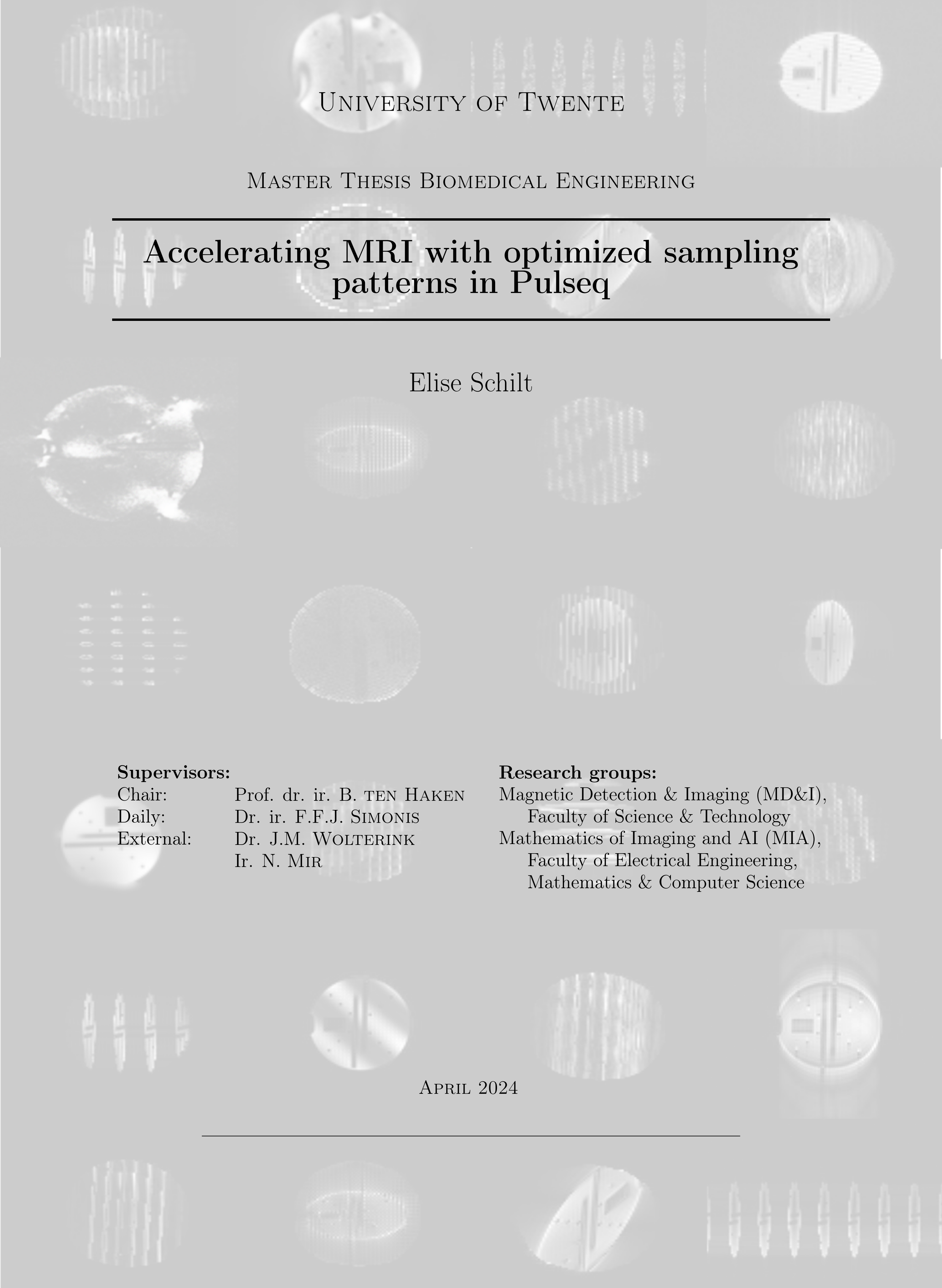


University of Twente

Master Thesis Biomedical Engineering

Accelerating MRI with optimized sampling patterns in Pulseq

Elise Schilt

Supervisors:

Chair:	Prof. dr. ir. B. ten Haken
Daily:	Dr. ir. F.F.J. Simonis
External:	Dr. J.M. Wolterink
	Ir. N. Mir

Research groups:
Magnetic Detection & Imaging (MD&I),
Faculty of Science & Technology
Mathematics of Imaging and AI (MIA),
Faculty of Electrical Engineering,
Mathematics & Computer Science

April 2024

Abstract

Yearly, about 1 million MRI scans are acquired in the Netherlands [1]. However, a major drawback of using MRI is the long duration of an MRI scan, leading to high costs, waiting lists, and images being affected by motion due to movement of the patient. In MRI, data is acquired in the spatial frequency (k-space) domain. Optimization of k-space sampling patterns, and improved reconstruction methods can reduce scanning time and/or improve image quality. In this work, optimized k-space sampling patterns are investigated, using custom-made pulse sequences in Pulseq [2, 3]. Sampling pattern optimization is done through conventional subsampling of a TSE and radial GRE sequence, as well as using an optimized sampling pattern learned by the deep learning model BJORK [4]. Sequences are successfully executed on a 1.5 T Siemens scanner (Aera, Siemens Healthineers, Erlangen), and data is reconstructed using the BJORK reconstruction model and a NUFFT.

The BJORK reconstruction model in this framework shows to be robust to different anatomies, trajectories, and data shapes. Comparison of reconstruction results show the BJORK reconstruction model performs better in reconstructing subsampled data than the NUFFT. The BJORK-optimized sampling pattern was successfully implemented in Pulseq and executed for both phantom and *in vivo* data. Eddy currents most likely influenced reconstruction results, but this is expected to be mitigated easily. The optimized sampling patterns, both subsampled and BJORK-optimized, showed quick acquisition of data. The acceleration needs to be further investigated by comparing this to state-of-the-art sequences. All in all, this work results in a robust framework to further investigate methods to accelerate MRI.

Contents

Abbreviations

ADC = analog-to-digital converter

BART = Berkeley Advanced Reconstruction Toolbox

BJORK = B-spline Parameterized Joint Optimization of Reconstruction and K-Space Trajectories

CG = conjugate gradient

CNN = convolutional neural network

CT = computed tomography

DC(F) = density compensation (function)

DIDN = Deep-Iterative Down-Up Network

(E)SPIRiT = iterative self-consistent parallel imaging reconstruction (E for Eigenvector maps)

ETL = echo train length

FID = free induction decay

FOV = field-of-view

FSE = fast spin echo, same as TSE

GRAPPA = GeneRalized Autocalibrating Partially Parallel Acquisitions

GRE = gradient echo

(i)FFT = (inverse) fast Fourier transform

ISMRM = International Society for Magnetic Resonance in Medicine

MRI = magnetic resonance imaging

PF = partial Fourier

PI = parallel imaging

PNS = peripheral nerve stimulation

PSF = point spread function

RF = radiofrequency

RSS = root sum-of-squares

SENSE = SENSitivity Encoding

SNR = signal-to-noise ratio

SSIM = Structural Similarity Index Measure

TBW = time-bandwidth product

TE$(_{eff})$ = (effective) echo time

TR = repetition time

(T)SE = (turbo) spin echo

UNN = unrolled neural network

Introduction

Magnetic resonance imaging (MRI) is a non-invasive imaging technique, widely used to diagnose patients and monitor disease. Since MRI does not use ionizing radiation such as computed tomography (CT), it is harmless for the patient. Also, MRI provides high-resolution images, and better soft tissue contrast than CT, so different types of tissues can be easily distinguished. This leads to easy diagnosis, which is why yearly, about 1 million MRI scans are done in the Netherlands [1]. However, a major drawback of using MRI is the long duration of an MRI scan. The duration of an MRI scan depends on the body part being imaged, and the desired resolution. On average, a brain MRI scan takes 30 minutes, leading to an average procedure time in the hospital of almost an hour [5]. This does not only increase the costs of the already expensive technique, it also creates waiting lists, meaning in the hospital Medisch Spectrum Twente in Enschede, in March 2024, patients have to wait more than a month to get their MRI examination [6]. The duration of the scan also increases the likelihood of patients moving, and thus images being affected by motion. Therefore, investigating methods to accelerate MRI is called for, while preserving diagnostic quality of the reconstructed images.

When making images with an MRI scanner, only already existing protocols can be chosen. These protocols are closed-source, so only some contrast and resolution parameters can be set in the scanner interface. Not every parameter is free to be adjusted, while the MRI scanner is capable of much more. These constraints make sure images acquired with these protocols are of high quality, but to examine acceleration of MRI acquisition, *new* protocols, that might result in lower quality images at first, need to be investigated. Also, the protocols on an MRI scanner are vendor specific, making it difficult to reproduce research done across scanners from different vendors.

MRI protocols come in the form of pulse sequences, consisting of magnetic gradients, radiofrequency (RF) pulses, and timings. When wanting to accelerate MRI acquisition, less data needs to be acquired, while maintaining the same diagnostic quality of the reconstructed image. This is called subsampling. Adjusting how (much) data is acquired is done by changing such pulse sequences. Therefore, an open-source and hardware-independent framework is needed to develop and execute pulse sequences on the MRI scanner. Pulseq is such a framework [2, 3], in which pulse sequences can be designed and executed on an MRI scanner of Siemens, Bruker or GE.

Instead of determining ways to optimally subsample data ourselves, artificial intelligence can also *learn* the most optimal way to sample data. Various deep learning models have been proposed learning optimal acquisition patterns [7], but physical feasibility and eventual reconstruction of images remains a challenge. To increase the feasibility, methods have been developed using the already known physics regarding MRI, and set hardware constraints. To solve the image reconstruction problem, methods simultaneously learning a sampling pattern and accompanying reconstruction model have emerged [8, 9]. One of those methods is BJORK: B-spline parameterized Joint Optimization of Reconstruction and K-space trajectories [4], outputting an optimal sampling pattern and accompanying reconstruction model. The research question that follows is:

How can we accelerate MRI using Pulseq and BJORK?

First the versatility of Pulseq and BJORK needs to be examined by designing and executing subsampled pulse sequences in Pulseq. By reconstructing this data comparing conventional methods and BJORK, the effect of subsampling on the resulting reconstructed image can be examined. Then, an optimized sampling pattern learned by BJORK can be examined, leading to the following two subquestions:

1. **To what extent can we subsample, and how does it impact reconstruction quality?**
2. **What is the feasibility of a BJORK-optimized sampling pattern, and how does it affect reconstruction quality?**

The first chapter will dig into the necessary background regarding MRI and deep learning, whereafter the second chapter will address the first subquestion. Subsampling and reconstruction will be examined, whereafter chapter three will address the second subquestion. Here, implementation of the optimized sampling pattern will be examined, followed by results, discussion and conclusion. The thesis will be concluded with a general discussion on the results and limitations of the research, and recommendations for future research.

1 Background

In this chapter, the theoretical basis of this thesis will be described. First, an introduction is given into MRI and subsampling, whereafter reconstruction techniques follow. Methods to accelerate acquisition are explained, concluding with an introduction to Pulseq.

1.1 Introduction to MRI

In MRI, magnetic fields are used to image structures. When an external magnetic field (B_0) is applied to an object such as tissue, hydrogen nuclei start to precess at their Larmor frequency. This Larmor frequency (f_0) is determined by the field strength and the gyromagnetic ratio (γ) of the specific nuclei through: $f_0 = \gamma B_0$. The spins of the nuclei tend to align with the magnetic field to achieve a lower energy state, where the sum of all those spins results in a precessing net magnetization (M). When applying an external magnetic field B_1 oscillating at the Larmor frequency of the nuclei, M is pushed out of alignment. B_1 is generated by RF coils. The short periods of time when the B_1-field is turned on are therefore called RF pulses. How much M is pushed out of alignment relative to B_0 depends on the flip angle of the RF pulse. For a 90° RF pulse, M will tip from the longitudinal M_z-plane into the transverse M_{xy}-plane. After the RF pulse, two processes occur responsible for the signal recorded in MRI: 1) Due to intermolecular interactions, the precessing spins of M dephase in the M_{xy}-plane, see the bottom of Figure 1. Since a spatially varying magnetic field induces a time varying electric field (Faraday), the resulting current from this process can be recorded in receiver coils. This recorded current decreases until all spins are out of phase, resulting in the free induction decay (FID) signal in Figure 2. 2) Due to thermal agitation, M also realigns with B_0 again, meaning M_z "grows-back" [10]. This is also visualized in Figure 1, at the top. Both the dephasing in the M_{xy}-plane, and the regrowth in the M_z-plane are called relaxation. Since the relaxation time is specific for different nuclei and B_0 strengths, the differences in the amount of magnetization at a certain time point can be used to differentiate between different tissues. The time between the RF pulse and the center of the acquired signal is the echo time (TE).

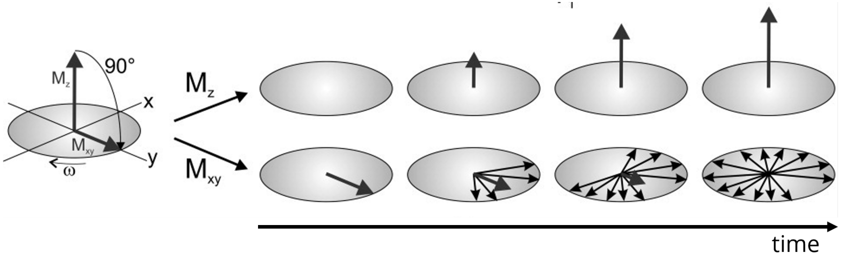


Figure 1: Relaxation of M_z (top) and M_{xy} (bottom) over time after a 90° RF pulse. Taken from [11].

A way to influence the de- and rephasing of spins is by applying an external gradient. When applying a dephasing and rephasing gradient subsequently, this results in not an FID as recorded signal, but a gradient echo (GRE), see the right side of Figure 2 [12].

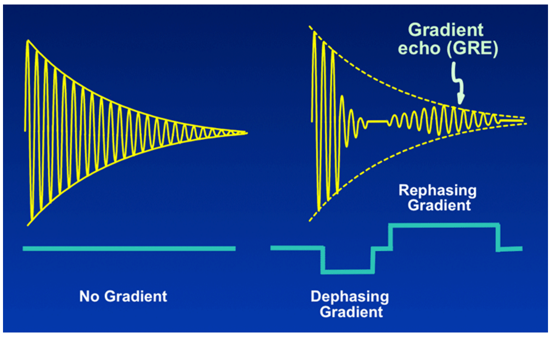


Figure 2: FID signal (left), where the decay of signal due to relaxation is visible. In the GRE (right), it is seen how the signal changes due to applied gradients. Adapted from [13, 14].

When applying these gradients in the x-, y-, and z-direction, different locations in the imaged object will experience different local fields, and therefore the *location* of the signal can be retrieved. So, the eventually recorded signal varying over time, $s(t)$, will contain spatial information about the imaged object through Equation (1) [10]:

$$s(t) = \mathcal{M}(k_x(t), k_y(t)) \tag{1}$$

These spatial frequencies (k_x, k_y) are "stored" in a grid called k-space. During signal acquisition, k-space is filled. When completely filled, so fully sampled, an inverse (Fast) Fourier transform (iFFT) of this k-space transforms the spatial frequencies into an image. This relationship is visualized in Figure 3. Higher spatial frequencies are located at the outer regions of k-space, and correspond to sharp transitions and fine details in the imaged object. Lower spatial frequencies can be found at the center of k-space and are associated with more smooth and large-scale features in the imaged object. The highest spatial frequency acquired in k-space, k_{max}, determines the pixel width Δw in the image, which is also visualized in Figure 3, and can be seen in Equation (2). In Figure 3, it can also be seen how the field-of-view (FOV) of the image is related to the distance between sampled points in k-space, also noted in Equation (3). The number of points sampled (N_x) can be retrieved through $k_{FOV} = N_x \cdot \Delta k$.

$$\Delta w = \frac{1}{k_{max}} \tag{2}$$

$$\Delta k = \frac{1}{FOV} \tag{3}$$

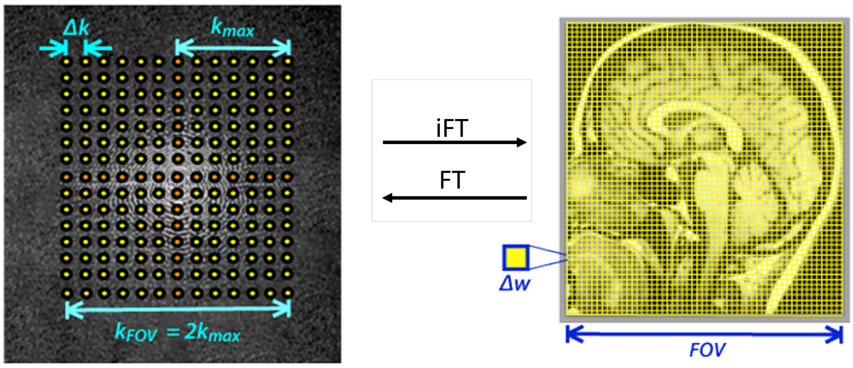


Figure 3: Relationship between k-space (left) and image-space (right), and accompanying distances Δk and Δw. Taken from [15, 16] with minor adjustments. K-space is converted to image-space by applying an inverse Fourier transform, and image-space is converted to k-space by applying a Fourier transform. Also see the mathematical relationships in Equation (2) and Equation (3) corresponding to this relationship.

The location of the sampled points in k-space is determined by the area, and direction of the gradients used, following Equation (4).

$$k_{x,y}(t) = \gamma \int_{t=0}^{t=T} G_{x,y}(t)\, dt \tag{4}$$

So, the position in k-space can be altered by altering the strength, direction, and duration of the applied gradients. By combining different gradients, a "route" through k-space forms, called a trajectory. An example of a trajectory through k-space can be seen in Figure 4a. This trajectory is a Cartesian trajectory, meaning data is acquired rectilinearly. The time each line takes is the repetition time (TR).

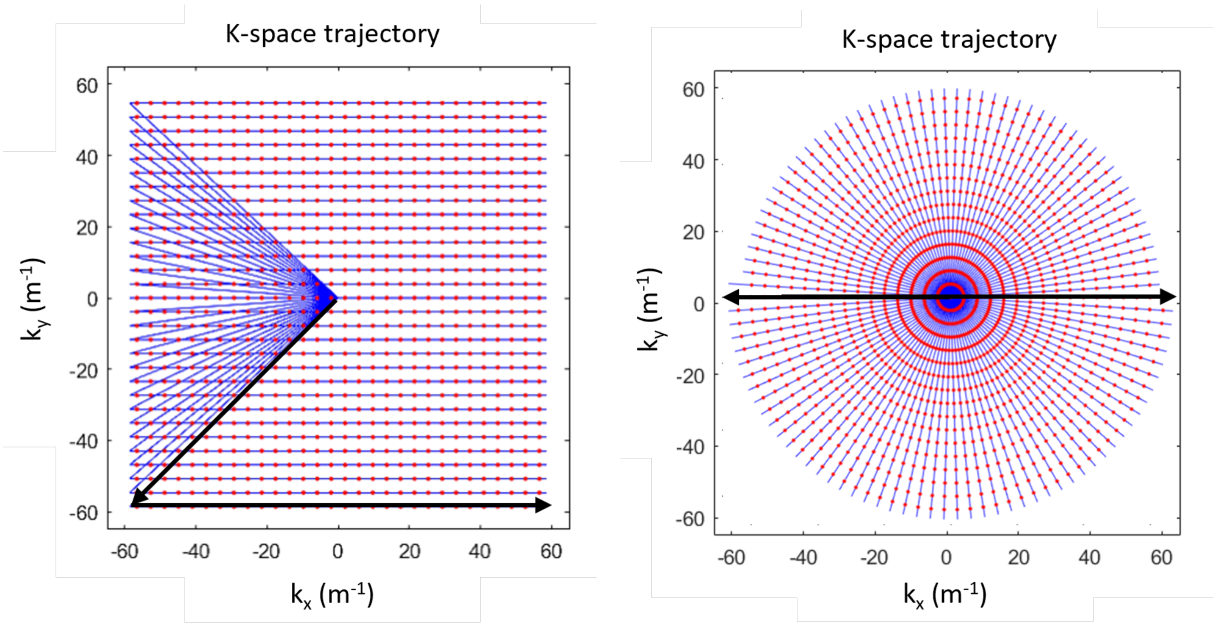
As can be seen in Figure 4a, the trajectory is "reset" to the center of k-space after each line (each TR). This is the effect of the RF pulse, but not the only use. RF pulses in combination with the gradient in the z-direction, are also responsible for slice selection. The thickness of the slice selected is determined by the bandwidth of the RF pulse (BW_{rf}) and the amplitude of the slice selective z-gradient ($G_{z,ss}$), see Equation (5).

$$\Delta z = \frac{BW_{rf}}{\left(\frac{\gamma}{2\pi} G_{z,ss}\right)} \tag{5}$$

The bandwidth of the RF pulse is the range of frequencies excited. This is related to the duration of the pulse (t_{rf}) through the time-bandwidth-product (TBW), see Equation (6). By shifting the frequency of the RF pulse, different slices can be excited.

$$TBW = t_{rf} \times BW_{rf} \tag{6}$$

When combining a number of RF pulses, gradients, and ADC read-out, a pulse sequence forms. In Figure 4a, the k-space trajectory resulting from a Cartesian sequence was shown. There are many other sequences not only leading to different k-space trajectories, like the non-Cartesian trajectory in Figure 4b, but also to different image contrasts due to timing differences in the sequence.



(a) Cartesian trajectory.

(b) Radial (non-Cartesian) trajectory.

Figure 4: K-space trajectories with in red the sampling points, and in blue the route through k-space. In black arrows the first part of the route: starting at (0,0) and going outwards, with sampling points being acquired along the horizontal line. Made in Pulseq [2, 3].

The GRE sequence was already shortly explained with Figure 2. An example of another sequence is the spin echo (SE) sequence. Here, not one RF pulse as in the GRE from Figure 2 is used, but two RF pulses are used in each TR. The first 90° pulse tips the spins into the transverse plane, whereafter the second 180° RF pulse flips them into the mirrored plane. The spins that were slowest before the 180° pulse will now have caught up, resulting in an echo when the spins are in phase with each other again. When using multiple 180° pulses consecutively in combination with a changing y-gradient, multiple echos can be acquired within one TR, resulting in multiple lines in k-space within one TR. Such a spin echo is called a turbo or fast spin echo (TSE or FSE). The TE to the middle echo in the train is called the effective echo time (TE$_{eff}$). By acquiring multiple lines in k-space during one TR, acquisition is sped up.

1.2 Subsampling k-space

Beside echo train imaging, another way to speed up acquisition is to not acquire a full k-space. The k-space is then subsampled, of which Figure 4b was already an example: the periphery of k-space is sparsely sampled. Since the center of radial trajectories is oversampled, the information most of interest is still acquired. Radial sequences are also less affected by motion, and subject to less coherent artifacts [17]. Especially golden angle radial sampling, in which the full radial spokes are acquired using a 111.25° angle between them, is very robust to motion [18].

Another subsampling scheme is partial Fourier imaging, first introduced by [19]. This technique exploits the conjugate symmetry of k-space, implying only half of k-space needs to be recorded, whereafter the data can be mirrored. In reality, the symmetry is not perfect, so still slightly more than half of the k-space needs to be recorded to get a good reconstructed image. When removing more than half of the k-space, artifacts become prevalent.

Artifacts such as aliasing (wrap-around) occur when the Nyquist-Shannon criterion is not met. This criterion states that the digital sampling rate should be at least twice the highest frequency being sampled [20]. In radial sequences, this means that with less than $\frac{\pi}{2}(n_{points})$ spokes, aliasing artifacts will occur [17]. Inspection of the point spread function (PSF) of the subsampled trajectory can aid in understanding the appearance of the artifact [21], as is visualized in Figure 5.

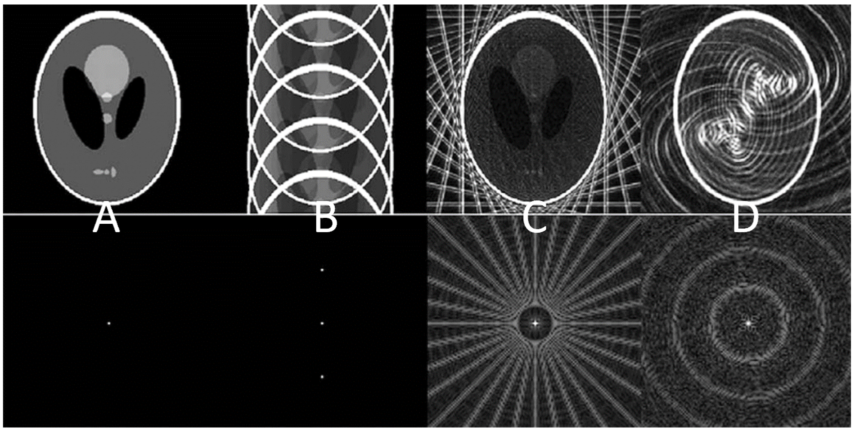


Figure 5: Reconstructed images of Shepp-Logan phantom (top) and PSF of the data (bottom). A) a fully sampled Cartesian reference. Here the PSF is one clear point in the middle. B) rectilinearly subsampled data. See how the type of artifact corresponds to the PSF, C) subsampled radial data resulting in streaks in the entire reconstruction, and D) subsampled spiral data. Figure taken from [17].

Parallel imaging (PI) is another technique using subsampled trajectories. In PI, data is acquired using multiple receiver coils, allowing reconstruction of missing data [22, 23]. By using the sensitivity of the coils surrounding the object, a degree of spatial encoding is provided [16], resulting in not needing to acquire the entire k-space. Since nowadays all clinical scanners use multiple coils to acquire data, PI is widely used. A downside with parallel imaging reconstruction is that noise is prone to be amplified [24].

When subsampling, one can also focus on leaving out exactly *that* data that is not of interest, thus only acquiring the relevant data. However, deciding what data is relevant and what not is challenging. Here, deep learning methods come into play. Since specific anatomies exhibit specific distributions of spatial frequencies, deep learning models can learn an "optimal" sampling pattern tuned to a specific imaging goal.

For fully sampled k-spaces on a rectilinear grid, performing an iFFT suffices to reconstruct the image. However, when using non-Cartesian trajectories or subsampling schemes, other reconstruction methods are needed.

1.3 Image reconstruction

Mathematically, image reconstruction in MRI is often written as Equation (7):

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \epsilon \tag{7}$$

Where $\mathbf{y} \in \mathbb{C}^M$ is the acquired signal consisting of M samples, $\mathbf{x} \in \mathbb{C}^N$ the reconstructed image with N pixels, $\mathbf{A}$ an encoding matrix relating the two, and ϵ some measured noise. When both the pixel coordinates of the image and the k-space sampling points lie on a Cartesian grid, the matrix A is simply a Fourier transform. To then retrieve image x from signal y, an inverse Fourier transform is required, as was also already seen in Figure 3. However, when the data is acquired using non-Cartesian trajectories, or when subsampling so $M < N$, a simple inverse Fourier transform does not suffice anymore [25]. Various methods exist to reconstruct either subsampled or non-gridded k-space data of which the ones relevant to this thesis are explained more in-depth.

When k-space is not sampled on a Cartesian grid, the most obvious way to reconstruct an image is via an operation called gridding. This means the acquired samples are interpolated onto a Cartesian grid, whereafter a normal inverse Fourier transform will give the reconstructed image. Gridding can be done through an adjoint non-uniform Fourier transform (NUFFT), sometimes with the help of a density compensation function (DCF) taking into account the sampling density of the data [17, 26, 27]. In gridding methods, the sample locations, so the performed trajectory, needs to be given as input. Important to note is that the actually performed trajectory could deviate from the theoretical trajectory due to hardware imperfections, which makes radial reconstruction even more challenging [4, 28]. Other reconstruction methods include SENSE, GRAPPA (both PI reconstruction methods), and deep learning models.

1.3.1 Parallel imaging (PI)

In PI, the susceptibility of the receiver coils over space is used to help localizing where the signal comes from. This can be done both in image- as in k-space [22, 24].

One method operating solely in the image domain is SENSE (sensitivity encoding) [29]. Here, each signal will be given a weighting based on how sensitive a coil is at that location. Then the unaliased pixels are retrieved by combining the sensitivity maps and the signal from one location in all the coils [24]. This mathematically requires the inversion of an encoding matrix containing the spatial encoding information (including coil sensitivities). In radially acquired data, a large number of pixels overlap, making the inversion of the encoding matrix computationally very demanding. In this case, it can be solved using an iterative conjugate gradient (CG) algorithm, which starts with an initial guess for the final reconstruction, and then iterates until convergence. A downside is that this stopping criterion needs to be exactly tuned so that all aliasing artifacts have been removed, but noise is not amplified extra. The result is a reconstructed image where the coils have already been combined. Some extra density compensation and intensity correction can also applied [17]. An important downside to SENSE is the need of accurate sensitivity maps.

A reconstruction method operating solely in the k-space domain is GRAPPA (generalized autocalibrating partially parallel acquisitions) [30]. In GRAPPA, the missing k-space data is synthesized per coil by computing coil weighting factors from a fully-sampled autocalibration region in k-space. This means no seperate coil sensitivity maps are needed, making GRAPPA more robust than SENSE. However, with radially acquired data, more calibration data needs to be collected to synthesize each missing data point [24]. This results in the need for more than one, or one large fully sampled non-Cartesian dataset for calibration, which could cancel the time gain subsampling gave in the first place [17].

Similar to GRAPPA, SPIRiT (iterative self-consistent parallel imaging) uses calibration data to fill in missing k-space data. Instead of only using acquired data to synthesize unknown points, here *all* points (also synthesized points) surrounding a missing data point are used to compute the missing data point. A data consistency term enforces that no actually acquired data points are changed. Then it iterates until both the calibration and data consistency terms are below a threshold. Since this is an iterative process, it takes longer than SENSE or GRAPPA, but it can be used for any arbitrary sampling pattern [31, 24].

A connection with SENSE is made when looking at ESPIRiT (E for Eigenvector maps). In ESPIRiT, a kernel slides over the calibration data, and the output is put into a calibration matrix. A singular value decomposition (SVD) is computed, and the singular values are converted into k-space kernels. When these singular values are close to zero, it indicates the presence of a null space. That implies that neighboring k-space points correlate throughout the coils. When the null space is too small, this means not all correlations in the data are captured, leading to incorrect sensitivity maps. That null space is determined by applying a (manually set) threshold. When using ESPIRiT in parallel imaging reconstruction, this relationship between coils is extended to the entire k-space to synthesize the missing data points. However, ESPIRiT can also be used to compute sensitivity maps of the coils themselves. After thresholding the singular values, the k-space kernels are cropped and an eigen-value decomposition is done (in image-space). Every location where the eigenvalue is not equal to one is said to contain no signal, so is set to zero. The set of eigenvectors with eigenvalues equal to one, look like sensitivity maps. The last step is cropping the sensitivity maps using a second threshold so that only the regions with signal are left [32, 24].

Since sensitivity maps of coils are specific for the object being imaged, sensitivity maps are needed for each individual experiment [4]. Often these are made by doing a pre-scan [24], but as explained above, they can also be computed by using calibration data as is done in the ESPIRiT method [32]. Since calibration data in GRAPPA needs to be fully sampled on a Cartesian grid, preference goes out to using ESPIRiT for non-Cartesian k-space trajectories. However, the manual thresholding in ESPIRiT is not ideal, and it can occur that there is a too small null space being spanned by the singular values of the calibration matrix, leading to low quality sensitivity maps. This can especially happen when using data with a low signal-to-noise ratio (SNR) [32].

1.4 Deep learning in MRI

1.4.1 Deep learning in image reconstruction

While SENSE, GRAPPA, and ESPIRiT are physics-based reconstruction methods, deep learning methods are learning-based. This means the mapping from k-space to a reconstructed image is implicitly learned [33]. Where SENSE, GRAPPA, and ESPIRiT might not capture all the relationships present in the data, learning-based methods can lack interpretability and can have difficulty generalizing for unseen data. The lack of interpretability can be (partly) overcome by incorporating certain aspects from a physical model into the algorithm [34], or by keeping a human in the loop. Also uncertainty estimation can aid in indicating where a learning-based model's output needs to be checked by a human [35]. This will decrease the possibility that elements in the final image are removed or added by the model, which is of utmost importance in medical imaging. Generalizability can be increased by ensuring the type of training data reflects the aim of the model. For instance when a model is designed to identify pathologies in scans of sick patients, generalizing the model will be challenging when only training on scans of healthy patients. Also, the processed training data should resemble authentic data [36], and the type of metric used in training should reflect what is important in the clinical setting [37]. A metric often used to assess similarity between an output and ground truth, is the structural similarity index measure (SSIM) [38]. The SSIM gives a value between 0 and 1, where an SSIM of 1 indicates complete similarity between the two images. However, it should be kept in mind that a high SSIM does not necessarily mean the diagnostic quality of the image is still sufficient.

Besides learning the mapping from the k-space domain to the image domain, deep learning can also be used to estimate the missing lines in k-space [39], or to denoise the data before or after reconstruction. The simplest denoiser is a convolutional neural network (CNN). A CNN uses convolutional layers to learn patterns and features in the data. A type of CNN often used for denoising images is a U-Net [40]. Since a U-Net uses a contraction and expansion path, it excels at learning image features at different representations of the image. CNN's can also be used to learn the conversion between k-space and an image, but the state-of-the-art within these methods are unrolled neural networks (UNN), sometimes combined with physical models. In these networks, each iteration is one layer of the network. UNNs excel at learning temporal relationships in data. This makes them effective in image reconstruction due to both checking if the image is still physically corresponding to the measurements, while also improving the images by comparing features in neighboring images [41].

1.4.2 Deep learning in optimizing k-space sampling

As mentioned in Section 1.2, when subsampling, one can also try to determine what data in k-space is the most important to obtain a good reconstructed image. When using deep learning, this comes down to an optimization problem. Since specific anatomies exhibit specific distributions of spatial frequencies, deep learning models could even learn an "optimal" sampling pattern tuned to a specific imaging goal. In SPARKLING [7], a target sampling density is used that is constrained with hardware limitations. This works quite well, but that does not find an optimal sampling density for a specific anatomy [4]. Also, reconstruction takes a long time [7], and it is not certain that this deep learning reconstruction method would work on actually acquired data. The challenge is to compute an actually physically feasible sampling pattern that truly saves time when executing. For instance, too sharp transitions in k-space result in the need for too rapid switching of gradient. Point distribution based (random) sampling patterns such as proposed in [42, 43], are not feasible unless performed in 3D, which also highlights the danger of using retrospectively subsampled data. Lastly, implementing learned sampling patterns can lead to the need for computationally expensive reconstruction methods due to the irregularity or sparsity of the sampled k-space data. Therefore, methods jointly learning a physically feasible optimized sampling pattern and accompanying reconstruction model have emerged.

BJORK One of these methods is BJORK: B-spline parameterized Joint Optimization of Reconstruction and K-space trajectories [4]. BJORK is a deep learning model jointly learning an optimized sampling pattern and reconstruction method. The sampling and reconstruction parts of the model can also be used individually from each other, meaning the reconstruction model can be applied to any raw data acquired with any trajectory.

The training was done with Algorithm 1. An overview of training can also be seen in Figure 6. During training, the sampling pattern $\omega \in \mathbb{R}^{N_s \times N_d}$, and the reconstruction neural network θ_f are optimized. N_s is the total number of sampling points and N_d the dimensionality of the image (so for 2D images N_d is 2). For the reconstruction model, the trainable parameters are $\theta \in \mathbb{R}^M$, with M being the parameters.

First, the trajectory ω is described by basis functions of b-splines $B \in \mathbb{R}^{N_s \times L}$ and coefficients $c \in \mathbb{R}^{L \times N_d}$, whereafter the reconstruction and trajectory weights are initialized. The algorithm further consists out of four loops. The first loop is over the number of optimization levels, which allows for multi-level optimization. When using more than one optimization level, more local adjustments can be made to the trajectory. This is done through decreasing the decimation rate N_s/L: how many sampling points per spoke are controlled by one interpolation kernel. Then on line 8 of the algorithm, for each epoch, training batches are loaded from the training set $\mathcal{X}$, which contains fully sampled MR images $\bar{x} \in \mathbb{C}_v^N$ with N_v being the number of voxels. The k-space $\mathbf{y}^K$ is simulated as if x^K had been acquired using ω_l, and some noise ϵ is added. The forward operator $A(\omega) \in \mathbb{C}^{N_s N_c \times N_v}$ is a non-Cartesian SENSE operator. Images are reconstructed using the UNN consisting of data consistency and denoising blocks. The data consistency blocks use a CG to compute the initial images through Equation (8):

$$\begin{aligned} \boldsymbol{x}_0 &= \arg\min_x \|\boldsymbol{Ax} - \boldsymbol{y}\|_2^2 + \lambda \|\boldsymbol{Rx}\|_2^2 \\ &= (\boldsymbol{A'A} + \lambda \boldsymbol{R'R})^{-1} \boldsymbol{A'y} \end{aligned} \tag{8}$$

Here R is the N_d-dimensional first-order finite difference operator. The reconstruction algorithm is iterated over, in which x_{i+1} is calculated through Equation (9):

$$x_{i+1} = \arg\min_x \|Ax - y\|_2^2 + \mu \|x - z_i\|_2^2 \tag{9}$$

Here μ is a positive penalty. This is solved using CG as well, but now with Toeplitz embedding to speed up the computation [44]. Then the denoising CNN $\mathcal{D}_\theta$ is applied to compute z_{i+1}. This denoising CNN is based on the Deep-Iterative Down-Up CNN (DIDN) from [45]. The loss function is a combination of L1 and SSIM. Finally, the reconstruction and trajectory weights are updated.

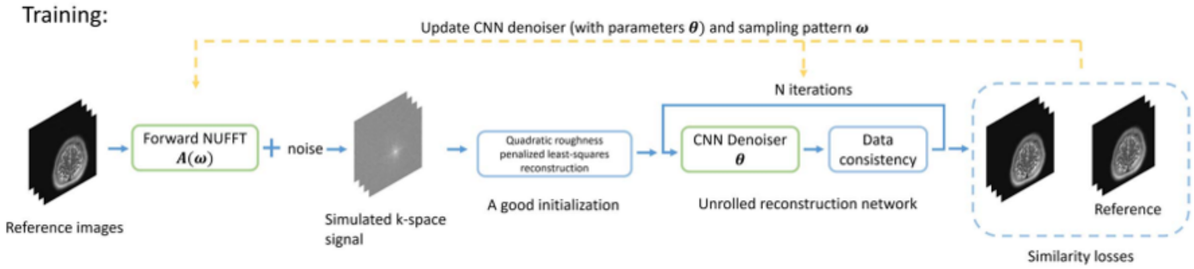


Figure 6: Overview of training method of BJORK, taken from [4]. Training starts with reference images that have been acquired using Cartesian trajectories. Then, a k-space is simulated as if the images had been acquired using a radial trajectory ω, using the forward NUFFT operator $\boldsymbol{A(\omega)}$. An initial estimation of a reconstructed image is computed using Equation (8), that is used as input to the UNN, consisting of denoiser and data consistency blocks. The reconstruction and sampling weights are updated based on the loss between the output and the reference, and the process repeats.

Algorithm 1 BJORK training algorithm, adapted from [4]

```
Require: Training set $\mathcal{X}$; denoiser $\mathcal{D}_\theta$ for initial CNN weights $\boldsymbol{\theta}_0$; initial trajectory
    $\omega_0$; levels of optimization $N_{\text{level}}$ ; number of epochs $N_{\text{epoch}}$ ; step size of denoiser
    update $\eta_{\mathcal{D}}$; step size of trajectory update $\eta_\omega$; penalty parameter for gradient/slew
    rate constraint $\mu_1$ and $\mu_2$.
Ensure: $\boldsymbol{\omega} = \boldsymbol{Bc}$.
$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta_0}$.
$\boldsymbol{\omega} \leftarrow \boldsymbol{\omega_0}$.
Pre-train $\mathcal{D}_\theta$ with fixed $\omega_0$.
for $l = 1$ to $N_{\text{level}}$ do
    Initialize new coefficient matrix $\boldsymbol{B}_l$.
    Initialize new coefficients $c_l^0$ with $\boldsymbol{\omega_{l-1}} \approx \boldsymbol{B}_l \boldsymbol{c}_l^0$.
    for $j = 1$ to $N_{\text{epoch}}$ do
        for training batches $x^K$ in $\mathcal{X}$ do
            Simulate k-space w.r.t. $\omega_l$:
            $y^K = A\left(\omega_l^K\right) x^K + \varepsilon$
            Reconstruction with UNN:
            Reconstruct initial images using Equation (8) with CG
            for $i = 1$ to $N_{\text{iter}}$ do
                $\mathbf{x}_{i+1}$: UNN reconstruction update of $\mathbf{z}_i$ using Equation (9)
                Apply CNN: $z_{i+1} = \mathcal{D}_{\boldsymbol{\theta}}\left(\boldsymbol{x}_{i+1}\right)$
            end for
            Calculate loss function:
            $L = \ell\left(\hat{\boldsymbol{x}}^K, \boldsymbol{x}^K\right) + \mu_1 \boldsymbol{\phi}_{\gamma\Delta t\,\mathrm{tg}_{\max}}\left(\left|\boldsymbol{D}_1 \omega_i^K\right|\right) + \mu_2 \boldsymbol{\phi}_{\gamma\Delta t^2 s_{\max}}\left(\left|\boldsymbol{D}_2 \boldsymbol{\omega}_i^K\right|\right)$
            Update denoiser and trajectory:
            $\boldsymbol{\theta}^K = \boldsymbol{\theta}^{K-1} - \eta_{\mathcal{D}} \nabla_{\boldsymbol{\theta}^{K-1}} L$
            $\boldsymbol{\omega}_l^K = \omega_l^{K-1} - \eta_\omega \nabla_{\omega_l^{K-1}} L$
        end for
    end for
end for
```

Implementation of BJORK The BJORK model used in this thesis was implemented and trained by T. de Haan [46], based on the Github[1] code and [4] in PyTorch [47]. It was trained on multi-coil knee MRI scans from the fastMRI dataset made by NYU school of medicine and Meta [48, 49]. The data consisted of coronal scans acquired using a TSE protocol with proton-density weighting, made with field strengths of 1.5 and 3 Tesla. The dataset contained raw k-spaces as well as DICOM images, but for training only the raw k-spaces were used. These fully sampled raw k-spaces were reconstructed into images for each coil, where the ground truth image and eventual input to the model was a root-sum-of-squares (RSS) combination of the coils. The training data was acquired using a 140 x 140 mm^2 FOV and a 320 x 320 k-space matrix. Echo train length varied between 4 and 5, TE was between 27 and 32 ms, and TR was between 2750 and 3000 ms. Fat suppression was used in a part of the scans. From the training dataset, 200 scans containing 7049 slices were used as training input, not distinguishing between scans performed using different protocols.

Parameters used in training by [46] can be seen Table 1. The initial trajectory ω_0 was radial, and consisted of 32 spokes and 1280 points. For the reconstruction part of the network, 4 reconstruction blocks were used containing 32 channels each. Note that the number of optimization levels was set to one, meaning the decimation rate stayed constant throughout training. Therefore, every spoke in the trajectory containing 1280 sampling points, is controlled by 40 b-spline kernels.

After training, the reconstruction model was tested on fastMRI validation data, and demonstrated a

[1] https://github.com/guanhuaw/Bjork

SSIM of 0.82 compared to the validation dataset. Compared to the Cartesian trajectory with which the training and validation data was acquired, the resulting trajectory had a 10x acceleration.

In this thesis, the BJORK model resulting from the training in [46] will first be used to reconstruct subsampled k-space data. Thereafter, the optimized sampling pattern obtained from this BJORK implementation will be made into a pulse sequence, and be used to acquire phantom and *in vivo* data.

Training parameter	Value used in [46]
N_{level}	1
$decim$	32
N_{epoch}	3
N_{iter}	6
CG_{iter}	10
Training batch size	4
μ	10
λ	0.01
g_{max}	0.9×45 mT/m
s_{max}	0.9×200 T/m/s
μ_1	1
μ_2	0.01
lr sampling	10^{-3}
lr reconstruction	10^{-5}

Table 1: Training parameters used in implementation of BJORK by [46]. decim = decimation rate, lr = learning rate, chosen based on [4]. Note that since $N_{level} = 1$, the decimation rate was not decreased during training.

1.5 Pulseq

When trying to execute an optimized sampling pattern on an MRI scanner, one will run into the problem that pulse sequences are vendor specific and MRI scanners are closed-source. Fortunately, open-source and hardware-independent frameworks have been developed which allow the design and execution of custom-made pulse sequences on MRI scanners of different vendors. Pulseq is such a framework [2, 3], consisting of Matlab and Python (PyPulseq) source code, downloadable from Github[2]. Pulseq also supplies demo pulse sequences, simple reconstruction scripts, and documentation. Tutorials and extremely useful slides from ISMRM virtual meetings can also be found on the Github page. When the sequence design is complete, a .seq file in C++, readable for the scanner, can be generated in Pulseq and uploaded on the scanner. The sequence can then be executed using the interpreter module on the scanner. For an overview of the Pulseq workflow, see Figure 7. In this thesis, Pulseq version 1.4.1 was used in Matlab with the Siemens interpreter.

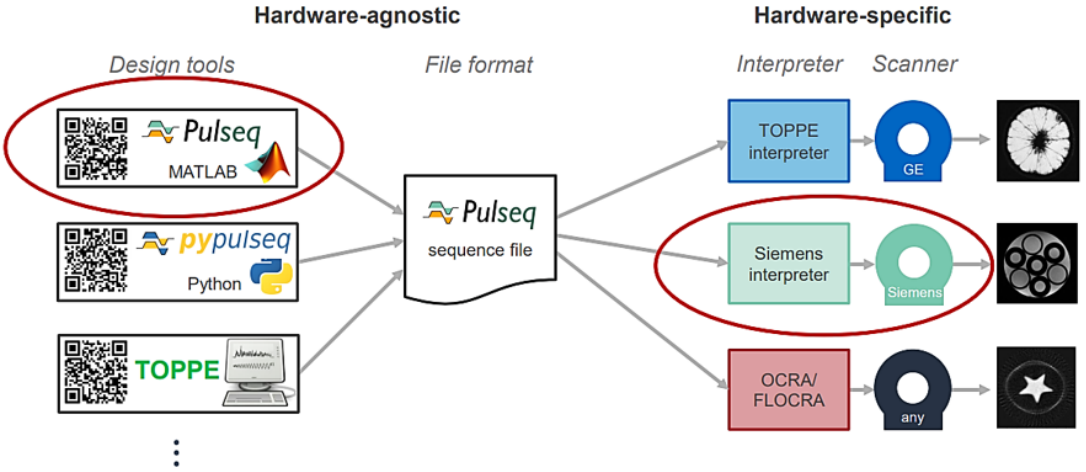


Figure 7: Pulseq workflow, taken from [50]. In this thesis, the circled blocks are used: Pulseq in Matlab and the Siemens interpreter. Note that this figure is misleading since not a reconstructed image, but raw data is the output of the scanner after running the sequence file.

Before digging into sequence design, it needs to be clear how Pulseq defines sequences. In Pulseq, a block concept is used to represent sequences. This means every sequence is a concatenation of RF, gradient and/or ADC blocks. The blocks cannot overlap, but can contain both an RF pulse, ADC event, and gradient waveforms along all three axes. The timings of the events within the blocks can be tuned using delays, where the duration of the block is defined by the longest event.

When starting to design a sequence in Pulseq, first the system properties need to be defined for the scanner that is gonna be used to run the sequence on. This contains the value of the B0 field of the scanner, the gyromagnetic ratio, appropriate raster times for the gradients, RF and ADC, a maximum gradient and slew rate, the ring-down time of the RF pulse and the deadtime of the RF pulse and ADC. For the 1.5 T Siemens scanner (Aera, Siemens Healthineers, Erlangen) used to execute the sequences on in this work, the system properties can be found in Table 2. Especially the raster times are of much importance when designing a sequence, since these define the time over which the specific event can be defined. This means that for instance for a gradient waveform with duration 1 ms, only 100 different amplitudes can be defined ($\frac{1\,ms}{10\,\mu s}$). When incorrectly or not defining the system properties correctly, the sequence will likely not run on the scanner.

[2] https://github.com/pulseq/pulseq

System properties	**Value**
Maximum gradient	45 mT/m
Maximum slew rate	200 T/m/s
RF dead time	100 μs
RF ring-down time	100 μs
ADC dead time	10 μs
ADC raster time	100 ns
RF raster time	1 μs
Gradient raster time	10 μs
Block duration raster	10 μs
Gyromagnetic ratio (gamma)	42.576 MHz/T
Main magnetic field (B0)	1.5 T

Table 2: System properties used in Pulseq to define the Siemens system correctly. Input for mr.opts function.

After the system properties are correctly defined, it depends on the type of sequence how to proceed. For more standard sequences, it is useful to first define the desired number of sampling points and number of lines. Thereafter, the FOV, slice thickness, TE and TR can be defined. All these parameters can then be used to design the separate RF, gradient, and ADC events. Event design is done using the "mr.make..." functions, specific for each type of event. In these functions, it is a must to also input the previously defined system properties, so that the designed event will not exceed certain limits such as the maximum slew rate or raster time.

For gradients, the most simple variant is a trapezoid-shape, which needs either an area, flat area or amplitude as input. For RF events, different types can be chosen, where a gradient in the z-direction can also be outputted. Needed inputs are at least the flip angle, slice thickness, time-bandwidth product and duration. A sequence plot resulting from a TSE designed in Pulseq can be seen in Figure 8, with the resulting trajectory in k-space in Figure 9.

When designing a more unconventional sequence from a known sampling pattern where a demo sequence cannot be used as a starting point, it is best to extract gradient waveforms from the sampling pattern, and input these as separate arbitrary gradient events. To then obtain the desired number of sampling points per line, TE, and TR, calculations need to be done regarding the desired timing. For instance, the ADC read-out timing needs to be exactly tuned to the arbitrary gradient event timing by specifying delays in either events.

Important to check before using the self-made sequence *in vivo* is whether the sequence does not exceed the maximum gradient and slew rate, and if peripheral nerve stimulation (PNS), caused by rapidly changing gradients, is within the limits. In the most recent version of Pulseq, a test report can be generated summarizing the most important aspects of the sequence. Also, there are updated scripts for calculating the PNS, as well as an option to calculate acoustic resonance. For the PNS calculation, the SAFE-model is used [51, 52]. For this, certain hardware files are needed from the scanner. More information about the PNS calculation can be found in Appendix A.

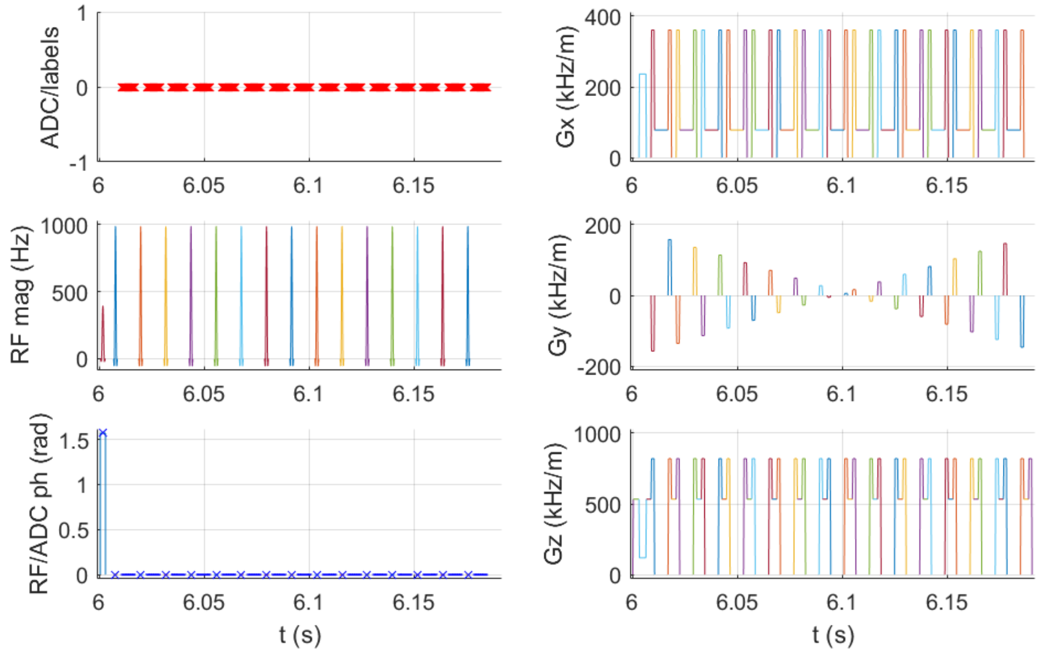


Figure 8: Sequence plot for fully sampled TSE sequence, zoomed in on the third echo train. In each echo train, 15 echos are acquired. In the right column the x-, y-, and z-gradients are plotted, where the simultaneous x- and y-gradient show to result in diagonal movement in k-space in Figure 9. Only when ADC is turned on, the gradients result in horizontal movement through k-space. In the left column, the ADC read-out, RF pulses and phase of ADC and RF are shown. The RF/ADC phase plot at the left bottom also shows the 90° pulse followed by a train of 180° pulses. Made in Pulseq.

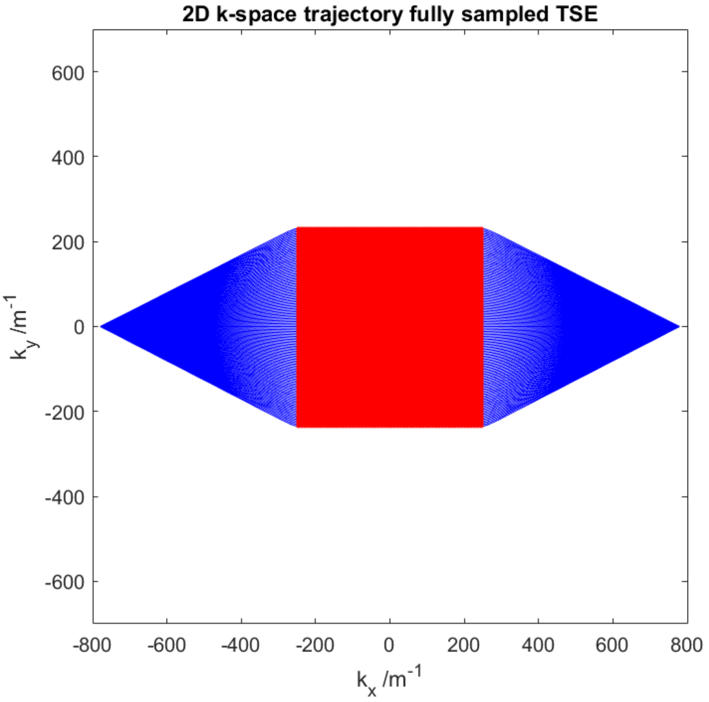


Figure 9: K-space trajectory for fully sampled TSE sequence. In red the sampling points (128 per line, 120 lines, in this plot not distinguishable), in blue the route taken. The k_{max} in this plot is seen to be about 250 m^{-1}, resulting in a FOV of around $\frac{1}{(500/128)} = 25$ cm according to Figure 3 and Equation (3). Made in Pulseq.

2 Subsampling

To accelerate MRI acquisition, k-space needs to be subsampled. Subsampled k-spaces can be acquired executing pulse sequences made in Pulseq. Since the sampling pattern used is not known to the scanner, only the raw data is provided as an output. Image reconstruction can be done using a NUFFT, but this will fail when the data is too sparse or irregularly sampled. Therefore, reconstruction can also be done with the BJORK model, which *should* be better at reconstructing irregularly sampled data. This led to the following subquestion:

To what extent can we subsample, and how does it impact reconstruction quality?

2.1 Method

To address this subquestion, a plastic imaging resolution phantom with liquid (see Figure 10) was scanned on a 1.5 T Siemens scanner (Aera, Siemens Healthineers, Erlangen), using a head coil (Head/Neck 20) of which 16 out of 20 coils were used to acquire data. Various custom-made Pulseq sequences were executed, both fully sampled and subsampled. Subsampling was done both retro- and prospectively.

A TSE and radial GRE were designed as fully sampled pulse sequences. Used scan parameters can be seen in Table 3. The fully sampled (according to [17] satisfying $n_{spokes} = \frac{\pi}{2}\ n_{points}$) radial GRE was also made into a golden angle version, where the angle between the spokes was 111.25°.

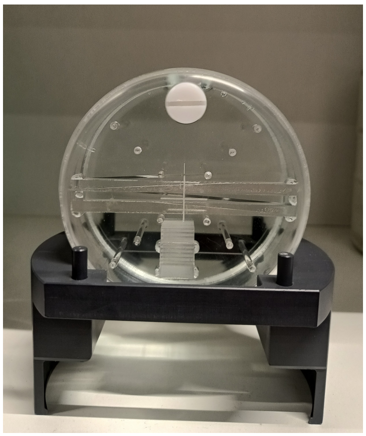


Figure 10: Photo of used resolution phantom in holder.

	Fully sampled TSE	**Radial GRE**
FOV	256 x 256 mm^2	256 x 256 mm^2
Slice thickness	3 mm	3 mm
Matrix size	128 x 120	128 x 201
Flip angle	90°/180°	20°
TR	2000 ms	20 ms
TE$_{(eff)}$	60 ms	8 ms
No. echos	15	-
ETL	8	-
No. spokes	-	201
No. dummies	1	10
Duration	18.00 s	4.22 s
Spatial resolution	2.02 x 2.13 mm^2	2.00 x 2.00 mm^2

Table 3: Set parameters for the fully sampled TSE and radial GRE sequence. ETL = echo train length.

Subsampling of the TSE was done using different techniques:

- (Only retrospectively) Mimicking a partial Fourier (PF) approach by removing 10 to 60% of the acquired k-space.
- (Only retrospectively) Removing 12 to 76 lines randomly (the number of lines removed corresponds roughly to the percentages used in the partial Fourier subsampling). See Figure 11 for an example of a resulting trajectory. For reproducibility, a random seed can be found in the respective code. Even though this will prospectively not result in time gain since not echo trains, but single echos are removed, it is still valuable to see how the BJORK model reconstructs this data compared to conventional Fourier based reconstruction methods.
- (Retro- and prospectively) Removing complete echo trains. A selection of two to four echo trains are removed. See Figure 12 for an example trajectory where train 2 and 8 were removed. Since the line in the center of k-space is acquired in the first echo train, it was chosen to remove trains corresponding to lines surrounding this echo to examine the full potential of the BJORK reconstruction model.

Subsampling of the radial GRE was done by removing lines in an alternating fashion. The number of spokes removed was 101, 151, and 191, so respectively approximately 50%, 75% and 95% of the total of 201 spokes.

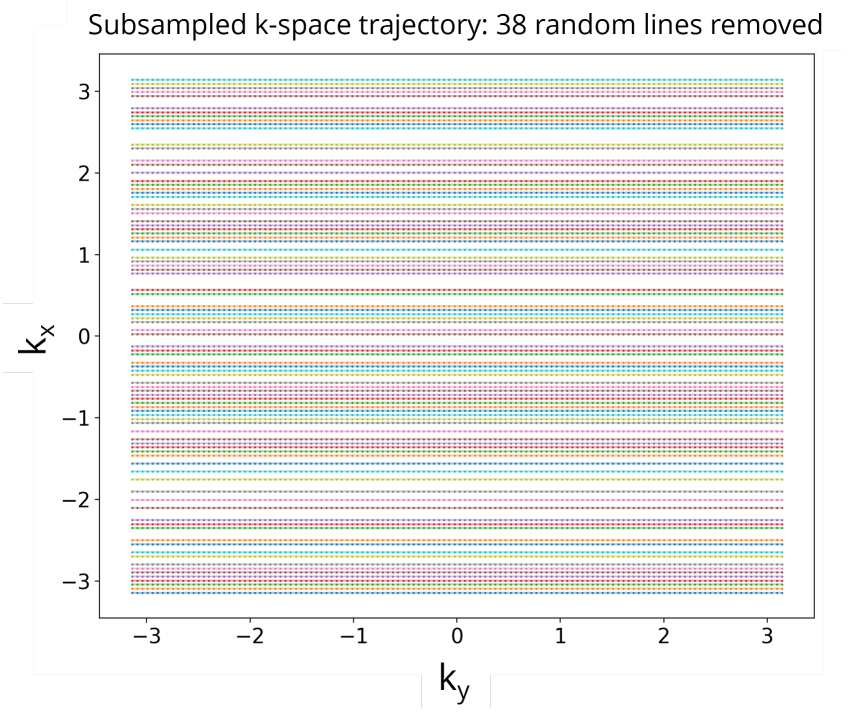


Figure 11: Retrospectively subsampled TSE k-space trajectory where 38 random lines have been removed. Note how random subsampling can result in big gaps between lines, both in the center as in the periphery.

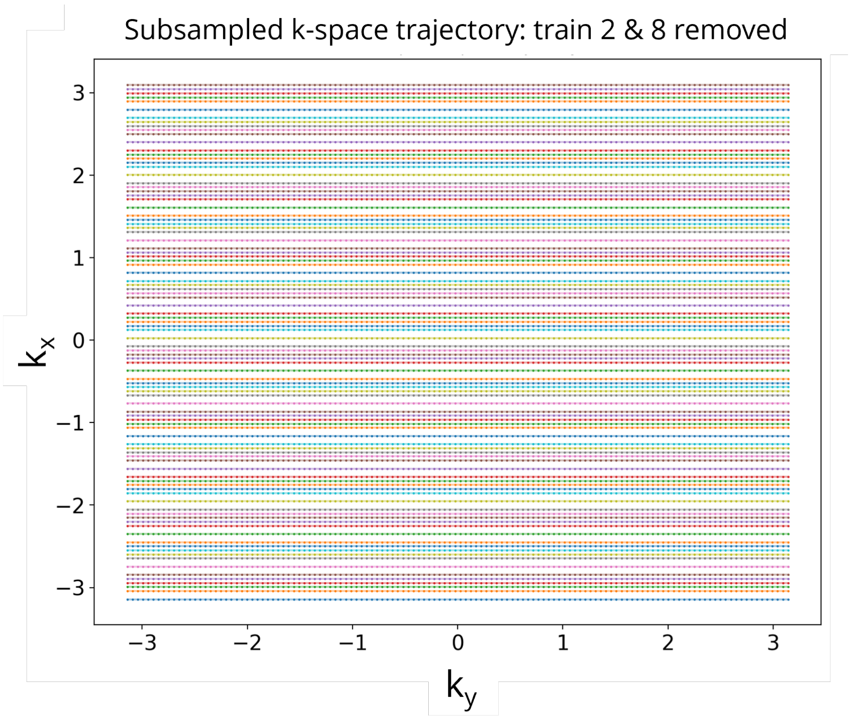


Figure 12: Subsampled TSE k-space trajectory where train 2 and 8 were removed. The trajectory starts at the bottom left. Note how echo train 1 contains the echo acquired at k = (0,0), contributing most to the contrast of the eventual image. The colors of the lines are not correlated with the echo or train number.

Reconstruction of raw data was done comparing a NUFFT with density correction and the BJORK reconstruction model. Besides the used trajectory and raw k-space data, also sensitivity maps were needed as input for the BJORK reconstruction model. Sensitivity maps were estimated using ESPIRiT [32], using a self-made Matlab script based on the Berkeley et al. ESPIRiT demo [53]. The set parameters and thresholds were determined by trial-and-error. For TSE sequences, sensitivity maps of the coils were computed on the fully sampled data using ESPIRiT with 24 calibration lines, kernel size 6, and thresholds 0.02 & 0.95. For radial GRE sequences, sensitivity maps were made using gridded k-space data, and thresholds 0.025 & 0.985. For both TSE and radial GRE, this was done in Matlab. The trajectory from the `.seq` file used was converted to the desired Python (`.npy`) format and normalized from $-\pi$ to π.

Since the originally implemented BJORK reconstruction model [46] was only functioning for an even number of lines in k-space, an odd number of lines resulting from subsampling caused errors. Therefore, an additional padding of 1 of the residual in the DIDN was done. The only change to the parameters of the model compared to the implementation of T. de Haan was changing the penalty in the reconstruction μ from 10 to 0.01. This was based on reconstruction results with different combinations of these parameters, see Appendix B. The change of this regularization parameter of the initial reconstruction was tested throughout reconstructions of data from various sequences, and with every dataset it showed improved reconstruction results when set to 0.01. Changes to the number of CG iterations, total number of iterations, and lambda were also tried, but did not improve the reconstruction significantly.

Results are evaluated using visual inspection and the SSIM if present.

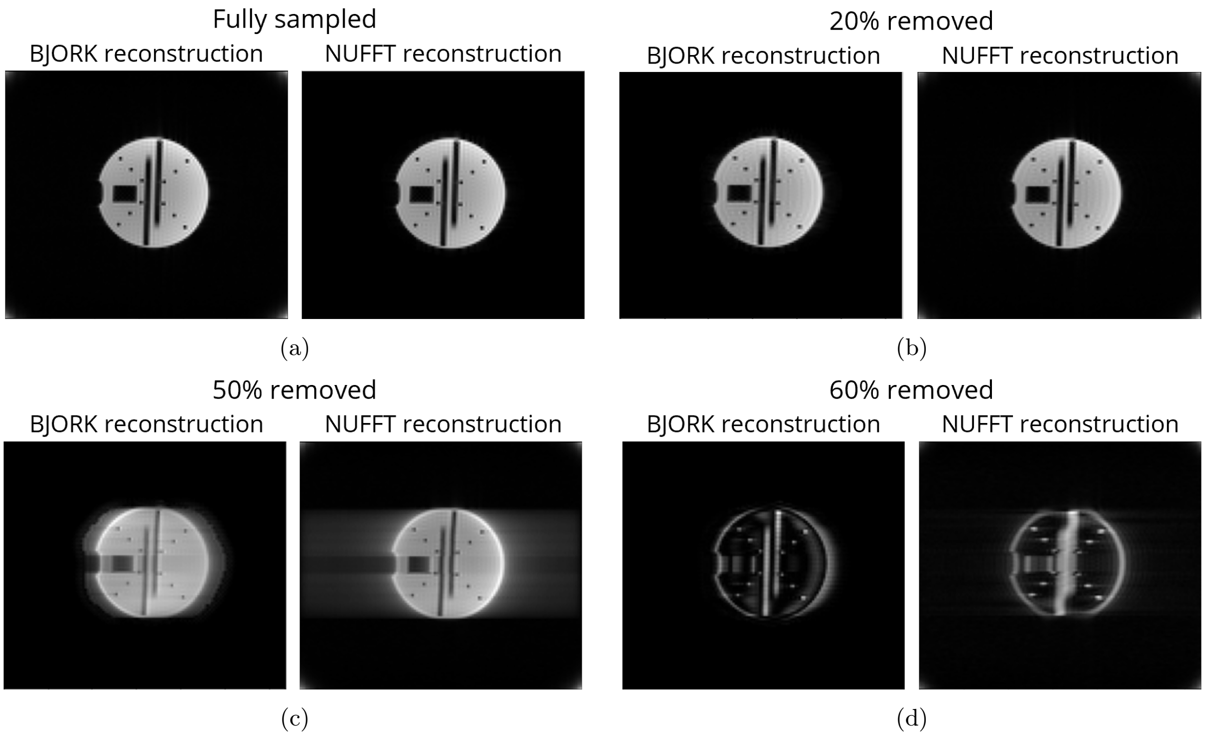
2.2 Results

2.2.1 TSE

Reconstructions of the fully sampled TSE can be seen in Figure 13a, where both the reconstruction with the BJORK reconstruction model and the NUFFT reconstruction are shown. There is no difference between the two reconstructions visible by eye. Reconstructions of the subsampled data will be discussed per type of subsampling.

Subsampling: Partial Fourier A selection of the reconstruction results can be seen in Figure 13. Slight ringing starts to appear on the side of the phantom in both reconstructions when 20% of the data is removed in Figure 13b. Artifacts become more prominent when removing more data. There is no obvious difference seen between the BJORK and NUFFT reconstruction up until Figure 13c, where a clear cropping effect of the model due to the use of the sensitivity maps is observed. At 60% in Figure 13d, meaning the complete center of k-space is not used for the reconstruction, only the sharp transitions in the image corresponding to the high spatial frequencies are visible. Here the BJORK reconstruction seems to be less deformed than the NUFFT reconstruction.

SSIM values relative to the fully sampled version of the respective reconstruction method in Figure 13a can be seen in Table 4. The SSIM of BJORK is seen to decrease less quickly than the SSIM of the NUFFT reconstructions, indicating the BJORK reconstructions are more robust to subsampling using a partial Fourier approach than the NUFFT reconstructions. The SSIM values also confirm the observations from the visual inspection, where no obvious difference was observed by eye between NUFFT and BJORK reconstruction until 50% of the data was removed.



(a) (b) (c) (d)

Figure 13: Selection of reconstruction results of retrospectively subsampled TSE data using a partial Fourier approach. From b) to d) an increasing portion of k-space is removed, where NUFFT reconstructions deteriorate faster than BJORK reconstructions. In Figure 13b, slight ringing within the entire phantom is observed in both reconstructions. In Figure 13c, clear smearing is seen in the NUFFT reconstruction, partly cropped off for the BJORK reconstruction. Figure 13d shows only contours of the phantom, with for the NUFFT reconstruction a more distorted shape.

PF %	SSIM BJORK	SSIM NUFFT
10%	0.9947	0.9946
20%	0.9831	0.9850
30%	0.9629	0.9565
40%	0.9244	0.9189
50%	0.8011	0.6362
60%	0.7222	0.5397

Table 4: SSIM values of PF subsampled reconstructions to fully sampled reconstructions. The SSIM is seen to decrease similarly for both reconstructions when subsampling, until 50% of the data is removed.

Subsampling: random A selection of the reconstruction results can be seen in Figure 14. When randomly removing lines of data, the BJORK reconstruction holds out much longer than the NUFFT reconstruction. When 12 random lines are removed, see Figure 14a, the NUFFT reconstruction already seems less homogeneous in contrast within the phantom compared to the BJORK reconstruction. Aliases start appearing in the NUFFT reconstruction when 20%, so 25 random lines of the data are removed, see Figure 14b. When about 40%, so 51 random lines are removed in Figure 14c, the BJORK reconstruction starts showing some artifacts within the phantom. The aliases in the NUFFT reconstruction become stronger in intensity when removing more lines. When about 60%, so 76 lines are removed in Figure 14d, the BJORK reconstruction starts producing aliases similar to those in the NUFFT reconstruction. The cropping effect of the model is clearly seen here, limiting the expression of the aliases present. Interesting to note, is that the aliases in the NUFFT reconstruction of 76 lines removed in Figure 14d, appear less well defined than the aliases present in the NUFFT reconstructions for 38 to 64 lines removed.

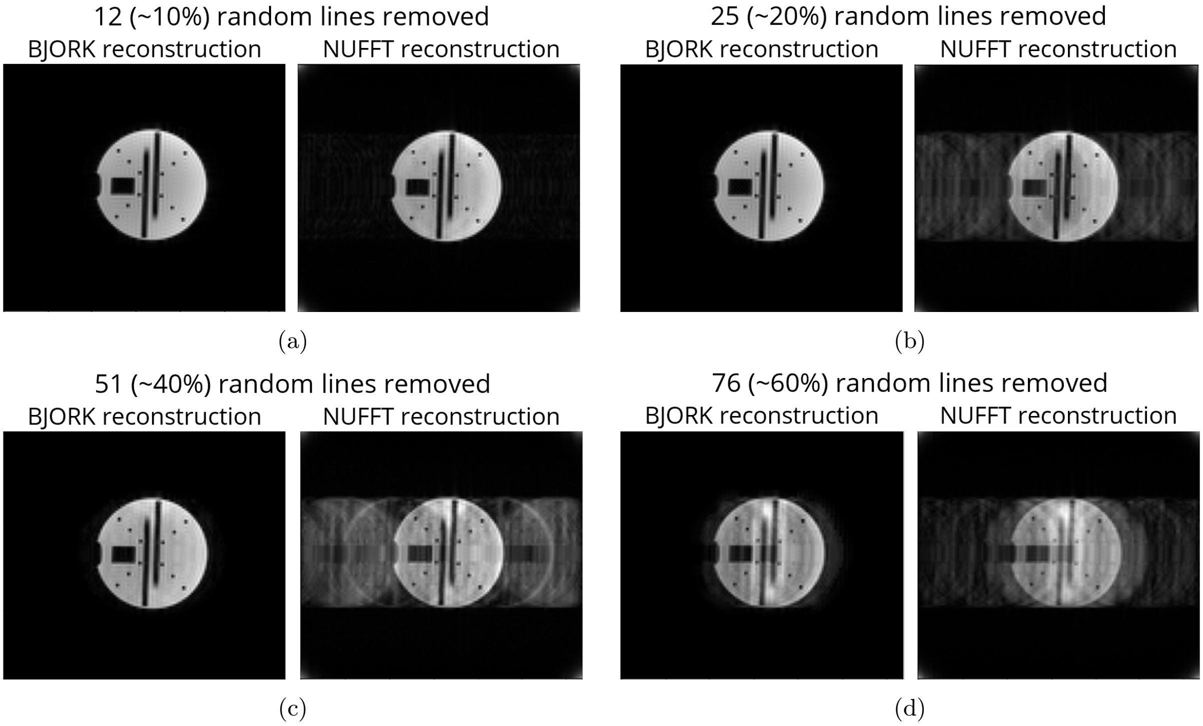


Figure 14: Selection of reconstruction results of retrospectively subsampled TSE data removing random lines. From a) to d) an increasing number of lines is removed. Inhomogeneity within the phantom is already seen for the NUFFT in Figure 14a. Aliases start appearing when subsampling, from Figure 14b onwards. The BJORK reconstruction starts showing artifacts within the phantom from Figure 14c onwards. In Figure 14d, the crop in the BJORK reconstruction seems to cut off aliases similar as present in the NUFFT reconstruction.

SSIM values compared to fully sampled TSE of the respective reconstruction method can be seen in Table 5. The BJORK SSIM stays much closer to 1 when subsampling more heavily than the NUFFT SSIM. Compared to the Partial Fourier results, the SSIM for BJORK is seen to decrease less quickly, indicating it is more robust to random subsampling than subsampling using a Partial Fourier approach. Also, the NUFFT performs much worse already at the start of random subsampling, compared to the start of the partial Fourier subsampling. This can be explained by looking at what data is removed, where in the PF subsampling this is data from the periphery in k-space, while for random subsampling, the removed lines are distributed randomly through k-space.

Random no. lines	SSIM BJORK	SSIM NUFFT
12	0.9996	0.8099
25	0.9982	0.6807
38	0.9620	0.6360
51	0.9683	0.6352
64	0.9560	0.6035
76	0.8720	0.6098

Table 5: SSIM values of randomly subsampled reconstructions to fully sampled reconstructions. For NUFFT reconstructions, SSIM decreases drastically when removing 25 lines, and stays relatively constant when subsampling more. BJORK SSIM is higher in all degrees of subsampling, and only significantly decreases when removing 76 random lines.

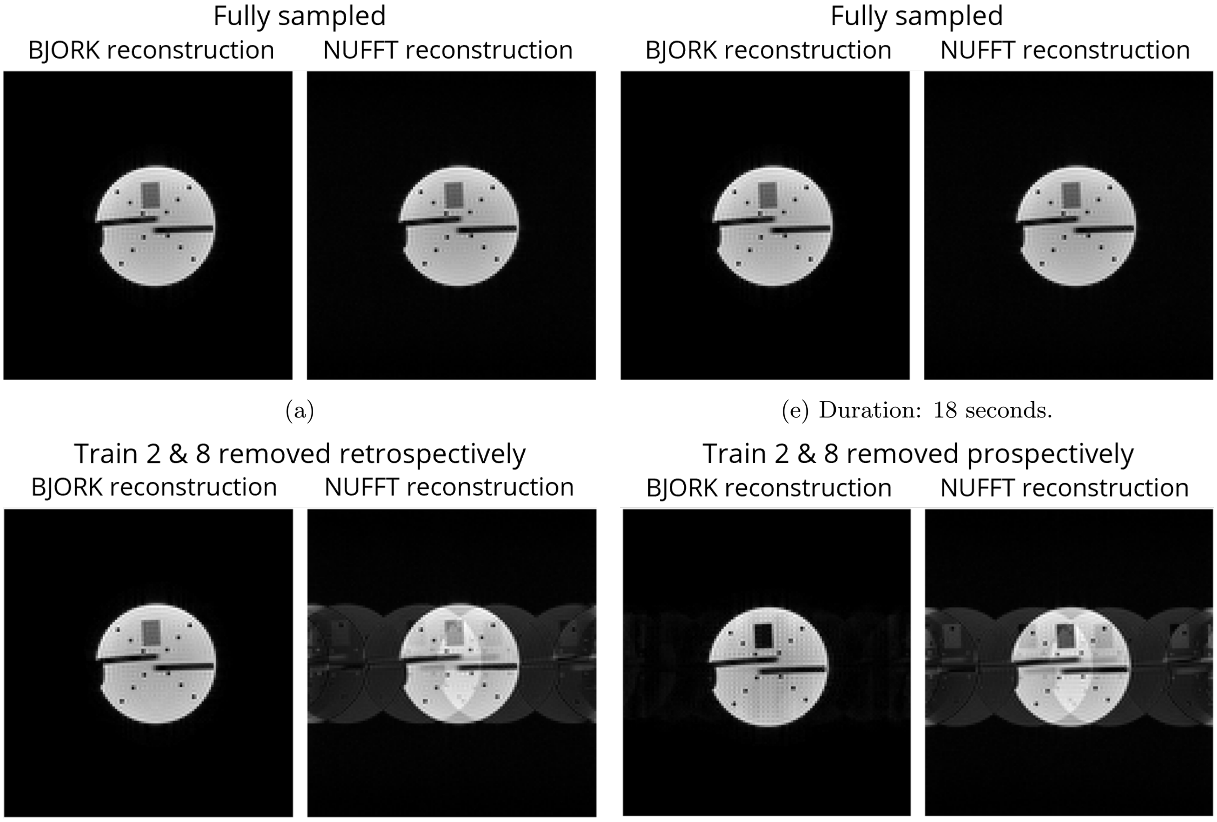
Subsampling: removing echo trains Echo train removal could be done both retro- and prospectively, enabling a comparison between the two approaches. The k-space trajectory where echo trains 2 and 8 were left out could already be seen as an example in Figure 12.

The reconstructions of the retrospectively subsampled data can be seen in Figure 15a to Figure 15d. Aliases already start appearing in the NUFFT reconstruction when the two echo trains corresponding to the echos surrounding the center echo are removed in Figure 15b. Interesting to see is that the aliases have quite distinct borders. In the respective BJORK reconstruction in Figure 15b, a slightly higher intensity is observed than in the fully sampled reconstruction in Figure 15a. When also removing the center line, slight artifacts appear within the phantom in the BJORK reconstruction, see Figure 15c. For the NUFFT, the same copies show as in Figure 15b, but with sharper edges of the copies. When removing train 2, 4, 6, and 8 in Figure 15d, the artifacts previously present within the phantom in the BJORK reconstruction disappear. For the NUFFT, very clear copies of similar intensity are observed.

The reconstructions of the prospectively subsampled data can be seen in Figure 15e to Figure 15h. At first glance these reconstructions seem very similar to the retrospectively subsampled reconstructions, but there are some differences to point out. First of all, the slice scanned for the prospectively subsampled scans deviated slightly from the slice scanned for the fully sampled TSE used for retrospective subsampling. This is why the square in the phantom is a different color, which could be the reason for the ghosts at the side of the NUFFT reonstruction of Figure 15f seeming slightly more clear than in Figure 15b. Also, the BJORK reconstruction seems to become more pixelated within the phantom. Vague aliases are clearly seen in the BJORK reconstruction in Figure 15g, while these were not yet visible in the BJORK reconstruction in Figure 15c. However, in Figure 15g, the object is more clearly distinguishable in the BJORK than in the NUFFT reconstruction. Also, copies of decreased intensity start appearing on the sides the BJORK reconstruction when removing half of the trains coherently in Figure 15h, which were also not present in the retrospectively subsampled version in Figure 15d.

Time gained per echo train prospectively removed was one TR, meaning that for four echo trains removed as in Figure 15h, the scan almost halved in duration, taking only ten seconds.

Unfortunately, since the fully sampled TSE is not of the same slice as the prospectively sampled ones, calculating the SSIM would not give a fair comparison between retro- and prospectively subsampled data.



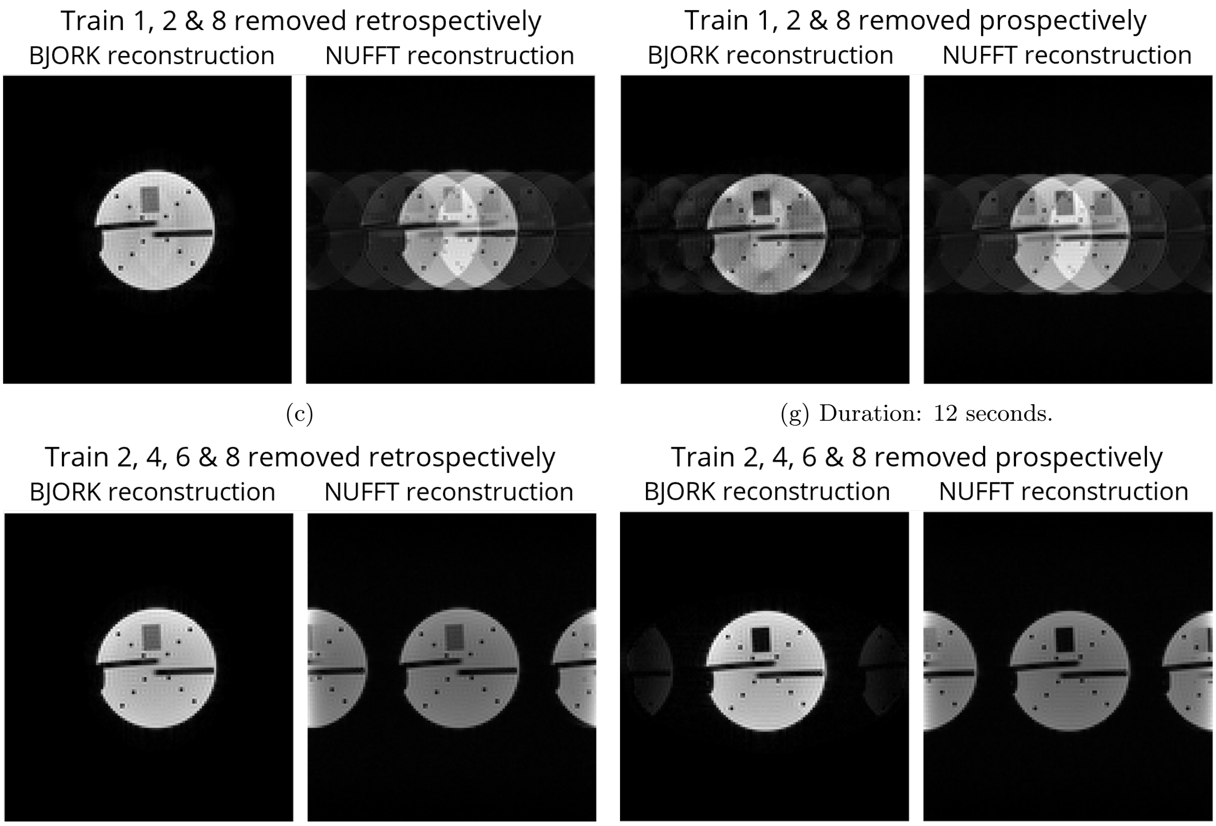
(a)

(e) Duration: 18 seconds.

(b)

(f) Duration: 14 seconds.

Figure 15: Figure continued on next page. Reconstructions results of retro- and prospectively subsampled TSE, removing entire echo trains. Retrospectively in Figure 15a to Figure 15d, prospectively in Figure 15e to Figure 15h. For prospectively subsampled scans, the duration is mentioned in the caption. Figure 15a and Figure 15e are the same; a different slice was scanned when prospectively subsampling. BJORK reconstructions outperform NUFFT reconstructions for both retro- and prospectively removed echo trains 2 and 8, but the BJORK reconstruction seems more pixelated in Figure 15f. The aliases present in the NUFFT reconstructions in Figure 15b and Figure 15f are similar.



(c)

(g) Duration: 12 seconds.

(d)

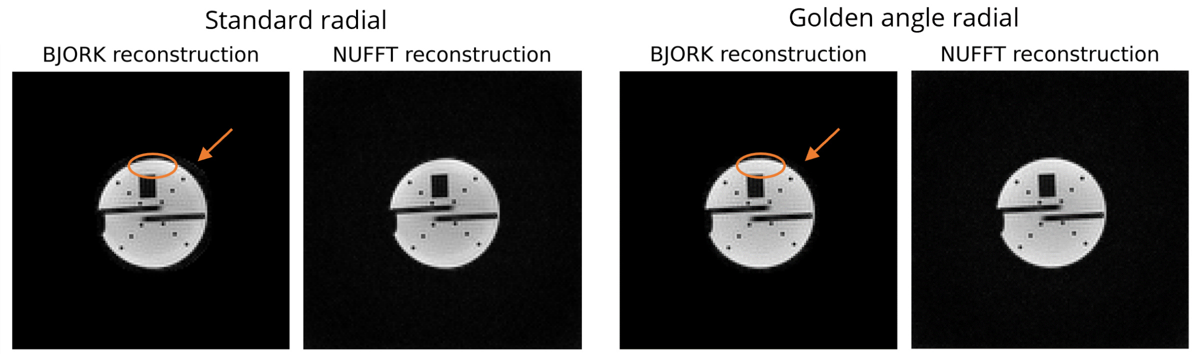
(h) Duration: 10 seconds.

Figure 15: Figure continued from previous page. When removing the echo train containing the middle line in k-space in Figure 15c and Figure 15g, the BJORK reconstruction starts showing artifacts as well. This is worse prospectively in Figure 15g, than retrospectively in Figure 15c. The NUFFT reconstructions are similar, and show aliases. When removing one more train, the NUFFT reconstructions in Figure 15d and Figure 15h show clear copies of the signal with decreased intensity, different from the ghost-like copies in the previous NUFFT reconstructions. In Figure 15h, the borders of the copies start appearing in the BJORK reconstruction, which is not the case for Figure 15d.

Scanning multiple slices was also possible using Pulseq. Since these were only done fully sampled to test and improve the versatility of the framework, the reconstructions are not of much interest here. Since the workflow needed to be altered slightly, explanation and the final BJORK reconstruction is still given in the appendix, in Appendix C.

2.2.2 Radial GRE

The reconstructions of the standard radial, meaning the spokes were acquired sequentially around the clock, can be seen in Figure 16a. The reconstructions of the golden angle version can be seen in Figure 16b. Both versions were reconstructed on a 128x128 matrix. No difference between BJORK and NUFFT reconstructions is visible by eye. Slight rings in the top of the phantom and a light circle around the phantom in the standard radial seem to have disappeared in the golden angle version. The true added value of golden angle compared to standard radial is best to be examined using data exposed to motion.



(a)

(b)

Figure 16: Reconstruction results of standard radial and golden angle radial data. Slight rings in Figure 16a, marked by the orange circle, have disappeared in the golden angle version in Figure 16b. Also a light circle around the phantom marked with an arrow, disappears when acquiring using a golden angle scheme.

Subsampling: removing alternating lines When subsampling the standard radial data retrospectively by removing alternating lines, results in Figure 17 show how much more robust radial data is against subsampling compared to the TSE data. In Figure 17a, 101 alternating lines were removed, meaning the reconstruction is made using only half of the spokes of the fully sampled version. The BJORK and NUFFT reconstruction seem to differ only slightly looking at the smoothness of the borders of the diagonal dots and horizontal bars, which seem more smooth in the NUFFT reconstruction. The color within the phantom seems to be more homogeneous in the BJORK reconstruction. Compared to the fully sampled data in Figure 16a, the inside of the phantom in the BJORK reconstruction is more grainy and shows a slight block-structure.

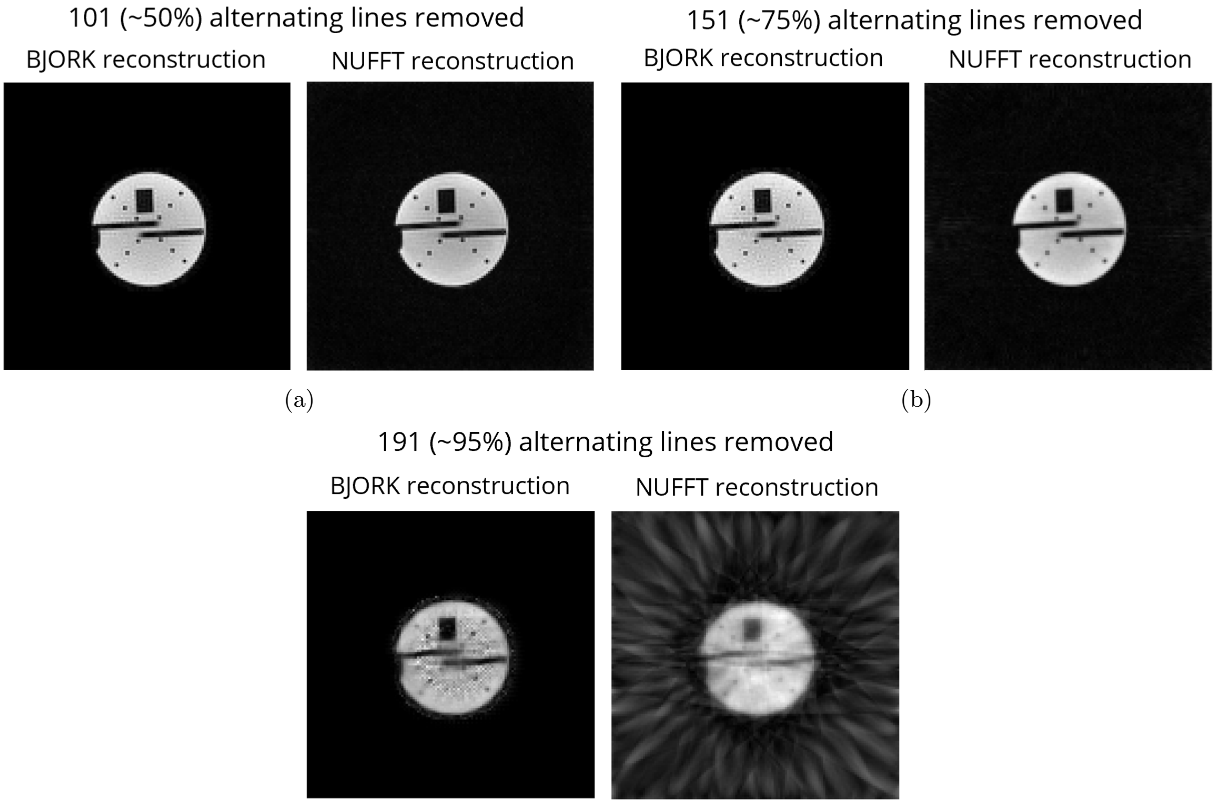
When removing an additional 50 lines in Figure 17b, so leaving only a quarter of the spokes, the NUFFT reconstruction becomes much more blurry than the BJORK reconstruction. The BJORK reconstruction does not seem to have deteriorated much further compared to removing only 101 spokes, except that the graininess has increased.

The reconstruction starts to show very prominent artifacts when removing an additional 40 lines, leaving only 10 spokes of data (5%) to form the image, see Figure 17c. In the NUFFT reconstruction, a classic star structure appears both in and outside of the phantom, which is recognisable from Figure 5. For the BJORK reconstruction, the graininess inside the phantom is amplified, leading to a block-like result with a visible cropping border around the phantom. However, the edges of the structures, especially the horizontal bars, inside of the phantom are still better defined than the details in the NUFFT reconstruction.

SSIM values in Table 6 confirm that BJORK reconstructions of subsampled data are not only visually more similar, but also quantitatively more similar to the fully sampled version. Especially the large difference between BJORK and NUFFT is confirmed for the most heavily subsampled data.

Alternating no.lines	SSIM BJORK	SSIM NUFFT
101	0.9977	0.9144
151	0.9934	0.8271
191	0.9250	0.2416

Table 6: SSIM values for alternating subsampled standard radial GRE, relative to the respective fully sampled reconstruction. The SSIM for BJORK reconstructions stay much closer to 1 than the SSIM for the NUFFT reconstructions.



(a)

(b)

(c)

Figure 17: Reconstructions results of retrospectively subsampled standard radial data by removing alternating lines. In Figure 17a, the BJORK and NUFFT reconstruction only differ looking at the smoothenss of the borders of the dots and bars, where the NUFFT reconstruction is more smooth. Compared to Figure 16b, the BJORK reconstruction is more grainy within the phantom. In Figure 17b, the NUFFT reconstruction becomes blurry, while the BJORK reconstruction only becomes a bit more grainy. When leaving only 5% of the data in Figure 17c, the NUFFT reconstruction exhibits the classic star structure, while in the BJORK reconstruction, the graininess has increased further, but details are still distinguishable.

2.3 Discussion

All sequences were executed successfully. Since they were designed based on relatively basic Pulseq demos, there is much room for more unconventional sequence design choices. Examining the spacing between lines *and* points in k-space, and examining the order of echos within an echo train could be interesting, since the present framework is versatile enough for such adjustments. Also adding pre-emphasis in the gradients would be a good addition to limit eddy currents, even though there were no clear eddy current-related artifacts visible in this data. Also implementing a 3D radial GRE could be of interest. Then, actually feasible random subsampling schemes (in one direction) could be acquired.

For every reconstruction, evaluation was done by visual inspection, and only sometimes using the SSIM. For further research, the fully sampled reconstruction of the same slice should be present and suitable for calculating the SSIM in all of the subsampling scenarios. These measures were appropriate in this stage of research, but since the diagnostic quality of the images is not addressed, *in vivo* measurements and assessment of radiologists would be an appropriate future step as well.

Subsampling: partial Fourier When removing a relatively small part of the periphery of k-space as in Figure 13b, hardly any differences are observed compared to the fully sampled reconstructions. This was seen both in the SSIM as in the figures. The slight ringing (truncation, Gibbs) appearing when subsampling is probably caused by the sharp cut-off between sampled and removed data in k-space. This artifact can be reduced by using a more smooth transition at the cut-off border [54]. The NUFFT and BJORK reconstructions were very similar up to removal of 50% of the data, as the SSIM confirmed in Table 4. After that, BJORK reconstructions seemed slightly better, but that effect is most likely due to the use of sensitivity maps, confirmed by the clear cropping seen in Figure 13c. The reconstructions for Figure 13c and Figure 13d clearly follow from the location of the data removed in k-space: for Figure 13c a part of the center is removed resulting in smearing, and for Figure 13d only the periphery of k-space is left resulting in only edges of the object remaining.

Subsampling: random As expected, the NUFFT could not handle randomly subsampled data well, since the data was too irregularly or sparsely sampled. Since the subsampling is incoherent, the object was still clearly distinguishable from the aliases. For Figure 14d, the object and aliases became less distinguishable, thus possibly indicating more coherent subsampling. The SSIM in Table 5 also showed the NUFFT reconstruction of 76 lines removed was *slightly* more similar to the fully sampled reconstruction than when removing 64 lines.

The BJORK reconstruction model held out much longer than the NUFFT for random subsampling, as is also indicated in Table 5. The BJORK reconstructions started showing artifacts within the phantom when roughly 40% of the lines were removed randomly, see Figure 14c. This is not confirmed by the SSIM, where 40% removal even has a slightly higher SSIM than 30% removal. This once again confirms the SSIM should always be used next to visual inspection of the reconstructions to truly examine the quality of a reconstruction.

Subsampling: removing echo trains Unfortunately, the SSIM could not be used as a fair measure of comparison between the subsampled and fully sampled NUFFT and BJORK reconstructions, but still the reconstructions give insights into to what extent we can subsample and how it impacts reconstruction quality.

For retrospective subsampling, the BJORK reconstructions showed hardly any artifacts. Only when removing the train corresponding to the center line as well, as was done for Figure 15c, slight copies were seen within the phantom. However, the BJORK reconstruction was still much better than the NUFFT. The coherent alisaing with similar intensity in Figure 15d is as expected for the coherent subsampling scheme used [22].

For prospectively subsampled data actually resulting in acceleration of acquisition, aliases were prevalent in the BJORK reconstructions in both Figure 15g and Figure 15h. This difference with retrospectively subsampled data is definitely in part caused by the used sensitivity maps. In retrospectively subsampled data, sensitivity maps based on the fully sampled dataset were used. This was done to show an appropriate comparison with literature, where retrospective subsampling is often combined with fully sampled sensitivity maps. An especially clear result of the use of fully sampled sensitivity maps is a crop around the phantom seen in the BJORK reconstruction in Figure 15c, cropping away any artifacts in the background. In the prospectively subsampled data, sensitivity maps were made of that same prospectively subsampled data, truly resulting in acceleration of acquisition. In Figure 15g, the phantom in the BJORK reconstruction is better distinguishable from the background than in the NUFFT, indicating even sensitivity maps made from subsampled data can result in better reconstructions compared to the NUFFT. For Figure 15h however, this meant that slight aliasing was already present in the sensitivity maps, resulting in them also being present in the BJORK reconstruction. Also the pixelation seen in Figure 15f is a sign of imperfections in the sensitivity maps, since this effect was also observed when using erroneous sensitivity maps earlier in this research.

Radial GRE There was no clear difference observed between the standard and golden angle radial, but since golden angle is expected to be more robust against motion artifacts, it is advised to test this sequence in a setting where motion is induced in the object. When subsampling the radial GRE sequence by removing alternating lines retrospectively, it was clearly seen that the radial sequence is more robust against subsampling than the TSE sequence. This is due to the center of k-space still being sampled by multiple lines, while mostly data on the periphery of k-space is lost. This effect is clearly seen in the NUFFT reconstruction for Figure 17b, where the borders of the phantom and structures within the phantom have become less defined (blurry). The artifacts visible in the NUFFT reconstruction when only

5% of the spokes are used, see Figure 17c, are recognizable from Figure 5. The BJORK reconstruction here shows clear graininess within the phantom, but still most of the fine details such as the dots in the phantom were distinguishable.

For the radial GRE, the BJORK reconstruction was significantly better up to a very high degree of subsampling. The SSIM in Table 6 also confirmed that the BJORK reconstructions for subsampled radial GRE data were very good, especially when comparing the degree of subsampling with the SSIM mentioned for subsampled TSE data: when using only 5% of radial GRE data and reconstructing with BJORK, the SSIM is higher than when 40% of TSE data is reconstructed with BJORK.

2.4 Conclusion

All TSE and radial GRE sequences were designed and executed successfully. For fully sampled data, the BJORK and NUFFT reconstruction were almost identical. The BJORK reconstruction showed to be more robust to subsampled data than a conventional NUFFT when subsampling heavily (f.e. from Figure 13c and Figure 14b onwards, as well as in Figure 17c). In general removing 50% of TSE data randomly or using the partial Fourier approach was the moment the BJORK reconstruction showed clear artifacts within and outside of the phantom. For removing echo trains, the impact of subsampling depends on whether the center echo is also removed, as expected. Removing half of the echo trains resulted in only slight aliases in the BJORK reconstruction, while accelerating acquisition almost 2x.

That BJORK reconstructions were more robust to subsampling than the NUFFT is not only due to the use of sensitivity maps, but also due to the CG-method used to estimate the initial reconstruction. The sensitivity maps do have great influence on the reconstruction, as was observed when prospectively subsampling TSE and using sensitivity maps computed from that subsampled data, see Figure 15h. There, certain artifacts were also present in the sensitivity maps, leading to them being present in the final reconstructions. The difference between retro- and prospectively subsampled TSE reconstructions once again highlights the importance of using truly acquired subsampled k-space data instead of simulated data. This does not only confirm the hypothesized time gain investigated, but also shows the true impact subsampling has on the reconstruction result.

Now that the versatility of BJORK reconstruction model has been investigated, and this framework has been established for custom-made sampling patterns, optimized sampling patterns learned by the BJORK model can be implemented.

3 Optimized acquisition

Now that the BJORK reconstruction model showed quite robust against subsampling with handcrafted subsampling patterns, we can try subsampling by only acquiring the points in k-space most important for reconstructing a good image. Determining which points in k-space are important is difficult to do from scratch, which is why the deep learning model BJORK is used to learn what points in k-space should be acquired. This lead to the following subquestion:

What is the feasibility of a BJORK-optimized sampling pattern, and how does it affect reconstruction quality?

3.1 Method

To address the subquestion, the same imaging resolution phantom as in Section 2 was scanned in the same set-up, as well as the head of a volunteer. All 20 coils in the head coil were used to acquire data. The trained BJORK reconstruction model accompanying the optimized sampling pattern used, was used to reconstruct acquired data. The overall workflow can be seen in Figure 18.

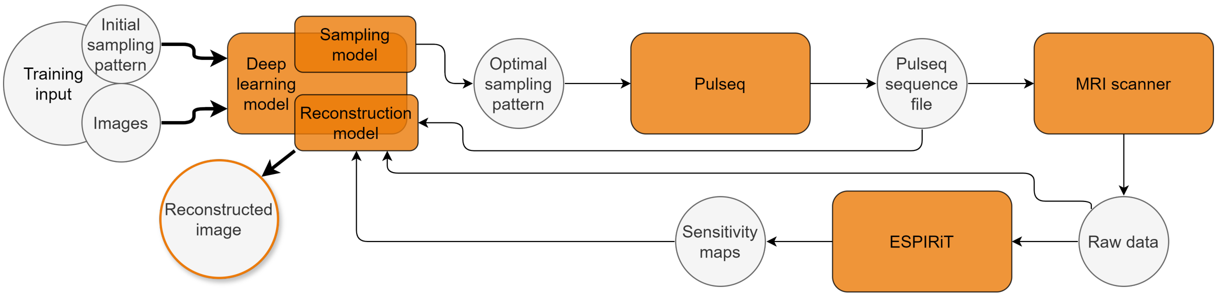


Figure 18: Used workflow for implementing the optimized BJORK pattern in the scanner and reconstructing the raw data. ESPIRiT, and Pulseq are used in Matlab, the BJORK deep learning model in Python.

This method section is divided into designing the optimized sequence, executing the optimized sequence and then reconstructing the data.

3.1.1 Sequence design

One of the optimized sampling patterns learned by the BJORK model implemented by [46] in PyTorch was taken to design the optimized sequence, see Figure 19. The choice for which optimal sampling pattern was based on the desire to not have spokes cross themselves, and the desire to have the spokes roughly cross the center of k-space. The trajectory was learned in [46] with the aim of resembling results from authors of the BJORK model [4], however implementation of the model by [46] deviated slightly for the set training parameters, see Table 11 in Appendix D.1 for an overview. The most important training parameters that were different from the current implementation of [46] were the maximum gradient and slew rate, and the FOV. The FOV deviated since the training data used in [46] had a FOV of 140 x 140 mm^2, instead of 220 x 220 mm^2.

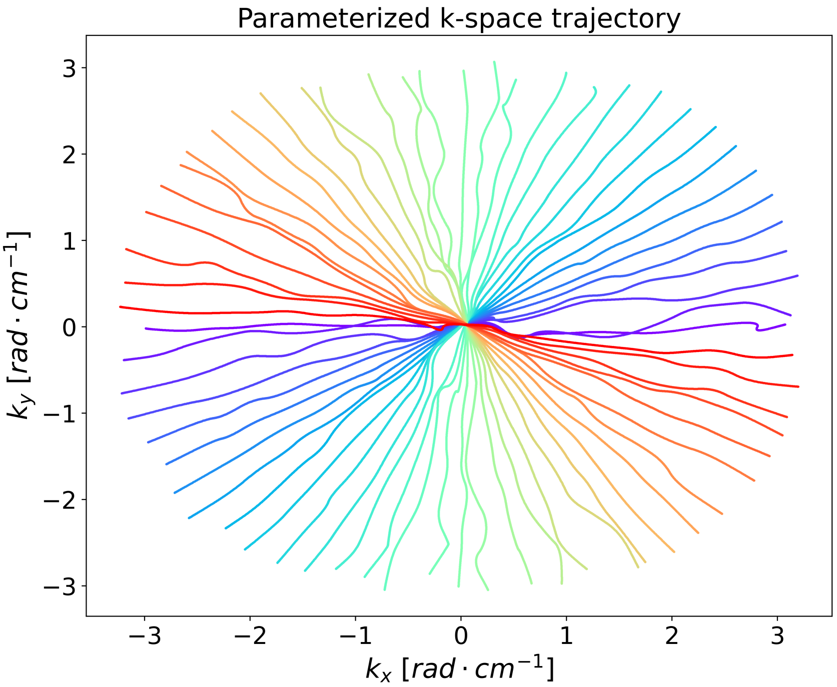


Figure 19: Optimized sampling pattern from trained BJORK network from [46]. The spokes are acquired in an anti-clockwise manner, starting from roughly the point $(-\pi, 0)$.

Solely a k-space sampling pattern as in Figure 19, does not make a pulse sequence for an MRI scanner yet. First, the sampling pattern needs to be converted to x- and y-gradients, before it can function as input for Pulseq. Then, the z-gradients, RF pulses, and ADC read-out need to be designed, including appropriate timings. The sequence parameters the original authors of the BJORK model [4] reached *in vivo* can be seen in Table 7. These were reached in a GE MR750 3.0 T scanner with a Nova Medical 32 channel Rx head coil. These parameters were used as an aim in designing the sequence in this thesis.

	Sequence parameters used in original [4]
Slice thickness	2 mm
Flip angle	90°
TR	318.4 ms
TE	3.56
Sampling time	4 μs
Duration	11 s

Table 7: Set sequence parameters for *in vivo* sequence in [4].

Since the derivative of the trajectory provides the gradient, gradients were retrieved rather easily. However, one issue was the parameterization of the trajectory. As mentioned in Section 1.4.2, each spoke of the trajectory was controlled by 40 b-spline kernels, meaning the derivative of such a spoke is defined over only 39 points, as can be seen in Figure 20. When using these gradient waveforms in Pulseq, the maximum slew rate was exceeded due to the large changes in gradient amplitude over a short distance in k-space. Therefore, it was chosen to interpolate the gradient waveforms using quadratic b-spline kernels, to a number of points defining the gradient resulting in a waveform not exceeding the maximum slew rate. For an example of such an interpolated gradient waveform for one spoke, see Figure 36a and Figure 36b in Appendix D.2.

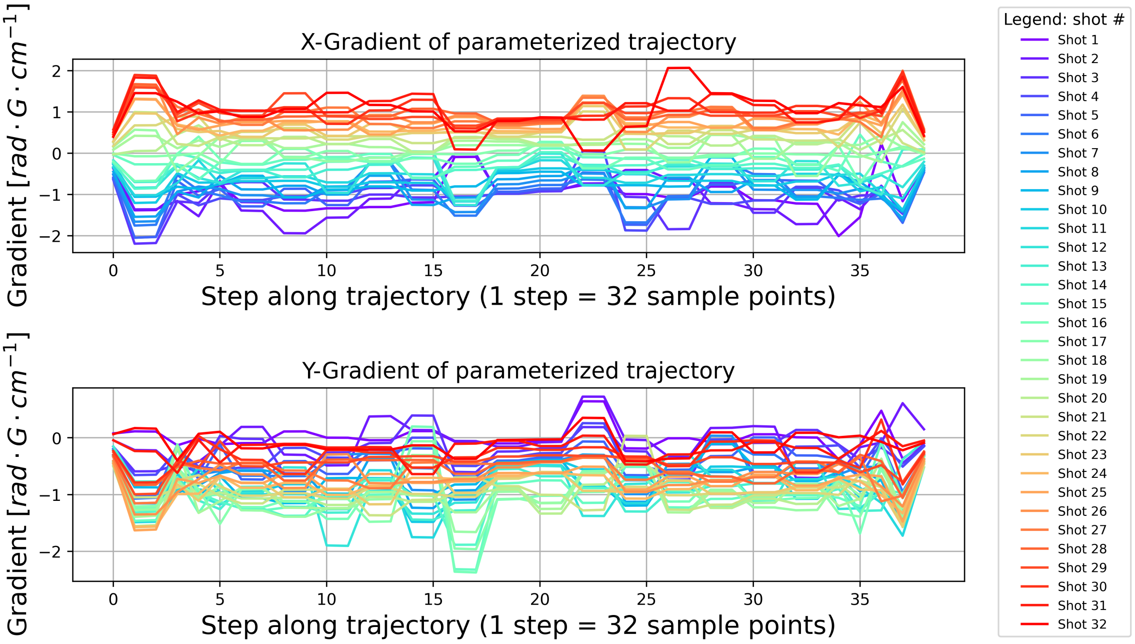


Figure 20: The x- and y-gradient waveforms obtained from optimized sampling pattern in Figure 19 for each spoke (shot). Since the sampling pattern uses a decimation rate of 32, the gradient is defined for 39 points. This means each step here defines 32 sample points. Note how none of the gradient waveforms start and end at zero.

This gradient waveform definition gives rise to a few issues. To achieve a TE somewhat close to 3.56 ms from Table 7, the original sampling time from [4] needs to be used: 4 μs per sampling point. Then, the gradient waveform for one spoke would take $4\,\mu s \times 1280$ points $= 5.1$ ms. However, the shortest duration achievable without exceeding the maximum slew rate, was 4.9 ms when using 100% of the maximum gradient and slew rate. When incorporating the fixed Siemens & Pulseq gradient raster time of 10 μs, this resulted in the gradient waveforms needing to be defined by at least 486 or 512 points depending on the set maximum slew rate. Since the Siemens & Pulseq ADC raster time is 100 ns, still 1280 points could be sampled, but this resulted in other problems:

- Since the gradient waveform was now shorter in time than when defining it over 1280 points, the FOV of 22 x 22 cm^2 could not be reached due to k_{max} being smaller. The FOV reached with gradient waveforms defined with 486 points was around 55 cm.
- When letting go of the TE target of 3.56 ms, and letting the gradient waveform be defined over 1280 points, each spoke takes at least $1280 \times 10\,\mu s = 12.8$ ms. This does make it possible again to reach a FOV similar to the FOV in [4], namely approximately 21 cm, but this will result in a different contrast.
- When taking the gradient waveform defined over 486 points, the duration was minimally 4.9 ms. When then setting the TE to 3.56 ms and neglecting any difference in time between the start of acquisition and the start of the gradient waveform, 1.01 ms remains in between the middle of the RF pulse and the start of the gradient waveform. Therefore, the RF pulse could not be longer than 1 ms, leading to the z-gradient also being not longer than 1 ms. When leaving the slice thickness and bandwidth the same, this lead to exceeding the slew rate again.

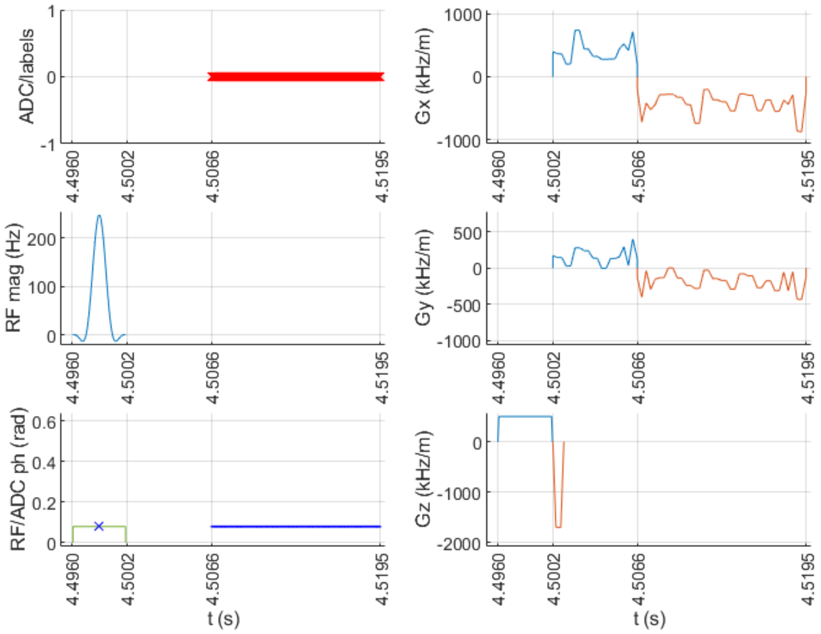
Other sequence elements that needed to be altered or added are listed below. For each element added, the system definition is set as one of the inputs, making sure the resulting element is constrained by the maximum gradient, maximum slew rate and appropriate Siemens raster times.

- Due to the raw gradient waveforms from the model not starting and ending at a value of zero as was seen in Figure 20, this was enforced on each waveform in Pulseq, again resulting in a small increase in the duration of each waveform.

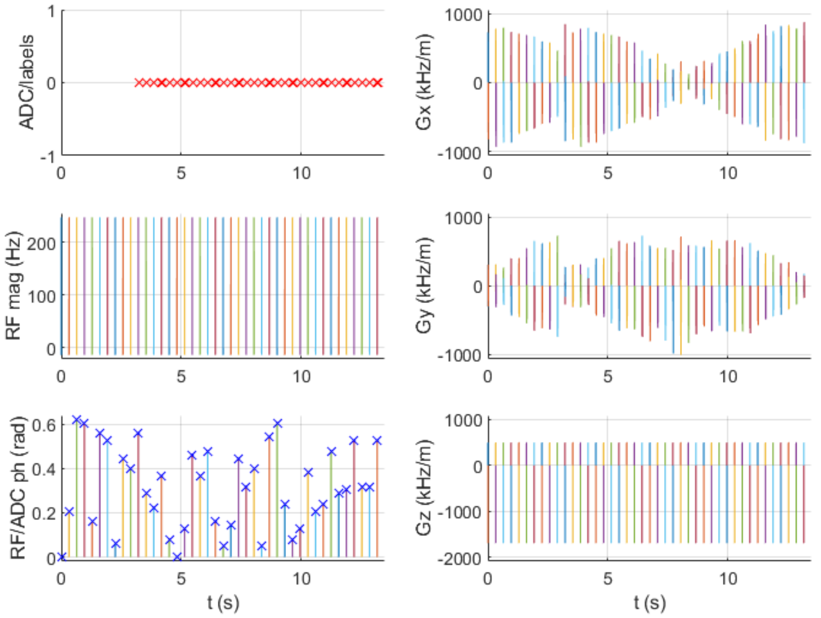
- The RF pulse designed was a sinc pulse using the mr.makeSincPulse function, with flip angle 90° and duration 4 ms. The apodization was set to 0.5, and the time-bandwidth product 4, resulting in a bandwidth of 1 kHz. The RF pulse duration led to a shortest possible TE of around 7 ms.
- Since the z-gradients were not included in the model output, these needed to be designed separately in Pulseq. These were defined using the same mr.makeSincPulse function as used for the RF pulse design, meaning a trapezoid z-gradient was made with the same duration as the RF pulse. The amplitude was calculated using the bandwidth of the RF pulse and the slice thickness, which was set to 2 mm. Using the same function, also a rephasing z-gradient was created to eliminate the phase dispersion caused by the slice selection z-gradient.
- An ADC event containing 1280 sampling moments was made with the same duration of the readout x-gradient. Since each ADC sampling point needs to be placed in the middle of the gradient raster, a dwell time was added automatically. To ensure this fits into the gradient waveform, the number of points was adjusted if necessary, which only occurred for some combinations of number of sampling points and number of points used to define the gradient.
- The x- and y-gradients given by the model are only defining the position of the sampling points in k-space, but not the route from the k-space center to the start of the spoke. To reach the starting point for each spoke, the appropriate x- and y-gradient was inversed and halved, and placed before the original gradient.
- Delays were incorporated with in mind the target TE and TR. When scanning multi-slice, the TR target was let go and made to fit the desired number of slices.
- Dummy cycles were added at the start of the sequence: extra TRs using all components except the ADC sample acquisition. This was done to drive the magnetization to a steady state before starting acquisition.
- RF spoiling was also incorporated, since this is more robust than gradient spoiling [55]. Spoiling was done to disturb the coherence of the spins that can develop in the M_{xy}-plane after a few cycles. A phase increment of 117 degrees was used [56, 55].

The main reason for these extensive adjustments in the sequence and in the timing, is that [4] had access to using a gradient raster time of 4 μs, but also that there was no information on the actual *design* of the sequence given in the original article. In general these design choices mean there is less resemblance between the designed sampling pattern and the originally optimized sampling pattern, but the sequence is still useful to answer the subquestion.

The resulting pulse sequence, using gradient waveforms defined with 1280 points, can be seen in Figure 21.



(a) BJORK sequence plot zoomed into the components for a spoke acquired around 4.5 seconds in the sequence. The separate sampling points in the ADC readout are not distinguishable from each other in this plot.



(b) Complete BJORK sequence. Here the RF spoiling can be seen in the graph at the left bottom. The 10 dummy scans can be seen in the left top, where no ADC acquisition is turned on for a certain period of time, while the gradients follow their usual regime.

Figure 21: BJORK sequence plots, where Figure 21a is a zoomed-in version of Figure 21b. In the right column the x-, y- and z-gradients over time. In the left column the ADC read-out, RF pulses and RF/ADC phase over time.

3.1.2 Executing the sequence

Multiple variants of the BJORK sequence were made into .seq files in Pulseq, executable by the scanner. In Table 8, an overview of the basis sequences made can be seen. In all sequences, an RF flip angle of 90° was used. Since a delay was added to get the TR close to the TR mentioned in the article, all TR's were similar for the single slice sequences: about 321 ms. The maximum gradient and slew rate determined the minimal number of points needed to define the gradient waveforms. The maxima were set to either 90% of the actual maximum, resulting in a PNS of 91%, or to 100% of the actual maximum, resulting in a PNS of 101%. Parameters that were varied outside this table were the number of dummy scans (0, 10 or 20), and whether RF spoiling was turned on.

	Sequence type		
Set parameters	**1**	**2**	**3**
max %	100%	90%	90% & 100%
n_g	486	540	1280
n_{adc}	1215	1280	1280
Resulting parameters			
Δsl	0.0019 m	0.0019 m & 0.20 m	0.0019 m
FOV	520 x 528 mm^2	520 x 528 mm^2	208 x 211 mm^2
TE	6.96 ms	7.50 ms	14.8 ms
TR	321.15 ms	321.15 ms	321.15 ms
dur	13.49 s	13.49 s	13.49 s
res	0.43 x 0.43 mm	0.39 x 0.39 mm	0.16 x 0.16 mm

Table 8: Some of the set parameters and resulting (calculated) parameters for single-slice BJORK sequences. Max %: what percentage of the actual maximum gradient and slew rate is set as a maximum value, n_g: number of points used to define each gradient waveform, n_{adc}: number of sampling points acquired per spoke, Δsl: slice thickness, FOV: $FOV_x \times FOV_y$, dur: total duration, res: spatial resolution x and y.

Sequence type 1 In sequence 1, the number of points defining the gradient waveforms was the lowest, resulting in the lowest TE. The FOV was the biggest, resulting in a worse resolution than in sequence 2 and 3. Due to difficulties with ADC placement needing to be in the middle of the gradient raster, only 1215 points could be sampled along each spoke instead of 1280, but this small deviation is not likely to influence reconstruction results much. Only one sequence from this type was executed, where 10 dummies were used, and RF spoiling was on.

Sequence type 2 For sequence 2, the number of points defining the gradient waveforms was slightly increased compared to sequence 1, resulting in a slightly higher TE. Spatial resolution and FOV were comparable to sequence 1. Two sequences of this type were executed, both RF spoiled, with 10 dummies. Sequence 2A used a slice thickness of 2 mm, sequence 2B a slice thickness of 200 mm.

Sequence type 3 Due to the largest number of points defining the gradient waveforms and therefore the goal k_{max} being reached, sequence type 3 resulted in the FOV closest to the FOV used by the original authors of BJORK [4] (see Table 11 in Appendix D.1). Although this accompanied a longer TE, this sequence was chosen to examine varying more parameters. All sequences of type 3 used 10 dummies, RF spoiling and a slice thickness of 2 mm. Five sequences of this type were executed, spread over two days (A & B on day 1, C to E on day 2). Therefore, the phantom placement was not exactly the same. See Table 9 for an overview of the parameters varied per sequence. For the multi-slice acquisitions, an interleaved scheme was examined to limit cross-talk between slices [57].

Sequence	Varied parameters
3A	single-slice, 90% of maximum gradient and slew rate
3B	single-slice, golden angle version of A
3C	single-slice, 100% of maximum gradient and slew rate
3D	3A multi-slice, consecutive acquisition
3E	3A multi-slice, interleaved acquisition

Table 9: Varied parameters for sequence 3. For E, interleaved acquisition means the slice acquisition order was 1, 3, 5, 2, 4.

***In vivo* measurements** Due to the large FOV and subsequent lower SNR in sequences 1 and 2 from Table 8, sequence 3 was chosen to proceed with for the *in vivo* measurements. To limit chances of PNS occurring, sequence 3A was chosen as a basis to vary multiple parameters, see Table 10 for an overview. Data was acquired in the coronal, transversal and sagittal plane.

Sequence	Varied parameters
3A-RF$_0$	single-slice, RF spoiling turned off
3A-RF$_1$	single-slice, RF spoiling turned on
3A-dum$_0$	single-slice, RF spoiling on, 0 dummies
3A-dum$_{10}$	single-slice, RF spoiling on, 10 dummies
3A-dum$_{20}$	single-slice, RF spoiling on, 20 dummies
3D	multi-slice 3A-RF$_0$ with 10 dummies, consecutive acquisition
3E	multi-slice 3A-RF$_0$ with 10 dummies, interleaved acquisition
3F	golden angle version of 3E

Table 10: Varied parameters for *in vivo* measurements with as basis sequence 3A.

3.1.3 Reconstruction

In [4], sensitivity maps for reconstruction were computed using a dual-echo Cartesian GRE image and ESPIRiT with 24 central phase-encoding lines. Here, sensitivity maps were also computed using ESPIRiT, but from the raw, quite heavily subsampled data also used to reconstruct the image. This was first done using BART due to computation speed, but later also here the self-made Matlab script based on the Berkeley et al ESPIRiT demo [53] proved to perform better. Thresholds were determined by examining the size of the null space, quality of the eventual crop and using a quick reconstruction in Matlab multiplied with the sensitivity maps to see the alignment with the imaged object. Thresholds used are mentioned per sequence type.

For the BJORK reconstruction model, the trajectory needed to be converted from the .seq file to the desired Python format. When the size of the data and the desired reconstruction size were similar such as was the case for the TSE sequences in Section 2 (128 points per line and a desired reconstruction of 128 by 128), normalizing the trajectory from $-\pi$ to π was sufficient. However, in the BJORK-case, the data consisted of 32 spokes of 1280 points each, while the desired reconstruction size (and thus also the size of the sensitivity maps) was 320 by 320. This required expansion of the trajectory by a factor of 4 after normalizing from $-\pi$ to π. This was done for the trajectories of all sequences.

All reconstructions done with the BJORK model were compared to a reconstruction done using an adjoint NUFFT with sensitivity maps, in combination with density correction (dc). To the latter will simply be referred to as the NUFFT reconstruction in this section. The density correction was necessary for the BJORK data since the center of k-space was much more densely sampled than the outer areas of k-space, see Figure 19. This difference was less great in the radial GRE due to higher number of spokes used there. The sensitivity maps were implemented to make the reconstruction comparable to a SENSE reconstruction. This was done to truly examine the added value of the complete BJORK architecture, where *CG*-SENSE is responsible for the initial reconstruction, and a denoiser network for the final reconstruction, for a recap see Section 1.4.2.

3.2 Results

All three types of the made BJORK sequence were executed successfully on the scanner. The results are split into reconstructions per sequence type. For all sequences, the comparison between the BJORK and the NUFFT reconstruction is shown. Since the difference between the initial and final reconstruction of the BJORK model itself also showed to be interesting in sequence type 3, that comparison is also shown there.

3.2.1 Sequence type 1

Important to note was that computing sensitivity maps for this data was challenging due to the low SNR: quite extreme thresholds (0.6 and 0.98) were needed to define the null space and cropping region, resulting in a small null space and thus a correlation being found between only 12 instead of all 20 coils. This low SNR was most likely caused by the large FOV.

In Figure 22 both the reconstruction with BJORK, and the reconstruction with a NUFFT are shown. The results are cropped because of the large FOV. The BJORK reconstruction shows to be much sharper, especially when looking at the diagonal dots in the phantom. There are inhomogeneities visible in contrast within the phantom in both reconstructions. These are more pronounced in the BJORK reconstruction due to the blurring in the NUFFT reconstruction.

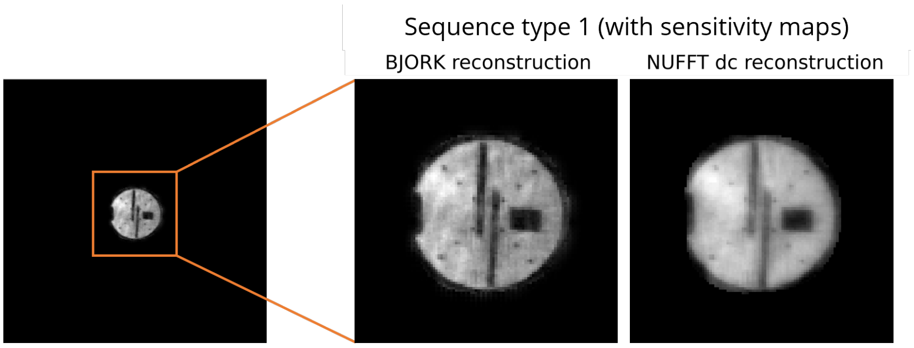
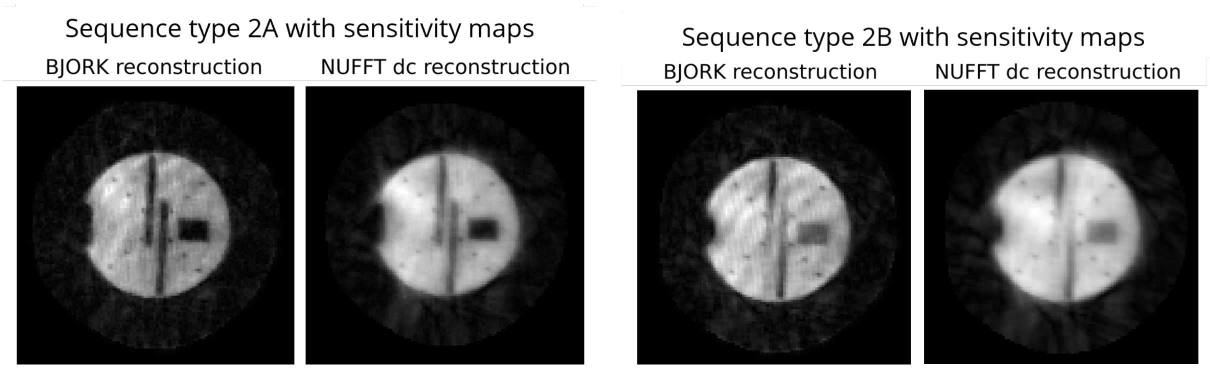


Figure 22: Reconstructions for sequence type 1: final BJORK and NUFFT reconstruction. Cropped due to the large FOV, visualized with the orange square. The BJORK reconstruction has an inhomogeneous contrast within the phantom, but the dots are distinguishable from the background. The NUFFT reconstruction is much more blurry than the BJORK reconstruction. In both, the sensitivity maps do not "fit" the phantom perfectly.

3.2.2 Sequence type 2

Thresholds used in the computation of sensitivity maps were 0.02 and 0.9 for both sequence 2A and 2B. When computing sensitivity maps for sequence 2B, also only a correlation between 12 instead of all 20 coils was found. Again the results are cropped due to the large FOV.

In Figure 23, the averaging effect of using a 200 mm slice instead of a 2 mm slice is immediately seen: both reconstructions in Figure 23b contain more blurring than both reconstructions in Figure 23a. More layers of the resolution phantom are expressed in the reconstruction, as is indicated by the lighter color of the rectangular block in the phantom. Furthermore, in both sequences, the BJORK reconstruction is sharper and has a higher contrast than the NUFFT reconstruction. In the BJORK reconstructions, a slight diagonal stripe pattern can be seen, not visible in the NUFFT reconstructions. For both BJORK and NUFFT, the sensitivity maps used did not "fit" the phantom perfectly, since also here, the large FOV caused relatively little signal to be present in the complete image, complicating the computation of sensitivity maps.



(a) (b)

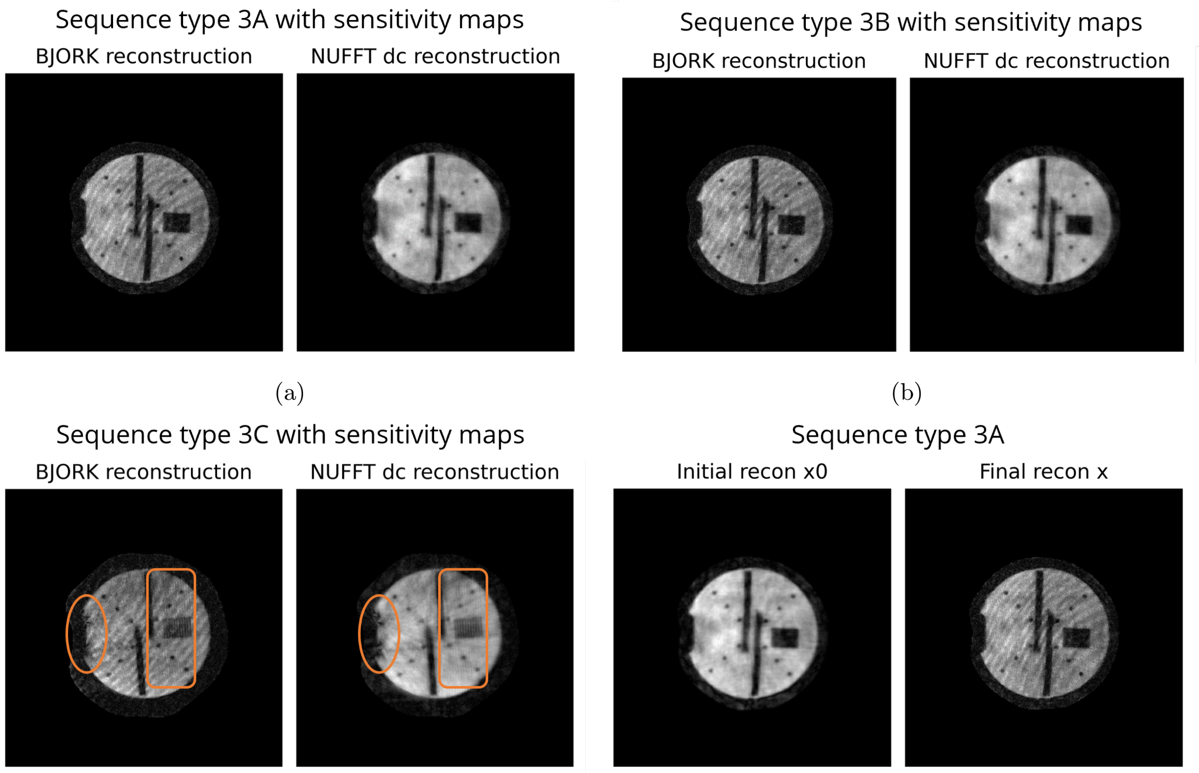
Figure 23: Reconstructions with BJORK model and NUFFT for sequence 2, cropped similarly as visualized in Figure 22 due to the large FOV. Reconstructions in Figure 23b are more blurry than in Figure 23a, but in both, the BJORK reconstruction is sharper than the NUFFT reconstruction. In both BJORK reconstructions, a light diagonal stripe-pattern is observed within the phantom. For all reconstructions, the sensitivity maps did not "fit" the phantom perfectly.

3.2.3 Sequence type 3

Thresholds used in computation of sensitivity maps were 0.06 & 0.9 for sequence 3C, and 0.025 & 0.985 for the othere sequences 3. The increase in thresholds used for sequence 3C compared to the others indicates the difficulty in obtaining accurate sensitivity maps for this data.

Single-slice phantom results Reconstruction results of the single-slice data can be seen in Figure 24a to Figure 24c. Edges and especially the diagonal dots in the BJORK reconstruction are sharp and well defined, but clear stripes in a wave-like pattern are seen across the entire phantom in all BJORK reconstructions. To examine where this pattern comes from, it is also interesting to see the difference between the initial guess and final reconstruction of the BJORK model in Figure 24d. Here, it is seen that an inhomogeneous contrast is already present within the phantom in the initial reconstruction, but that the wave-like pattern is not yet expressed clearly. It is (logically) very similar to the NUFFT reconstruction with dc and sensitivity maps in Figure 24a to Figure 24c, but slightly brighter, especially on the left of the phantom.

Comparing Figure 24a and Figure 24b, no differences between the two are observed visually. The reconstructions from Figure 24c do show clear differences compared to Figure 24a and Figure 24b. Artifacts are visible at the edge of the air bubble on the left of the phantom, which is slightly less expressed in the BJORK reconstruction compared to the NUFFT. The artifact persists when using different sensitivity maps. Since the phantom is flipped in orientation compared to sequence 3A and 3B, the wave-like pattern through the final BJORK reconstruction is flipped as well, indicating the phantom orientation does not play a role in the cause of the pattern. Interestingly enough, this orientation makes the thin stripes in the rectangular block on the right of the phantom appear quite clearly. These originate from a small staircase in the phantom, see Figure 10. They also seem to be elongated vertically throughout the phantom, which is not the case in the actual phantom. Again, the edges of the bars and dots in the phantom are sharper in the final BJORK reconstruction than in the initial or NUFFT reconstruction.



(a)

(b)

(c)

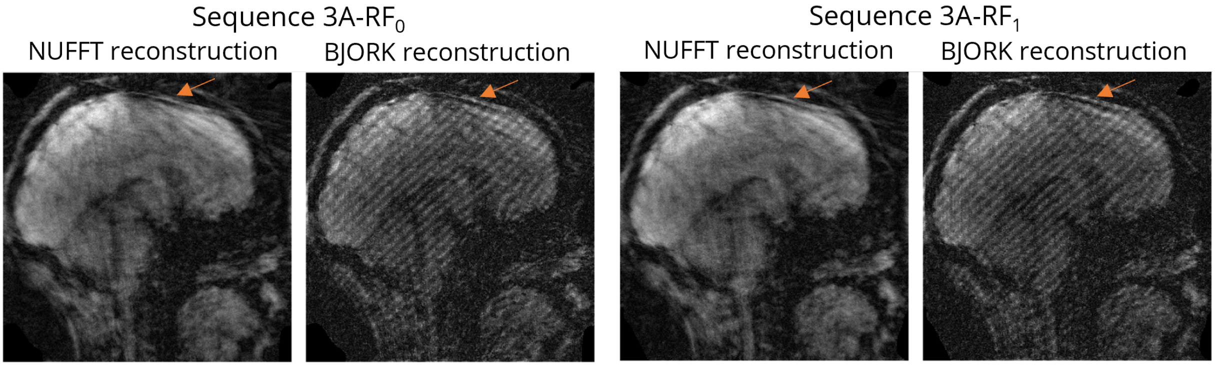
(d)

Figure 24: In a) to c) the BJORK and NUFFT reconstruction results of single-slice sequence 3. There is no difference observed between Figure 24a and Figure 24b. In Figure 24c, artifacts at the edges of the air bubble, marked by the circle, have appeared. Also, due to the orientation of the phantom, the ladder in the rectangular block is visible. The stripes extend out of the phantom, indicated with the rectangle. In all BJORK reconstructions, the diagonal stripe-like pattern is visible. In d) the initial guess x0 and final reconstruction x of the BJORK model for sequence 3A can be seen, where the stripe pattern is not yet visible in the initial reconstruction.

Multi-slice phantom results Since the same wave-like pattern is observed through all reconstructions of sequence type 3, reconstruction results of the multi-slice sequences 3D and 3E can be seen in Figure 37 in Appendix D.3. Acquiring slices in a non-subsequent order does not seem to improve results, only very slight differences can be seen in the size of the dark spots on the borders of the phantom.

Single-slice *in vivo* Only sagittal results are shown here, since this orientation allows inspecting a big variety in contrast and degree of detail. An example of the other orientations (transversal and coronal) scanned can be seen in Appendix D.3.

Single-slice *in vivo* BJORK sequences were acquired to assess the influence of the number of dummies and RF spoiling. In Figure 25, no improvement is observed visually when turning on RF spoiling. Streaks stemming from the top of the head are seen in both reconstructions, as well as a low overall resolution.

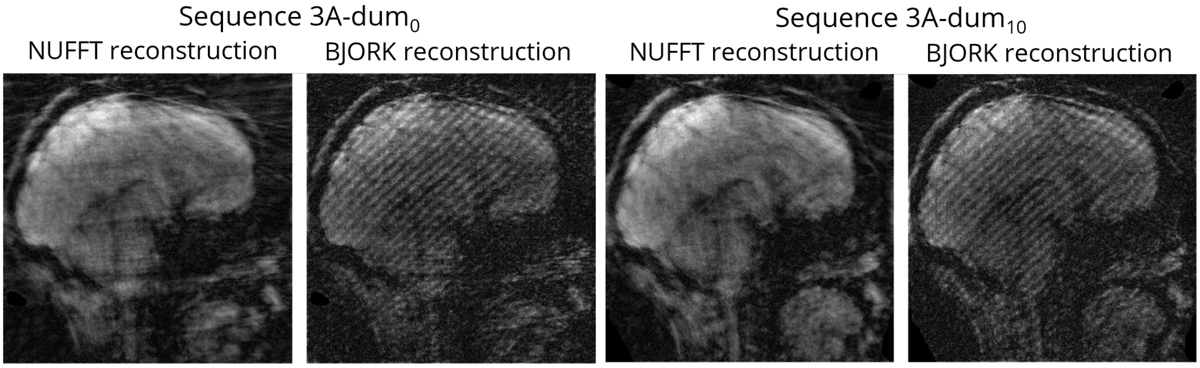


(a) Sagittal results sequence 3A-RF$_0$.

(b) Sagittal results sequence 3A-RF$_1$.

Figure 25: Reconstruction results of sequence 3A-RF executed *in vivo*. Both sequences used 10 dummies. No visual difference is observed between turning RF spoiling on or off. Especially note the streaks stemming from the top of the head in both reconstructions, marked by the arrow.

When keeping the RF spoiling on and increasing the number of dummies, there is clear improvement observed. When using zero dummies in Figure 26a, horizontal streaks are seen across the entire image. When increasing the number of dummies to ten in Figure 26b, the horizontal streaks diminish. Also the lower area of the head improves in quality. Still for using both zero and ten dummies, streaks stemming from the top of the head are observed. The transition from ten to twenty dummies did not show any improvement.



(a) Sagittal results sequence 3A-dum$_0$.

(b) Sagittal results sequence 3A-dum$_{10}$.

Figure 26: Reconstruction results of sequence 3A-dum$_0$ and 3A-dum$_{10}$ executed *in vivo*. RF spoiling was on for both. Note how the horizontal streaks present across the entire image when using zero dummies in Figure 26a, diminish significantly when increasing the number of dummies in Figure 26b. The streaks stemming from the top of the head are still present in both reconstructions.

Multi-slice *in vivo* Consecutive and interleaved slice acquisition results can be seen in Appendix D.3. Golden angle acquisition was also possible *in vivo* multi-slice, as can be seen in Figure 27. Here the slice acquisition order was also interleaved, so slice acquisition order 1, 3, 5, 2, and 4.

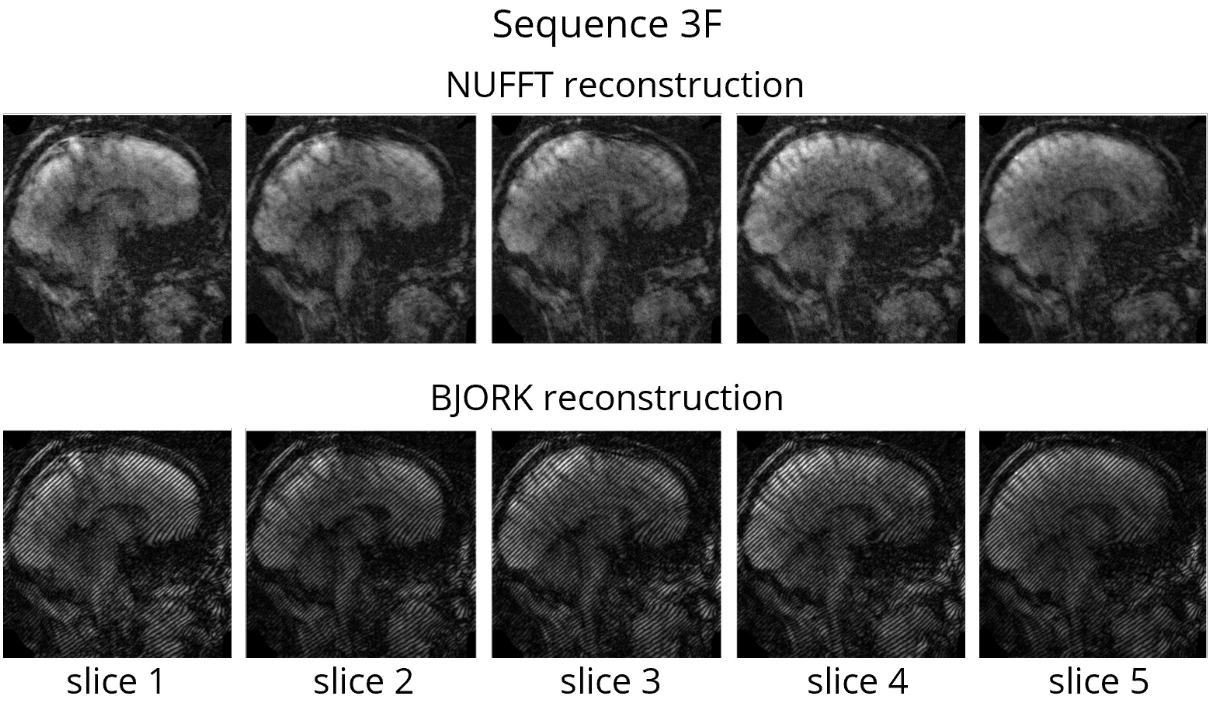


Figure 27: Reconstruction results of multi-slice sequence 3F executed *in vivo*. The top of the figure is the NUFFT reconstruction, the bottom the final BJORK reconstruction. Note how the streaks stemming from the top of the head have diminished a lot compared to Figure 26b, especially in slice 4 and 5. The BJORK reconstructions still show the wave-like artifact throughout all slices.

3.3 Discussion

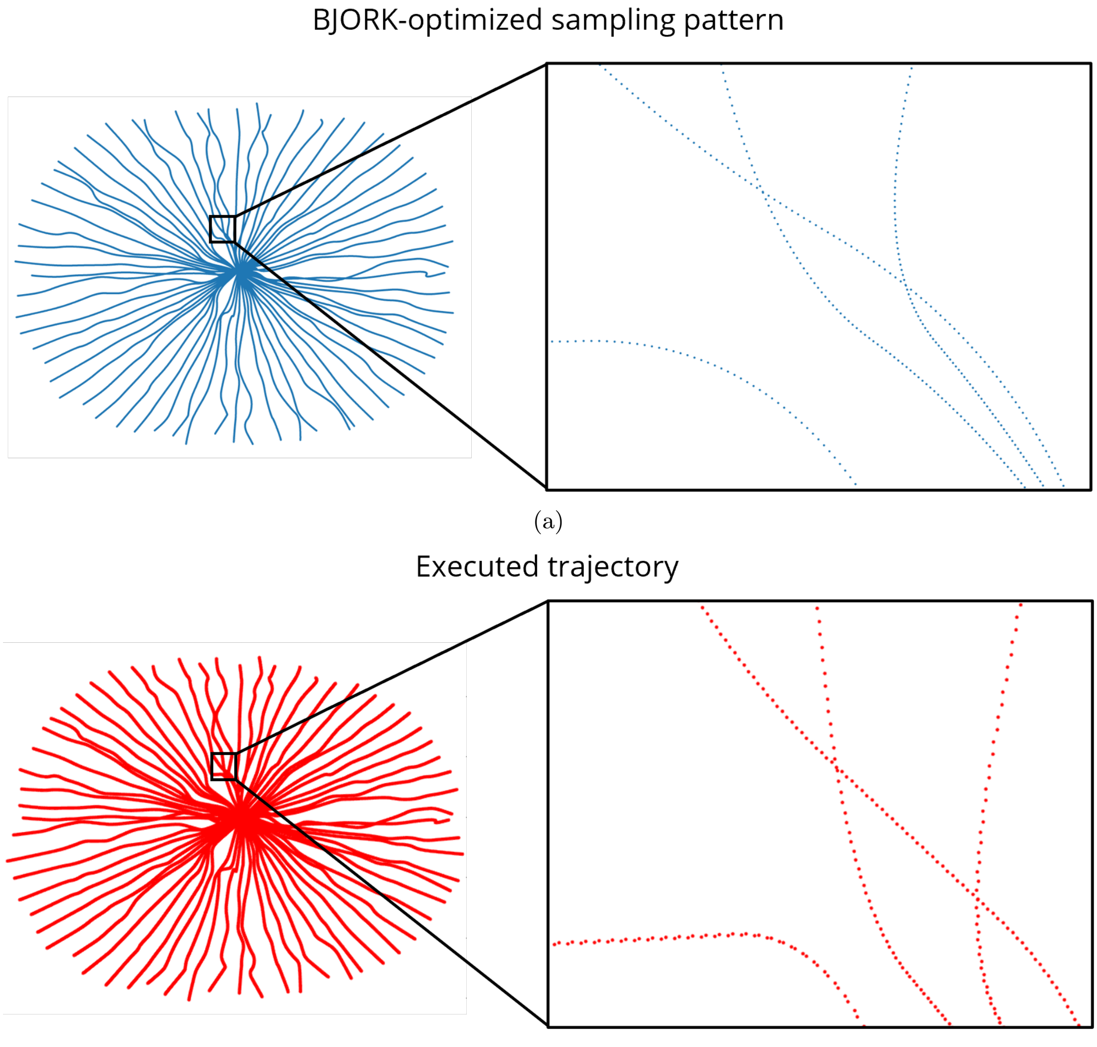
In this discussion, the design of the optimized sequence, and reconstruction results are discussed.

3.3.1 Sequence design

Sequence design was based solely on the output sampling pattern of [46], and thus the read-out gradient waveforms of the BJORK model, in combination with values for TE, TR, etc. from the original article [4] as an aim. That meant all components and timing besides the read-out gradient waveforms needed to be designed from scratch. This lack of specifics provides some freedom in sequence design, but also leaves room for sub-optimal choices. Therefore, it would be interesting to see what *sequence* design would be optimal, since sequence design choices do not only influence the resulting contrast, but also determine for instance slice thickness, and SNR. That would mean incorporating the RF pulses, z-gradients, and timings into the model. These timings should include TE and TR, but also raster- and dwell times of the MRI scanner, since these showed to be a big issue during implementation of the sampling pattern. A last component easily added to the BJORK model would be a constraint so that all gradient waveforms start and end at zero, which now had to be fixed separately during sequence design. All these adjustments are expected to decrease the number of steps needed between BJORK and actually executing the sequence on the scanner, improving the workflow.

When designing the sequence, I also implemented RF spoiling and dummy scans. That the original BJORK sequence was RF spoiled was mentioned in [4], but not *how* this was done. Also the "route" to the starting point of each spoke was not mentioned and needed to be handcrafted, in which the inverse of half of the gradient waveform of each spoke was used. Gradient spoiling could have been incorporated in here as well, but since this was not mentioned in the original implementation in [4], this was not done. It was also not mentioned if there were dummy cycles in their sequence. Therefore, the choices regarding spoiling, rephasing gradients and dummies might have been suboptimal. Investigating the optimal number of dummies and amount of spoiling further could improve resulting image quality. Decreasing the flip angle (which was set to 90°) could also be an interesting addition to try to improve the SNR [58].

The resulting designed trajectory through k-space deviated slightly from the optimized sampling pattern learned by the BJORK model due to all the adaptations needed for the pattern to be feasible in the scanner. In Figure 28, this is visualized, with in Figure 28a the learned sampling pattern, and in Figure 28b the eventually used trajectory. The spacing between points deviates in some spokes, as well as the degree of curvature. Since the actually used trajectory is used as an input in the reconstruction model, these deviations have not influenced the reconstructions in that sense, but striving for a trajectory closer to the learned optimal sampling pattern, might improve the data acquired in the sense of acquiring the most optimal points.



(a)

(b)

Figure 28: In a) the optimized sampling pattern learned by BJORK, and b) the executed trajectory, designed based on a). Since there is also scaling involved between the sampling pattern and the eventual trajectory, the zoomed-in boxes are slightly different in FOV. Note how not all curves in the patterns are the same, as well as the spacing between sampling points.

3.3.2 BJORK results

Even though not all reconstructions done with the BJORK model are as good, the BJORK sequence was designed and executed successfully, with reconstructed images as a result. This especially holds when keeping in mind the massive subsampling done by only acquiring 32 spokes with 1280 points: when acquiring 1280 points and wanting a fully sampled k-space, $\frac{\pi}{2}1280 = 2010$ spokes would need to be acquired [17]. This means that now with 32 spokes, not even 2% of hypothetically fully sampled data was acquired. It is also quite remarkable when realizing the BJORK model was trained on knee data, meaning the optimal sampling pattern used was optimal for imaging a knee. This does not only differ in contrasts from the phantom or *in vivo* head, but also differs significantly in shape. The original authors of the model [4] even show different optimal sampling patterns for the model trained on brain data than trained on knee data. Since this model still managed to make relatively good reconstructions, it indicates the model is quite robust for different anatomies.

Phantom results sequence 1 to 3 Reconstruction results from sequence 1 & 2 compared to sequence 3 did not only show differences in SNR due to the difference in FOV. Also the wave-like pattern in results from sequence 3 in Figure 24, was not yet so distinct in Figure 22 and Figure 23. A hint of the pattern is observed in Figure 23b, indicating the low SNR might have suppressed the pattern slightly earlier.

Influence of set maximum gradient and slew rate The maximum gradient and slew rate were set to 90% or 100% in sequence 3A and 3C respectively. It is unclear if the increase in artifacts visible for Figure 24c compared to Figure 24a was caused by the higher gradient and slew rate in Figure 24c, since the orientation of the phantom was also different, making the comparison not completely fair. On top of that, the quality of the sensitivity maps for results from sequence 3C was not good, which showed to have great influence on previous reconstruction results as well. The higher gradient and slew rate could also have resulted in larger eddy currents, thus leading to the actually executed trajectory deviating more from the trajectory in the sequence file for sequence 3C, compared to sequence 3A. Correction for eddy currents could be done by implementing pre-emphasis in the gradient waveforms [59], which is an alteration strongly advised to be implemented in Pulseq. Since it is not entirely clear how the scanner implements the sequence file, shimming could also play a role in the quality of the reconstructions [60]. Since the field is normally shimmed using the set FOV in the scanner, it might be important to set the FOV in such a way the scanner can read it properly. For all optimized sequences, the FOV was set in the definitions of the .seq file, making them appear in the Siemens interface, but it was not clear if that is also the FOV used to shim the field.

***In vivo* data** For *in vivo* measurements, the authors of the original implementation of BJORK [4], applied an additional fat saturation pulse before the 90° pulse. This was not implemented here, indicating their sequence design was slightly more advanced. This could contribute to why the *in vivo* reconstructions are not of great quality. As was seen in previous reconstructions, the quality of sensitivity maps used in BJORK reconstruction have a big influence on the quality of the reconstruction. As could be seen in Figure 25 and Figure 26, the sensitivity maps did not fit the object imaged as well as for example was the case in Section 2.2, confirming the relatively bad quality of sensitivity maps.

Influence of dummy scans and RF spoiling Even though *in vivo* BJORK reconstruction results had the same wave-like artifacts as the phantom data, still the influence of dummies and RF spoiling could be assessed successfully. RF spoiling showed no improvement in sagittal images. It would be interesting to examine this again when the wave-like artifact has been removed, since the difference might then be better observable. The quality of the reconstruction improved when using dummy scans as could be seen in Figure 26, but there was no difference observed between ten or twenty dummy scans. Using dummy cycles did mean an increase in acquisition time, since each dummy takes 1 TR.

Multi-slice data For the multi-slice phantom data, altering the slice acquisition scheme did not seem to improve results. However *in vivo*, interleaved acquisition showed to improve the result slightly. Golden angle acquisition did not show improvement in phantom data, as expected. It did seem to reduce blurring in the *in vivo* data in Figure 27. This should be assessed again by imaging an area *in vivo* with more motion inside, such as the bowel.

3.3.3 Wave-like artifact

In all reconstruction results from sequence 2 and 3, in figures Figure 23 to Figure 27, wave-like patterns were visible. It is yet unclear whether these artifacts originate from the used trajectory, acquired data, sensitivity maps or the reconstruction network. The orientation of the phantom was already ruled out when comparing results from sequence 3B and 3C in Figure 24b and Figure 24c respectively. Since the sensitivity maps were more difficult to obtain for sequence 3C (a higher threshold was needed to define the null space), the background area with signal around the phantom is larger compared to that area in sequences 3A and 3B. This area in 3C, especially visible at the left top, reveals that the stripe pattern in the final reconstruction might already be present in the background of the initial reconstruction. However, when looking at a gridding reconstruction where sensitivity maps have been multiplied with the reconstructed magnitude coil images, no stripes are visible, see Figure 29. The only pattern visible in this reconstruction is circular smearing. This indicates the cause of the wave-like artifact is not inherent to the magnitude data, trajectory or sensitivity maps. A noise spike was not seen in the raw data, so that is also not likely to have caused the wave-like artifact [61].

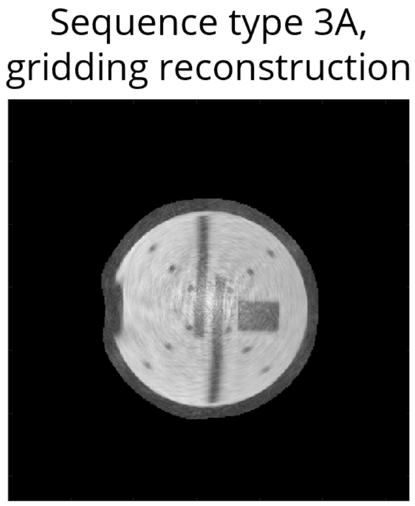
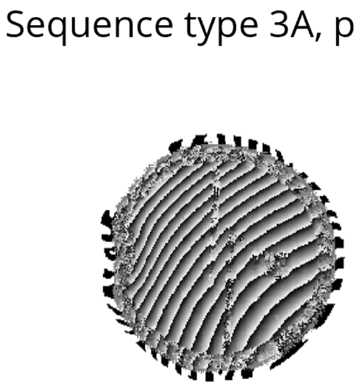


Figure 29: Gridding reconstruction of data from sequence 3A. The individual coil images have been multiplied with the sensitivity maps that were also used in the BJORK reconstruction. The coils were combined using the root-sum-of-squares. Note the circular smearing within the phantom, but no wave-like artifact.

Since the phase image of all sequences, of which an example can be seen in Figure 30 from sequence 3A, contains the stripe pattern present in the final reconstruction as well, this creates the hypothesis that the phase is causing the wave-like pattern in the reconstructions. The reason for such wave-like patterns in phase images are probably eddy currents, resulting in the executed trajectory to deviate from the trajectory in the sequence file. Eddy currents can cause a shift in resonance frequency, resulting in phase fluctuations [28, 59]. High gradients and slew rate increase the chance of eddy currents influencing the data, but in general gradient echo sequences are also more sensitive to such phase errors compared to spin echo sequences. In spin echo sequences, the 180° RF pulse mitigates such phase errors, which could be a reason these effects were not seen in the TSE results. The reason why the wave-like pattern was so much more pronounced in final BJORK reconstructions compared to solely SENSE is likely due to the CG-algorithm either converging too early or too late, or due to the regularization parameter, leading to amplification of the artifact. Tuning these parameters could give insight on what could be improved in the reconstruction model.



Sequence type 3A, phase

Figure 30: Phase image of the final BJORK reconstruction of sequence 3A. A clear striped, wave-like pattern can be seen, while usually the phase is uniform within the phantom.

3.4 Conclusion

Designing the optimized sequence in Pulseq and executing this on the MRI scanner was done successfully, but a lot of changes were needed for the sequence to adhere to the safety constraints of Pulseq and the Siemens scanner. Therefore, the now adapted BJORK-optimized sampling pattern is feasible, but the BJORK model should be improved to reduce the adaptations needed.

Eddy currents most likely influenced reconstruction results, especially visible with smaller FOVs. These should be relatively easily mitigated by incorporating gradient pre-emphasis. Since the sequence is handcrafted, changing the gradient waveform slightly to incorporate this pre-emphasis will fit into the current workflow. The BJORK reconstruction model already showed to be robust in Section 2, but now showed to not only be able to reconstruct phantom data recorded with the optimized sampling pattern, but also an *in vivo* brain of a volunteer. All this despite BJORK being trained on data very different in structure, and size.

4 Conclusion & Discussion

Overall, BJORK and Pulseq have been successfully implemented. Custom pulse sequences can be made in Pulseq and executed on the scanner, and the data can be reconstructed. Retro- and prospective subsampling of more conventional sequences was done successfully, and reconstruction using BJORK often performed better than a NUFFT reconstruction. Subsampling showed to accelerate MRI, while maintaining image quality. However, as mentioned in the respective discussion in Section 2.3, assessing reconstruction results based on diagnostic quality of *in vivo* images should be done to truly examine the possible degree of acceleration in a clinical setting.

Also, the execution of the optimized sampling pattern was successful. While reconstruction showed room for improvement, the BJORK reconstruction model showed to be robust to not only different anatomies, but also to completely different trajectories and data shapes. Before it can be concluded that the optimized sampling patterns have successfully accelerated MRI, the results need to be compared to reconstructions from a normal, similarly subsampled radial sampling pattern. That way, the true added value of the curves in the spokes can be examined. Unfortunately, at that time not all adaptations were done to the sequence yet to make this sequence feasible in the scanner. When also comparing results from a BJORK-optimized sequence learned for a specific anatomy to the state-of-the-art sequence for that specific anatomy, a conclusion can be drawn on whether BJORK-optimized sampling patterns also accelerate MRI in a clinical setting. Making sequences close to sequences present on the Siemens scanner was tried, but due to constraints in Pulseq and unknown components in the Siemens sequence, this did not succeed.

Even though the workflow as in Figure 18 showed to perform well, it would be ideal to combine the different steps in Python only. Pulseq also runs on Python (PyPulseq), and since eventually a self-made script to compute ESPIRiT sensitivity maps was used, this could also be transferred to Python. This would reduce the steps in the workflow to the following seen in Figure 31, also diminishing risk of errors in converting data formats. It would be even more optimized if the raw data was automatically sent to the computer used for reconstruction, with the eventual outlook of live reconstruction on the scanner. Until then, it is strongly recommended to always also compute rough reconstructions of raw data using BART in Matlab, before proceeding with more advanced reconstruction methods. Then, there is always a reference orientation available of the scanned object, and the ordering of the raw data has been confirmed.

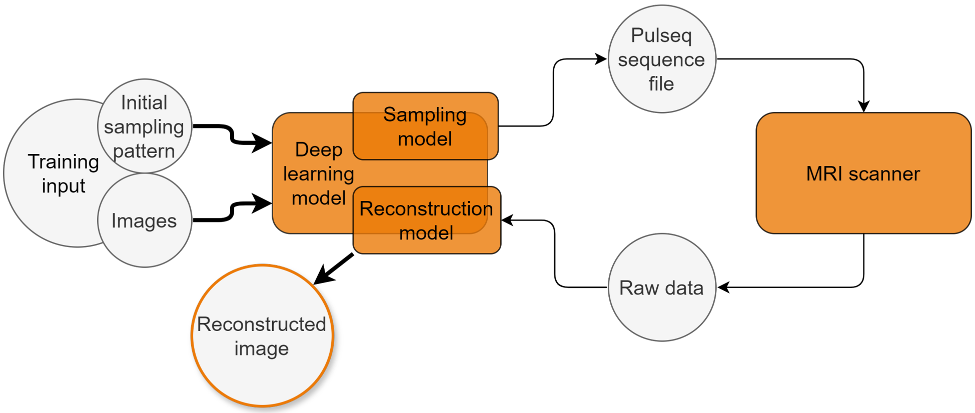


Figure 31: Proposed future workflow.

The subsampling and BJORK results from Section 2.2 and Section 3.2 have already been discussed in the respective chapters, but since the BJORK model, and the interaction between Pulseq and Siemens touches upon both, some general points are discussed here.

Current implementation of BJORK model Since I have only worked with the already trained BJORK model and did not re-train it myself, there are certain remarks for decisions made during the implementation of the model. First of all, training was done on k-space data cropped to 320 by 320.

It is known that using zero-padded k-space data instead of the true raw data to train deep learning models, leads to overly optimistic results [36]. While in cropping, data is not artificially generated, it does lead to a network trained on data of a lower resolution, possibly not utilizing the potential of the model architecture. Secondly, it was not made clear how sensitivity maps were retrieved, only that it was done using the ESPIRiT method on *fully sampled* data. Both in Section 3.2, and in Section 2.2, the sensitivity maps were not retrieved from additional fully sampled data, but from the data acquired with the BJORK or subsampled sequence respectively. Using sensitivity maps from fully sampled data will obviously improve the reconstruction results, but since that requires an additional scan, this is not desired when trying to accelerate MRI. Since the model results from only using sensitivity maps from fully sampled data, this could mean it is less robust to worse quality sensitivity maps. Also, when computing these maps myself, I have seen there is a lot of manual thresholding involved, inducing a bias in the method. It could be interesting to find more robust methods to obtain sensitivity maps from subsampled data, preferably better intertwined with the workflow.

A possible improvement could be to change the lower the learning rates during training to the values T. de Haan found in his experiments in [46]. The learning rates of the sampling layer and reconstruction network had been set to 10^{-3} and 10^{-5} respectively, based on [4], but T. de Haan saw that for both a small and middle sized dataset, $1 \cdot 10^{-4}$ was best for reconstruction and $5 \cdot 10^{-4}$ was best for sampling. The reason why these better learning rates were not adopted was unclear. Also the reasoning behind defining the value of other weighting factors, such as μ and λ was unclear. Eventually different values for these parameters showed improved results, based on trial-and-error. Investigating the optimal value for these parameters could be even more beneficial.

Pulseq & Siemens Since Pulseq is open-source software, it is being developed continuously. However, documentation regarding the interaction between Pulseq and the scanner sometimes lacks clarity, also because the makers of Pulseq do not have access to all the information they would like from Siemens. For example, it is unclear what parameters from Pulseq are registered by the scanner and used to adjust for example shimming. For instance, the FOV and slice thickness can be loaded into the Siemens interface when correctly specified in the .seq file, but it is not known if these values are also actually used. In general it is not known to what parameters the scanner shims the field when executing a .seq file. It is suspected the scanner uses some defaults when the .seq file does not contain appropriate information. This was seen when the slice thickness in the .seq file was not specified, when this resulted in a slice thickness of 200 mm in the Siemens interface. It is not known what slice thickness was used and if additional adjustments were done, except that the raw data did not contain any useful signal (only noise). Another argument for this hypothesis is that when building a sequence adhering to all safety and timing limits in Pulseq, but not specifying all raster times in the eventual .seq file, the scanner does not run the sequence. For now, it is advised to define the definitions in the .seq file in a way that is readable by the scanner, so that *if* the scanner alters certain components based on these, that the alterations are hopefully appropriate.

An important remark regarding Pulseq and Siemens is that it is unclear if the trajectory in the .seq file is executed accurately. It is assumed the actually executed trajectory deviates slightly due to field inhomogeneities, gradient imperfections and possible eddy currents, but maybe the scanner itself makes some standard alterations as well. Field mapping could be interesting to use to at least examine the hardware causes for deviation from the .seq trajectory [62]. Since the trajectory is used as input to the reconstruction model, a mismatch between acquired data and trajectory could have great influence on the reconstruction results.

Since Pulseq provides a lot of freedom in sequence design, when trying to replicate an already existing sequence, a lot of details regarding a sequence need to be known. *Exactly* replicating sequences shows to be very challenging without additional information from the vendor [63]. Since it is very difficult to make fair comparisons between results without such sequences, it is recommended to endure in collaborating with vendors to retrieve source code of sequences.

Future research Recommendations for future research regarding specifics have already been discussed in Section 2.3 and Section 3.3, but some general ideas and recommendations still remain.

The most important suggestion is to compare reconstruction results to a state-of-the art reconstruction, already mentioned earlier. The quality of reconstruction results should be *quantified*. In this research, this has only been done for a few of the subsampling reconstructions using the SSIM. It was not possible for all reconstructions due to the lack of an appropriate ground truth image. It was chosen to not use an SNR metric due to the BJORK reconstruction method setting the background to zero based on sensitivity maps, making the comparison with a reconstruction with a non-zero background inappropriate. Research into other quantifying parameters was not done due to time constraints. It is recommended to either look into other parameters quantifying image quality, or ensuring that an appropriate ground truth image is available for each experiment performed.

Now that a robust workflow is established, more applications or variations in pulse sequences could be of interest. For example using sequences with many "random" events and parameters for fingerprinting [64], or examining different flip angles for every few spokes for multi-contrast imaging. Also, optimizing the flip angle used, could be interesting to possibly increase the signal [58, 65]. Designing 3D sequences is also possible in Pulseq, paving way for creating randomly subsampled k-space sampling patterns and being able to execute these in the scanner. Golden angle BJORK was already executed successfully, so this could also be implemented in 3D. However, using motion data for assessing the added value of these golden angle sequences is a must. This could also be examined together with cardiac gating. When instead of eliminating motion artifacts, motion (flow) wants to be examined, the phase issues in BJORK should first be solved.

Next to using Pulseq as it is to design pulse sequences, a recently published article proposes using a chatGPT environment for Pulseq, making it easier to build basic sequences [66]. While this does not yet suffice for more complex sequences, it could be interesting as a basis, and to keep an eye on their work.

In this research, for the first time at the University of Twente, k-space sampling patterns optimized by a deep learning model have been executed on the MRI scanner. When addressing the limitations and recommendations stated, this work can be used for accelerating MRI in the future: making health care faster, cheaper, and more accessible for patients.

Acknowledgements

As you have already seen on the front page of this thesis, the many failed image reconstructions almost got to me. Fortunately, the belief of my supervisors in my ability to conquer Matlab and Python errors, kept me going. Especially thanks to my daily supervisor Frank for his infectious enthusiasm, and for always being available to help out. Last, but certainly not least, thanks to my friends and family for being my rubber duck or well deserved distraction, and for letting me proudly play MRIah Carey many times on full volume:

References

[1] Eurostat in Statista. *Number of magnetic resonance imaging (MRI) scan examinations in the Netherlands from 2009 to 2020.* Nov. 2022. URL: https://www.statista.com/statistics/609151/magnetic-resonance-imaging-scan-examinations-in-the-netherlands/.

[2] Kroboth S. Jia F. Littin S. Yu H. Leupold J. Nielsen J. F. Stöcker T. Layton K. J. and M. Zaitsev. “Pulseq: A rapid and hardware-independent pulse sequence prototyping framework”. In: *Magnetic Resonance in Medicine* 77 (4 Apr. 2017), pp. 1544–1552. ISSN: 15222594. DOI: 10.1002/mrm.26235.

[3] Kroboth S. Zaitsev M. and K. Layton. *Open file format for MR sequences - Version 1.4.1.* 2023. URL: http://pulseq.github.io/http://pulseq.github.io/http://creativecommons.org/licenses/by/4.0/http://creativecommons.org/licenses/by/4.0/.

[4] Luo T. Nielsen J.F. Noll D.C. Wang G. and J.A. Fessler. “B-Spline Parameterized Joint Optimization of Reconstruction and K-Space Trajectories (BJORK) for Accelerated 2D MRI”. In: *IEEE Transactions on Medical Imaging* 41 (9 Sept. 2022), pp. 2318–2330. ISSN: 1558254X. DOI: 10.1109/TMI.2022.3161875.

[5] Sartoretti T. Sartoretti E. and S. Sartoretti-Schefer. “Reduction of procedure times in routine clinical practice with Compressed SENSE magnetic resonance imaging technique”. In: *PLOS ONE* 14 (4 Apr. 2019), e0214887. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0214887.

[6] Ziekenhuischeck and Nederlandse Zorgautoriteit. *Wachttijden Medisch Spectrum Twente.* Mar. 2024. URL: https://www.ziekenhuischeck.nl/ziekenhuizen/medisch-spectrum-twente/behandeling/wachttijden/.

[7] Weiss P. Chauffert N. Mauconduit F. El Gueddari L. Destrieux C. Zemmoura I. Vignaud A. Lazarus C. and P. Ciuciu. “SPARKLING: variable-density k-space filling curves for accelerated T2*-weighted MRI”. In: *Magnetic Resonance in Medicine* 81 (6 June 2019), pp. 3643–3661. ISSN: 15222594. DOI: 10.1002/mrm.27678.

[8] Senouf O. Vedula S. Michailovich O. Zibulevsky M. Weiss T. and A. Bronstein. “PILOT: Physics-Informed Learned Optimized Trajectories for Accelerated MRI”. In: (Sept. 2019). URL: http://arxiv.org/abs/1909.05773.

[9] C. G. Radhakrishna and P. Ciuciu. “Jointly Learning Non-Cartesian k-Space Trajectories and Reconstruction Networks for 2D and 3D MR Imaging through Projection”. In: (2023). DOI: 10.3390/bioengineering. URL: https://doi.org/10.3390/bioengineering10020158.

[10] D. G. Nishimura. *Principles of Magnetic Resoncance Imagin.* 1.3. Lulu, 2010.

[11] B.A. Jung and M. Weigel. “Spin echo magnetic resonance imaging”. In: *Journal of Magnetic Resonance Imaging* 37 (4 Apr. 2013), pp. 805–817. ISSN: 1053-1807. DOI: 10.1002/jmri.24068.

[12] M. Markl and J. Leupold. “Gradient echo imaging”. In: *Journal of Magnetic Resonance Imaging* 35 (6 June 2012), pp. 1274–1289. ISSN: 10531807. DOI: 10.1002/jmri.23638.

[13] A.D. Elster. *MRI questions - Gradient Echo (GRE).* 2014. URL: https://mriquestions.com/gradient-echo.html.

[14] A.D. Elster. *MRI questions - Signal strength.* 2014. URL: https://mriquestions.com/how-is-signal-higher.html.

[15] A.D. Elster. *MRI questions - FOV and k-space.* 2014. URL: https://mriquestions.com/field-of-view-fov.html.

[16] K. Miller. *MRI image formation - lecture notes.* Apr. 1999. URL: users.fmrib.ox.ac.uk/~karla/teaching/image_formation.ppt.

[17] Hamilton J.I. Griswold M.A. Gulani V. Wright K. L. and N. Seiberlich. “Non-Cartesian parallel imaging reconstruction”. In: *Journal of Magnetic Resonance Imaging* 40 (5 Nov. 2014), pp. 1022–1040. ISSN: 15222586. DOI: 10.1002/jmri.24521.

[18] L. Feng. “Golden-Angle Radial MRI: Basics, Advances, and Applications”. In: *Journal of Magnetic Resonance Imaging* 56 (1 July 2022), pp. 45–62. ISSN: 15222586. DOI: 10.1002/jmri.28187.

[19] D.A. Feinberg et al. “Halving MR Imaging Time by Conjugation: Demonstration at 3.5 kG”. In: *Radiology* 161 (1986), pp. 527–531.

[20] C.E. Shannon. “Communication in the Presence of Noise”. In: *Proceedings of the IRE* 37.1 (1949), pp. 10–21. DOI: 10.1109/JRPROC.1949.232969.

[21] Michael Lustig et al. “Compressed sensing MRI: A look at how CS can improve on current imaging techniques”. In: *IEEE Signal Processing Magazine* 25 (2 2008), pp. 72–82. ISSN: 10535888. DOI: 10.1109/MSP.2007.914728.

[22] D.J. Larkman and R.G. Nunes. “Parallel magnetic resonance imaging”. In: *Physics in Medicine and Biology* 52 (7 Apr. 2007). ISSN: 00319155. DOI: 10.1088/0031-9155/52/7/R01.

[23] M. Lustig. *ISMRM MR Academy - Compressed Sensing in MRI*. 2013. URL: https://www.youtube.com/watch?v=AP6JczMW8C8&t=464s&ab_channel=ISMRM-ISMRT.

[24] Franson D. Hamilton J. and N. Seiberlich. “Recent advances in parallel imaging for MRI”. In: *Progress in Nuclear Magnetic Resonance Spectroscopy* 101 (Aug. 2017), pp. 71–95. ISSN: 00796565. DOI: 10.1016/j.pnmrs.2017.04.002.

[25] Jeffrey A Fessler. “Optimization methods for MR image reconstruction (long version)”. In: (Mar. 2019). URL: http://arxiv.org/abs/1903.03510.

[26] Philip J. Beatty, Dwight G. Nishimura, and John M. Pauly. “Rapid gridding reconstruction with a minimal oversampling ratio”. In: *IEEE Transactions on Medical Imaging* 24 (6 June 2005), pp. 799–808. ISSN: 02780062. DOI: 10.1109/TMI.2005.848376.

[27] J.A. Fessler and B.P. Sutton. “Nonuniform Fast Fourier Transforms Using Min-Max Interpolation”. In: 51 (2 2003), pp. 560–74.

[28] Ryan K. Robison et al. “Correction of B0 eddy current effects in spiral MRI”. In: *Magnetic Resonance in Medicine* 81 (4 Apr. 2019), pp. 2501–2513. ISSN: 15222594. DOI: 10.1002/mrm.27583.

[29] Weiger M. Scheidegger M.B. Pruessmann K.P. and P. Boesiger. “SENSE: Sensitivity encoding for fast MRI”. In: *Magnetic Resonance in Medicine* 42 (5 1999), pp. 952–962. ISSN: 07403194. DOI: 10.1002/(SICI)1522-2594(199911)42:5<952::AID-MRM16>3.0.CO;2-S.

[30] Jakob P.M. Heidemann R.M. Nittka M. Jellus V. Wang J. Kiefer B. Griswold M.A. and A. Haase. “Generalized Autocalibrating Partially Parallel Acquisitions (GRAPPA)”. In: *Magnetic Resonance in Medicine* 47 (6 2002), pp. 1202–1210. ISSN: 07403194. DOI: 10.1002/mrm.10171.

[31] Michael Lustig and John M. Pauly. “SPIRiT: Iterative self-consistent parallel imaging reconstruction from arbitrary k-space”. In: *Magnetic Resonance in Medicine* 64 (2 2010), pp. 457–471. ISSN: 15222594. DOI: 10.1002/MRM.22428.

[32] Lai P. Murphy M.J. Virtue P. Elad M. Pauly J.M. Vasanawala S.S. Uecker M. and M. Lustig. “ESPIRiT - An eigenvalue approach to autocalibrating parallel MRI: Where SENSE meets GRAPPA”. In: *Magnetic Resonance in Medicine* 71 (3 2014), pp. 990–1001. ISSN: 15222594. DOI: 10.1002/mrm.24751.

[33] Liu J.Z. Cauley S.F. Rosen B.R. Zhu B. and M.S. Rosen. “Image reconstruction by domain-transform manifold learning”. In: *Nature* 555 (7697 Mar. 2018), pp. 487–492. ISSN: 0028-0836. DOI: 10.1038/nature25988.

[34] Mani M.P. Aggarwal H.K. and M. Jacob. “MoDL: Model-Based Deep Learning Architecture for Inverse Problems”. In: *IEEE Transactions on Medical Imaging* 38 (2 Feb. 2019), pp. 394–405. ISSN: 1558254X. DOI: 10.1109/TMI.2018.2865356.

[35] Arghya Pal and Yogesh Rathi. “A review and experimental evaluation of deep learning methods for MRI reconstruction”. In: (Sept. 2021). URL: http://arxiv.org/abs/2109.08618.

[36] Tamir J.I. Wang K. Shimron E. and M. Lustig. “Implicit data crimes: Machine learning bias arising from misuse of public data”. In: (2023). DOI: 10.1073/pnas.

[37] G. Varoquaux and V. Cheplygina. “Machine learning for medical imaging: methodological failures and recommendations for the future”. In: *npj Digital Medicine* 5 (1 Apr. 2022), p. 48. ISSN: 2398-6352. DOI: 10.1038/s41746-022-00592-y.

[38] Z. Wang et al. “Image quality assessment: from error visibility to structural similarity”. In: *IEEE Transactions on Image Processing* 13.4 (2004), pp. 600–612. DOI: 10.1109/TIP.2003.819861.

[39] Moeller S. Weingärtner S. Akçakaya M. and K. Uğurbil. “Scan-specific robust artificial-neural-networks for k-space interpolation (RAKI) reconstruction: Database-free deep learning for fast imaging”. In: *Magnetic Resonance in Medicine* 81 (1 Jan. 2019), pp. 439–453. ISSN: 0740-3194. DOI: 10.1002/mrm.27420.

[40] Fischer P. Ronneberger O. and T. Brox. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. In: (May 2015). URL: http://arxiv.org/abs/1505.04597.

[41] Li Y. Monga V. and Y.C. Eldar. “Algorithm Unrolling: Interpretable, Efficient Deep Learning for Signal and Image Processing”. In: (Dec. 2019). URL: http://arxiv.org/abs/1912.10557.

[42] Wang A.Q. Dalca A.V. Bahadir C.D. and M.R. Sabuncu. “Deep-learning-based Optimization of the Under-sampling Pattern in MRI”. In: (July 2019). URL: http://arxiv.org/abs/1907.11374.

[43] Ferdia Sherry et al. “Learning the Sampling Pattern for MRI”. In: *IEEE Transactions on Medical Imaging* 39 (12 Dec. 2020), pp. 4310–4321. ISSN: 1558254X. DOI: 10.1109/TMI.2020.3017353.

[44] H. C. Raymond and Michael K.NG. “Conjugate gradient for toeplitz systems”. In: *Society for Industrial and Applied Mathematics* 38 (3 1996), p. 427482.

[45] Park B. Yu S. and J. Jeong. “Deep iterative down-up CNN for image denoising”. In: vol. 2019-June. IEEE Computer Society, June 2019, pp. 2095–2103. ISBN: 9781728125060. DOI: 10.1109/CVPRW.2019.00262.

[46] T. de Haan. *Neural Field and Spline-based Representations for Optimizing the k-space Trajectory for MRI with Deep learning.* Oct. 2023. URL: http://essay.utwente.nl/97500/.

[47] Gross S. Bai J. Chintala S. et al Paszke A. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: (Dec. 2019).

[48] Knoll F. Sodickson D.K. Lui Y.W. et al Zbontar J. “fastMRI: An Open Dataset and Benchmarks for Accelerated MRI”. In: (Nov. 2018). URL: http://arxiv.org/abs/1811.08839.

[49] Florian Knoll et al. “FastMRI: A publicly available raw k-space and DICOM dataset of knee images for accelerated MR image reconstruction using machine learning”. In: *Radiology: Artificial Intelligence* 2 (1 Jan. 2020). ISSN: 26386100. DOI: 10.1148/ryai.2020190007.

[50] M. Zaitsev. *Pulseq on Siemens or how to use Pulseq MR pulse sequence environment on Siemens scanners.* 2023.

[51] F.X. Hebrank and M. Gebhardt. “SAFE-Model-A New Method for Predicting Peripheral Nerve Stimulations in MRI”. In: *Proc. Intl. Soc. Mag. Res. Med.* 8 (2000).

[52] Westin C.F. Szczepankiewicz F. and M. Nilsson. “Gradient waveform design for tensor-valued encoding in diffusion MRI”. In: *Journal of Neuroscience Methods* 348 (Jan. 2021), p. 109007. ISSN: 0165-0270. DOI: 10.1016/J.JNEUMETH.2020.109007.

[53] et al M. Lustig. *ESPIRiT Maps Demo.* URL: https://people.eecs.berkeley.edu/~mlustig/software/{ESPIRiT}_demo/demo_{ESPIRiT}_maps.html.

[54] Kai Tobias Block, Martin Uecker, and Jens Frahm. “Suppression of MRI truncation artifacts using total variation constrained data extrapolation”. In: *International Journal of Biomedical Imaging* 2008 (1 2008). ISSN: 16874188. DOI: 10.1155/2008/184123.

[55] Y. Zur, M. L. Wood, and L.J. Neuringer. “Spoiling of Transverse Magnetization in Steady-State Sequences”. In: *Magnetic Resonance in Medicine* 21 (2 Oct. 1991), pp. 251–263. DOI: 10.1002/mrm.1910210210.

[56] K. Scheffler. “A pictorial description of steady-states in rapid magnetic resonance imaging”. In: *Concepts in Magnetic Resonance* 11 (5 1999), pp. 291–304. ISSN: 1043-7347. DOI: 10.1002/(SICI)1099-0534(1999)11:5<291::AID-CMR2>3.0.CO;2-J.

[57] Cho J. Kim Y. Cho Y. et al. Lee S. “A study on a method to reduce the effect of the cross-talk artifact in a simultaneous, multiple-slice, plane, oblique MRI scan”. In: *Journal of the Korean Physical Society* 61 (5 Sept. 2012), pp. 807–814. ISSN: 0374-4884. DOI: 10.3938/jkps.61.807.

[58] R.R. Ernst and W.A. Anderson. “Application of Fourier Transform Spectroscopy to Magnetic Resonance”. In: *The Review of Scientific Instruments* 37 (1 Jan. 1966).

[59] C.B. Ahn and Z.H. Cho. “Analysis of the Eddy-Current Induced Artifacts and the Temporal Compensation in Nuclear Magnetic Resonance Imaging”. In: *IEEE TRANSACTIONS ON MEDICAL IMAGING* IO (1 1991), p. 47.

[60] M. Wendt and Siemens Medical Solutions USA Inc. *MRI Hot Topics - Second Order Shimming of High Field Magnets s medical.*

[61] Seethamraju R.T. Patel P. Fahn P.F. Kirsch J.E. Huang S.Y. and A.R. Guimaraes. “Body MR Imaging: Artifacts, k-Space, and Solutions.” In: *Radiographics : a review publication of the Radiological Society of North America, Inc* 35 (5 2015), pp. 1439–60. ISSN: 1527-1323. DOI: 10.1148/rg.2015140289.

[62] P. Keller. *Technologies for Precision Magnetic Field Mapping.* Metrolab Instruments, Geneva Switzerland. URL: www.metrolab.com.

[63] Bauman G. Bieri O. Lee N.G. and K.S. Nayak. “Replication of the bSTAR sequence and open-source implementation”. In: *Magnetic Resonance in Medicine* (2023). ISSN: 15222594. DOI: 10.1002/mrm.29947.

[64] C. Tippareddy et al. “Magnetic resonance fingerprinting: an overview”. In: *European Journal of Nuclear Medicine and Molecular Imaging* 48 (13 Dec. 2021), pp. 4189–4200. ISSN: 1619-7070. DOI: 10.1007/s00259-021-05384-2.

[65] Frahm J. Matthaei D. Haase A. and K.D. Merboldt. “FLASH Imaging. Rapid NMR Imaging Using Low Flip-Angle Pulses”. In: *JOURNAL OF MAGNETIC RESONANCE* 67 (1986), pp. 258–266.

[66] Rajput J.R. Dang H.N. Golkov V. Cremers D. Knoll F. Zaiss M. and A. Maier. “Exploring GPT-4 as MR Sequence and Reconstruction Programming Assistant”. In: (2024), pp. 94–99. DOI: 10.1007/978-3-658-44037-4_28. URL: https://link.springer.com/10.1007/978-3-658-44037-4_28.

[67] Fischer H. Hentschel D. Reinfelder H.E. Budinger T.F. and F. Schmitt. “Physiological effects of fast oscillating magnetic field gradients.” In: *Journal of computer assisted tomography* 15 (6 1991), pp. 909–14. ISSN: 0363-8715. DOI: 10.1097/00004728-199111000-00001.

[68] Nyenhuis J.A. Bourland J.D. and D.J. Schaefer. “Physiologic effects of intense MR imaging gradient fields.” In: *Neuroimaging clinics of North America* 9 (2 May 1999), pp. 363–77. ISSN: 1052-5149.

[69] P. Ehses. *mapVBVD*. 2016.

Appendices

A PNS calculations

PNS estimations were done using the Pulseq implementation of [52]. The necessary scripts can be downloaded from Github[3]. The implementation is based on the (Siemens) SAFE model from [51]. In this model, the gradient waveforms are filtered, rectified and weighted by a scale factor, resulting in the stimulation signal. With empirical research, stimulation thresholds were found, and parameters of the model were established accordingly. When the threshold for the stimulation signal is exceeded, the nerves will be stimulated. The specific parameters used in the model such as time constants, are confidential hardware specifications specific for the system, and can be found on the scanner at .../Medcom/MRIproduct/Measurement/InitMeas.

An example of the predicted PNS for a BJORK sequence (version 3 from Table 8 using 90% of the maximum gradient and slew rate), can be seen in Figure 32. Here the predicted PNS over the entire sequence can be seen, split into contributions from the x-, y-, and z-gradient. The clear pause between peaks corresponds to the time between the acquisition of a spoke and the start of the RF pulse of the next one. A zoomed in version can be seen in Figure 33, where the contribution of the separate gradients can be distinguished better. In this sequence, the z-gradient contributes most to the predicted PNS due to its trapezoid shape, and for the rephasing z-gradient its large area.

According to [67, 68], PNS can become uncomfortable when the threshold is exceeded by half, meaning a PNS of 150% in these plots.

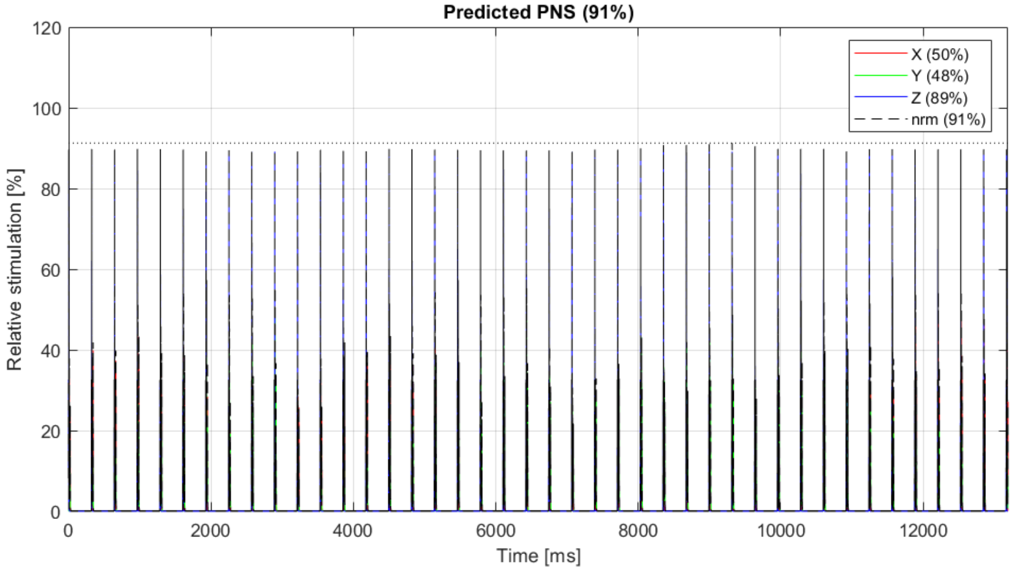


Figure 32: Complete output plot from the Pulseq PNS prediction. Clear spikes can be seen corresponding to the moments in the sequence the gradients are turned on.

[3] https://github.com/filip-szczepankiewicz/safe_pns_prediction?tab=readme-ov-file

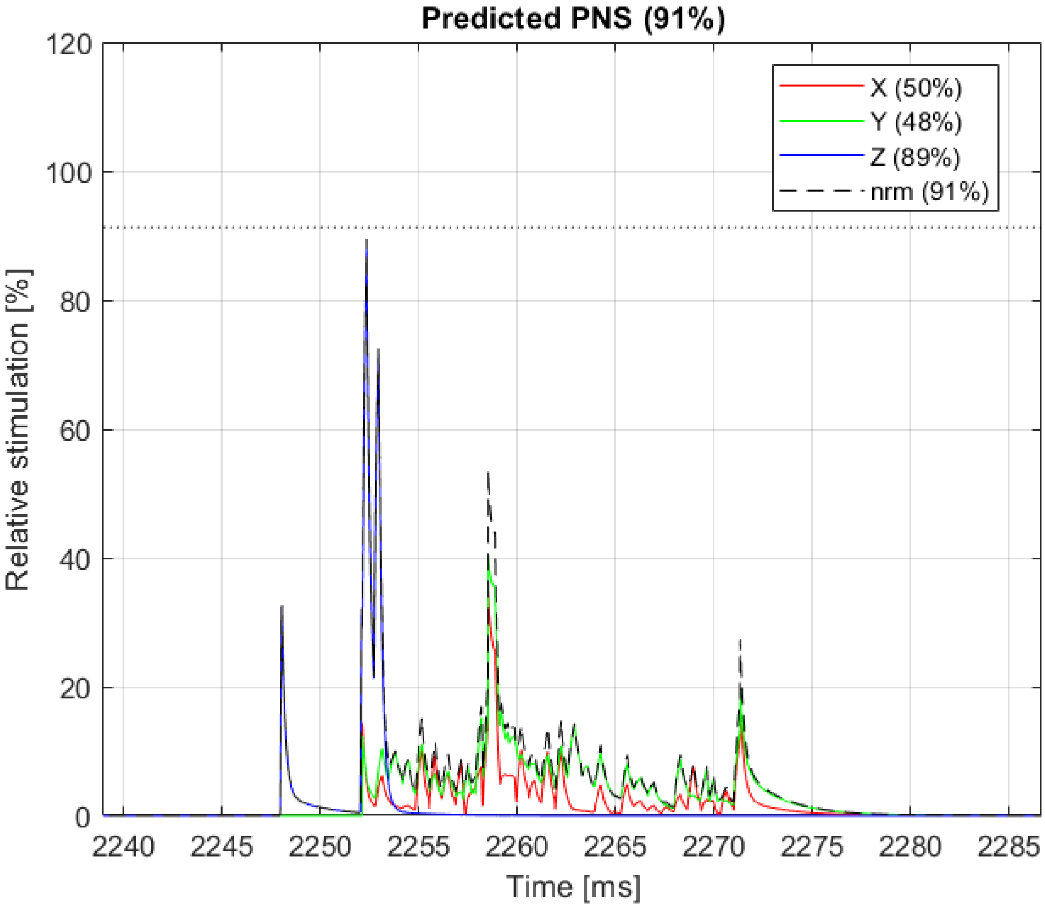


Figure 33: Output of Pulseq PNS prediction, zoomed in on arbitrary spoke in the sequence. A distinction is made between the x-, y-, and z-gradient. The stimulation threshold is given by the dashed line 'nrm'.

B Reconstruction parameters μ and λ

The positive penalty μ from Equation (9), and the regularization parameter λ from Equation (8) were changed to obtain the best reconstruction results, see Figure 34. Little difference is observed when keeping μ constant and changing λ, except for when μ is 10. The combination μ and λ both 0.01 was chosen for all BJORK reconstructions in this thesis.

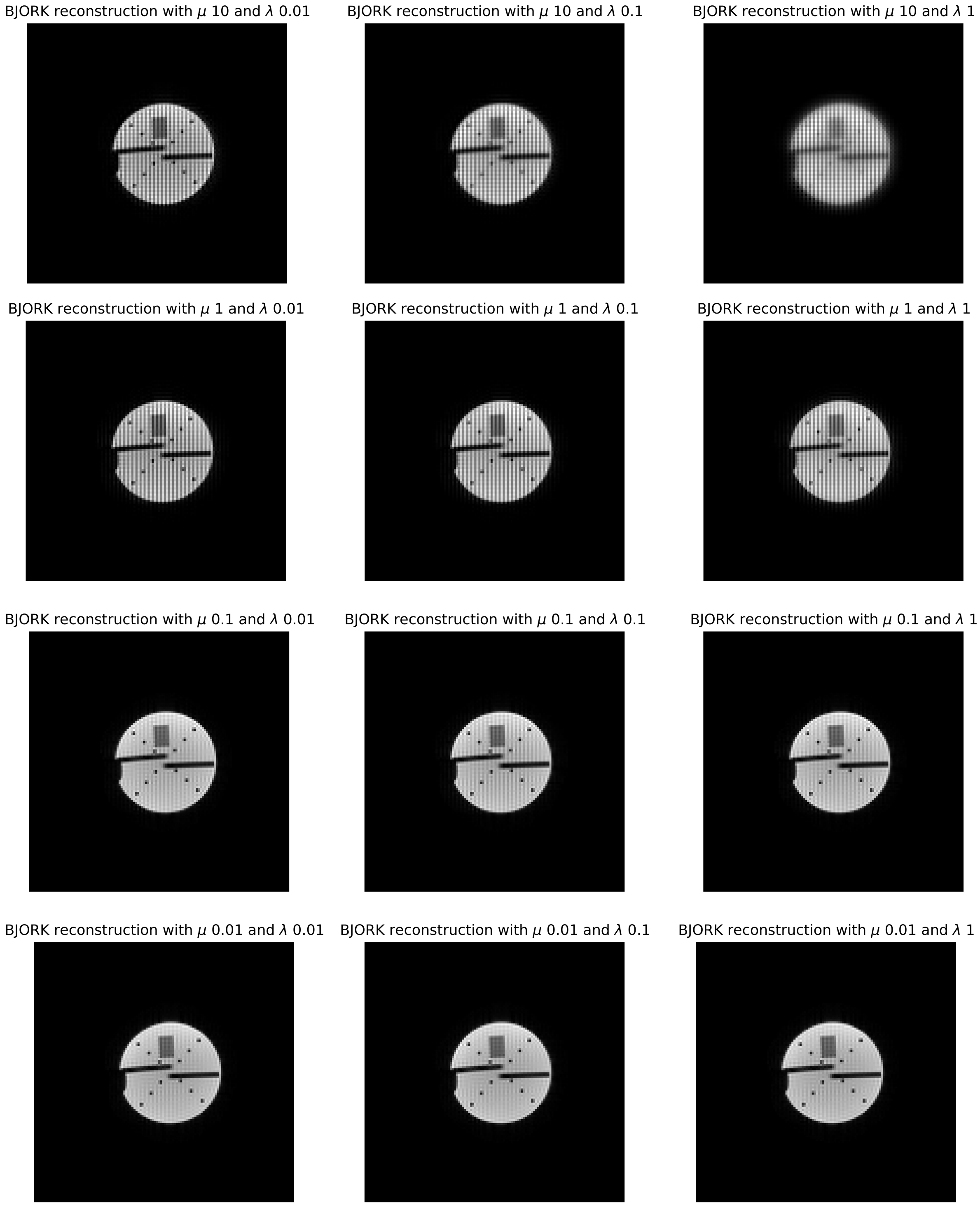

BJORK reconstruction with μ 10 and λ 0.01
BJORK reconstruction with μ 10 and λ 0.1
BJORK reconstruction with μ 10 and λ 1
BJORK reconstruction with μ 1 and λ 0.01
BJORK reconstruction with μ 1 and λ 0.1
BJORK reconstruction with μ 1 and λ 1
BJORK reconstruction with μ 0.1 and λ 0.01
BJORK reconstruction with μ 0.1 and λ 0.1
BJORK reconstruction with μ 0.1 and λ 1
BJORK reconstruction with μ 0.01 and λ 0.01
BJORK reconstruction with μ 0.01 and λ 0.1
BJORK reconstruction with μ 0.01 and λ 1

Figure 34: Reconstruction results of fully sampled TSE data for different μ and λ used in the BJORK reconstruction model.

C Multi-slice TSE

A fully sampled TSE sequence with 5 slices was designed in Pulseq and reconstructed with both BJORK and a conventional NUFFT. Since the output of the scanner after running the .seq file is only a raw k-space, the data needed additional processing before the reconstruction can start. For single-slice TSE sequences, the raw data is ordered exactly in the same way as it is acquired, so first the echos from the first train, then the echos from the second train, etc. For the multi-slice sequence this was different. First the first train of slice 1 was acquired, then the first train of slice 2, etc. As a consequence, the data needed to be grouped according to the slice number, before reconstruction with the current models was possible. It is good practice to always check the ordering of the used trajectory and raw k-space by plotting lines consecutively before reconstructing with the present models. Preferable this is also done during sequence design so that improvements can still be made. BJORK reconstruction results of the multi-slice TSE can be seen in Figure 35. There is some mirroring seen in slice 1 and 2: the rectangular block seems to vaguely develop at the bottom of the phantom as well.

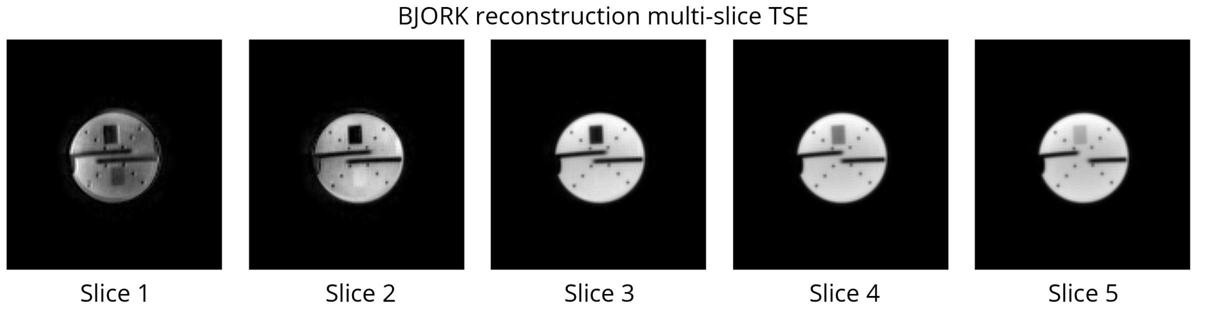


Figure 35: BJORK reconstruction of multi-slice TSE, slices acquired consecutively.

D BJORK implementation

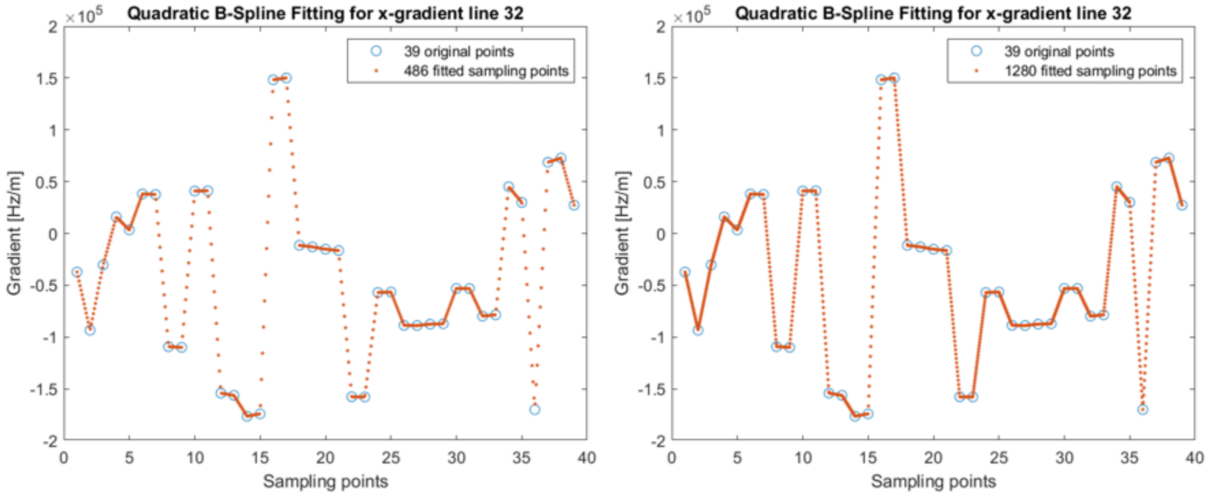
D.1 Set parameters during training in [4] and [46]

	Set parameters during training original [4]	**Set parameters during training in current implementation [46]**
Maximum gradient	26.7 mT/m	23.7 mT/m
Maximum slew rate	150 T/m/s	180 T/m/s
FOV	220 x 220 mm^2	140 x 140 mm^2
Matrix size	320 x 320	320 x 320
No. spokes	32	32
No. sampling points	1280	1280

Table 11: Set parameters for training in [4] versus the set sequence parameters in the implementation of [46] that is used in this work.

D.2 Interpolation of gradient waveforms

Interpolation of the gradient waveforms defined over 39 points was done using a second order (quadratic) spline in Matlab with the functions spapi and fnval. In Figure 36b and Figure 36a, interpolation to 486 and 1280 points can be seen respectively.



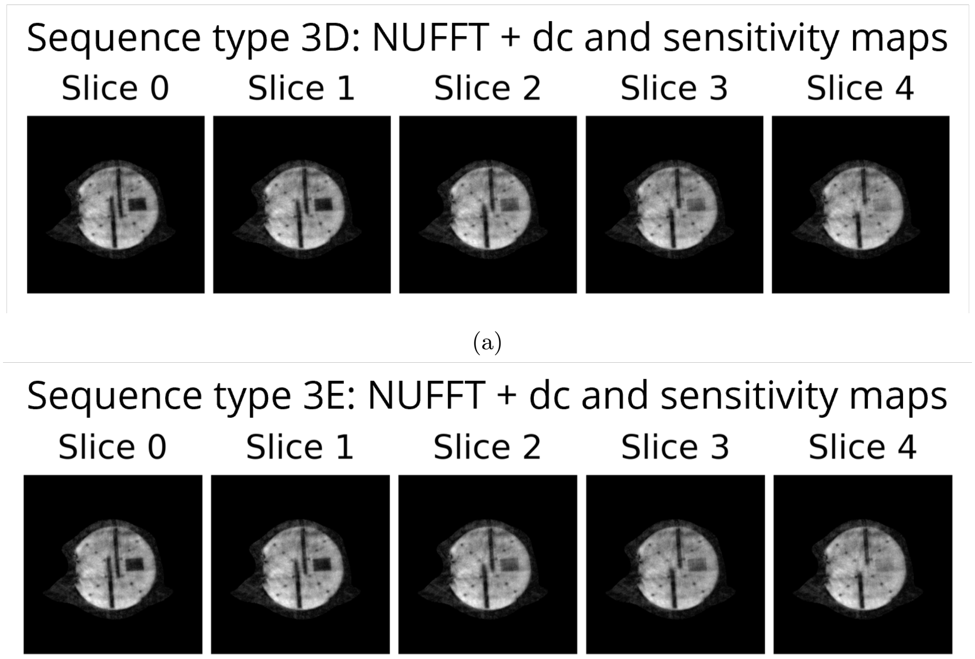
(a)

(b)

Figure 36: Interpolation of gradient waveform of spoke 32 using a quadratic spline fit. The waveform was originally defined over 39 points and interpolated to a) 486 points, adn b) 1280 points.

D.3 BJORK results

Phantom: 3D and 3E Phantom multi-slice results can be seen in Figure 37. In order to examine the slice acquisition scheme, only the NUFFT with dc and sensitivity maps are shown. No difference is observed between either acquisition scheme.


Sequence type 3D: NUFFT + dc and sensitivity maps
Slice 0
Slice 1
Slice 2
Slice 3
Slice 4
Sequence type 3E: NUFFT + dc and sensitivity maps
Slice 0
Slice 1
Slice 2
Slice 3
Slice 4

(a)

(b)

Figure 37: NUFFT with dc and sensitivity maps reconstruction results of multi-slice sequence 3D and 3E, using consecutive or interleaved slice acquisition respectively. No difference is observed between using either acquisition scheme.

***In vivo:* coronal and transversal orientations** In Figure 38, coronal and transversal reconstruction results of sequence 3A-dum$_{10}$ can be seen.

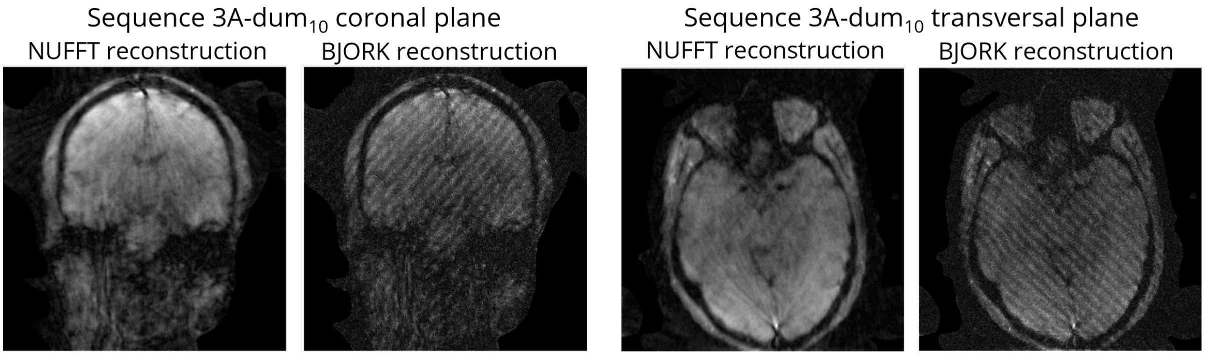


Figure 38: Coronal and transversal planes scanned using sequence 3A-dum$_{10}$. The same streaks that appeared from the top of the head in Figure 26b can be seen here.

***In vivo:* consecutive slice acquisition** Contrary to the multi-slice phantom data in Figure 37, here a difference *was* observed between acquiring the slices consecutively or interleaved. The consecutive acquisition results can be seen in Figure 39, the interleaved in Figure 40. Note how the bottom part of the head is much less well defined in Figure 39 compared to in Figure 40.

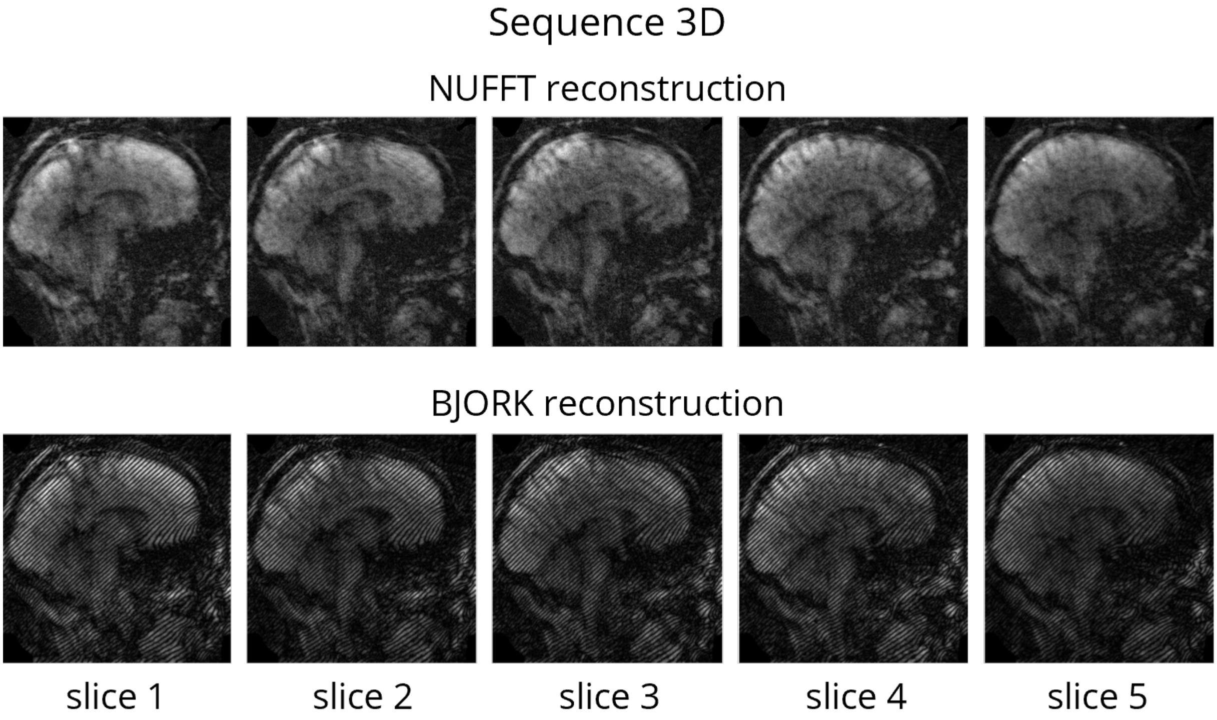


Figure 39: Multi-slice *in vivo*. Consecutive acquisition, so slice order: 1, 2, 3, 4, 5.

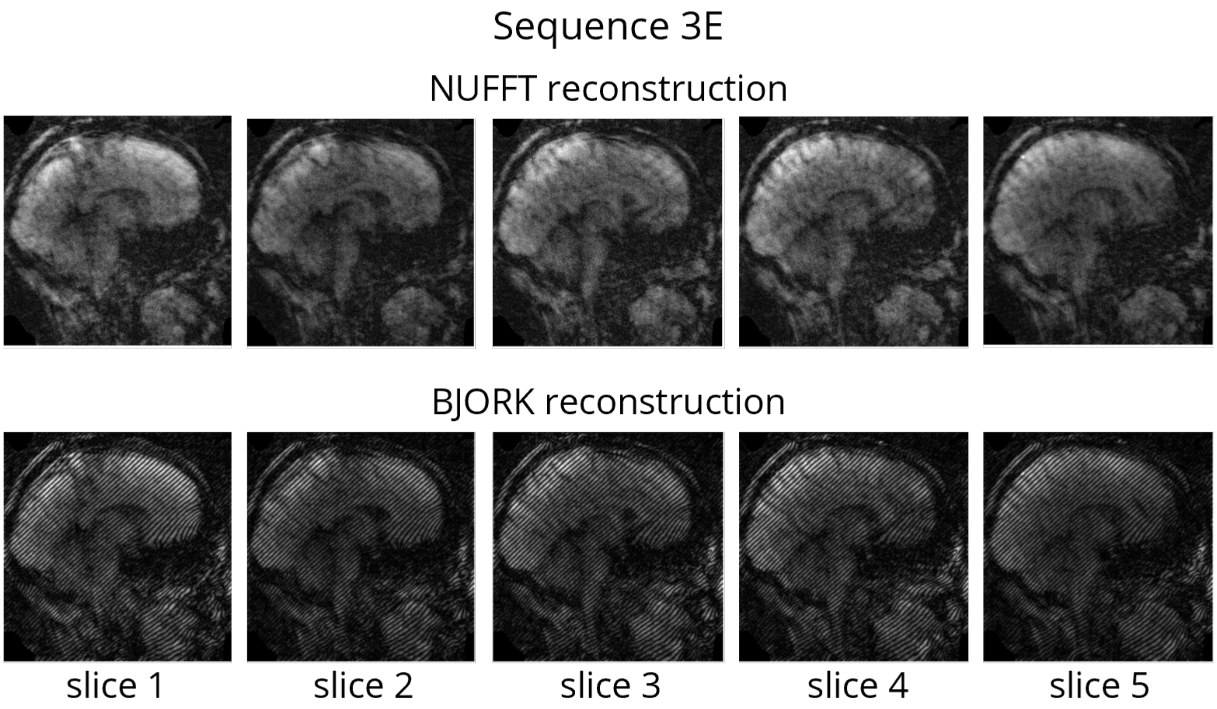


Figure 40: Multi-slice *in vivo*. Interleaved acquisition, so slice order: 1, 3, 5, 2, 4. The bottom part of the head seems to be better defined than in Figure 39.

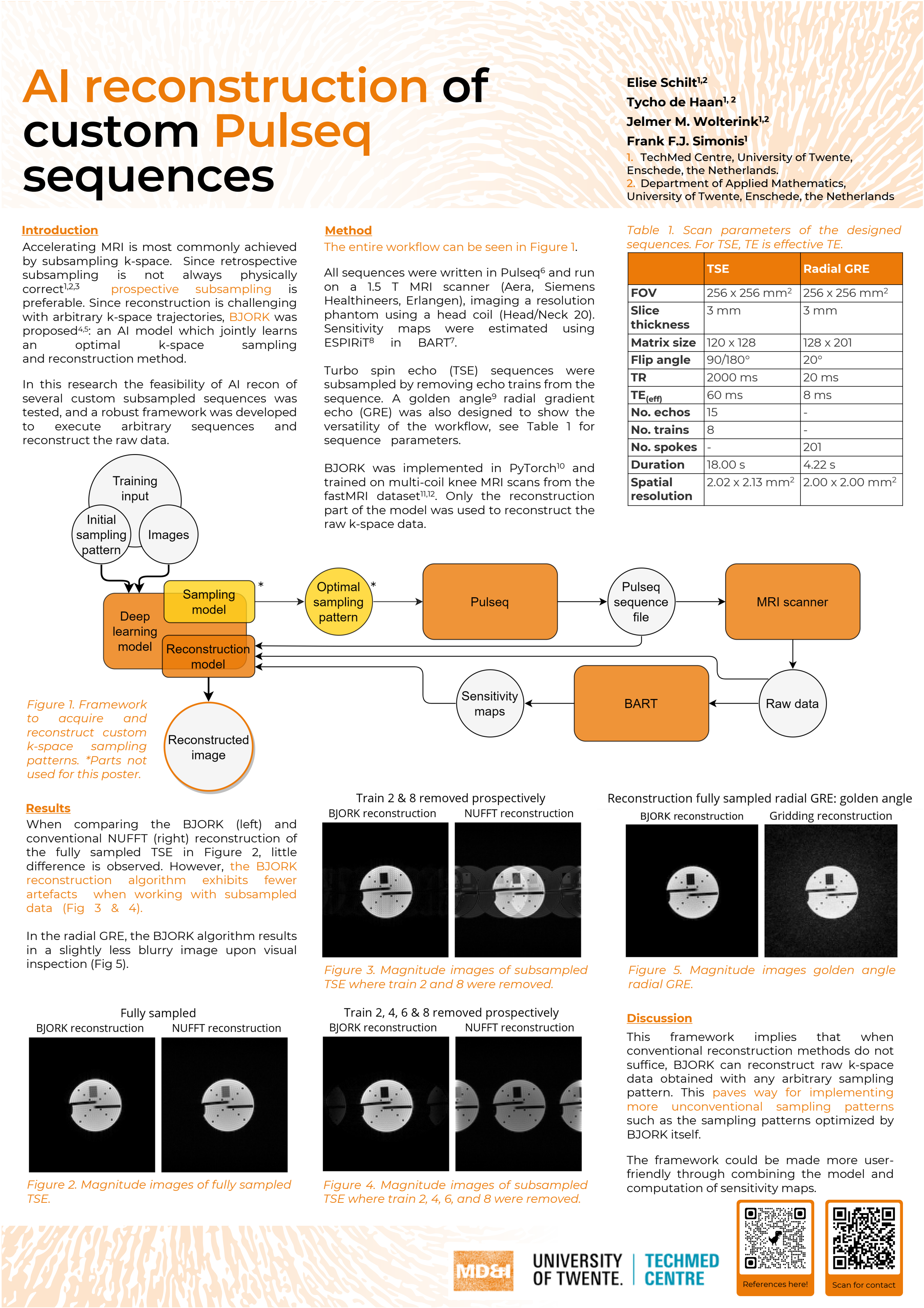
E Code availability

The code in this thesis is used as follows:

1. Load BJORK model, including training script from [46] ⇒ `final_traj.npy`;
2. Optimal sampling pattern → Pulseq file:
   (a) Use `get_grad_for_pulseq.py` to get gradient waveforms ⇒ `opt_grad.mat`;
   (b) Use `python_gradients_to_pulseq.m` with `opt_grad.mat` to make the sequence file ⇒ `BJORK.seq`;
3. Raw MRI data (`data.dat`) → reconstructed image:
   (a) Use MapVBVD [69] to load `data.dat` in Matlab ⇒ `data.mat`
   (b) Compute sensitivity maps using `demo_ESPIRiT_maps_adapted.m` with `data.mat` ⇒ `smaps.mat`;
   (c) Sort and transform `data.mat` and `BJORK.seq` to the correct format using `preprocessing_for_python_model.m` ⇒ `kspace.mat` & `traj.mat`;
   (d) Reconstruct data using `general_recon_clean.py` with `smaps.mat`, `kspace.mat` & `traj.mat` ⇒ reconstructed image!

For access to the code used in this thesis, refer to research groups MD&I or MIA. Small changes were made to some Pulseq files and the ESPIRiT demo. For Pulseq in the functions read.m and calcPNS.m, all indicated with `"% ES: ..."`. Changes in the ESPIRiT demo were related to the size of the reconstructions, and making it robust for when the null space did not span all coils.

F ISMRM poster (accepted for Benelux chapter 2024)



AI reconstruction of custom Pulseq sequences

Elise Schilt[1,2]
Tycho de Haan[1,2]
Jelmer M. Wolterink[1,2]
Frank F.J. Simonis[1]
1. TechMed Centre, University of Twente, Enschede, the Netherlands.
2. Department of Applied Mathematics, University of Twente, Enschede, the Netherlands

Introduction

Accelerating MRI is most commonly achieved by subsampling k-space. Since retrospective subsampling is not always physically correct[1,2,3] prospective subsampling is preferable. Since reconstruction is challenging with arbitrary k-space trajectories, BJORK was proposed[4,5]: an AI model which jointly learns an optimal k-space sampling and reconstruction method.

In this research the feasibility of AI recon of several custom subsampled sequences was tested, and a robust framework was developed to execute arbitrary sequences and reconstruct the raw data.

Method

The entire workflow can be seen in Figure 1.

All sequences were written in Pulseq[6] and run on a 1.5 T MRI scanner (Aera, Siemens Healthineers, Erlangen), imaging a resolution phantom using a head coil (Head/Neck 20). Sensitivity maps were estimated using ESPIRiT[8] in BART[7].

Turbo spin echo (TSE) sequences were subsampled by removing echo trains from the sequence. A golden angle[9] radial gradient echo (GRE) was also designed to show the versatility of the workflow, see Table 1 for sequence parameters.

BJORK was implemented in PyTorch[10] and trained on multi-coil knee MRI scans from the fastMRI dataset[11,12]. Only the reconstruction part of the model was used to reconstruct the raw k-space data.

Table 1. Scan parameters of the designed sequences. For TSE, TE is effective TE.

	TSE	Radial GRE
FOV	256 x 256 mm^2	256 x 256 mm^2
Slice thickness	3 mm	3 mm
Matrix size	120 x 128	128 x 201
Flip angle	90/180°	20°
TR	2000 ms	20 ms
$\text{TE}_{(eff)}$	60 ms	8 ms
No. echos	15	-
No. trains	8	-
No. spokes	-	201
Duration	18.00 s	4.22 s
Spatial resolution	2.02 x 2.13 mm^2	2.00 x 2.00 mm^2

*Figure 1. Framework to acquire and reconstruct custom k-space sampling patterns. *Parts not used for this poster.*

Results

When comparing the BJORK (left) and conventional NUFFT (right) reconstruction of the fully sampled TSE in Figure 2, little difference is observed. However, the BJORK reconstruction algorithm exhibits fewer artefacts when working with subsampled data (Fig 3 & 4).

In the radial GRE, the BJORK algorithm results in a slightly less blurry image upon visual inspection (Fig 5).

Figure 3. Magnitude images of subsampled TSE where train 2 and 8 were removed.

Figure 5. Magnitude images golden angle radial GRE.

Figure 2. Magnitude images of fully sampled TSE.

Figure 4. Magnitude images of subsampled TSE where train 2, 4, 6, and 8 were removed.

Discussion

This framework implies that when conventional reconstruction methods do not suffice, BJORK can reconstruct raw k-space data obtained with any arbitrary sampling pattern. This paves way for implementing more unconventional sampling patterns such as the sampling patterns optimized by BJORK itself.

The framework could be made more user-friendly through combining the model and computation of sensitivity maps.