Medical Imaging

Prof. Dr. Tobias Knopp

October 17, 2022

Institute für Biomedizinische Bildgebung

Organizational Matters

Team

- Lecture: Tobias Knopp (tobias.knopp@tuhh.de)
- Exercise: Martin Möddel (martin.hofmann@tuhh.de)

Disclaimer

- All lecture material is for your private use only. You are not allowed to share it.
- It is not allowed to screen capture live video sessions. You are violating the DSGVO since no student allowed you to do that.

Lecture

- Inverted classroom instead of classical lecture.
- Screencasts will be provided each week.
- If you wish so you can learn by yourself.
- Weekly meeting to answer your questions. Meetings will be either in-person or virtually after weekly announcements. No hybrid meetings are planned.
- No repetition or summary, so please watch the screencast in order to ask questions.
- Meetings will close early if there are no more questions to make up for the time you spend studying at home

Lab Course & Exercises

Exercises will cover numerical problems related to the lecture. It will be your task to solve these problems with the methods provided within the scope of the lecture and lab course. The lab course aims to teach you some core programming paradigms related to medical imaging and scientific programming.

Lab Course & Exercises

- Programming based exercises cover larger topics (e.g inverse problems)
- You usually will have multiple weeks to solve an exercise.
- Exercise to be solved with the Julia programming language.
- Exercises are designed to be solved in groups with 2-4 people, which you set up in self organization in Stud.IP.
- The solution **must** be a single Julia file: **GroupL_ExerciseN.jl** where N is the number of the exercise sheet and L is the group number. No other naming scheme or file extension (like zip) are allowed.
- The solutions are uploaded into the corresponding group folder in Stud.IP.
- You can earn at most 10% bonus for the exam.

Lab Course & Exercises

- You can seek help from each other using the Stud.IP forum.
- You can ask questions regarding the exercise sheets in the weekly meeting taking place within the second part of the lecture time slot where Martin will take over.
- Solutions will be provided after the topic is closed.
- Screencasts will discuss solutions in more detail.

Literature

- Bildgebende Verfahren in der Medizin; O. Dössel; Springer, Berlin, 2016
- The Mathematics of Medical Imaging: A Beginner's Guide; T. Feemann; Springer, 2015

Content

- Foundations
  - Introduction to Medical Imaging
  - Signal Processing
- Inverse Problems
  - Introduction to Inverse Problems
  - Physical Principles of Computed Tomography
  - Analytical Image Reconstruction
- Discrete Inverse Problems
  - Physical Principles of Magnetic Particle Imaging
  - Regularization Techniques
  - Iterative Reconstruction
- Advanced Sampling Methods
  - Physical Principles of Magnetic Resonance Imaging
  - Nonequidistant Fast-Fourier Transform
  - Compressed Sensing
  - Learning-Based Image Reconstruction

Medical Imaging

Prof. Dr. Tobias Knopp

October 17, 2022

Institute für Biomedizinische Bildgebung

Introduction to Medical Imaging

Introduction to Medical Imaging

Radiology

- discipline within medicine
- deals with imaging of the human body for *diagnostic* and *therapeutic* purposes

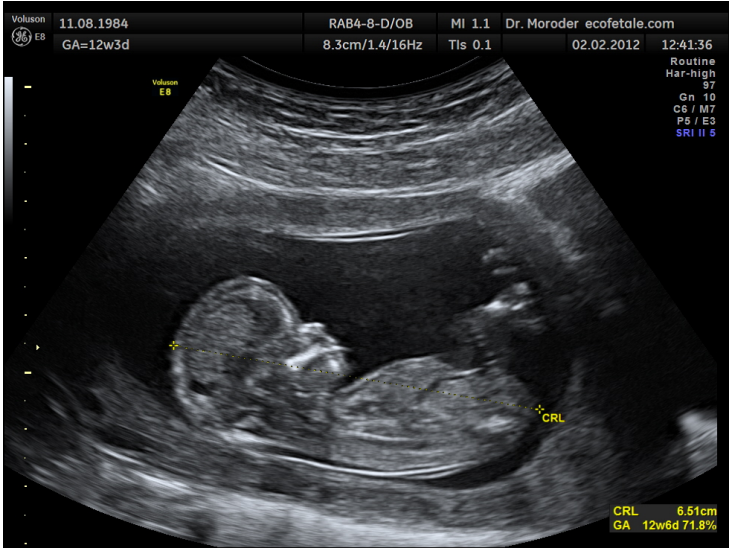
Medical Imaging

- *Make images of the inner human body*
- usually non- or minimally invasive (in contrast to surgical interventions)

Sample Image

Computer Tomograph

Sample Image



Sample Videos

- https://www.youtube.com/watch?v=Wh4aEc4yPh0
- https://www.youtube.com/watch?v=o9EEszvO8PU

Clinically Relevant Imaging Methods

- X-Ray imaging
- computed tomography (CT)
- magnetic resonance imaging (MRI)
- sonography (ultrasound)
- positron emission tomography (PET)
- single-photon emission computed tomography (SPECT)
- magnetic particle imaging (MPI, pre-clinical)
- optical methods (e.g. microscopy, optical coherence tomography (OCT))

Classification of Imaging Methods

- projection vs. tomographic method
- requires radiation or not
- uses a tracer or not

Projection vs. Tomographic Methods

Projection Methods

- Image a line integral through along a certain angle
- Depth information is lost
- Example: **X-ray imaging**

Tomographic Methods

- tomography derived from greek words *tomos* (engl. slice) and graphein (engl. drawing)
- is capable of determining slice images of the inner human body without invasive surgery
- Example: **CT, MRI, PET, SPECT, OCT, Ultrasound, MPI**

Ionizing Radiation

Ionizing radiation is harmful for patient and operator

Ionizing Radiation
X-ray, CT, PET, SPECT

Free of Ionizing Radiation
MRI, MPI, Ultrasound and optical methods (OCT)

Tracer

Some imaging techniques use tracer material that is injected prior to an examination

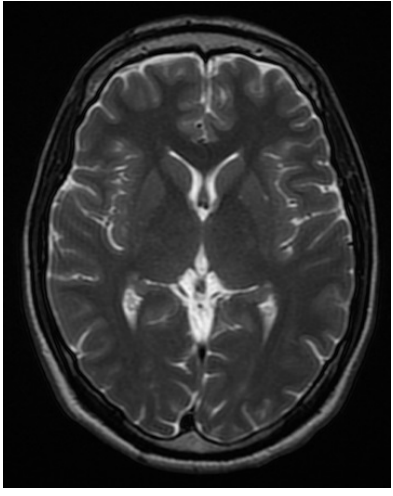
Tracer Based
PET, SPECT, MPI

Works Without a Tracer
MRI, X-ray, CT, Ultrasound, and optical methods (OCT)

However, in **X-ray, CT**, and **MRI** tracer is also used as **contrast agent**

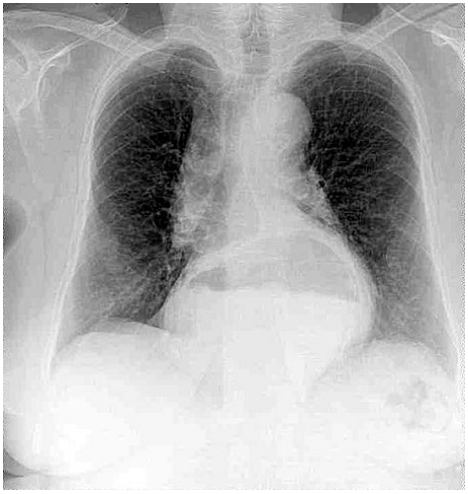
Tracer Example

Sample Image



$\Rightarrow$ Magnetic resonance imaging (MRI)

Sample Image

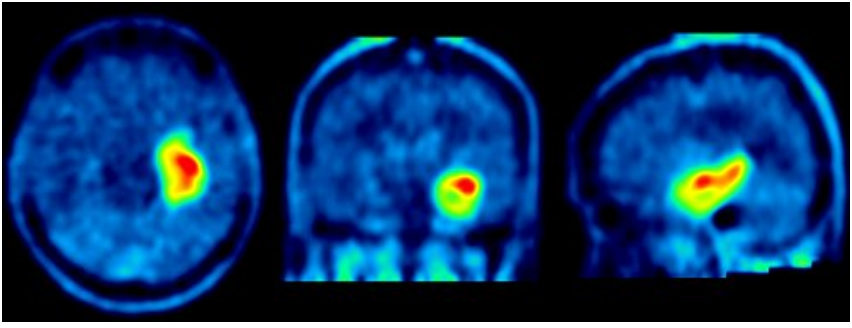


$\Rightarrow$ X-ray

Sample Image

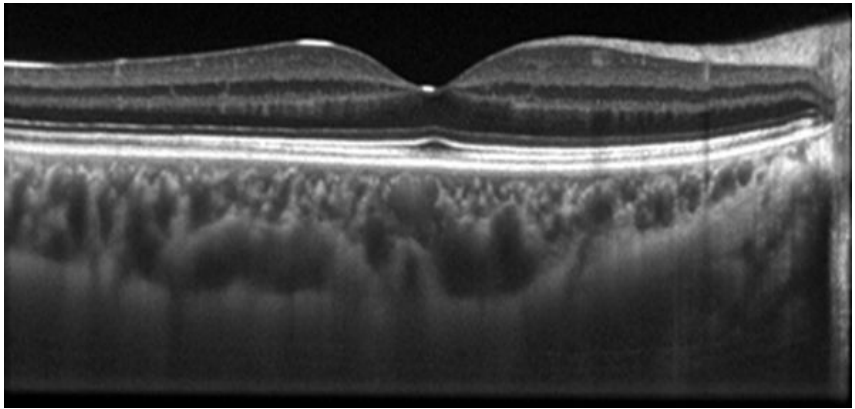
$\Rightarrow$ Computed tomography (CT)

Sample Image



$\Rightarrow$ Positron emission tomography (PET)

Sample Image



$\Rightarrow$ Optical coherence tomography (OCT)

Tomographic Imaging

Goal
Make slice or volume images from 3D objects without the need to actually cut the object.

Tissue is not imaged *directly* but instead a certain physical parameter I is imaged

Example
MRI: density of hydrogen CT: absorption coefficient for X-rays.

Mathematically

$$I : \mathbb{R}^4 \to \mathbb{R}, \quad I(x, y, z, t).$$

Very often imaging is restricted to static objects ($I(x, y, z)$) or 2D planes ($I(x, y)$) through an object.

Remark

- For some imaging modalities (e.g. CT) I carries some physical quantity and in turn has a physical unit.
- For the mathematical treatment we will usually omit the units

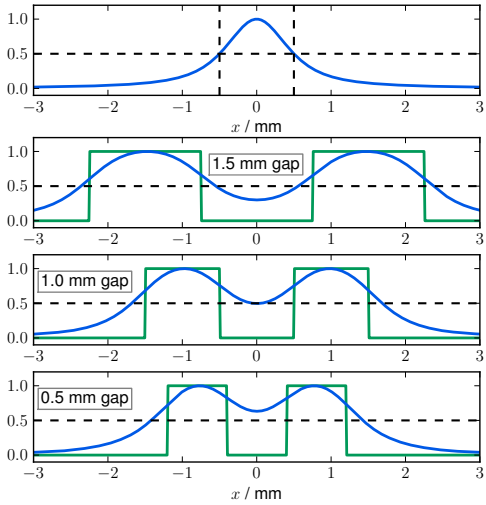
Imaging Performance

The following metrics are used to compare imaging methods

- Image contrast
- Spatial resolution
- Temporal resolution
- Sensitivity

Spatial Resolution

Resolution defines the ability of a system to distinguish two dots.



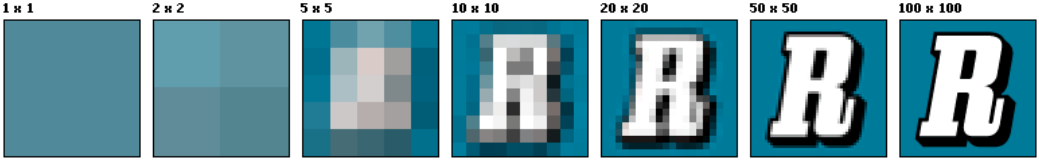
Distinguished is usually defined as a property that holds true if the signal value at the gap is less than 0.5 of the value in the dots.

Spatial Resolution

Notes

- Resolution can be defined on both continuous and discrete signals
- In general: resolution $\neq$ pixel resolution
- Each dimension has its own resolution named after its dimension name: spatial resolution, temporal resolution, spectral resolution

Example



Contrast

- Essential in order to differentiate different structures
- No contrast $\Rightarrow$ no differentiation
- Example: $I_{\text{bone}} = I_{\text{tissue}}$ would be problematic

Definition

$$C = \frac{\max\{I\} - \min\{I\}}{\max\{I\} + \min\{I\}}.$$

Contrast

Example
For instance the function

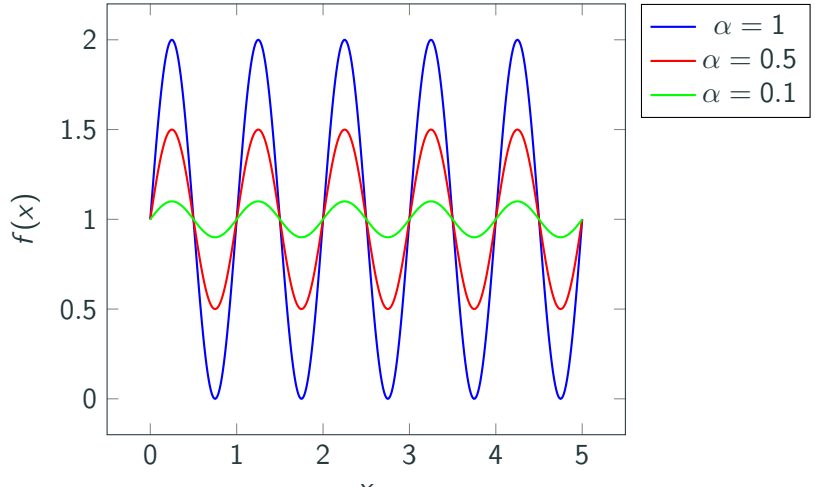
$$f(x) = 1 + \alpha \sin(2\pi x), \quad |\alpha| \leq 1$$

has the contrast

$$C = \frac{1 + \alpha - (1 - \alpha)}{1 + \alpha + 1 - \alpha} = \frac{2\alpha}{2} = \alpha.$$

Contrast

As is shown in the following figure, the function $f(x)$ has high contrast for large α and small contrast for small α.



Signal to Noise Ratio (SNR)

Defines how noisy a signal is. The SNR is formally a property of a single random variable (with mean or expected value μ and standard deviation σ) and defined as

$$\text{SNR} = \frac{\mu}{\sigma}$$

For (dimensional) signals one can define a global SNR. To this end one can take a mean value in the numerator and the standard deviation in a signal-free region as the denominator.

Medical Imaging

Prof. Dr. Tobias Knopp

October 17, 2022

Institute für Biomedizinische Bildgebung

Color Mapping

Color Mapping

- Tomographic image $I(x, y)$ carries real valued information (e.g. 3.9343)
- How to display it on the computer screen

$\Rightarrow$ Map real value to color

Color Mapping

Gray colormap
A gray color c is usually represented in number form as an element of $[0, 1]$ where 0 is the color **black** and 1 is the color **white**. In-between all shades of gray are defined.

General colormap
A general color c is usually represented as an RGB tuple $c = (r, g, b) \in [0, 1]^3$. A **colormap** $f : [0, 1] \to [0, 1]^3$ maps an input value between 0 and 1 to an output color.

Color Mapping

Remark
The colormap is defined on the domain of real numbers in the interval $[0, 1]$. In practice colormaps are build using a set of discrete colors. Using linear interpolation it is possible to define a continuous function based on the discrete values.

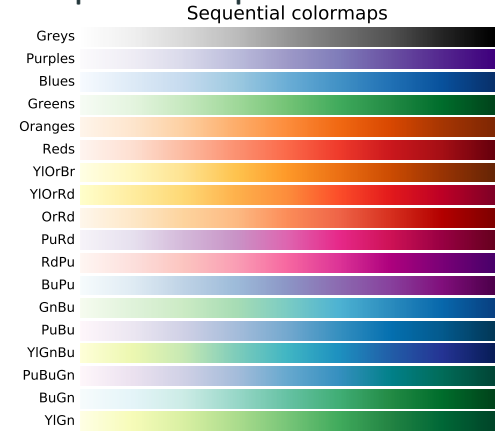
Example
Let $c_k \in [0, 1]^3$ for $k = 1, \ldots, K$ be K colors. Then we can define

$$f(\alpha) = \begin{cases} c_\beta & \text{if } \beta \text{ is an integer} \\ (1 - w)c_{\lfloor \beta \rfloor} + wc_{\lfloor \beta \rfloor + 1} & \text{otherwise} \end{cases}$$

to be the linearly interpolated colormap with $\beta = \alpha(K - 1) + 1$, which is α scaled to $[1, K]$, and weighting $w = \beta - \lfloor \beta \rfloor$.

Color Mapping

Example Colormaps



Color Mapping

Ingredients

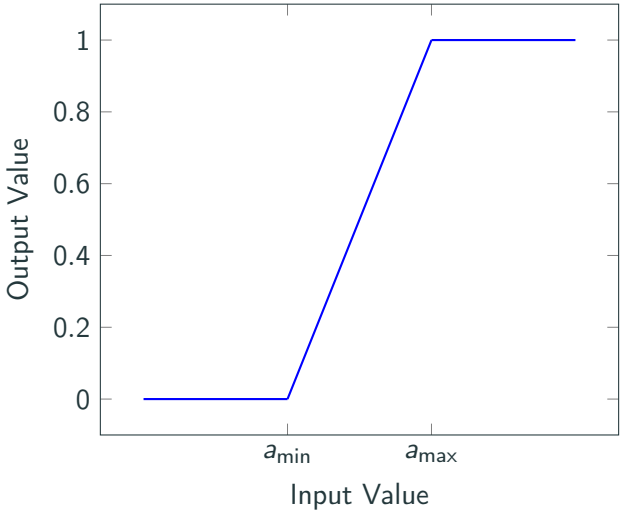
- a colormap $f(\alpha)$
- a minimal value a_{min} that maps to the darkest color $f(0)$
- a maximal value a_{max} that maps to the brightest color $f(1)$

Color Mapping

Windowing
The mapping between real valued quantity I and the color c is called **windowing** and can be expressed by a function

$$g(a) = \begin{cases} 0, & \text{for } a \leq a_{\text{min}} \\ \frac{a - a_{\text{min}}}{a_{\text{max}} - a_{\text{min}}}, & \text{for } a_{\text{min}} < a < a_{\text{max}} \\ 1, & \text{for } a \geq a_{\text{max}} \end{cases} \tag{1}$$

Color Mapping



Color Mapping

Algorithm
For each image pixel at position x, y calculate:

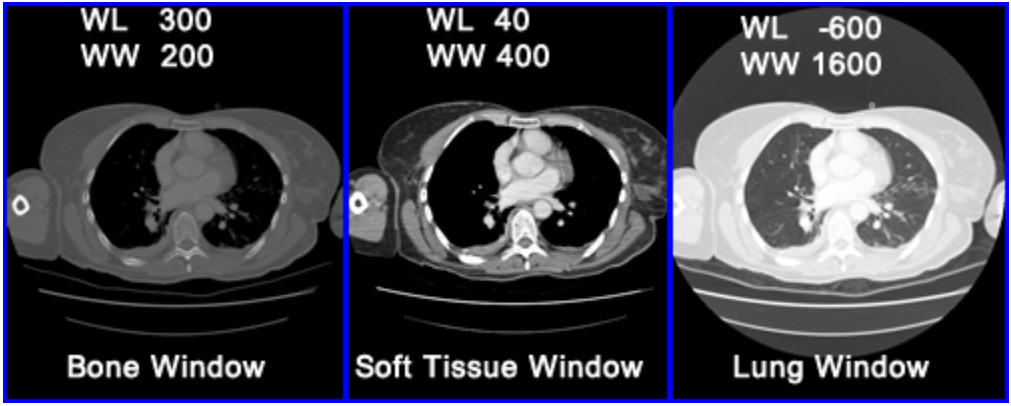
$$I_{\text{colorized}}(x, y) = f(g(I(x, y)))$$

Instead of $a_{\min}$ and $a_{\max}$ it is common to consider instead

$$\text{WW} = a_{\max} - a_{\min} \quad \text{(Window Width or Contrast)}$$
$$\text{WL} = \frac{a_{\max} + a_{\min}}{2} \quad \text{(Window Level or Brightness)}$$

Remark
The human eye can only differentiate a certain number of gray values. It is very common that WW does not span the entire range of image values ($[\min\{I\}, \max\{I\}]$) but WW and WL are adapted to a certain range that the radiologist wants to differentiate. In usual applications there are usually sliders for adjusting WW and WL.

Color Mapping



CT Windows

In CT there one has defined dedicated windows for specific applications

	WL	WW
lung window	-600	1600
bone window	300	2000
soft tissue window	60	360
brain window	40	80
CT angiography window	100	900

Lung CT Example Dataset

- To play around with image contrast parameters you can download the file `lung.tif.zip` from Stud.IP and unzip it.
- Then download the software ImageJ (`https://imagej.nih.gov/ij/`) or use the web-based instance of ImageJ (Run ImageJ in the Browser!)
- Open the TIF image (or the unzipped TIF) in ImageJ and open the menu *Image / Adjust / Window/Level*
- Play around with WW and WL and try to select different parts of the thorax slice (e.g. the lungs, soft tissue, bones).

Medical Imaging

Prof. Dr. Tobias Knopp

October 25, 2022

Institute für Biomedizinische Bildgebung

Signal Processing

Signal Processing

- Understanding tomographic imaging techniques requires profound knowledge of signal processing
- In particular Fourier analysis plays an important role
- Basics of signal processing will be recapitulated

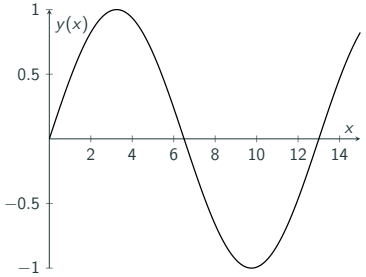
Signal

- A **signal** can be seen as a message that is send from a *sender* to a *recipient*.
- Usually they transport a physical quantity
- Signals can either be discrete or continuous (i.e. a function of $\mathbb{N}$ or $\mathbb{R}$)
- Typically a signal depends on *time* and/or *space*

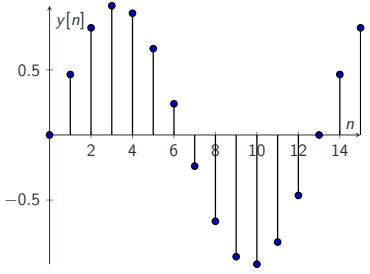
Signal Examples

- $s(x)$: spatial 1D signal, e.g. detector array in CT
- $f(x, y)$: spatial 2D signal, e.g. slice through an object in CT
- $c(t)$: temporal 1D signal, e.g. ECG

continuous signal



discrete signal



Fundamental Signals

- Heaviside step function

$$\text{step}(x) = \begin{cases} 1, & \text{for } x \geq 0 \\ 0, & \text{for } x < 0 \end{cases} \tag{1}$$

- Rectangular function

$$\text{rect}(x) = \begin{cases} 1, & \text{for } |x| < \frac{1}{2} \\ 0.5, & \text{for } |x| = \frac{1}{2} \\ 0, & \text{for } |x| > \frac{1}{2} \end{cases} \tag{2}$$

- Sinc function

$$\text{si}(x) = \text{sinc}(x) = \frac{sin(\pi x)}{\pi x} \tag{3}$$

Fundamental Signals

- Dirac delta distribution

$$\delta(x - x_0) = \begin{cases} 0, & \text{for } x \neq x_0 \\ \infty, & \text{for } x = x_0 \end{cases}$$
$$= \lim_{\tilde{x} \to 0} \frac{1}{\tilde{x}} \text{rect}\left(\frac{x - x_0}{\tilde{x}}\right)$$

- The Dirac delta distribution is no function in a classical sense as ∞ can only be considered in the limes and is not a valid function value.
- The Dirac delta distribution can be defined over the integral

$$\int_{-\infty}^{\infty} \delta(x - x_0) f(x) \, \mathrm{d}x = f(x_0) \tag{4}$$

In particular it holds that

$$\int_{-\infty}^{\infty} \delta(x - x_0) \, \mathrm{d}x = 1 \tag{5}$$

I.e. Dirac delta distribution has unit area.

Systems

- A system $\mathcal{L}$ takes as input a function $f(x)$ and outputs a function $g(x)$

$$g(x) = \mathcal{L}\{f(x)\}$$

- A system is linear if

$$\mathcal{L}\{\sum_i a_i f_i(x)\} = \sum_i a_i \mathcal{L}\{f_i(x)\}$$

- A system is said to be *time invariant* / *shift invariant* if

$$\mathcal{L}\{f(x - x_0)\} = \mathcal{L}\{f\}(x - x_0).$$

  This means a shift of the input signal by x_0 leads to a shift by x_0 of the output signal.

- A system is said to be LTI or LSI if it is linear and time / shift invariant.

Impulse Response and Convolution

Any LSI system can be described by a convolution integral

$$\begin{aligned} g(x) &= \mathcal{L}\{f(x)\} \\ &= \int_{-\infty}^{\infty} f(\tilde{x})h(x-\tilde{x})\,\mathrm{d}\tilde{x} \\ &= (f * h)(x) \end{aligned}$$

h is the so-called *impulse response* or *point-spread function (PSF)* that is obtained by applying a Dirac delta to the LSI system

$$\begin{aligned} \mathcal{L}\{\delta(x)\} &= \int_{-\infty}^{\infty} \delta(\tilde{x})h(x-\tilde{x})\,\mathrm{d}\tilde{x} \\ &\underset{\text{subst. } y=x-\tilde{x}}{=} \int_{-\infty}^{\infty} \delta(x-y)h(y)\,\mathrm{d}y \\ &\underset{\delta \text{ is even}}{=} \int_{-\infty}^{\infty} \delta(y-x)h(y)\,\mathrm{d}y = h(x) \end{aligned}$$

Derivation of the Convolution Integral

Idea: express $f(x)$ as a sum of shifted rectangular functions

$$f(x) = \lim_{x_0 \to 0} \sum_{n=-\infty}^{\infty} f(nX_0)\text{rect}\left(\frac{x - nX_0}{X_0}\right)$$

Applying LSI properties yields

$$\begin{aligned} g(x) = \mathcal{L}\{f(x)\} &= \lim_{X_0 \to 0} \sum_{n=-\infty}^{\infty} f(nX_0)\mathcal{L}\left\{\text{rect}\left(\frac{x - nX_0}{X_0}\right)\right\} \\ &= \int_{-\infty}^{\infty} f(\tilde{x})\mathcal{L}\{\delta(x - \tilde{x})\}\,\mathrm{d}\tilde{x} \\ &= \int_{-\infty}^{\infty} f(\tilde{x})h(x - \tilde{x})\,\mathrm{d}\tilde{x} \end{aligned}$$

Fourier Transformation

It is known from analysis lectures that (almost) any p-periodic function $s(x)$ can be expanded into a *Fourier series*

$$s(x) = \sum_{n=-\infty}^{\infty} c_n \mathrm{e}^{2\pi \mathrm{i} \frac{nx}{p}}$$

consisting of complex sinus functions $\mathrm{e}^{\mathrm{i}x} = \cos x + \mathrm{i} \sin x$. The Fourier coefficients c_n can be calculated by

$$c_n = \frac{1}{p} \int_{-p/2}^{p/2} s(x) \mathrm{e}^{-2\pi \mathrm{i} \frac{nx}{p}} \, \mathrm{d}x$$

From Fourier Series to Fourier Transformation

In order to also express non-periodic functions by Fourier series (i.e. any general signal/function) one can consider the limes $p \to \infty$

$$
\begin{aligned}
s(x) &= \lim_{p\to\infty} \sum_{n=-\infty}^{\infty} c_n \mathrm{e}^{2\pi \mathrm{i} \frac{nx}{p}} \\
&\underset{f:=\frac{n}{p}}{=} \int_{-\infty}^{\infty} S(f) \mathrm{e}^{2\pi \mathrm{i} f x} \,\mathrm{d}f =: \mathcal{F}^{-1}\{S(f)\}
\end{aligned}
$$

Here, $f := \frac{n}{p}$ is the frequency and $S(f) = S(\frac{n}{p}) = c_n$ is a continuous function of frequency. This is in contrast to the discrete spectrum of the periodic Fourier series.

The Fourier transform $S(f)$ of $s(x)$ can be calculated by

$$S(f) = \int_{-\infty}^{\infty} s(x) \mathrm{e}^{-2\pi \mathrm{i} f x} \, \mathrm{d}x =: \mathcal{F}\{s(x)\}$$

$S(f)$ is often called the *spectrum* of $s(x)$. The Fourier relation is often indicated by

$$s(x) \; —\circ S(f)$$

Example

The Fourier transformation of the rectangular function can be calculated to be

$$\begin{aligned}\mathcal{F}\{\mathrm{rect}(x)\} &= \int_{-\infty}^{\infty} \mathrm{rect}(x)\mathrm{e}^{-2\pi \mathrm{i} f x}\,\mathrm{d}x \\ &= \int_{-\frac{1}{2}}^{\frac{1}{2}} \mathrm{e}^{-2\pi \mathrm{i} f x}\,\mathrm{d}x \\ &= \frac{\mathrm{i}}{2\pi f}\left(\mathrm{e}^{-\mathrm{i}\pi f} - \mathrm{e}^{\mathrm{i}\pi f}\right) \\ &= \frac{\sin(\pi f)}{\pi f} = \mathrm{sinc}(f)\end{aligned}$$

In the last step the euler formula $\sin(f) = \frac{\mathrm{e}^{\mathrm{i}f} - \mathrm{e}^{-\mathrm{i}f}}{2\mathrm{i}}$ has been used.

Transfer Function

Instead of describing an LSI system as a convolution with the PSF $h(x)$ one can equivalently describe it in Fourier space by considering the *transfer function* $H(f) = \mathcal{F}\{h(x)\}$.

Theorem
A convolution $g(x) = (s * h)(x)$ in spatial space corresponds to a multiplication in Fourier space:

$$G(f) = S(f)H(f)$$

where $G(f) = \mathcal{F}\{g(x)\}$, $S(f) = \mathcal{F}\{s(x)\}$, and $H(f) = \mathcal{F}\{h(x)\}$.

Proof

$$
\begin{aligned}
G(f) &= \mathcal{F}\{g(x)\} = \mathcal{F}\{(s * h)(x)\} \\
&= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} s(\tilde{x}) h(x - \tilde{x}) \, \mathrm{d}\tilde{x} \mathrm{e}^{-2\pi \mathrm{i} f x} \, \mathrm{d}x \\
&= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} s(\tilde{x}) h(x - \tilde{x}) \mathrm{e}^{-2\pi \mathrm{i} f (x - \tilde{x})} \mathrm{e}^{-2\pi \mathrm{i} f \tilde{x}} \, \mathrm{d}\tilde{x} \, \mathrm{d}x \\
&= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} s(\tilde{x}) h(z) \mathrm{e}^{-2\pi \mathrm{i} f \tilde{x}} \mathrm{e}^{-2\pi \mathrm{i} f z} \, \mathrm{d}\tilde{x} \, \mathrm{d}z \\
&= \int_{-\infty}^{\infty} s(\tilde{x}) \mathrm{e}^{-2\pi \mathrm{i} f \tilde{x}} \, \mathrm{d}\tilde{x} \int_{-\infty}^{\infty} h(z) \mathrm{e}^{-2\pi \mathrm{i} f z} \, \mathrm{d}z \\
&= S(f) H(f)
\end{aligned}
$$

Application of the Convolution theorem

- Efficient application of filter (low pass, high pass)
- Deconvolution / image sharpening

Discrete Fourier transformation

In order to numerically calculate Fourier coefficients one has to sample the Fourier integral a discrete and equidistant sampling points.

This makes the Fourier coefficients periodic so that the data in both domains (spatial and frequency) is discrete (line spectrum) and periodic.

Given a sequence $s_0, \ldots, s_{N-1}$ the discrete Fourier transformation (DFT) is defines as

$$S_m = \sum_{n=0}^{N-1} s_n \mathrm{e}^{-2\pi \mathrm{i} \frac{nm}{N}}, \quad m = 0, \ldots, N-1$$

Discrete Fourier transformation

The DFT is a unitary transformation and its inverse can be calculated by

$$s_n = \frac{1}{N} \sum_{m=0}^{N-1} S_m \mathrm{e}^{2\pi \mathrm{i} \frac{nm}{N}}, \quad n = 0, \ldots, N-1$$

When defining the vectors $\boldsymbol{S} := (S_m)_{m=0}^{N-1}$ and $\boldsymbol{s} := (s_n)_{n=0}^{N-1}$, and the discrete Fourier matrix $\boldsymbol{F} := \left(\mathrm{e}^{-2\pi \mathrm{i} \frac{nm}{N}}\right)_{m=0,\ldots,N-1;n=0,\ldots,N-1}$ the DFT can be written in matrix-vector form

$$\boldsymbol{S} = \boldsymbol{F}\boldsymbol{s}$$

Fast Fourier Transformation

A naive implementation of the DFT would require $\mathcal{O}(N^2)$ arithmetic operations.

The fast Fourier transformation is a fast algorithm capable of carrying out FFT in only $\mathcal{O}(N \log N)$. It uses a recursive divide and conquer principle.

Important is that the FFT can only be applied to equidistant sampling positions.

Fast Fourier Transformation

Assumption: We will derive the DFT for $N = 2^r$.

Basic idea: split the sum into two sums, which are itself regular DFTs.

$$\begin{aligned} S_m &= \sum_{n=0}^{N-1} s_n \mathrm{e}^{-2\pi \mathrm{i} \frac{nm}{N}} \\ &= \sum_{n=0}^{\frac{N}{2}-1} s_n \mathrm{e}^{-2\pi \mathrm{i} \frac{nm}{N}} + \sum_{n=0}^{\frac{N}{2}-1} s_{n+\frac{N}{2}} \mathrm{e}^{-2\pi \mathrm{i} \frac{(n+\frac{N}{2})m}{N}} \end{aligned}$$

Now we will discuss two cases. $m = 2l$ (even) and $m = 2l + 1$ (odd) for $l = 0, \ldots, \frac{N}{2} - 1$.

Case $m = 2l$:

$$
\begin{aligned}
S_{2l} &= \sum_{n=0}^{\frac{N}{2}-1} s_n \mathrm{e}^{-2\pi \mathrm{i} \frac{n2l}{N}} + \sum_{n=0}^{\frac{N}{2}-1} s_{n+\frac{N}{2}} \mathrm{e}^{-2\pi \mathrm{i} \frac{(n+\frac{N}{2})2l}{N}} \\
&= \sum_{n=0}^{\frac{N}{2}-1} s_n \mathrm{e}^{-2\pi \mathrm{i} \frac{nl}{\frac{N}{2}}} + \sum_{n=0}^{\frac{N}{2}-1} s_{n+\frac{N}{2}} \mathrm{e}^{-2\pi \mathrm{i} \frac{nl}{\frac{N}{2}}} \underbrace{\mathrm{e}^{-2\pi \mathrm{i} l}}_{1} \\
&= \underbrace{\sum_{n=0}^{\frac{N}{2}-1} (s_n + s_{n+\frac{N}{2}}) \mathrm{e}^{-2\pi \mathrm{i} \frac{nl}{\frac{N}{2}}}}_{\text{DFT length } \frac{N}{2}}
\end{aligned}
$$

Case $m = 2l + 1$:

$$
\begin{aligned}
S_{2l+1} &= \sum_{n=0}^{\frac{N}{2}-1} s_n \mathrm{e}^{-2\pi \mathrm{i} \frac{n(2l+1)}{N}} + \sum_{n=0}^{\frac{N}{2}-1} s_{n+\frac{N}{2}} \mathrm{e}^{-2\pi \mathrm{i} \frac{(n+\frac{N}{2})(2l+1)}{N}} \\
&= \sum_{n=0}^{\frac{N}{2}-1} s_n \mathrm{e}^{-2\pi \mathrm{i} \frac{nl}{\frac{N}{2}}} \mathrm{e}^{-2\pi \mathrm{i} \frac{n}{N}} + \sum_{n=0}^{\frac{N}{2}-1} s_{n+\frac{N}{2}} \mathrm{e}^{-2\pi \mathrm{i} \frac{nl}{\frac{N}{2}}} \underbrace{\mathrm{e}^{-2\pi \mathrm{i} l}}_{1} \mathrm{e}^{-2\pi \mathrm{i} \frac{n}{N}} \underbrace{\mathrm{e}^{-2\pi \mathrm{i} \frac{1}{2}}}_{-1} \\
&= \sum_{n=0}^{\frac{N}{2}-1} s_n \mathrm{e}^{-2\pi \mathrm{i} \frac{nl}{\frac{N}{2}}} \mathrm{e}^{-2\pi \mathrm{i} \frac{n}{N}} - \sum_{n=0}^{\frac{N}{2}-1} s_{n+\frac{N}{2}} \mathrm{e}^{-2\pi \mathrm{i} \frac{nl}{\frac{N}{2}}} \mathrm{e}^{-2\pi \mathrm{i} \frac{n}{N}} \\
&= \underbrace{\sum_{n=0}^{\frac{N}{2}-1} \mathrm{e}^{-2\pi \mathrm{i} \frac{n}{N}} \left(s_n - s_{n+\frac{N}{2}} \right) \mathrm{e}^{-2\pi \mathrm{i} \frac{nl}{\frac{N}{2}}}}_{\text{DFT length } \frac{N}{2}}
\end{aligned}
$$

Fast Fourier Transformation

Instead of one length N DFT we can apply two length $\frac{N}{2}$ DFTs.

The transformation can be applied again and in the second step we need to apply four length $\frac{N}{4}$ DFTs.

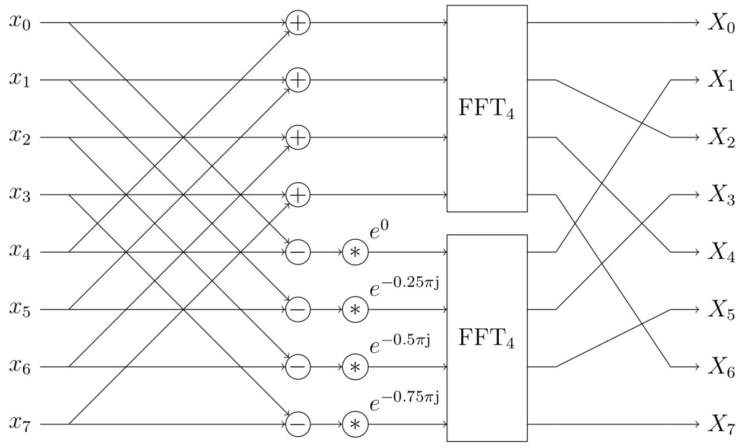
After $r = \log(N)$ steps we need to apply N DFTs of length 1.

In each step the algorithm needs $\frac{N}{2}$ additions, $\frac{N}{2}$ subtractions, and $\frac{N}{2}$ multiplications. The algorithmic complexity it thus $\mathcal{O}(N)$ arithmetic operations.

Since the DFT requires $\log(N)$ steps, the overal time complexity is $\mathcal{O}(N \log(N))$ compared to $\mathcal{O}(N^2)$ of an ordinary DFT.

Fast Fourier Transformation

The FFT is usually implemented inplace and can be visualized as follows



Fast Fourier Transformation

- We have derived the FFT for $N = 2^r$
- The same can be done for other basis 3,5,7,11,13, ...
- By prime factorization one can apply the FFT to general N
- Alternatively, one can pad the vector with zeros to the next power of 2
- The most popular FFT library is the FFTW (Fastest Fourier Transform in the West). It is used in the Julia package FFTW.jl

Medical Imaging

Prof. Dr. Tobias Knopp

November 8, 2022

Institute für Biomedizinische Bildgebung

Inverse Problems

Inverse Problems

In most tomographic imaging methods the task of reconstructing a slice/volume image of the object is an *inverse problem*.

Let I be a multi-dimensional function describing the unknown image, O be a function that describes the *raw measurement data* collected with a tomographic device and S be an operator that maps I to O. Then, the imaging equation for any tomographic imaging method can be written in the form

$$O = S(I). \tag{1}$$

Before we dive into tomography, we discuss the key terminology of inverse problems.

Direct Problem

- Given: The input / cause (i) for a system (S)
- Task: Determine the output of the system

$$o = S(i) \tag{2}$$

Examples:

- Given a current in a electromagnetic coil with a defined geometry. Calculate the magnetic field in space that is generated by the current.
- Given some object within the bore of a tomographic device. Calculate the signals, the device will measure.

Inverse Problem

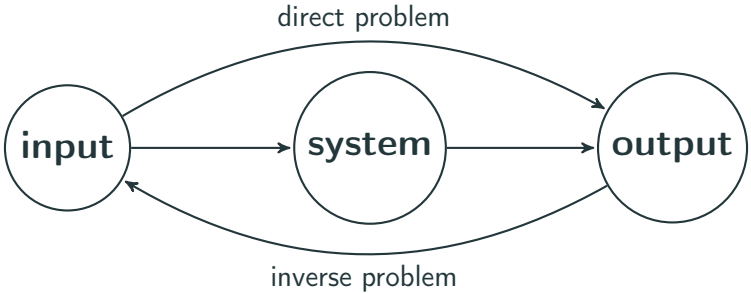
- Given: The output of a system o (i.e. usually some noisy measurements)
- Task: Determine the input to the system i such that

$$S(i) \approx o \tag{3}$$

Examples:

- Given the magnetic field at a finite number of spatial positions. Determine the coil geometry / current that could have been the cause for the observations.
- Given some measurements from a tomographic device. Calculate the object within the scanner bore.

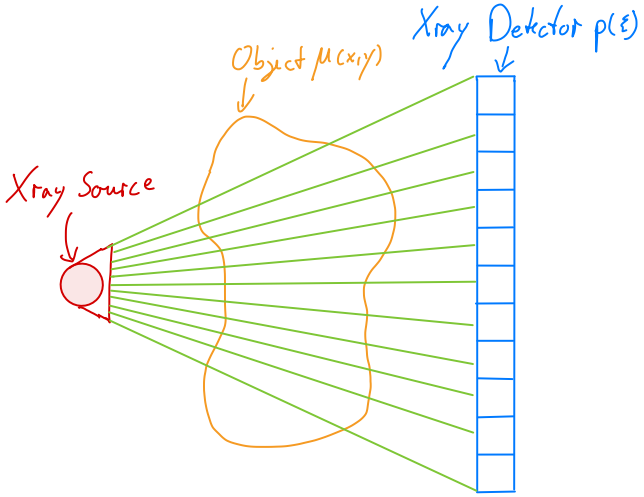
Overview



Radiography

Radiography

During a *radiography* the object under examination is illuminated with X-ray.



Radiography

When the ray passes the object it will be damped/attenuation due to interactions with the matter of the object. In particular the ray is absorbed and scattered. The attenuation coefficient μ is given by

$$\mu = \mu_{\mathsf{S}} + \mu_{\mathsf{A}}$$

where μ_{S} is the scattering coefficient and μ_{A} is the absorption coefficient. The unit of μ is $\frac{1}{m}$. μ is spatially dependent and thus we consider it to be a function $\mu : \mathbb{R}^3 \to \mathbb{R}_+$

Attenuation in Homogeneous Medium

Let $I : \mathbb{R} \to \mathbb{R}_+$ be the intensity of the X-ray. Let it pass along the η axis. Then one observes

$$\begin{aligned} & I(\eta + \Delta\eta) = I(\eta) - \mu \Delta\eta I(\eta) \\ \Leftrightarrow & I(\eta + \Delta\eta) - I(\eta) = -\mu \Delta\eta I(\eta) \\ \Leftrightarrow & \frac{I(\eta + \Delta\eta) - I(\eta)}{\Delta\eta} = -\mu I(\eta) \end{aligned}$$

When considering the limit $\Delta\eta \to 0$ one obtains

$$\lim_{\Delta\eta \to 0} \frac{I(\eta + \Delta\eta) - I(\eta)}{\Delta\eta} = \frac{\mathrm{d}I}{\mathrm{d}\eta} = -\mu I(\eta),$$

which is an ordinary differential equation.

Attenuation in Homogeneous Medium

By separation of variables one obtains

$$\frac{\mathrm{d}I}{I} = -\mu \, \mathrm{d}\eta$$

Integration yields

$$\int \frac{1}{I} \, \mathrm{d}I = \int -\mu \, \mathrm{d}\eta$$

and in turn

$$\ln |I| = -\mu\eta + c.$$

Exponentiation leads to

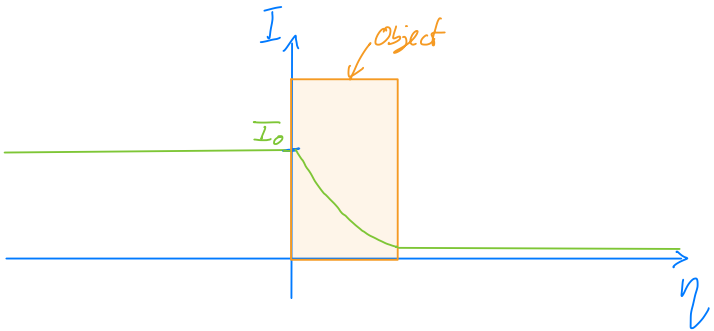
$$I(\eta) = \tilde{c}\mathrm{e}^{-\mu\eta}$$

Attenuation in Homogeneous Medium

Using the initial condition $I(0) = I_0$ which is the X-ray intensity at the source one obtains the **Lambert-Beer law**

$$I(\eta) = I_0 \mathrm{e}^{-\mu\eta}$$

Note that the Lambert-Beer law is only fulfilled for homogeneous media where μ is constant.



Attenuation in Inhomogeneous Medium

In an inhomogeneous medium μ depends on η so that

$$\frac{\mathrm{d}I}{I} = -\mu(\eta)\,\mathrm{d}\eta.$$

Integration leads to

$$\int \frac{1}{I}\,\mathrm{d}I = -\int \mu(\eta)\,\mathrm{d}\eta$$

so that

$$\ln|I| = -\int \mu(\eta)\,\mathrm{d}\eta + c.$$

Attenuation in Inhomogeneous Medium

Exponentiation leads to

$$I(\eta) = \tilde{c}\exp\left(-\int \mu(\eta)\,\mathrm{d}\eta\right).$$

Using the initial condition $I(0) = I_0$ one obtains

$$I(\eta) = I_0\exp\left(-\int \mu(\eta)\,\mathrm{d}\eta\right).$$

Attenuation in Inhomogeneous Medium

We now only consider the intensity at the detector

$$I_D = I(\eta_D) = I_0 \exp\left(-\int_0^{\eta_D} \mu(\eta)\,\mathrm{d}\eta\right)$$

Dividing by I_0 and taking the logarithm leads to

$$\ln(I_D/I_0) = -\int_0^{\eta_D} \mu(\eta)\,\mathrm{d}\eta =: -p$$

Here p is the so-called *projection*.

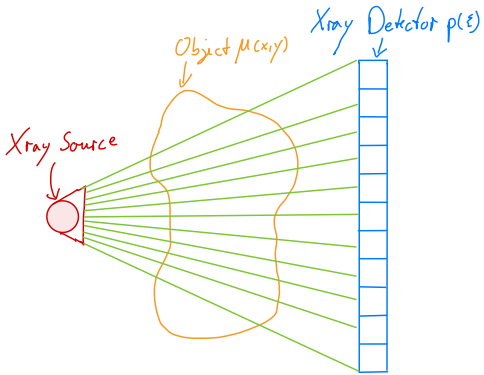
Remarks

- The intensity I_D measured at the detector has always to be related to the intensity I_0 and the X-ray source.
- Typically X-ray data is visualized in the logarithmic form $p = -\ln(I_D/I_0)$.
- In X-ray and CT systems that source intensity can be usually adjusted to generate different contrasts.

Geometries

Until now we have considered a single X-ray passing through the medium and being detected with a single detector pixel.

In practice the source emits the X-ray in a the form of a fan (2D) or a cone (3D).

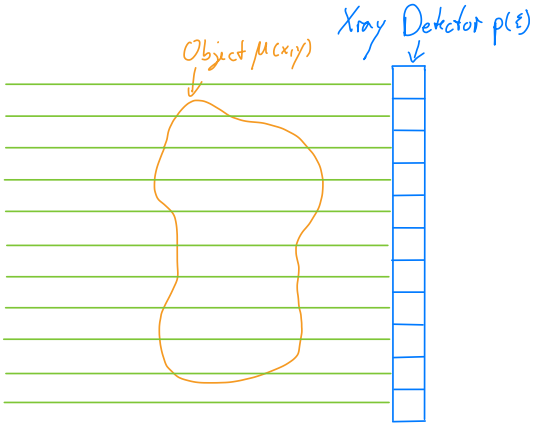


The detected projection value is in the case of a fan beam X-ray source a 1D function $p : \mathbb{R} \to \mathbb{R}$.

In classical radiography cone beam is used and the detector is a 2D function $p : \mathbb{R}^2 \to \mathbb{R}$.

Parallel Beam Geometry

A major simplification that we make from now on is that the X-ray source is moved to $-\infty$ yielding the so-called *parallel beam geometry*.



Radiography as an Inverse Problem

We next consider radiography as an inverse problem. The system equation for the 2D setting in parallel beam geometry reads

$$p(\xi) = \int_0^{\eta_D} \mu(\eta, \xi)\, \mathrm{d}\eta.$$

where $p : \mathbb{R} \to \mathbb{R}_+$ are the measured projections and $\mu : \mathbb{R}^2 \to \mathbb{R}_+$ is the attenuation coefficient.

Direct Problem

The direct problem is easily solvable. After discretization one just has to sum up the values of μ along the beam line.

Inverse Problem

The inverse problem reads: Given p, determine μ. Is that problem solvable?

Existence of a Solution

A solution does exist. For instance a trivial solution is

$$\mu_{\text{trivial}}(\eta, \xi) := \frac{p(\xi)}{\eta_D}$$

since

$$\int_0^{\eta_D} \mu_{\text{trivial}}(\eta, \xi)\, \mathrm{d}\eta = \int_0^{\eta_D} \frac{p(\xi)}{\eta_D}\, \mathrm{d}\eta = \left[\eta \frac{p(\xi)}{\eta_D}\right]_0^{\eta_D} = \eta_D \frac{p(\xi)}{\eta_D} = p(\xi).$$

Uniqueness of a Solution

The existence of a solution is a necessary condition but is that solution unique? Lets consider

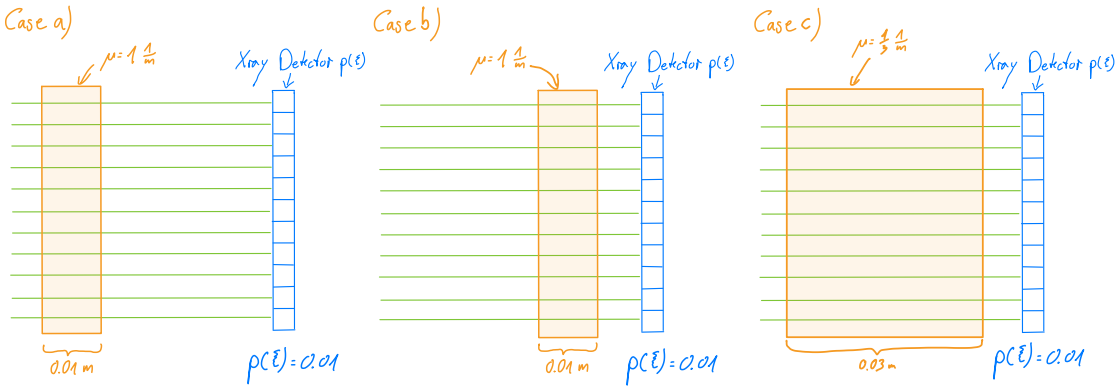
$$\mu_\beta(\eta, \xi) := \begin{cases} \frac{p(\xi)}{\beta} & \eta \leq \beta \\ 0 & \eta > \beta \end{cases}$$

with $\beta \in (0, \eta_D]$, which yields

$$\int_0^{\eta_D} \mu_\beta(\eta, \xi)\, \mathrm{d}\eta = \int_0^{\beta} \frac{p(\xi)}{\beta}\, \mathrm{d}\eta = \left[\eta \frac{p(\xi)}{\beta}\right]_0^{\beta} = \beta \frac{p(\xi)}{\beta} = p(\xi).$$

Thus, the inverse problem has infinite solutions. In practice this means that this particular inverse problem is not solvable, i.e. it is not possible to determine $\mu(\xi, \eta)$ from $p(\xi)$.

Limitations of X-ray imaging



Summary

- Radiography allows to project the attenuation coefficients along a certain direction.
- During this process *depth information* is lost.
- The inverse problem of determining the attenuation coefficients μ from the projections is not solvable. Therefore, in practice, the medical doctor looks at the projection images and tries to *decompose* it by incorporating *prior knowledge* of the underlying anatomy.

Medical Imaging

Prof. Dr. Tobias Knopp

November 15, 2022

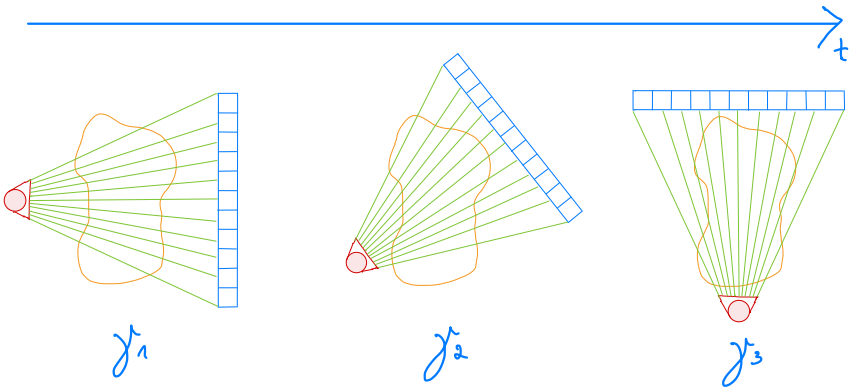
Institut für Biomedizinische Bildgebung

Computed Tomography

Computed Tomography

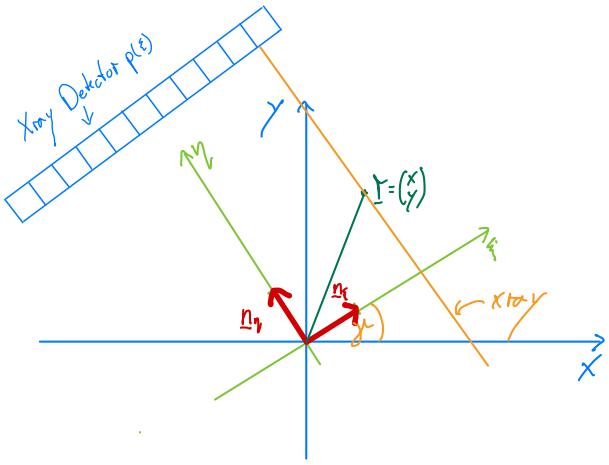
The aim of Computed Tomography is to reconstruct $\mu(x, y)$ from given detector data $p(\xi)$.

Basic idea of CT: Rotate the X-ray source and the detector (the so called gantry) around the object.



Computed Tomography

We differentiate the patient coordinate system (x, y) and the coordinate system of the gantry (ξ, η)



The unit vectors of the (ξ, η) coordinate system are given by

$$\boldsymbol{n}_\xi = \begin{pmatrix} \cos\gamma \\ \sin\gamma \end{pmatrix}$$

$$\boldsymbol{n}_\eta = \begin{pmatrix} -\sin\gamma \\ \cos\gamma \end{pmatrix}$$

Change of basis
One can convert $(x, y) =: \boldsymbol{r}$ coordinates into (ξ, η) coordinates using orthogonal projections

$$\xi = \langle \boldsymbol{r}, \boldsymbol{n}_\xi \rangle = (x, y)^T \begin{pmatrix} \cos\gamma \\ \sin\gamma \end{pmatrix} = x\cos\gamma + y\sin\gamma$$

$$\eta = \langle \boldsymbol{r}, \boldsymbol{n}_\eta \rangle = (x, y)^T \begin{pmatrix} -\sin\gamma \\ \cos\gamma \end{pmatrix} = -x\sin\gamma + y\cos\gamma$$

In matrix-vector form

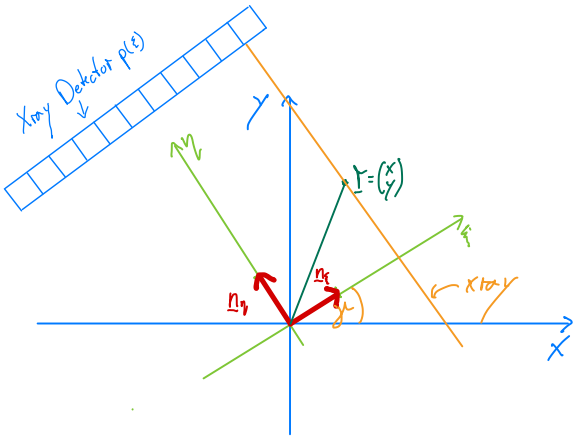
$$\begin{pmatrix} \xi \\ \eta \end{pmatrix} = \begin{pmatrix} \cos\gamma & \sin\gamma \\ -\sin\gamma & \cos\gamma \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \boldsymbol{R}_\gamma \begin{pmatrix} x \\ y \end{pmatrix}$$

Since $\boldsymbol{R}_\gamma$ is orthogonal we have

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \cos\gamma & -\sin\gamma \\ \sin\gamma & \cos\gamma \end{pmatrix} \begin{pmatrix} \xi \\ \eta \end{pmatrix} = \boldsymbol{R}_\gamma^{\mathsf{T}} \begin{pmatrix} \xi \\ \eta \end{pmatrix} = \begin{pmatrix} \xi\cos\gamma - \eta\sin\gamma \\ \xi\sin\gamma + \eta\cos\gamma \end{pmatrix}$$

Imaging Sequence

During a CT measurement the gantry is rotated by 180° or 360° around the patient and the projection data $p(\xi, \gamma)$ are measured. The goal of CT is to reconstruct $\mu(x, y)$ given the projection data $p(\xi, \gamma)$.



Radon Transform

The detector data $p(\xi, \gamma)$ can be calculated via the integration of $\mu(x, y)$ along an X-ray. The X-ray is described by a path $\boldsymbol{\delta}_{\xi,\gamma} : [a, b] \to \mathbb{R}^2$ at position ξ with angle γ and interval boundaries $a \in \mathbb{R}$ at the source and $a < b \in \mathbb{R}$ at the detector. The relation is mathematically described by the Radon transform R:

$$\begin{aligned} p(\xi, \gamma) &= R\{\mu(x, y)\} \\ &= \int_{\boldsymbol{\delta}_{\xi,\gamma}} \mu(x, y) \, \mathrm{d}s \end{aligned}$$

The parametrization of the X-ray is given by

$$\boldsymbol{\delta}_{\xi,\gamma}(\eta) = \boldsymbol{R}_\gamma^\mathsf{T} \begin{pmatrix} \xi \\ \eta \end{pmatrix} = \begin{pmatrix} \xi \cos\gamma - \eta \sin\gamma \\ \xi \sin\gamma + \eta \cos\gamma \end{pmatrix}.$$

Radon Transform

Thus, we can calculate the line integral with

$$\begin{aligned} p(\xi,\gamma) &= R\{\mu(x,y)\} \\ &= \int_{\boldsymbol{\delta}_{\xi,\gamma}} \mu(x,y)\,\mathrm{d}s \\ &= \int_a^b \mu(\boldsymbol{\delta}_{\xi,\gamma}(\eta))\,\left\|\boldsymbol{\delta}'_{\xi,\gamma}(\eta)\right\|_2\,\mathrm{d}\eta \\ &= \int_a^b \mu(\xi\cos\gamma - \eta\sin\gamma, \xi\sin\gamma + \eta\cos\gamma)\,\mathrm{d}\eta \end{aligned}$$

since

$$\|\boldsymbol{\delta}'_{\xi,\gamma}(\eta)\|_2 = \left\| \begin{pmatrix} \sin\gamma \\ \cos\gamma \end{pmatrix} \right\|_2 = 1.$$

Radon Transform

Without loss of generality we can set $a = -\infty$ and $b = \infty$ since $\mu(x, y)$ can be assumed to be zero outside the circle covered by the CT system. Thus, the Radon transform reads

$$p(\xi, \gamma) = \int_{-\infty}^{\infty} \mu(\xi \cos\gamma - \eta \sin\gamma, \xi \sin\gamma + \eta \cos\gamma)\, \mathrm{d}\eta$$

Remark
The Radon transform and the question about its invertability have been investigated by Johann Radon already in 1917, without any concrete application in mind.

Johann Radon: Über die Bestimmung von Funktionen längs gewisser Mannigfaltigkeiten. In: Berichte über die Verhandlungen der Königlich-Sächsischen Gesellschaft der Wissenschaften zu Leipzig. Mathematisch-Physische Klasse. Band 69, 1917, S. 262–277.

Radon Transform as an Inverse Problem

Compare the inverse problems of radiography and CT:

Radiography

$$p(\xi) = \int_{-\infty}^{\infty} \mu(\xi, \eta)\, \mathrm{d}\eta$$

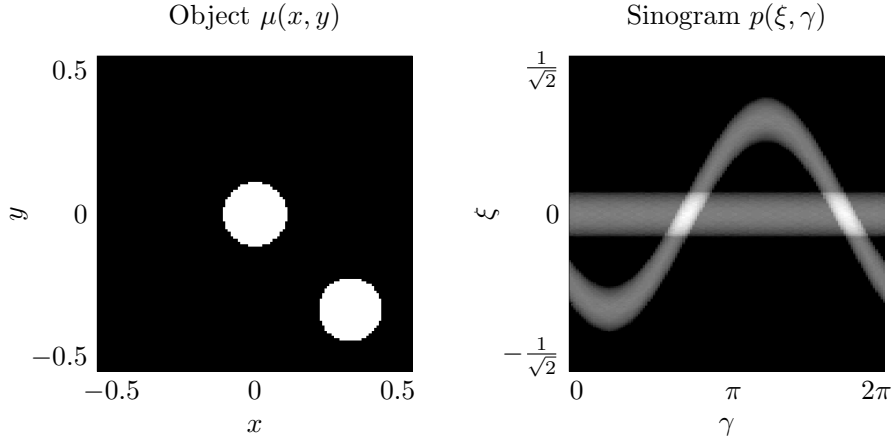
Computed Tomography

$$p(\xi, \gamma) = \int_{-\infty}^{\infty} \mu(\xi \cos\gamma - \eta \sin\gamma, \xi \sin\gamma + \eta \cos\gamma)\, \mathrm{d}\eta$$

Key Observation: The radiography imaging operator maps from a 2D into a 1D space and thus looses information. The CT operator maps from a 2D space into a 2D space and thus might preserve all information.

Sinogram

The raw data $p(\xi, \gamma)$ is also named a sinogram and can be displayed as an image. It is called a sinogram due to the sinus shaped structures.



Fourier Slice Theorem

The Fourier Slice theorem provides answers the fundamental question if it is possible to reconstruct $\mu(x, y)$ from given $p(\xi, \gamma)$, i.e. it shows that the Radon operator is bijective/invertible.

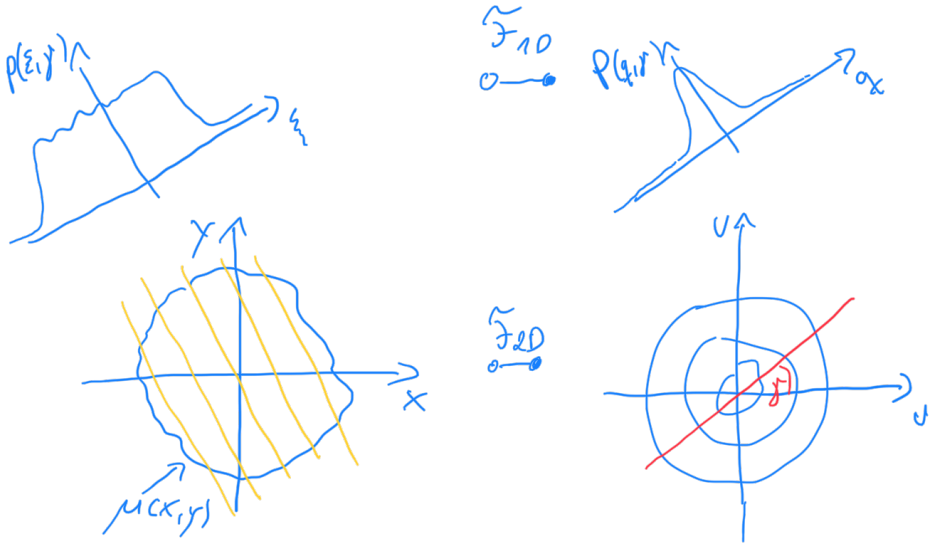
Theorem
Let $P(q, \gamma) := \mathcal{F}_{1D}\{p(\xi, \gamma)\}$ *and* $F(u, v) := \mathcal{F}_{2D}\{\mu(x, y)\}$. *Furthermore let*

$$u = q \cos \gamma$$
$$v = q \sin \gamma$$

Then it holds that

$$F(u, v) = F(q \cos \gamma, q \sin \gamma) = P(q, \gamma)$$

Fourier Slice Theorem



Proof

$$P(q,\gamma) = \int_{-\infty}^{\infty} p(\xi,\gamma)\mathrm{e}^{-2\pi\mathrm{i}\xi q}\,\mathrm{d}\xi$$
$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mu(\xi\cos\gamma - \eta\sin\gamma, \xi\sin\gamma + \eta\cos\gamma)\mathrm{e}^{-2\pi\mathrm{i}\xi q}\,\mathrm{d}\eta\,\mathrm{d}\xi$$

We now make the coordinate transform

$$x = \xi\cos\gamma - \eta\sin\gamma$$
$$y = \xi\sin\gamma + \eta\cos\gamma$$

Using the Jacobian determinant we have

$$\mathrm{d}x\,\mathrm{d}y = \left|\det\frac{\partial(x,y)}{\partial(\xi,\eta)}\right|\,\mathrm{d}\xi\,\mathrm{d}\eta$$
$$= \left|\det\begin{pmatrix}\cos\gamma & -\sin\gamma\\ \sin\gamma & \cos\gamma\end{pmatrix}\right|\,\mathrm{d}\xi\,\mathrm{d}\eta = 1\,\mathrm{d}\xi\,\mathrm{d}\eta$$

Proof

Thus we have

$$\begin{aligned} P(q,\gamma) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mu(x,y)\mathrm{e}^{-2\pi\mathrm{i}(x\cos\gamma+y\sin\gamma)q}\,\mathrm{d}x\,\mathrm{d}y \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mu(x,y)\mathrm{e}^{-2\pi\mathrm{i}(x(q\cos\gamma)+y(q\sin\gamma))}\,\mathrm{d}x\,\mathrm{d}y \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mu(x,y)\mathrm{e}^{-2\pi\mathrm{i}(ux+vy)}\,\mathrm{d}x\,\mathrm{d}y \\ &= F(q\cos\gamma, q\sin\gamma) = F(u,v), \end{aligned}$$

which completes the proof.

Remarks

The Fourier slice theorem answers the question if $\mu(x, y)$ can be reconstructed from $p(\xi, \gamma)$ in the continuous case.

Answer
It can be reconstructed for any $\mu(x, y)$ for which the continuous Fourier transform $F(u, v) := \mathcal{F}_{2D}\{\mu(x, y)\}$ exists. A sufficient criterion for this is that μ is a function of the space $L_1(\mathbb{R}^2)$, i.e. μ has to fulfill

$$\|\mu(x, y)\|_1 := \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} |\mu(x, y)| \, \mathrm{d}x \, \mathrm{d}y < \infty$$

Analytic Image Reconstruction

- The Fourier slice theorem allows to *analytically* solve the inverse problem of determining the tomographic image by direct inversion of the imaging operator.
- This in turn yields a *direct* image reconstruction method.
- Methods that instead tackle the inverse problem in its original form are often named *algebraic image reconstruction* (ART) methods.

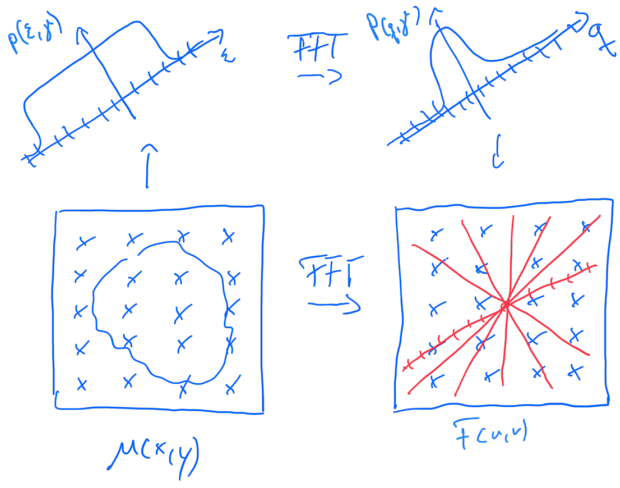
Fourier Based Reconstruction

Using the Fourier slice theorem one can derive the following direct reconstruction algorithm

1. $\forall \gamma$ calculate $P(q, \gamma) = \mathcal{F}_{1D}\{p(\xi, \gamma)\}$
2. $\forall u = q \cos \gamma, v = q \sin \gamma$ calculate $F(u, v) = P(q, \gamma)$
3. calculate $\mu(x, y) := \mathcal{F}_{2D}^{-1}\{F(u, v)\}$

Practical Issue

In the discrete setting the Fourier transforms are realized using the FFT. However, since the FFP is only applicable for **equidistant** node points one has the situation that the points (u, v) and $(q\cos\gamma, v = q\sin\gamma)$ do not match.



Practical Issue

Consequently, the Fourier based reconstruction has to resample the data in Fourier space using e.g. interpolation techniques. Interpolation in Fourier space leads, however, to larger numerical errors in image space.

$\Rightarrow$ Fourier based reconstruction usually not used in todays CT scanners.

Remark
Nowadays fast FFT for **non-equidistant** node points are known. Will be discussed later in the lecture.

Filtered Backprojection

- Standard reconstruction technique used in CT scanners
- Uses Fourier slice theorem (as well)

Derivation

Expressing $\mu(x, y)$ in terms of its inverse Fourier transform $F(u, v)$:

$$\mu(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(u, v) \mathrm{e}^{2\pi \mathrm{i}(ux+vy)} \, \mathrm{d}u \, \mathrm{d}v$$

We now perform a coordinate transform from Cartesian into polar coordinates:

$$u = q \cos \gamma$$
$$v = q \sin \gamma$$

Derivation

The Jacobian determinant is given by

$$
\begin{aligned}
\mathrm{d}u\,\mathrm{d}v &= \left|\det\frac{\partial(u,v)}{\partial(q,\gamma)}\right| \mathrm{d}q\,\mathrm{d}\gamma \\
&= \left|\det\begin{pmatrix} \frac{\partial u}{\partial q} & \frac{\partial v}{\partial q} \\ \frac{\partial u}{\partial \gamma} & \frac{\partial v}{\partial \gamma} \end{pmatrix}\right| \mathrm{d}q\,\mathrm{d}\gamma \\
&= \left|\det\begin{pmatrix} \cos\gamma & \sin\gamma \\ -q\sin\gamma & q\cos\gamma \end{pmatrix}\right| \mathrm{d}q\,\mathrm{d}\gamma \\
&= |q\cos^2\gamma + q\sin^2\gamma|\,\mathrm{d}q\,\mathrm{d}\gamma \\
&= |q|\,\mathrm{d}q\,\mathrm{d}\gamma
\end{aligned}
$$

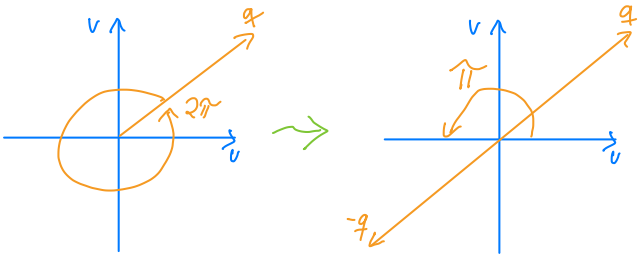
Derivation

Thus, we have

$$\mu(x,y) = \int_0^{2\pi} \int_0^{\infty} F(q\cos\gamma, q\sin\gamma) \mathrm{e}^{2\pi \mathrm{i}(xq\cos\gamma + yq\sin\gamma)} |q| \,\mathrm{d}q\,\mathrm{d}\gamma$$

$$\underset{\text{FS theorem}}{=} \int_0^{2\pi} \int_0^{\infty} P(q,\gamma) \mathrm{e}^{2\pi \mathrm{i}(xq\cos\gamma + yq\sin\gamma)} |q| \,\mathrm{d}q\,\mathrm{d}\gamma$$

Changing the integral limits



Derivation

yields

$$\mu(x,y) = \int_0^{\pi} \int_{-\infty}^{\infty} P(q,\gamma) \mathrm{e}^{2\pi \mathrm{i} q(x\cos\gamma + y\sin\gamma)} |q| \,\mathrm{d}q \,\mathrm{d}\gamma$$

The inner integral can be defined to be a function $h(\xi,\gamma)$:

$$h(\xi,\gamma) = \int_{-\infty}^{\infty} P(q,\gamma)|q| \mathrm{e}^{2\pi \mathrm{i} q\xi} \,\mathrm{d}q$$

$$\mu(x,y) = \int_0^{\pi} h(x\cos\gamma + y\sin\gamma, \gamma) \,\mathrm{d}\gamma$$

Algorithm

With this we now can formulate the filtered backprojection algorithm (input $p(\xi, \gamma)$, output $\mu(x, y)$):

1. $\forall \gamma$ calculate $P(q, \gamma) = \mathcal{F}_{1D}\{p(\xi, \gamma)\} = \int_{-\infty}^{\infty} p(\xi, \gamma) \mathrm{e}^{-2\pi \mathrm{i} q \xi} \, \mathrm{d}\xi$
2. $\forall \gamma$ calculate $h(\xi, \gamma) = \int_{-\infty}^{\infty} P(q, \gamma) |q| \mathrm{e}^{2\pi \mathrm{i} q \xi} \, \mathrm{d}q$
3. calculate $\mu(x, y) = \int_0^{\pi} h(x \cos\gamma + y \sin\gamma, \gamma) \, \mathrm{d}\gamma$

Remarks

- The first two steps of the algorithm apply the filter $|q|$ in Fourier space. $|q|$ is a high pass or edge filter.
- Instead of applying the filter in Fourier space one can alternatively apply it directly in spatial domain.

Issue
What is the Fourier transform of $|q|$?

Fourier transform of $|q|$

Consider:

$$w_\varepsilon(\xi) = \frac{\varepsilon^2 - (2\pi\xi)^2}{(\varepsilon^2 - (2\pi\xi)^2)^2} \quad \circ\!\!-\!\!\bullet \quad |q|\mathrm{e}^{-\varepsilon|q|}$$

In the limit $\varepsilon \to 0$ one obtains

$$|q|\mathrm{e}^{-\varepsilon|q|} \to |q|$$

and

$$w_0(\xi) = -\frac{1}{(2\pi\xi)^2}$$

Fourier transform of $|q|$

$\Rightarrow$ If the Fourier integrals converge (depends on $p(\xi, \gamma)$!) we can apply the filter in image space via a convolution:

$$h(\xi, \gamma) = (p(\tilde{\xi}, \gamma) * w_0(\tilde{\xi}))(\xi)$$

Since $w_0(\xi)$ has "local" support (after truncation), the convolution can be effectively applied in image space. This has been done in first generation CTs.

Backprojection

Lets have a look at the inner part of the filtered backprojection, i.e. the integration

$$f(x, y) = \int_0^{\pi} h(x \cos\gamma + y \sin\gamma, \gamma)\, \mathrm{d}\gamma$$

Here,

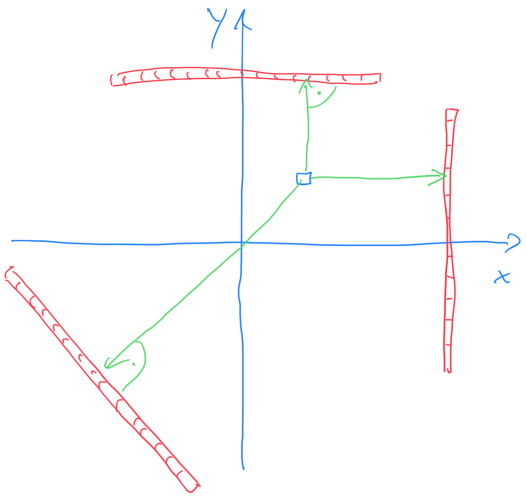
$$\xi = x \cos\gamma + y \sin\gamma$$

describes a line within $\mathbb{R}^2$.

There are now two interpretation of the integration

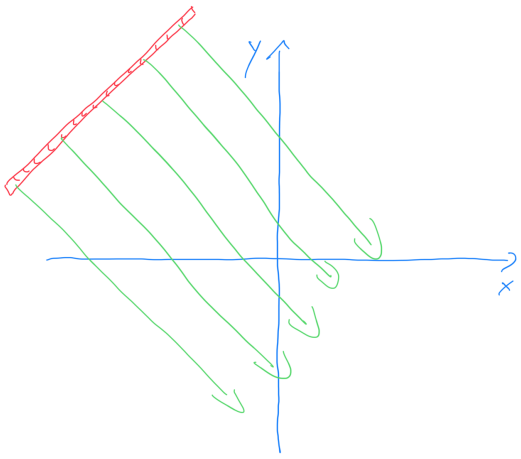
Backprojection – Interpretation 1

- One selects a certain pixels x, y
- Then, for each angle γ one throws the shortest line to the detector and pics the value.
- Thus, pixels are successively filled.



Backprojection – Interpretation 2

- Entire projection is projected back in a single step over the entire xy plane.
- The backprojected data are added to an image buffer.



Interpolation

Remark In a discrete setting the filtered backprojection requires an interpolation step.

$$\underbrace{h}_{\text{sampled at equidistant nodes}}\left(\underbrace{x\cos\gamma + y\cos\gamma}_{\text{In dependence of } \gamma \text{ non-regular}}, \gamma\right)$$

This interpolation is uncritical, since it happens in spatial domain so that numerical errors have only local effects.

Discretization

Number of angles

$L \in \mathbb{N}$: $\gamma_l = \frac{l}{L}\pi, \quad l = 0, \ldots, L-1$

Number of detector pixels

$M \in \mathbb{N}$: $\xi_m = A\left(\frac{m+0.5}{M} - 0.5\right), \quad m = 0, \ldots, M-1$

where A is the size of the detector.

Number of image pixels

$N_x \in \mathbb{N}$: $x_{n_x} = \Omega_x \left(\frac{n_x+0.5}{N_x} - 0.5\right), \quad n_x = 0, \ldots, N_x - 1$

$N_y \in \mathbb{N}$: $y_{n_y} = \Omega_y \left(\frac{n_y+0.5}{N_y} - 0.5\right), \quad n_y = 0, \ldots, N_y - 1$

where Ω_x and Ω_y are the side lengths of the image.

Time Complexity

Filtered Backprojection

$$\mathcal{O}(LM \log M + N^2 L) \underset{\text{if } L \approx N \approx M}{=} \mathcal{O}(N^2 \log N + N^3) = \mathcal{O}(N^3)$$

Fourier Slice based Reconstruction

$$\mathcal{O}(N^2 \log N)$$

Thus, FBP is a little bit slower, which is usually not critical in practice.

Parallel Implementation

The FBP can be impemented in a massively parallelized fashion. In particular reconstruction can already start *during* data acquisition.

$\rightarrow$ low latency

Filtering

In practice the projection $p(\xi, \gamma)$ are affected by noise

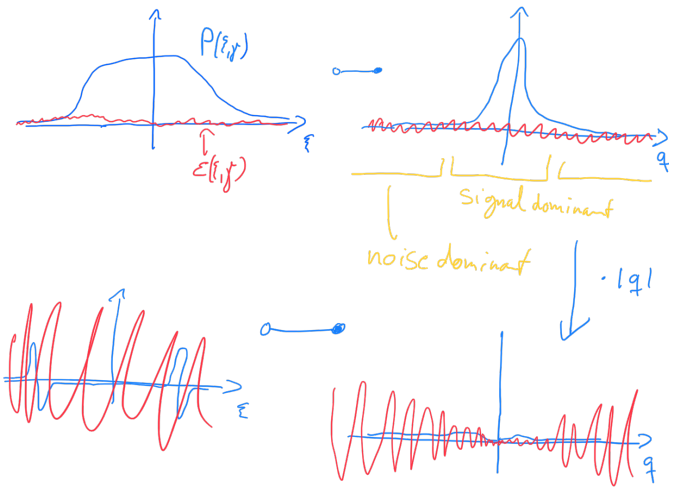
$$p(\xi, \gamma) = p^{\text{true}}(\xi, \gamma) + \varepsilon(\xi, \gamma)$$

The ramp filter $|p|$ leads to a noise amplification since the noise $\mathcal{F}\{\varepsilon(\xi, \gamma)\}$ is frequency independent.

$\rightarrow$ It is thus important to band-limit the filter

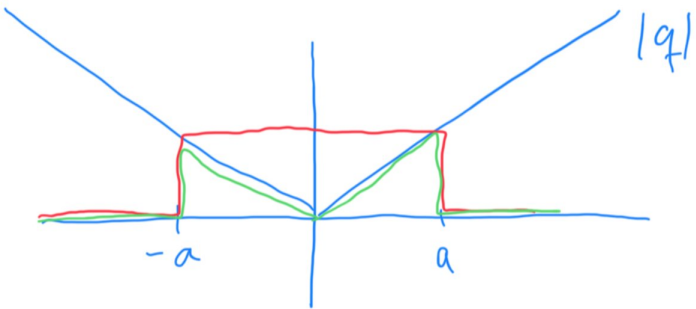
Filtering

- Illustration of the noise amplification of analytical image reconstruction.
- While low frequency components are signal dominated, high frequencies are noise dominated (upper right).
- High pass filter amplifies noise (lower left and right).



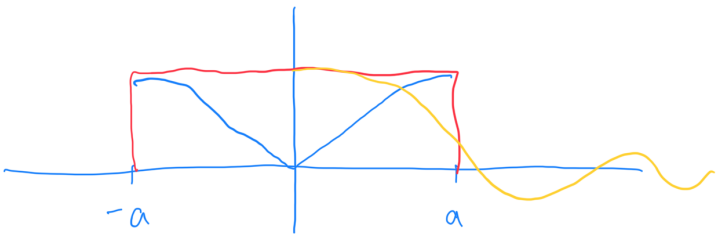
Ram-Lak Filter (Ramachandran - Lakshminarayan)

Replace $|q|$ with $|q|\text{rect}(\frac{q}{2a})$



Shepp-Logan Filter

Replace $|q|$ with $|q|\text{rect}(\frac{q}{2a})\text{sinc}(\frac{q}{a})$



Summary

- CT is an extension of radiography where the gantry is rotated.
- This solves the *uniqueness* issue of radiography and in turn allows for solving the inverse problem.
- The imaging operator can be *analytically* inverted and allows for direct image reconstruction.
- The inversion of the Radon transform is *noise amplifying*. This can mitigated by *filtering*.

Medical Imaging

Prof. Dr. Tobias Knopp

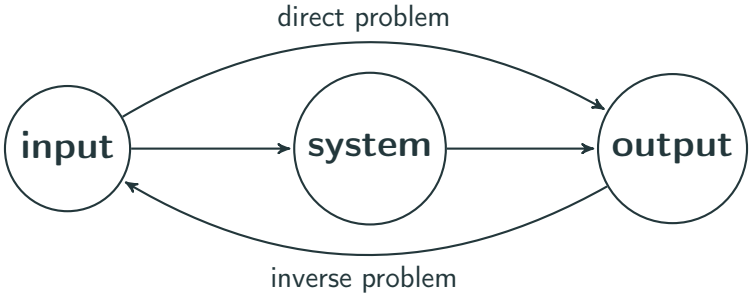
15. November 2022

Institut für Biomedizinische Bildgebung

Inverse Problems

Inverse Problems

Recall



Remarks

- So far we have learned that the existence and the uniqueness are important properties of an ill-posed problem.
- Next we will discuss what can further impact the solution negatively.
- In particular we discuss the *ill-posedness* of an inverse problem

Definition

Hadamard: An inverse problem is *well-posed* if the following conditions are fulfilled:

- **Existence:** The problem must have a solution.
- **Uniqueness:** There must be only one solution to the problem.
- **Stability:** The solution must depend continuously on the data.

Otherwise, the problem is *ill-posed*.

Remark

The last condition of Hadamard is tailored towards continuous inverse problems. For discrete ill-posed problems one usually considers the condition number of the system matrix as a metric for the ill-posedness.

Example - Existence

Consider the problem

$$\begin{pmatrix} 1 \\ 2 \end{pmatrix} x = \begin{pmatrix} 1 \\ 2.2 \end{pmatrix}.$$

It has no solution since x cannot be equal to 1 and 1.1 at the same time. But what we can do is to solve the **existence** issue by considering the least squares problem

$$\text{argmin}_x \left\| \begin{pmatrix} 1 \\ 2 \end{pmatrix} x - \begin{pmatrix} 1 \\ 2.2 \end{pmatrix} \right\|_2^2.$$

It has a unique solution $x = 1.08$.

Example - Uniqueness

An example of an inverse problem where the uniqueness is not given is the underdetermined system

$$x_1 + x_2 = 1.$$

It has infinite solutions. One can enforce a solution by adding the additional restriction that the solution $\boldsymbol{x} = (x_1, x_2)^\mathsf{T}$ should have minimum 2-norm, i.e. $\|\boldsymbol{x}\|_2^2 = (x_1^2 + x_2^2)$ is minimum. In this case the unique solution is given by $x_1 = x_2 = \frac{1}{2}$.

Example - Stability

The stability issue essentially means that small changes in the measurements can lead to very large changes in the solution.

Example:

$$\boldsymbol{A} = \begin{pmatrix} 0.16 & 0.1 \\ 0.17 & 0.11 \\ 2.02 & 1.29 \end{pmatrix} \quad \boldsymbol{A} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0.26 \\ 0.28 \\ 3.31 \end{pmatrix} =: \boldsymbol{b}$$

Example - Stability

Now lets assume the measurement $\boldsymbol{b}$ is disturbed by noise, i.e. $\tilde{\boldsymbol{b}} = \boldsymbol{b} + \begin{pmatrix} 0.01 \\ -0.03 \\ 0.02 \end{pmatrix}$,

which is about 1% noise.

The least squares solution $\text{argmin}_{\boldsymbol{x}} \|\boldsymbol{A}\boldsymbol{x} - \tilde{\boldsymbol{b}}\|_2$ is in this case given by $\begin{pmatrix} 7.01 \\ -8.4 \end{pmatrix}$. It has nothing in common with the true solution $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$. The reason for this is that the matrix $\boldsymbol{A}$ is *ill-conditioned*. Ill-conditioned problems are *effectively underdetermined*.

Example - Stability

For instance the vector $(-1.00, 1.57)^\mathsf{T}$ is almost in the nullspace of $\boldsymbol{A}$ since

$$\boldsymbol{A} \begin{pmatrix} -1.00 \\ 1.57 \end{pmatrix} = \begin{pmatrix} -0.0030 \\ 0.0027 \\ 0.0053 \end{pmatrix}$$

Hence one can add large amounts of this vector to a potential solution, with only minor impact on the right hand side. Minor perturbations of the right hand side (due to noise) thus can lead to large differences in the calculated solution.

Discrete Linear Inverse Problems

Discrete Linear Inverse Problems

A discrete and linear inverse problem can be formulated as a linear system of equations

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$$

where $\boldsymbol{A} \in \mathbb{C}^{M \times N}$ is the system matrix, $\boldsymbol{x} \in \mathbb{C}^{N}$ is the input vector and $\boldsymbol{b} \in \mathbb{C}^{M}$ is the output vector.

Discrete Linear Inverse Problems

In reality the measurement vector $\boldsymbol{b}$ is affected by (additive) noise $\boldsymbol{\varepsilon}$ such that one actually measures

$$\tilde{\boldsymbol{b}} = \boldsymbol{b} + \boldsymbol{\varepsilon}.$$

Hence the actual inverse problem reads

$$\boldsymbol{A}\boldsymbol{x} \approx \tilde{\boldsymbol{b}} = \boldsymbol{b} + \boldsymbol{\varepsilon}.$$

Note that that the relation between the model $\boldsymbol{A}\boldsymbol{x}$ and the measurement $\tilde{\boldsymbol{b}}$ is thus only a approximation.

Least Squares Approach

The standard approach to solve an inconsistent linear system of equation is to minimize the difference between $\boldsymbol{A}\boldsymbol{x}$ and $\tilde{\boldsymbol{b}}$. This is named the residual vector

$$\boldsymbol{r} := \boldsymbol{A}\boldsymbol{x} - \tilde{\boldsymbol{b}}$$

and the least squares approach is to minimized the norm of the residual vector:

$$\boldsymbol{x}_{\mathsf{LS}} = \underset{\boldsymbol{x}}{\mathsf{argmin}} \|\boldsymbol{r}\|_2^2 = \underset{\boldsymbol{x}}{\mathsf{argmin}} \|\boldsymbol{A}\boldsymbol{x} - \tilde{\boldsymbol{b}}\|_2^2. \tag{1}$$

Least Squares Approach

The least squares problem can be reformulated as a linear system of equations:

Theorem
The least squares problem (1) *is equivalent to the normal equation of first kind*

$$\boldsymbol{A}^{\mathsf{H}}\boldsymbol{A}\boldsymbol{x} = \boldsymbol{A}^{\mathsf{H}}\tilde{\boldsymbol{b}}.$$

and always has a solution.

Least Squares Approach - Proof Equivalence

To simplify the proof we assume that $\boldsymbol{A}$, $\boldsymbol{x}$, and $\tilde{\boldsymbol{b}}$ are real. Let $\boldsymbol{x}_*$ satisfy $\boldsymbol{A}^{\mathsf{H}}\boldsymbol{A}\boldsymbol{x}_* = \boldsymbol{A}^{\mathsf{H}}\tilde{\boldsymbol{b}}$ then for any $\boldsymbol{x} \in \mathbb{R}^N$

$$
\begin{aligned}
\|\boldsymbol{A}\boldsymbol{x} - \tilde{\boldsymbol{b}}\|_2^2 =& \|\boldsymbol{A}(\boldsymbol{x}_* + \boldsymbol{x} - \boldsymbol{x}_*) - \tilde{\boldsymbol{b}}\|_2^2 \\
=& (\boldsymbol{A}(\boldsymbol{x}_* + \boldsymbol{x} - \boldsymbol{x}_*) - \tilde{\boldsymbol{b}})^{\mathsf{T}}(\boldsymbol{A}(\boldsymbol{x}_* + \boldsymbol{x} - \boldsymbol{x}_*) - \tilde{\boldsymbol{b}}) \\
=& ((\boldsymbol{A}\boldsymbol{x}_* - \tilde{\boldsymbol{b}}) + \boldsymbol{A}(\boldsymbol{x} - \boldsymbol{x}_*))^{\mathsf{T}}((\boldsymbol{A}\boldsymbol{x}_* - \tilde{\boldsymbol{b}}) + \boldsymbol{A}(\boldsymbol{x} - \boldsymbol{x}_*)) \\
=& (\boldsymbol{A}\boldsymbol{x}_* - \tilde{\boldsymbol{b}})^{\mathsf{T}}(\boldsymbol{A}\boldsymbol{x}_* - \tilde{\boldsymbol{b}}) + (\boldsymbol{A}(\boldsymbol{x} - \boldsymbol{x}_*))^{\mathsf{T}}(\boldsymbol{A}(\boldsymbol{x} - \boldsymbol{x}_*)) \\
& + 2(\boldsymbol{x} - \boldsymbol{x}_*)^{\mathsf{T}} \underbrace{(\boldsymbol{A}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{x}_* - \boldsymbol{A}^{\mathsf{T}}\tilde{\boldsymbol{b}})}_{=\boldsymbol{0}} \\
=& \|\boldsymbol{A}\boldsymbol{x}_* - \tilde{\boldsymbol{b}}\|_2^2 + \|\boldsymbol{A}(\boldsymbol{x} - \boldsymbol{x}_*)\|_2^2 \\
\geq& \|\boldsymbol{A}\boldsymbol{x}_* - \tilde{\boldsymbol{b}}\|_2^2
\end{aligned}
$$

Thus $\boldsymbol{x}_*$ is a solution to the least squares problem (1).

Least Squares Approach - Proof Existence

Now suppose $\boldsymbol{x}_*$ is a solution to the least squares problem (1). The function

$$f(\boldsymbol{x}) = \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2 \tag{2}$$

thus has an extremum at $\boldsymbol{x}_*$. Since f is differentiable, the gradient is zero at $\boldsymbol{x}_*$, i.e. $\nabla f(\boldsymbol{x}_*) = \boldsymbol{0}$. Lets calculate the gradient

$$\begin{aligned}\frac{\partial f}{\partial x_j} f(\boldsymbol{x}) &= \frac{\partial f}{\partial x_j} \sum_{m=1}^{M} \left(\sum_{n=1}^{N} A_{m,n} x_n - b_m \right)^2 \\ &= 2 \sum_{m=1}^{M} \left(\sum_{n=1}^{N} A_{m,n} x_n - b_m \right) A_{m,j}\end{aligned}$$

and thus

$$\nabla f(\boldsymbol{x}) = \boldsymbol{A}^{\mathsf{T}}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})$$

Least Squares Approach - Proof Equivalence

Consequently we have

$$\nabla f(\boldsymbol{x}_*) = \boldsymbol{A}^\mathsf{T}(\boldsymbol{A}\boldsymbol{x}_* - \boldsymbol{b}) = \boldsymbol{0}$$

and thus the minimizer of the least squares problem is a solution of the normal equation.

Least Squares Approach - Proof Existence

The existence of a solution is guaranteed since

$$\lim_{\|\boldsymbol{x}\| \to \infty} f(\boldsymbol{x}) = \infty \tag{3}$$

and we can thus find a compact subset $U \subset \mathbb{R}^N$ where

$$f(\boldsymbol{x}) \leq f(\boldsymbol{0}) = \|\boldsymbol{b}\|_2^2$$

Since f is continuous and U is compact, the function will have a minimum in U according to the extreme value theorem.

Least Squares Approach - Unique Solution

Lemma
If $rank(\boldsymbol{A}) = N$ *the solution of the normal equation*

$$\boldsymbol{A}^{\mathsf{H}}\boldsymbol{A}\boldsymbol{x} = \boldsymbol{A}^{\mathsf{H}}\tilde{\boldsymbol{b}}.$$

is unique.

For real matrices we have $rank(\boldsymbol{A}) = rank(\boldsymbol{A}^{\mathsf{T}}\boldsymbol{A})$. Hence $\boldsymbol{A}^{\mathsf{T}}\boldsymbol{A}$ is a square $N \times N$ matrix with full rank. For such matrices an inverse always exists. The unique solution is given by

$$\boldsymbol{x}_* = (\boldsymbol{A}^{\mathsf{T}}\boldsymbol{A})^{-1}\boldsymbol{A}^{\mathsf{T}}\tilde{\boldsymbol{b}} \tag{4}$$

The matrix $\boldsymbol{A}^{+} := (\boldsymbol{A}^{\mathsf{T}}\boldsymbol{A})^{-1}\boldsymbol{A}^{\mathsf{T}}$ is the so-called *pseudoinverse*.

Why does the Least Squares Approach often fail?

Let us consider the residual vector of the true solution $\boldsymbol{x}$:

$$\boldsymbol{r} = \boldsymbol{A}\boldsymbol{x} - \tilde{\boldsymbol{b}} = \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} - \boldsymbol{\varepsilon} = -\boldsymbol{\varepsilon}.$$

Thus the true solution has a **non-zero** residual. The least squares approach does, however, minimize the residual even below the "optimal" residual of $\|\boldsymbol{\varepsilon}\|_2$. It will in particular find solutions with

$$\|\boldsymbol{r}\|_2 < \|\boldsymbol{\varepsilon}\|_2$$

In this case the solution is fitted to the noise leading to undesired results.

How can the noise amplification be quantified?

Theorem
Let $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ *and* $\boldsymbol{A}\tilde{\boldsymbol{x}} = \boldsymbol{b} + \varepsilon =: \tilde{\boldsymbol{b}}$. *Then the following inequality holds*

$$\frac{\|\boldsymbol{x} - \tilde{\boldsymbol{x}}\|_2}{\|\boldsymbol{x}\|_2} \leq \mathit{cond}(\boldsymbol{A}) \frac{\|\boldsymbol{b} - \tilde{\boldsymbol{b}}\|_2}{\|\boldsymbol{b}\|_2}$$

Thus an error in the measurement data can be amplified by the factor $\mathrm{cond}(\boldsymbol{A})$, which is the condition number of $\boldsymbol{A}$.

Regularization

Regularization

In order to handle ill-posed problems where the least squares method fails one applies so-called *regularization techniques* that stabilize the solution. In particular the linear system is exchanged with a similar system that is better conditioned.

Definition
The Tikhonov regularization technique considers the following optimization problem

$$\boldsymbol{x}_{\mathsf{LS}}^{\lambda} = \underset{\boldsymbol{x}}{\mathsf{argmin}} \|\boldsymbol{A}\boldsymbol{x} - \tilde{\boldsymbol{b}}\|_2^2 + \lambda \|\boldsymbol{x}\|_2^2 \tag{5}$$

Here, λ is the so-called regularization parameter

Theorem
The regularized least squares problem can be equivalently solved by the regularized normal equation

$$(\boldsymbol{A}^{\mathsf{H}}\boldsymbol{A} + \lambda \boldsymbol{I})\boldsymbol{x} = \boldsymbol{A}^{\mathsf{H}}\tilde{\boldsymbol{b}}.$$

Regularization - Proof Equivalence

Reformulate the minimization problem

$$\begin{aligned}\underset{\boldsymbol{x}}{\operatorname{argmin}} \|\boldsymbol{A}\boldsymbol{x} - \tilde{\boldsymbol{b}}\|_2^2 + \lambda\|\boldsymbol{x}\|_2^2 &= \underset{\boldsymbol{x}}{\operatorname{argmin}} \left\| \begin{pmatrix} \boldsymbol{A}\boldsymbol{x} - \tilde{\boldsymbol{b}} \\ \sqrt{\lambda}\boldsymbol{I}\boldsymbol{x} \end{pmatrix} \right\|_2^2 \\ &= \underset{\boldsymbol{x}}{\operatorname{argmin}} \left\| \begin{pmatrix} \boldsymbol{A} \\ \sqrt{\lambda}\boldsymbol{I} \end{pmatrix} \boldsymbol{x} - \begin{pmatrix} \tilde{\boldsymbol{b}} \\ \boldsymbol{0} \end{pmatrix} \right\|_2^2\end{aligned}$$

into the standard least squares form. The normal equation of this least squares problem is given by

$$\begin{pmatrix} \boldsymbol{A} \\ \sqrt{\lambda}\boldsymbol{I} \end{pmatrix}^{\mathsf{T}} \begin{pmatrix} \boldsymbol{A} \\ \sqrt{\lambda}\boldsymbol{I} \end{pmatrix} \boldsymbol{x} = \begin{pmatrix} \boldsymbol{A} \\ \sqrt{\lambda}\boldsymbol{I} \end{pmatrix}^{\mathsf{T}} \begin{pmatrix} \tilde{\boldsymbol{b}} \\ \boldsymbol{0} \end{pmatrix}$$

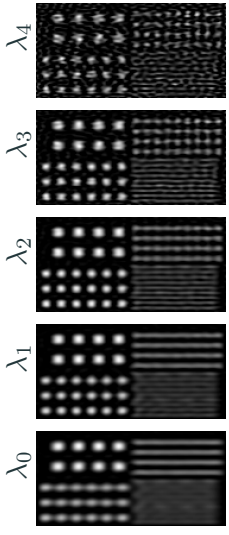
which is equivalent to

$$(\boldsymbol{A}^{\mathsf{T}}\boldsymbol{A} + \lambda\boldsymbol{I})\boldsymbol{x} = \boldsymbol{A}^{\mathsf{T}}\tilde{\boldsymbol{b}}.$$

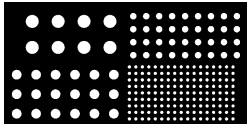
Regularization

- The regularization term (or penalty term) $\lambda \|\boldsymbol{x}\|_2^2$ controls that the solution gets not too large.
- Regularization reduces noise in the calculated solution, i.e. it smoothes $\boldsymbol{x}_{\mathrm{LS}}^{\lambda}$.
- Regularization does, however, also introduces a *bias* (i.e. systematic "global" deviation) between $\boldsymbol{x}$ and $\boldsymbol{x}_{\mathrm{LS}}^{\lambda}$.
- In practice one has to trade of between a too noise and a too smooth solution by appropriate choice of lambda.

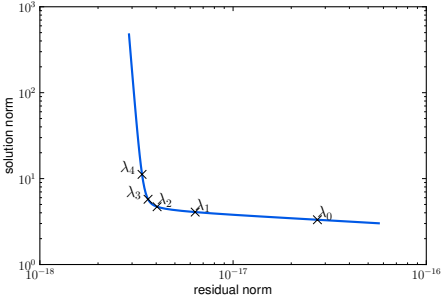
Influence of the regularization parameter



Original phantom

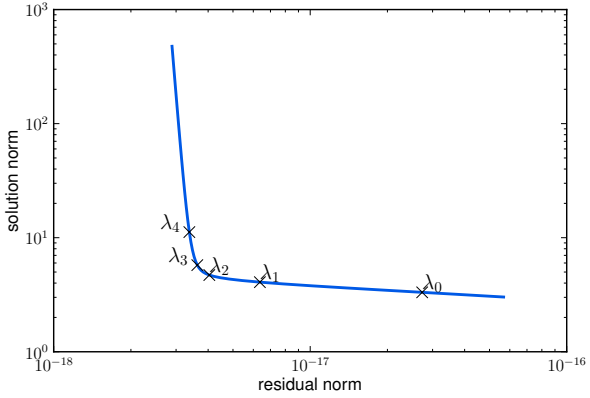


L curve



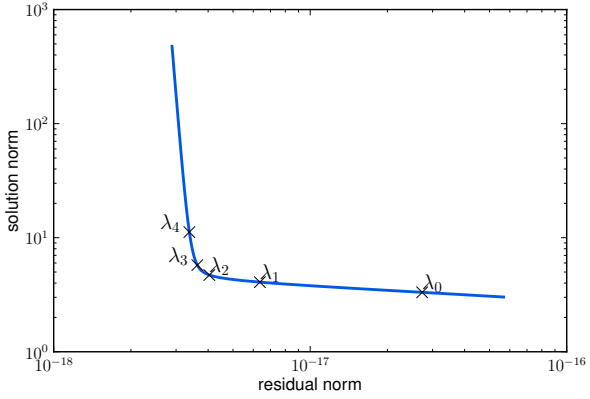
How to Choose the Regularization Parameter?

Choosing λ is a challenging problem in practice. What is often done is to compute various solutions for different λ and plot the solution norm $\|\boldsymbol{x}_{\mathsf{LS}}^{\lambda}\|_2$ versus the residual norm $\|\boldsymbol{A}\boldsymbol{x}_{\mathsf{LS}}^{\lambda} - \tilde{\boldsymbol{b}}\|_2^2$



How to Choose the Regularization Parameter?

This curve typically has an L-curved shape and the best promise is usually found in the corner of the L. The corner is that point where the residual is close to minimal but the solution norm $\|\boldsymbol{x}_{\mathrm{LS}}^{\lambda}\|_2$ is still not "blown up" due noise amplification.



Singular Value Decomposition

Singular Value Decomposition

The singular value decomposition is a matrix decomposition that allows to

- solve a linear system of equations
- apply regularization efficiently
- understand ill-posed problems, i.e. do a fine grained analysis of the ill-posedness of an inverse problem

Singular Value Decomposition

As before we consider the linear system

$$\boldsymbol{A}\boldsymbol{x} \approx \tilde{\boldsymbol{b}} = \boldsymbol{b} + \boldsymbol{\varepsilon} \quad \text{where} \quad \boldsymbol{A} \in \mathbb{C}^{M \times N}$$

Theorem
Any $M \times N$ matrix can be decomposed into

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathsf{H}}$$

where $\boldsymbol{U} \in \mathbb{C}^{M \times r}$ and $\boldsymbol{V} \in \mathbb{C}^{N \times r}$ are rectangular with orthogonal columns and $r = \mathsf{rank}(\boldsymbol{A}) \leq \min(M, N)$. The diagonal matrix

$$\boldsymbol{\Sigma} = \mathsf{diag}(\boldsymbol{\sigma}) \in \mathbb{R}_{+}^{r \times r}$$

contains the *singular values* $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_r)^{\mathsf{T}}$ in descending order.

(without proof)

Remarks

- The SVD is not unique but at least one SVD exists
- Calculating the SVD has a complexity of $\mathcal{O}(N^3)$ if $M \approx N$. It thus has the same complexity as Gaussian Elimination. The constant infront of the N^3 factor is large though
- We have defined the *compact* SVD. There is also a non-compat version

$$\boldsymbol{A} = \tilde{\boldsymbol{U}}\tilde{\boldsymbol{\Sigma}}\tilde{\boldsymbol{V}}^{\mathsf{H}}$$

where $\tilde{\boldsymbol{U}} \in \mathbb{C}^{M \times M}$ and $\tilde{\boldsymbol{V}} \in \mathbb{C}^{N \times N}$ are unitary ($\tilde{\boldsymbol{U}}^{\mathsf{H}}\tilde{\boldsymbol{U}} = \boldsymbol{I}$, $\tilde{\boldsymbol{V}}^{\mathsf{H}}\tilde{\boldsymbol{V}} = \boldsymbol{I}$) and $\boldsymbol{\Sigma} = \mathbb{R}^{M \times N}$ is a rectangular diagonal matrix that may contain zero entries at its main diagonal.

Solving Linear Systems

In the following we consider $r = N$, $M \geq N$, i.e. an overdetermined system, where $\boldsymbol{V}$ is square and unitary, i.e. the inverse exists.

$$
\begin{aligned}
&\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \\
\Rightarrow &\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathsf{H}}\boldsymbol{x} = \boldsymbol{b} \\
\Rightarrow &\underbrace{\boldsymbol{U}^{\mathsf{H}}\boldsymbol{U}}_{\boldsymbol{I}}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathsf{H}}\boldsymbol{x} = \boldsymbol{U}^{\mathsf{H}}\boldsymbol{b} \\
\Rightarrow &\underbrace{\boldsymbol{\Sigma}^{-1}\boldsymbol{\Sigma}}_{\boldsymbol{I}}\boldsymbol{V}^{\mathsf{H}}\boldsymbol{x} = \boldsymbol{\Sigma}^{-1}\boldsymbol{U}^{\mathsf{H}}\boldsymbol{b} \\
\Rightarrow &\boldsymbol{x} = \boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{U}^{\mathsf{H}}\boldsymbol{b}
\end{aligned}
$$

Remark: The matrix $\boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{U}^{\mathsf{H}}$ is the same as the pseudo inverse $\boldsymbol{A}^{+} = (\boldsymbol{A}^{\mathsf{H}}\boldsymbol{A})^{-1}\boldsymbol{A}^{\mathsf{H}}$.

$$
\begin{aligned}
\boldsymbol{A}^{+} &= (\boldsymbol{A}^{\mathsf{H}}\boldsymbol{A})^{-1}\boldsymbol{A}^{\mathsf{H}} \\
&= (\boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{U}^{\mathsf{H}}\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathsf{H}})^{-1}\boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{U}^{\mathsf{H}} \\
&= (\boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathsf{H}})^{-1}\boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{U}^{\mathsf{H}} \\
&= (\boldsymbol{V}\boldsymbol{\Sigma}^{2}\boldsymbol{V}^{\mathsf{H}})^{-1}\boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{U}^{\mathsf{H}} \\
&= \underbrace{(\boldsymbol{V}^{\mathsf{H}})^{-1}}_{\boldsymbol{V}}\boldsymbol{\Sigma}^{-2}\boldsymbol{V}^{-1}\boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{U}^{\mathsf{H}} \\
&= \boldsymbol{V}\boldsymbol{\Sigma}^{-2}\boldsymbol{\Sigma}\boldsymbol{U}^{\mathsf{H}} \\
&= \boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{U}^{\mathsf{H}}
\end{aligned}
$$

Solving Linear Systems

Theorem: The solution $\boldsymbol{x} = \boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{U}^{\mathsf{H}}$ can also be expressed as

$$\boldsymbol{x} = \sum_{i=1}^{r} \frac{\boldsymbol{U}_{\cdot,i}^{\mathsf{H}}\boldsymbol{b}}{\sigma_i}\boldsymbol{V}_{\cdot,i}$$

Proof:

$$\begin{aligned}
&\begin{pmatrix} \boldsymbol{V}_{\cdot,1} & \cdots & \boldsymbol{V}_{\cdot,r} \end{pmatrix} \begin{pmatrix} \frac{1}{\sigma_1} & & \\ & \ddots & \\ & & \frac{1}{\sigma_r} \end{pmatrix} \begin{pmatrix} \boldsymbol{U}_{\cdot,1}^{\mathsf{H}} \\ \vdots \\ \boldsymbol{U}_{\cdot,r}^{\mathsf{H}} \end{pmatrix} \boldsymbol{b} \\
&= \begin{pmatrix} \boldsymbol{V}_{\cdot,1} & \cdots & \boldsymbol{V}_{\cdot,r} \end{pmatrix} \begin{pmatrix} \frac{1}{\sigma_1}\boldsymbol{U}_{\cdot,1}^{\mathsf{H}}\boldsymbol{b} \\ \vdots \\ \frac{1}{\sigma_r}\boldsymbol{U}_{\cdot,r}^{\mathsf{H}}\boldsymbol{b} \end{pmatrix} \\
&= \sum_{i=1}^{r} \frac{\boldsymbol{U}_{\cdot,i}^{\mathsf{H}}\boldsymbol{b}}{\sigma_i}\boldsymbol{V}_{\cdot,i}
\end{aligned}$$

SVD for Ill-Posed Problems

Let us have a look at the SVD based solution in case of a noisy linear system. If we insert $\tilde{\boldsymbol{b}} = \boldsymbol{b} + \boldsymbol{\varepsilon}$ then we have

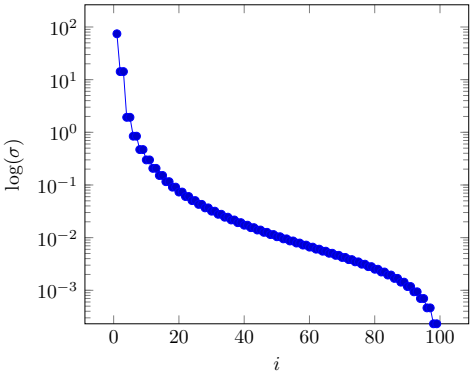
$$\begin{aligned}\tilde{\boldsymbol{x}} &= \sum_{i=1}^{r} \frac{\boldsymbol{U}_{\cdot,i}^{\mathsf{H}}(\boldsymbol{b} + \boldsymbol{\varepsilon})}{\sigma_i} \boldsymbol{V}_{\cdot,i} \\ &= \sum_{i=1}^{r} \frac{\boldsymbol{U}_{\cdot,i}^{\mathsf{H}}\boldsymbol{b}}{\sigma_i} \boldsymbol{V}_{\cdot,i} + \sum_{i=1}^{r} \frac{\boldsymbol{U}_{\cdot,i}^{\mathsf{H}}\boldsymbol{\varepsilon}}{\sigma_i} \boldsymbol{V}_{\cdot,i}\end{aligned}$$

The numerator in the right sum has a constant standard deviation independently of r. The denominator, however, decreases with increasing i for ill-posed problems.

$\rightarrow$ The small singular values ($\frac{1}{\sigma}$) amplify the noise.

SVD for Ill-Posed Problems

Example singular values of a discrete convolution matrix




```
using PGFPlots, ToeplitzMatrices,
      LinearAlgebra
N = 100
t = range(-1,1,length=N)
sigma = 1
a = exp.(-(t.^2)./sigma)
A = Circulant(a) |> Matrix
U,S,V = svd(A)


p = Plots.Linear(1:N, S)
p = Axis(p, ymode="log", ymin=S[end-1],
            xlabel=L"$i$", ylabel=L"log($\sigma$)")
```

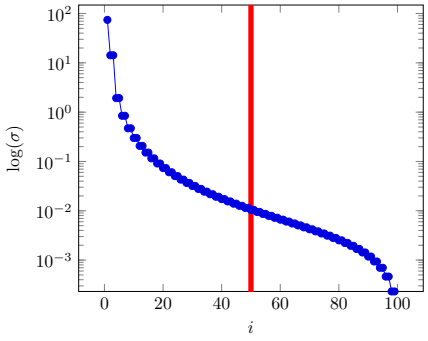
Remark: Recall that the noise amplification is bounded by the condition number $\text{cond}(\boldsymbol{A}) = \frac{\sigma_1}{\sigma_r}$.

Truncated SVD

One regularization method is to neglect small singular values that are responsible for the noise amplification:

$$\tilde{\boldsymbol{x}} = \sum_{i=1}^{\alpha} \frac{\boldsymbol{U}_{\cdot,i}^{\mathsf{H}} \tilde{\boldsymbol{b}}}{\sigma_i} \boldsymbol{V}_{\cdot,i}$$

where α with $1 \leq \alpha \leq r$ is the truncation parameter that acts like a regularization parameter.



Truncated SVD

Remarks

- The truncation suppresses the noise amplification but it smoothes the solution (like Tikhonov regularization).
- The truncation is a filter (rect function) that acts on the singular values. Since it lets small singular values pass, it is a *low-pass filter*.

Convolution Matrix

Note that if $\boldsymbol{A}$ is a (periodic) convolution matrix, we have

$$\boldsymbol{A} = \boldsymbol{F}\boldsymbol{\Sigma}\boldsymbol{F}^{\mathsf{H}}$$

where $\boldsymbol{F} = \boldsymbol{U} = \boldsymbol{V}$ is the discrete Fourier matrix and the singular values in $\boldsymbol{\Sigma}$ contain the transfer function (Fourier coefficients of the convolution kernel).

Filtering of singular values is thus directly related to the filtering we considered for Fourier transformation. In fact, a Fourier transform can be defined for other bases than the trigonometric functions.

Tikhonov Regularization with the SVD

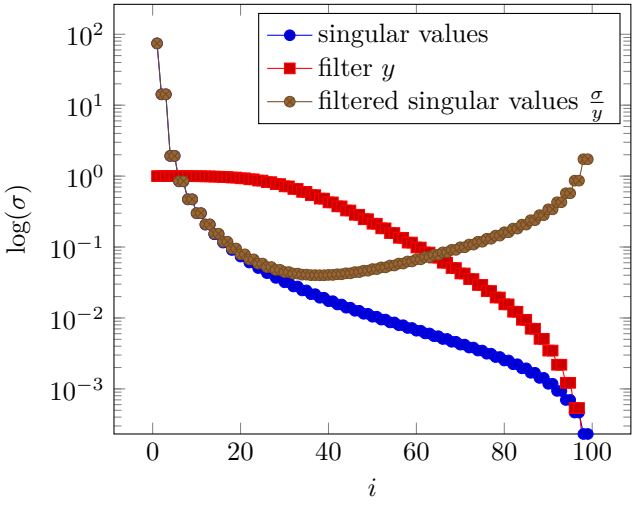
Next, let us investigate the relation between Tikhonov regularization and the SVD.

$$
\begin{aligned}
\boldsymbol{x}_\lambda &= (\boldsymbol{A}^{\mathsf{H}}\boldsymbol{A} + \lambda \boldsymbol{I})^{-1}\boldsymbol{A}^{\mathsf{H}}\boldsymbol{b} \\
&= (\boldsymbol{V}\boldsymbol{\Sigma}^2\boldsymbol{V}^{\mathsf{H}} + \lambda \boldsymbol{I})^{-1}\boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{U}^{\mathsf{H}}\boldsymbol{b} \\
&= (\boldsymbol{V}(\boldsymbol{\Sigma}^2 + \lambda \boldsymbol{I})\boldsymbol{V}^{\mathsf{H}})^{-1}\boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{U}^{\mathsf{H}}\boldsymbol{b} \\
&= (\boldsymbol{V}^{\mathsf{H}})^{-1}(\boldsymbol{\Sigma}^2 + \lambda \boldsymbol{I})^{-1}\boldsymbol{V}^{-1}\boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{U}^{\mathsf{H}}\boldsymbol{b} \\
&= \boldsymbol{V}\underbrace{(\boldsymbol{\Sigma}^2 + \lambda \boldsymbol{I})^{-1}\boldsymbol{\Sigma}}_{\mathsf{diag}\left(\left(\frac{\sigma_i}{\sigma_i^2+\lambda}\right)_{i=1}^{r}\right)}\boldsymbol{U}^{\mathsf{H}}\boldsymbol{b} \\
&= \sum_{i=1}^{r} \frac{\sigma_i \boldsymbol{U}_{\cdot,i}^{\mathsf{H}}\boldsymbol{b}}{\sigma_i^2 + \lambda}\boldsymbol{V}_{\cdot,i} = \sum_{i=1}^{r} y_i \frac{\boldsymbol{U}_{\cdot,i}^{\mathsf{H}}\boldsymbol{b}}{\sigma_i}\boldsymbol{V}_{\cdot,i}
\end{aligned}
$$

where $y_i = \frac{\sigma_i^2}{\sigma_i^2+\lambda}$ are the filter factors.

Tikhonov Regularization with the SVD

The following figure showcases the effect of the filter on the singular values.



Compare this to the truncated SVD where the singular values were cutted.

Tikhonov regularization also has an effective cutoff (at about $i = 40$ in this example) but the transition is smoother.

Tikhonov Regularization with the SVD

Remarks

- Explicit version of Tikhonov regularization
- SVD takes $\mathcal{O}(N^3)$ but solutions for different λ can be calculated in $\mathcal{O}(N^2)$

$\rightarrow$ One time cost that pays off.

L-Curve with the SVD

The L-curve needs for $\gamma = 1, \ldots, \Gamma$ the values

$$(\|\boldsymbol{A}\boldsymbol{x}_{\lambda_\gamma} - \tilde{\boldsymbol{b}}\|_2^2, \|\boldsymbol{x}_{\lambda_\gamma}\|_2^2)$$

Using a linear solver without matrix decomposition (e.g. Gaussian elimination) this requires $\mathcal{O}(N^3\Gamma)$ operations.

Using a matrix decomposition technique (e.g. SVD) this requires $\mathcal{O}(N^2\Gamma)$ operations.

But with the SVD we can obtain these values even faster.

L-Curve with the SVD

Due to the orthogonality of $\boldsymbol{V}$ we have

$$\|\boldsymbol{A}\boldsymbol{x}_{\lambda_\gamma} - \tilde{\boldsymbol{b}}\|_2^2 = \sum_{i=1}^{r} \left| \frac{\lambda_\gamma}{\sigma_i^2 + \lambda_\gamma} \boldsymbol{U}_{\cdot,i}^{\mathsf{H}} \tilde{\boldsymbol{b}} \right|^2$$

$$\|\boldsymbol{x}_{\lambda_\gamma}\|_2^2 = \sum_{i=1}^{r} \left| \frac{\sigma_i}{\sigma_i^2 + \lambda_\gamma} \boldsymbol{U}_{\cdot,i}^{\mathsf{H}} \tilde{\boldsymbol{b}} \right|^2$$

One can observe that $\boldsymbol{U}_{\cdot,i}^{\mathsf{H}} \tilde{\boldsymbol{b}}$ is independent of λ an thus can be precomputed once. In total, an L-curve using the SVD thus can be obtained in $\mathcal{O}(N^2 + \Gamma N)$ steps. For $\Gamma \in \mathcal{O}(N)$ this is the same complexity as calculating the solution $\boldsymbol{x}_\lambda$. Furthermore, the matrix-vector operation $\boldsymbol{U}^{\mathsf{H}} \tilde{\boldsymbol{b}}$ needs only to be computed once.

Summary

- Inverse problems are hard
- They are prone to noise amplification
- They require special treatment to yield a satisfying solution
- One can reduce the noise amplification but has to live with a bias
- The come up in many real-live problems

Medical Imaging

Prof. Dr. Tobias Knopp

23. November 2022

Institut für Biomedizinische Bildgebung

Iterative Reconstruction

Iterative Reconstruction

The SVD is a very advanced tool but what if

- The system matrix is sparse
- The system matrix is huge and does not fit into the main memory

In the second case, one usually has a formula for the matrix elements, which implies that the entire system matrix does not need to be setup in memory.

In both cases it is better to use *iterative solvers*.

Iterative Reconstruction

There are various iterative solvers, of which several can be grouped into the following two classes

- Krylov subspace methods
- Row- or column action methods

There are further classes, which we will, however not discuss at this point.

Iterative solver do not require element-wise access to the system matrix. Instead they require operations involving the system matrix. This can for instance be

- Matrix-vector multiplications with $\boldsymbol{A}$ or $\boldsymbol{A}^{\mathsf{H}}$, or
- Operations involving the matrix rows or columns.

Kaczmarz Method

- Also names *algebraic reconstruction technique* (ART) in the context of computed tomography.
- Fixed-point iteration, which converges to the solution $\boldsymbol{x}$ of the linear system $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ if it exists.
- Let $l \geq 1$ be the iteration number and $\boldsymbol{x}^0 = \boldsymbol{0}$ be the start vector, then the Kaczmarz iteration is defined as

$$\boldsymbol{x}^{l+1} = \boldsymbol{x}^l + \frac{b_j - \boldsymbol{A}_{j,\cdot}\boldsymbol{x}^l}{\|\boldsymbol{A}_{j,\cdot}^{\mathsf{T}}\|_2^2} \boldsymbol{A}_{j,\cdot}^{\mathsf{H}}$$

- Row index j is usually chosen to sweep over all matrix rows so that one has two nested for loops and $j = l \bmod M$.

Kaczmarz Method – Derivation

Let us consider the j-th equation of the linear system $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$. It can be expressed as

$$\boldsymbol{A}_{j,\cdot}\boldsymbol{x} = b_j.$$

Normalization of the vector $\boldsymbol{A}_{j,\cdot}$ yields

$$\begin{aligned} 0 &= \frac{b_j}{\|\boldsymbol{A}_{j,\cdot}^{\mathsf{T}}\|_2} - \frac{\boldsymbol{A}_{j,\cdot}}{\|\boldsymbol{A}_{j,\cdot}^{\mathsf{T}}\|_2}\boldsymbol{x} \\ &= d - \boldsymbol{n}^{\mathsf{H}}\boldsymbol{x}, \end{aligned}$$

where, $d = \frac{b_j}{\|\boldsymbol{A}_{j,\cdot}^{\mathsf{T}}\|_2}$ and $\boldsymbol{n} = \frac{\boldsymbol{A}_{j,\cdot}^{\mathsf{H}}}{\|\boldsymbol{A}_{j,\cdot}^{\mathsf{T}}\|_2}$ This is a hyperplane (e.g. line / plane in 2D and 3D) equation in Hessian normal form. Each vector within the hyperplane is orthogonal to the normal vector $\boldsymbol{n}$ and d is the distance to the origin. $\boldsymbol{n}$ points in direction of the hyperplane and for an orthogonal projection we just need to know the distance of a point to the plane.

Kaczmarz Method – Derivation

The distance of an arbitrary point $\tilde{\boldsymbol{x}}$ to the plane is

$$\mathrm{dist}(\tilde{\boldsymbol{x}}) = d - \boldsymbol{n}^{\mathsf{H}}\tilde{\boldsymbol{x}}$$

To project $\tilde{\boldsymbol{x}}$ onto the hyperplane, one thus has to add $\mathrm{dist}(\tilde{\boldsymbol{x}})\boldsymbol{n}$ to $\tilde{\boldsymbol{x}}$, i.e.

$$\begin{aligned}
\tilde{\boldsymbol{x}} + \mathrm{dist}(\tilde{\boldsymbol{x}})\boldsymbol{n} &= \tilde{\boldsymbol{x}} + (d - \boldsymbol{n}^{\mathsf{H}}\tilde{\boldsymbol{x}})\boldsymbol{n} \\
&= \tilde{\boldsymbol{x}} + \left(\frac{b_j}{\|\boldsymbol{A}_{j,\cdot}^{\mathsf{T}}\|_2} - \frac{\boldsymbol{A}_{j,\cdot}}{\|\boldsymbol{A}_{j,\cdot}^{\mathsf{T}}\|_2}\tilde{\boldsymbol{x}} \right) \frac{\boldsymbol{A}_{j,\cdot}^{\mathsf{H}}}{\|\boldsymbol{A}_{j,\cdot}^{\mathsf{T}}\|_2} \\
&= \tilde{\boldsymbol{x}} + \frac{b_j - \boldsymbol{A}_{j,\cdot}\tilde{\boldsymbol{x}}}{\|\boldsymbol{A}_{j,\cdot}^{\mathsf{T}}\|_2^2}\boldsymbol{A}_{j,\cdot}^{\mathsf{H}}
\end{aligned}$$

Converting this into an iteration process, we end up with the Kaczmarz iteration.

Kaczmarz Method – Geometric Interpretation

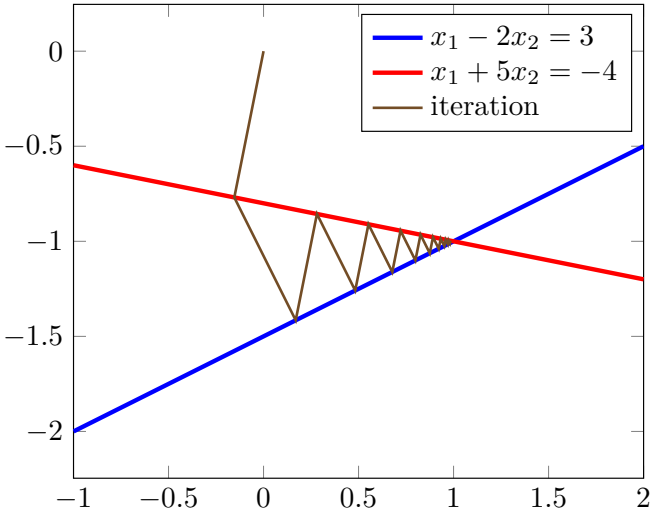
Consider 2×2 linear system

$$\begin{pmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

In this case each equation of the linear system describes a line in the $\mathbb{R}^2$. In the point where the lines intersect, the linear system has its solution.

The Kaczmarz iteration performs in each step an orthogonal projection on the hyperplane spanned by the j-th matrix row and the corresponding j-th element of the right hand side.

Kaczmarz Method – Geometric Interpretation



Kaczmarz Method – Convergence

Convergence speed (i.e. number of required iterations) depends on the similarity of successive matrix rows. If successive matrix rows are similar, more iterations are required.

Example: Convolution matrix

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

$\langle \boldsymbol{A}_{1,\cdot}, \boldsymbol{A}_{2,\cdot} \rangle_2 = 2$
but
$\langle \boldsymbol{A}_{1,\cdot}, \boldsymbol{A}_{4,\cdot} \rangle_2 = 0$
$\rightarrow$ not clever to run over matrix rows in order

Randomized Kaczmarz Method

There are two possible ways to improve the convergence speed of the Kaczmarz method

- If the structure of the system matrix is known, run through the matrix such that successive row indices have a small inner product
- Otherwise: Run in random order through the matrix rows

The second option is known as the *Randomized Kaczmarz* and can be shown to converge faster then the non-random Kaczmarz.

Regularized Kaczmarz

Kaczmarz method can be shown to solve

$$\|\boldsymbol{x}\|_2 \to \min \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$$

This problem is considered if the linear system is under-determined and has infinite solutions. In contrast, the problem

$$\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2 \to \min$$

is considered for over-determined linear systems. Furthermore, for ill-conditioned linear systems one actually wants to solve

$$\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2 + \lambda\|\boldsymbol{x}\|_2^2 \to \min$$

How can this be done with Kaczmarz method?

Regularized Kaczmarz

Apply Kaczmarz algorithm to an extended system

$$\underbrace{\begin{pmatrix} \boldsymbol{A} & \lambda^{\frac{1}{2}}\boldsymbol{I} \end{pmatrix}}_{\tilde{\boldsymbol{A}} \in \mathbb{C}^{M \times (N+M)}} \underbrace{\begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{v} \end{pmatrix}}_{\tilde{\boldsymbol{x}} \in \mathbb{C}^{(N+M)}} = \boldsymbol{b}$$

Here, $\boldsymbol{v} \in \mathbb{C}^M$ in an auxiliary vector. Multiplying out yields

$$\begin{aligned} \boldsymbol{A}\boldsymbol{x} + \lambda^{\frac{1}{2}}\boldsymbol{v} &= \boldsymbol{b} \\ \Rightarrow \boldsymbol{v} &= -\lambda^{-\frac{1}{2}}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) \end{aligned}$$

Thus, the auxiliary vector will be the scaled residual after convergence.

What does the extended Kaczmarz calculate?

Regularized Kaczmarz

$$
\begin{aligned}
&\|\tilde{\boldsymbol{x}}\|_2 \to \min \quad \text{subject to} \quad \tilde{\boldsymbol{A}}\tilde{\boldsymbol{x}} = \boldsymbol{b} \\
\Leftrightarrow &\|\tilde{\boldsymbol{x}}\|_2^2 \to \min \quad \text{subject to} \quad \tilde{\boldsymbol{A}}\tilde{\boldsymbol{x}} = \boldsymbol{b} \\
\Leftrightarrow &\|\boldsymbol{x}\|_2^2 + \|\boldsymbol{v}\|_2^2 \to \min \quad \text{subject to} \quad \boldsymbol{v} = -\lambda^{-\frac{1}{2}}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) \\
\Leftrightarrow &\|\boldsymbol{x}\|_2^2 + \| - \lambda^{-\frac{1}{2}}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})\|_2^2 \to \min \\
\Leftrightarrow &\|\boldsymbol{x}\|_2^2 + \lambda^{-1}\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2 \to \min \\
\Leftrightarrow &\lambda\|\boldsymbol{x}\|_2^2 + \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2 \to \min
\end{aligned}
$$

Thus, the extended Kaczmarz solves the Tikhonov regularized least squares problem.

Kaczmarz – Required operations

Kaczmarz method requires two elementary operations involving the system matrix $\boldsymbol{A}$

- An inner product $\alpha \leftarrow \boldsymbol{A}_{j,\cdot}\tilde{\boldsymbol{x}} = \sum_{n=1}^{N} \boldsymbol{A}_{j,n}\tilde{\boldsymbol{x}}_n$
- A vector update $\tilde{\boldsymbol{x}} \leftarrow \tilde{\boldsymbol{x}} + \alpha \boldsymbol{A}_{j,\cdot}^{\mathsf{H}}$
  i.e. $\tilde{\boldsymbol{x}}_n \leftarrow \tilde{\boldsymbol{x}}_n + \alpha\overline{\boldsymbol{A}_{j,n}}$ for $n = 1, \ldots, N$

Both are vector-vector operations that can be easily accelerated in case that the system matrix $\boldsymbol{A}$ is sparse. In that case only those indices are considered in the calculation for which $\boldsymbol{A}_{j,n} \neq 0$.

Kaczmarz – Required operations

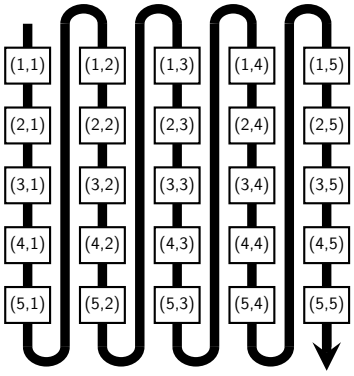
To implement the (non-regularized) Kaczmarz method in a generic fashion, it can be implemented as follows:

```
for i in rowIndexCycle
  j = rowindex[i]
  tau = dot_with_matrix_row(A, x, j)
  alpha = (b[j]-tau) / normA[j]
  kaczmarz_update!(A, x, j, alpha)
end
```

Here, `dot_with_matrix_row` and `kaczmarz_update!` are two functions that need to be implemented for each type of matrix, for instance it can be implemented for `Matrix{Float64}` and for `SparseMatrixCSC{Float64, Int64}`.

Kaczmarz – Required operations

In Julia dense matrices are stored in column major order, which means that the elements of the columns are stored next to each other in memory.



Performing row operations on such a data structure is very expensive since CPU caching cannot be utilized. To implement the Kaczmarz algorithm efficiently, one should thus first transpose the data and then use a transpose wrapper

```
julia> A = transpose(rand(3,3))
3×3 LinearAlgebra.Transpose{Float64,
                            Array{Float64,2}}:
 0.84271    0.342911  0.555876
 0.931374   0.886989  0.163034
 0.0734473  0.034807  0.296932
```

Conjugated Gradient

Conjugated Gradient

Conjugated Gradient (CG) is a popular Krylov subspace method that solves $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ for symmetric positive definite $\boldsymbol{A}$, i.e. $\boldsymbol{z}^{\mathsf{H}}\boldsymbol{A}\boldsymbol{z} > 0$ for any $\boldsymbol{z} \neq \boldsymbol{0}$.

For general $\boldsymbol{A}$ one can apply CG to the normal equation

$$\boldsymbol{A}^{\mathsf{H}}\boldsymbol{A}\boldsymbol{x} = \boldsymbol{A}^{\mathsf{H}}\boldsymbol{b}$$

Regularization can also be added.

Conjugated Gradient

Algorithm 1 Conjugated Gradient Algorithm

$\boldsymbol{r}_0 \leftarrow \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$

$\boldsymbol{v}_0 \leftarrow \boldsymbol{r}_0$

for $k = 0, \ldots, N-1$ **do**

  $\boldsymbol{z}_k \leftarrow \boldsymbol{A}\boldsymbol{v}_k$

  $\alpha_k \leftarrow \frac{\boldsymbol{r}_k^{\mathsf{H}} \boldsymbol{r}_k}{\boldsymbol{v}_k^{\mathsf{H}} \boldsymbol{z}_k}$

  $\boldsymbol{x}_{k+1} \leftarrow \boldsymbol{x}_k + \alpha_k \boldsymbol{v}_k$

  $\boldsymbol{r}_{k+1} \leftarrow \boldsymbol{r}_k - \alpha_k \boldsymbol{z}_k$

  $\beta_k \leftarrow \frac{\boldsymbol{r}_{k+1}^{\mathsf{H}} \boldsymbol{r}_{k+1}}{\boldsymbol{r}_k^{\mathsf{H}} \boldsymbol{r}_k}$

  $\boldsymbol{v}_{k+1} \leftarrow \boldsymbol{r}_{k+1} + \beta_k \boldsymbol{d}_k$

end for

Conjugated Gradient

Remarks:

- The CG algorithm converges in (less than) N iterations, often much faster.
- The convergence directly depends on the conditioning of the system matrix $\boldsymbol{A}$. The better it is conditioned, the faster is the convergence.
- In each iteration step 4. is the expensive one $\mathcal{O}(N^2)$.
  $\Rightarrow$ Total time complexity $\mathcal{O}(N^3)$.
- If $\boldsymbol{A}$ is not stored explicitly (Fourier transform, Radon transform), the CG algorithm allows for *matrix-free* calculation of the matrix-vector products.

Medical Imaging

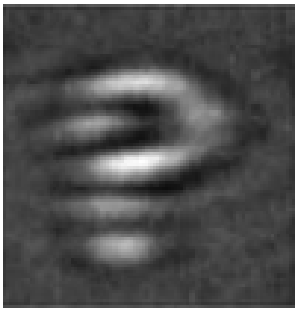
Prof. Dr. Tobias Knopp

29. November 2022

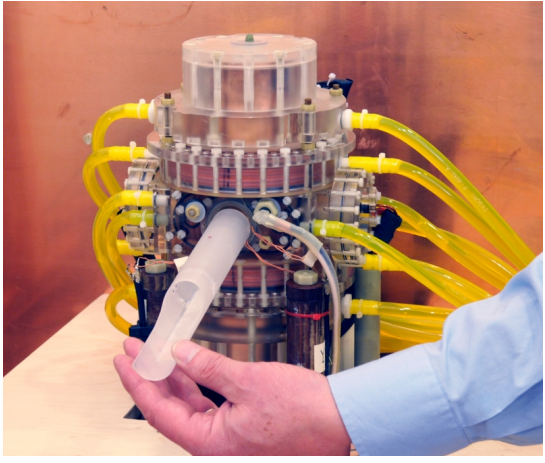
Institut für Biomedizinische Bildgebung

What is Magnetic Particle Imaging

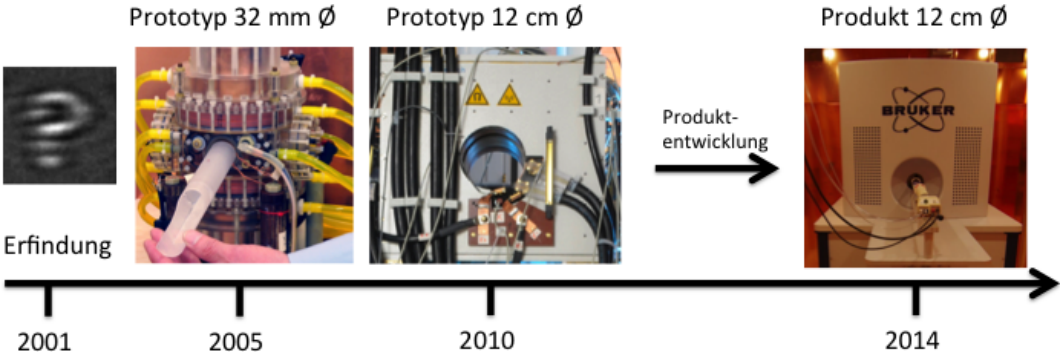
- Tomographic imaging method that allows to image super-paramagnetic nanoparticles (SPIOs)
- Invented by Bernhard Gleich in 2001 at Philips Research
- First publication: *B. Gleich and J. Weizenecker, Tomographic imaging using the nonlinear response of magnetic particles Nature. 435 30 (2005)*



First MPI Prototype



History of MPI



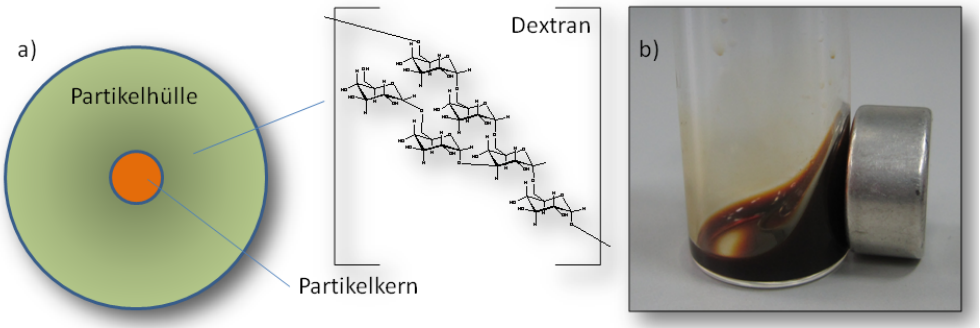
Comparison with existing Modalities

Tabelle 1: Quantitative comparison of different imaging modalities.

	CT	MRI	PET	SPECT	**MPI**
spatial resolution	0.5 mm	1 mm	4 mm	10 mm	1–3 mm
acquisition time	1 s	1 s – 1 h	1 min	1 min	< 0.1 s
sensitivity	medium	medium	very high	very high	high
quantifiability	yes	no	yes	yes	yes
harmfulness	X-ray	heating	β/γ radiation	γ radiation	heating

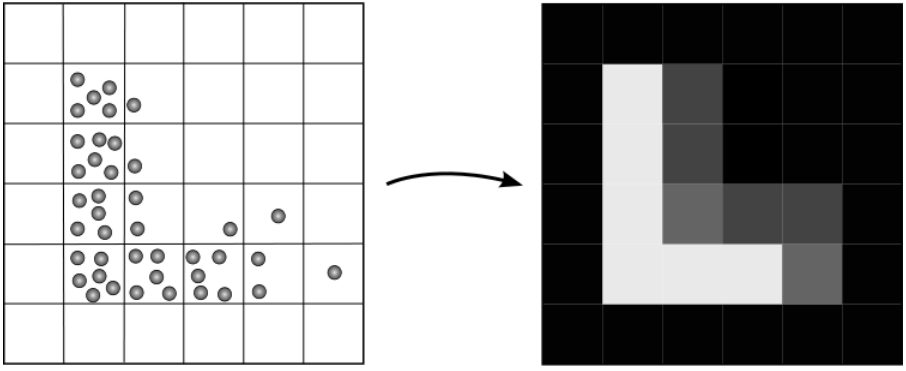
Magnetic Nanoparticles

- Particles consist of an iron-oxide core and a hull that prevents agglomeration and particle-particle interaction



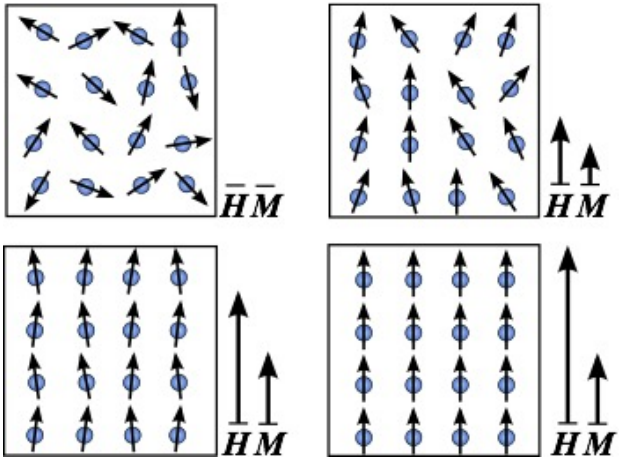
What is Magnetic Particle Imaging

- MPI images the spatially dependent concentration of SPIOs
- Concentration: Particles per Voxel



Saturation Effect

- Particles align with external magnetic field $\boldsymbol{H}$
- Saturation when all particles are aligned



Partice Magnetization

$$\boldsymbol{M} := \frac{1}{\Delta V} \sum_{j=0}^{N^{\mathrm{P}}-1} \boldsymbol{m}_j \tag{1}$$

where $\boldsymbol{m}_j$ are the magnetic moments within a voxel.

Under equilibrium assumptions $\boldsymbol{M}$ can be expressed as

$$\boldsymbol{M}(\boldsymbol{H}) = M(H)\boldsymbol{e}_{\boldsymbol{H}}, \tag{2}$$

where $\boldsymbol{e}_{\boldsymbol{H}}$ is the direction of the magnetic field and

$$M(H) = c\, m\mathcal{L}\left(\beta H\right) \tag{3}$$

is the length of the magnetization vector in dependence of the strength of the magnetic field $H := \|\boldsymbol{H}\|_2$.

Particle Magnetization

$M(H)$ depends on the Langevin function

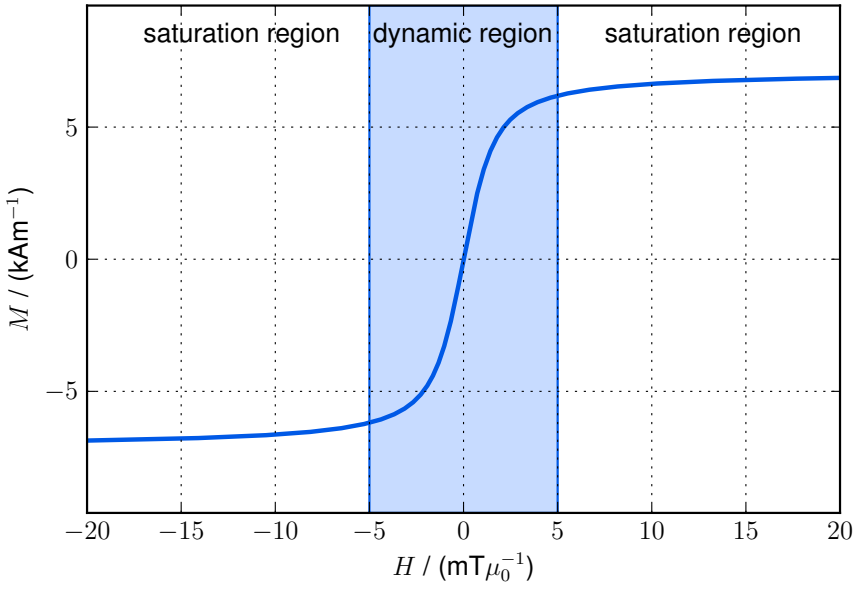
$$\mathcal{L}(\xi) := \begin{cases} \coth(\xi) - \dfrac{1}{\xi} & \xi \neq 0 \\ 0 & \xi = 0 \end{cases} \tag{4}$$

and the scaling factor

$$\beta := \frac{\mu_0 m}{k_\mathrm{B} T^\mathrm{P}}. \tag{5}$$

μ_0 is the permeability of free space, k_B is the Boltzmann constant, T^P is the particle temperature, and $m = V_\mathrm{core} M^\mathrm{S}_\mathrm{core}$ is the magnetic moment of a single particle. The latter is determined by the saturation magnetization of the material $M^\mathrm{S}_\mathrm{core}$ of which the particle core is made (usually magnetite) and the particle core volume $V_\mathrm{core} = \frac{1}{6}\pi D^3_\mathrm{core}$ derived from the core diameter D_core.

Particle Magnetization



Signal and Spatial Encoding

Any tomographic imaging method needs two ingredients:

- Signal Encoding
- Spatial Encoding

Signal Encoding Signal encoding describes the process that the underlying tomographic image generates some kind of signal.

Spatial Encoding Spatial encoding describes, how the spatial position of a voxel can be encoded into the signal. Usually this means to create a spatial dependency of the signal.

Signal and Spatial Encoding

Remark Signal and spatial encoding are in the end happening simultanuously. The do, however, help understanding the imaging methods conceptually.

Example In computed tomography, the signal is encoded by passing an X-ray through the object. This also partly does spatial encoding in one direction of the imaging plane.

Full spatial encoding is achieved by rotation of the gantry. This leads to the station that the signal response of a delta peak in image space yields a different fingerprint in the raw data signal, i.e. no two positions yield the same sinogram.

Signal Generation

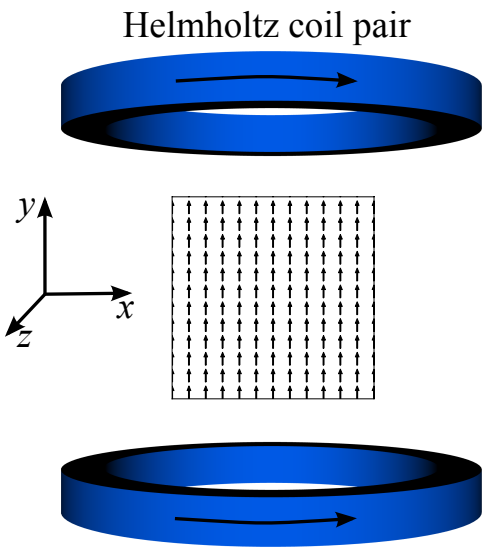
Signal generation in MPI is done by cyclic excitation of the magnetic nanoparticles using dynamic magnetic fields. To illustrate the signal generation consider a homogeneous sinusoidal drive field

$$\boldsymbol{H}^{\mathrm{D}}(t) = -A\cos(2\pi f t)\boldsymbol{e}^{\boldsymbol{H}}, \tag{6}$$

with field amplitude A, frequency f, and field direction $\boldsymbol{e}^{\boldsymbol{H}}$. If an ensemble of Langevin particles is excited by this field it generates the signal

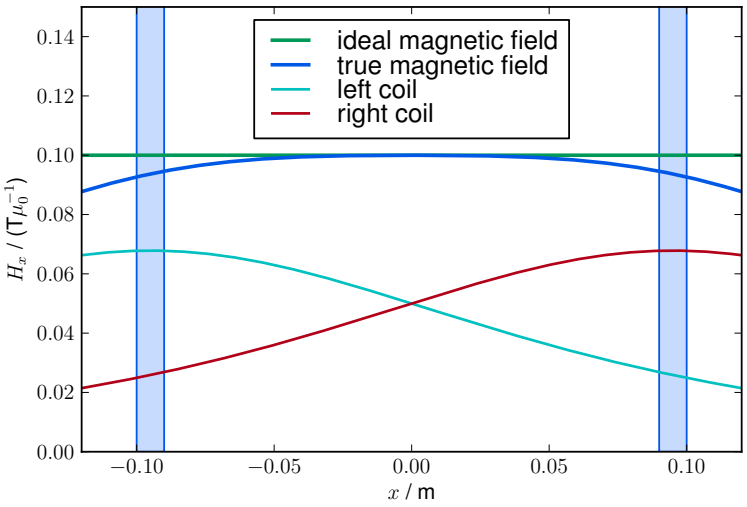
$$\boldsymbol{M}(t) = \boldsymbol{M}(\boldsymbol{H}^{\mathrm{D}}(t)), \tag{7}$$

- A homogeneous field can be generated by two coils with currents flowing in the same direction.

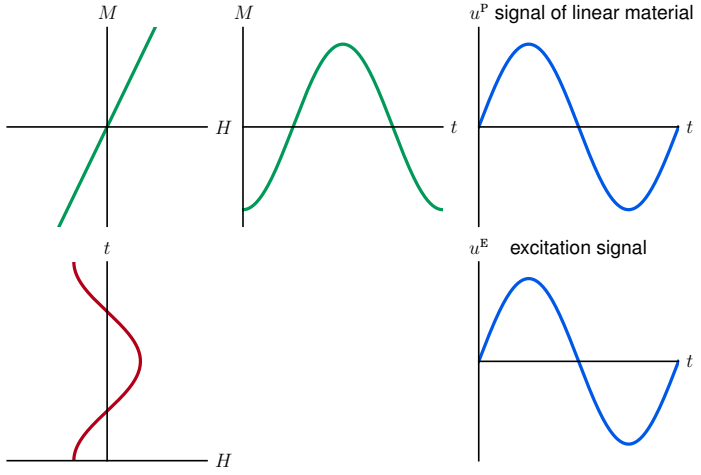


Signal Encoding

Magnetic field is not perfectly homogenous

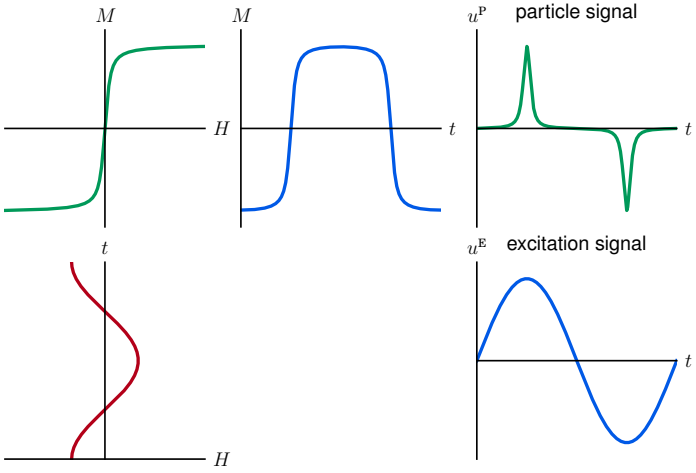


Signal Encoding



Linear material $\rightarrow$ excitation signal and particle signal cannot be distinguished.

Signal Encoding



Non-linear material $\rightarrow$ excitation signal and particle signal **can** be distinguished.

MPI Imaging Equation – Frequency Space

The voltage signal $u(t)$ is periodic and allows us to expand the voltage signal $u(t)$ into a Fourier series:

$$u(t) = \sum_{k=-\infty}^{\infty} \hat{u}_k \mathrm{e}^{2\pi \mathrm{i} k t / T}$$

and the spectrum consists of discrete lines at multiples of the frequency $f = 1/T$, which is also called the fundamental or base frequency. These multiples

$$f_k = kf, k \in \mathbb{Z} \tag{8}$$

are usually called harmonic frequencies or just harmonics. The Fourier coefficients can be computed by

$$\hat{u}_k = \frac{1}{T} \int_0^T u(t) \mathrm{e}^{-2\pi \mathrm{i} k f t} \mathrm{d}t, \quad k \in \mathbb{Z}. \tag{9}$$

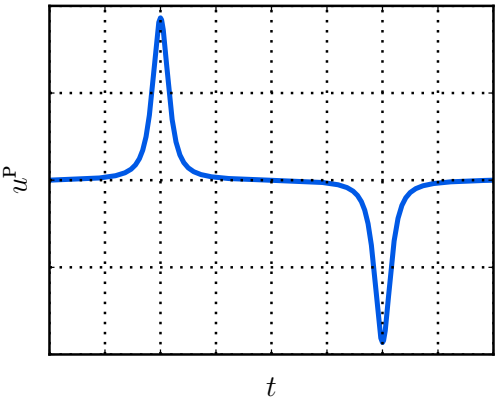
MPI Imaging Equation – Frequency Space

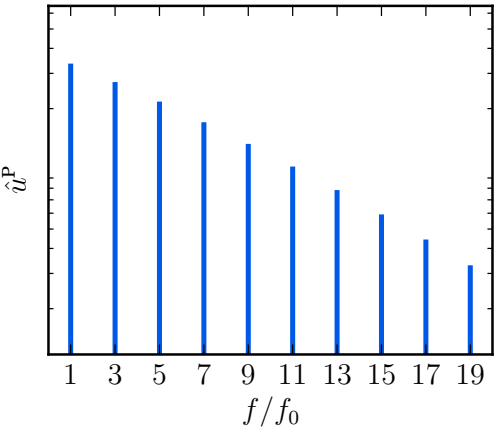
As the induced voltage is real, the Fourier coefficients obey the relation

$$\begin{aligned}\hat{u}_k &= \frac{1}{T}\int_0^T u(t)\mathrm{e}^{-2\pi\mathrm{i}kft}\mathrm{d}t \\ &= \frac{1}{T}\int_0^T \left(u(t)\mathrm{e}^{2\pi\mathrm{i}kft}\right)^* \mathrm{d}t \\ &= \left(\hat{u}_{-k}\right)^* .\end{aligned} \tag{10}$$

Therefore, one usually neglects the negative frequencies in MPI as they do not carry any additional information.

Signal Encoding





The generation of higher harmonics for a non-linear magnetization curve can be mathematically described by expanding the Langevin function into a Taylor series

$$\mathcal{L}(\xi) = \frac{1}{3}\xi - \frac{1}{45}\xi^3 + \frac{2}{954}\xi^5 - \frac{1}{4725}\xi^7 + \dots \,. \tag{11}$$

If one considers the particle magnetization M, one can see that the argument $\frac{\mu_0 H m}{k_B T^P}$ is applied to the Langevin function. For a sinusoidal field excitation $H(t) = -A\cos(2\pi f t)$, the dynamic part of the particle magnetization can be written as

$$\mathcal{L}(\tilde{\xi}\cos(2\pi f t)) = \frac{\tilde{\xi}}{3}\cos(2\pi f t) - \frac{\tilde{\xi}^3}{45}\cos^3(2\pi f t) + \dots \,, \tag{12}$$

where $\tilde{\xi} = -\frac{\mu_0 A m}{k_B T^P}$.

Using the trigonometric formula

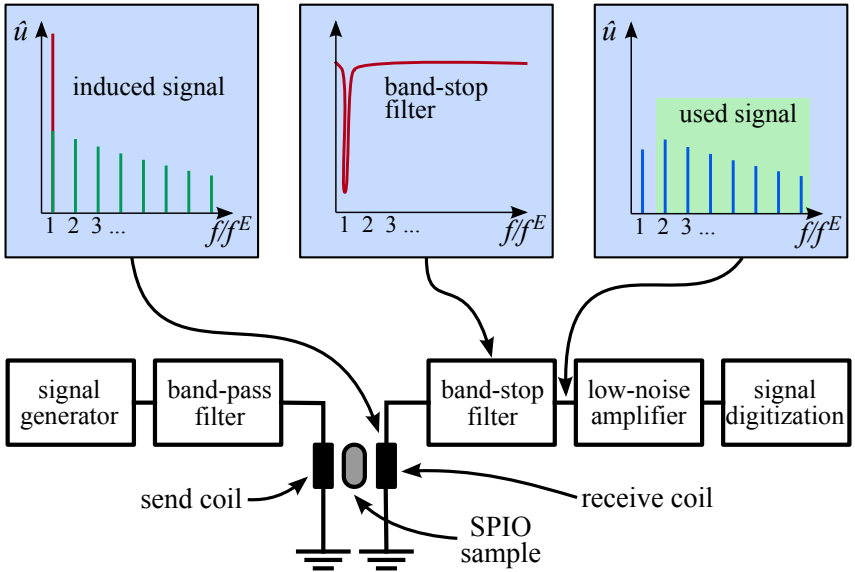
$$\cos^3(x) = \frac{1}{4}\left(3\cos(x) + \cos(3x)\right), \tag{13}$$

one obtains

$$\begin{aligned}\mathcal{L}(\tilde{\xi}\cos(2\pi ft)) &= \frac{\tilde{\xi}}{3}\cos(2\pi ft) - \frac{\tilde{\xi}^3}{60}\cos(2\pi ft) + \frac{\tilde{\xi}^3}{180}\cos(2\pi(3f)t) + \dots \\ &= \frac{20\tilde{\xi} - \tilde{\xi}^3}{60}\cos(2\pi ft) + \frac{\tilde{\xi}^3}{180}\cos(2\pi(3f)t) + \dots\,. \end{aligned} \tag{14}$$

Hence, the third harmonic, which corresponds to the frequency $3f$ is present in the spectrum of the induced voltage for a sinusoidal excitation. By including higher order terms $\cos^5$, $\cos^7$, ..., one can verify that all odd harmonics are present in the signal spectrum. The even harmonics are missing, as all even derivatives of the Langevin function have a zero-crossing at the point $\xi = 0$, at which the Taylor series is expanded.

Analog Signal Chain



Spatial Encoding

Recall at this point that $\boldsymbol{H}^{\mathrm{D}}$ is homogeneous and thus all particles in space behave the same.

What we do next is to superimpose a second magentic field $\boldsymbol{H}^{\mathrm{S}}(\boldsymbol{r})$, which is static but spatially dependent:

$$\boldsymbol{H}_{\mathrm{S}}(\boldsymbol{r}) = \begin{pmatrix} G_x & 0 & 0 \\ 0 & G_y & 0 \\ 0 & 0 & G_z \end{pmatrix} \boldsymbol{r} \tag{15}$$

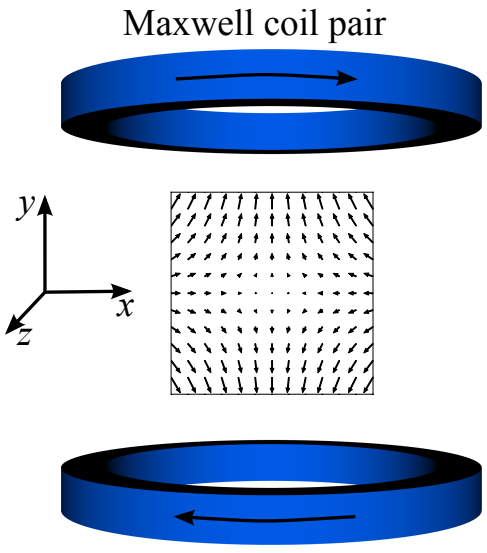
The effective excitation signal

$$\boldsymbol{H}(\boldsymbol{r}, t) = \boldsymbol{H}^{\mathrm{D}}(t) + \boldsymbol{H}^{\mathrm{S}}(\boldsymbol{r}) \tag{16}$$

is thus unique at each spatial position.

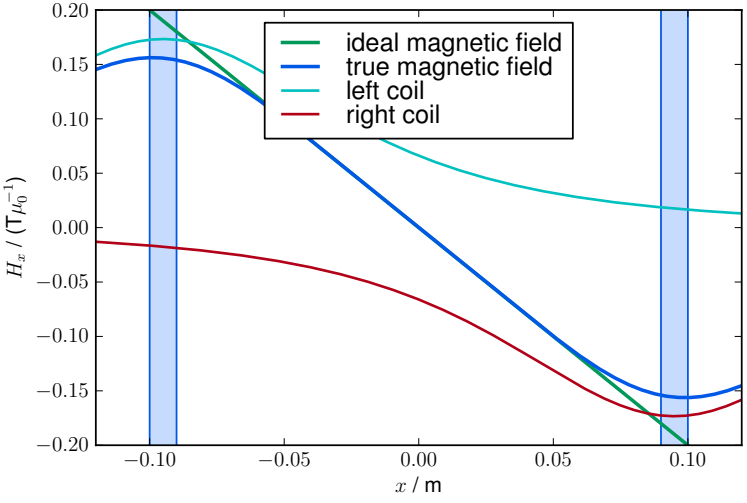
Spatial Encoding

- The gradient field $\boldsymbol{H}^{\mathrm{S}}$ has a field-free point in the center.
- The field increases in all directions in space.
- It can be generated using two coils and current flowing in opposing directions.

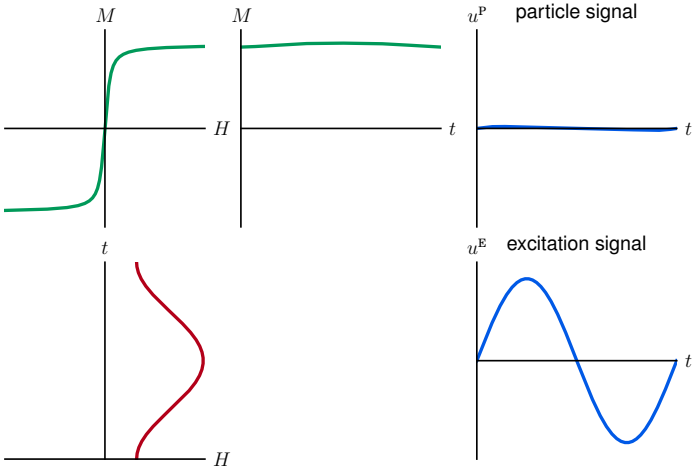


Signal Encoding

Magnetic field is not perfectly linear



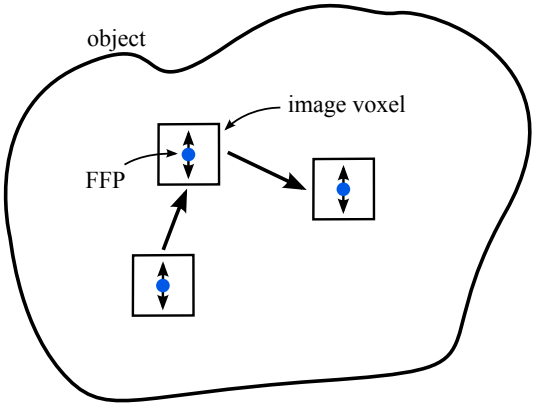
Spatial Encoding



Large offset supresses signal.

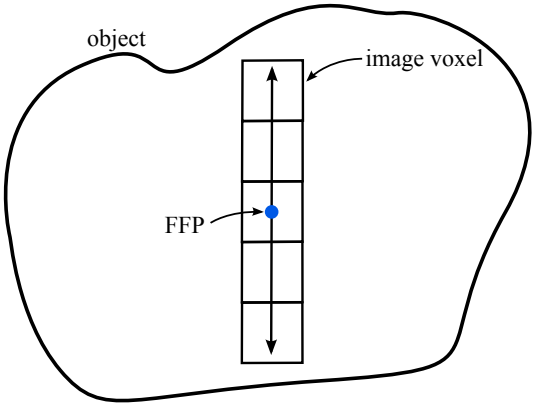
Signal Encoding

Single-voxel imaging – not very effective



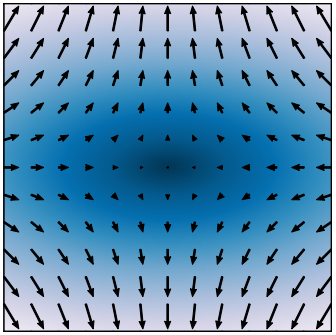
Signal Encoding

Line imaging – much more effictive

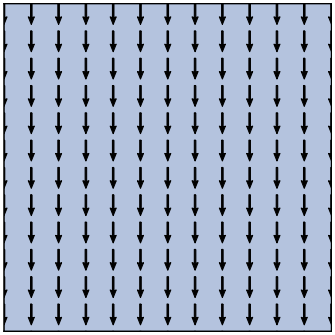


FFP Shift

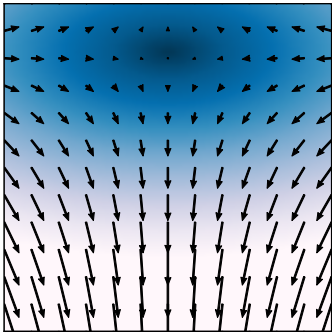
selection field



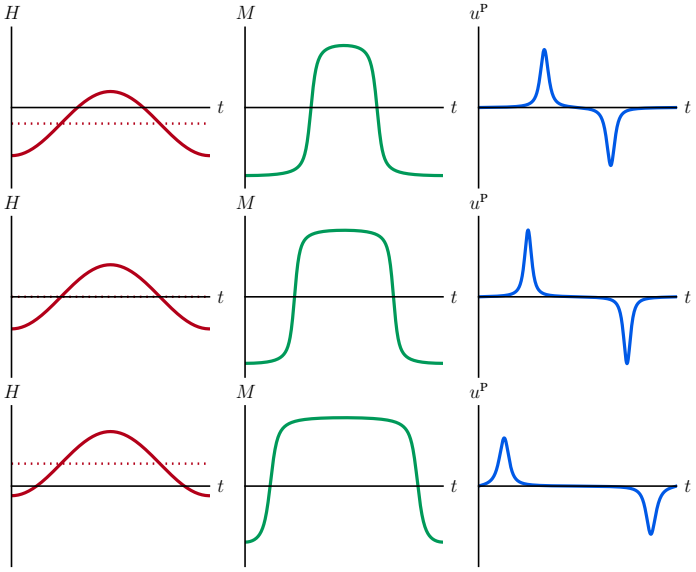
drive field



superposition



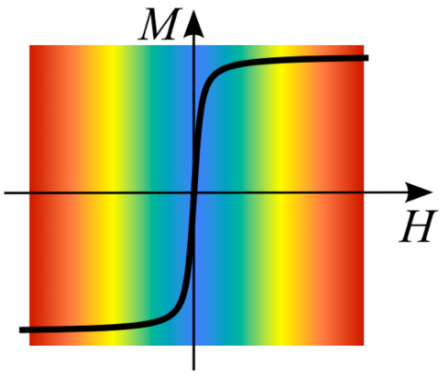
Spatial Encoding



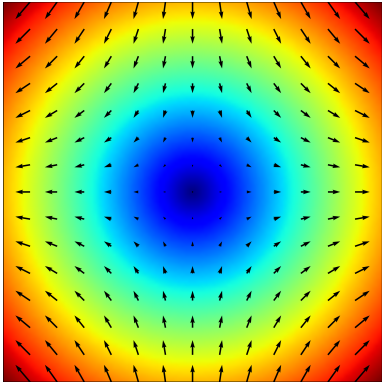
- Each point in space generates a different signal
- Basically the signal peak is shifted

Spatial Encoding

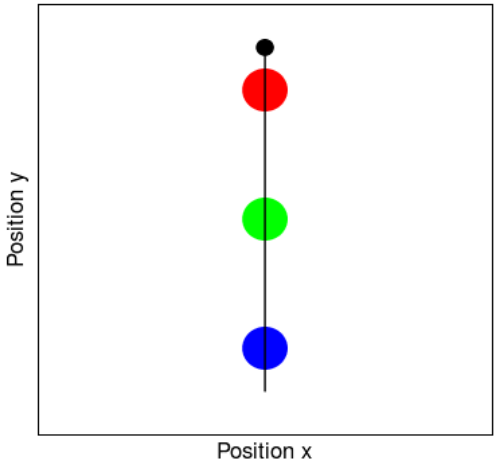
Particle Magnetization

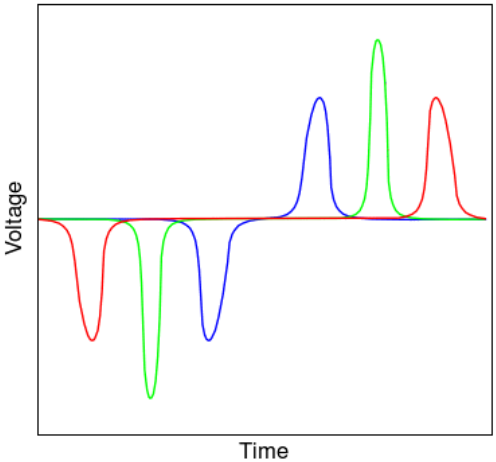


Gradient Field

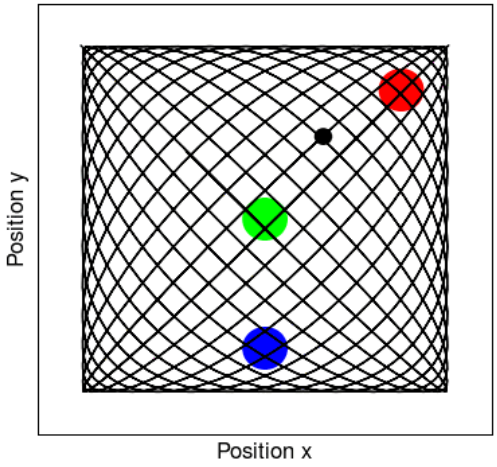


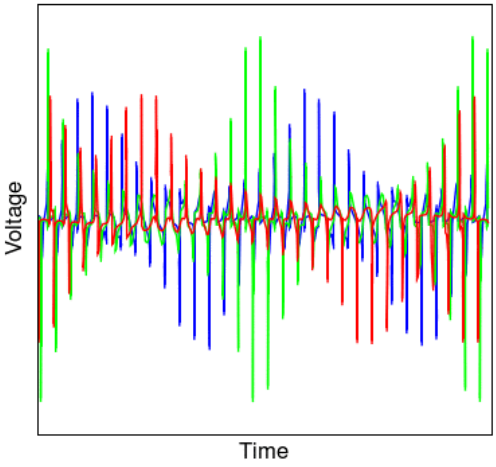
1D Imaging Sequence



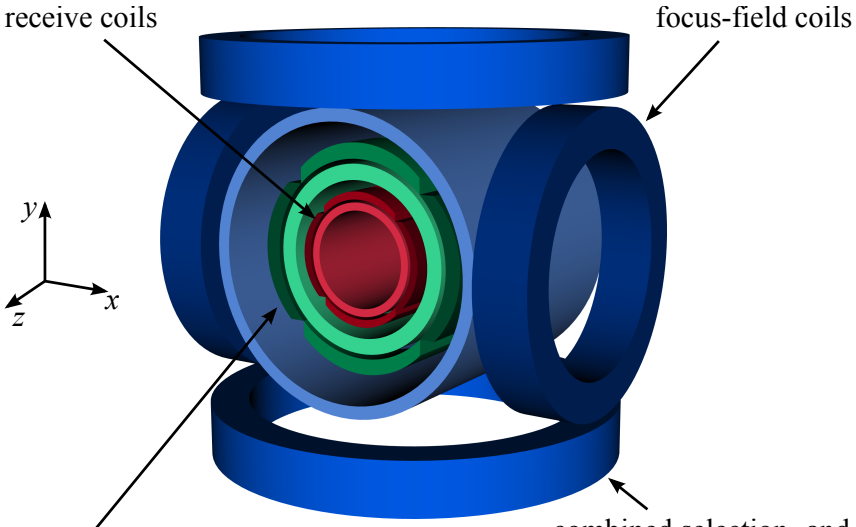


2D Imaging Sequence





MPI Coil Setup



Theorem
The relation between the particle distribution c and the voltage u induced in a receive coil with sensitivity $\boldsymbol{p}$ is linear and can be expressed as

$$u(t) = \int_{\mathbb{R}^3} s(\boldsymbol{r}, t) c(\boldsymbol{r}) d^3 r, \tag{17}$$

where

$$s(\boldsymbol{r}, t) = -\mu_0 \boldsymbol{p}(\boldsymbol{r}) \cdot \frac{\partial \overline{\boldsymbol{m}}(\boldsymbol{r}, t)}{\partial t}. \tag{18}$$

denotes the system function in time space.

Theorem
The relation between the particle distribution c and the frequency components of the induced voltage $\hat{u}_k$ is linear and can be expressed as

$$\hat{u}_k = \int_{\mathbb{R}^3} \hat{s}_k(\boldsymbol{r}) c(\boldsymbol{r}) d^3 r. \tag{19}$$

where

$$\hat{s}_k(\boldsymbol{r}) = -\frac{\mu_0}{T} \int_0^T \boldsymbol{p}(\boldsymbol{r}) \cdot \frac{\partial \overline{\boldsymbol{m}}(\boldsymbol{r}, t)}{\partial t} e^{-2\pi \mathrm{i} k t / T} dt \tag{20}$$

denotes the system function in frequency space.

Discrete MPI Imaging Equation

Sampling of time and space leads to

Discrete Setting

$$\hat{u}_k = \sum_{n=1}^{N} s_{k,n} c_n \quad \Leftrightarrow \boldsymbol{u} = \boldsymbol{S}\boldsymbol{c}$$

where

$$\begin{aligned} k &\in I_K, \\ I_K &= \{1, \dots, K\}, \\ \boldsymbol{u} &= (u_k)_{k \in I_K}, \\ \boldsymbol{c} &= (c_n)_{n \in I_N}, \\ \boldsymbol{S} &= (s_{k,n})_{k \in I_K; n \in I_N} \end{aligned}$$

Sampling of Space

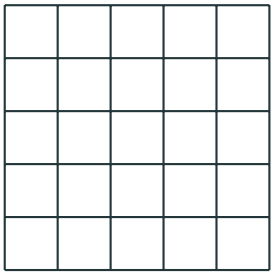
Remark

- The sampling positions $\boldsymbol{r}_n$, $n \in I_N$ represent a 2D / 3D grid. E.g.

$$\boldsymbol{r}_n = \boldsymbol{r}_{n_x, n_y, n_z}$$

  for $n_d \in I_{N_d}$, $d = x, y, z$ and $N = N_x N_y N_z$.

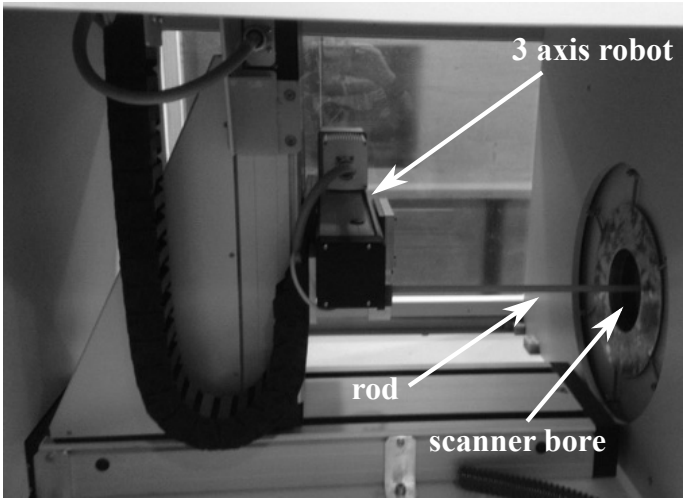
- Thus, one row of the system matrix $\boldsymbol{S}$ also represents an image (in 2D) or volume (in 3D).



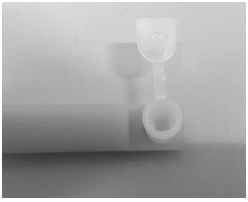
How to determine the System Matrix

- Physical modeling of $\boldsymbol{S}$ is challenging (Relaxation effects, unknown parameters).
- Therefore, system matrix $\boldsymbol{S}$ is usually explicitly measured using a robot.
- The delta sample is a voxel filled with MNP and can be mathematically represented as a unit vector $\boldsymbol{e}_j$ where $j \in I_N$.
- Since $\boldsymbol{S}\boldsymbol{e}_j = \boldsymbol{u}_j = \boldsymbol{S}_{\cdot,j}$ the calibration measurement picks the j-th column of $\boldsymbol{S}$.

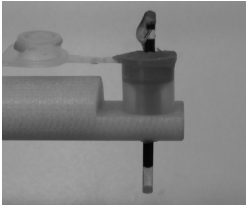
How to determine the System Matrix



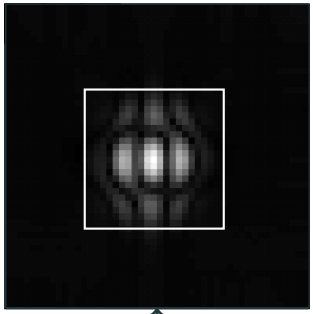
3D delta sample

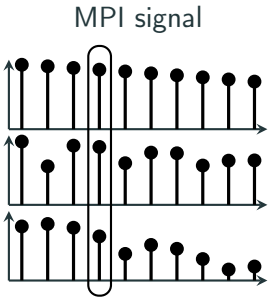


2D delta sample



MPI System Matrix

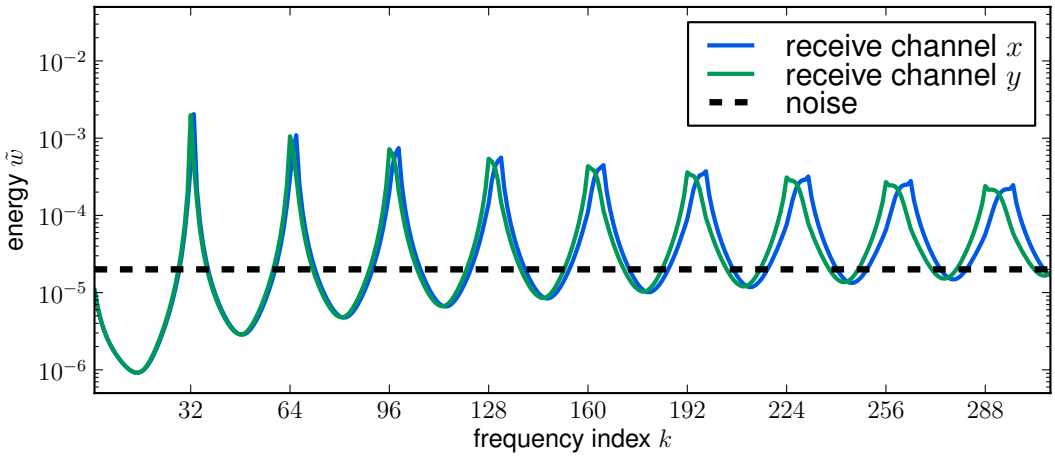




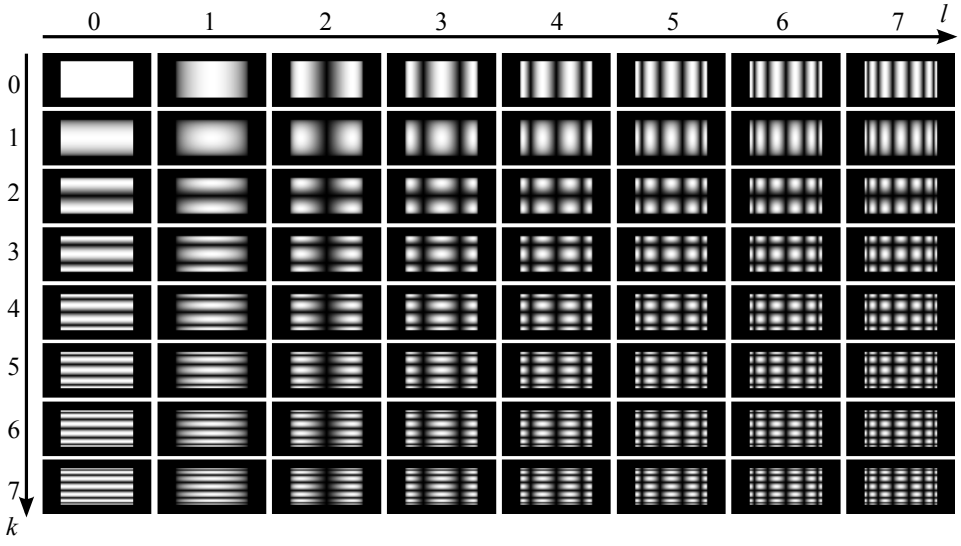
Rahmer et al. *BMC Med. Imag.* **9** (2009).

MPI System Matrix (2D)

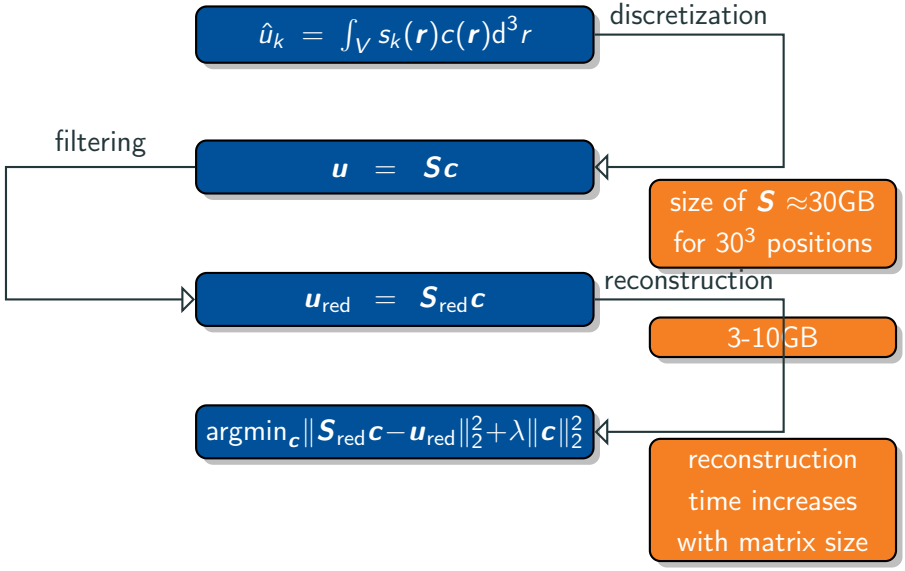
MPI System Matrix Row Energy (2D)



Tensor products of Chebyshev polynomials



Setting - Algebraic Reconstruction



Summary

- MPI is a tracer based imaging method exploiting the non-linear magnetization behavior of magnetic nanoparticles
- It applies different magnetic fields to achieve signal and spatial encoding
- Image reconstruction is done by solving a linear system of equations using regularization methods

Medical Imaging

Prof. Dr. Tobias Knopp

6. Dezember 2022

Institut für Biomedizinische Bildgebung

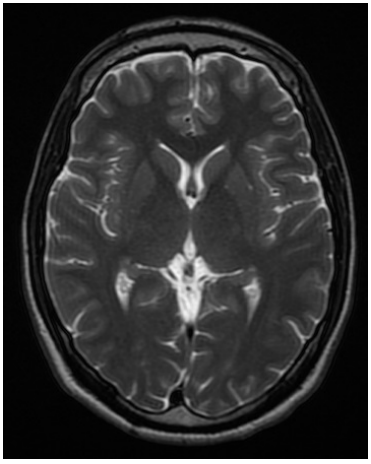
Magnetic Resonance Imaging

Magnetic Resonance Imaging

- Tomographic imaging technique (usually 3D)
- Very good soft tissue contrast (CT just bones)
- No ionizing radiation
- Very flexible: allow generating different imaging constrast by modification of the imaging protocol
- In most cases one images the distribution of hydrogen in the human body

Magnetic Resonance Imaging

Left: Picture of a modern 3T MRI system. Right: Picture of a brain MRI scan.

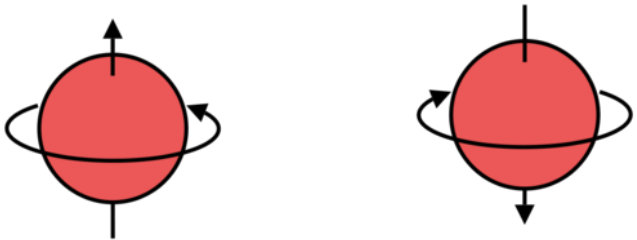


History

- 1946: Discovery of the magnetic resonance principle by Bloch and Purcel (nobel price 1952)
- 1973: First tomographic image by Lauterbur (nobel price 2003)
- since 1984: MRI in clinical routine human body

Basic Principle

Hydrogen atoms have a so-called *nuclear spin* leading to a magnetic dipole moment:

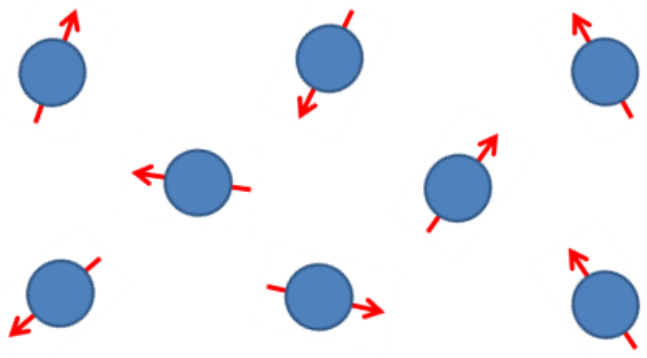


A classical picture would be a rotating atom, which establishes a magnetic moment $\boldsymbol{m}$.

Remark: Precise description of MR physics requires quantom mechanics, which is beyond the scope of this lecture

Basic Principle

Without an external magnetic field the magnetic moments have no preferred direction and follow a Boltzmann statisic (thermodynamic equilibrium):



Magnetization

The magnetization is the (vectorial) sum of all individual magnetic moments relating to a small volume element ΔV:

$$\boldsymbol{M} = \frac{1}{\Delta V} \sum_{j=0}^{N-1} \boldsymbol{m}_k \tag{1}$$

Due to the missing preferred direction of the nuclear spins the hydrogen atoms do not yield a measureable magnetization:

$$\Rightarrow \boldsymbol{M} = 0$$

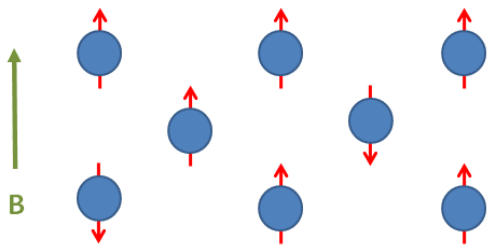
B_0 Field

When applying a static (homogeneous) magnetic field

$$\boldsymbol{B}_0 = B_0 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

the nuclear spins align with the magnetic field.

The magnetic moments either align in a parallel or in an anti-parallel way.



B_0 Field

If both spin states would occur equally often, no macroscopic magnetization could be observed. However, fortunately, the state spin up occurs about $10^{-6} \times B_0$ more often than the state spin down.

$\Rightarrow$ the stronger B_0 the more spins are in the state spin up

Consequently, one observes a macroscopic magnetization that is aligned in z direction:

$$\boldsymbol{M} = M_0 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

Signal Encoding

Signal Encoding

A static magnetization is difficult to measure. In particular one cannot distinguish between the magnetization $\boldsymbol{M}$ and the applied field $\boldsymbol{B}_0$. On thus needs a way to make $\boldsymbol{M}$ and $\boldsymbol{B}_0$ somehow different.

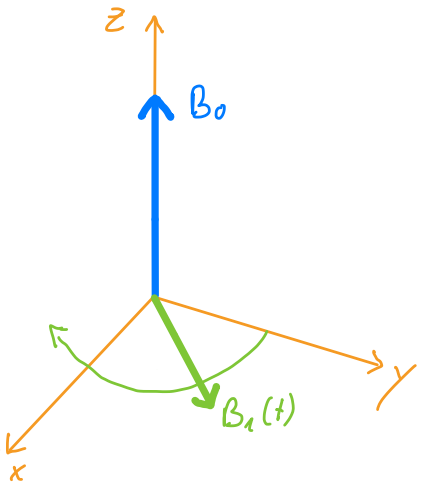
Ideas:

- Let $\boldsymbol{M}$ point in a different direction than $\boldsymbol{B}_0$
- Make $\boldsymbol{M}$ time-dependent, since this allows use an inductive measurement

Signal Encoding

To bring the magnetization into the xy plane, one applies a radio frequency field $\boldsymbol{B}_1(t)$ that is orthogonal to $\boldsymbol{B}_0$ and rotates in the xy plane. $\boldsymbol{B}_1$ has an angular frequency ω_E that matches the resonance frequency ω_L of the nuclear spins.

$$\boldsymbol{B}_1 = B_1 \begin{pmatrix} \cos(\omega_\mathrm{E} t) \\ \sin(\omega_\mathrm{E} t) \\ 0 \end{pmatrix}$$



Lamor Frequency and Gyromagnetic Ratio

The angular velocity ω_L of the magnetization depends on the applied field strength B_0 and the gyromagnetic ratio γ.

$$\omega_L = \gamma B_0$$

ω is named the *Lamor frequency*.

Gyromagnetic Ratio

γ depends on the considered matter:

Proton	γ [MHz / T]
^{1}H	42.45
^{13}C	10.71
^{19}F	40.08
^{13}N	11.27
^{31}P	17.25

In MRI usually only the hydrogen atom is considered since the human consists of 65% water (H_2O).

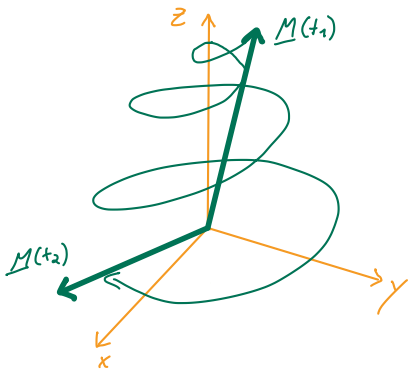
Progression of Magnetization

Due to the 90° excitation, the magnetization is elongated in a spiral movement into the xy plane. This happens despite $B_1 \ll B_0$ since the frequency of the B_1 field matches the resonance frequency of the spins.

Thus, a magnetization

$$\boldsymbol{M}(t) = M_0 \begin{pmatrix} \cos(\omega_E t) \\ \sin(\omega_E t) \\ 0 \end{pmatrix}$$

is established.



Relaxation

The varying magnetization can be measured using electromagnetic coils (induction principle).

However, the induced signal is shadowed by the inductively coupling signal of the rotating B_1 field.

$\Rightarrow$ Switch off B_1 field

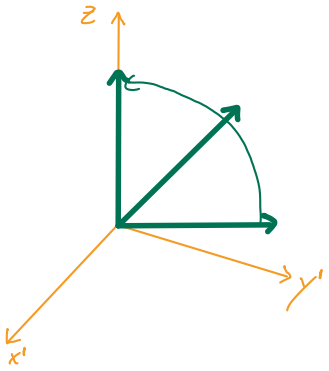
After switch off, the magnetization tries to align with the B_0 field, which can be described by two different relaxation processes.

We consider on the following two slides a rotating coordinate system where the x' and y' coordinate rotate with the magnetization around the z axis and thus the magnetization would point would be static within the xy plane if no relaxation would occur.

Longitudinal Relaxation

Increase of the z component of the magnetization

$$M_z = M_0(1 - \mathrm{e}^{-\frac{t}{T_1}})$$

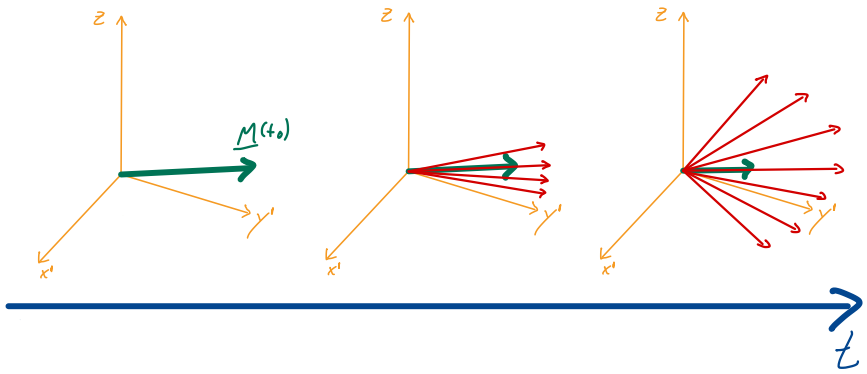


Transversal Relaxation

Dephasing of the magnetic moments of the spins

$$M_{xy} = M_0 \mathrm{e}^{-\frac{t}{T_2}}$$

$\Rightarrow$ After switching of the B_1 field one can measure for a certain amount of time a magnetization signal in a receive coil.



Magnetization after Excitation

Taking into account both relaxation processes, the magnetization after switching of the B_1 field is given by

$$\boldsymbol{M}(t) = M_0 \begin{pmatrix} \cos(\omega_E t) \mathrm{e}^{-t(\frac{1}{T_1}+\frac{1}{T_2})} \\ \sin(\omega_E t) \mathrm{e}^{-t(\frac{1}{T_1}+\frac{1}{T_2})} \\ 1 - \mathrm{e}^{-\frac{t}{T_1}} \end{pmatrix}$$

Remarks

- At this point do not distinguish between the relaxation times T_2^* and T_2 to keep the explanation simple. But note that what we described on the previous slide is actually called T_2^* ($\frac{1}{T_2^*} = \frac{1}{T_2} + \frac{1}{T_{\text{inhom}}}$ see `https://en.wikipedia.org/wiki/Relaxation_(NMR)` for details).
- T_2 times are much faster than T_1. More precisely $T_2^* \leq T_2 \leq T_1$.
- T_1 and T_2 are tissue dependent and give an additional contrast mechanism.

tissue	T_1 [ms]	T_2 [ms]
muscle	480	30
fat	190	60
gray brain matter	400	90
white brain matter	350	80

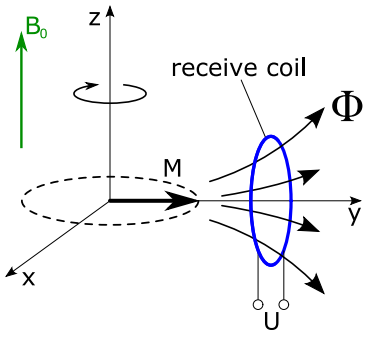
Measurement of Magnetization

Induction law

$$u(t) = -\mu_0 \frac{\mathrm{d}}{\mathrm{d}t} \int_{\mathbb{R}^3} \boldsymbol{p}(\boldsymbol{r})^{\mathsf{T}} \boldsymbol{M}(\boldsymbol{r}, t) d^3 r$$

where

- $u(t)$ is the voltage induced in the receive coil
- $\boldsymbol{p}(\boldsymbol{r})$ is the receive coil sensitivity
- $\boldsymbol{p}(\boldsymbol{r}) := \frac{\boldsymbol{B}(\boldsymbol{r})}{I}$, magnetic field at unit current (1A)

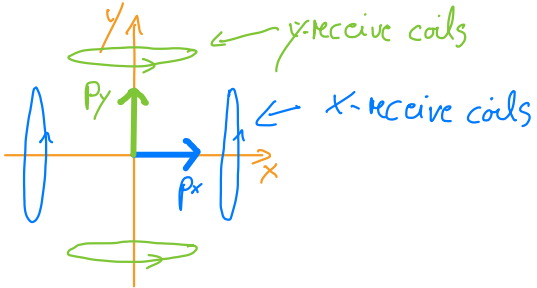


Measurement of Magnetization

Since $\boldsymbol{M}(t)$ rotates in the xy plane one uses two orthogonal receive coils with sensitivities

$$\boldsymbol{p}_x = \begin{pmatrix} p \\ 0 \\ 0 \end{pmatrix} \quad \text{and} \quad \boldsymbol{p}_y = \begin{pmatrix} 0 \\ p \\ 0 \end{pmatrix}$$

For instance two Helmholtz coil pairs can be used:



Measurement of Magnetization

In turn the induced signals are given by

$$u_x(t) = -\mu_0 p \int_{\mathbb{R}^3} \frac{\mathrm{d}}{\mathrm{d}t} M_x(\boldsymbol{r}, t) d^3 r$$

$$u_y(t) = -\mu_0 p \int_{\mathbb{R}^3} \frac{\mathrm{d}}{\mathrm{d}t} M_y(\boldsymbol{r}, t) d^3 r$$

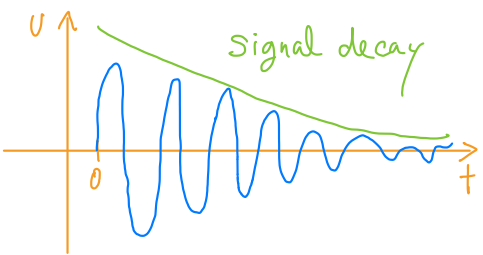
Measurement of Magnetization

If we consider for a moment only the signal from a single voxel in the center we obtain:

$$u_x(t) = -\mu_0 p \frac{\mathrm{d}}{\mathrm{d}t} \cos(\omega_\mathrm{E} t) \mathrm{e}^{-t(\frac{1}{T_1}+\frac{1}{T_2})}$$

$$u_y(t) = -\mu_0 p \frac{\mathrm{d}}{\mathrm{d}t} \sin(\omega_\mathrm{E} t) \mathrm{e}^{-t(\frac{1}{T_1}+\frac{1}{T_2})}$$

Thus, due to dephasing the signal decays exponentially. This is also called Free Induction Decay (FID).



Assumptions

We will in the following neglect relaxation effects. In practice this means that one has to measure fast enough that the dephasing did not progress too far. In practice one will apply several excitations.

In turn our model for the magnetization will be

$$\boldsymbol{M}(\boldsymbol{r}, t) = M_0(\boldsymbol{r}) \begin{pmatrix} \cos(\omega_{\mathrm{E}} t) \\ \sin(\omega_{\mathrm{E}} t) \\ 0 \end{pmatrix}$$

Signal Equation

Consequently the signal equations are given by

$$u_x(t) = -\mu_0 p \int_{\mathbb{R}^3} \frac{\mathrm{d}}{\mathrm{d}t} M_0(\boldsymbol{r}) \cos(\omega_\mathrm{E} t) d^3 r$$
$$u_y(t) = -\mu_0 p \int_{\mathbb{R}^3} \frac{\mathrm{d}}{\mathrm{d}t} M_0(\boldsymbol{r}) \sin(\omega_\mathrm{E} t) d^3 r$$

yielding

$$u_x(t) = \mu_0 p \omega_\mathrm{E} \int_{\mathbb{R}^3} M_0(\boldsymbol{r}) \sin(\omega_\mathrm{E} t) d^3 r$$
$$u_y(t) = -\mu_0 p \omega_\mathrm{E} \int_{\mathbb{R}^3} M_0(\boldsymbol{r}) \cos(\omega_\mathrm{E} t) d^3 r$$

Signal Equation

One can combine both equations by mapping the voltages on the complex plane:

$$\begin{aligned} u_{xy}(t) &= -u_y(t) + \mathrm{i}u_x(t) \\ &= \mu_0 p \omega_\mathrm{E} \int_{\mathbb{R}^3} M_0(\boldsymbol{r}) \left(\cos(t\omega_\mathrm{E}) + \mathrm{i}\sin(t\omega_\mathrm{E})\right) d^3r \\ &= \mu_0 p \omega_\mathrm{E} \int_{\mathbb{R}^3} M_0(\boldsymbol{r}) \mathrm{e}^{\mathrm{i}t\omega_\mathrm{E}} d^3r \end{aligned}$$

Spatial Encoding

Spatial Encoding

Until now all spins in space behave the same

$\Rightarrow$ No image can be obtained.

Slice Selection
Idea: not all spins in space are excited but only those within a certain slice. To this end, a *gradient* is applied *during excitation*:

$$B_z(z) = B_0 + zG_z$$

The field thus increases linearly in z direction.

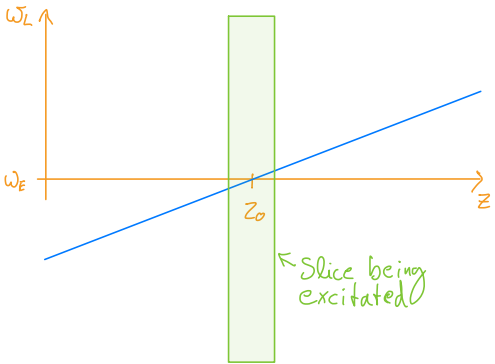
Spatial Encoding

Lamor frequency

$$\omega_L(z) = \gamma B_z(z) = \gamma(B_0 + zG_z)$$

is now slice dependent.

$\Rightarrow$ chose ω_E of the B_1 excitation such that $\omega_L(z_0) = \omega_E$ if the slice z_0 should be excited.



Spatial Encoding

The signal equation due to slice selection becomes

$$u_{xy}(t) = \mu_0 p \omega_{\mathsf{E}} \int_{\mathbb{R}^2} M_0(x, y) \mathrm{e}^{\mathrm{i} t \omega_{\mathsf{E}}} \, \mathrm{d}x \, \mathrm{d}y$$

Spatial Encoding

By slice selection all spins within a certain slice are excited.

But within the slice still all spins behave the same

$\Rightarrow$ Next step: spatial encoding within plane

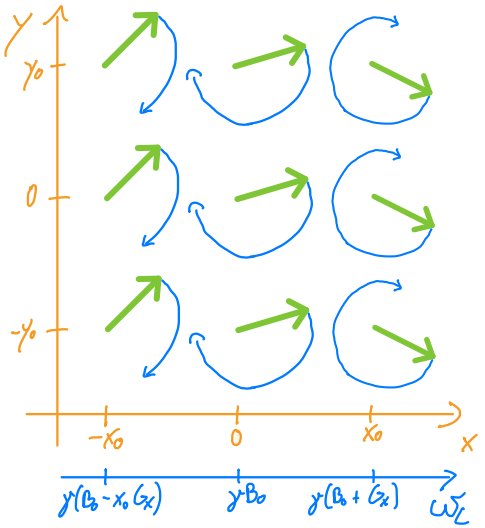
Frequency Encoding

- Change angular frequency of magnetization *during data acquisition*.
- This is done by applying a gradient field that changes the z component of the magnetic field linearly in the x direction:

$$B_z(x) = B_0 + xG_x$$

- The Lamor frequency is thus given by

$$\omega_L(x) = \gamma B_z(x) = \gamma(B_0 + xG_x) = \omega_0 + \gamma x G_x$$



Frequency Encoding

Due to frequency encoding, the frequency of the magnetization gets spatially dependent:

$$u_{xy}(t) = \mu_0 p \omega_{\mathrm{E}} \int_{\mathbb{R}^2} M_0(x, y) \mathrm{e}^{\mathrm{i}t(\omega_0 + \gamma x G_x)} \, \mathrm{d}x \, \mathrm{d}y$$

We now can pull out the carrier frequency $\mathrm{e}^{\mathrm{i}t\omega_0}$ yielding

$$u_{xy}(t) = \mu_0 p \omega_{\mathrm{E}} \mathrm{e}^{\mathrm{i}t\omega_0} \int_{\mathbb{R}^2} M_0(x, y) \mathrm{e}^{\mathrm{i}t\gamma x G_x} \, \mathrm{d}x \, \mathrm{d}y$$

Frequency Encoding

Thus, by dividing the induced signal by $\mu_0 p \omega_E e^{it\omega_0}$ one obtains

$$\tilde{u}_{xy}(t) = \frac{u_{xy}(t)}{\mu_0 p \omega_E e^{it\omega_0}} = \int_{\mathbb{R}^2} M_0(x,y) e^{it\gamma x G_x} \, dx \, dy$$

This is a Fourier integral along the x direction.

Missing: Spatial encoding in y direction.

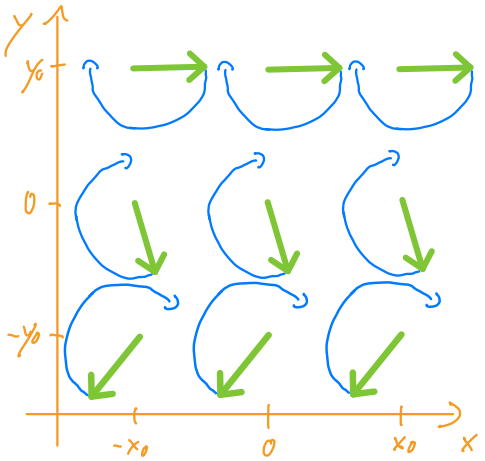
Phase Encoding

- **Idea**: Apply gradient *before* data acquisition.

$\Rightarrow$ accelerates the precession (positively and negatively) of the magnetization for a short time.

$\Rightarrow$ phase of magnetization is linearly varying in y direction.

$\Rightarrow$ spatial encoding achieved.



Phase Encoding

Since the phase encoding was applied *before* data acquisition, the magnetization is rotating with $\mathrm{e}^{\mathrm{i}(t\omega_\mathrm{E}+\phi_y)}$ where φ_y is the phase that depends on the duration and strength of the phase encoding gradient. We chose the time such that $\varphi_y = \gamma y G_y$ such that the imaging equation *during* data acquisition becomes

$$\tilde{u}_{xy}(t) = \int_{\mathbb{R}^2} M_0(x,y)\mathrm{e}^{\mathrm{i}(t\gamma x G_x + \gamma y G_y)}\,\mathrm{d}x\,\mathrm{d}y$$

If we define $k_x = \frac{\gamma G_x t}{2\pi}$ and $k_y = \frac{\gamma G_y}{2\pi}$ we obtain a regular 2D Fourier integral

$$\tilde{u}_{xy}(k_x,k_y) = \int_{\mathbb{R}^2} M_0(x,y)\mathrm{e}^{2\pi\mathrm{i}(xk_x+yk_y)}\,\mathrm{d}x\,\mathrm{d}y$$

In order to fill the entire Fourier space (also named *k-space*) several excitations with different phase encodings have to be applied.

Image Reconstruction

We have derived a special imaging equation for 2D sequences. More general, the MRI signal equation can be written as

$$s(\boldsymbol{k}) = \int_{\mathbb{R}^3} M_0(\boldsymbol{r}) \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{k}^\mathsf{T} \boldsymbol{r}} \, \mathrm{d}^3 r$$

where $\boldsymbol{r} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$ and $\boldsymbol{k} = \begin{pmatrix} k_x \\ k_y \\ k_z \end{pmatrix}$

After applying the Fourier inversion theorem on obtains

$$M_0(\boldsymbol{r}) = \int_{\mathbb{R}^3} s(\boldsymbol{k}) \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}^\mathsf{T} \boldsymbol{r}} \, \mathrm{d}^3 k$$

Remarks

- Image reconstruction thus can be done explicitly. In the discrete setting it corresponds to a matrix-vector operation that usually can be performed by the FFT in an $\mathcal{O}(N \log N)$ fashion.
- Image reconstruction in MRI is thus **not** an ill-posed inverse problem.
- In practice on often applies subsampling in which case the inverse problem again gets ill-posed.
- When chosing the gradient trajectory $\boldsymbol{k}(t)$ nonequidistantly, the FFT needs to be replaced by the NFFT, which is dicussed in the next lecture.
- Inhomogenous coils and relaxation times lead to more complicated signal modells when being taken into account.

Pulse Sequences

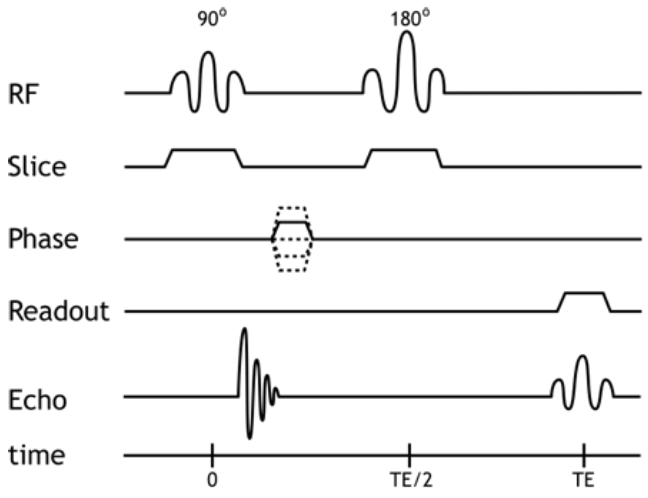
Pulse Sequences

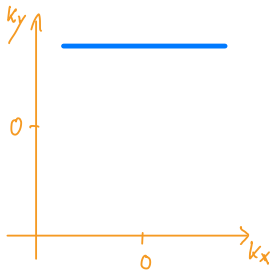
The applied dynamic magnetic fields during an MR acquisition can be expressed using a *pulse sequence diagram*.

In this lecture only a quick overview about basic pulse sequences is given.

Spin Echo Sequence

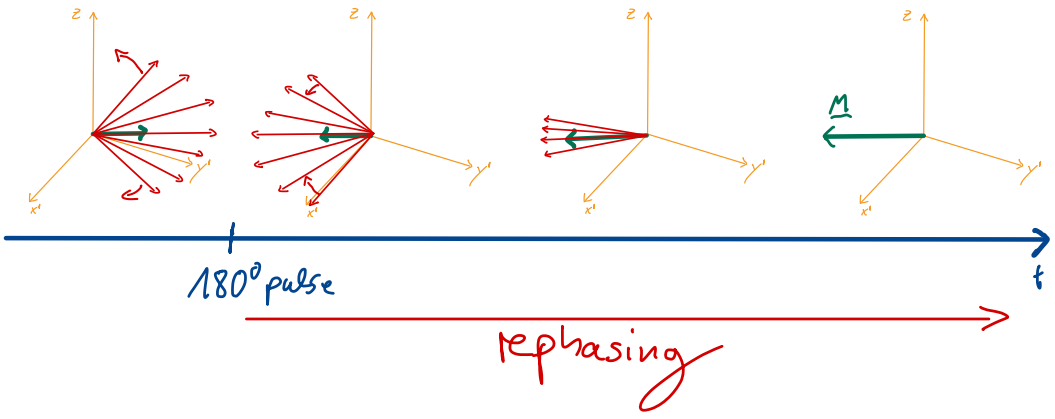
Basic *spin echo sequence*. The 180° pulses are necessary to rephase the spins.





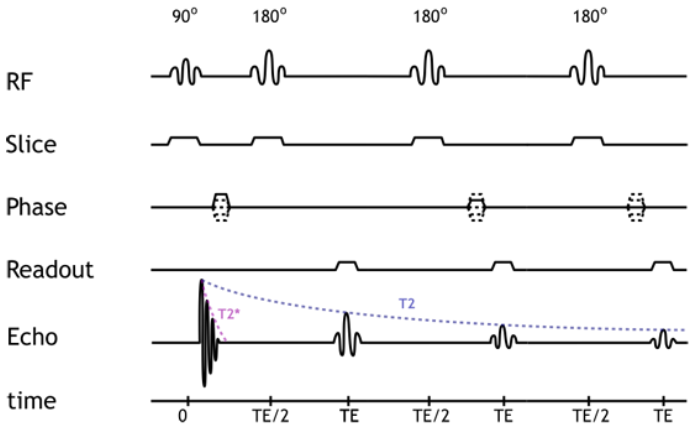
180° Pulse

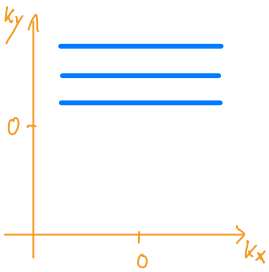
The 180° degree pulse flips the drifting (macroscopic) magnetic moments (ensembles of similar phasing magnetic moments). Due to the flip, the moments align and a measurable magnetization is established again.



Fast Spin Echo Sequence

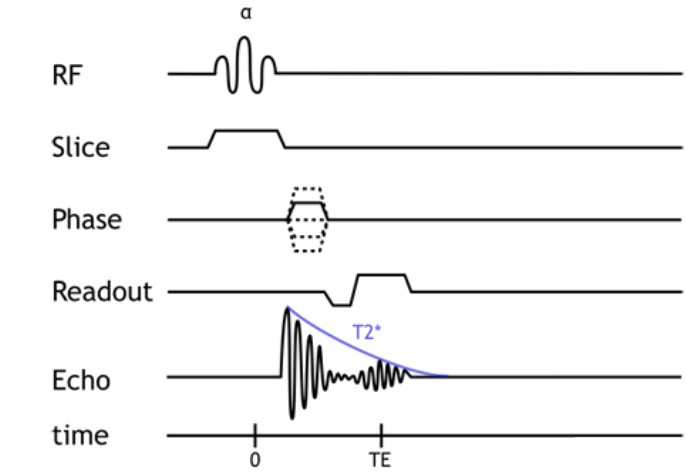
Use multiple 180 degree pulses to speed up data acquisition.

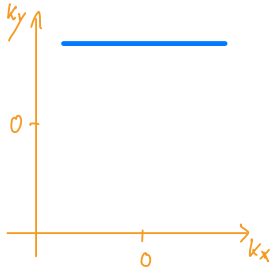




Gradient Echo Sequence

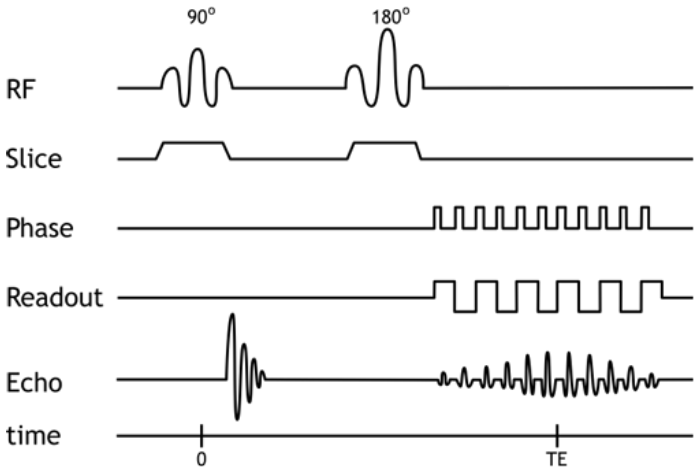
Echos can also be generated by gradients (even faster).

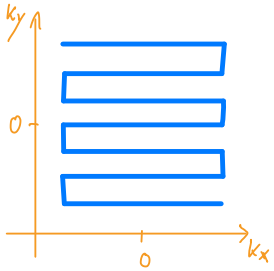




Echo Planar Imaging Sequence

One can also measure several phase encoding gradients within a single excitation.

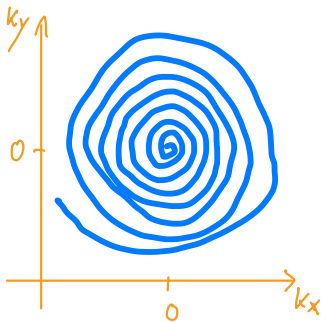


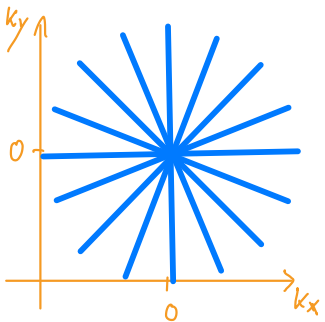


Sampling Trajectories

Nonequidistant Sampling Trajectories

Changing the gradients G_x and G_y both continuously during data acquisition allows for arbitratry k-space sampling. Beside Cartesian trajectories also non-equidistant trajectories can be applied. Spiral trajectories (left) allow for collecting more data within a single excitation. Radial trajectories (right) are robust against motion.

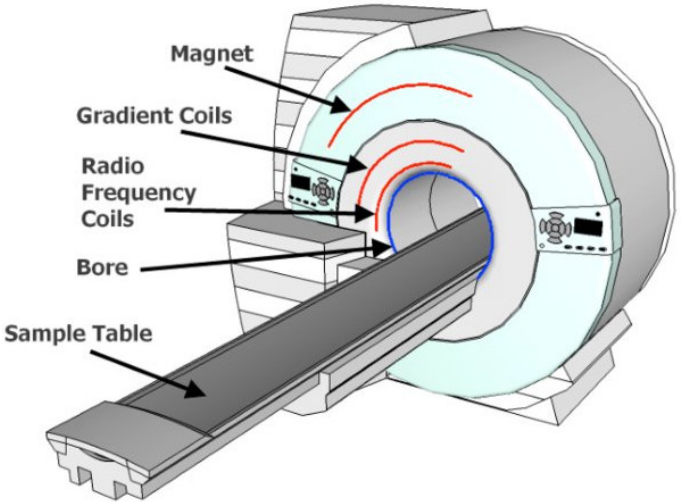




System Overview

System Overview

Schematic overview of a typical MRI scanner including the three field generators.



Summary

- MRI is a versatile imaging modality.
- It has long scan times due to sequential data collection.
- Basic image reconstruction is simple and just an FFT ($\rightarrow$ not noise amplifying).
- Nonequidistant trajectories require the NFFT.
- Subsampling and field imperfection lead to more sophisticated image reconstruction methods.

Medical Imaging

Prof. Dr. Tobias Knopp

14. Dezember 2022

Institut für Biomedizinische Bildgebung

Non-Equidistant Fast Fourier Transform

Non-Equidistant Fast Fourier Transform

The non-equidistant fast Fourier transform (NFFT) is an approximative algorithm that performs the non-equidistant discrete Fourier transform (NDFT) efficiently.

Definition
The NDFT is defined as

$$f_j = f(x_j) = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} \hat{f}_k \mathrm{e}^{-2\pi \mathrm{i} k x_j}, \quad j = 0, \ldots, M-1$$

It takes a sequence of N equidistantly distributed samples $\hat{f}_k$ and calculates the Fourier sum at M non-equidistant sampling nodes $x_j \in [-0.5, 0.5)$.

Remarks

- Naive NDFT would require $\mathcal{O}(MN)$ arithmetic operations
- The FFT requires x_j to be equidistant, i.e.

$$x_j = -\frac{1}{2} + \frac{j}{N}, \qquad j = 0, \ldots, N-1$$

Key Idea of the NFFT

The key idea of the NFFT is to approximate the *anharmonic* complex exponential $\mathrm{e}^{-2\pi \mathrm{i} k x_j}$ by a sum of harmonic complex exponentials

$$\underbrace{\mathrm{e}^{-2\pi \mathrm{i} k x_j}}_{\text{anharmonic}} \approx \sum_{l=-\frac{L}{2}}^{\frac{L}{2}-1} \beta_{l,j} \underbrace{\mathrm{e}^{-2\pi \mathrm{i} \frac{kl}{L}}}_{\text{harmonic}}$$

Remarks

- Harmonic here is means that the frequency $\gamma_l = \frac{l}{L}$ is a multiple of the basis frequency $\gamma_1 = \frac{1}{L}$.
- Similar to other approximations (or interpolations) we use a base function and shift its frequency (c.f. spline interpolation).
- Most important point of the approximation is that the two indices k and j are not in the exponent anymore but they are factorized.

Derivation of the NFFT

We consider a general window function
$\varphi \in L^1(\mathbb{R}) \cap L^2(\mathbb{R}) \cap \mathrm{BV}(\mathbb{R})$ and its periodization

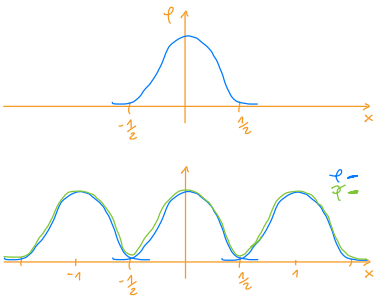
$$\tilde{\varphi}(\tilde{x}) := \sum_{p=-\infty}^{\infty} \varphi(\tilde{x} + p).$$

The function has a uniformly convergent Fourier series

$$\tilde{\varphi}(\tilde{x}) = \sum_{k=-\infty}^{\infty} c_k(\tilde{\varphi}) \mathrm{e}^{-2\pi \mathrm{i} k \tilde{x}}$$

with coefficients

$$c_k(\tilde{\varphi}) = \int_{-\frac{1}{2}}^{\frac{1}{2}} \tilde{\varphi}(\tilde{x}) \mathrm{e}^{2\pi \mathrm{i} k \tilde{x}} \, \mathrm{d}\tilde{x}$$



Note: The sign is flipped compared to the regular definition of the Fourier series.

Derivation of the NFFT

We now substitute $\tilde{x}$ by $\tilde{x} = x - x'$ with $x \in \mathbb{R}$ yielding with $\frac{\mathrm{d}\tilde{x}}{\mathrm{d}x'} = -1$

$$
\begin{aligned}
c_k(\tilde{\varphi}) &= \int_{x+\frac{1}{2}}^{x-\frac{1}{2}} \tilde{\varphi}(x - x')\mathrm{e}^{2\pi \mathrm{i} k(x-x')}(-1)\,\mathrm{d}x' \\
&\overset{\text{periodicity}}{=} \int_{\frac{1}{2}}^{-\frac{1}{2}} \tilde{\varphi}(x - x')\mathrm{e}^{2\pi \mathrm{i} k(x-x')}(-1)\,\mathrm{d}x' \\
&= \int_{-\frac{1}{2}}^{\frac{1}{2}} \tilde{\varphi}(x - x')\mathrm{e}^{2\pi \mathrm{i} k(x-x')}\,\mathrm{d}x'
\end{aligned}
$$

Derivation of the NFFT

Then we approximate the integral at equidistant nodes $\frac{l}{L}$ using a rectangular quadrature rule

$$c_k(\tilde{\varphi}) \approx \frac{1}{L} \sum_{l=-\frac{L}{2}}^{\frac{L}{2}-1} \tilde{\varphi}\left(x - \frac{l}{L}\right) \mathrm{e}^{2\pi \mathrm{i} k (x - \frac{l}{L})}$$

We will later chose $L = \alpha N$ where $\alpha > 1$ is the so-called *oversampling factor*. If $c_k(\tilde{\varphi}) \neq 0$ we obtain

$$\mathrm{e}^{-2\pi \mathrm{i} k x} \approx \frac{1}{\alpha N c_k(\tilde{\varphi})} \sum_{l=-\frac{\alpha N}{2}}^{\frac{\alpha N}{2}-1} \tilde{\varphi}\left(x - \frac{l}{\alpha N}\right) \mathrm{e}^{-2\pi \mathrm{i} k \frac{l}{\alpha N}} \tag{1}$$

Thus we can indeed approximate the anharmonic exponential by a sum of harmonic exponentials ($\beta_{l,j} = \frac{1}{\alpha N c_k(\tilde{\varphi})} \tilde{\varphi}\left(x - \frac{l}{\alpha N}\right)$).

Derivation of the NFFT

Since φ is a kernel function we know that only few of the summands in (1) have a significant contribution to the sum.

Key idea: Consider only significant summands such that the sum needs just be evaluated partially.

To this end we truncate the function φ at $\pm\frac{m}{\alpha N}$ and replace it by

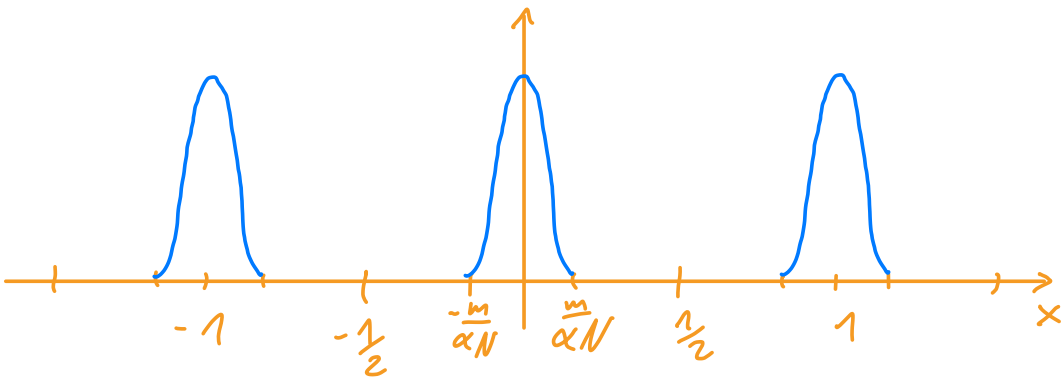
$$\psi(x) := \begin{cases} \varphi(x) & \text{if } x \in \left[-\frac{m}{\alpha N}, \frac{m}{\alpha N}\right] \\ 0 & \text{else} \end{cases}.$$

With the periodization $\tilde{\psi}(\tilde{x}) := \sum_{p=-\infty}^{\infty} \psi(\tilde{x}+p)$ this yields our final approximation

$$\mathrm{e}^{-2\pi \mathrm{i} k x} \approx \frac{1}{\alpha N c_k(\tilde{\varphi})} \sum_{l=-\frac{\alpha N}{2}}^{\frac{\alpha N}{2}-1} \tilde{\psi}\left(x - \frac{l}{\alpha N}\right) \mathrm{e}^{-2\pi \mathrm{i} k \frac{l}{\alpha N}} \tag{2}$$

Derivation of the NFFT

Illustration of the truncation:



Derivation of the NFFT

We now get back to the NDFT and replace the term $\mathrm{e}^{-2\pi \mathrm{i} k x}$ with our approximation yielding

$$
\begin{aligned}
f_j &= \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} \hat{f}_k \mathrm{e}^{-2\pi \mathrm{i} k x_j} \\
&\overset{(2)}{\approx} \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} \hat{f}_k \frac{1}{\alpha N c_k(\tilde{\varphi})} \sum_{l=-\frac{\alpha N}{2}}^{\frac{\alpha N}{2}-1} \tilde{\psi}\left(x_j - \frac{l}{\alpha N}\right) \mathrm{e}^{-2\pi \mathrm{i} k \frac{l}{\alpha N}} \\
&= \underbrace{\sum_{l=-\frac{\alpha N}{2}}^{\frac{\alpha N}{2}-1} \tilde{\psi}\left(x_j - \frac{l}{\alpha N}\right) \underbrace{\sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} \underbrace{\frac{\hat{f}_k}{\alpha N c_k(\tilde{\varphi})}}_{\text{apodization}} \mathrm{e}^{-2\pi \mathrm{i} k \frac{l}{\alpha N}}}_{\text{DFT/FFT}}}_{\text{discrete convolution}}
\end{aligned}
$$

Algorithm NFFT

Algorithm 1 Pseudocode NFFT

input: $\hat{f}_k \in \mathbb{C}$, $k = -\frac{N}{2}, \ldots, \frac{N}{2} - 1$, $x_j \in [-\frac{1}{2}, \frac{1}{2})$, $j = 0, \ldots, M-1$, $\alpha > 1$ and $m \in \mathbb{N}$

output: $f_j \in \mathbb{C}$, $j = 0, \ldots, M-1$

1: **for** $k = -\frac{N}{2}, \ldots, \frac{N}{2} - 1$ **do**

2: $\quad \hat{g}_k = \frac{\hat{f}_k}{\alpha N c_k(\tilde{\varphi})}$

3: **end for**

4: compute the data $(g_l)_{l=-\frac{\alpha N}{2}}^{\frac{\alpha N}{2}-1}$ using an FFT of $(\hat{g}_k)_{k=-\frac{N}{2}}^{\frac{N}{2}-1}$.

5: **for** $j = 0, \ldots, M-1$ **do**

6: $\quad f_j = \sum_{l=-\frac{\alpha N}{2}}^{\frac{\alpha N}{2}-1} g_l \tilde{\psi}\left(x_j - \frac{l}{\alpha N}\right)$

7: **end for**

Complexity Analysis

The three steps have an individual time complexity of

1. $\mathcal{O}(N)$
2. $\mathcal{O}(\alpha N \log(\alpha N))$
3. $\mathcal{O}(mM)$

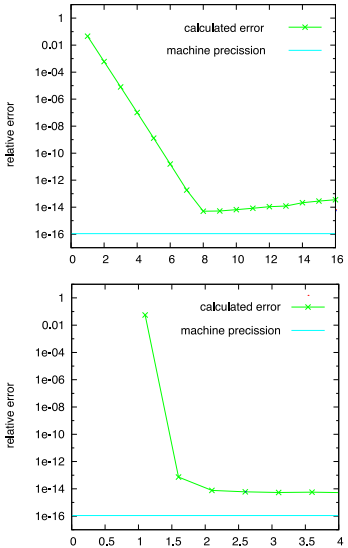
Thus, the total complexity of the NFFT is

$$\mathcal{O}(\alpha N \log(\alpha N) + mM) \ll \mathcal{O}(NM)$$

Approximation Error

One can derive approximation error estimations for the NFFT. For specific φ (i.e. Kaiser-Bessel functions) one can show that the approximation error can be adjusted to be lower than tha floating point precission (64 bit $\rightarrow \alpha = 2$ and $m = 6$ for the Kaiser-Bessel window).
Since α and m are independent of N and M we end up with an algorithmic complexity of

$$\mathcal{O}(N \log N + M) \ll \mathcal{O}(NM)$$



Matrix-Vector Notation

The NFFT can also been expressed in matrix vector notation. Let

$$\hat{\boldsymbol{f}} := (\hat{f}_k)_{k=-\frac{N}{2}}^{\frac{N}{2}-1} \in \mathbb{C}^N, \quad \boldsymbol{f} := (f_j)_{j=0}^{M-1} \in \mathbb{C}^M$$

and

$$\boldsymbol{A} := \left(\mathrm{e}^{-2\pi \mathrm{i} k x_j}\right)_{j=0,\ldots,M-1;k=-\frac{N}{2},\ldots,\frac{N}{2}-1} \in \mathbb{C}^{M\times N}.$$

then

$$\boldsymbol{f} = \boldsymbol{A}\hat{\boldsymbol{f}} \approx \underbrace{\boldsymbol{B}}_{\text{convolution matrix}} \underbrace{\boldsymbol{F}}_{\text{DFT matrix}} \underbrace{\boldsymbol{D}}_{\text{diagonal matrix}} \hat{\boldsymbol{f}}.$$

Here we note that $\boldsymbol{B}$ is a sparse matrix (i.e. has only few non-zero entries).

Implementation

The window function φ and $c_k(\tilde{\varphi})$ are usually expensive to calculate and should therefore be cached. There are two possibilities for φ

1. Create a lookup table for φ
2. Store $\boldsymbol{B}$ in a sparse matrix format (CRS / CCS)

Adjoint NFFT

In addition to the regular NFFT one often also needs the adjoint NFFT. It maps from non-equidistant samples to equidistant samples, wheras the NFFT is the other way around.

$$\hat{f}_k = \sum_{j=0}^{M-1} f_j \mathrm{e}^{2\pi \mathrm{i} k x_j} \quad k = -\frac{N}{2}, \ldots, \frac{N}{2} - 1$$

$$\approx \underbrace{\frac{1}{\alpha N c_k(\tilde{\varphi})} \underbrace{\sum_{l=-\frac{\alpha N}{2}}^{\frac{\alpha N}{2}-1} \underbrace{\left(\sum_{j=0}^{M-1} f_j \tilde{\psi}\left(x_j - \frac{l}{\alpha N} \right) \right)}_{\text{discrete convolution}} \mathrm{e}^{2\pi \mathrm{i} k \frac{l}{\alpha N}}}_{\text{DFT/FFT}}}_{\text{apodization}}$$

Adjoint NFFT

In matrix vector notation:

$$\hat{\boldsymbol{f}} = \boldsymbol{A}^{\mathrm{H}} \boldsymbol{f} \approx \boldsymbol{D}^{\mathrm{H}} \boldsymbol{F}^{\mathrm{H}} \boldsymbol{B}^{\mathrm{H}} \boldsymbol{f}$$

Window Function (Kaiser-Bessel)

There are various suitable window functions for which error estimations have been derived. The best is the Kaiser-Bessel window, which is defined as

$$\varphi(v) := \begin{cases} \frac{1}{2m} I_0\left(bm\sqrt{1-(\frac{\alpha N v}{m})^2}\right) & \text{falls} \quad |v| \leq \frac{m}{\alpha N} \quad (b := \pi(2-\frac{1}{\alpha})), \\ 0 & \text{falls} \quad |v| > \frac{m}{\alpha N} \end{cases}$$

where $I_k : \mathbb{C} \to \mathbb{C}$, $k \in \mathbb{N}_0$ is the modified Bessel function of the first kind:

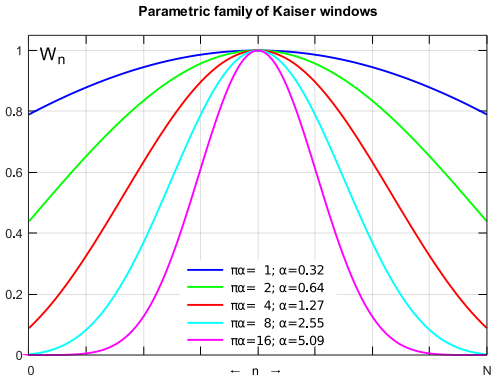
$$I_k(x) := \sum_{r=0}^{\infty} \frac{(\frac{x}{2})^{2r+k}}{(r+k)!r!}.$$

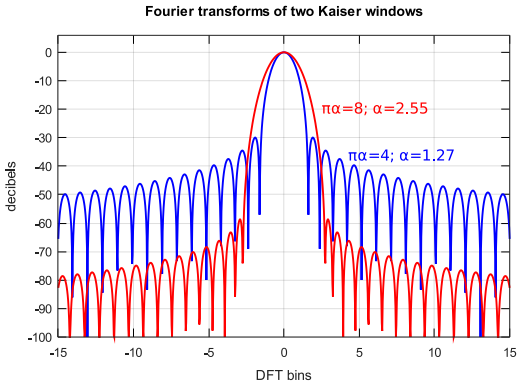
Window Function (Kaiser-Bessel)

The Fourier transform of the Kaiser-Bessel window can be shown to be

$$\begin{aligned}\hat{\varphi}(z) &= \frac{1}{\alpha N}\operatorname{sinc}\left(\sqrt{\left(\frac{2\pi m z}{\alpha N}\right)^2 - b^2 m^2}\right) \\ &= \frac{1}{\alpha N}\sum_{n=0}^{\infty}\frac{1}{(2n+1)!}\left(b^2 m^2 - \left(\frac{2\pi m z}{\alpha N}\right)^2\right)^n .\end{aligned}$$

Window Function (Kaiser-Bessel)





https://en.wikipedia.org/wiki/Kaiser_window

Multidimensional NFFT

The NDFT and the NFFT can be also formulated / derived for multidimensional signals:

NFFT

$$f_j := \sum_{\boldsymbol{k} \in I_{\boldsymbol{N}}^d} \hat{f}_{\boldsymbol{k}} \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k} \boldsymbol{x}_j}, \qquad j = 0, \dots M-1$$

Adjoint NFFT

$$\hat{f}_{\boldsymbol{k}} = \sum_{j=0}^{M-1} f_j \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{k} \boldsymbol{x}_j}, \qquad \boldsymbol{k} \in I_{\boldsymbol{N}}^d$$

where the index set $I_{\boldsymbol{N}}^d$ with $\boldsymbol{N} = (N_0, \dots, N_{d-1})^T \in \mathbb{N}^d$ is defined as

$$I_{\boldsymbol{N}}^d := \left\{ -\frac{N_1}{2}, \dots, \frac{N_1}{2} - 1 \right\} \times \cdots \times \left\{ -\frac{N_d}{2}, \dots, \frac{N_d}{2} - 1 \right\}$$

and d is the dimensionality of the tranform.

Inverse NFFT

In general the adoint NFFT is not (exactly) the inverse NFFT, i.e.

$$\boldsymbol{A}^{\mathrm{H}}\boldsymbol{A} \neq \boldsymbol{I}$$

However, one can derive an approximation to the (pseudo)inverse quite efficiently.

To this end we first consider the NDFT

$$f(x_j) = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} \hat{f}_k \mathrm{e}^{-2\pi \mathrm{i} k x_j}, \quad j = 0, \ldots, M-1$$

We now extend the sum to $\pm\infty$, which leads to the Fourier series

$$f(x_j) = \sum_{k=-\infty}^{\infty} \hat{f}_k \mathrm{e}^{-2\pi \mathrm{i} k x_j}, \quad j = 0, \ldots, M-1$$

where the coefficients $\hat{f}_k$ have been zero-padded.

Inverse NFFT

The Fourier coefficients $\hat{f}_k$ can be calculated by

$$\hat{f}_k = \int_{-\frac{1}{2}}^{\frac{1}{2}} f(x) \mathrm{e}^{2\pi \mathrm{i} k x_j} \, \mathrm{d}x \qquad k = -\frac{k}{2}, \ldots, \frac{k}{2} - 1$$

Since f is only known at the sampling nodes x_j, we can only consider these when approximating the integral by a sum. When applying a rectangular quadrature rule, one obtains

$$\hat{f}_k \approx \sum_{j=0}^{M-1} w_j f(x_j) \mathrm{e}^{2\pi \mathrm{i} k x_j} \qquad k = -\frac{k}{2}, \ldots, \frac{k}{2} - 1$$

where w_j, $j = 0, \ldots, M-1$ are the quadrature weights. This is the adjoint NFFT with a pre-weighting. In matrix-vector notation this implies $\boldsymbol{A}^{\mathrm{H}} \boldsymbol{W} \approx \boldsymbol{A}^{+}$, i.e. $\boldsymbol{A}^{\mathrm{H}} \boldsymbol{W} \boldsymbol{A} \approx \boldsymbol{I}$.

Summary

- The NDFT is a generalization of the DFT
- The NFFT is an efficient implementation of the NDFT, which exploits a numerical approximation of the complex exponential
- The approximation error is known and can be adjusted to reach machine precision.
- In practice the convolution usually takes most of the computation time. With optimized parameters ($\alpha = 1.25$, $m = 2$) it is possible to make the convolution as fast as the FFT.
- There are various implementations of the NFFT. One reference implementation is the C library NFFT 3 (`https://github.com/NFFT/nfft`). Also a Julia package exists: `https://github.com/tknopp/NFFT.jl`

Medical Imaging

Prof. Dr. Tobias Knopp

6. Januar 2023

Institut für Biomedizinische Bildgebung

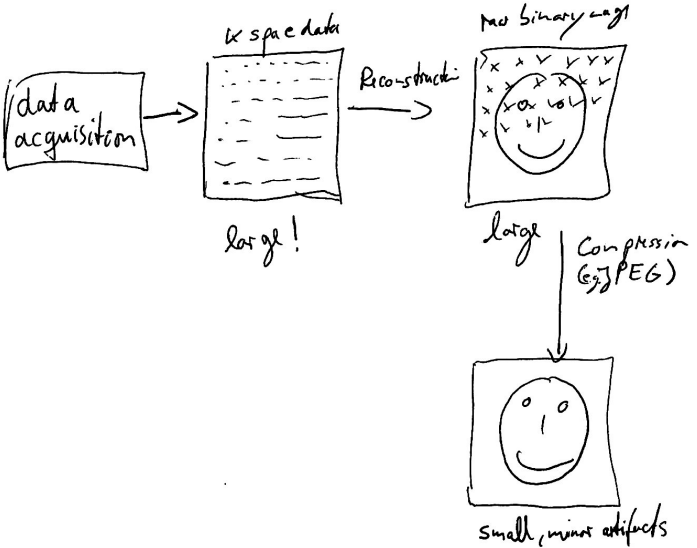
Compressed Sensing

Compressed Sensing

Motivation
MRI scans are slow and usually require minutes of acquisition time.

$\Rightarrow$ Acceleration techniques wanted!

Typical Data Flow

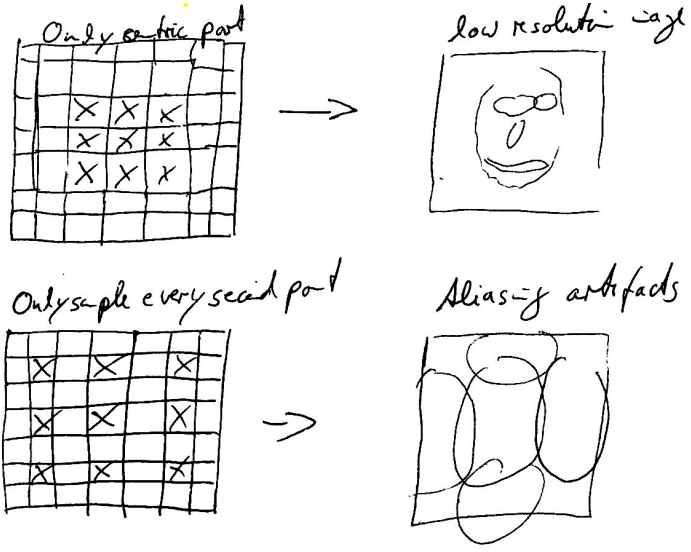


Typical Data Flow

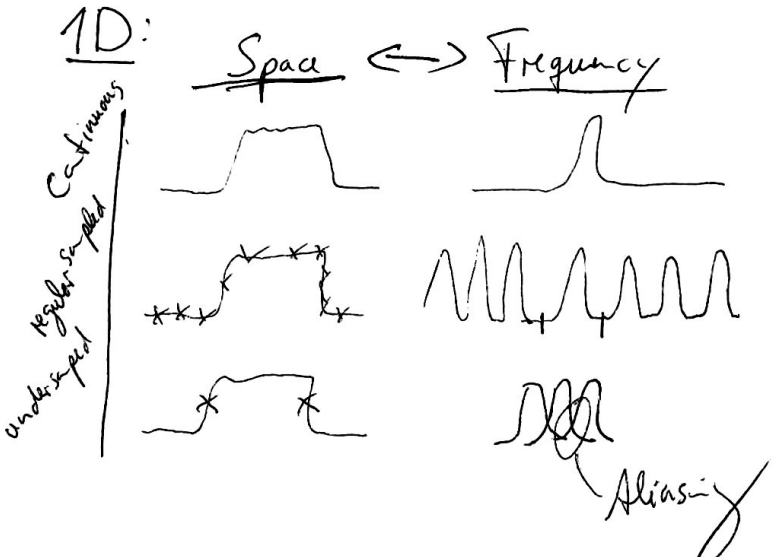
Observation
A lot of data is acquired / processed but in the end only a fraction of data is stored.

Wanted
Measure only few data and "somehow" combine the reconstruction and the compression step.

Subsampling in k-space



Subsampling in k-space



Nyquist Criterion

Sampling frequency has to be twice the signal bandwidth

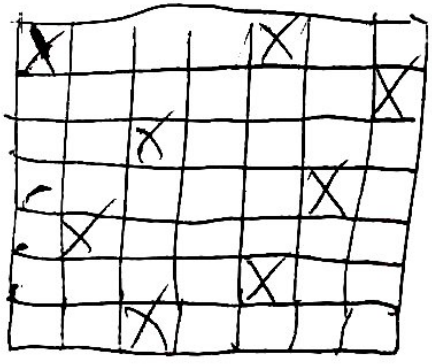
MRI: sampling in frequency space

Can Nyquist Criterion be Beaten?

- Not if the sampling is done in an equidistant way. This is always assumed when deriving the Nyquist criterion.
- If the sampling is done at random points one can beat the Nyquist criterion.
- Equidistant sampling is also named *coherent* sampling, while non-equidistant sampling is named *incoherent* sampling.

Ingredient 1 for Compressed Sensing

Incoherent sampling



Ingredient 2 for Compressed Sensing

Why is it possible to reduce the file size with JPEG?

- Most images are *compressible*.
- The second ingredient therefore is that the underlying image is compressible.

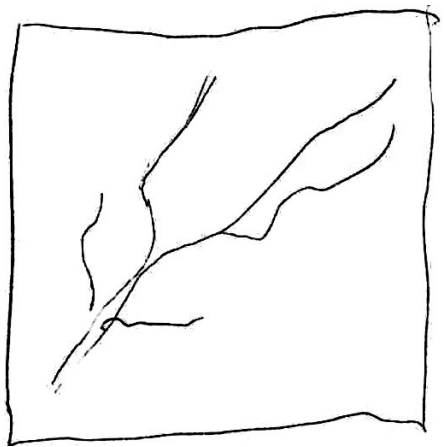
Compressibility

- Express much information with only "few coefficients"
- few coefficients: $\rightarrow$ *sparsity*

Sparsity in Image Space

Only few pixels are non-zero.

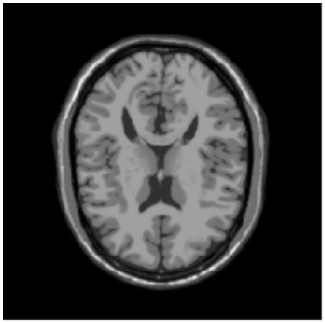
Example: angiogram



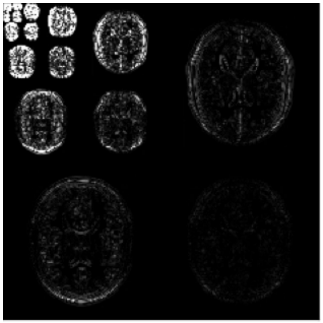
Sparsity in under Transformation

If the data is not sparse in image space one can usually apply a *sparsifying* transformation such as a Wavelet transform or a Block DCT.

Image Space



Wavelet Space



Remark
Wavelet transformation and block DCT are also used in regular compression algorithms.

Compressed Sensing Reconstruction

Imaging Equation

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$$

Subsampled Imaging Equation

$$\boldsymbol{A}_{\text{red}}\boldsymbol{x} = \boldsymbol{b}_{\text{red}}$$

$\Rightarrow$ underdetermined linear system

Compressed sensing reconstruction

We now seek for a sparse solution.

$\Rightarrow \boldsymbol{x}$ should have few non-zero entries

Ansatz

$$\boldsymbol{x}_{\mathrm{CS}} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \underbrace{\|\boldsymbol{A}_{\mathrm{red}}\boldsymbol{x} - \boldsymbol{b}_{\mathrm{red}}\|_2^2}_{\text{data term}} + \underbrace{\lambda\|\boldsymbol{x}\|_0}_{\text{sparsity term}} \tag{1}$$

$\|\boldsymbol{x}\|_0 :=$ number of non-zero elements in $\boldsymbol{x}$

Compressed Sensing Reconstruction

However, using the L_0 norm leads to a very computationally intensive problem (NP-hard) that is unfeasible to compute in practice. Therefore one usually uses alternatively

$$\boldsymbol{x}_{\mathrm{CS}} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \underbrace{\|\boldsymbol{A}_{\mathrm{red}}\boldsymbol{x} - \boldsymbol{b}_{\mathrm{red}}\|_2^2}_{\text{data term}} + \underbrace{\lambda\|\boldsymbol{x}\|_1}_{\text{sparsity term}} \tag{2}$$

with

$$\|\boldsymbol{x}\|_1 := \sum_{n=1}^{N} |x_i|$$

$\Rightarrow$ Convex problem that can be efficiently solved.

Sparsity Transformation

In case that $\boldsymbol{x}$ is not sparse it is required to first apply a sparsity transformation before the L_1 norm is evaluated:

$$\boldsymbol{x}_{\mathrm{CS}} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \underbrace{\|\boldsymbol{A}_{\mathrm{red}}\boldsymbol{x} - \boldsymbol{b}_{\mathrm{red}}\|_2^2}_{\text{data term}} + \underbrace{\lambda\|\boldsymbol{W}\boldsymbol{x}\|_1}_{\text{sparsity term}} . \tag{3}$$

Here, $\boldsymbol{W} \in \mathbb{C}^{N\times N}$ is the sparsity transformation matrix, e.g. a Wavelet transform or a block DCT. It is also possible to use a total-variation term for sparsification.

Medical Imaging - Deep Learning Based Image Reconstruction

Institute for Biomedical Imaging, Hamburg University of Technology

- Prof. Dr.-Ing. Tobias Knopp

Table of Contents

1. Introduction

1.1 Classical Regularization

From the lecture on inverse problem we know that *prior knowledge* is important when solving inverse problems. Prior knowledge can help us to

- reduce the noise amplification caused by the inversion of the linear system
- enforce uniqueness in case of undersampling ($\rightarrow$ compressed sensing)

But how did this look like:

Tikhonov Regularization

$$\underset{\boldsymbol{x}}{\operatorname{argmin}} \underbrace{\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2}_{\text{data consistency}} + \underbrace{\lambda\|\boldsymbol{x}\|_2^2}_{\text{prior knowledge}}$$

ℓ_1 Regularization

$$\underset{\boldsymbol{x}}{\operatorname{argmin}} \|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2 + \lambda\|\boldsymbol{x}\|_1$$

Total-Variation Regularization

$$\underset{\boldsymbol{x}}{\operatorname{argmin}} \|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2 + \eta TV(\boldsymbol{x})$$

where $TV(\boldsymbol{x})$ is the total variation, i.e. a norm that measures edges.

Fused-Lasso Regularization

$$\underset{\boldsymbol{x}}{\operatorname{argmin}} \|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2 + \lambda\|\boldsymbol{x}\|_1 + \eta TV(\boldsymbol{x})$$

ℓ_1 Regularization with Sparsifying Transform

$$\underset{\boldsymbol{x}}{\operatorname{argmin}} \|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2 + \lambda\|\boldsymbol{W}\boldsymbol{x}\|_1$$

What is the characteristic of our prior knowledge?

- In the first case we basically just want to avoid that our solution is *exploding*.
- In the second case we seek for a solution with only few non-zero entries.
- In the third case we seek for a solution with only few edges.
- In the forth case we bring additional prior knowledge in, i.e. that the wavelet coefficients for natural images are sparse.

What are we doing in all cases?

We try to characterize the solution space by penalizing certain characteristics of the solution $\boldsymbol{x}$.

> **Note**
>
> These types of regularization are very generic because we only describe the solution using certain characteristics of the function (so-called features). For instance we know that noise has high frequencies and thus many sharp edges. Therefore applying a TV penality reduces the amount of noise. But what if $\boldsymbol{x}$ itself contains sharp edges? Then our prior knowledge can be wrong.
>
> These classical approaches are said to be *feature driven* while the machine learning approaches we discuss later are *data driven*.

1.2 Time Complexity

Let us next take a look at the time complexity of classical regularization methods.

Tikhonov Regularization

The time complexity for applying Tikhonov regularization is between $\mathcal{O}(N^2)$ and $\mathcal{O}(N^3)$. The later is an upper bound when using e.g. the SVD. The former is a lower bound, e.g. when applying one CGNR iteration or one *outer* Kaczmarz iteration. In practice, iterative solvers are somewhere in-between.

Advanced Regularization

Advanced regularization techniques are non-linear. They usually require an iterative algorithm, which commonly converges much slower than linear ones. Their complexity is $\mathcal{O}(PN^2)$ where P is the number of iterations. While the algorithmic complexity is the same as for Tikhonov regularization, in practice P is usually a factor of $10 - 1000$ larger. This in practice is a real problem and the reason why these methods are popular in research but often not integrated into imaging devices.

1.3 Parameter Choice

Finally, one key issue with classical regularization is the dependency on the regularization parameter(s). While there are methods for optimizing the regularization parameter (e.g. the L-curve), in practice regularization is most often done by hand. But this, again, makes it unsuitable for clinical application, where the image generation needs to be done automatically.

1.4 Summary of Challenges

We identified three major challenges for classical image reconstruction:

1. The form of prior knowledge is just a heuristic and does not describe our solution space well.
2. While Tikhonov regularization is fast, more advanced techniques like TV, Fused-Lasso or ℓ_1-Wavelet regularization are very expensive.
3. Parameter choice is a major problem.

All of these points can be tackled by machine learning as we will see later. Let's first introduce machine learning.

2. Machine Learning

We here only roughly introduce machine learning. Let us consider a function $f : \mathbb{R}^N \to \mathbb{R}^M$, i.e.

$$f(x) = y,$$

and let us assume that we don't know f but we only know L data pairs $(x_l, y_l), l = 1, \dots, L$ with $f(x_l) = y_l$. We call this the training dataset. Then, the task that is solved by *machine learning* is to approximate f by a function $f_\theta : \mathbb{R}^N \to \mathbb{R}^M$, i.e.

$$f_\theta(x) = y,$$

where $\theta \in \mathbb{R}^Q$ are parameters that need to be optimized. To find suitable parameters one commonly solves an optimization problem that looks like

$$\sum_{l=1}^{L} D(f_\theta(x_l), y_l) \overset{\theta}{\to} \min,$$

where D is a distance measure. For instance one can use the mean squared error (MSE)

$$D_{\text{MSE}}(\tilde{y}, y) = \|\tilde{y} - y\|_2^2.$$

The aim now is that $f(x) \approx f_\theta(x)$ not only for the training data but also for unseen data.

> **Note**
>
> Machine learning is thus basically function approximation. It is related to interpolation but it is not required that $f(x) = f_\theta(x)$ for $x_l, l = 1, \dots, L$.
>
> Thus, ML is very similar to classical function approximation where a certain basis (polynomials, splines, ...) are used as a model f_θ.

2.1 Artificial Neural Networks

There are many machine learning methods. Those which got very popular in the last decade are artificial neural networks (ANN). These consists of several layers to which certain operations are applied. A simple ANN with two layers can be formulated as

$$f_{\text{ANN}}(\boldsymbol{x}) = \sigma(\boldsymbol{H}_2(\sigma(\boldsymbol{H}_1\boldsymbol{x} + \boldsymbol{c}_1)) + \boldsymbol{c}_2)$$

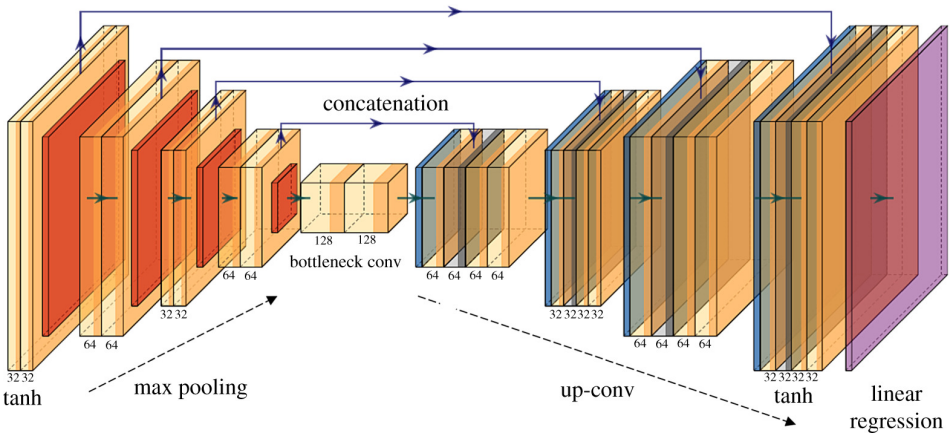
with

- $\boldsymbol{H}_1, \boldsymbol{H}_2$ being matrices transforming the input vector (linear transformations). These can be *dense matrices* (dense layers) or *convolutions* (conv layers).
- $\boldsymbol{c}_1, \boldsymbol{c}_2$ being bias vectors.
- σ being a non-linear activation function.

> **Note**
>
> An artificial neural network is said to be deep if it contains various nested layers. A network is said to be wide if it has many connections/parameters within a layer (i.e. a dense layer).

Example

The following showcases a so-called decoder/encoder ANN, which consists of several convolution layers and down/upsampling operations.



We next outline different ways to use ANNs for image reconstruction. In all cases we consider

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$$

to be our inverse problem and we consider that we have training samples $(\boldsymbol{x}_l, \boldsymbol{b}_l)$, $l = 1, \dots, L$.

3. Machine Learning for Image Reconstruction

In the following we sketch some methods to use machine learning for image reconstruction. The overview is not meant to be complete since the field of ML-based image reconstruction is still evolving.

3.1 Learn the Inverse Directly

The first idea is to directly learn the the inverse. This means we formulate

$$f_\theta^{\text{inverse}}(\boldsymbol{b}) = \boldsymbol{x}$$

and optimize the parameters $\boldsymbol{\theta}$.

Pros

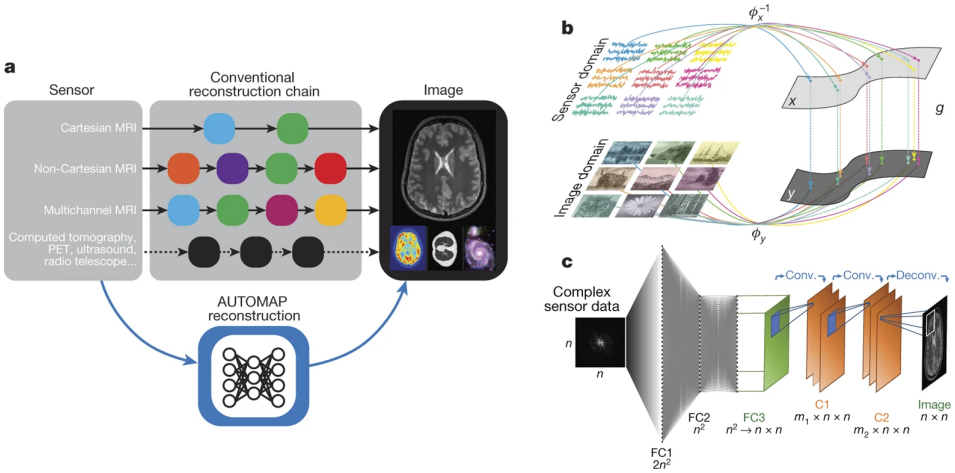
- We can take all imperfections into account since we don't assume any (simplified) physical model.

Cons

- We don't use information of the imaging operator, i.e. we learn things that we already know. In the case of MRI the ANN $f_\theta^{\text{inverse}}$ needs to learn the Fourier transform.
- The ANN needs to be *expressive* enough to learn the (inverse) imaging operator. This requires dense layers, which makes training much more complicated (both from the accuracy and the training time/memory point of view).
- In order to train such an ANN it needs a very large training dataset.

Example AUTOMAP

See: Zhu, B., Liu, J., Cauley, S. et al. Image reconstruction by domain-transform manifold learning. Nature 555, 487–492 (2018)



3.2 Postprocessing Networks

Learning the inverse directly is not really the best option. We next consider models that use a traditional image reconstruction in its core and afterwards apply the ANN in a post-processing step.

So let's assume $f^{\text{classic reco}}(\boldsymbol{b})$ is a classical image reconstruction method. For instance the FFT in the case of MRI and the FBP in the case of CT. Then a postprocessing network would be formulated as

$$f_\theta^{\text{postprocessing}}(f^{\text{classic reco}}(\boldsymbol{b})) = \boldsymbol{x}$$

Looks very similar but the important difference is that $f_\theta^{\text{postprocessing}}$ is an ANN acting in image space, which is a well known task in the field of image processing. Hence, well known network architectures can be used here.

Pros

- Less complex model needed. CNNs are sufficient.
- Less training data needed.
- Physics is taken into account.

Cons

- Not as generic/powerful than learning the inverse.

3.3 Use Neural Networks with Iterative Solver

Advanced regularization techniques usually use iterative solvers like FISTA, ADMM, CG(NR), or the Kaczmarz method. Let us for the moment take a much more basic approach: the Landweber iteration. Similar to the other methods it aims at solving the least squares problem

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) \text{ where } f(\boldsymbol{x}) := \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2.$$

Since we want to minimize $\boldsymbol{f}$ we can iterate over $\boldsymbol{x}$ and follow the gradient in order to minimize $\boldsymbol{f}$, i.e. we use gradient descent:

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \omega \nabla f(\boldsymbol{x}_k)$$

Here, ω is a relaxation factor which needs to be appropriately chosen (not discussed at this point).

> **Note**
>
> Side note: The conjugate gradient method follows the same approach but does not use the steepest decent but follows conjugate directions, i.e. the history of descents is taken into account, which accelerates convergence.

For our specific $\boldsymbol{f}$ we can calculate the gradient explicitly

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \omega \boldsymbol{A}^H(\boldsymbol{A}\boldsymbol{x}_k - \boldsymbol{b}).$$

We can more generally say that this form of iteration is written as

$$\boldsymbol{x}_{k+1} = \mathrm{DC}(\boldsymbol{x}_k)$$

where DC is the function that ensures *data consistency*, i.e. the minimization of the residual. With this more general formulation, DC could for instance also be one *outer* Kaczmarz iteration, i.e. one sweep over all rows.

With this knowledge we can integrate machine learning into iterative algorithms. Two different variants that are discussed next.

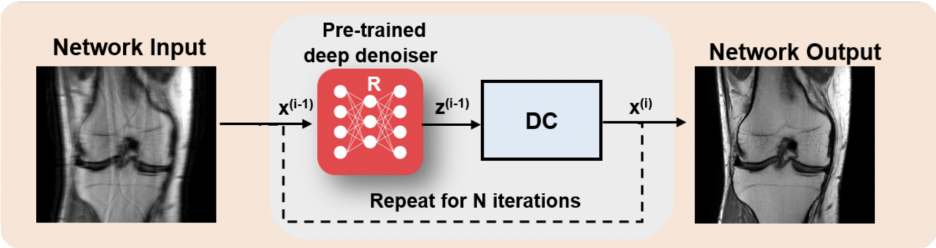
3.3.1 Plug-and-Play Approach

The easiest and most robust way is to train a neural network for a specific task like denoising, super-resolution or in general *image enhancement*. Then we apply the enhancement neural network after each DC step:

$$\boldsymbol{x}_{k+1} = f_\theta^{\text{enhance}}(\text{DC}(\boldsymbol{x}_k)).$$

We call this *plug and play* since the neural network is not trained for the actual image reconstruction but outside the reconstruction pipeline.

The following picture is taken from this publication and graphically showcases the plug-and-play approach:



3.3.2 Unrolled Iteration

The plug-and-play approach can be refined by training the network within the reconstruction problem. In order to do so, one needs to use a fixed number of iterations and *unroll* the network.

For instance for $P = 3$ iterations we can do this by:

$$\boldsymbol{x}_{\text{final}} = f_{\theta_3}^{\text{enhance}}(\text{DC}(f_{\theta_2}^{\text{enhance}}(\text{DC}(f_{\theta_1}^{\text{enhance}}(\text{DC}(\boldsymbol{x}_0))))).$$

Now we can train $\boldsymbol{\theta_1}$, $\boldsymbol{\theta_2}$, and $\boldsymbol{\theta_3}$ in an end-to-end fashion. Here, there are two variants:

- One can use different NN parameters in each iteration.
- One can share the weights.

The former is more powerful but more difficult to train because it has more parameters.

Note

What might not be obvious is that the unrolled iteration can also significantly speed up the reconstruction process. The reason is that the neural networks are not only able to do image enhancement in the classical sense (i.e. denoising) but they can also generate short paths. This is because the number of iterations is fixed and known during training such that the algorithm is forced to converge in the predefined number of iterations. In practice, often just about $P = 10$ iterations are needed.

Note

All ML-based reconstruction methods we outlined were parameter-free. They need to be trained on different noise classes and then usually can automatically adapted to a certain noise class. This adresses a further challenge of the feature-based reconstruction methods.

4. Summary

In this short lecture we have discussed the challenges of classical (feature-based) image reconstruction and regularization techniques. We then outlined some ways how machine learning can be used to provide a tailored domain specific form of prior knowledge. You will need some first experience in machine learning and the respective frameworks (Python: PyTorch, TensorFlow; Julia: Flux.jl, Lux.jl) before you can start implementing the methods discussed in this lecture.