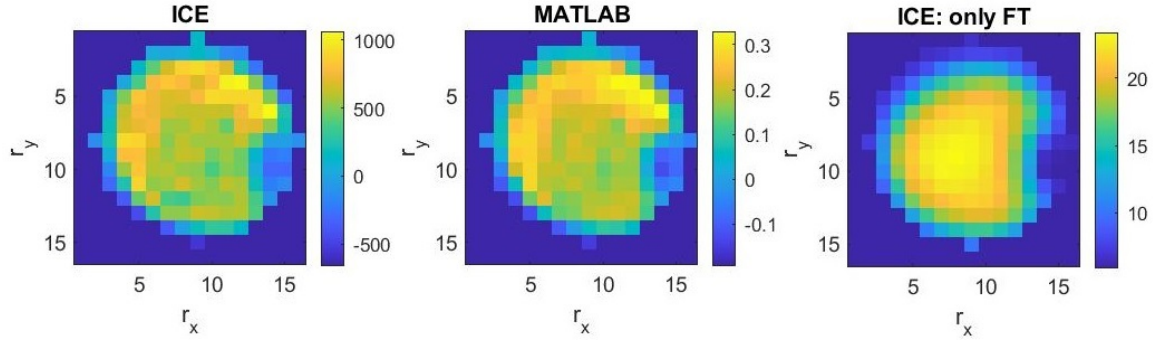


Figure 3.20: Combined effects of *frequency-offset correction* and *density compensation* on the real-part images on a 16×16 matrix.

In Fig. 3.20, there is no significant difference between the ICE (left) and MATLAB (middle) images. But, a closer comparison was performed via the quotient image below. When the ICE image from the *complete reconstruction* is compared to the ICE image only after applying the *spatial DFT* (right), it can be seen that the final result contains high signal regions, e.g., at the coordinates $r_x = 12$ and $r_y = 5$, as well as some image details in the center of the image.

The imaginary-part images are depicted in Fig. 3.21.

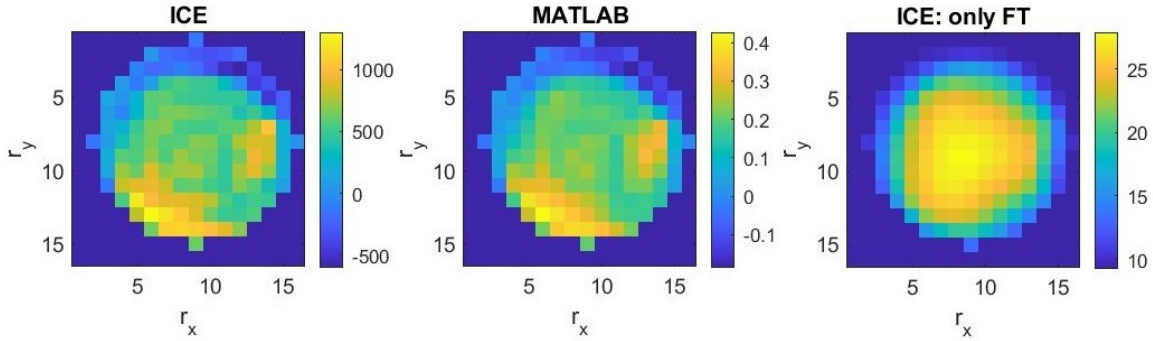


Figure 3.21: Identical to images in Fig. 3.20, only showing the imaginary, instead of the real, part of the signal.

In Fig. 3.21, when is the *frequency-offset-corrected* and *density-compensated* ICE image (left)

compared to the untreated ICE image (right), there can be seen a clear gain of details, just as was the case in Fig. 3.20.

The quotient images, resulting from Figs. 3.20 and 3.21, are depicted in Fig. 3.22.

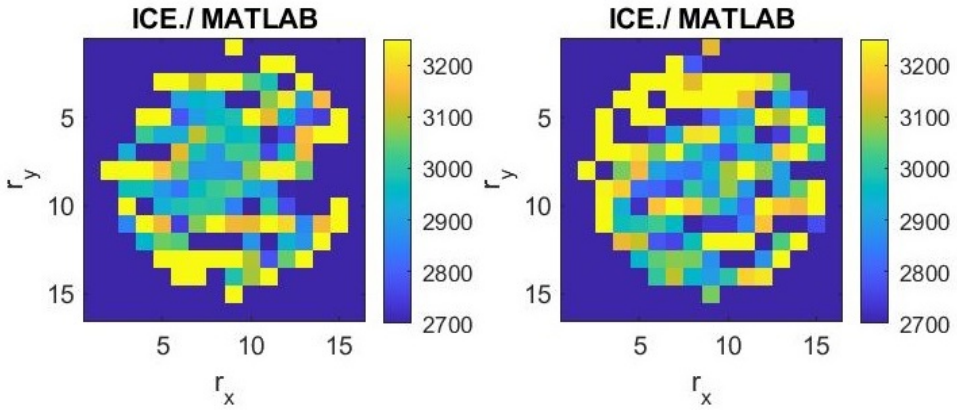


Figure 3.22: Comparison of the *online* and *offline pipeline* via the real- (left) and imaginary-part (right) quotient images with an identical *normalized contrast* in both of $c = 15.6\%$, pointing to a suboptimal inter-pipeline conformity, caused by significant artefacts in the *online results*, and the necessity for further improvement.

Based on results from previous subsections, the suboptimal *normalized contrast* in the real and imaginary-part quotient images between the two *complete reconstruction* pipelines, in Fig. 3.22, was caused mainly by the *frequency-offset correction*, and displaced to other image regions, probably by a cumulative effect of the data processing. Nevertheless, there seems to be no recognizable pattern on the 16×16 matrix, as only random variations are observed inside the phantoms, in Fig. 3.22, which is a good sing that there are no systematic errors in the online implementation.

FIDs, from a voxel location in the middle of the image, are plotted in Fig. 3.23.

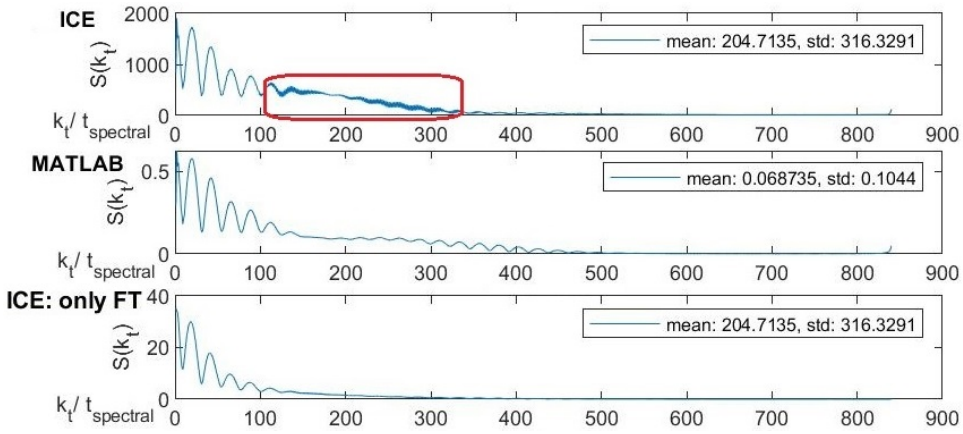


Figure 3.23: Effects of the *complete reconstruction* (i.e., after applying the *FOV shift*, *frequency-offset correction*, and *density compensation*) on the time domain, from the *online* as well as *offline reconstruction pipeline*. *Cross-correlation* between the ICE and MATLAB FIDs, $\kappa$, amounts to 0.9808. A high-frequency artefact is highlighted in the ICE time signal.

As was shown in Fig. 3.18, the high-frequency artefacts in FIDs from the *online reconstruction*, in Fig. 3.23 and 3.28, were introduced through the *density compensation*. Expect for the high-

frequency artefact, the ICE and MATLAB plots appear identical with a high *cross-correlation* of 0.9808.

When comparing ICE from the *complete reconstruction* to the ICE plot only after the *spatial DFT* (i.e., untreated), it shows the same feature as for the *density compensation*, in Fig. 3.18. The FID retains an above-zero signal for a longer period of time. This can be especially seen, when comparing the MATLAB, as it is not affected by the high frequency laid over it. The untreated ICE signal falls to zero at approximately $k_t = 300$, but the FID signal in the MATLAB and ICE from the *complete reconstruction* goes to zero approximately 200 $t_{spectral}$ later, i.e., at $k_t = 500$.

The corresponding spectra to FIDs in Fig. 3.23, are plotted in Fig. 3.24.

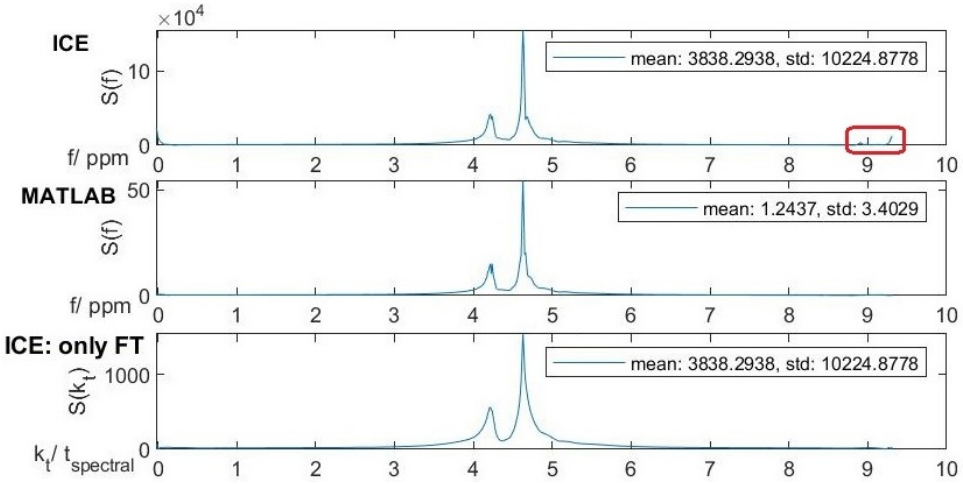


Figure 3.24: Effects of the *complete reconstruction* on the frequency domain for the both pipelines with a high *cross-correlation* between the ICE and MATLAB spectra $\kappa = 0.9822$.

The high frequency in the FID from ICE, in Fig. 3.23, shows, again, at the right end of the ICE spectrum, in Fig. 3.24, as a rather insignificant peak. This is not the case for the MATLAB plot. This artefact can, also, not be observed in the ICE spectrum only after the *spatial DFT*, as it was not present in the corresponding FID.

The effect of the *complete reconstruction* can be observed on the form of the most prominent peak, at 4.7*ppm*. The peak in the treated spectra, ICE as well as MATLAB, has a narrower base and is in general more narrow than just after performing the *spatial DFT*.

The same object was also measured with a higher spatial resolution of 64×64 voxels, and all of the data-processing steps (i.e., *complete reconstruction*) were performed on that data. The real-part ICE (left) and MATLAB (middle) images are depicted in Fig. 3.25, together with the real-part image from ICE, also from the *complete reconstruction*, on the smaller matrix, i.e., 16×16 voxels (right).

When the ICE (left) is compared to MATLAB (middle) image, in Fig. 3.25, they are visually not identical and artefacts in the ICE image can be seen. The structure of the artefacts is especially apparent in the quotient image, depicted below. When the ICE image on a 64×64 matrix (left) is compared to its 16×16 equivalent (right), it can be observed that the overall appearance of the image is the same, but of course, with a greater detail in the 64×64 image, as the reconstruction on the smaller matrix was subject to substantial *partial-volume* artefacts due to the overly large, in-plane voxel dimensions.

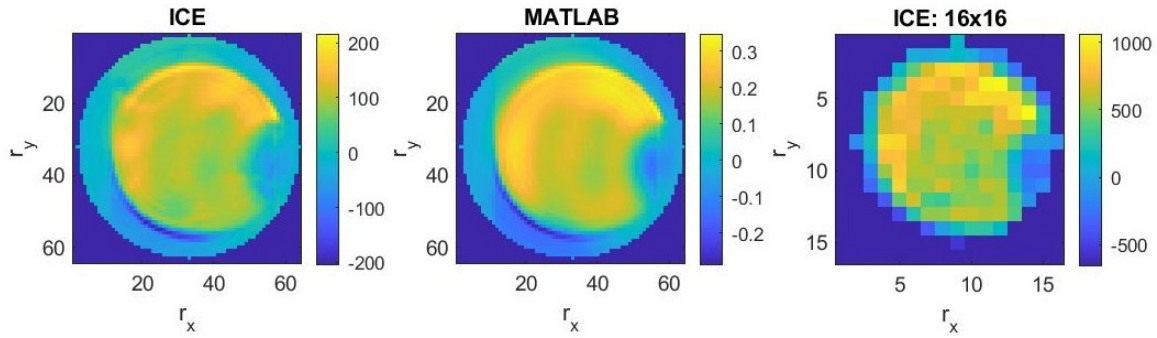


Figure 3.25: Effects of increasing the spatial resolution from 16×16 to 64×64 voxels. There are clearly visible artefacts in the ICE image (left), when visually compared to the MATLAB standard (middle).

The imaginary parts of the first FID points on the 64×64 matrix are depicted in Fig. 3.26.

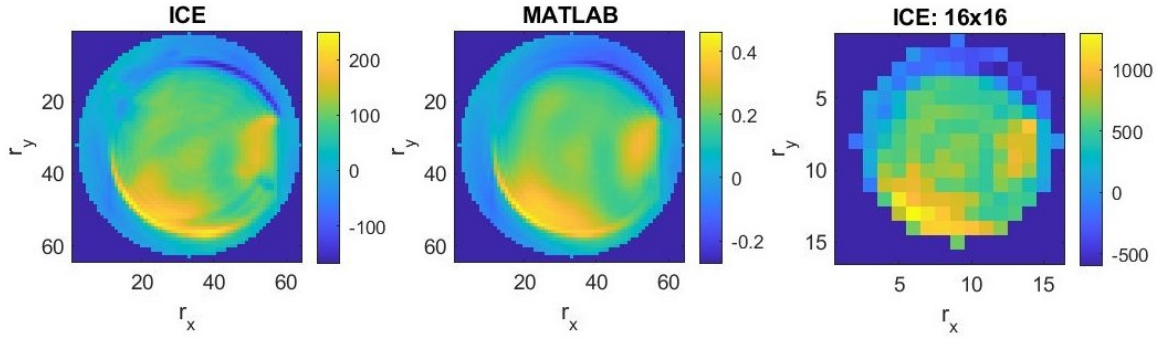


Figure 3.26: Effects of increasing the size of the matrix from 16×16 to 64×64 voxels, and thereby, increasing the spatial resolution. The ICE result is shown in the leftmost subplot, MATLAB in the middle, and the reference image with 16×16 voxels is depicted in the rightmost subplot.

The ICE image, in Fig. 3.26, is again similar to the MATLAB standard, but, also, contains visible artefacts. The effects of increasing the spatial resolution are identical to the real-part images, described above.

The real- (left) and imaginary-part (right) quotient images of the higher-resolution matrices are depicted in Fig. 3.27. The *normalized contrast*, in the real-part quotient image, amounts to 18.5%, which is beyond the acceptable threshold. Therefore, further improvements are necessary in the following work. The imaginary-part quotient image shows the same qualities as for the real part. One artefact is common to the real, as well as the imaginary-part image, which is a tilted line of low quotient values (blue) crossing the phantom from the lower left to it upper right, with areas of high quotient values (yellow) around it.

Based on the previous results, it is assumed that those artefacts arose from the *frequency-offset correction*, as there were the highest *normalized contrast* values in the quotient images, which were then further increased by the combination of all the data operations. But, to be certain, where those artefacts exactly come from, the reconstructions of all the individual data-processing steps should be performed with a higher spatial resolution, i.e., the 64×64 matrix. The following procedure is explained in the Chapter on *Discussion and Outlook*.

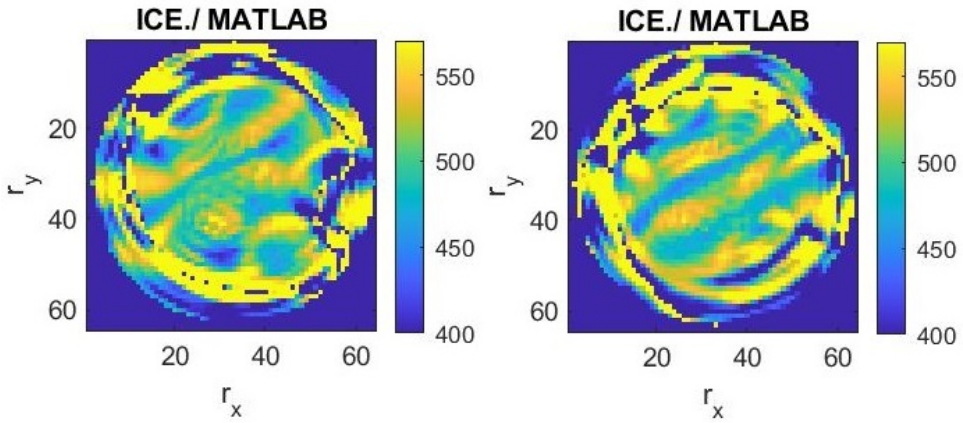


Figure 3.27: Inter-pipeline differences with a *normalized contrast* in the real-part quotient image (left) $c = 18.5\%$ and in the imaginary-part quotient image of $c = 23.6\%$.

The FIDs, from a voxel in the middle of the image (i.e., $r_x = 32$ and $r_y = 32$), are displayed in Fig. 3.28.

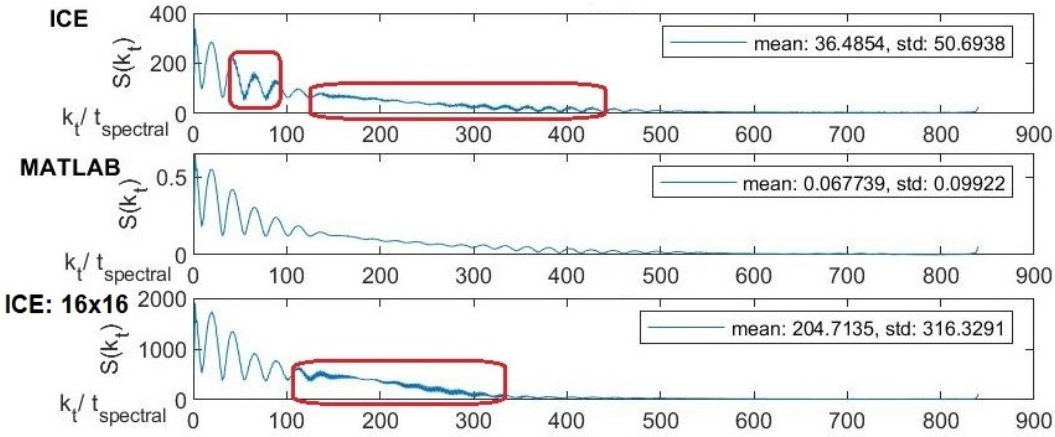


Figure 3.28: Effects of increasing the spatial resolution, from 16×16 to 64×64 voxels, on the time domain of the same voxel location (i.e., from the middle of the phantom) with a high *cross-correlation* between the FIDs from ICE and MATLAB $\kappa = 0.9948$. High-frequency artefacts are highlighted in red, having different forms in the two different spatial resolutions.

In contrast to the FID from the *offline reconstruction* (i.e., MATLAB), the ICE FID was affected by the high-frequency artefact. Nevertheless, the *cross-correlation* coefficient, $\kappa_{all}$, for those two signals was very high, with a value of 0.9948. This was also the case for the FID on the central position, on the 16×16 matrix, shown in the bottom line in Fig. 3.28.

The *cross-correlation* coefficient was then calculated for all of the image positions to create a so-called *cross-correlation* map. This was done with the FIDs, from the *offline* and *online reconstruction*, on the 64×64 matrix and the result is depicted in Fig. 3.29.

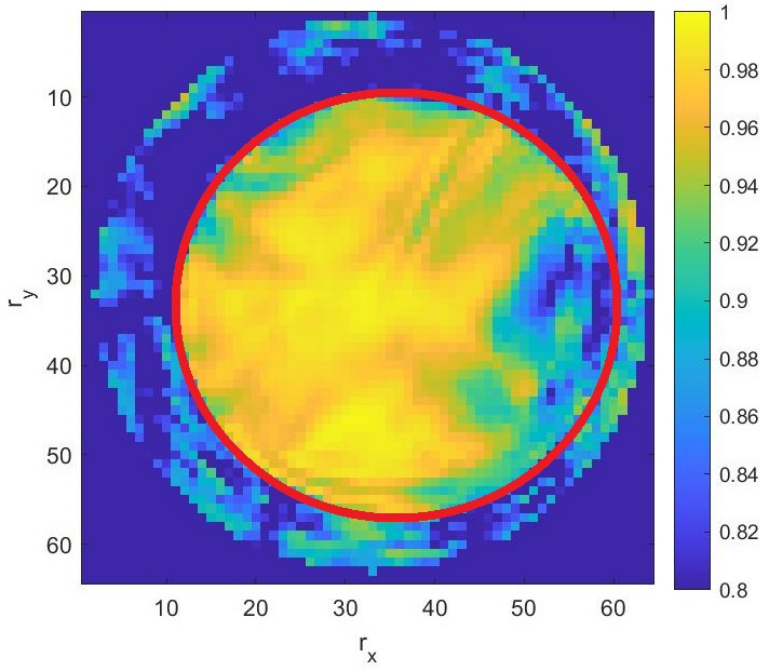


Figure 3.29: Map of *cross-correlation* coefficients between the ICE and MATLAB FIDs, after performing all of the data-processing steps, $\kappa$ at all image positions $r_x$ and $r_y$. The surface of the phantom is inside the red circle. On the right side of the phantom, there was an air bubble in the phantom, therefore no high *cross-correlation* was expected in that region.

It can be seen that the *cross-correlation* coefficients, in Fig. 3.29, are, inside the phantom, very close to one with the lowest value of 0.92 for the few greenish voxels, which still represents a very high correlation.

The corresponding spectra, to the FIDs in Fig. 3.28, are shown in Fig. 3.30.

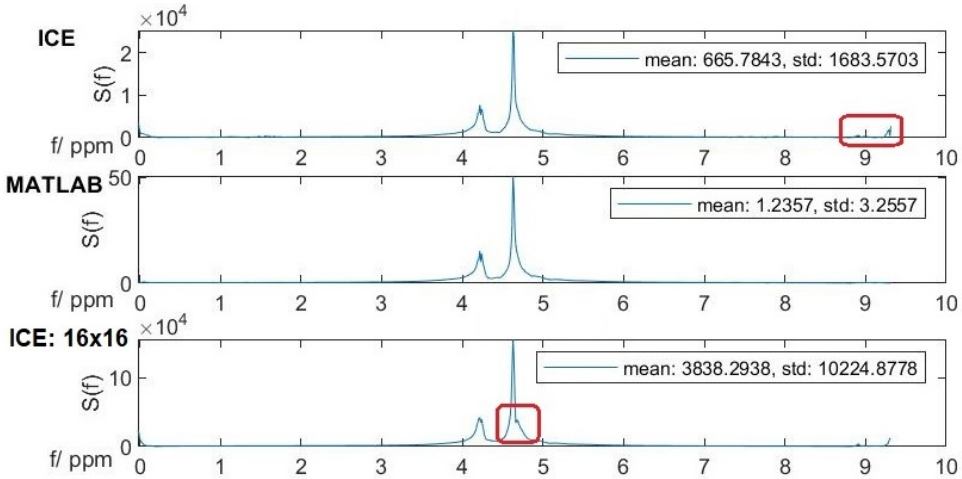


Figure 3.30: Effects of increasing the spatial resolution on the frequency domain of the two pipelines with a very high *cross-correlation* between the ICE and MATLAB spectra $\kappa = 0.9905$. The high-frequency artefact in the ICE, and a disruption in the ICE spectrum from the 16×16 matrix are highlighted in red.

When ICE is compared to MATLAB, in Fig. 3.30, the only significant difference is, again, the high-frequency artefact, otherwise there is a high *cross-correlation* of 0.9905 between the two spectra.

The ICE spectrum, from the middle of the 64×64 matrix, shows greater smoothness of the peak than the spectrum from the same location on the smaller matrix. The disruption, highlighted in the bottom spectrum, in Fig. 3.30, is not present in the corresponding spectrum on the 64×64 matrix (i.e., the top spectrum). Nevertheless, the disruption in spectrum from the smaller matrix size does not interfere with the overall appearance of the spectrum, especially not the main feature of the spectrum, i.e., the two resonances. When we reconsider Fig. 3.24, we see that the small disruption, on the right flank of the highest peak, is also present in the MATLAB spectrum (i.e., the established standard), and therefore, poses no threat on the spectral quality.

A *cross-correlation* map was, also, calculated for the spectra from ICE and MATLAB, on the 64×64 matrix. The result is depicted in Fig. 3.31.

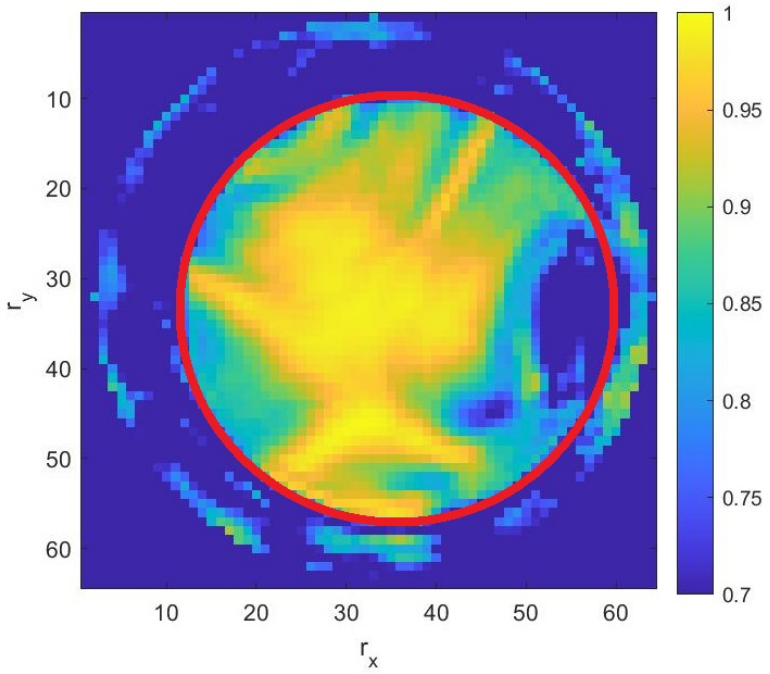


Figure 3.31: Map of *cross-correlation* coefficients between the ICE and MATLAB spectra, after performing all of the data-processing steps. The phantom surface is surrounded by the red circle. An air bubble was on the right side of the phantom and no high *cross-correlation* was expected in that region.

Unlike in the FID *cross-correlation* map (i.e., in Fig. 3.29), the spectrum cross-correlation values, in Fig. 3.31, attain lower values, down to 0.4109, which is no longer a high correlation. But, the lowest *correlation* coefficient was at the voxel location $r_x = 55$, $r_y = 33$, where there was a water bubble inside the phantom. Therefore, good quality spectra were not even expected in this region because of the strong *magnetic susceptibility* artefacts. Thus, the quality of the spectra from the *online reconstruction* can be considered in a very good compliance with the *offline results*, i.e., the reference standard.

## 3.2 Functor runtime

In most of the *functors* there was no problem with respect to their runtime, as each of them was performed within lower units of seconds, also for larger matrix sizes, such as 64×64 pixels. But, this can not be said about the *spatial Fourier transformation functor*, the runtime of which was strongly dependent on the matrix size, as depicted in Fig. 3.32.

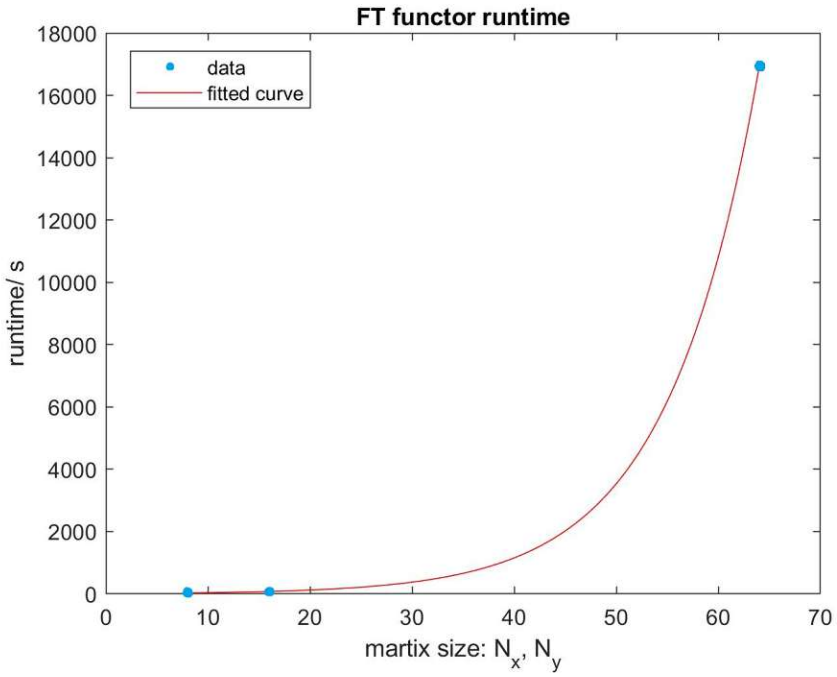


Figure 3.32: Runtimes of the *DFT functor* from the *online reconstruction pipeline* on the vertical axis, given in seconds, in dependence on the matrix size, where, e.g., 16 stands for 16×16. Measured data points are depicted as blue dots and the red line is the exponential fit.

To approximate the the dependence of the measured runtimes of the *DFT functor* on the matrix size, in Fig. 3.32, an exponential function provided the most appropriate fit. In the evaluation process, also, linear and quadratic fits were tested, but could not fit the runtime data as accurately as the exponential function. Possible solutions to the runtime issue of the *DFT functor* will be described in the *Discussion and Outlook* Chapter.

# 4 Discussion and Outlook

As fully-integrated reconstructed pipelines in ICE are generally not available as products on the scanner for fast MRSI sequences, an important progress was made in the frame of this thesis. Of course, the implementation is not yet fully optimized with respect to reconstruction runtimes, multi-channel acquisitions, and handling of 3D MRSI data, but, a firm starting point considering many important aspects was provided. Also, robust solution strategies were developed, which can be easily extended to cover the remaining aspects (e.g., *spatial DFT* in the z-direction), as well as some original and reliable acceleration methods (e.g., data-processing based on direct memory address increments).

## 4.1 Evaluation of functor runtime

Runtime of all *functors* from the CRT reconstruction pipeline was assessed as a function of matrix size. With the *functors FOV shift*, *frequency-offset correction*, and *density compensation* there was no problem regarding the runtime, as each of them was performed in under 4 seconds, even for larger matrix size (i.e., 64×64 voxels). Only the *spatial DFT* was examined in a greater detail, because of its overly long execution time, and should be further optimized.

Compared to solution 2, i.e., steady modification of *access specifiers*, the duration of the *Cartesian spatial DFT* was reduced by a factor of 4, when solution 1, i.e., direct memory address increments, was used. The higher computational demand of solution 2 was caused by repeated modification of the *access specifiers* and the use of ICE *functions*, which performed additional verification operations, and thereby, further increased the computational costs. Therefore, one possible acceleration strategy would be to implement the *spatial DFT* for the *concentric ring* trajectory according to solution 1, which was implemented for the *Cartesian DFT*.

Another way, how to accelerate the *spatial DFT* for the *concentric ring* trajectory, would be to, first, bring the data on a *Cartesian* grid and then perform *fast FT*, instead of the *discrete FT*. In that case, a uniform *k-space* density must be guaranteed, before the re-gridding, to minimize reconstruction errors. This could be easily achieved by applying the *density compensation* prior to the re-gridding [34]. Other possibility would be to implement and optimize a *non-uniform fast Fourier transformation* (NUFFT).

Also, the reconstruction seemed to run only on one CPU core. Certainly, that was one of the main reasons, why the reconstruction took such a long time.

## 4.2 Limitations

One of the limiting aspects of the presented software solution, regarding the spectral quality, are the high-frequency artefacts in the reconstructed FIDs and spectra, caused by the *density compensation*. The origin of those artefacts should be identified and subsequently eliminated.

As mentioned above, the runtime of the *spatial DFT functor* for the *concentric ring* data was problematic, as this reconstruction step took an unacceptably long time, especially for higher spatial resolutions. Nevertheless, the quality, in which this *functors* delivered the results, was excellent. Thus, it can be judged as a successful first step with future improvements to come.

Limited mainly by the runtime of the *spatial DFT*, extensive experimental validations were conducted only on a 16×16 matrix. Only the *complete reconstruction* was performed on data with higher spatial resolution (i.e., 64×64 voxels). In the images reconstructed with all of the data-processing steps on the 16×16 matrix, as well as on the 64×64 matrix, emerged significant artefacts, when compared to the offline standard. The image artefacts were quantified using the *normalized image contrast*, which was for the *complete reconstruction* in both spatial resolutions unacceptably high, i.e., around 20%.

Therefore, first, the online reconstruction, especially the *spatial DFT*, which demands an overwhelming majority of the reconstruction time, should be accelerated. Possible acceleration strategies

were mentioned in the previous Section on *Evaluation of functor runtime*. After that, each of the individual data-processing steps should be performed on a 64×64 matrix to see more details in the images. In this manner, the causes of the artefacts should be pursued to their origin and corrected.

Acceptable deviations of the online from the offline reconstruction should be around one percent of the *normal image contrast*. *Normal contrast* deviations around ten percent and above cannot be considered as a good result. When all of the results of all individual *functors* have been analysed on the 64×64 matrix, and the algorithms accordingly corrected, the same should be performed on in vivo measurements to guarantee the quality on relevant data.

Furthermore, the current version of the online CRT reconstruction pipeline is able to handle only 2D spectroscopic data. In future, the abilities of the online CRT reconstruction should be extended for the third spatial dimension to handle 3D data sets. Also, the ability of the online reconstruction to handle data from multiple receiver-coils should be, likewise, added. Those functionality extensions may be guided by the implementation of the online Cartesian reconstruction, as all of them have already been successfully implemented there.

## 4.3 Outlook

First, the online reconstruction should be accelerated by adjusting it in such a way that it runs on more CPU and/ or GPU cores simultaneously (i.e., parallelization). Second, to achieve even shorter reconstruction times, the *spatial DFT* could be implemented according to solution 1, i.e., based on direct memory address increments. Those two improvements combined should accelerate the *spatial DFT* sufficiently, as solution 1 brings about an acceleration factor of four. When the reconstruction will run on more CPUs, the runtime should be further decreased by a factor equal to the number of CPUs used, e.g., eight or sixteen. Another possibility would be to interpolate the data on a Cartesian grid and then perform a fast FT instead of the slow discrete FT.

The computational demand of the *spatial DFT* could be further lowered by applying a circular mask, already in the reconstruction, as is the case in the offline standard. In this thesis the circular mask was applied only manually in the evaluation process to generate images, but, all of the image pixels were reconstructed, which was not necessary. The computational load would be decreased significantly, approximately by 40 % for a 16×16 matrix and by 28 % for a 64×64 matrix.

Another major improvement regarding the reconstruction-time efficiency might be to transform all the *image functors* into *scan functors*, thus, enable immediate online processing of the data. The question is, whether this would be even possible because of the temporal interleaves, and also the fact that the data of the circles are spread over several ADCs, and therefore, are not arranged properly in a systematic *k-space*. If this would be possible, the whole reconstruction would run simultaneously with the acquisition and especially the *spatial DFT functor* would not have to wait until all of the data has been measured to receive the complete data object. Instead, it would be possible for all the *functors* to process the data in relatively small fractions (i.e., measured ADCs). When all of the data measured and separately processed , it would just need to be brought together at the end of the pipeline. This would lead to a net reconstruction time, $t_{reco,net}$, of approximately:

$$t_{reco,net} = t_{reco} - t_{acq} \tag{4.1}$$

In Eq. 4.1, $t_{reco}$ stands for the total reconstruction time and $t_{acq}$ for the acquisition time.

Regarding the calculation of the points on rings and their coordinates, besides the *rectangular* discrete integration method, there is another discrete integration possibility that relies on *trapezoids* instead. It can be further verified, which of those two methods delivers more precise results for the CRT trajectory coordinates, and whether all of the points on ring should be calculated by one of the two discrete integrations, or only the first point on circle and the rest should be calculated geometrically, as was done in the online reconstruction. It is assumed that calculating only the first point on each ring by means of a discrete gradient integration will deliver more accurate results. The reason for this assumption is that any discrete integration is not precise and carries

always inaccuracies with it, unlike the geometrical approach, which is precise and the uncertainty is contained only within the determination of the first point on each ring.

Ultimately, if the ICE reconstruction can be performed on the scanner sufficiently fast, this would be extremely important for distributing the sequence method to other research sites worldwide. This is of great importance because, currently, the huge amounts of raw data are stored and processed via a poorly optimized offline reconstruction pipeline that requires installation of various software packages and makes the process unnecessarily complicated.

## 4.4 Conclusion

The work accomplished in this thesis was just a first step. Even with parallelization and further optimization, it will be very challenging to reach a clinically attractive reconstruction time of a few minutes, at the most, for a typical 3D whole-brain MRSI data set. But, it is too early to say much, as reconstruction times can finally be shortened without spending more on faster computer hardware, which is not mostly decided by the MR scanner manufacturers anyway.

# Bibliography


[1] Kaseman, D., Iyer, R. *NMR: Introduction*. https://chem.libretexts.org/Bookshelves/, 2022.

[2] College of Chemistry and Biochemistry, University of California, USA. *Illustrated Glossary of Organic Chemistry*. http://www.chem.ucla.edu/harding/IGOC/N/natural-abundance.html, 2022.

[3] Feynman, R. *The Feynman Lectures on Physics*, pages 184 – 187. Addison-Wesley Pub. Co., Glenview, USA, 1988.

[4] Badurek, G. *Lecture: Medical and Biological Applications of Nuclear Physics II*. TU Vienna, Austria, 2020.

[5] Questions and answers in MRI. *Net Magnetization*. https://mri-q.com/net-magnetization-m.html, 2022.

[6] De Graaf, R. *In vivo NMR Spectroscopy - Principles and Techniques*, page 25 ff. John Wiley and Sons Ltd, Hoboken, USA, 2019.

[7] Lumen Learning. *Quantum Numbers*. https://courses.lumenlearning.com/cheminter/chapter/quantum-numbers/, 2022.

[8] Obodovskiy, I. *Chapter 2 - Nuclei and Nuclear Radiations*, pages 45 – 47. Radiation - Fundamentals, Applications, Risks and Safety, 2019.

[9] Murphy, A. *Gradient echo sequences*. https://radiopaedia.org/articles/gradient-echo-sequences-1, 2020.

[10] Murphy, A. *Spin echo sequences*. https://radiopaedia.org/articles/spin-echo-sequences?lang=us, 2020.

[11] Abdulla, S. *K-space*. https://www.radiologycafe.com/frcr-physics-notes/mr-imaging/k-space/, 2021.

[12] Moratal, D. *et al. K-Space tutorial: an MRI educational tool for a better understanding of k-space*. Biomedical Imaging and Intervention Journal, 2008.

[13] George, R. *et al. Bandwidth and image quality*. https://mrimaster.com/technique%20bandwidth.html, 2022.

[14] Gujar, S. *et al. Magnetic Resonance Spectroscopy*. Journal of Neuro-Ophthalmology, 2005.

[15] Rösgen, J. *Molecular Basis of Osmolyte Effects on Protein and Metabolites*. Elsevier, Galveston, USA, 2007.

[16] Ali, R., Nagalli, S. *Hyperammonemia*. https://www.ncbi.nlm.nih.gov/books/NBK557504/, 2021.

[17] Klose, U. *Measurement sequences for single voxel proton MR spectroscopy*. European Journal of Radiology, 2008.

[18] Ernst, R, Anderson, W. *Application of Fourier Transform Spectroscopy to Magnetic Resonance*. Review of Scientific Instruments, 1966.

[19] Questions and answers in MRI. *Chemical Shift Imaging*. https://mriquestions.com/csi.html, 2022.

[20] Al-iedani, O. *et al. Fast magnetic resonance spectroscopic imaging techniques in human brain applications in multiple sclerosis*. Journal of Biomedical Science, 2017.

[21] Strasser, B. *Acceleration of Magnetic Resonance Spectroscopic Imaging Sequences via Parallel Imaging and Spatio-Spectral Encoding*, page 14 ff. TU Vienna, Austria, 2017.

[22] Erasmus, L. *et al. A short overview of MRI artefacts.* South African Journal of Radiology, 2004.

[23] Haouimi, A. *Chemical shift artefact*. https://radiopaedia.org/articles/chemical-shift-artifact-1, 2021.

[24] Questions and answers in MRI. *MRS Artifacts*. https://mriquestions.com/mrs-artifacts.html, 2021.

[25] Hangel, G. *et al. Ultra-high resolution brain metabolite mapping at 7 T by short-TR Hadamard-encoded FID-MRSI.* NeuroImage, 2018.

[26] Independent Imaging. *The Difference between XRay, UltraSound, MRI, CT Scan.* https://www.independentimaging.com/abcs-imaging-difference-xray-ultrasound-mri-ct-scan/, 2022.

[27] Newman, P. *et al. Regions of the Electromagnetic Spectrum.* https://imagine.gsfc.nasa.gov/science/toolbox/spectrum-chart.html, 2013.

[28] Gonzalez, A., Sawyers, T. *MRI vs. PET Scan.* https://www.healthline.com/health/mri-vs-pet-scan, 2019.

[29] Tkac, I. *et al. Water and lipid suppression techniques for advanced 1H MRS and MRSI of the human brain: Experts' consensus recommendations.* NMR in Biomedicine, 2020.

[30] Questions and answers in MRI. *Water Suppression*. https://mriquestions.com/water- suppression.html, 2021.

[31] Andronesi, O. *et al. Motion correction methods for MRS: experts' consensus recommendations.* NMR in Biomedicine, 2020.

[32] Morelli, J. *et al. An Image-based Approach to Understanding the Physics of MR Artifacts.* RadioGraphics, 2011.

[33] Valkovic, L. *et al. Dynamic 31P–MRSI using spiral spectroscopic imaging can map mitochondrial capacity in muscles of the human calf during plantar flexion exercise at 7 T.* NMR in Biomedicine, 2016.

[34] Pipe, J., Menon, P. *Sampling Density Compensation in MRI: Rationale and an Iterative Numerical Solution.* Magnetic Resonance in Medicine, 1999.

[35] Becker, E. *High Resolution NMR: chapter 4 - Chemical Shifts*, page 83. Academic Press, San Diego, USA, 2000.

[36] Near, J. *et al. Preprocessing, analysis and quantification in single-voxel magnetic resonance spectroscopy: experts' consensus recommendations.* NMR in Biomedicine, 2019.

[37] Maudsley, A. *et al. Advanced magnetic resonance spectroscopic neuroimaging: Experts' consensus recommendations.* NMR in Biomedicine, 2020.

[38] Medical University of Vienna, Austria. *Infrastructure at the MR Center of Excellence, 3 and 7 Tesla*. https://hfmr.meduniwien.ac.at/en/3-and-7-tesla/, 2022.

[39] Moser, E. *et al. Windows on the Human Body – in Vivo High-Field Magnetic Resonance Research and Applications in Medicine and Psychology.* Sensors, 2010.

[40] Siemens Healthcare GmbH. *7T MRI Scanners, MAGNETOM Terra*. https://www.siemens-healthineers.com/magnetic-resonance-imaging/7t-mri-scanner/magnetom-terra, 2022.

[41] Werthner, H. *ICE User's Guide, Manual for ICE programmers*, page 29 ff. Siemens AG B Med, Erlangen, Germany, 2018.

[42] National Institute of Standards and Technology, Physics. *What Are Imaging Phantoms*. https://www.nist.gov/physics/what-are-imaging-phantoms, 2022.

[43] The MathWorks, Inc. *fftshift*. https://www.mathworks.com/help/matlab/ref/fftshift.html, 2022.

[44] Boyd, D. *Systems Analysis and Modeling: chapter 8 - Stochastic Analysis*, pages 213 – 214. Academic Press, San Diego, USA, 2001.