EEE 473/573 Medical Imaging
Project Report

MRI Simulator & Demonstration of Non-Idealities and Effects of Scanning Parameters

Efe Eren Ceyani – 21903359

Batuhan Uykulu – 21802986

Mert Altunsoy – 22101161

Abstract

In this project, we built an MRI simulator with some non-idealities and examined how different parameters affect the simulation result. We first built an ideal MRI simulator without non-idealities to ensure that our simulation is working. Then, we reckoned some of the non-idealities during the steps of simulation. To emphasis the effects of several parameters, we also examined few of the MRI artifacts which emerge from scanning parameters. In overall, simulator is capable of replicating the imaging process on a three-dimensional brain dataset, and able to demonstrate technical details.

YouTube link: https://www.youtube.com/watch?v=-cDhkx6QyEw/

Introduction

Magnetic resonance imaging (MRI) utilizes the physical phenomenon of nuclear magnetic resonance (NMR), in which particles' spin motion is disturbed by a magnetic field and particles respond with a signal of characteristic frequency which depends on the type of the particle, to produce a tomographic image. In comparison to other imaging techniques, MRI has great soft tissue contrast. In general, target particle of MRI is hydrogen because hydrogen atoms are encountered a lot in biological organisms, which are common subjects for MRI.

In order to improve on the results that we have covered in the class, we built the MRI simulator with flexible parameters so that some of the non-ideal conditions or artifacts can be simulated and the effects of several parameters can be observed.

Methods

Dataset

We have used the Brain Web dataset for brain model which is used as the imaging subject[1]. The dataset includes T_1, T_2, and proton density values for each of the tissues in the brain model. In the model there are 10 tissue labels in total: background, cerebrospinal fluid (CSF), grey matter, white matter, fat, muscle/skin, skin, skull, glial matter, meat (connective)[2]. There are two discrete brain models, which are the normal brain and brain with MS legions. We preferred the normal brain, but the simulations can also be performed on the other dataset. Also, to simulate B_0 inhomogeneity, instead of simulating the inhomogeneous field by ourselves, we found a realistic inhomogeneous field matrix from Brain Web dataset[3]. There are three inhomogeneous magnetic fields for T_1, T_2, and proton density contrasts, so we included B_0 inhomogeneity to our simulation with respect to the contrast mechanism used.

Simulation

To create a proper MRI simulation, we had to implement the three main processes in MRI, which are slice selection, k-space acquisition, and image reconstruction, in order.

Slice selection is essential in MRI because viewing tomographic images of the human body is important for medicine. To implement slice selection, we must only excite a certain part of the brain (or any other subject organ), so that the only signal we receive is coming from the target area. In order to achieve this, we must use gradient fields so that the spatial position is encoded in the signal.

In the simulation, we have used rectangular pulses, which are not realizable in real life due to the infinite spectral bandwidth of the rectangular pulse. The corresponding flip angle with respect to the spatial position is given as the following:

$$\alpha(z) = \gamma A rect\left(\frac{z - \bar{z}}{\Delta z}\right) \quad (1)$$

where γ is the gyromagnetic ratio, $\bar{z}$ is the center slice, and Δz is the slice thickness. A must be chosen carefully such that the desired flip angle is achieved. Using this RF pulse, we will only excite the slices which are around the target slice center. We used a tip angle of $\frac{\pi}{2}$ for all simulations.

[1] https://brainweb.bic.mni.mcgill.ca/anatomic_normal.html
[2] https://brainweb.bic.mni.mcgill.ca/tissue_mr_parameters.txt
[3] https://brainweb.bic.mni.mcgill.ca/about_sbd.html

K-space acquisition is also essential in MRI because we want to encode spatial information in the effective spin density with frequency-encoding. To simulate MRI, one must always remember that instead of imaging an organ directly, we are imaging its effective spin density because depending on the scanning parameters such as time of repetition (TE) and time of repetition (TR), the resulting image differs a lot. Depending on the application, one might choose different scanning parameters to have useful contrast between target tissues.

In the simulation, we have modeled effective spin density with the following equation:

$$f(x,y) = AM_0 \sin\alpha\, e^{-T_E/T_2(x,y)} \frac{1 - e^{-T_R/T_1(x,y)}}{1 - \cos\alpha\, e^{-T_R/T_1(x,y)}} \tag{2}$$

where $T_1(x,y)$ and $T_2(x,y)$ are tissue maps with T_1 and T_2 values with corresponding locations, T_E and T_R are scanning parameters depending on the contrast mechanism used, and M_0 is the magnitude of the magnetization vector which is parallel with the direction of $\mathbf{B}_0$ and is given by:

$$M_0 = \frac{B_0 \gamma^2 \hbar^2}{4kT} P_D \tag{3}$$

where $\hbar$ is the Planck's constant, k is the Boltzmann's constant, T is the temperature in K (in simulation, we assumed it was 4K because MRI uses superconductive magnets), and P_D is the proton density of the tissue. Proton density is also a source of non-ideality in our simulation because we have acknowledged it spatially.

After we have modeled the effective spin density, we also modeled the received baseband signal, which is essential to cover the k-space. The baseband signal at time t is given by the following equation:

$$s(t) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y) e^{-j2\pi(k_x x + k_y y)}\, dxdy \tag{4}$$

where $k_x = \gamma G_x (t - T_E)/2\pi$, and $k_y = \gamma n \Delta G_y t_y / 2\pi$. t_y is the elapsed time while y-gradient moves to another line in Fourier domain. Equation (4) tells us that relationship between the received signal and the effective spin density is essentially a 2D Fourier transform relation, which consists of frequency and phase encodings. Scanning parameters, T_E and T_R, used for each contrast mechanism can be seen in Figure 1.1[4].

	T_1	T_2	PD
T_E	15ms	100ms	15ms
T_R	600ms	6000ms	6000ms

Figure 1.1 Scanning parameters used for each contrast mechanism.

Since we are dealing with rectilinear data, reconstructing the MRI image from the received baseband signal is a simple task, which is to perform an inverse 2D Fourier transform on the baseband signal. However, if we had a polar k-space data, we would had to interpolate the data to a rectangular grid, then take the 2D Fourier transform of the rectangular grid.

[4] Prince, J. L., & Links, J. M. (2015). Medical Imaging Signals and Systems (2nd ed.). Pearson: 413, 452, 455-463, 472,473.

Results

In this section, we will show the gathered k-space data and reconstructed images and examine how various non-idealities/artifacts and scanning parameters can affect the MRI image output.

Different Contrasts

In MRI, it is possible to change T_E and T_R in order to obtain a useful contrast in the image across various tissues. This is possible because different tissues have different T_1, T_2, and proton densities, which means that different tissues' longitudinal/transversal magnetization recovers/decays in different rates.

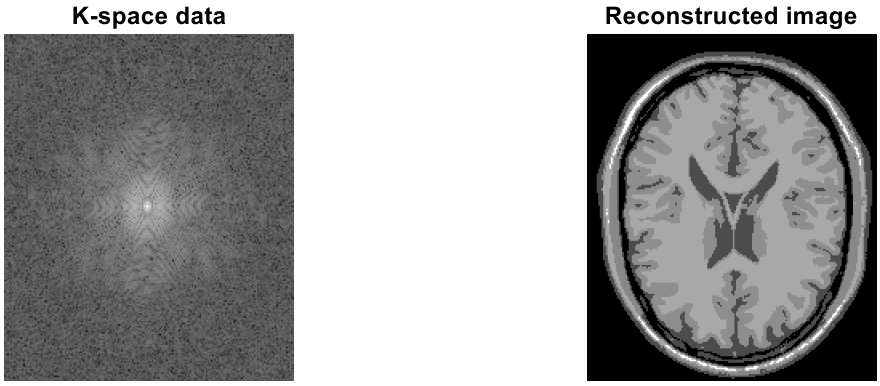

Figure 2.1 K-space data and the respective reconstructed image of the brain, T_1 contrast.

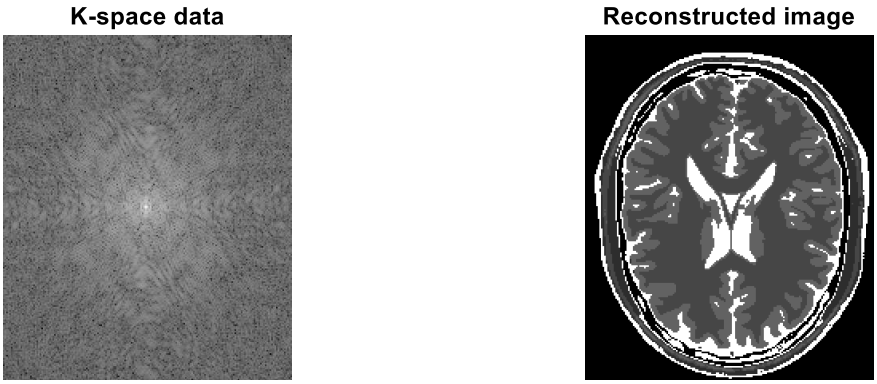

Figure 2.2 K-space data and the respective reconstructed image of the brain, T_2 contrast.

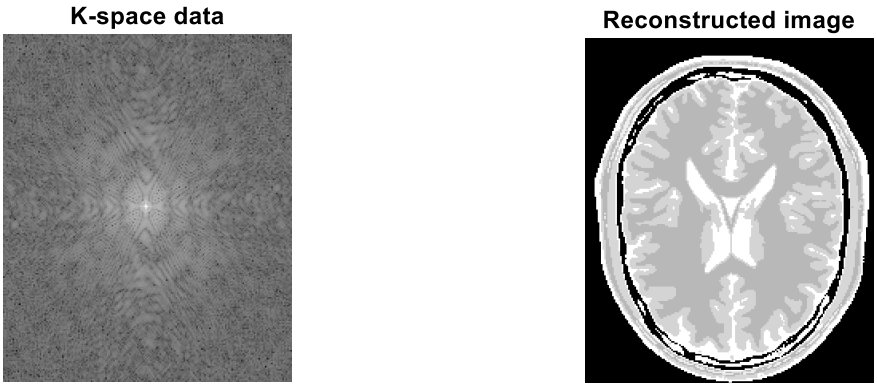

Figure 2.3 K-space data and the respective reconstructed image of the brain, PD contrast.

As it can be seen in Figures 2.1, 2.2, and 2.3, in different contrast mechanisms, tissues look different. Scanning parameters used can be seen in Figure 1.1. In T_1 imaging, CSF is dark, and white matter is brighter than the grey matter. In T_2 imaging, CSF is bright, however, white matter is darker than the grey matter. In PD imaging, CSF is still bright, and white matter is darker than the grey matter.

In all the contrasts, we included the proton density of the tissues while calculating M_0 which can be considered as an extra non-ideality.

T_2 and T_2^*

The transversal magnetization decays faster than the T_2 rate in real-life experiments, which happens due to the inhomogeneities in the magnetic field. T_2^* values of tissues can also be considered as their experimental T_2 values. In spin echo sequences, it is possible to image the "true" T_2 values of the tissues due to the 180° pulse used, however, in gradient echo sequences, additional dephasing elements, which cause T_2^* decay, cannot be eliminated. The difference between T_2 and T_2^* decays can be seen in Figure 2.4. T_2^* image can be seen in Figure 2.5, which can be compared with Figure 2.2.

In this project we are not examining multiple pulse sequences, hence, we will only show the result of reconstructing an image with T_2^* values, which can be seen in Figure 2.5. Scanning parameters used for T_2^* are T_E = 30ms, T_R = 3500ms[5].

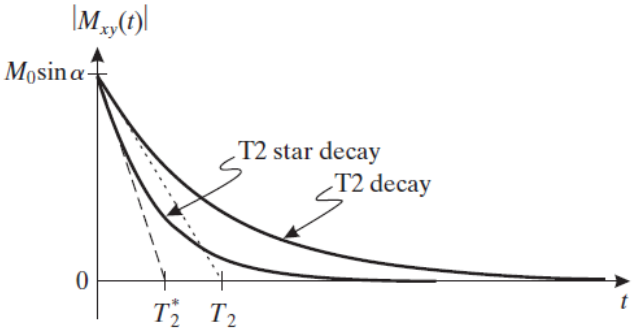

Figure 2.4 Comparison between T_2 and T_2^* rate[6].

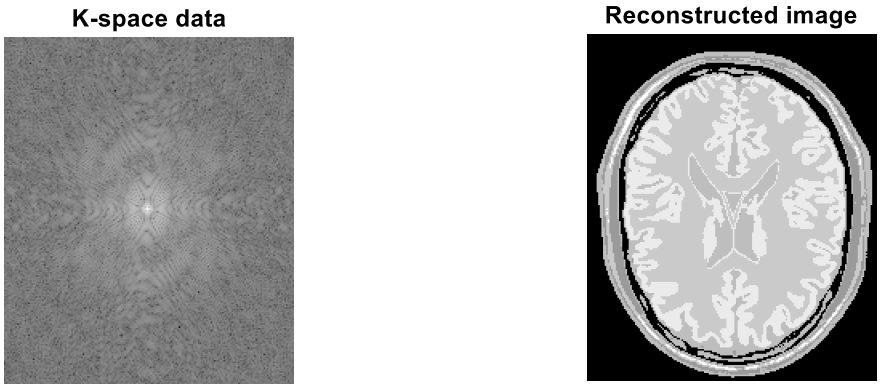

Figure 2.5 K-space data and the respective reconstructed image of the brain, T_2^* contrast.

[5] Chavhan GB et al (2009). Principles, techniques, and applications of T2*-based MR imaging and its special applications. Radiographics, 29:1433-1449.
[6] Prince, J. L., & Links, J. M. (2015). Medical Imaging Signals and Systems (2nd ed.). Pearson: 423.

B_0 inhomogeneity

Homogeneous magnetic fields are important in MRI because even the smallest inhomogeneity can cause difference in the Larmor frequency, which will then affect the reconstructed image. To simulate this effect more realistically, instead of artificially creating a spatially varying field, we have downloaded an inhomogeneous magnetic field data, which was used in practice, from the same dataset we have used for brain models.

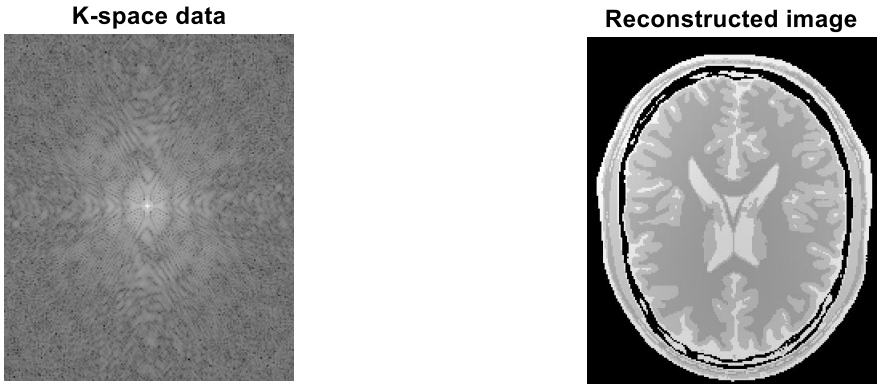

Figure 2.6 K-space data and the respective reconstructed image of the brain, with inhomogeneous magnetic field, PD contrast.

Reconstructed image can be seen in Figure 2.6, which then can be compared with Figure 2.3. The inhomogeneity was not drastic, so due to the digital nature of the simulation it hard to see brightness difference between neighboring tissues, however, it is possible to observe spatially unique differences between Figure 2.3 and Figure 2.6.

Noise

In all electronic circuits, there is always an intrinsic noise called thermal noise in signals. In MRI, the noise can appear both from receiver coils and electrolytes in the human body. The variance of the thermal noise can be characterized as the following:

$$\sigma^2 = \frac{2kTR}{T_A} \tag{5}$$

where k is the Boltzmann's constant, T is the temperature, R is the effective resistance seen by the receiver coils, and T_A is the total data acquisition time. The variance of the noise affects the signal-to-noise ratio (SNR) directly. A crucial point about SNR is that SNR is inversely proportional to image resolution[7].

While the type of the noise can vary, in our simulation, we assumed the simplest model for noise, which is the additive white Gaussian noise (AWGN) model, with a variable variance. We added AWGN to the effective spin density rather than adding it directly to the reconstructed image because receiver coils receive the data from the effective spin density. Ideal and noisy effective spin densities can be seen in Figure 2.7.

[7] Prince, J. L., & Links, J. M. (2015). Medical Imaging Signals and Systems (2nd ed.). Pearson: 479-481.

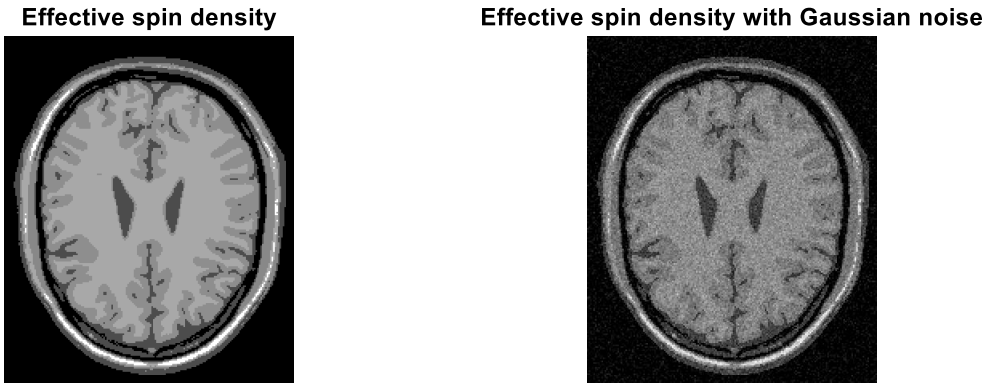

Figure 2.7 Effective spin density of the brain, with and without noise, T_1 contrast.

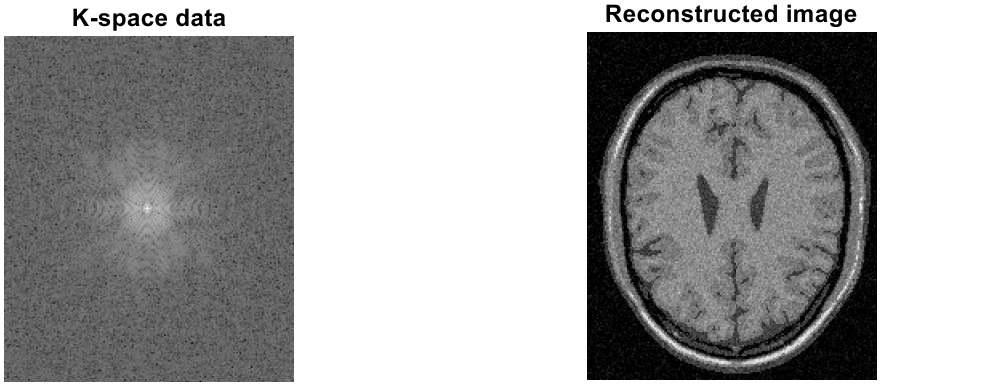

Figure 2.8 K-space data and the respective reconstructed image of the brain, with AWGN, T_1 contrast.

As expected, the reconstructed image also has noise in it, which can be seen in Figure 2.8. To get rid of the noise, we must apply low-pass filter to the k-space data. We applied a rectangular filter to the k-space; however, other shapes can also be used as filters such as circular, elliptical, etc.

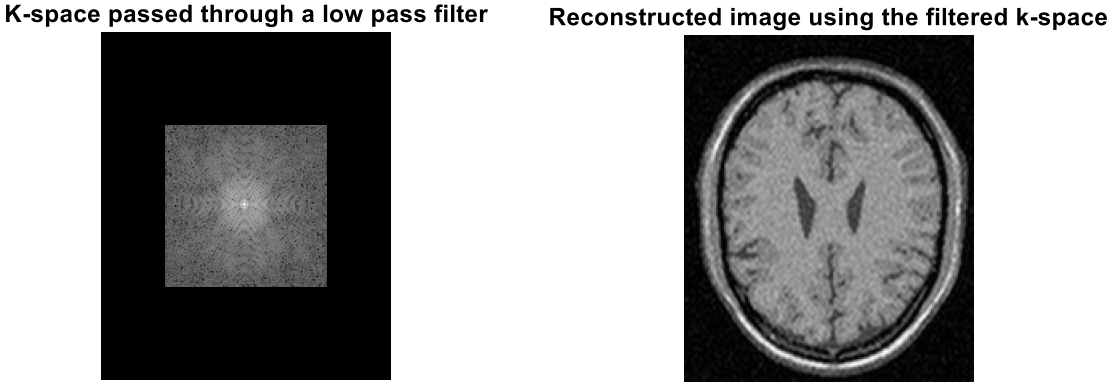

Figure 2.9 K-space data and the respective reconstructed image of the brain, low-pass filtered, T_1 contrast.

As it can be seen in Figure 2.9, after the k-space was filtered, the noise in the reconstructed image was decreased. Figures 2.7 and 2.9 outlines the trade off between resolution and SNR in MRI in a simple yet effective way.

Spike Noise Artifact

Like thermal noise, spike noise can also occur in the receiver coils. Spike noise artifact is a good example to demonstrate the importance of every single element in the k-space. Even an error in one of the k-space elements may alter the reconstructed image immensely.

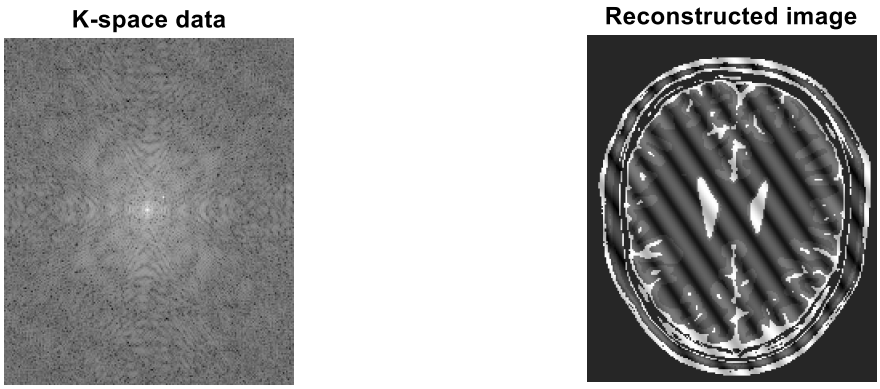

Figure 2.10 K-space data and the respective reconstructed image of the brain, with spike noise artifact, T_2 contrast.

As it can be seen in Figure 2.10, there is a single bright spot in the first quadrant of the k-space, however, its effect on the reconstructed image is visible compared to Figure 2.2. In our simulation, it is possible to add more than one bright spot, and their positions are up to the user.

Wrap Around Artifact

Wrap around artifact is a great example to examine to understand the effects of field of view (FOV) and sampling rate in MRI. The problem occurs when a user desires to view a smaller portion of the image without changing other parameters. If the target object does not exceed the FOV, there will not be any problem in the phase encoding stages, however, if the target object's size exceeds the FOV, the parts exceeding the FOV will be assigned to incorrect spatial positions because phase encoding must encode the image between 0° and 360°, however, the extending parts will not reside in this range.

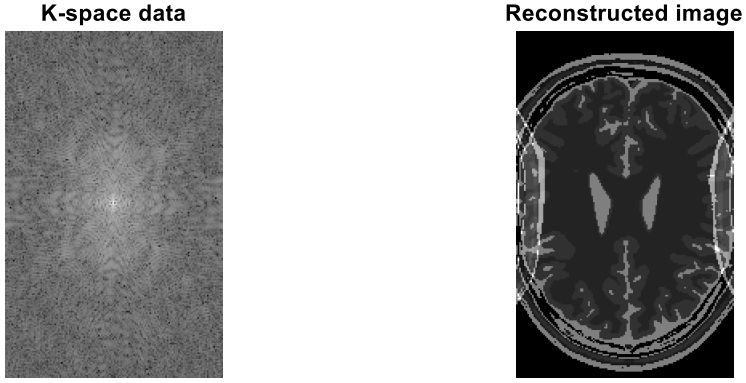

Figure 2.11 K-space data and the respective reconstructed image of the brain, with wrap around artifact, T_2 contrast.

As it can be seen in Figure 2.11, the reconstructed image looks like it has been "wrapped around". However, there is a misconception about the name of this artifact, which the name suggests that the image has been wrapped around like a paper. The left side of the original image will appear on the right side of the new image, and vice versa. This can be verified experimentally by comparing Figures 2.2 and 2.11.

In our simulation, it is possible to change the intensity of this artifact and apply it on the y-axis as well.

Slice Location & Thickness

The ability to choose any slice location with variable thickness is a desired feature in tomographic imaging, which is also true for MRI. In MRI, it is possible to change the center slice and slice thickness by modifying the RF pulse according to equation (1).

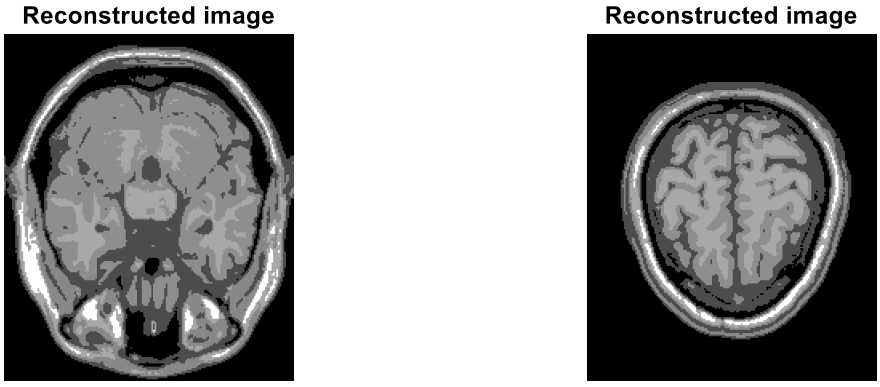

Figure 2.12 Reconstructed images of the brain with different slice locations, T_1 contrast.

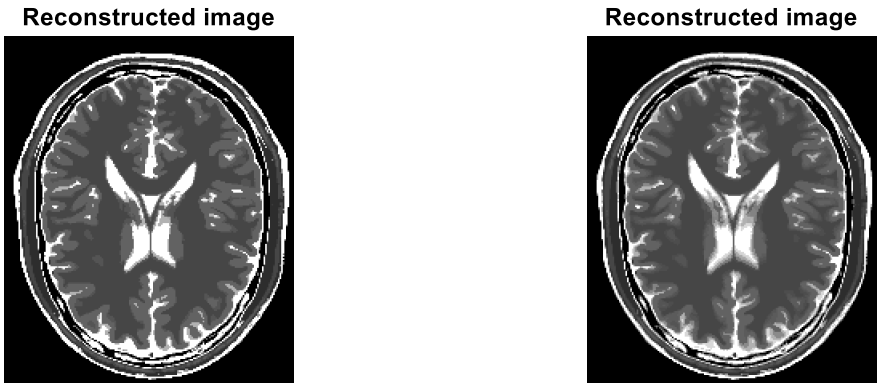

Figure 2.13 Reconstructed images of the brain with different slice thicknesses (3mm and 5mm), T_2 contrast.

As it can be seen in Figure 2.12, it is possible to choose different slices. For example, while the image on the left is on the eye level, one on the right is closer to top of the skull. In other figures than Figure 2.12, we always showed the axial slice of the brain.

In Figure 2.13, it is possible to see the effect of slice thickness in MRI. As slices get thicker, the resolution is lost, and the partial volume effect, in which small tissues become invisible due to the decrease in resolution, is strengthened, however, thicker slices let us to cover more of the object in single scan, which might be useful in some cases.

Discussion

To recap, we have simulated the steps of imaging, considered non-idealities, and shown few of the artifacts emerging from the environment and the scanning parameters. The results align with the theory behind them, and non-idealities/artifacts emphasizes the importance of the scanning parameters. To investigate the effects of some of the parameters, we have included artifacts to understand them visually.

As an improvement, we could have added two more non-idealities: RF pulse with finite bandwidth and chemical shift in fat particles. In the project, we used ideal rectangular pulse for slice selection, however, it is impossible to use RF pulses with infinite bandwidth in real life. We did not consider this in our simulation because the effect of it on the result of the simulation would have been minimal. The other non-ideal case is the chemical shift phenomenon. In chemical shift, fat particles have different shielding constant than water particles, which means that two types of particles experience different effective magnetic fields. This results in a shift in phase, which corresponds to a shift in spatial position of fat particles in the final image[8]. We tried implementing this phenomenon by calculating baseband signals separately for fat and non-fat particles, however, other than change in brightness of fat particles, we were not able to observe any spatial shift.

We were also planning on simulating two types of pulse sequences, which are gradient echo and spin echo sequences; however, we decided on only simulating the k-space acquisition process with an algorithm similar to gradient echo. The reason why we have changed our minds about sequences is that we wanted to focus on the imaging part of MRI and examine the effects of scanning parameters, such as resolution and FOV.

References

https://brainweb.bic.mni.mcgill.ca/

Chavhan GB et al (2009). *Principles, techniques, and applications of T2*-based MR imaging and its special applications*. Radiographics, 29.

Laszlo Balkay (2022). loadminc (https://www.mathworks.com/matlabcentral/fileexchange/32644-loadminc), MATLAB Central File Exchange. Retrieved January 10, 2022.

Nishimura, D. G. (2010). *Principles of Magnetic Resonance Imaging*. Lulu.com.

Prince, J. L., & Links, J. M. (2015). *Medical Imaging Signals and Systems* (2nd ed.). Pearson.

[8] Nishimura, D. G. (2010). Principles of Magnetic Resonance Imaging. Lulu.com: 53-55.

Appendix

MATLAB code for loadminc.m[9] (used for loading the data):

```
function [imaVOL,scaninfo] = loadminc(filename)
%function [imaVOL,scaninfo] = loadminc(filename)
%
% Function to load minc format input file.
% This function use the netcdf MATLAB utility
%
% Matlab library function for MIA_gui utility.
% University of Debrecen, PET Center/LB 2010
if nargin == 0
     [FileName, FilePath] = uigetfile('*.mnc','Select minc
file');
     filename = [FilePath,FileName];
     if FileName == 0;
          imaVOL = [];scaninfo = [];
          return;
     end
end

ncid=netcdf.open(filename,'NC_NOWRITE');
scaninfo.filename = filename;

%[ndims nvars natts dimm] = netcdf.inq(ncid);
%[varname, xtype, dimids, atts] =
netcdf.inqVar(ncid,netcdf.inqVarID(ncid,'xspace'));
% for i=1:atts
%attname =
netcdf.inqattname(ncid,netcdf.inqVarID(ncid,varname),i-1)
%attval =
netcdf.getAtt(ncid,netcdf.inqVarID(ncid,varname),attname)
%end%

pixsizex =
netcdf.getAtt(ncid,netcdf.inqVarID(ncid,'xspace'),'step');
pixsizey =
netcdf.getAtt(ncid,netcdf.inqVarID(ncid,'yspace'),'step');
pixsizez =
netcdf.getAtt(ncid,netcdf.inqVarID(ncid,'zspace'),'step');
x_start =
netcdf.getAtt(ncid,netcdf.inqVarID(ncid,'xspace'),'start')
;
if isempty(x_start)
    x_start = 0;
```

[9] Laszlo Balkay (2022). loadminc (https://www.mathworks.com/matlabcentral/fileexchange/32644-loadminc), MATLAB Central File Exchange. Retrieved January 10, 2022.

```
end
y_start =
netcdf.getAtt(ncid,netcdf.inqVarID(ncid,'yspace'),'start')
;
if isempty(y_start)
    y_start = 0;
end
z_start =
netcdf.getAtt(ncid,netcdf.inqVarID(ncid,'zspace'),'start')
;
if isempty(z_start)
    z_start = 0;
end

scaninfo.pixsize = abs([pixsizex pixsizey pixsizez]); %
abs: Strange could happen
scaninfo.space_start = ([x_start y_start z_start]);

varid = netcdf.inqVarID(ncid,'image-max');
slice_max = netcdf.getVar(ncid,varid,'float');
scaninfo.mag = slice_max;
maxx = max(slice_max(:));
if maxx == round(maxx)
   precision = 'short';
   scaninfo.float = 0;
else
   precision = 'float';
   scaninfo.float = 1;
end


varid = netcdf.inqVarID(ncid,'image');
volume = netcdf.getVar(ncid,varid,precision);

varid = netcdf.inqVarID(ncid,'image-min');
slice_min = netcdf.getVar(ncid,varid,precision);
scaninfo.min = slice_min;
scaninfo.num_of_slice  = size(volume,3);

netcdf.close(ncid);

volume = double(volume);
imsize = size(volume);
% permute the slice image dim. This is for the permut
command in thex for
% loop
imaVOL = zeros(imsize([2,1,3]));
slice_min = double(slice_min);
```

```
slice_max = double(slice_max);

if length(slice_min) >1 % ha minden slice-hoz el van
tárolva a max-min érték
    for i=1: size(volume,3)
        currentslice = volume(:,:,i);
        imaVOL(:,:,i) = permute( ((currentslice -
min(currentslice(:))) / ( max(currentslice(:))-
min(currentslice(:)) )...
            *(slice_max(i)- slice_min(i))) -
slice_min(i),[2 1]);
    end
else
    imaVOL = permute( ( (volume - min(volume(:))) / (
max(volume(:))- min(volume(:)) )*...
        ((slice_max- slice_min))) - slice_min,[2 1 3]);
end
if strcmp(precision,'short')
    imaVOL = int32(imaVOL);
end

scaninfo.imfm = [size(volume,1) size(volume,2)];
scaninfo.Frames = 1;
scaninfo.start_times = [];
scaninfo.tissue_ts = [];
scaninfo.frame_lengths = [];
scaninfo.FileType    = 'mnc';


% tmp
%imaVOL(imaVOL>0.4)=0.4;
```

MATLAB code for project:

```
% EEE 473/573 Medical Imaging Term Project
% MRI Simulator
% Efe Eren Ceyani 21903359
% Batuhan Uykulu 21802986
% Mert Altunsoy 22101161

%% Simulation
%%%%%%%%%%%%%%%%%%%%%%%% Control simulation parameters
from here, then
%%%%%%%%%%%%%%%%%%%%%%%% execute the program.

contrast = "T2"; % Possible choices: "T1", "T2", "PD".

% Non-ideal conditions:
b_inhomogeneity = true;
t2_star = false;

noise = false;
noise_mean = 0;
noise_variance = 10^(-58); % Suggested range for the
variance: 10^(-60) - 10^(-58).
rect_x = 100;
rect_y = 100;

% Artifacts.
spike_noise = false;
spike_noise_array = [100 100];
spike_noise_strength = 10^(-24); % Advised range for
exponential: -25 to -20.

wrap_around = false;
wrap_around_strength_x = 0.75; % Ranges from 0 to 1. 1 -->
Nothing changes |||| close to 0 --> a lot of wrapping.
wrap_around_strength_y = 0.75;
wrap_around_x = true;
wrap_around_y = false;

% Scanning parameters are from the course book.
% Short TE, Medium TR.
T1_TE = 15;
T1_TR = 600;

% Medium TE, Long TR.
T2_TE = 100;
T2_TR = 6000;
```

```
% Short TE, Long TR.
PD_TE = 15;
PD_TR = 6000;

% Scanning parameters for T2_star.
T2_STAR_TE = 30;
T2_STAR_TR = 3500;

% You may change fov's to observe wrap-around artifact but
it is better to
% change it from the settings of wrap_around.
fov_x = 181;
fov_y = 217;
flip_angle = pi/2; % We haven't tried another value other
than pi/2.
slice_center = 90; % Possible choices: 1-181. Default is
90.
slice_thickness = 1; % Possible choices: 1, 3, 5 (unit is
mm).


%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%

%% Initialize Parameters

% General NMR & MRI parameters:
B0 = 1.5; % Tesla.
BOLTZMANN = 1.381*10^(-23); % Joule/Kelvin.
GYRO = 42.58*10^6; % MHz/Tesla, for 1H.
PLANCK = 6.626*10^(-34); % Joule-seconds.
TEMPERATURE = 4; % Kelvin.
SAMPLING_FREQ = 100000; % Hz.

% Normal brain model:
https://brainweb.bic.mni.mcgill.ca/brainweb/anatomic_norma
l.html
% In total there are 10 types of tissues (ranging from 0
to 9)
% In order, these are the brain tissue labels:
% Background, CSF, Grey Matter, White Matter, Fat,
Muscle/Skin, Skin
% Skull, Glial Matter, and Connective.
% For these tissues, T1, T2, T2*, and PD can be found in:
%
https://brainweb.bic.mni.mcgill.ca/brainweb/tissue_mr_para
meters.txt
T1_VAL = [0, 2569, 833, 500, 350, 900, 2569, 0, 833, 500];
```

```
T2_VAL = [0, 329, 83, 70, 70, 47, 329, 0, 83, 70];
T2_STAR_VAL = [0, 58, 69, 61, 58, 30, 58, 0, 69, 61];
PD_VAL = [0, 1, 0.86, 0.77, 1, 1, 1, 0, 0.86, 0.77];

%% Load Data

% Load the data using 'loadminc.m'.
% Written by: Laszlo Balkay
%
https://www.mathworks.com/matlabcentral/fileexchange/32644
-loadminc
[data, scan_info] =
loadminc('phantom_1.0mm_normal_crisp.mnc');

% Initialize contrast maps with the size of the original
data.
brain_map = zeros(size(data));
T1_map = zeros(size(data));
T2_map = zeros(size(data));
PD_map = zeros(size(data));

% Choose imaging contrast.
switch contrast
    case "T1"
        contrast_val = T1_VAL;
        TE_scan = T1_TE;
        TR_scan = T1_TR;
    case "T2"
        contrast_val = T2_VAL;
        TE_scan = T2_TE;
        TR_scan = T2_TR;
    case "PD"
        contrast_val = PD_VAL;
        TE_scan = PD_TE;
        TR_scan = PD_TR;
end

% Create the proper time decay map corresponding to the
chosen contrast.
for tissue_label = 1:10
    brain_map(data==tissue_label-1) =
contrast_val(tissue_label);
    T1_map(data==tissue_label-1) = T1_VAL(tissue_label);
    T2_map(data==tissue_label-1) = T2_VAL(tissue_label);
    PD_map(data==tissue_label-1) = PD_VAL(tissue_label);
end

%% Slice Selection
```

```
% Create ideal rectangular pulse centered around
slice_center, with
% width of slice_thickness, and amplitude of flip_angle.
z_s = linspace(0,size(brain_map, 3),size(brain_map, 3));

rectangular = zeros(size(brain_map, 3));
rectangular(abs(z_s-slice_center)./(slice_thickness) <=
(1/2)) = flip_angle;

% Choose slice using the RF pulse.
% "rectangular" array has the flip angles of each location
and the ones which
% are larger than 0 are selected.
brain_map_sliced = brain_map(:,:,rectangular > 0);
T1_map = T1_map(:,:,rectangular > 0);
T2_map = T2_map(:,:,rectangular > 0);
PD_map = PD_map(:,:,rectangular > 0);

%% K-space Acquisition

% Initialize k-space matrix. Unless the FOV is altered, x-
space will range
% between -90 and 90, and y-space will range between -108
and 108. These
% values come from the size of the dataset.
xspace = linspace(-(size(brain_map, 2)-1)/2,
(size(brain_map, 2)-1)/2, size(brain_map, 2));
yspace = linspace(-(size(brain_map, 1)-1)/2,
(size(brain_map, 1)-1)/2, size(brain_map, 1));
[kx, ky] = meshgrid(xspace, yspace);

% Wrap around artifact.
% Essentially, this statement modifies the FOV value.
if wrap_around
    if wrap_around_x
        fov_x = round(fov_x*wrap_around_strength_x);
    end
    if wrap_around_y
        fov_y = round(fov_y*wrap_around_strength_y);
    end
end

% Now, we can find proper gradient values since we have
FOV values.
% Readout and phase encoding gradient value.
% From lecture note 13, page 18.
G_x_value = SAMPLING_FREQ/(GYRO*fov_x);
```

```
A_y = 1/(GYRO*fov_y);

% T2_star.
% Same procedures were used, so no explanation.
if t2_star
    T2_star_map = zeros(size(data));
    for tissue_label = 1:10
        T2_star_map(data==tissue_label-1) =
T2_STAR_VAL(tissue_label);
    end
    T2_star_map = T2_star_map(:,:,rectangular>0);
    T2_map = T2_star_map;
    if contrast == "T2"
        TE_scan = T2_STAR_TE;
        TR_scan = T2_STAR_TR;
    end
end

% Effective spin density.
if b_inhomogeneity
    % Load the magnetic inhomogeneity model.
    % We had found three different magnetic fields for
each contrast
    % mechanism, however, using any of the fields for any
of the contrast
    % would work fine.
    switch contrast
        case "T1"
            b_matrix = loadminc("t1_inhomogeneity.mnc");
        case "T2"
            b_matrix = loadminc("t2_inhomogeneity.mnc");
        case "PD"
            b_matrix = loadminc("pd_inhomogeneity.mnc");
    end
    % Acquire the correct slice of the inhomogeneous
magnetic field.
    b_matrix = b_matrix(:,:,rectangular>0);
    % Calculate magnetization with inhomogeneous magnetic
field.
    M0 =
((abs(b_matrix).*((2*pi*GYRO)^2)*(PLANCK^2))/(4*BOLTZMANN*
TEMPERATURE)).*PD_map;
    % Calculate effective spin density with respect to the
new
    % magnetization.
```

```
    esd = M0.*sin(flip_angle).*(exp(-
TE_scan./T2_map)).*(1-exp(-TR_scan./T1_map))./(1-
cos(flip_angle)*exp(-TR_scan./T1_map));
else
    % Calculate Magnetization
    M0 =
((B0*((2*pi*GYRO)^2)*(PLANCK^2))/(4*BOLTZMANN*TEMPERATURE)
).*PD_map;
    % Calculate effective spin density.
    esd = M0.*sin(flip_angle).*(exp(-
TE_scan./T2_map)).*(1-exp(-TR_scan./T1_map))./(1-
cos(flip_angle)*exp(-TR_scan./T1_map));
end

% View effective spin density.
figure;
imshow(esd(:,:,1), []);
title("Effective spin density");

% Noise.
if noise
    % Creates noised image with desired mean & variance.
    esd_noise = imnoise(esd, "gaussian", noise_mean,
noise_variance);
    figure;
    imshow(esd_noise(:,:,1), []);
    title("Effective spin density with Gaussian noise");
    % Switch to noisy effective spin density.
    esd = esd_noise;
end

% Initialize baseband array.
baseband = zeros(fov_y, fov_x);

% Acquire baseband signal.
% For every y-space element,
for y = 1:fov_y
    % For every x-space element,
    for x = 1:fov_x
        % Calculate the corresponding baseband signal
coming from the
        % effective spin density.
        baseband(y, x) = sum(sum(esd.*exp(-
1i*2*pi*GYRO*(G_x_value*(kx.*x/SAMPLING_FREQ -
fov_x/(2*SAMPLING_FREQ)) + A_y*ky.*(y-fov_y/2)))));
    end
end
```

```
% Organize the k-space array.
kspace = fftshift(baseband, 2);

% Spike noise artifact.
if spike_noise
    for index = 1:size(spike_noise_array, 1)
        % Update points in the k-space according to the
simulation
        % parameters.
        spike_point_u = spike_noise_array(2*(index-1)+1);
        spike_point_v = spike_noise_array(2*(index-1)+2);
        kspace(spike_point_u, spike_point_v) =
kspace(spike_point_u, spike_point_v) +
spike_noise_strength;
    end
end

% View k-space.
figure;
imshow(log(abs(kspace)), []);
title("K-space data");

%% Image Reconstruction
% Take the inverse 2D-FT of k-space.
image = ifftshift(ifft2(kspace));
figure;
imshow(abs(image), []);
title("Reconstructed image");

if noise
    if wrap_around
        % Rebuild k-space with the updated FOV.
        % This step is essential if the user wants to run
noise and
        % wrap_around at the same time.
        xspace = linspace(-round((fov_x-1)/2),
round((fov_x-1)/2), fov_x);
        yspace = linspace(-round((fov_y-1)/2),
round((fov_y-1)/2), fov_y);
        [kx, ky] = meshgrid(xspace, yspace);
    end
    xdim = abs(xspace) <= rect_x/2;
    ydim = abs(yspace) <= rect_y/2;

    % Create a rectangular filter.
    [xrect, yrect] = meshgrid(xdim, ydim);
    rect_filter = (xrect == 1).*(yrect==1);
```

```
    % Apply filter.
    kspace_filtered = kspace.*rect_filter;

    figure;
    imshow(log(abs(kspace_filtered)), []);
    title("K-space passed through a low pass filter");

    % Recontruct the filtered image
    image_filtered = ifftshift(ifft2(kspace_filtered));

    figure;
    imshow(abs(image_filtered), []);
    title("Reconstructed image using the filtered k-
space");

end
```