MEDICAL IMAGING

Lecture Skriptum

Benjamin Bergmann

Last updated: 4.2.2026

Contents

Inverse Problems

What is an Inverse Problem?

There exit a "Forward Problem" which estimate the effect from the cause and then there is inverse Problem which estimates the cause from the effect. In the medical context that would be finding the cause illness given from a certain syntom/effect. Typically, the forward problem is "easy" and well described. The challenge here is: We need to solve the inverse problem given only the observed effect of the forward problem.

As an Example from the real world: forward problem: The street becomes wet when it rains. backward problem would be: We observe that the street is wet. Why?

There are multiple different causes:
- Rain
- Fog
- Cleaning

And this can be already problematic as we have multiple different options for what the cause could be.

Example 1 — Computer Tomography .

Forward Problem X-ray emitter and detector rotating around the body. Detectors measure the number of photons passing through the body and hitting the detector

Inverse Problem Reconstruct the interior of the body from the measured detector signals.

Note that a CT Scan can be very large in file size. A scan from shoulder to belt line is already 18GB of data for just a single scan. So we bascially have y and we want to get to x

Example 2 — Deconvolution . **Forward Problem** Observe a blurred image

$$f = k * u$$

on a domain $\Omega \subset \mathbb{R}^2$.

Inverse Problem Estimate the sharp image $u : \Omega \to \mathbb{R}$ given the blur kernel $k : \Omega \times \Omega \to \mathbb{R}_+$ On of the oldest clasical methods to do that is the Wiener Filter. Deconvolution is linked to Fourier F:

$$f = k * u$$

$$F(f) = F(k) \odot F(u)$$

If we wanna do the inverse:

$$F^{-1}\{F(f)\} = F^{-1}\{F(k) \odot F(u)\} = f$$

where $\odot$ is a pointwise multiplication. So a estimate $\hat{u}$ would be

$$\hat{u} = F^{-1}\left(\frac{F(f)}{F(k)}\right)$$

The only problem here is when we have 0 frequencies in the kernel. The Wiener Filtering introduces

$$\hat{u} = F^{-1}\left(\frac{F(f)}{I\sigma^2 F(k)}\right)$$

What is an Inverse Problem? (formal)

Definition 1 (Inverse Problem) .

Given a matrix $A \in \mathbb{R}^{m\times n}$ and a vector $x \in \mathbb{R}^n$ the forward problem is $y = Ax \in \mathbb{R}^m$. The inverse problem is: Given A and y, estimate x.

Vector Space

Definition 2 (Vector Space) . A non-empty set V is a vector space over a field $\mathbb{F} \in \{\mathbb{R}, \mathbb{C}\}$ if there are operations of vector addition: $+ : V \times V \to V$ and scalar multiplication: $\cdot : \mathbb{F} \times V \to V$ satisfying the following axioms:

Vector addition

1. $u + v \in V \quad \forall u, v \in V$
2. $u + v = v + u$
3. $(u + v) + w = u + (v + w) \quad \forall u, v, w \in V$
4. $\exists 0 \in V : u + 0 = u \quad \forall u \in V$
5. $\forall u \in V : \exists -w : u + (-w) = 0$

Scalar multiplication

1. $av \in V \quad \forall a \in \mathbb{F}, \forall v \in V$
2. $(ab)v = a(bv) \quad \forall a, b \in \mathbb{F}, v \in V$
3. $a(u + v) = au + av \quad \forall a \in \mathbb{F}, \forall u, v \in V$
4. $(a + b)v = av + bv \quad \forall a, b \in \mathbb{F}, \forall v \in V$
5. $\exists 1 \in \mathbb{F} : 1 * u = u \quad \forall u \in V$

Example 3 — Vector Space .

- $\mathbb{R}^n = \left\{(x_1, ..., x_n)^T : x_1, ..., x_n \in \mathbb{R}\right\}$
- $\mathcal{C}(\mathbb{R}^n, \mathbb{R})$ set of function $f : \mathbb{R}^n \to \mathbb{R}$ that are continuous
- $\mathcal{C}^1(\mathbb{R}^n, \mathbb{R})$ set of function $f : \mathbb{R}^n \to \mathbb{R}$ that are continuous and once continuously differentiable
- $L^2(\mathbb{R}^n, \mathbb{R}) = \left\{f : \mathbb{R}^n \to \mathbb{R} : \int_{\mathbb{R}^n} |f(x)|^2 \, dx < \infty\right\}$ Lebesgue space
- $H^1(\mathbb{R}^n, \mathbb{R}) = \left\{f \in L^2(\mathbb{R}^n, \mathbb{R}) : \int_{\mathbb{R}^n} |f'(x)|^2 \, dx < \infty\right\}$ Sobolev space $(p = 2)$, Hilbert space

Inverse Problem

Definition 3 (Inverse Problem) . Let X, Y be vector spaces and $A : X \to Y$. The forward problem is defined as $y = Ax$ for any $x \in X$. The inverse problem is to find $x \in X$ such that $Ax = y$ for any $y \in Y$.

So we want to get $A^{-1}(y) = \hat{x}$

Well-Posedness (Hadamard)

We can now start to categorize inverse problems:

Definition 4 (Well-Posedness) . The inverse problem $Ax = y$ is well-posed if:

1. **Existence:** a solution exists
2. **Uniqueness:** the solution is unique
3. **Stability:** the solution depends continuously on the data

If one condition fails, the problem is ill-posed.

Example 4 — Well-Posedness . Is this example well posed?

Let $X, Y \in \mathbb{R}$ and $A : \mathbb{R} \to \mathbb{R}, x \to x^2$

Answer:

- Existence: for $y = -1$ no solution exists (if we would map to $\mathbb{R}^+$ it would be okay)
- Uniqueness: for $y = 1$, $x = \pm 1$ which is not unique
- Stability: yes, since A is continuous

Example 5 — Well-Posedness . Let $X, Y \in \mathbb{R}^2$ and $A = \begin{pmatrix} 2 & 3 \\ 1 & 2 \end{pmatrix} \in \mathbb{R}^{2\times 2}$.

Is the inverse problem $Ax = y$ for $y \in Y$ well-posed?

- Existence: $\exists A^{-1}$? Since $\det(A) = 4 - 3 = 1 \neq 0$, the matrix is invertible.
- Uniqueness: Yes, because $\det(A) \neq 0$.
- Stability: Yes, as A^{-1} is continuous.

Inner Product

Definition 5 (Inner Product) . An inner product on a vector space Y over a $\mathbb{F}$ is a map

$$\langle ., . \rangle : Y \times Y \to \mathbb{F}$$

with the following properties:

1. **Symmetry:** $\langle x, y \rangle = \overline{\langle y, x \rangle} \quad x, y \in Y$
2. **Additivity:** $\langle x, y + z \rangle = \langle x, y \rangle + \langle x, z \rangle \quad x, y, z \in Y$

3. **Homogeneity:** $\langle \lambda x, y \rangle = \lambda \langle x, y \rangle \quad x, y \in Y \quad \lambda \in \mathbb{R}$
4. **Positivity:** $\langle x, x \rangle \geq 0$ and $\langle x, x \rangle = 0 \iff x = 0$

Vector Norm

Definition 6 (Inner Product) . A vector norm is a vector space Y over a field F is a map $\|.\| : Y \to \mathbb{R}$ with:

1. **non-negativity:** $\|x\| \geq 0 \quad \forall x \in V, \|x\| = 0 \Leftrightarrow x = 0$
2. **positive homogeneity:** $\|\lambda x\| = |\lambda| \ \|x\| \ \forall x \in Y, \lambda \in \mathbb{F}$
3. **triangle inequality:** $\|x + y\| \leq \|x\| + \|y\| \ x, y \in V$

Example 6 — vector norm .

$$\|x\|_p = \sqrt[p]{\sum_{i=1}^{n} |x_i|^p} \quad x \in X \subset \mathbb{R}^n$$

Matrix Norm

Definition 7 (Inner Product) . Let $\|\cdot\|_a$ and $\|\cdot\|_b$ be vector norms on $\mathbb{R}^n$ and $\mathbb{R}^m$, respectively. Given a matrix $A \in \mathbb{R}^{m \times n}$, the **induced matrix norm** $\|A\|_{a,b}$ is defined as:

$$\|A\|_{a,b} = \max_{x \in \mathbb{R}^n : \|x\|_a \leq 1} \|Ax\|_b = \sup_{\{x \in \mathbb{R}^n \setminus \{0\}\}} \frac{\|Ax\|_b}{\|x\|_a}$$

$$\|Ax\|_b \leq \|A\|_{ab} \|x\|_a$$

Example 7 — Matrix norm .

- If $a, b = 2$: $\|A\|_{2,2} = \|A\|_2 = \sigma_{\max}(A) = \sqrt{\lambda_{\max}(A^T A)}$
- If $a, b = 1$: $\|A\|_{1,1} = \|A\|_1 = \max_j \sum_i |A_{ij}|$
- If $a, b = \infty$: $\|A\|_\infty = \max_i \sum_j |A_{ij}|$

Injection, Surjection, Bijection

Definition 8 (Injection, Surjection, Bijection) . These properties of mappings $A : X \to Y$ are defined as

- **Injection:** $A : X \to Y$ is injective if $Ax_1 = Ax_2 \Rightarrow x_1 = x_2$.
- **Surjection:** $A : X \to Y$ is surjective if $\forall y \in Y, \exists x \in X : Ax = y$.

- **Bijection:** $A : X \to Y$ is bijective if it is both injective and surjective. $\forall y \in Y, \exists! x \in X : Ax = y \Leftrightarrow \exists A^{-1} : x = A^{-1}y$.

Null Space and Range Space

Definition 9 (Null Spance and Range Space) . Let $A : X \to Y$ where X, Y are vector spaces.
- **Nullspace of A:** $N(A) = \{x \in X : Ax = 0\}$
- **Range space of A:** $R(A) = \{Ax \in Y : x \in X\}$

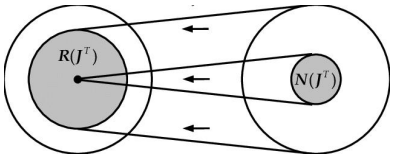


Figure 1 : Null space and range space taken from here

Connection to Hadamard's Definition

- **Existence** $\Leftrightarrow$ Surjection $\Leftrightarrow$ $R(A) = Y$
- **Uniqueness** $\Leftrightarrow$ Injection $\Leftrightarrow$ $N(A) = \{0\}$
- **Existence & Uniqueness** $\Leftrightarrow$ Bijection

Definition of the Linear Inverse Problem

Definition 10 (Linear Inverse Problem) . Given $A : X \to Y$ and observation $y \in Y$ the inverse problem is called linear if A is linear which means that $A(\alpha x_1 + \beta x_2) = \alpha A(x_1) + \beta A(x_2)$

Example 8 — Linear Inverse Problem . A... is the Radon transform

$$(Ax)_i = y_i = \int_{\Gamma_i} x(s)ds$$

$$A(\hat{x}) = A(\lambda_1 \cdot x_1 + \lambda_2 x_2) = \hat{y}_i = \int_{\Gamma_i} \hat{x}(s)ds = \int_{\Gamma_i} \lambda_1 x_1(s) + \lambda_2 \cdot x_2(s)ds$$

$$= \lambda_1 \underbrace{\int_{\Gamma_i} x_1(s)ds}_{y_i^1} + \lambda_2 \underbrace{\int_{\Gamma_i} x_2(s)ds}_{y_i^2} = \lambda_1 y_i^1 + \lambda_2 y_i^2 = \lambda_1 A(x_1)_i + \lambda_2 A(x_2)_i$$

Nullspace of linear A $\Rightarrow \{0\} \in \mathcal{N}(A)$

Decomposition of Square Matrices

Definition 11 (Decomposition of the Square Matrix) . Let $A \in \mathbb{R}^{n\times n}$, recall Eigenvalues λ_i and Eigenvectors v_i:

$$Av_i = \lambda_i v_i \quad \text{for } i = 1, ..., n$$

$$\det(A - \lambda I) = 0$$

If v_i are linearly independent: $Av_i = \lambda_i v_i \Rightarrow AQ = Q\Lambda \Rightarrow A = Q\Lambda Q^{-1}$ Where $Q = (v_1, ..., v_n)$.

Remark. If A is hermitian $\Leftrightarrow A^* = A$, we have that all λ_i are real & v_i are orthonormal.

$$v_i^T v_j = 0 \quad \text{for } i \neq j$$

$$A = Q\Lambda Q^T$$

Singular Value Decomposition

Definition 12 (Singular Value Decomposition) . Let $X \in \mathbb{R}^n, Y \in \mathbb{R}^m$ be an inverse problem $Ax = y$ with a $A \in \mathbb{R}^{m\times n}$. The Goal:

$$A = U\Lambda V^T$$

- $U \in \mathbb{R}^{m\times p}$, $\Lambda \in \mathbb{R}^{p\times p}$, $V \in \mathbb{R}^{p\times n}$
- p is the number of non-zero singular values $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_p > 0$.

Link between SVD and Eigendecompostion

$$A \in \mathbb{R}^{m\times n}$$

$$\begin{cases} (1) & Ax = y \\ (2) & A^T\hat{x} = \hat{y} \end{cases} \Leftrightarrow \underbrace{\begin{pmatrix} 0 & A \\ A^T & 0 \end{pmatrix}}_{B\in\mathbb{R}^{(m+n)\times(m+n)}} \cdot \begin{pmatrix} \hat{x} \\ x \end{pmatrix} = \begin{pmatrix} y \\ \hat{y} \end{pmatrix}$$

$B = B^T$: $\quad Bw_i = \lambda_i w_i$

$$\begin{pmatrix} 0 & A \\ A^T & 0 \end{pmatrix}\begin{pmatrix} u_i \\ v_i \end{pmatrix} = \lambda_i \begin{pmatrix} u_i \\ v_i \end{pmatrix} \Leftrightarrow \begin{cases} \text{1st:} & Av_i = \lambda_i u_i \\ \text{2nd:} & A^T u_i = \lambda_i v_i \end{cases}$$

1st:

$$\lambda_i Av_i = \lambda_i^2 u_i$$

$$A(\lambda_i v_i) = \lambda_i^2 u_i$$

$$AA^T u_i = \lambda_i^2 u_i$$

$$U = (u_1 \mid ... \mid u_m)$$

2nd:

$$A^T(\lambda_i u_i) = \lambda_i^2 v_i$$

$$A^T A v_i = \lambda_i^2 v_i$$

$$V = (v_1 \mid ... \mid v_n)$$

Solving Inverse Problems

We have 3 possible cases:
- Eigenvalue Problem ($p = n = m$)
- Least Squares ($p = m > n$)
- Minimum Length ($p = n > m$)

Eigenvalues ($p = n = m$)

Let $A \in \mathbb{R}^{n \times n}$. Eigendecomposition:

$$Av_i = \lambda_i v_i \quad i = 1, ..., n$$

Assuming A is invertible (assume $\exists A^{-1}$):

$$A^{-1}Av_i = \lambda_i A^{-1} v_i$$

Since $A^{-1}A = I_d$:

$$v_i = \lambda_i A^{-1} v_i$$

Rearranging for A^{-1} (where $\lambda_i \neq 0$):

$$\frac{1}{\lambda_i} v_i = A^{-1} v_i$$

Conclusions:
- Eigenvectors of A and A^{-1} are the same.
- Eigenvalues of $\lambda(A^{-1}) = \frac{1}{\lambda(A)}$.
- This implies $\lambda_i \neq 0$, which is equivalent to $\det(A) \neq 0$.

Least Squares ($m > n$)

We have $Ax = y$ and $A \in \mathbb{R}^{m \times n}$ and $m > n$ which leads us to a overdetermined system

$$e_i = a_i^T x - y_i$$

Idea: minimize the squared error

$$\hat{x} = \arg\min E(x) \quad \text{where} \quad E(x) := \frac{1}{2} \sum_{i=1}^{m} \left(a_i^T x - y_i\right)^2 = \frac{1}{2} \left\|Ax - y\right\|_2^2 = \frac{1}{2} \left\|e\right\|_2^2$$

Here we define $e = Ax - y$. How do we solve this optimization problem?

$\nabla E(x) = 0$ where $\nabla E(x) \in \mathbb{R}^n$

$$\frac{\partial E}{\partial x} = \frac{\partial e}{\partial x} \frac{\partial E}{\partial e} = \frac{\partial e}{\partial x} \frac{1}{2} 2e = A^T e = A^T (Ax - y) = A^T Ax - A^T y = 0$$

$$(A^T A)x = A^T y$$

$$x = \left(A^T A\right)^{-1} A^T y$$

Example 9 — 2x2 CT Reconstruction .

$$x \in \mathbb{R}^4 \quad y \in \mathbb{R}^5$$

x_1	x_2
x_3	x_4

Table 1 : Grid representation of variables x_i

We send rays through the matrix in 3 directions (top to bottowm, diagonal and left to right):

$y_1 = x_1 + x_3$ (Column 1 sum)

$y_2 = x_2 + x_4$ (Column 2 sum)

$y_3 = x_1 + x_2$ (Row 1 sum)

$y_4 = x_3 + x_4$ (Row 2 sum)

$y_5 = x_1 + x_4$ (Diagonal sum)

We can bring this now in the form $Ax = y$ which leads us to equation that looks like this:

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{pmatrix}$$

Minimum Length $(p = n > m)$

Let $Ax = y$ with $A \in \mathbb{R}^{m \times n}$. Since $n > m$, $(A^T A)^{-1}$ does not exist. This is an **underdetermined system**. If we had multiple solutions we would exactly solve $Ax = y$. We pick one using a priori knowledge:

$$\min_x \frac{1}{2} \|x\|_2^2 \quad \text{s.t.} \quad Ax = y$$

Recap: Lagrange Multipliers We want to solve $\min_x E(x)$ subject to $C(x) = 0$. For that we define Lagrangian:

$$\mathcal{L}(x, \tau) = E(x) + \langle C(x), \tau \rangle$$

where $\tau \in \mathbb{R}^m$ is the Lagrange multiplier. Then we can find a solution by setting $\nabla \mathcal{L}(x, \tau) = 0$

$$\frac{\partial}{\partial x} \mathcal{L} = \frac{\partial E}{\partial x} + \frac{\partial C}{\partial x} \tau = 0$$

$$\frac{\partial}{\partial \tau} \mathcal{L} = C(x) = 0$$

From the initial setting where we want to find the minimum of x we can define now E and C:

$$E(x) = \frac{1}{2} \|x\|_2^2$$

$$C(x) = y - Ax = 0 \quad \Leftrightarrow \quad h(x, \tau) = \frac{1}{2} \|x\|_2^2 + \langle y - Ax, \tau \rangle$$

Then we can write out the Lagrange terms and solve them for x:

$$\frac{\partial}{\partial x}\mathcal{L} = x - A^T\tau = 0$$

$$x = A^T\tau$$

$$\frac{\partial}{\partial \tau}\mathcal{L} = y - Ax = 0$$

$$y = Ax = A(A^T\tau) = (AA^T)\tau$$

$$\tau = (AA^T)^{-1}y$$

$$x = A^T(AA^T)^{-1}y$$

Generalized Inverse

Let $X \in \mathbb{R}^n$, $Y \in \mathbb{R}^m$ and the inverse problem $Ax = y$ with $A \in \mathbb{R}^{m\times n}$.

We define the generalized inverse as:

$$A_g^{-1} = \left(U_p\Lambda_p V_p^T\right)^{-1} = \left(V_p^T\right)^{-1}\Lambda_p^{-1}U_p^{-1} = V_p\Lambda_p^{-1}U_p^T$$

Note that here U is the eigenvectors of AA^T and V are the eigenvectors of A^TA. And note that $U_p^TU_p = I$, that's why we can say $U_p^T = U_p^{-1}$ and $U_p = \left(U_p^T\right)^{-1}$. The same holds for V_p. Check if Generalized Inverse computes the exact Least Squares and Minimum Length solutions:

I. $p = m = n$***:***

As we have shown above, if $m = n$ the calulations can be done via Eigenvectors and Eigenvalues:

$$A_g^{-1} = V_p\Lambda_p^{-1}U_p^T \qquad |\cdot A = U_p\Lambda_p V_p^T$$

$$A_g^{-1}A = V_p\Lambda_p^{-1}\underbrace{U_p^TU_p}_{I}\Lambda_p V_p^T = V_p\underbrace{\Lambda_p^{-1}\Lambda_p}_{I}V_p^T = I$$

II. $p = m > n$***:***

$$\begin{aligned}
x &= (A^TA)^{-1}A^Ty \\
&= \left(\left(U_p\Lambda_pV_p^T\right)^T\left(U_p\Lambda_pV_p^T\right)\right)^{-1}\left(U_p\Lambda_pV_p^T\right)^Ty \\
&= \left(V_p\Lambda_p\underbrace{U_p^TU_p}_{\text{Id}}\Lambda_pV_p^T\right)^{-1}(V_p\Lambda_pU_p^T)y \\
&= \left(V_p\Lambda_p^2V_p^T\right)^{-1}V_p\Lambda_pU_p^Ty \\
&= V_p\Lambda_p^{-2}\underbrace{V_p^TV_p}_{\text{Id}}\Lambda_pU_p^Ty \\
&= V_p\Lambda_p^{-1}U_p^Ty = A_g^{-1}y
\end{aligned}$$

III. $p = m < n$***:***

$$\begin{aligned} x &= A^T(AA^T)^{-1}y \\ &= \left(V_p\Lambda_p U_p^T\right)\left(U_p\Lambda_p V_p^T V_p\Lambda_p U_p^T\right)^{-1}y \\ &= \left(V_p\Lambda_p U_p^T\right)\left(U_p\Lambda_p^2 U_p^T\right)^{-1}y \\ &= V_p\Lambda_p \underbrace{U_p^T U_p}\Lambda_p^{-2}U_p^T y \\ &= V_p\Lambda_p^{-1}U_p^T y = A_g^{-1}y \end{aligned}$$

IV. $0 < p < \min(m, n)$***:***

However, A_g^{-1} still exists. It computes a solution that interpolates between Least Squares & Minimum Length solutions.

Regularization

Consider polynomial regression

$$p(a) = \sum_{i=1}^{n} x_i a^{i-1} = x_1 \cdot 1 + x_2 \cdot a + ... + x_n a^{n-1}$$

Where x represents the coefficients of the polynomial.

$$p(a) \Leftrightarrow Ax = \begin{pmatrix} 1 & a_1 & a_1^2 & ... & a_1^{n-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & a_m & a_m^2 & ... & a_m^{n-1} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix}$$

How to choose n?
- manually
- very large + regularization

We can incorporate Prior Knowledge: Least squares problem + regularization:

$$\hat{x} = \arg\min_x \frac{1}{2}\left\|Ax - y\right\|_2^2 + R(x)$$

Example: $R(x) = \frac{\lambda}{2}\left\|x\right\|_2^2$ (Tikhonov regularization, aka weight decay)

$$\frac{1}{2}\left\|Ax - y\right\|^2 + \frac{\lambda}{2}\left\|x\right\|_2^2 \to \min$$

We derive it an set it to 0:

$$\frac{1}{2} \cdot 2 \cdot A^T(Ax - y) + \frac{\lambda}{2} \cdot 2x = 0$$

$$A^T Ax + \lambda x = A^T y$$

$$x = \left(A^T A + \lambda I\right)^{-1} A^T y$$

Regularization Types

Name	$R(x)$	Intuition
Tikhonov	$\lambda \|Gx\|_2^2$ $(\|Gx - \hat{x}\|_2^2)$	Existence of Inverse
L^2	$\lambda \|x\|_2^2$ $(G = I)$	Minimum length/norm
H^1	$\lambda \|\nabla x\|_2^2$ $(G = \nabla)$	Smooth gradients
L^1	$\lambda \|x\|_1$	Sparse solutions
Total variation (TV)	$\lambda \|\nabla x\|_1$	Sparse gradients (piece-wise constant solutions)

The Proximal Mapping

1. Projection onto a set S

$$\text{proj}_S(x) = \arg\min_{y \in S} \frac{1}{2} \|x - y\|_2^2$$

2. Proximal mapping of a function $g(x)$

$$\text{prox}_g(x) = \arg\min_y \frac{1}{2} \|x - y\|_2^2 + g(y)$$

where we can for example put in the indicator function of the set S

$$g(y) = \begin{cases} 0 & \text{if } y \in S \\ \infty & \text{else} \end{cases}$$

Example 10 — Soft Thresholding . Let's set: $g(x) = |x|$

To find the proximal mapping for the absolute value (L1 norm), we solve:

$$\text{prox}_{|\cdot|}(x) = \arg\min_y \frac{1}{2} |x - y|^2 + |y|$$

The derivative of $|y|$ is:

$$\frac{d}{dy} |y| = \begin{cases} 1 & y > 0 \\ [-1, 1] & y = 0 \\ -1 & y < 0 \end{cases}$$

We simplify and solve

$$\frac{d}{dy}\left(\frac{1}{2} |x - y|^2\right) + \partial g(y) = 0$$

- **Case $y > 0$:** $-(x - y) + 1 = 0 \quad \Rightarrow \quad y = x - 1 > 0 \quad \Rightarrow \quad x > 1$
- **Case $y < 0$:** $-(x - y) - 1 = 0 \quad \Rightarrow \quad y = x + 1 < 0 \quad \Rightarrow \quad x < -1$

- **Case $y = 0$:** $-(x-0) + [-1,1] = 0 \quad \Rightarrow \quad x \in [-1,1]$

Thus, the Soft Thresholding operator is:

$$\text{prox}_{|\cdot|}(x) = \begin{cases} x-1 & \text{if } x > 1 \\ x+1 & \text{if } x < -1 \\ 0 & \text{else} \end{cases}$$

This would now be for example the sparse solution that we have seen in the table above. The proximal map kills all of the small graidents and shirks the rest of the gradients. It is essentially a way of first performing graident descent and then you look with the proximal map where to go and at this point you look then how suitable it is or how much penalty the regularizer gives you. In our case the regularizer shrinks the gradient and sets all the values in ± 1 to 0.

A Probabilistic Perspective of Regularization

Assume observed measurements $y_i \in \mathbb{R}^m$ follow a Gaussian distribution:

$$y \sim \mathcal{N}(Ax, \Sigma) \iff p(y|x) = |2\pi\Sigma|^{-\frac{1}{2}} \exp\left(-\frac{1}{2} \|Ax - y\|^2_{\Sigma^{-1}}\right)$$

Moreover, we know the gradients of the solution follows a Gaussian prior:

$$\nabla x \sim \mathcal{N}(0, \eta I) \iff p(x) = |2\pi\eta I|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}\eta \|x\|^2\right)$$

Using Bayes' Rule to find the posterior distribution:

$$p(x|y) = \frac{p(y|x) \cdot p(x)}{p(y)}$$

$$\log p(x|y) = \log p(y|x) + \log p(x) - \log p(y)$$

$$\log p(x|y) = -\frac{1}{2} \|Ax - y\|^2_{\Sigma^{-1}} - \log Z_1 - \frac{1}{2\eta} \|x\|^2 - \log Z_2 - \log p(y)$$

Since Z_1, Z_2, and $p(y)$ are constants that do not depend on x:

$$\max_x \log p(x|y) = \max_x -\frac{1}{2} \|Ax - y\|^2_{\Sigma^{-1}} - \frac{1}{2\eta} \|x\|^2$$

$$\min_x -\log p(x|y) = \min_x \underbrace{\frac{1}{2} \|Ax - y\|^2_{\Sigma^{-1}}}_{\substack{D(x,y) \text{ (Data Fidelity)} \\ \alpha \propto \log(p(y|x))}} + \underbrace{\frac{1}{2\eta} \|x\|^2}_{\substack{R(x) \text{ (Regularizer)} \\ \alpha \propto \log(p(x))}}$$

Conclusion: The variational formulation of inverse problems corresponds to the Maximum A Posteriori (MAP) estimation.

X-rays and Computed Tomography

Discovery of X-rays

In 1895, Wilhelm Röntgen noticed "rays of mysterious origin", which he called X-rays. Within a month (22.12.1895), the first radiograph of the hand of Röntgen's wife was made in Würzburg. This immediate application to imagine the human body marks the birth of medical imaging.

Nature and Properties of X-rays

X-rays are electromagnetic waves. They are a form of ionizing radiation—radiation with enough energy to eject electrons from an atom.

What needs to hold: Bound energy < Unbound energy + Electron Energy.

The binding energy of hydrogen is 13.6 eV. For a Medical CT you need around 100keV, for Mammography you need around 20keV.

Forms of Ionizing Radiation

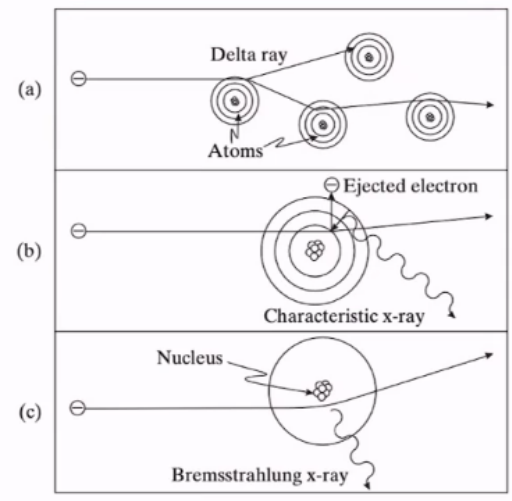
1. **Particulate Radiation**: Any subatomic particle (proton, neutron, electron) with enough kinetic energy to be ionizing.
2. **Electromagnetic Radiation**: Can act as a wave or a particle (photon). If energy > 13.6 eV (binding energy of hydrogen electron), it is considered ionizing.

> *Remark.*
>
> $E = h\nu \quad$ and $\quad \lambda = \frac{c}{\nu}$
>
> The variables stand for:
> - h: Planck's constant
> - ν: frequency
> - λ: wavelength
> - c: speed of light

Interaction of Energetic Electrons with Matter

- **Collision transfer** (99% $\rightarrow$ heat): Collision with other electrons until kinetic energy is exhausted. If they bump into each other, then energy can be transfered to the other electron which then will emit infrared photons, which is heat.
- **Radiative transfer** (1% $\rightarrow$ X-ray):

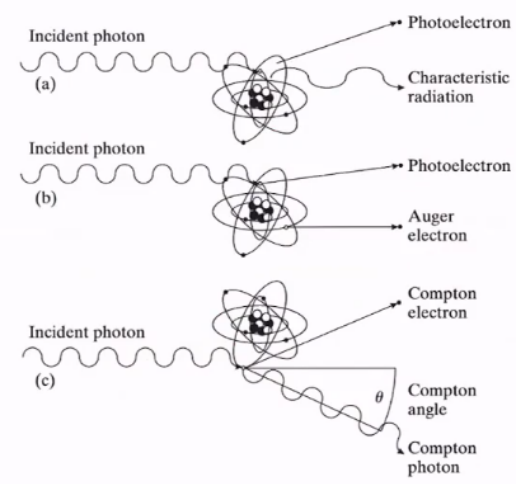
- Eject inner shell electron, generating **characteristic X-ray radiation**.
- Electron flies close to the atom nucleus and is braked by nucleus, generating **Bremsstrahlung X-ray**.



Interaction of Electromagnetic Radiation with Matter

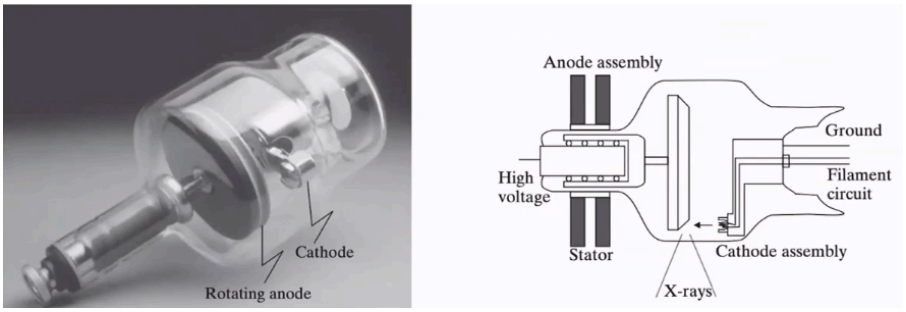
- **Photoelectrical Effect**: Photon hit an atom and thereby ejects an inner shell electron. The energy loss of the photon is $h\nu - E_B$. Afterwards the photon moves on with a smaller frequency. Then a free flying electorn can fill the hole again and that leads to an X-Ray. It could also happen that that released X-Ray directly hits again an electron on the outer-shell and that is then a Auger electron.
- **Compton Scattering**: Photon interacts with outer-shell electrons, yielding a Compton electron and a scattered photon with less energy. This energy loss depends on the deflection angle and is defined by this formula:

$$E_c = h\nu - h\nu' = h(\nu - \nu')$$

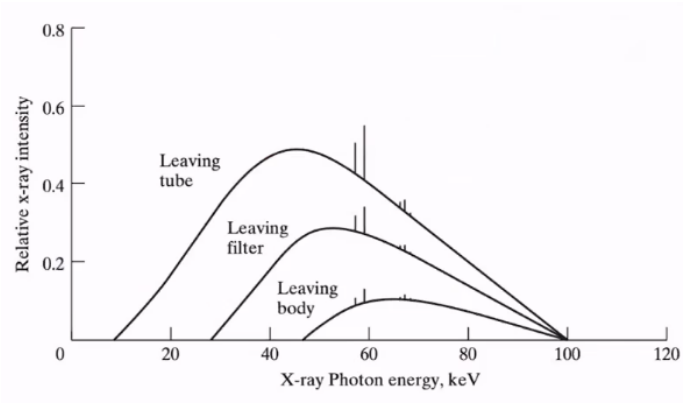


X-Ray Generation

You have a tube current that creates a tube voltage between a tungsten rotating anode and a static cathode. The voltage eject electrons which hit the rotating anode and then it kicks out X-rays.

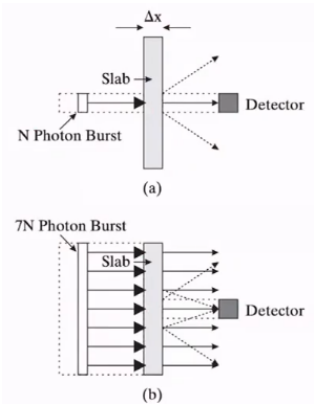


We do want the high energy photons that come from the "leaving the body" distribution.



Attenuation of Electromagnetic Radiation

Consider a narrow beam geometry with an X-ray source and a detector.



Let N be the number of photons leaving the source and N' be the photons hitting the detector. Suppose n photons are lost in a thickness Δx:

$$n = N\mu\Delta x$$

where μ is some kind of material coefficient. The change in photons is:

$$\Delta N = N' - N = -n = -\mu N \Delta x$$

In the limit $\Delta x \to 0$:

$$dN = -\mu N dx \to \frac{dN}{N} = -\mu dx$$

Integrating both sides:

$$\int \frac{dN}{N} = -\int \mu dx \rightarrow \log(N) = -\int \mu dx + C$$

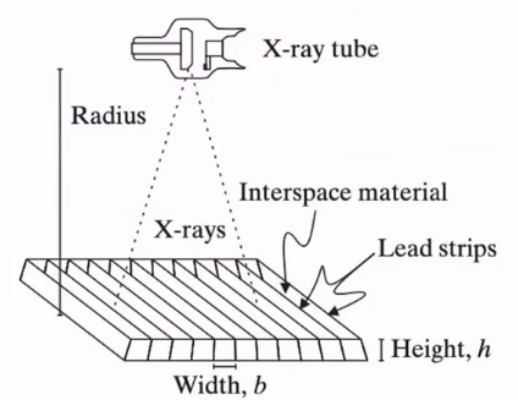
For $x = 0$, $N(0) = N_0$, thus $C = \log(N_0)$.

$$N(x) = N_0 \exp\left(-\int \mu(x)dx\right)$$

We have now 2 quantities to describe this setting:
- photon fluence rate (number of photons over some area of some time) $\psi = \left(\frac{N}{A}\delta t\right)$
- itensity of a beam $I = \psi E$

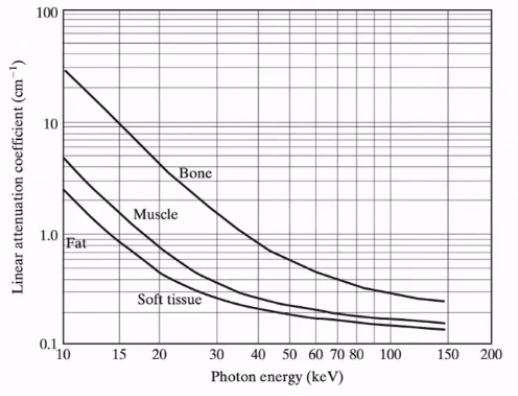
Narrow Beam vs. Broad Beam

Scattering (Compton effect) causes photons to hit the detector from multiple angles with different energy levels and the monoenergetic assumption often fails. The rescue is to use the detector collimation to ensure only primary (non-scattered) rays are measured.

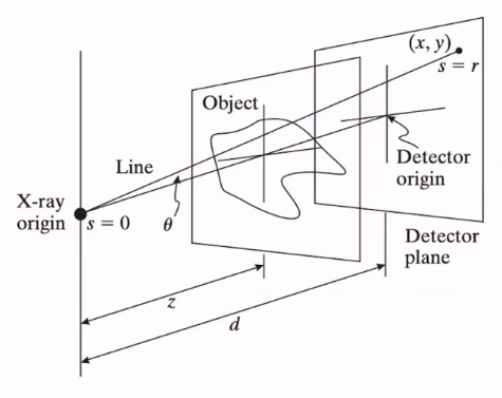


Attenuation of different tissue types

Different tissue types have different μ. That enables us to image the inner world of the body.



Projection Radiographic System



$$N(x) = N_0 \cdot \exp\left(-\int_0^x \mu(\hat{x}, E) d\hat{x}\right)$$

We have the Spectrum S as

$$S(x, E) = S_0(E) \cdot \exp\left(-\int_0^x \mu(\hat{x}, E) d\hat{x}\right)$$

So we get a formula for the intensity:

$$I(x) = \int_0^\infty \underbrace{\frac{N(x)}{A \cdot \Delta t}}_{S(x,E)} \cdot E dE$$

$$I(x) = \int_0^\infty S_0(E) E \exp\left(-\int_0^S \mu(\hat{x}, E) d\hat{x}\right) dE$$

And this is rather complicated, so we do some simplification: Assuming monoenergetic X-rays with effective energy E:

$$S(E) = \delta(\overline{E})$$

We say here that the Spectrum should be describale by some $\delta(\overline{E})$ which is the monoenergetic radiation

$$I(x) = I_0 \cdot \exp\left(-\int_0^x \mu(\tilde{x}, \overline{E}) d\tilde{x}\right)$$

This is a non-linear problem. The unkown here are the μ. We already have the measurements. Let's further simplify:

$$\frac{I(x)}{I_0} = \exp\left(-\int_0^x \underbrace{\mu(\tilde{x}, \overline{E})}_{\mu(\tilde{x})} d\tilde{x}\right)$$

$$\log \frac{I(x)}{I_0} = -\int_0^x \mu(\tilde{x}) d\tilde{x} \quad \Leftrightarrow \quad \underbrace{\log \frac{I_0}{I(x)}}_{\hat{y}} = \int_0^x \mu(\tilde{x}) d\tilde{x}$$

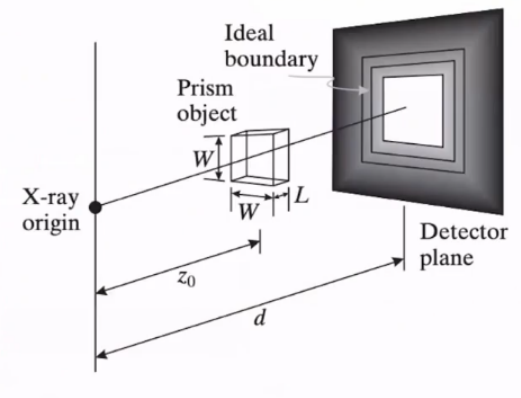
$$\hat{y}_i = \int_0^x \mu(\hat{x}) d\hat{x}$$

With that we have a linear inverse problem. The only thing that changed was first I was realated to a Poisson Distribution and was related to counting. Now it is more gaussian. And that is somehing we can work with leaving us with essentially

$$\hat{y}_i = \vec{a}_i^T \vec{\mu}$$

Blurring and Noise

Blurring sources are Penumbra (due to focal spot size), Compton scattering, and detector resolution.



Definition 13 (local contrast) .

$$C = \frac{I_t - I_b}{I_b}$$

I_t: target intensity (lesion), I_b: background intensity

Definition 14 (Signal to noise ratio) .

$$\text{SNR} = \frac{I_t - I_b}{\sigma_b} = \frac{C \cdot I_b}{\sigma_b}$$

σ_b: Std in the background

Recall

$$I = \frac{N \cdot E}{A \Delta t} \Rightarrow I_b = \frac{N_b E}{A_b \Delta t}$$

N is the number of measured photons $\Rightarrow$ counting discrete events. We know

$$N \sim \text{Poisson}\left(N \left(\frac{E}{A_b \Delta t} \right)^2 \right)$$

with variance

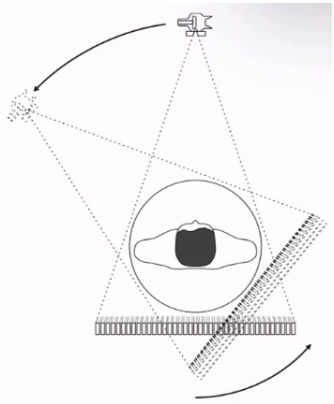
$$\sigma_b^2 = N_b \left(\frac{E}{A\Delta t} \right)^2$$

$$\Rightarrow \text{SNR} = \frac{C \cdot I_b}{\sigma_b} = \frac{C \cdot \frac{N_b E}{A_b \Delta t}}{\sqrt{N_b} \frac{E}{A_b \Delta t}} = C \cdot \sqrt{N_b}$$

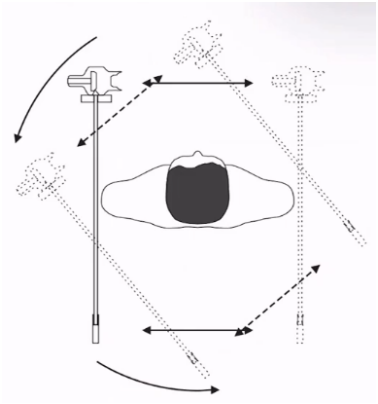
To increase SNR:
1. Increase C (maybe with contrast agent)
2. Increase photon count N_b (not so healthy for the patient)

Computed Tomography (CT)

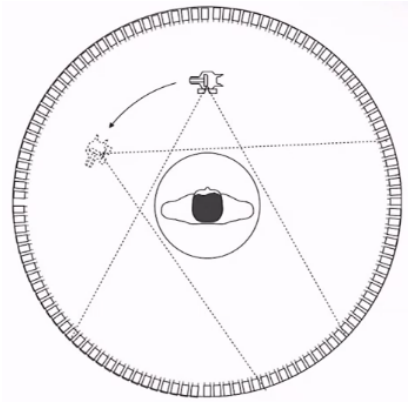
Tomography (from Greek **tomos** "slice" and **grapho** "to write") involves imaging by sectioning a volume using projected radiographs from different orientations. Compared to X-ray we have here a lot more unkowns as we not need to reconstruct a projection, but we need to reconstruct every single slice. You basically have your X-ray source again and a line of detectors with a filter to compensate for the compton scattering.



In the first generation you just had 1 detector, so it took forever to scan.



And nowadays you have detectors all around you and it only takes seconds to scan.



First the scanners measured the energy. Nowadays they are able to measure each photon individually. With that you get much sharper scans. And historic overview can be found here:

Generation	Source	Source Collimation	Detector	Detector Collimation	Source-Detector Movement	Advantages	Disadvantages
1G	Single x-ray tube	Pencil beam	Single	None	Move linearly and rotate in unison	Scattered energy is undetected	Slow
2G	Single x-ray tube	Fan beam, not enough to cover FOV	Multiple	Collimated to source direction	Move linearly and rotate in unison	Faster than 1G	Lower efficiency and larger noise because of the collimation in detectors
3G	Single x-ray tube	Fan beam, enough to cover FOV	Many	Collimated to source direction	Rotate in synchrony	Faster than 2G, continuous rotation using a slip ring	More expensive than 2G, low efficiency
4G	Single x-ray tube	Fan beam covers the FOV	Stationary ring of detectors	Cannot collimate detectors	Detectors are fixed, source rotates	Higher efficiency than 3G	High scattering since detectors are not collimated
5G (EBCT)	Many tungsten anodes in single large tube	Fan beam	Stationary ring of detectors	Cannot collimate detectors	No moving parts	Extremely fast, capable of stop-action imaging of beating heart	High cost, difficult to calibrate
6G (Helical CT)	3G/4G	3G/4G	3G/4G	3G/4G	3G/4G plus linear patient table motion	Fast 3D images	A bit more expensive
7G (Multiple-row detector CT)	Single x-ray tube	Cone beam	Multiple arrays of detectors	Collimated to source direction	3G/4G/6G motion	Fast 3D images	Expensive

Image formation

Recall from radiography:

$$I_d = \int_0^{E_{\max}} S_0(E) \cdot E \cdot \exp\left(- \int_{\Gamma_d} \mu(s, E) ds \right) dE$$

$S_0(E)$ is the polyenergetic spectrum from the X-ray source, so the distribution of the energy of the photons. E is the Energy level. $-\int_{\Gamma_d} \mu(s, E) ds$ is the linear attenuation, the $\mu(s, E)$ is the linear attenuation coefficient that we want to recover as it gives us the distribtuion of the different materials inside the body along a ray. Γ_d is the curve that the ray traverses, so in our case a straight line (we neglect the particle interaction inbetween).

Let's assume an $\overline{E}$ exists and $\overline{E}$ is the monoenergetic effective energy that yields the same intensity I_d. With that we can simplify to

$$I_d = I_0 \exp\left(- \int_{\Gamma_d} \mu(s, \overline{E}) ds \right)$$

and this is a linear problem. So basically we assume that the spectrum is very peaky anyway, so that almost all photons have the same energy. When the spectrum is broader spreaded, this assumption

breaks and we get a lot of artifacts that are called "beam hardening". This whole problem with the artifacts vanishes when we use the newer technology of just counting single photons.

$$g_d = -\ln\left(\frac{I_d}{I_0}\right) = \int_{\Gamma_d} \mu(s, \overline{E}) ds = \underbrace{a_d^T}_{d^{\text{th}} \text{ column of } A} \qquad \mu \Rightarrow Ax = y$$

Parallel-Ray Reconstruction

In newer Scaners, we measure with a cone instead of parallel lines, as the source rotates, different rays from different time-points are parallel to each other and in post-processing this can be rearranged again to have parallel rays.

Let's fix a 2D-line:

$$\Gamma(l, \theta) = \left\{ \begin{pmatrix} x \\ y \end{pmatrix} : x \cdot \cos\theta + y \cdot \sin\theta = l \right\}$$

Then, the line integral reads as:

$$g(l, \theta) = \int_{-\infty}^{\infty} f(x(s), y(s)) ds$$

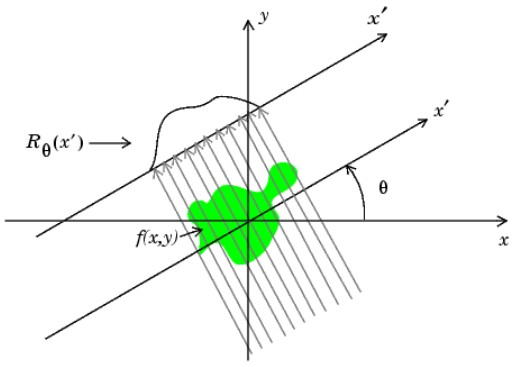
where we have

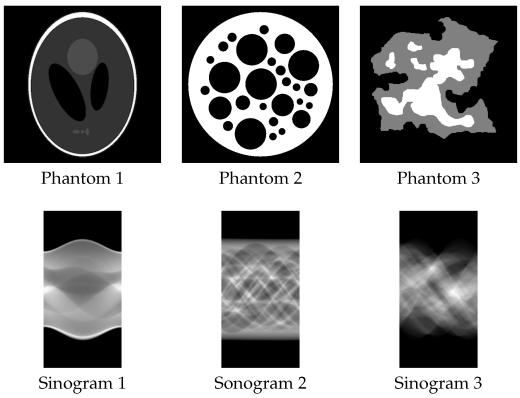
$$\begin{pmatrix} x(s) \\ y(s) \end{pmatrix} = \underbrace{\begin{pmatrix} l \cdot \cos\theta \\ l \cdot \sin\theta \end{pmatrix}}_{\text{original vector}} + \underbrace{\begin{pmatrix} -\sin\theta \\ \cos\theta \end{pmatrix}}_{\text{normal vector}} s$$

Then we get the Radon Transform

$$g(l, \theta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) \cdot \underbrace{\delta(x\cos\theta + y\sin\theta - l)}_{\text{Dirac } \delta = \begin{cases} \infty & \text{if } \lambda = 0 \\ 0 & \text{else} \end{cases} \text{ of } \Gamma(l,\theta)} dx dy$$

For a fixed angle θ, we call $g(l, \theta_i)$ a **projection** and for all theta we call it **sinogram**.





Reconstruction Methods

Backprojection

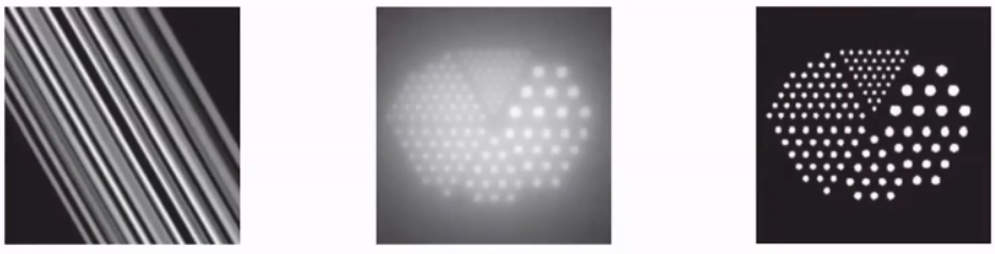
Idea: Simply project or smear each measurement $g(l, \theta)$ back onto the image, because we at the spots with the heighest intensities there must have been the most material. For one angle:

$$b_\theta(x, y) = g(x \cos\theta + y \sin\theta, \theta)$$

Taking all angles into account we:

$$f_b(x, y) = \int_0^\pi g(x \cos\theta + y \sin\theta, \theta)\, d\theta$$

The first image is the b_θ and the middle image would be now the $f_b(x, y)$. The right image is the ground truth.



Projection-Slice Theorem (Central Slice Theorem)

$$g(l, \theta) \overset{1D \text{ FFT}}{\longrightarrow} G(\rho, \theta) = \mathcal{F}_{1D}(g(l, \theta)) = \int_{-\infty}^{\infty} g(l, \theta) \cdot \exp(-i2\pi\rho l) dl$$

$$= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y)\delta(x \cos\theta + y \sin\theta - l) \exp(-i2\pi\rho l) dl dy dx$$

The only time when the δ is not 0 is when $l = x \cos\theta + y \sin\theta$. With that we can get rid of one integral:

$$= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) \cdot \exp(-i2\pi\rho(x \cos\theta + y \sin\theta)) dx dy$$

Recall definition of 2D Fourier transform: $u = \rho \cos\theta, v = \rho \sin\theta$

$$F(u,v) = \mathcal{F}(f) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\cdot \exp(-i2\pi(ux+vy))dxdy$$

$$\Rightarrow \underbrace{F(\rho\cos\theta, \rho\sin\theta)}_{\text{2D Fourier transform of image}} = \underbrace{G(\rho,\theta)}_{\text{1D Fourier transf. of projection}}$$

Filtered Backprojection (FBP):

$$f(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v)\exp(i2\pi(ux+vy))dudv$$

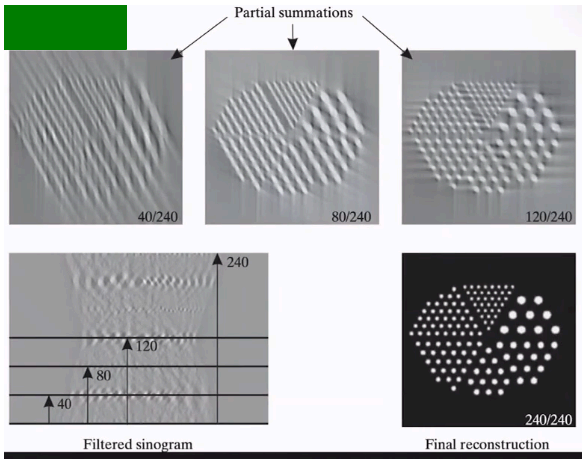
Change of variables: $u = \rho\cos\theta, v = \rho\sin\theta \Rightarrow (u) = \begin{pmatrix}\rho\cos\theta\\ \rho\sin\theta\end{pmatrix}$

$$\int_\Gamma f(g(x))dx = \int f(\xi)\cdot|\det\frac{\partial g}{\partial\xi}|\ d\xi$$

$$\frac{\partial u}{\partial(\rho,\theta)} = \begin{pmatrix}\cos\theta & \rho\sin\theta\\ \sin\theta & -\rho\cos\theta\end{pmatrix} \Rightarrow |\frac{\partial(u)}{\partial(\rho,\theta)}| = |-\rho\cos^2\theta - \rho\sin^2\theta\ | = \rho$$

$$\Rightarrow f(x,y) = \int_{-\infty}^{\infty}\int_0^{\pi} F(\rho\cos\theta,\rho\sin\theta)\exp(i2\pi\rho(\cos\theta x+\sin\theta y))\ |\rho|\ d\theta d\rho$$

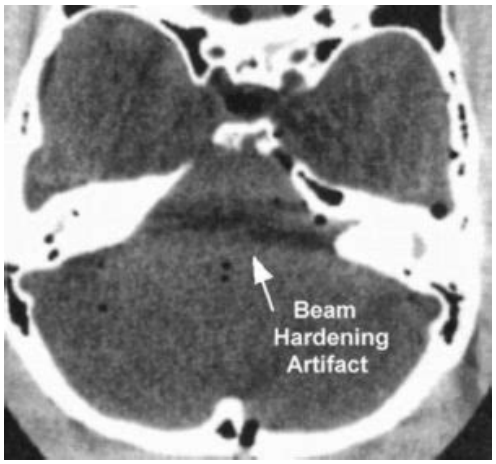
$$= \int_{-\infty}^{\infty}\int_0^{\pi} G(\rho,\theta)\cdot\exp(i2\pi\rho(\cos\theta x+\sin\theta y))\ |\rho|\ d\theta d\rho$$



Artifacts and Hounsfield Units

Aliasing: Streak artifacts due to insufficient number of projections.

Beam Hardening: Caused by energy-selective attenuation; low-energy photons are absorbed more easily, shifting the spectrum toward "harder" (higher energy) X-rays.



Definition 15 (Hounsfield Units (HU)) . Standardized scale to compare different materials in CT scans:

$$h = 1000 \cdot \frac{\mu - \mu_{\text{Water}}}{\mu_{\text{Water}} - \mu_{\text{Air}}}$$

Substance		HU
Air		−1000
Fat		−120 to −90
Bone	Cancellous	+300 to +1900
Other blood	Unclotted	+13 to +50
	Clotted	+50 to +75
Fluids	Water	0
	Urine	−5 to +15
	CSF	+15

Substance		HU
Parenchyma	Lung	-700 to -600
	Kidney	$+20$ to $+45$
	Liver	60 ± 6
	Muscle	$+35$ to $+55$
	White matter	$+20$ to $+30$
	Grey matter	$+37$ to $+45$

Remark. **Historical Note**: The development of CT was funded in part by EMI (the Beatles' record label), leading to Hounsfield's Nobel Prize.

Learned Reconstruction Methods

Recall: Inverse Problems

Let $X = \mathbb{R}^n$ be the image space and $Y = \mathbb{R}^m$ be the measurement space. The inverse problem is defined as: $Ax = y$ where $A \in \mathbb{R}^{m \times n}$ is the forward operator.

One instances of that in Medical Imaging is Computed Tomography (CT) where y is the sinogram data A is the Radon transform.

Deep Learning Approaches

There are three main paradigms for integrating Deep Learning into the reconstruction pipeline:

1. Post-processing: Applying a Neural Network (NN) to an initial reconstruction (e.g., Filtered Backpropagation FBP) to remove artifacts.

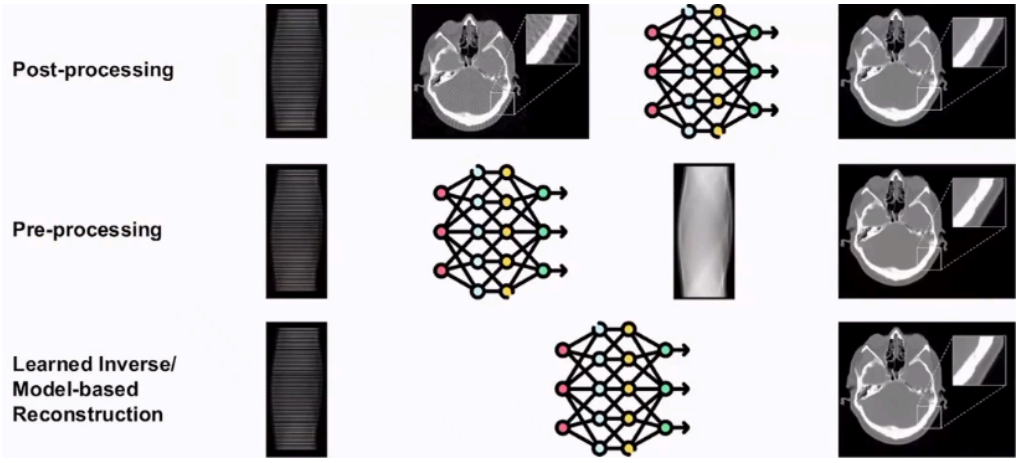
$$y \to \text{FBP} \to x_{\text{initial}} \to \text{NN} \to x_{\text{final}}$$

2. Pre-processing: Applying a NN to the raw data (sinogram/k-space) before reconstruction.

$$y \to \text{NN} \to y_{\text{full}} \to \text{FBP} \to x$$

3. Learned Inverse / Model-based Reconstruction: Replacing or augmenting the reconstruction operator itself.

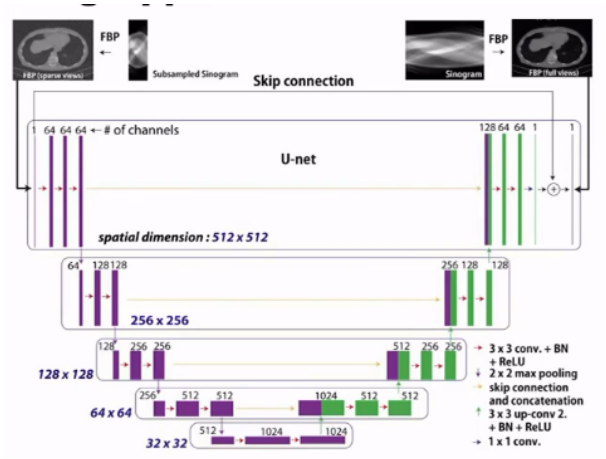
$$y \to \text{NN} \to x$$

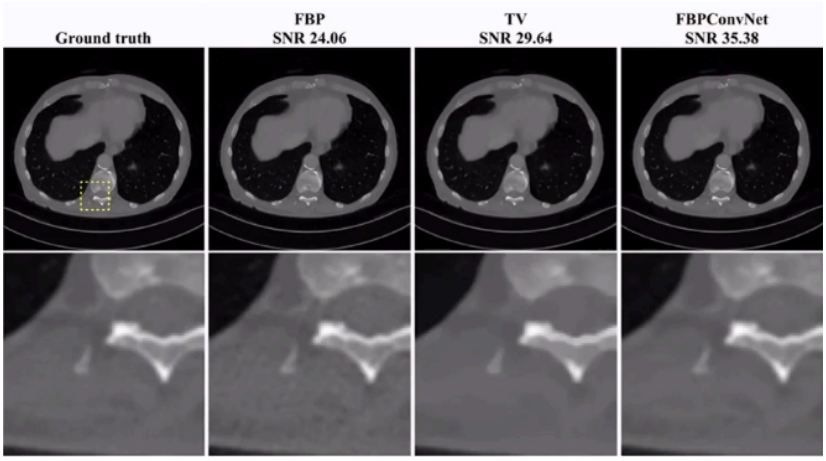


Post-processing Approach: FBPConvNet

The FBPConvNet uses a U-Net architecture to refine sparse-view FBP reconstructions.

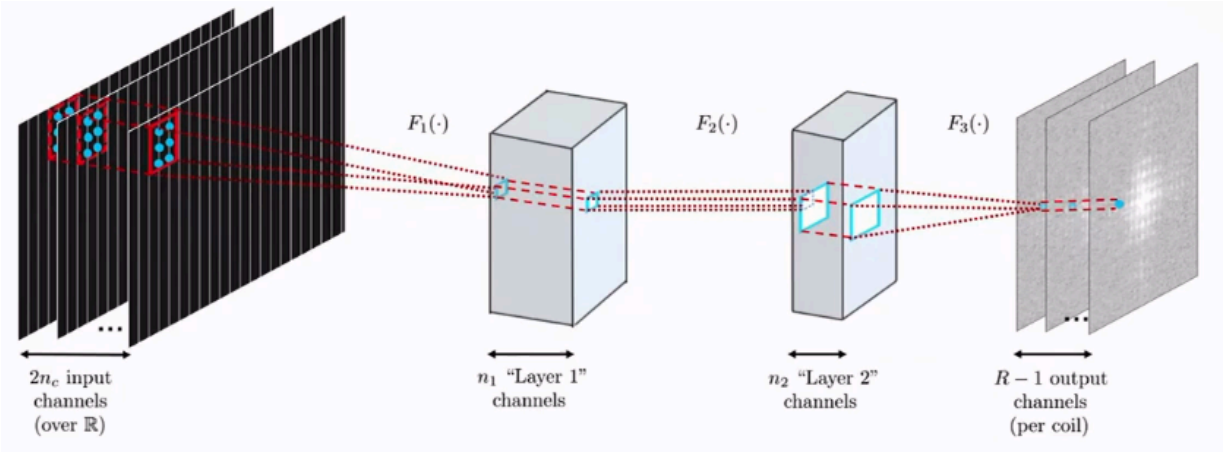
- Architecture: U-Net with skip connections and concatenation.
- Spatial Dimension: 512×512.

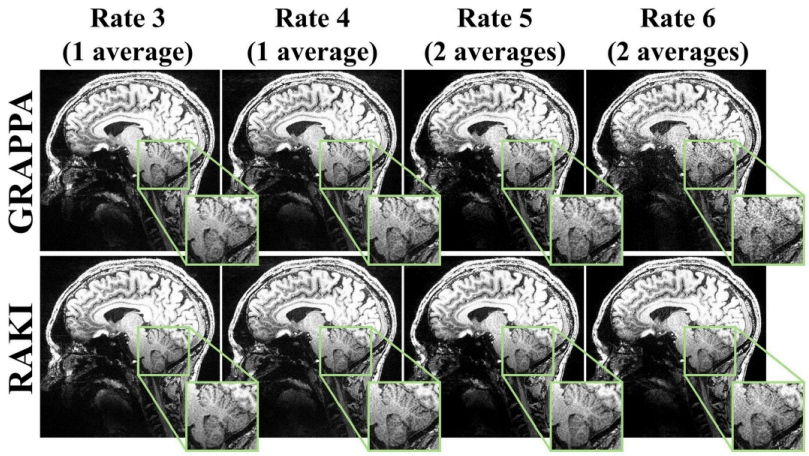




Pre-processing Approach: RAKI

RAKI (Scan-specific Robust Artificial-neural-networks for K-space Interpolation) is a database-free method for fast MRI imaging. Here they do not measure every single fourier coefficent, but uses CNN layers to learn to interpolate missing data from the auto-calibration signal (ACS) of the specific scan. It also outperforms classical GRAPPA, especially at high acceleration rates.





Model-based Reconstruction

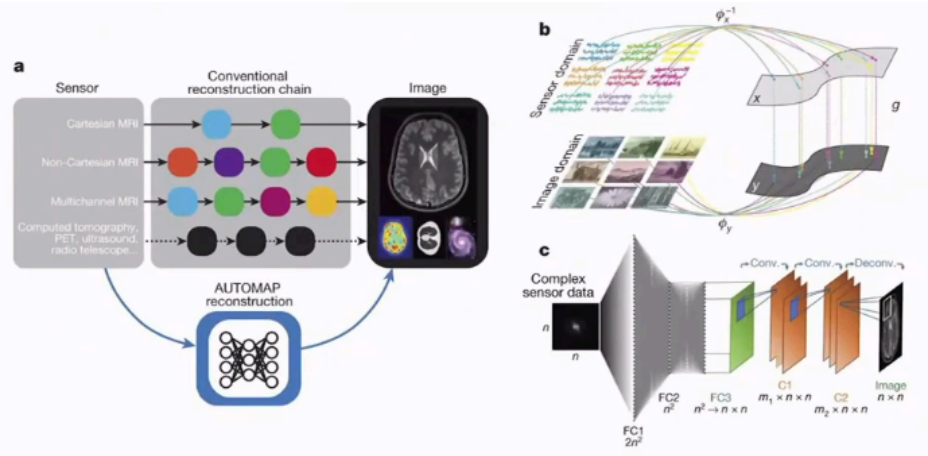
In model-based approaches, we have 2 options for getting a solution:

- via a reconstruction operator $B(y)$ that approximates the inverse A^{-1} directly which is a robust approximation (Fourier scans are often noisy)
- via an iterative algorithms by solving

$$B(y) = \arg\min_x \frac{1}{2}\|Ax - y\|_2^2 + R(x)$$

Learned Inversion: AUTOMAP

The most popular algorytm for learned inversion is AUTOMAP. You put in the fourier data and then they directly put it to a fully connected layer followed by some Convolution layers. That might be not a good idea as when the fourier signal is only shifted by one pixel, you have entirely different results.



Learned Model-based Reconstruction

Recall: Lipschitz Continuous Gradients

Let $f : \mathbb{R}^N \to \mathbb{R}$ be differentiable and $0 < L < \infty$ such that:

$$\|\nabla f(x) - \nabla f(y)\| \leq L \|x - y\| \quad \forall x, y \in \mathbb{R}^N$$

Then, we have the quadratic upper bound:

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2} \|x - y\|^2 \quad \forall x, y \in \mathbb{R}^N$$

Proximal Gradient Method (PGM)

To optimize a function $f : \mathbb{R}^N \to \mathbb{R}$ decomposable into $f(x) = g(x) + h(x)$, where $g(x) \in C^{1,k}(\mathbb{R}^N)$ and h is closed (l.s.c), convex, and proper (proximal mapping can be computed).

Iteration Rule:

$$x_{k+1} = \text{prox}_{\frac{1}{L}h}\left(x_k - \frac{1}{L}\nabla g(x_k)\right)$$

where L is the Lipschitz constant.

Definition of Proximal Operator:

$$\text{prox}_{\frac{1}{L}h}(\bar{x}) = \arg\min_x \left(\frac{1}{L}h(x) + \frac{1}{2}\|x - \bar{x}\|_2^2\right)$$

How to read the Proximal Operator formula:

The x inside the norm is your ideal destination according to the gradient descent of the smooth function g. Notice that $g(x)$ is evaluated at the new potential point x, not at the old point $\bar{x}$. The formula is searching for an x that is a compromise. The quadratic term $\|x - \bar{x}\|_2^2$ penalizes you for moving too far from your gradient target, while $h(x)$ pulls you toward points that keep the non-smooth penalty low.

Accelerated Version: FISTA (Fast Iterative Shrinkage & Thresholding Algorithm).

Derivation of PGM

Assuming ∇g is Lipschitz continuous:

$$g(y) \leq g(x) + \langle \nabla g(x), y - x \rangle + \frac{L}{2} \|y - x\|^2$$

This is $g(x)$, but we want also to have a $h(x)$ to have the same form as in the definition above. So we add $h(y)$ to both sides:

$$f(y) = g(y) + h(y) \leq g(x) + h(y) + \langle \nabla g(x), y - x \rangle + \frac{L}{2} \|y - x\|^2 =: \tilde{f}(y)$$

We want to have a upper bound to our function and we want to optimize f. To minimize the upper bound $\tilde{f}(y)$, we take the subgradient with respect to y:

$$\frac{\partial}{\partial y}\tilde{f}(y) = \frac{\partial}{\partial y}h(y) + \nabla g(x) + L(y - x)$$

And then we can set it to 0

$$\frac{\partial}{\partial y}h(y) + Ly = -\nabla g(x) + Lx$$

$$y + \frac{1}{L}\frac{\partial}{\partial y}h(y) = x - \frac{1}{L}\nabla g(x)$$

$$\underbrace{\left(I + \frac{1}{L}\frac{\partial}{\partial y}h\right)(y)}_{\text{This is a function of y}} = x - \frac{1}{L}\nabla g(x)$$

Solving for y using the resolvent (equivalent to proximal mapping):

$$y = \underbrace{\left(I + \frac{1}{L}\partial h\right)^{-1}}_{\text{resolvent equivalent to prox}} \left(x - \frac{1}{L}\nabla g(x)\right)$$

$$y = \text{prox}_{\frac{1}{L}h}\left(x - \frac{1}{L}\nabla g(x)\right)$$

Example 11 — Lasso .

$$\min_x \frac{1}{2}\left\|Ax - y\right\|_2^2 + \lambda \left\|x\right\|_1$$

Where $g(x) = \frac{1}{2}\left\|Ax - y\right\|_2^2$ and $h(x) = \lambda \left\|x\right\|_1$. Site note: The derivative of $\lambda \left\|x\right\|_1$ is not Lipschitz continuous

gradient:

$$\nabla g(x) = A^T(Ax - y)$$

Proximal Operator (Soft Thresholding):

$$\text{prox}_{\frac{1}{L}h}(y)_i = \max\left(|y_i| - \frac{\lambda}{L}, 0\right) \cdot \text{sign}(y_i)$$

Computing L:

$$\left\|\nabla g(x) - \nabla g(z)\right\|_2 \leq L \left\|x - z\right\|_2$$

$$\left\|A^TA(x - y) - A^{T(Az-y)}\right\|_2 = \left\|A^TA(x - z)\right\|_2 \leq L \left\|x - z\right\|_2$$

$$\left\|A^TA\right\|_2 \left\|x - z\right\|_2 \leq L \left\|x - z\right\|_2$$

$$\left\|A^TA\right\|_2 \leq L$$

(The largest singular value of A)

Example 12 — Fields of Experts regularization for MRI reconstruction .

$$A(x) = M \odot F(x) = D_M F x$$

where D_M is a diagonal matrix

Our goal is to solve

$$\min_x \frac{1}{2}\left\|A(x) - y\right\|_2^2 + \rho(Wx)$$

Where $\frac{1}{2}\left\|A(x) - y\right\|_2^2$ is smooth and convex, and ρ is smooth (but not necessarily convex). The $\frac{1}{2}\left\|A(x) - y\right\|_2^2$ will be our $h(x)$ and the ρ is our $g(x)$

$$\nabla g(x) = W^T \nabla_x \rho(Wx)$$

$$\begin{aligned}\text{prox}_{\frac{1}{L}l}(z) &= \arg\min_x \frac{1}{2}\|x-z\|^2 + \frac{1}{L}h(x) \\ &= \arg\min_x \underbrace{\frac{1}{2}\|x-z\|^2 + \frac{1}{L}\frac{1}{2}\|A(x)-y\|_2^2}_{l(x)}\end{aligned}$$

Now we can compute again the graident and set it to 0:

$$\frac{\partial l}{\partial x} = (x-z) + \frac{1}{L}A^{T(x)}(A(x)-y) = 0$$

$$x + \frac{1}{L}F^T M(MFx - y) = z$$

$$x + \frac{1}{L}F^T MMFx = z + \frac{1}{L}F^T My$$

Assuming $M^2 = M$ (projection matrix):

$$\left(I + \frac{1}{L}F^T MF\right)x = z + \frac{1}{L}F^T My$$

Given $F^T F = I$:

$$\left(F^T F + \frac{1}{L}F^T MF\right)x = z + \frac{1}{L}F^T My$$

$$F^T\left(I + \frac{1}{L}M\right)Fx = z + \frac{1}{L}F^T My$$

$$\left(I + \frac{1}{L}M\right)Fx = Fz + \frac{1}{L}My$$

$$Fx = \frac{Fz + \frac{1}{L}My}{I + \frac{1}{L}M}$$

$$x = F^T\left(\frac{Fz + \frac{1}{L}My}{I + \frac{1}{L}M}\right)$$

Key Learning Principles:

1. Bilevel Optimization: Learning parameters by solving an optimization problem within another. (= supervised learning)
2. Contrastive Learning: Learning representations by comparing positive and negative pairs. (= semi-supervised)
3. Distribution Matching: Ensuring the reconstructed distribution matches the ground truth distribution. (= unsupervised)
4. Plug & Play (PnP): Using a pre-trained deep denoiser as a proximal operator in iterative algorithms. (= unsupervised)

Bilevel Optimization

Given a set of paired training samples $D = (x_i, y_i)_{\{i=1\}}^n$, in bilevel optimization, we want to solve the following learning problem:

Upper level problem:

$$\min_\theta L(\theta) = \sum_{i=1}^n \|\hat{x}_i(\theta, y_i) - x_i\|_2^2 \text{ subject to}$$

Lower level problem

$$\hat{x}_i(\theta, y_i) = \arg\min_x \left\{ E_\theta(x, y_i) = \frac{1}{2} \|Ax - y_i\|_2^2 + R_\theta(x) \right\}$$

The challenge is to compute the gradient

$$\frac{\partial L}{\partial \theta} = \sum_{i=1}^n \left(\frac{\partial \hat{x}_i(\theta, y_i)}{\partial \theta} \right)) (\hat{x}_i(\theta, y_i) - x_i)$$

The difficult part:

$$\frac{\partial \hat{x}_i(y_i, \theta)}{\partial \theta}$$

Let's do it step by step:

1. Solve the lower level problem (with sufficient precision).
$$\nabla_x E_\theta(\hat{x}_i(y_i, \theta), y_i) \approx 0$$
2. Assume that $E_\theta \in C^2(\mathbb{R}^N)$ with invertible Hessian $H(\theta) = \nabla_x^2 E_\theta$. Then, the Implicit Function Theorem (IFT) guarantees the existence of a continuously differentiable local solution map $\theta \to \hat{x}_i(y_i, \theta)$.
3. The first order optimality condition of E_θ is:
$$\frac{\partial}{\partial \theta}(\nabla_x E(\hat{x}(\theta), \theta)) = \frac{\partial}{\partial \theta}(0) = 0$$
$$\frac{\partial \hat{x}(\theta)}{\partial \theta} \nabla_x^2 E(\hat{x}(\theta), \theta) + \frac{\partial}{\partial \theta} \nabla_x E(\hat{x}(\theta), \theta) = 0$$
$$\frac{\partial \hat{x}(\theta)}{\partial \theta} = - \underbrace{\frac{\partial}{\partial \theta} \nabla_x E(\hat{x}(\theta), \theta)}_{\substack{\text{Jacobian o the} \\ \text{lower level energy gradient}}} \underbrace{\left(\nabla_x^2 E(\hat{x}(\theta), \theta)\right)^{-1}}_{\substack{\text{inverse Hessian of} \\ \text{lower level energy}}}$$

Unrolling (truncated optimization):

Using the IFT requires that we have $\nabla E(\hat{x}) \approx 0$. So, we need to approximate this using an optimization algorithm (e.g., PGD)

$$x_{k+1} = T(x_k; \theta) \quad \text{for } k = 0 \ldots K-1$$

E.g., if T implements gradient descent, we have $T : x \to x - \frac{1}{2} \nabla E_\theta(x)$. If we use K-steps, we get a computational chain:

$$x_0(y) \overset{T_\theta}{\to} x_1(\theta) \overset{T_\theta}{\to} x_2(\theta) \ldots \overset{T_\theta}{\to} x_K(\theta)$$

In unrolling, we simply set $\hat{x} = x_K(\theta)$; $L(\theta) = \sum_{i=1}^{n} \|x_K^i(\theta) - x_i\|_2^2$. The gradient $\frac{\partial L(\theta)}{\partial x_K}$ can simply be computed by back-propagation. The advantage is that it is easy implementable, but you have large memory consumtion

Jacobian-free backpropagation (*truncated backpropagation*)

Assume x_K approaches the optimum $\nabla E(x_K) \approx 0$. Instead of backprop through the entire sequence $(x_k)_{\{k=0\}}^{K}$, only the last K_B steps are considered:

$$\frac{\partial}{\partial \theta} x_K(\theta) \approx \sum_{k=K-K_B}^{K_B} \frac{\partial}{\partial \theta} T_{\theta(x_{k-1})} \cdot \nabla_x T_{\theta(x_k)} \ldots \nabla_x T_{\theta(x_K)}$$

If the lower level problem is sufficiently regular, this approximation error decays exponentially with K_B. Interestingly, in practice often $K_B = 1$ works quite well. So this method is easy to implement and has a small memory requirement. However you only get a approximation of the real solution.

Contrastive Learning

In contrast to bilevel-opt. that tries to learn a reconstruction scheme end-to-end, contrastive learning learns regularizers by contrasting "good" & "bad" images.

Adversarial Regularization (*AR*)

Let $\{x_i\}_{i=1}^{n} \sim p_X$ be samples from desired images (= no measurements required). Let $A^\dagger$ be a simple reconstruction operator (e.g., FBP, regularized pseudo-inverse). Then $A^\dagger$ yields a push-forward distribution $p_{A^\dagger y}$ of denablaed images.

The key idea of AR is to train a model to be a discriminator (c.f. GAN), i.e.,

$$L(\theta) = \frac{1}{n} \sum_{i=1}^{n} R_\theta(x_i) - \frac{1}{m} \sum_{j=1}^{m} R_\theta(A^\dagger y_j) + \underbrace{\lambda E_x \left[(\|\nabla_x R_\theta(x)\| - 1)_+^2 \right]}_{\text{gradient penalty in Wasserstein GANs}}$$

where $R_\theta : \mathbb{R}^N \to \mathbb{R}_+$ and $y_j = Ax_j + n$ with $n \sim \mathcal{N}(0, \sigma^2 \, \mathrm{I})$.

Gradient penalty in Wasserstein GANs: penalizes deviations from the 1-Lipschitz assumption in WGANs. The Lipschitz constant gives us the maximal gradient of a function. An advantage is that it is an easy training problem (at least to code). A disadvantage is that the training could be unstable and the balancing regularization & data fidelity is hard during inference.

Distribution Matching

Recall that regularizer $R(x)$ is associated with a Gibbs distribution:

$$p_\theta(x) = \frac{1}{z} \exp(-R_\theta(x))$$

The goal here is to align (match) $p_\theta(x)$ and $p_x(x)$.

Maximum Likelihood Training

$$p_\theta(x) = \frac{1}{z_\theta}\exp(-R_\theta(x)) \text{ with } z_\theta = \int_{\mathbb{R}^n} \exp(-R_\theta(x))dx$$

$$\begin{aligned}\theta &= \arg\max_\theta \mathbb{E}_{x\sim p_x}[\log p_\theta] = \arg\min_\theta \mathbb{E}_{x\sim p_x}[-\log p_\theta] = \arg\min_\theta D_{KL}(p_x \parallel p_\theta) \\ &= \arg\min_\theta \mathbb{E}_{x\sim p_x}[R_\theta(x)] + \log z_\theta\end{aligned}$$

$$\begin{aligned}\nabla_\theta D_{KL}(\cdot\parallel\cdot) &= \mathbb{E}_{x\sim p_x}[\nabla_\theta R_\theta(x)] + \frac{\partial}{\partial\theta}\log z_\theta = \mathbb{E}_{x\sim p_x}[\nabla_\theta R_\theta(x)] + \frac{1}{z_\theta}\partial\frac{z_\theta}{\partial\theta} \\ &= \mathbb{E}_{x\sim p_x}[\nabla_\theta R_\theta(x)] + \frac{1}{z_\theta}\int_{\mathbb{R}^n}\exp(-R_\theta(x))\cdot(-1)\cdot\nabla_\theta R_\theta(x)dx \\ &= \mathbb{E}_{x\sim p_x}[\nabla_\theta R_\theta(x)] - \int_{\mathbb{R}^n}\frac{\exp(-R_\theta(x))}{\int_{\mathbb{R}^n}\exp(-R_\theta(\tilde{x}))d\tilde{x}}\cdot\nabla_\theta R_\theta(x)dx \\ &= \mathbb{E}_{x\sim p_x}[\nabla_\theta R_\theta(x)] - E_{x\sim p_\theta}[\nabla_\theta R_\theta(x)]\end{aligned}$$

Which is the contrastive divergence An advantage is the easy training objective, but we need to sample from p_θ which is not that easy.

Score Matching

If we change the divergence from KL to the Fisher divergence, we get:

$$\arg\min_\theta \mathcal{L}_{\text{ESM}}(\theta) = \frac{1}{2}\mathbb{E}_{x\sim p_x}\left[\|\nabla_x \log p_x(x) - \nabla_x \log p_\theta(x)\|_2^2\right]$$

This aligns the Stein score $\nabla \log p_x$, but in practice, we cannot compute this. In reality, we approximate p_x by Gaussian smoothing and get $p_\sigma = p * G_\sigma$. Then, we get the denoising score matching objective:

$$\arg\min_\theta \mathcal{L}_{\text{DSM}}(\theta) = \frac{1}{2}\mathbb{E}_{x\sim p_x, n\sim G_\sigma}\left[\left\|\nabla_x \log p_\theta(x+n) - \left(-\frac{n}{\sigma^2}\right)\right\|_2^2\right]$$

$$s_\theta(x+n) \approx y = x + n$$

So one plus point here is a easy training and a problem is how to choose σ (training & inference schedule)

Proof of equivalence of ESM & DSM

$$\mathcal{L}_{\text{ESM}}(\theta) = \mathbb{E}_{y\sim p_\sigma}\left[\frac{1}{2}\left\|s_\theta(y) - \nabla_y \log p_\sigma(y)\right\|_2^2\right] = \mathbb{E}_{y\sim p_\sigma}\left[\frac{1}{2}\|s_\theta(y)\|_2^2 - \langle s_\theta(y), \nabla_y \log p_\sigma(y)\rangle + C\right]$$

$$\begin{aligned}S(\theta) &= \int_Y p_\sigma(y)\langle s_\theta(y), \nabla_y \log p_\sigma(y)\rangle dy = \int_Y \langle s_\theta(y), \nabla_y p_\sigma(y)\rangle dy \\ &= \int_Y \langle s_\theta(y), \nabla_y \int_X p_x(x)p(y|x)dx\rangle dy = \int_Y\int_X p_x(x)\cdot\langle s_\theta(y), \nabla_y p(y|x)\rangle dxdy \\ &= \int_Y\int_X p_x(x)\cdot p(y|x)\cdot\langle s_\theta(y), \nabla_y \log p(y|x)\rangle dxdy \\ &= \mathbb{E}_{x,y\sim p_{x,y}}\left[\langle s_\theta(y), \nabla_y \log p(y|x)\rangle\right]\end{aligned}$$

$$\Rightarrow \mathcal{L}_{\text{ESM}}(\theta) = \mathbb{E}_{y\sim p_\sigma}\left[\frac{1}{2}\|s_\theta(y)\|_2^2\right] - \mathbb{E}_{x,y\sim p_{x,y}}\left[\langle s_\theta(y), \nabla_y \log p(y|x)\rangle\right] + C_1$$

$$\Leftrightarrow \mathbb{E}_{x,y\sim p_{x,y}}\left[\frac{1}{2}\|s_\theta(y)\|_2^2 - \langle s_\theta(y), \nabla_y \log p(y|x)\rangle + C_1 + \frac{1}{2}\left\|\nabla_y \log p(y|x)\right\|_2^2\right]$$

$$= \mathbb{E}_{x,y\sim p_{x,y}}\left[\frac{1}{2}\left\|s_\theta(y) - \nabla_y \log p(y|x)\right\|_2^2\right]$$
$$= \mathcal{L}_{\text{DSM}}(\theta) + C_2$$

$$C_2 = -\|\nabla y \log p(y|x)\|_2^2 + \left\|\nabla_y \log p_\sigma(y)\right\|_2^2$$

$$p(y|x) = |2\pi\sigma^2 I\ |^{-\frac{1}{2}} \exp\left(-\frac{1}{2\sigma^2}\|x-y\|_2^2\right)$$

To summarize: The idea of score matiching is to match the log gradients of the distribution. This matching is hard, that's why we approximate it with smoothing, adding some noise. And we can show that by adding this noise, we can come up with a training objective which is $\arg\min_\theta \mathcal{L}_{\text{DSM}}(\theta) = \frac{1}{2}\mathbb{E}_{x\sim p_x, n\sim G_\sigma}\left[\left\|\nabla_x \log p_\theta(x+n) - \left(-\frac{n}{\sigma^2}\right)\right\|_2^2\right]$ which is easy to compute.

Plug & Play Optimization

We would like to solve:

$$\hat{x} = \arg\min_x \frac{1}{2}\ \|Ax-y\|^2 + \lambda R(x)$$

The Half Quadratic Splitting (HQS) algorithm decouples these terms:

$$\hat{x} = \arg\min_{\{x,z\}} \frac{1}{2}\ \|Ax-y\|^2 + \lambda R(z) \quad \text{s.t. } x = z$$

This corresponds to the augmented Lagrangian (penalty method):

$$L_\mu(x,z) = \frac{1}{2}\ \|Ax-y\|^2 + \lambda R(z) + \frac{\mu}{2}\ \|x-z\|^2$$

HQS Steps:
- $x_k = \arg\min_x \frac{1}{2}\ \|Ax-y\|^2 + \frac{\mu}{2}\ \|x-z_{k-1}\|^2 = \text{prox}_{\frac{1}{\mu}\|Ax-y\|^2}(z_{k-1})$
- $z_k = \arg\min_z \frac{\mu}{2}\ \|x_k - z\|^2 + \lambda R(z) = \text{prox}_{\frac{\lambda}{\mu}R}(x_k)$

Key Idea: (z_k) is an image denoising problem. We can replace this optimization sub-problem (the proximal operator) with a pre-trained deep image denoiser.

Pros (+): We only need to train a denoiser once.
Cons (-): Have to tune hyperparameters (e.g., μ) during inference.

***Algorithm: Plug-and-play image restoration with deep denoiser prior* (*DPIR*)**

Input: Deep denoiser prior model, denablaed image y, denablaation operation A, image noise level σ, σ_k of denoiser prior model at k-th iteration for a total of K iterations, trade-off parameter λ.
Output: Restored image z_K.

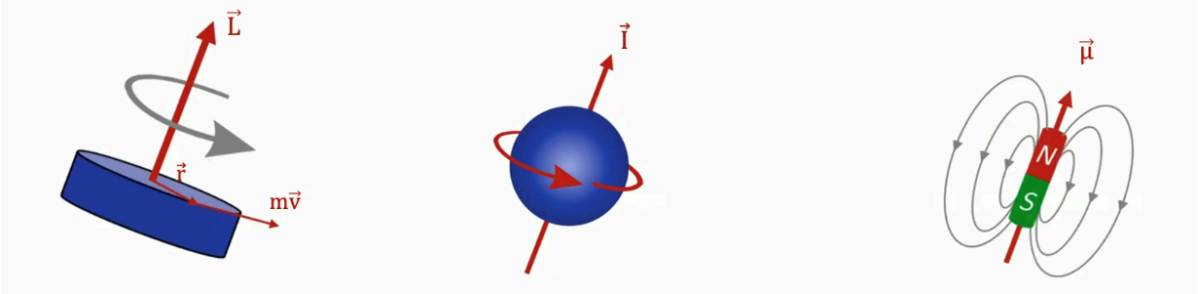
1. Initialize z_0 from y, pre-calculate $\alpha_k > \lambda \frac{\sigma^2}{\sigma_k^2}$.
2. **for** $k = 1, 2, ..., K$ **do**
3. $\quad x_k = \arg\min_x \|y - A(x)\|^2 + \alpha_k \, \|x - z_{\{k-1\}}\|^2$
4. $\quad z_k = \text{Denoiser}(x_k, \sigma_k)$
5. **end**

Magnetic Resonance Imaging

From Spin to Magnetic Resonance Imaging

The study of MRI often begins from a classical physics viewpoint, where we accept the existence of nuclear spin without diving into the full quantum mechanics motivation.

Nuclear Spin, Magnetic Dipole Moment and Torque



A rotating object with mass m leads to angular momentum: $\vec{L} = \vec{r} \times (m\vec{v})$. The spin of a proton leads to magnetic angular momentum $\vec{I}$. It is modeled as a magnetic dipole with a moment $\vec{\mu} = \gamma \vec{I}$, where γ is the gyromagnetic ratio.

Gyromagnetic Ratios

Element	Gyromagnetic ratio γ [MHz/T]
^{1}H	42.58
^{3} He	32.43
^{23} Na	11.26
^{31}P	17.24

Exposure to an external magnetic field $\vec{B}_0$ leads to a torque $\vec{\tau}$ that attempts to align the magnetic moment $\vec{\mu}$:

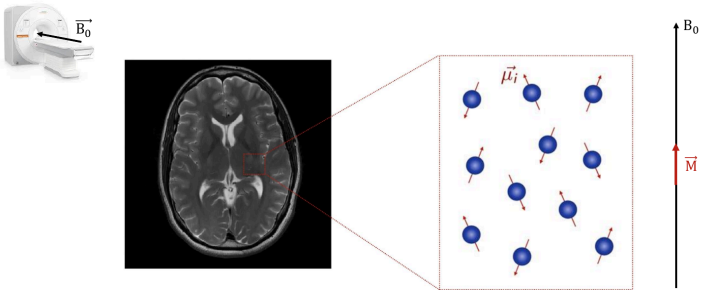
$$\vec{\tau} = \vec{\mu} \times \vec{B}_0$$

In the absence of a field, thermal motion leads to random orientation of the magnetic moments which means no overall magnetization (humans are not inherently magnetic). In the presence of a field

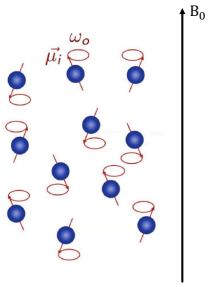
B_0, thermal motion is still present, but the magnetic moments align enough to create a small bulk magnetization $\vec{M} = \sum_i \vec{\mu}_i$ with magnitude

$$M = \frac{\rho \gamma^2 \hbar B_0}{4kT}$$

This alignment is the first effect we will later use for MRI.



Another phenonom is precession: The magnetic momentum precesses around the external field. They are not fully aligned with the magnetic field B_0.



Definition 16 (Larmor Frequency) . The frequency of precession is the Larmor frequency:

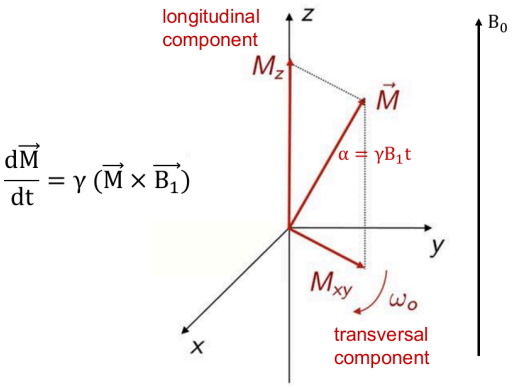
$$\omega_0 = \gamma B_0$$

For a proton (^1H), $\frac{\gamma}{2\pi} \approx 42.6 \frac{\text{MHz}}{T}$. This is a key equation for MR imaging.

Interaction with Radiofrequency field B_1

When an Radiofrequency field B_1 is applied at the Larmor frequency, it tips the magnetization away from the longitudinal axis. The B_1 field is normal to the B_0 field. The resulting change of the magnetization M can then be described by

$$\frac{dM}{dt} = \gamma(M \times B_1)$$



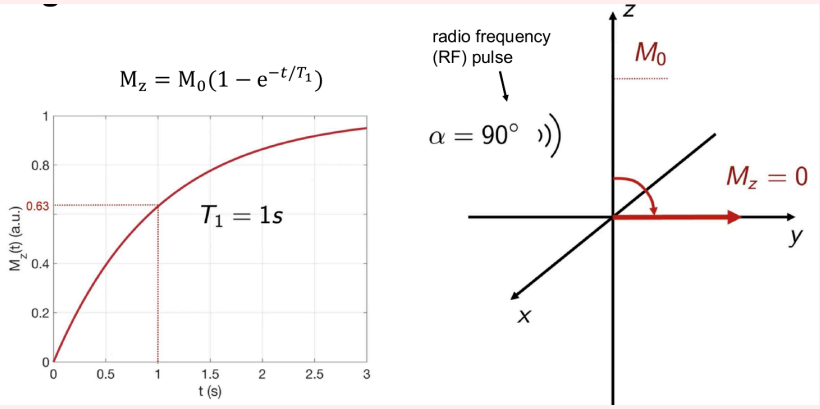
With that setting we can now turn of the B_1 so that the particles can relax again. All particles are now still spinning with the precession from the B_1 field in a sychronized manner for a small amount of time. During that time they induce a current in the coil that produced the B_1 field due to the changing magnetic field from them which is called magnetic flux. This current in the B_1 coil is the only thing we can measure in MRI.

$$u = -\frac{\partial \psi}{\partial t}$$

In MRI we get now different images from one scan, that are the different contrasts:

Definition 17 (Longitudinal Relaxation (T1)) . When we start again with just a B_0 field, then we can switch on the B_1 field and we make it so strong, that we have a flip angle of 90 degrees. So we just have XY magnetization. Then we turn off the B_1 again and measure the time how long it take until the magnetization relaxes again in the B_0 direction. This is what this formula describes and the curve that we can see in the image:

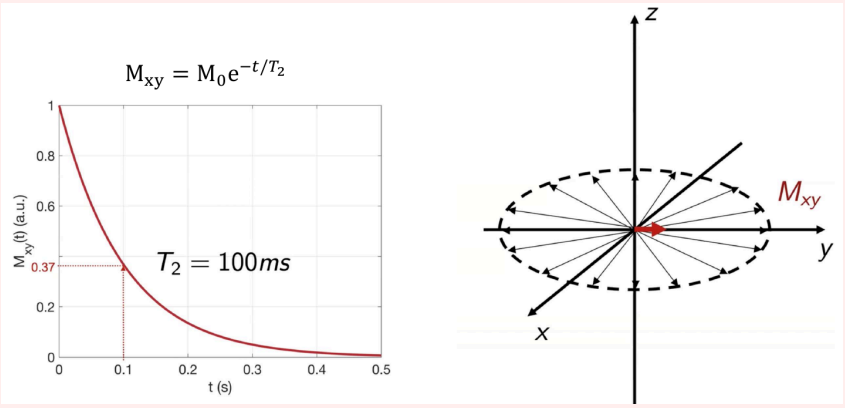
$$M_z = M_0\left(1 - e^{-\frac{t}{T_1}}\right)$$



We can then just measure the magnitude of M_z which the particles reached again after the time T_1.

Definition 18 (Transversal Relaxation (T2)) . We start with a Transversal B_1 field again and have this time the B_0 field turned off. Then we can turn the B_1 field off and measure the behaviour of the magnetic field which follows this line:

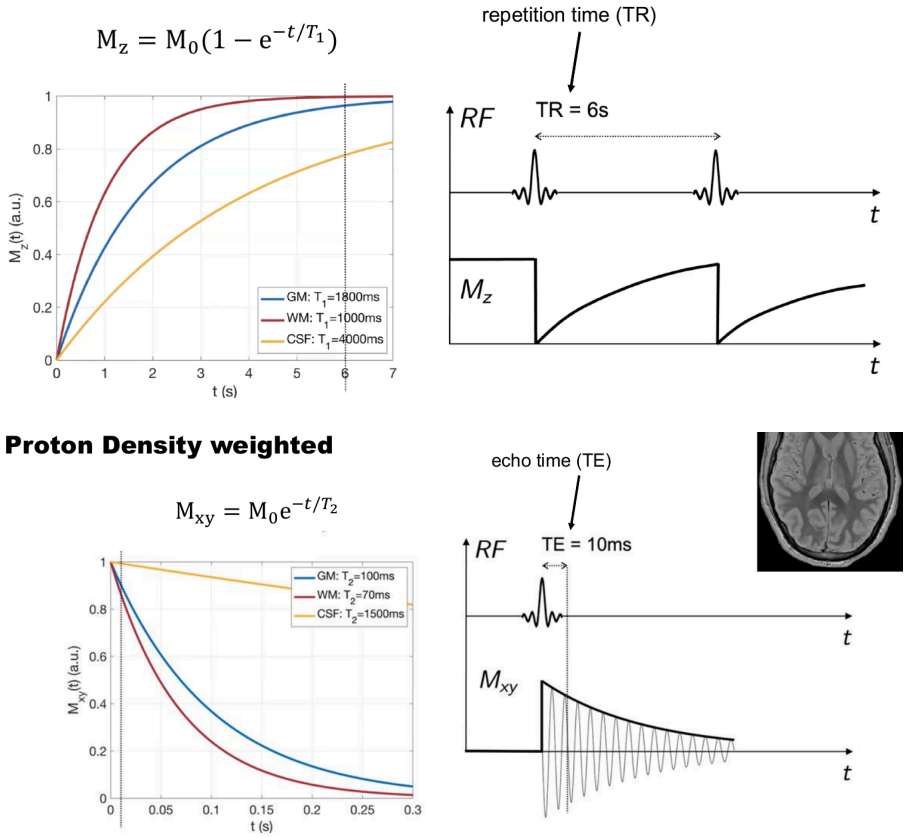
$$M_{xy} = M_0 e^{-\frac{t}{T_2}}$$



The magnetization will fall of exponentially and thermal motion will dominate again in the end. Here we can measure again how fast that happens.

Contrast Information

Different tissue types now have different T_1 and T_2 times. Again the reminder, we don't measure the longitudinal M_0 component, we measure only the transversal component.



By tailoring the Repetition Time (TR) and Echo Time (TE), we can choose the most suitable contrast to differentiate structures:

- T1-weighted: Short TR, short TE.
- T2-weighted: Long TR, long TE ($M_{xy} = M_0 e^{-\frac{t}{T_2}}$).
- Proton Density (PD) weighted: Long TR, short TE ($M_z = M_0\left(1 - e^{-\frac{t}{T_1}}\right)$).

Proton Density means that the tissues with the highest concentration or highest amount of hydrogen atoms appear the brightest. With that configuration, we counter the effects of the T1 weighting and the T2 weighting and measure basically when nothing yet had time to relax. That gives away so to say the densities of all tissues no matter their T1 or T2 time.

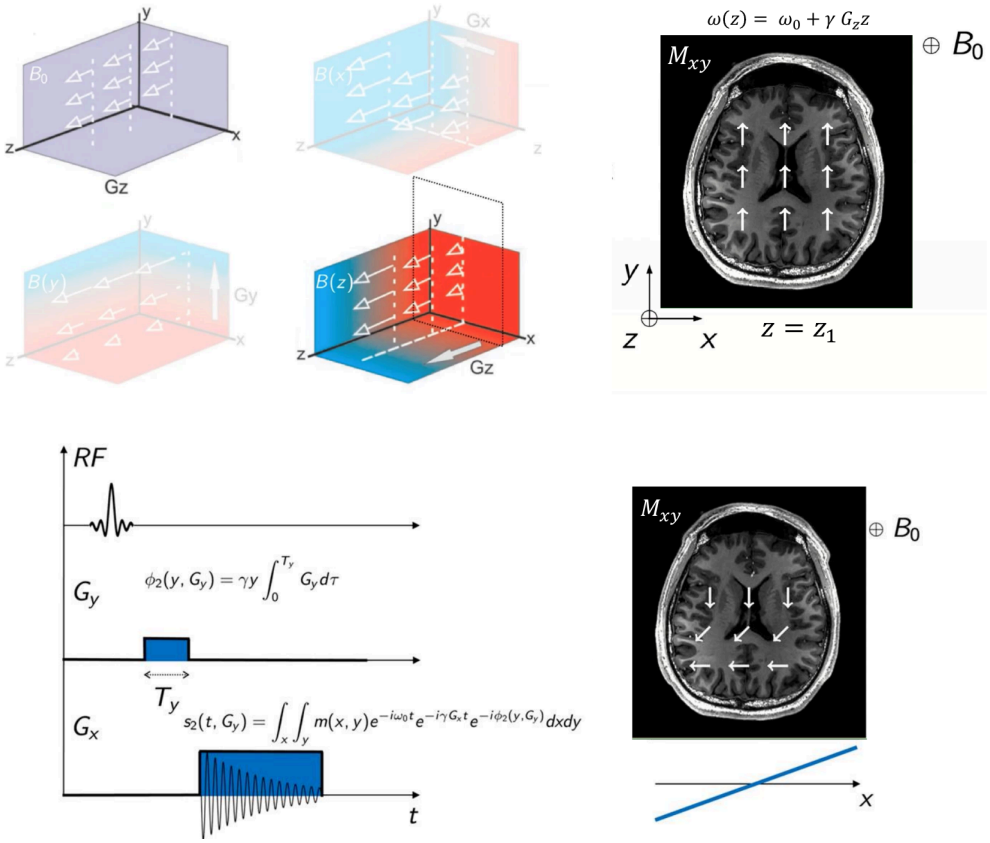
How to get now spatial information?

To get an image, spatial information must be encoded using gradient fields $\vec{G}$. For that we need 3 more coils that introduce that gradient field. The local Larmor frequency becomes position-dependent, the particles rotate at different locations with different frequencies.

Our larmor frequency formula $\omega_0 = \gamma B_0$ becomes:

- X-gradient: $\omega(x) = \omega_0 + \gamma G_x x$
- Y-gradient: $\omega(y) = \omega_0 + \gamma G_y y$
- Z-gradient: $\omega(z) = \omega_0 + \gamma G_z z$

This allows for slice selection (z-axis) and frequency/phase encoding (x and y axes) to fill the k-space (space of Fourier coefficients), which is then transformed into an image via a 2D Fourier Transform.

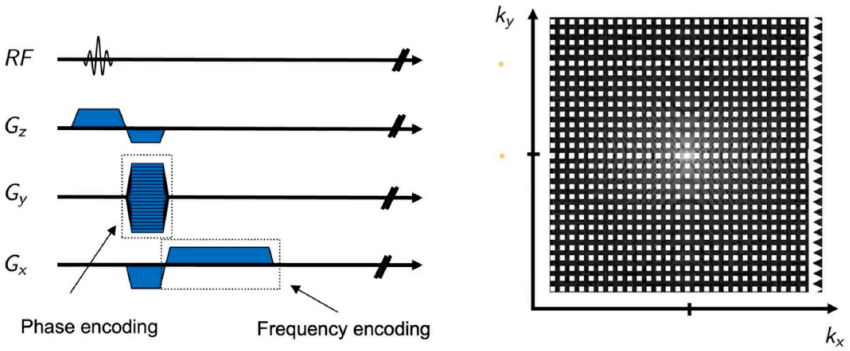


So with a MRI device we are measuring Fourier coefficients. The formula for that is

$$s(k_x, k_y) = \int_x \int_y m(x,y) e^{-ik_x x} e^{-ik_y y} dxdy$$

where $m(x,y)$ is the magnetization and $s(k_x, k_y)$ are our coefficients in the Fourier space.

So in the end we enable the G_z plane and then we choose a frequency on this G_z plane. Then we send a Radiofrequency pulse with the selected frequency so that only the particles on this specific transversal plane are tipped over. Then we choose a G_y which encodes a different phase for every single y coordinate. Afterwards we apply a neagtive G_x to move the magnetization there to the left negative k-space and then we turn the G_x positive again so that it moves to the other direction and hereby we sample the values for the k-space. For every point you have to wait for around 6 seconds and repeat the process again and again. And that is why we dont want to measure the whole Fourier Space, but rather measure only selected points and interpolate or do some postprocessing.



$$s(k_x, k_y) = \int_x \int_y m(x,y) e^{-ik_x x} e^{-ik_y y} dxdy$$

$$\phi(x,t) = \gamma x \int_0^t G_x d\tau \equiv k_x x$$

$$\phi(y, G_y) = \gamma y \int_0^{T_y} G_y d\tau \equiv k_y y$$

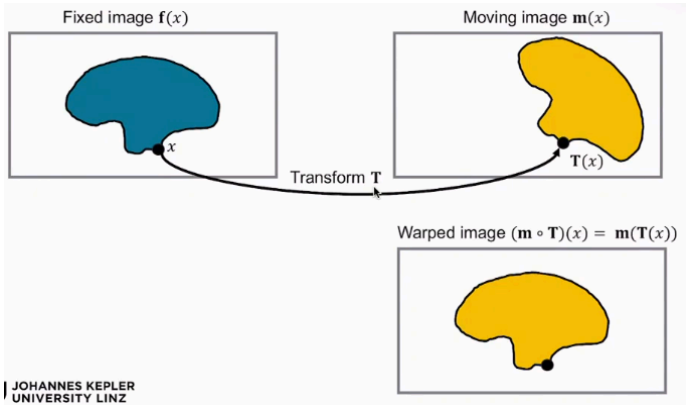
And due to the Fourier Transform, MRI is a linear inverse problem.

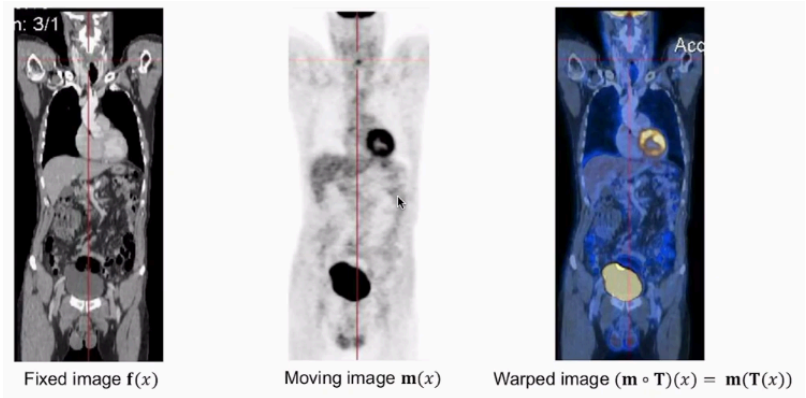
Image Registration

What is Image Registration?

Image registration is the process of transforming different sets of data into one coordinate system.

Definition 19 (Fundamental Components) .
- **Fixed image** $f(x)$: The reference image that remains stationary.
- **Moving image** $m(x)$: The image that is deformed to match the fixed image.
- **Transformation** T: A mapping $T : x \to T(x)$ that defines how the moving image is warped.
- **Warped image**: The result of applying the transformation to the moving image, denoted as $(m \circ T)(x) = m(T(x))$.





Variational Approach to Registration

Registration is typically formulated as an optimization problem where we seek the optimal transformation parameters θ:

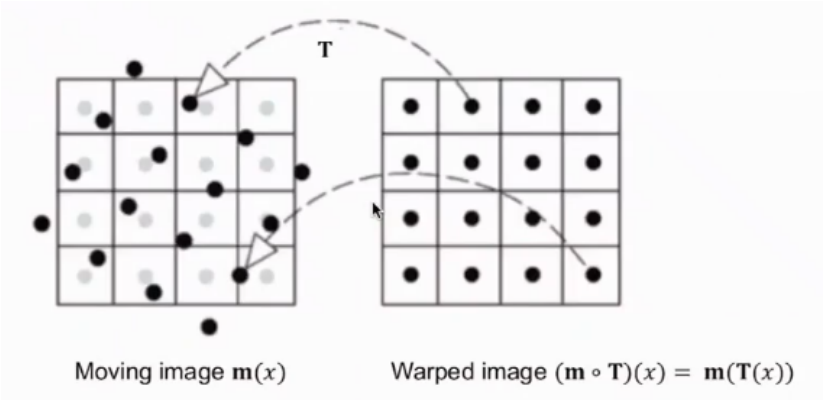
Definition 20 (Variational Formulation) .

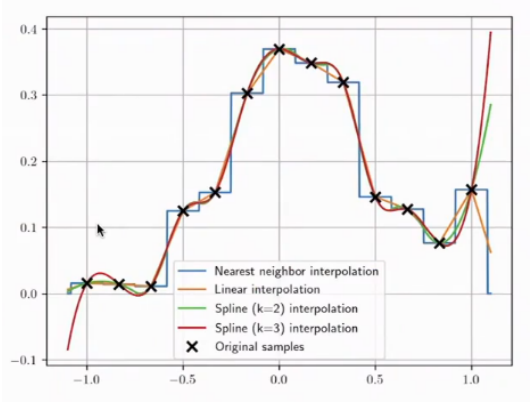
$$\min_{\theta} S(f, m \circ T_\theta) + R(T_\theta)$$

where

- $S(f, m \circ T_\theta)$ is the Similarity Metric which measures how well the fixed image f matches the wraped image $m \circ T_\theta$.
- $R(T_\theta)$ is the Regularization term (ensures the transformation is physically plausible or smooth and not completely overfits local noise).

After we applyed the transform to the moved image, we need to wrap the image on the underying pixelgrid. There are different interpolation strategies for that:





To choose a constant gradient is not a good idea as the function is not continuous and the gradient is 0. Most common is linear or spline interpolation, but spline interpolation is not currently implemented in pytorch.

Interpolation by B-splines

$$\hat{m}(x) = \sum_{k=1}^{N_k} \theta_k \beta^n \left(\frac{x - \mu_k}{\sigma} \right)$$

- $\hat{m}(x)$: The interpolated value at position x. In image processing, this is often the estimated intensity of a pixel at a non-integer coordinate.
- N_k: The total number of basis functions (or control points) being used.

- θ_k: The interpolation coefficients (weights). These are the values we solve for to ensure the spline fits our data points.
- β^n: The B-spline basis function of degree n. This determines the "shape" of the influence each control point has.
- x: The specific location where you want to calculate the value.
- μ_k: The center of the k^{th} basis function. This is typically the pixel center.
- σ: The spacing or scale between the basis functions. This controls how "stretched" or compressed the B-spline is.

For the 0-th order it looks like this: It is 1 inside a small window and 0 everywhere else. It represents the simplest form of interpolation where you just take the value of the closest pixel.

$$\beta^0(x) = \begin{cases} 1 \text{ if } |x| < \frac{1}{2} \\ \frac{1}{2} \text{ if } |x| = \frac{1}{2} \\ 0 \text{ else} \end{cases}$$

$$\beta^n(x) = \left[\underbrace{\beta^0 * \beta^0 * \ldots * \beta^0}_{(n+1) \text{ times}} \right](x)$$

- $*$: This symbol denotes convolution.
- n: The degree of the spline.

The Logic: A B-spline of degree n is created by convolving the box function (β^0) with itself n times.

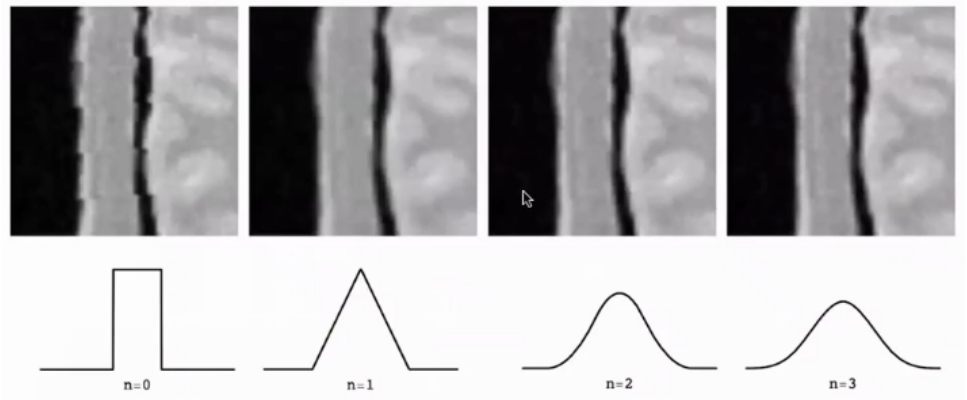
Examples:
- β^1 (linear) results in a triangle shape.
- β^3 (cubic) results in a smooth, bell-like curve.

As n increases, the function becomes smoother and converges toward a Gaussian distribution.

The regularizer helps, because when there are bumps, we dont want to fit them perfectly. This makes sense it we have a high resolution image with some noise.

Thin plate splines:

$$\min_{\theta} \frac{1}{2} \|\hat{m}_\theta(x) - m(x)\|^2_{\Omega_M} + \alpha \int_{\Omega_M} |\nabla^2 \hat{m}(x)|^2 \, dx$$



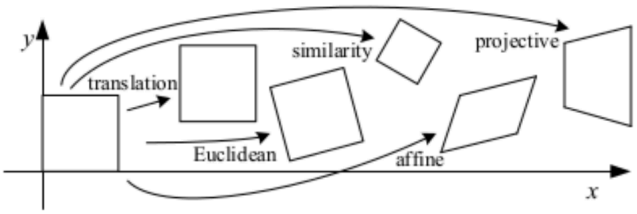
TRANSFORMATION MODELS

Global Linear Transformation Models:

- **Rigid**: Rotation and translation (6 degrees of freedom in 3D).
- **Affine**: Includes scaling and shearing.

$$T(x) = Ax + b$$

$$\begin{aligned} \text{translation} : T(x) &= x + b \\ \text{rotation} : T(x) &= Rx + 0 \\ \text{rigid / Euclidean} : T(x) &= Rx + b \\ \text{affine} : T(x) &= Ax + b \end{aligned}$$



Non-linear Transformation Models:
- Allows for local deformations (e.g., organ movement, breathing).
- Often parameterized by B-Splines or displacement fields.

Similarity Metrics

The choice of similarity metric depends on whether the images are from the same modality (intra-modal, e.g. fit 2 MRI scans that were taken at 2 different time points) or different modalities (inter-modal, e.g. fit a MRI scan and a CT scan together).

- Sum of Squared Differences (SSD): Best for intra-modal images with linear intensity relationships.

$$S_{\text{SSD}}(m \circ T, f) = \int (f(x) - m(T(x)))^2 dx$$

- Normalized Cross Correlation (NCC): Robust to linear intensity changes. Cross Correlation gives high values when both vectors point in similar directions and negative value when they point into the opposite direction.

$$S(m \circ T, f) = -\frac{\int_{\Omega_f} \big(f(x) - \mu_f\big)(m(T(x)) - \mu_m) dx}{\sqrt{\int_{\Omega_f} \big(f(x) - \mu_f\big)^2 dx} \cdot \sqrt{\int_{\Omega_f} \big(m(T(x)) - \mu_m\big)^2 dx}}$$

  So for different modalities it is good that it utilizes correlation instead of a perfect fit, but it is a bit more complicated to compute.
- Normalized Gradient Field (NGF): Matches the edges/gradients of the images.

$$S(m \circ T, f) = \int_{\Omega_f} 1 - \left| \frac{\nabla f(x)^\top}{\|\nabla f(x)\|_\eta^2} \cdot \frac{\nabla m(T(x))}{\|\nabla m(T(x))\|_\eta^2} \right| dx$$

$$\|x\|_\eta^2 = \eta^2 + \sum_i x_i^2$$

The advantage is that it focuses on edges not intensities and not directions. Multiple modalities are supported.

- Mutual Information (MI): The standard for multi-modal registration (e.g., MR to CT). It measures the statistical dependence between image intensities.

$$S(m \circ T, f) = D_{\mathrm{KL}}\left(p_{m,f} \parallel \underbrace{p_m \otimes p_f}_{\text{outer product}} \right)$$

where p_f is the histogram of $f(x)$, p_m is the histogram of $m(T(x))$ and $p_{m,f}$ is the joint histogram of f and $m(T(x))$

$$= - \sum_{\hat{m} \in B_M} \sum_{\hat{f} \in B_F} p_{m,f}\left(\hat{m}, \hat{f}\right) \cdot \log\left(\frac{p_{m,f}\left(\hat{m}, \hat{f}\right)}{p_m(\hat{m}) \cdot p_f\left(\hat{f}\right)} \right)$$

Here B_M and B_F denote the bins of the histograms of m and f, respectively. The advantages are that they are suited for multiple modalities and they are very powerful (as they are most general). Their disadvantages are that they are very non-convex and have many hyperparameters.

REGULARIZATION

Regularization prevents "unrealistic" warping, such as folding the image onto itself.

- Implicit regularization: Built into the model architecture or transformation model (e.g., low-resolution B-spline grid or only rigid transform).
- Explicit regularization: A penalty term added to the loss function (e.g., Diffusion, Elastic, or Total Variation regularizers).

Diffusion regularization:

$$R(\theta) = \int_{\Omega_m} |\nabla T_\theta(x)|^2 \, dx = \|\nabla T\|_{\Omega_m}^2$$

It is easy to compute but there are smooth edges in the transform T as hard edges which mean big gradients are penalized.

Bending energy:

$$R(\theta) = \int_{\Omega_m} \|\nabla^2 T_\theta(x)\|_F^2 \, dx$$

It is easy to compute and it penalizes the curvature.

Jacobian regularization:

$$R(\theta) = \int_{\Omega_m} |1 - \det \nabla T_\theta(x)| \, dx$$

It is more complex to compute and it penalizes local area changes.

Optimization and Deep Learning

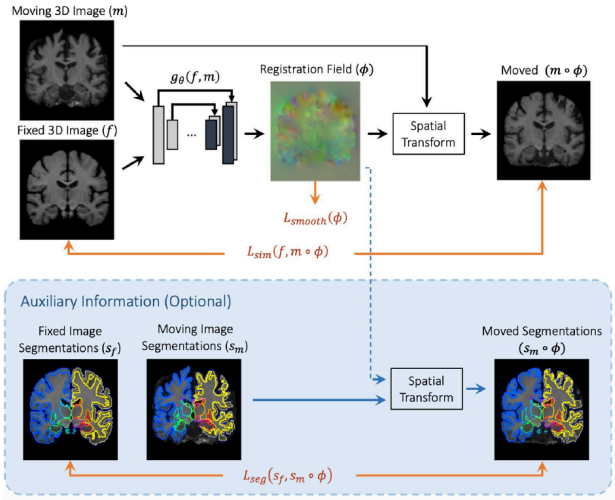
Optimization Tricks

- Coarse-to-fine strategy: Start by registering downsampled (low-res) versions of the images and gradually increase resolution to avoid local minima.
- Sequential complexity: Start with rigid/affine transforms before moving to non-linear deformations.

Deep Learning Approaches

Deep learning has shifted registration from iterative optimization to "one-shot" prediction.

- VoxelMorph: A Unet-based framework that learns to predict the displacement field between two images in a single forward pass.



- Implicit Neural Representations (INR): Representing the transformation as a continuous function $T(x)$ parameterized by a neural network (e.g., using periodic activation functions like SIREN).

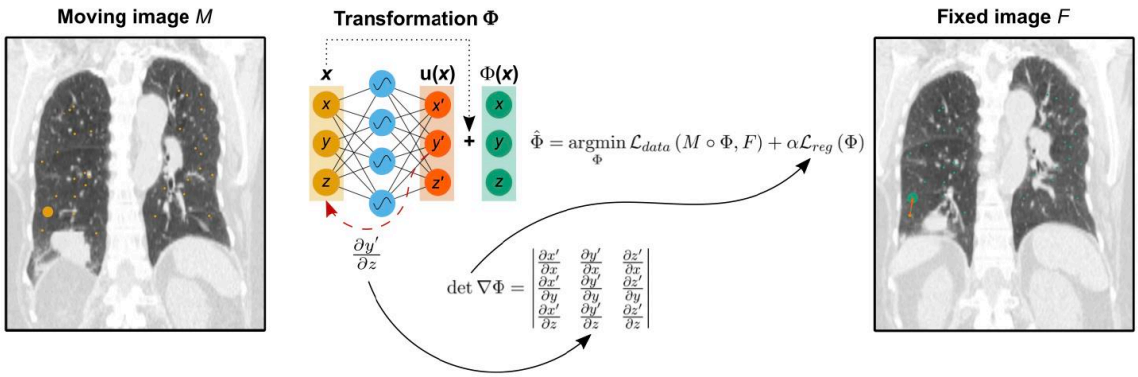


Image Segmentation

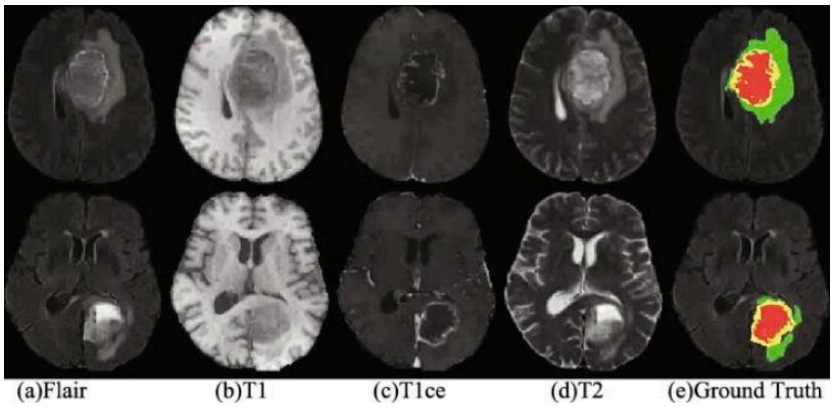
What is Image Segmentation?

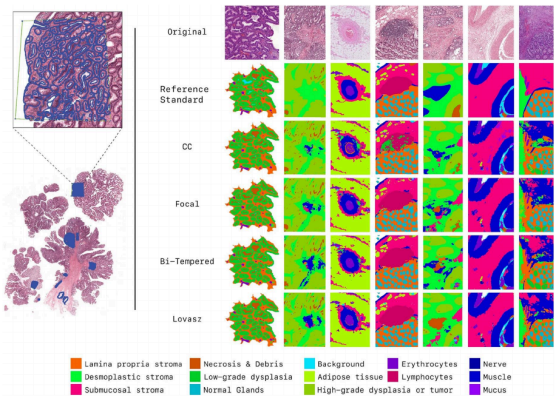
Image segmentation is the process of partitioning a digital image into multiple segments (sets of pixels, also known as image objects).

- Goal: Assign a semantic label to every pixel (2D) or voxel (3D) in an image.
- Output: A dense label map.

Example 13 — Medical Applications .

- Tumor delineation: Identifying the boundaries of a tumor in MRI or CT scans.
- Organ segmentation: Separating liver, lungs, or heart from surrounding tissue.
- Lesion burden estimation: Quantifying the total area or volume affected by a disease.

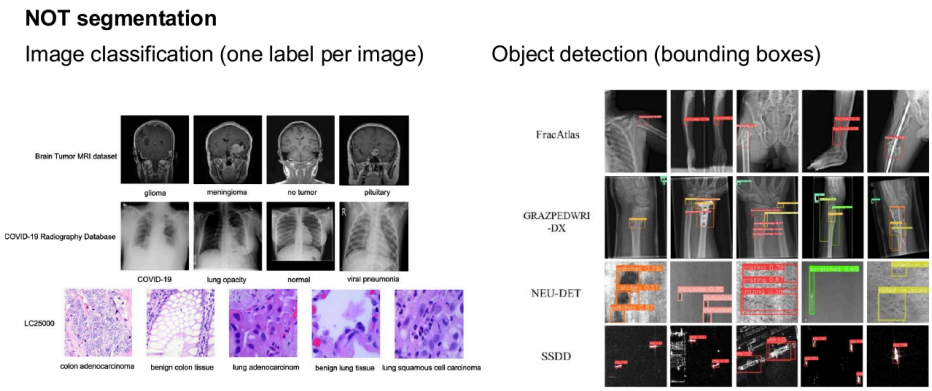




Segmentation vs. Other Tasks

It is important to distinguish segmentation from other computer vision tasks:

- Image Classification: Assigning a single label to the entire image (e.g., "glioma" vs "no tumor").
- Object Detection: Identifying objects and drawing bounding boxes around them.
- Segmentation: Identifying the exact shape of the object at the pixel level.



Clinical Significance

- Quantitative Analysis: Allows for precise measurement of volume, shape, and thickness.
- Treatment Planning: Essential for radiotherapy (delineating the target volume) and surgery.
- Longitudinal Follow-up: Comparing scans over time to check for progression or stability using criteria like RANO.
- Sensitivity: In medicine, small boundary errors can have a massive clinical impact.

Mathematical Formulation

Definition 21 (Segmentation as a Labeling Function) .

Let $\Omega \subset \mathbb{R}^d$ be the image domain ($d \in \{2, 3, ...\}$). An image x is a function $x : \Omega \to \mathbb{R}^C$ (color space). The segmentation problem is defined as finding a mapping:

$$S : \Omega \to \mathcal{L}$$

that assigns to every location a label and where $\mathcal{L} = \{0, 1, ..., K\}$ is the set of labels (0 is usually the background).

The labeling induces a partitioning of the domain into $\Omega_k = \{i \in \Omega \mid S(i) = k\}$. These partitions must be:

- Non-overlapping: $\Omega_k \cap \Omega_l = \emptyset$ for $k, l \in \mathcal{L} \wedge k \neq l$.
- Complete: $\bigcup_{l \in \mathcal{L}} \Omega_l = \Omega$.

That basically translates to "I assign every pixel a label and only one".

Types of Segmentation

- Binary: Separating a single foreground structure from the background ($K = 1$).
- Multi-class: Segmenting multiple anatomical structures ($K > 1$).
- Semantic: All pixels of the same class (e.g., all cells) share one label.

- Instance: Separating individual objects (e.g., each individual cell gets a unique ID).
- Panoptic: Combines semantic and instance segmentation.

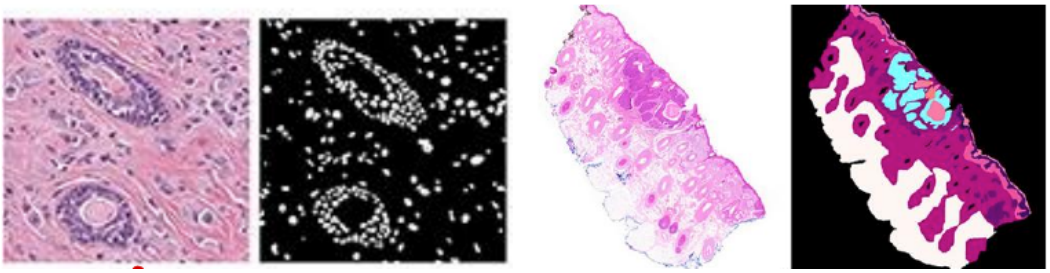
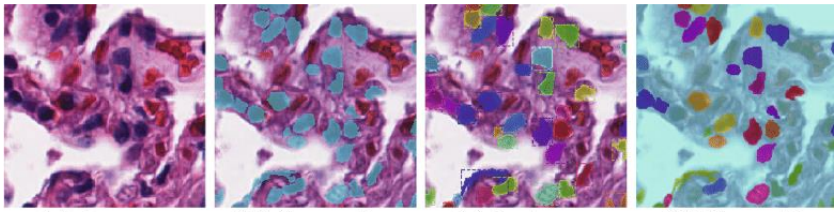


Figure 50 : Binary vs Multi-class segmentation



(a) Image (b) Semantic Segmentation (c) Instance Segmentation (d) Panoptic Segmentation

CLASSICAL SEGMENTATION METHODS

1. Thresholding:
   - Global: One value for the entire image (e.g., Otsu's method).
   - Local: Adaptive thresholds based on local neighborhoods.
2. Region-based:
   - Region Growing: Starts with seed points and expands to similar neighbors.
   - Watershed: Interprets the image as a topographic map and "floods" it from local minima.
3. Graph Cuts:
   - Represents the image as a graph where pixels are nodes.
   - Minimizes an energy function $E(x)$ consisting of unary (likelihood) and pairwise (smoothness) costs.
   - Solved using min-cut/max-flow algorithms.

Graph Cuts

Let the image be represented as a graph $G(V, E)$. Assume a binary segmentation task, i.e., $L = \{0, 1\}$.

A graph cut is the set of edges whose removal makes a graph disconnected. The cost of the cut is the sum of its edge weights. The energy $E(x)$ is defined as:

$$E(x) = \underbrace{\sum_{p \in V} x_p F_p + (1 - x_p) B_p}_{\text{unary term}} + \underbrace{\lambda \sum_{p,q \in E} W_{pq} \, |x_p - x_q|}_{\text{pairwise term}}$$

Where:

- $x_p = \begin{cases} 0 \text{ if background at pixel } p \\ 1 \text{ if foreground at pixel } p \end{cases}$
- F_p: Cost of assigning foreground
- B_p: Cost of assigning background
- W_{pq}: Similarity of p and q (typically $e^{-\alpha \, \|I_p - I_q\|^2}$)

- λ: Regularization weight

Algorithm Steps:
1. Build graph G with one node per pixel.
2. Add source/sink edges for unary costs (back- and foreground).
3. Add neighborhood edges with weights $W_{\{pq\}}$ (reflecting similarity of nodes p and q).
4. Compute minimum $s-t$ cut (Boykov-Kolmogorov Algorithm).
5. Assign labels from cut.

Relation to Discrete TV

Note that the pairwise term can be written as:

$$\sum_{p,q\in E} W_{pq}\ |x_p - x_q| = \|\nabla x\|_W$$

which is the discrete anisotropic TV.

Then, we can also simplify the unary term:

$$\sum_{p\in V} x_p F_p + (1-x_p)B_p = \sum_{p\in V} B_p + \sum_{p\in V} x_p(F_p - B_p) = \langle 1, B\rangle + \langle x, F-B\rangle$$

We get the following minimization problem:

$$\min_{x\in\{0,1\}} \langle x, F-B\rangle + \lambda\ \|\nabla x\|_W$$

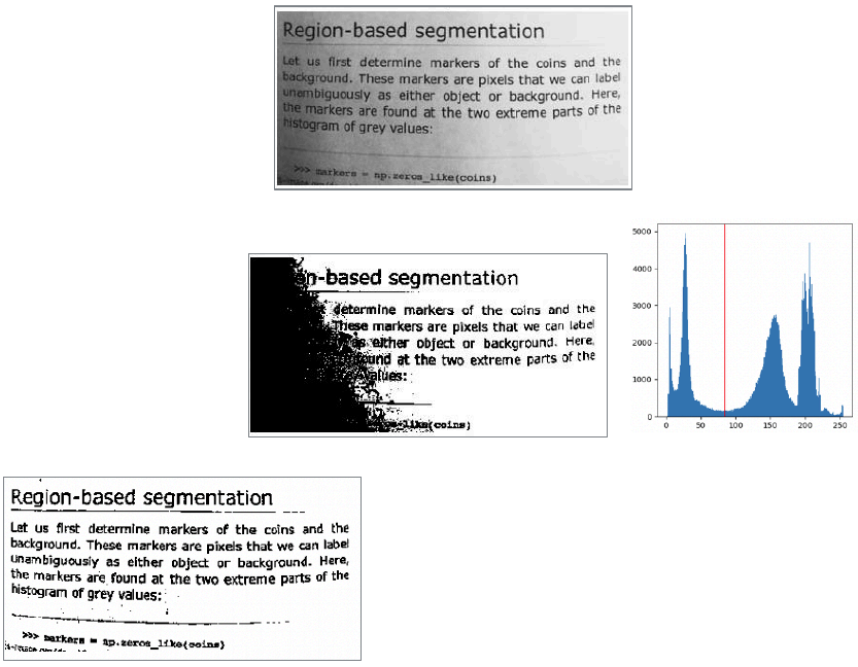


Figure 52 : Global Thresholding vs Local Thresholding

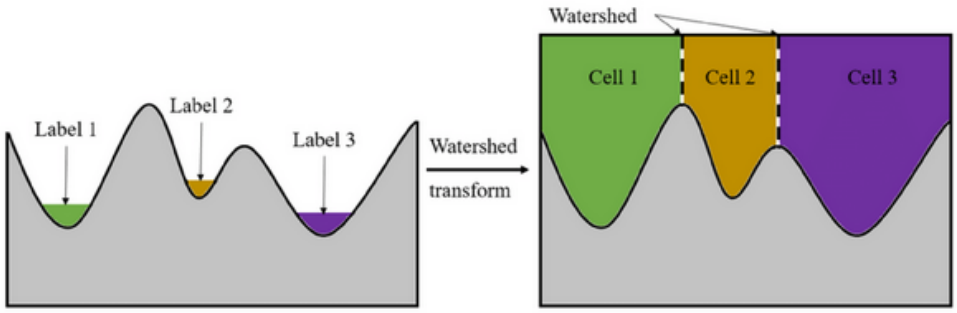


Figure 53 : Watershed Algorythm

DEEP LEARNING FOR SEGMENTATION

U-Net

The U-Net is the gold standard for medical image segmentation.

- Architecture: Symmetric encoder (contracting path) and decoder (expansive path).
- Skip Connections: Concatenate high-resolution features from the encoder to the decoder to preserve spatial detail.
- The Receptive Field gets exponentially bigger with each deeper layer of the encoder. This is important as the Receptive Field should be at least as large as the things we want to segment.
- Training Objective: Usually Weighted Cross Entropy (CE).

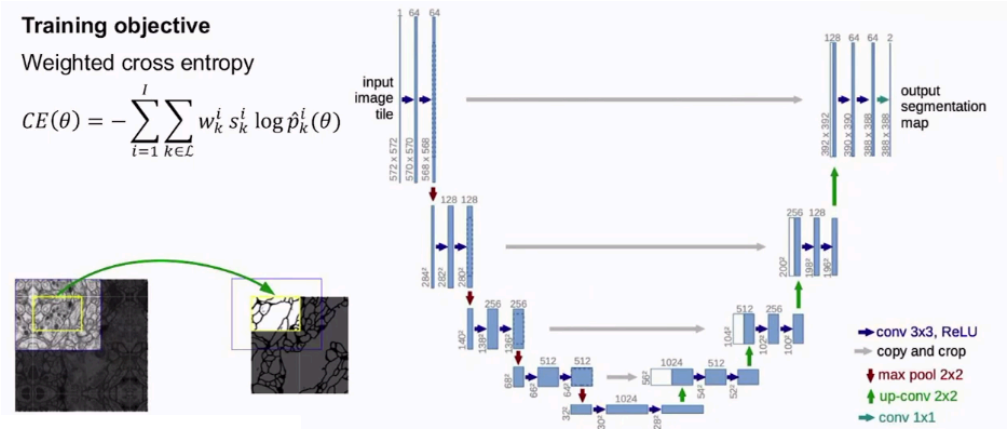


Figure 54 : U-Net

V-Net

Designed for volumetric (3D) medical images.

- Objective: Uses the Dice Loss to handle class imbalance (e.g., when the tumor is much smaller than the background).

$$D(\theta) = 1 - \frac{2\sum_{i=1}^{I} \hat{p}_i(\theta) s_i}{\sum_{i=1}^{I} \hat{p}_i(\theta) + \sum_{i=1}^{I} s_i} = 1 - \underbrace{2\frac{\langle \hat{p}(\theta), s\rangle}{\langle \hat{p}(\theta), 1\rangle + \langle s, 1\rangle}}_{\text{Dice Coefficient}}$$

where s is the ground truth and $\hat{p}(\theta)$ is the probability that a voxel is part o that ground truth class.

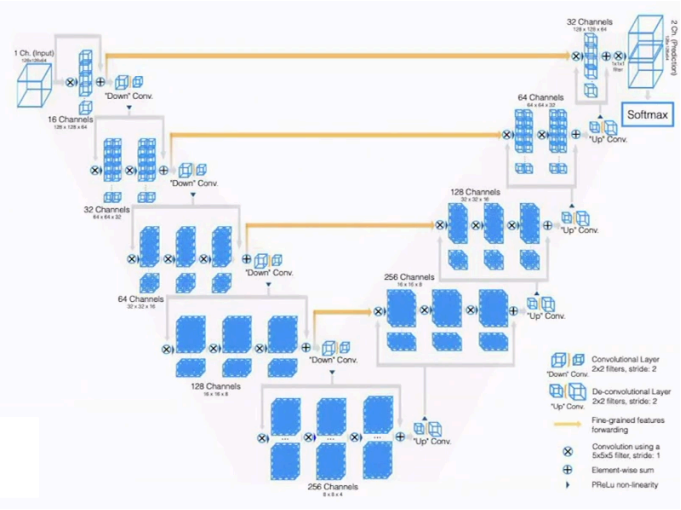


Figure 55 : V-Net

Comparison of Cross Entropy & Dice Loss

$$\mathrm{CE} = -\sum_i w_i [s_i \log \hat{p}_i + (1 - s_i) \log(1 - \hat{p}_i)]$$

$$D = 1 - 2\frac{\langle \hat{p}(\theta), s \rangle}{\langle \hat{p}(\theta), 1 \rangle + \langle s, 1 \rangle} = 1 - 2\frac{A}{B}$$

Derivatives

$$\begin{aligned}
\frac{\partial\ \mathrm{CE}}{\partial \theta} &= -\sum_i w_i \left[\frac{\partial \hat{p}_i}{\partial \theta} \frac{s_i}{\hat{p}_i} - \frac{\partial \hat{p}_i}{\partial \theta} \frac{1 - s_i}{1 - \hat{p}_i} \right] \\
&= -\sum_i w_i \frac{\partial \hat{p}_i}{\partial \theta} \frac{1}{\hat{p}_i(1 - \hat{p}_i)} [(1 - \hat{p}_i) s_i - \hat{p}_i (1 - s_i)] \\
&= -\sum_i w_i \frac{\partial \hat{p}_i}{\partial \theta} \frac{1}{\hat{p}_i(1 - \hat{p}_i)} [s_i - \cancel{\hat{p}_i s_i} - \hat{p}_i + \cancel{\hat{p}_i s_i}] \\
&= -\sum_i w_i \frac{\partial \hat{p}_i}{\partial \theta} \frac{1}{\hat{p}_i(1 - \hat{p}_i)} [s_i - \hat{p}_i]
\end{aligned}$$

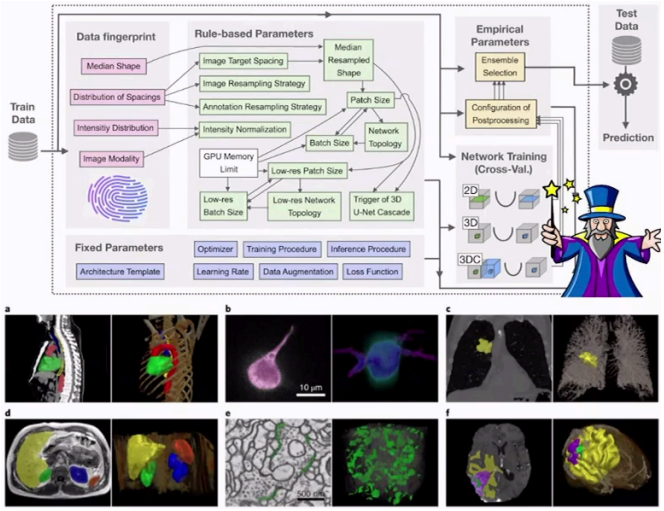
$$\begin{aligned}
\frac{\partial D}{\partial \theta} &= 0 - \frac{\partial \hat{p}_i}{\partial \theta} 2 \frac{S \cdot B - A \cdot \mathbf{1}}{B^2} \\
&= -\frac{\partial \hat{p}_i}{\partial \theta} \cdot 2 \frac{sB - A \cdot \mathbf{1}}{B^2}
\end{aligned}$$

When we compare the gradients:

- local vs global information: Due to the A in the formula, this gradient has global information in it, because A is the correlation between the prediction and the target.
- stable & no vanishing gradient for CE (in logits) as the term $\hat{p}_i(1 - \hat{p}_i))$ cancels (the gradient from the sigmoid with respect to the logits is exactly this term). This does not hold for the Dice
- CE has a strong signal initially during training. Dice works better when the images are already almost aligned.

Advanced Architectures

nnU-Net: A "self-configuring" method that automatically adapts the U-Net architecture and hyper-parameters to a specific dataset. It is thought as a Out of the box experience.



UNETR: Uses Transformers as the encoder to capture long-range dependencies, paired with a U-shaped decoder.

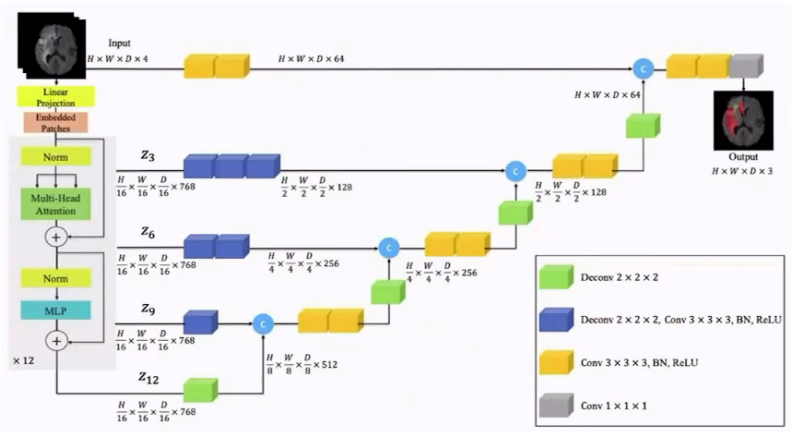
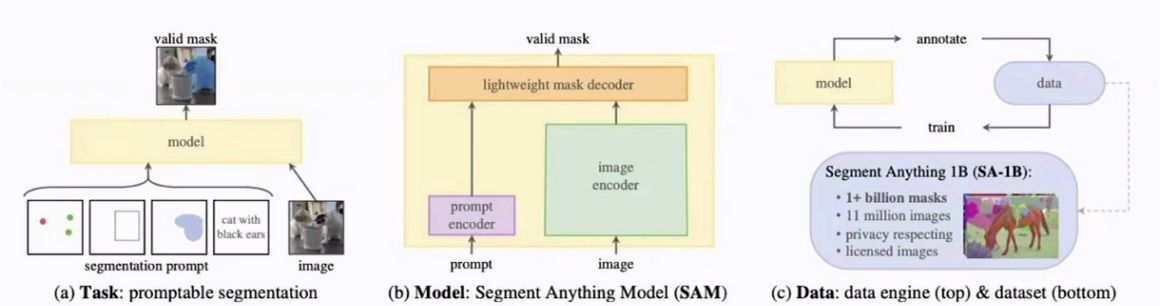


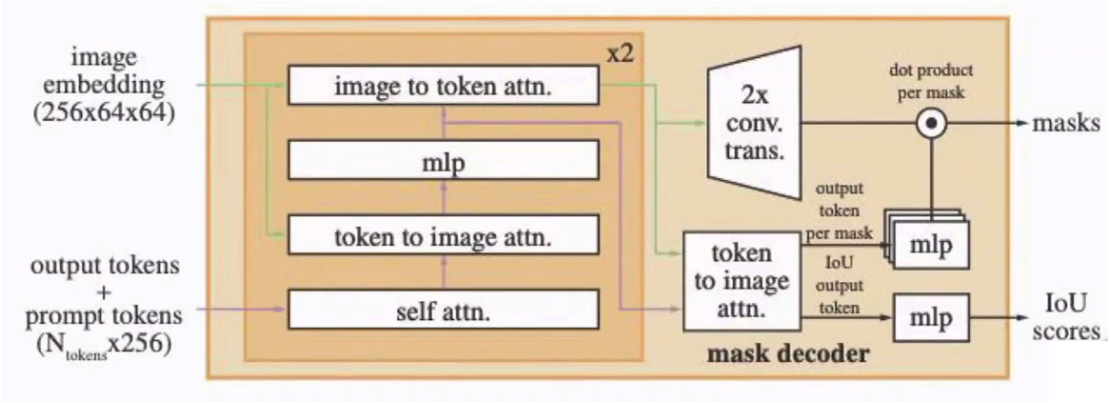
Figure 2. Overview of UNETR architecture. A 3D input volume (e.g. $C = 4$ channels for MRI images), is divided into a sequence of uniform non-overlapping patches and projected into an embedding space using a linear layer. The sequence is added with a position embedding and used as an input to a transformer model. The encoded representations of different layers in the transformer are extracted and merged with a decoder via skip connections to predict the final segmentation. Output sizes are given for patch resolution $P = 16$ and embedding size $K = 768$.

Segment Anything Model (SAM): A promptable foundation model for segmentation, recently adapted for medical images (SAM-Med).



(a) **Task**: promptable segmentation (b) **Model**: Segment Anything Model (**SAM**) (c) **Data**: data engine (top) & dataset (bottom)

Figure 1: We aim to build a foundation model for segmentation by introducing three interconnected components: a prompt-able segmentation *task*, a segmentation *model* (SAM) that powers data annotation and enables zero-shot transfer to a range of tasks via prompt engineering, and a *data* engine for collecting SA-1B, our dataset of over 1 billion masks.



Segmentation Loss Odyssey

Distribution-based:

- **Weighted Cross Entropy**: Penalizes errors in rare classes more heavily.

$$L_{CE}(\theta) = -\frac{1}{I}\sum_{i=1}^{I}\sum_{k\in\mathcal{L}} w_k^i s_k^i \log \hat{p}_k^i(\theta)$$

- **TopK**

$$L_{TopK}(\theta) = -\frac{1}{\sum_{i=1}^{I}\sum_{k\in\mathcal{L}} \mathbb{1}\{s_k^i = k \text{ and } \hat{p}_k^i < t\}} \sum_{i=1}^{I}\sum_{k\in\mathcal{L}} \mathbb{1}\{s_k^i = k \text{ and } \hat{p}_k^i < t\} \log \hat{p}_k^i(\theta)$$

- **Focal Loss**: Focuses on hard-to-classify pixels by down-weighting easy ones.

$$L_{\text{Focal}(\theta)} = -\frac{1}{I}\sum_{i=1}^{I}\sum_{k\in\mathcal{L}} \left(1-\hat{p}_k^i(\theta)\right)^{\gamma} s_k^i \log \hat{p}_k^i(\theta)$$

Region-based:

- **Dice Loss**: Measure the overlap between prediction and ground truth.

$$L_D(\theta) = 1 - 2\frac{\sum_{k\in\mathcal{L}} w_k \sum_{i=1}^{I} \hat{p}_i^k(\theta) s_i^k}{\sum_{k\in\mathcal{L}} w_k \sum_{i=1}^{I} \left(\hat{p}_i^k(\theta) + s_i^k\right)}$$

where $w_k = \frac{1}{\left(\sum_{i=1}^{I} s_i^k\right)^2}$.

- **Intersection over Union** (IoU)

$$L_D(\theta) = 1 - \frac{\sum_{k\in\mathcal{L}} \sum_{i=1}^{I} \hat{p}_i^k(\theta) s_i^k}{\sum_{k\in\mathcal{L}} w_k \sum_{i=1}^{I} \left(\hat{p}_i^k(\theta) + s_i^k - \hat{p}_i^k(\theta) s_i^k\right)}$$

- **Tversky Loss**: Generalization of Dice that allows controlling the trade-off between False Positives and False Negatives.

$$L_D(\theta) = \frac{\sum_{k\in\mathcal{L}}\sum_{i=1}^{I}\hat{p}_i^k(\theta)s_i^k}{\sum_{k\in\mathcal{L}}\sum_{i=1}^{I}\hat{p}_i^k(\theta)s_i^k + \alpha\sum_{k\in\mathcal{L}}\sum_{i=1}^{I}\hat{p}_i^k(\theta)(1-s_i^k) + \beta\sum_{k\in\mathcal{L}}\sum_{i=1}^{I}(1-\hat{p}_i^k(\theta))s_i^k}$$

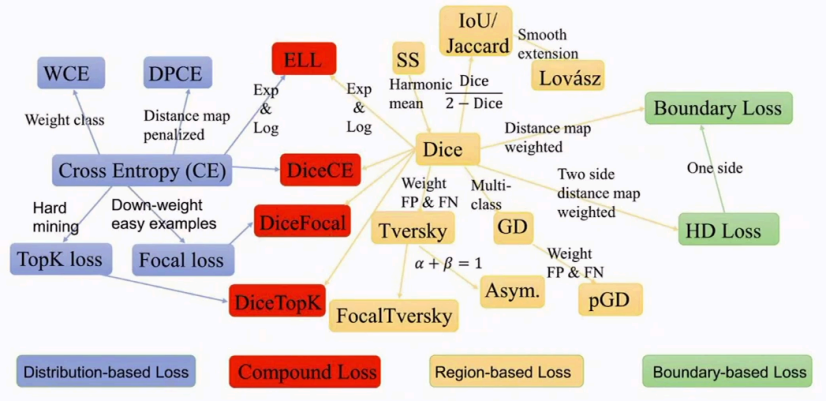
Boundary-based:

- **Hausdorff Distance (HD)**: Penalizes the distance between the boundaries of the predicted and ground truth masks.

$$L_{HD-DT}(\theta) = \frac{1}{I}\sum_{i=1}^{I}(s_i - \hat{p}_k^i(\theta)) \cdot \left(d_{S_i}^2 + d_{P_i}^2\right)$$

where d_S and d_P are the distance transforms of ground truth and segmentation.

Note: this formula only approximated the Hausdorff Distance



A **combination** of different loss terms is used in most cases. Typically, CE and DSC loss are combined.

EVALUATION

Checkout Metrics Reloaded

Federated Learning

Data Protection in Healthcare

Data protection is critical in healthcare due to the high sensitivity of patient records (medical history, genetics, diagnoses). It is regulated by laws such as:

- **GDPR (EU)**: General Data Protection Regulation, which regulates how personal data of EU residents is collected, stored, and processed.
- **HIPAA (USA)**: Health Insurance Portability and Accountability Act, which sets national standards for protecting sensitive patient health information.

Breaches of these regulations can lead to identity theft, loss of trust, and severe legal penalties.

Personal Data and Re-identification

Definition 22 (Personal Data (GDPR)) . Personal data is any information relating to an identified or identifiable living individual. Data that has been de-identified or pseudonymized but can still be used to re-identify a person remains personal data.

Anonymization: To be truly anonymized, the process must be irreversible.

Remark. **Re-identification Risk**: A famous study by Sweeney (2000) showed that 87% of US citizens can be uniquely identified using only their ZIP code, birth date, and sex. Thats why you dont use the birth date anymore, but the age.

From Centralized to Federated Learning

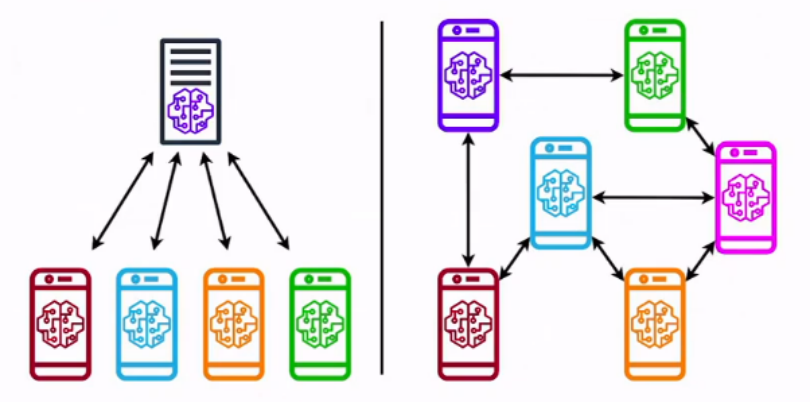
- Centralized ML: Training data from all sources is moved to a central server.
- Distributed On-Site Learning: Models are trained locally at each site with no information exchange.
- Federated Learning (FL): A collaborative learning approach where data remains at the source, and only model updates (weights) are shared with a central server.

Comparison: Centralized vs. Federated Learning

CENTRALIZED LEARNING	FEDERATED LEARNING
Trained on centralized data	Trained on distributed data

Data resides on the cloud or centralized server	Data resides at the various nodes in the network
Training takes place primarily in the cloud	Training happens primarily at the edge
Nodes/edge devices share local data	Nodes/edge devices share local version of the mode
Cannot operate on heterogeneous data	Can operate on heterogeneous data
Low user data privacy	High user data privacy

Centralized vs Decentralized Federated Learning



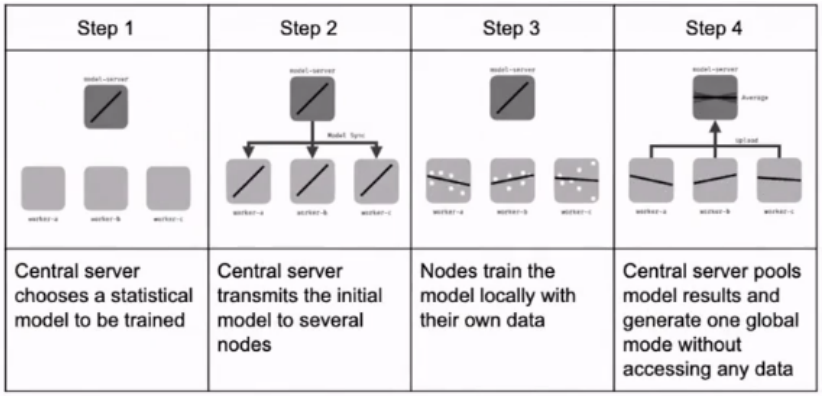
Centralized Federated Learning

Let's define the terms:

t communication round

Server
- w^t model weights
- T number of rounds
- C fraction of sampled clients of each round

Client
- w_k^t weights on the client site
- E number of local epochs
- η learning rate
- P_k subset of the data for client k
- n_k number of samples at the k client



Mathematical Formulation

Let $D = (x_i, y_i)_{i=1}^n$ be a dataset distributed to K clients C_k where $k \in \{1, ..., k\}$. We denote by $P = \{1, ...n\}$ and each client has a subset P_k such that $P = \bigcup_{k=1}^n P_k$. The goal is to solve:

$$\min_w f(w) = \min \frac{1}{n} \sum_{i=1}^{N} f_i(w)$$

where $f_i(w) = l(x_i, y_i, w)$ is a loss function. Then we have that the total loss function

$$f(w) = \frac{1}{n} \sum_{i=1}^{n} f_i(w) = \sum_{i=1}^{n} \frac{1}{n} f_i(w) = \sum_{k=1}^{K} \frac{1}{n} n_f F_k(w)$$

with $F_k(w) = \frac{1}{n_k} \sum_{i \in P_k} f_i(w)$ which is the loss at the distributed clients. So the loss function of the sample is the same, but now we combined the indices into the clients and then we write it by n_k. It is still the same thing, we just shifted the indices. At the client we do the same thing as globally.

Algorithms: FedSGD and FedAVG

FedSGD A simple version where each client performs one step of gradient descent per round.

Server executes:

1. **for** each round $t = 1, 2, ..., T$ **do**
    1. $S_t \subseteq \{1, ..., K\}$
    2. **for** each client $k \in S_t$ **do in parallel:**
        1. $g_k = \nabla F_k(w_t) = \nabla \left(\frac{1}{n_k} \sum_{i \in P_k} f_i(w_t) \right)$
    3. we add the gradients weighted by the sample number on the client side $g_t \leftarrow \sum_{k \in S_t} g_k \frac{n_k}{n}$
    4. $w_{t+1} \leftarrow w_t - \eta g_t$

The advantage is that it is a simple algorithm and theory of SGD applies but we have a high server-client communication utilization, which might be slow.

FedAVG

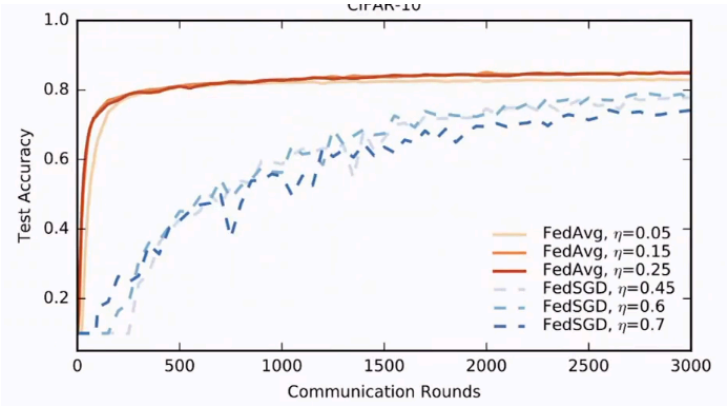
at server initialize w^0.
for each round $t = 1, 2, ..., T$ **do**
1. sample clients $S_t \subseteq \{1, ..., K\}$
2. $w_k^1 = w^t$
3. **for** each *local* epoch $e = 1, 2, ..., E$ **do**
    1. compute mini-batch gradient $g_k(w_k^e)$
    2. $w_k^{e+1} \leftarrow w_k^e - \eta g_k(w_k^e)$
4. return w_k^E to server

at the server we aggregate the weights and add them together weighted by their sample number:

$$m_t \leftarrow \sum_{k \in S_t} n_k$$

$$w^{t+1} \leftarrow \sum_{k \in S_t} \frac{n_k}{m_t} w_k^E$$

The advantage is that it substantially reduced communication utilization with a simple aggregation, but the convergence is slower compared to SGD in theory.



Non-IID Data Challenges

Typically, we assume that data samples $(x_i, y_i) \sim P(X, Y)$ are i.i.d. (independent and identically distributed).

We have now a 2-step modelling approach:
1. $k \sim p(k)$: draw a client from the distribution of clients
2. $(x, y) \sim P_{k(X,Y)}$: draw data sample from distribution at client

We assume clients are **non-IID** (which is the fact in reality most of the time) if $P_k \neq P_l$ for $k, l \in \{1, ..., K\}, l \neq k$. Let us assume that we can decompose P_k into $P_k(y \mid x) \cdot P_k(x)$ or $P_k(x \mid y) \cdot P_k(y)$

Non-IID Cases:

1. **Feature distribution skew:** So here the feature will be different
   $P_k(x) \neq P_l(x)$ but $P_k(y \mid x) = P_l(y \mid x)$
   e.g., shift in demographics, different devices

2. **Label distribution skew:** So we have the same labels, but the distribution of labels is different
   $P_k(y) \neq P_l(y)$ but $P_k(x \mid y) = P_l(x \mid y)$
   e.g., certain deseases only occur for elderly people
3. **Concept shift:** same label but different features
   $P_k(x \mid y) \neq P_l(x \mid y)$ but $P_k(y) = P_l(y)$
   e.g., houses around the globe are all look very different and have different features, but are still houses
4. **Concept shift:** inter-reader variability
   $P_k(y \mid x) \neq P_l(y \mid x)$ but $P_k(x) = P_l(x)$
   e.g., personal preferences, certain doctors scale the images a bit larger

Scaffold

In FedSGD, we compute the updates as:

$$w_h^{e+1} \leftarrow w_h^e - \eta \frac{1}{K} \sum_{k=1}^{K} \underbrace{\nabla F_k(w_h^e)}_{g_k(w_h^e)}$$

Since computing the full sum of all gradients is often not possible, control variables are introduced to get rid of the stochastisity of the gradient. If we for example have only one summand here, then it is not a good estimate. If we take all clients into account, then 2 clients can go into completely different directions, but the overall averaged direction is the option that works for everyone. On the otherhand if we go first in the one direction, then in the other and then back again with another, is not so good. Therefore we introduce control variables to get rid of this stochastisity:

- $c_k \approx \nabla F_k(w_h^e)$ (local control variable)
- $c \approx \frac{1}{K} \sum_{k=1}^{K} \nabla F_k(w_h^e)$ (global control variable)

Then, SCAFFOLD updates as:

$$g_k(w_h^e) - c_k + c \approx \frac{1}{K} \sum_{k=1}^{K} \nabla F_k(w_h^e)$$

So we take our existend graident g_k and substract the local aspect c_k and then add our global gradient c. So we always have at every step the influence of the global best gradient.

for each round $t = 1, 2, ...T$:

- sample clients $S_t \subseteq \{1, ..., K\}$
- distribute (w_t, c) to S_t
- **for each client** $k \in S_t$ **do in parallel**:
  - $w_k^1 \leftarrow w_t$
  - **for** $e = 1, ..., E$ **do**:
    - compute the minibatch gradient $g_k(w_k^e)$
    - $w_k^{e+1} \leftarrow w_k^e - \eta(g_k(w_k^e) - c_k + c)$
  - $c_k^+ \leftarrow g_k(w_t)$ or $c_k - c + \frac{1}{E\eta}(w_t - w_k^E)$
  - communicate deltas to server: $(\Delta w_k, \Delta c_k) \leftarrow \left(w_k^E - w_t, c_k^+ - c_k\right)$
  - $c_k \leftarrow c_k^+$
- **on the server do**:
  - $(\Delta w_t, \Delta c_t) = \frac{1}{|S_t|} \sum_{k \in S_t} (\Delta w_k, \Delta c_k)$
  - $w_{t+1} \leftarrow w_t + \eta \Delta w_t$

- $c_{t+1} \leftarrow c_t + \frac{|S_t|}{n}\Delta c_t$

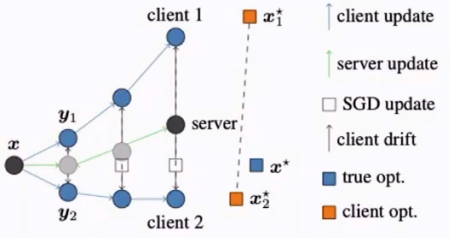


Figure 1. Client-drift in FEDAVG is illustrated for 2 clients with 3 local steps ($N = 2$, $K = 3$). The local updates $\boldsymbol{y}_i$ (in blue) move towards the individual client optima $\boldsymbol{x}_i^\star$ (orange square). The server updates (in red) move towards $\frac{1}{N}\sum_i \boldsymbol{x}_i^\star$ instead of to the true optimum $\boldsymbol{x}^\star$ (black square).

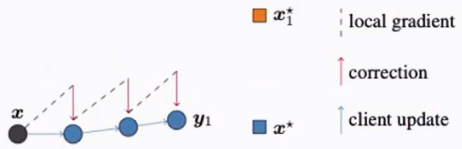
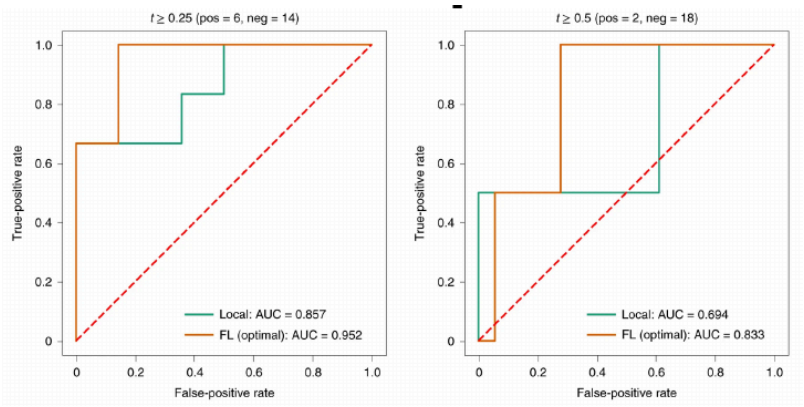


Figure 2. Update steps of SCAFFOLD on a single client. The local gradient (dashed black) points to $\boldsymbol{x}_1^\star$ (orange square), but the correction term $(\boldsymbol{c} - \boldsymbol{c}_i)$ (in red) ensures the update moves towards the true optimum $\boldsymbol{x}^\star$ (black square).

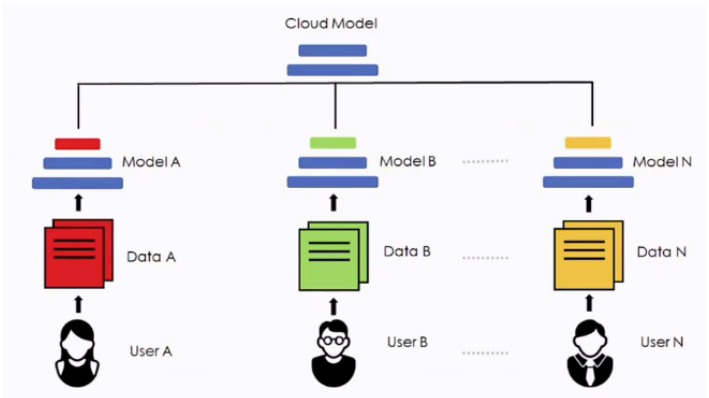
Remark. During Covid-19 this was tested world-wide and it brought in an performance boost compared to the loacl model.



PERSONALIZATION TECHNIQUES

Do we want to have the same model everywhere? Somethimes it makes sense to let it adapt to local cirucumstances. To improve performance on heterogeneous data, models can be personalized:

Personalization Layers: Splitting the model into global layers (shared, blue) and local layers (private to each client).



If we now have a classification task, which part of the network should we make global and which make me local? The last layers are responsible for the classification so it would make sense to locallize the head of the model.

FedBN

Keeping Batch Normalization parameters local to account for feature shifts turned out to work really well.

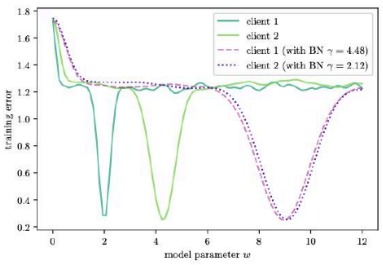


Figure 1: Training error on local datasets for two clients respectively with and w/o BN, where BN harmonizes the loss surface.

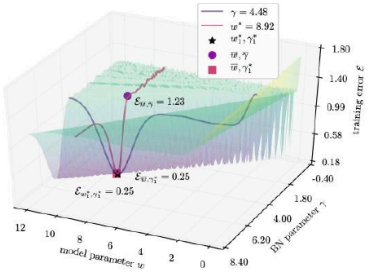


Figure 2: Error surface of a client for model parameter $w \in [0.001, 12]$ and BN parameter $\gamma \in [0.001, 4]$. Averaging model *and* BN parameters leads to worse solutions.

Hypernetworks

What is this hypernetwork doing? It gives us client specific weights. We take a vector, embedding or whatever and feed it to a network and the network gives us weights for the network that we want to train. The output is then a regression problem because we want to predict continuous values and we have a huge output → number of model parameters. So this is a hard task. The basic principle again is then that we give the predicted models from the server to the client and the client gives us then the weights back.

$$\mathcal{L} = \arg \min_{\varphi, \{v_k\}_{k=1}^K} \frac{1}{K} \sum_{k=1}^{K} F_k\Big(h_\varphi(v_k)\Big)$$

where $w_k = h_\varphi(v_k)$ and v_k is the input we give the hypernetwork and h_φ is the hypernetwork itself.

Algorithm for training hypernetwork:

1. **for** each round $t = 1, 2, ..., T$ **do**
   1. sample clients $S_t \subseteq \{1, ..., K\}$
   2. set $w_k = h_\varphi(v_k)$ for all $k \in S_t$ and $\overline{w}_k = w_k$
   3. **for** each $e \in \{1, ..., E\}$ **do**
      1. sample mini-batch B
      2. $\overline{w}_k^e \leftarrow \overline{w}_k - \eta \nabla_w F_k(B)$
   4. $\Delta w_k = \overline{w}_k^E - w_k$ transfer to server
   5. **aggregate at server:**

$$\varphi = \varphi - \alpha\Big(\nabla_\varphi h_\varphi(v_k)\Big)\Delta w_k$$

$$v_k = v_k - \alpha\Big(\nabla_v h_\varphi(v_k)\Big)\Delta w_k$$

Privacy and Security in Federated Learning

Despite data staying local, FL is vulnerable to several attacks:

1. Inference Attacks: Inferring class representatives, membership, or even training samples from gradients (Deep Leakage from Gradients).
2. Malicious Server: A server using a GAN to reconstruct client data.
3. Poisoning Attacks: Backdoor or replacement attacks to manipulate the global model.

Example 14 — Inference of class representatives .

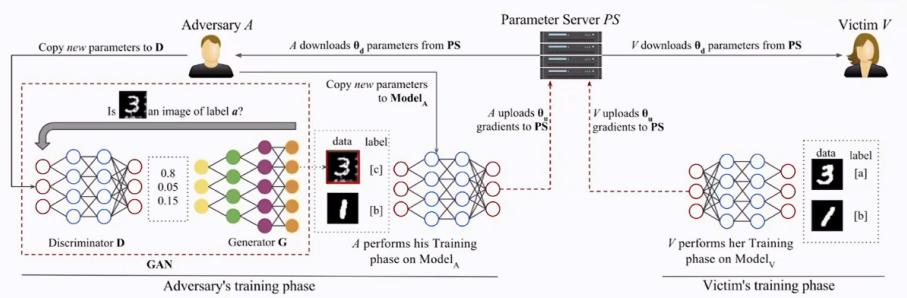
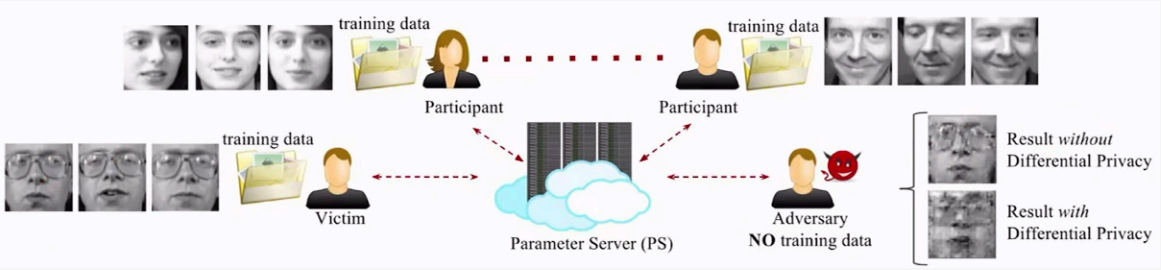


Figure 4: GAN Attack on collaborative deep learning. The victim on the right trains the model with images of 3s (class *a*) and images of 1s (class *b*). The adversary only has images of class *b* (1s) and uses its label *c* and a GAN to fool the victim into releasing information about class *a*. The attack can be easily generalized to several classes and users. The adversary does not even need to start with any true samples.



Example 15 — Inference of Training Samples and Labels .

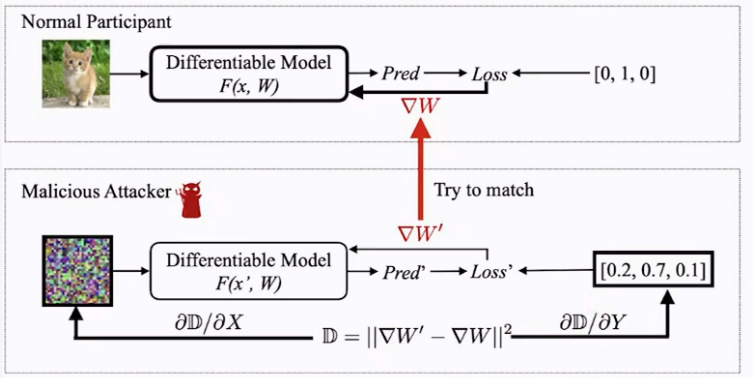
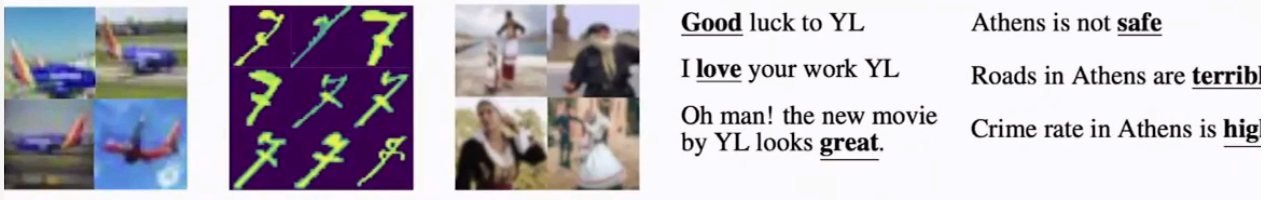


Figure 2: The overview of our DLG algorithm. Variables to be updated are marked with a bold border. While normal participants calculate ∇W to update parameter using its private training data, the malicious attacker updates its dummy inputs and labels to minimize the gradients distance. When the optimization finishes, the evil user is able to obtain the training set from honest participants.

Sharing the gradient can give away a lot of information.

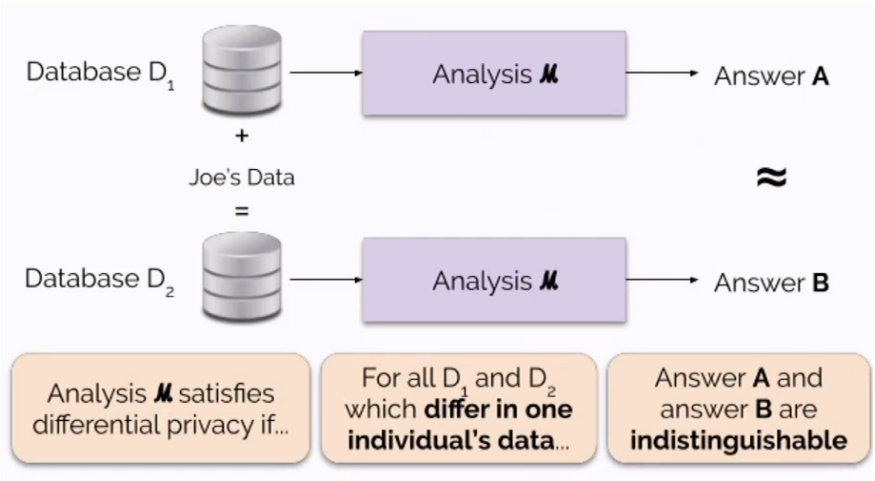
Example 16 — Poison attack .



ure 1: Illustration of tasks and edge-case examples for our backdoors. Note that these examples are *not* fou he train/test of the corresponding datasets. (a) Southwest airplanes labeled as "truck" to backdoor a CIFAR- ssifier. (b) Images of "7" from the ARDIS dataset labeled as "1" to backdoor an MNIST classifier. (c) Peop traditional Cretan costumes labeled incorrectly to backdoor an ImageNet classifier (intentionally blurred). (sitive tweets on the director Yorgos Lanthimos (YL) labeled as "negative" to backdoor a sentiment classifi Sentences regarding Athens completed with words of negative connotation to backdoor a next word predict

Federated Learning with Differential Privacy (DP)

We want to make it harder for attackers to retrieve images from the client side. We can achieve this with differential privacy.



Definition 23 (differential privacy) . A randomized mechanism (algorithm) $\mathcal{M} : X^n \to \mathbb{R}$ satisfies (ε, δ)-DP if for all measurable sets $S \subseteq \mathcal{R}$ and for any two adjacent datasets $D, \overline{D} \subset \mathcal{X}^n$ (i.e., differing in one individual's data)

$$\mathbb{P}[\mathcal{M}(D) \in S] \leq \exp(\varepsilon) \cdot \mathbb{P}\left[\mathcal{M}\left(\overline{D}\right) \in S\right] + \delta,$$

where $\varepsilon, \delta > 0$.

Note: If $\delta = 0$, $\mathcal{M}$ is called pure ε-differential private.

- M: The "Mechanism" or algorithm (analysis/query) running on the data.
- D and $\overline{D}$: Two "adjacent" datasets. They are identical except one contains a specific individual's data (e.g., "Joe's Data") and the other does not.
- S: Any possible outcome of the analysis.
- ε (Epsilon): The privacy budget.
  - A smaller ε means higher privacy (probabilities are closer).

  - A larger ε allows for more divergence, favoring utility over privacy.
- δ (Delta): The "failure probability." The small chance the privacy guarantee might fail.

Definition 24 (Sensitivity) . Let W be a metric space with distance function $d_W(\cdot,\cdot)$. The sensitivity $S_W(h)$ of a function $h : X^n \to W$ is the amount that the function value varies when a single entry changes:

$$S_W(h) := \sup_{w,\overline{w}:d_w(w,\overline{w})=1} d_W(h(w), h(\overline{w}))$$

$\Rightarrow$ Note the relation to the Lipschitz constant of a function.

Sensitivity Analysis

Assume the model weights w_k are bounded: $\|w_k\| \leq C$. Then, the sensitivity of the k-th client update in FedAVG is given by:

$$\begin{aligned}
S_k &= \sup_{P_k,\overline{P}_k} \| \arg\min_w F_{k(w;P_k)} - \arg\min_w F_k\left(w;\overline{P}_k\right) \| \\
&= \sup_{P_k,\overline{P}_k} \| \arg\min_w \frac{1}{|P_k|} \sum_{i\in P_k} f_i(w) - \arg\min_w \frac{1}{|P_k|} \sum_{j\in\overline{P}_k} f_j(w) \| \\
&= \frac{1}{|P_k|} \sup_{P_k,\overline{P}_k} \| w_k - \overline{w}_k \| \\
&= \frac{2C}{|P_k|}
\end{aligned}$$

To ensure that the local training mechanism

$$M_\varepsilon = \left[\arg\min_w \frac{1}{|P_k|} \sum_{i\in P_k} f_i(w)\right] + n,$$

where $n \sim N(0, \sigma^2 I)$, preserves (ε, δ)-DP, we need to add noise with level:

$$\sigma_k \geq c \cdot \frac{S_k}{\varepsilon}$$

where $c \geq \sqrt{2\ln\left(\frac{1.25}{\delta}\right)}$.

FedAVG with DP Algorithm

For each round $t = 1...T$:
1. **Sample client** $k \in \{1, ..., K\}$
2. **Update the local weights:**

$$w_k^t \leftarrow \arg\min_w \left\{ F_k(w) + \frac{\mu}{2} \|w - \tilde{w}^{t-1}\|_2^2 \right\}$$

3. **Clip the local weights:**

$$w_k^t \leftarrow \frac{w_k^t}{\max\left(1, \frac{\|w_k^t\|}{C}\right)}$$

4. **Add noise:**

$$\tilde{w}_k^t \leftarrow w_k^t + n_k^t, \quad n_k^t \sim \mathcal{N}(0, \sigma_k)$$

5. **Send to server**
6. **At server:**

$$w^t = \sum_{i=1}^{k} \frac{n_k}{n} w_k^t$$

7. **The server broadcasts:**

$$\tilde{w}^t = w^t + n_s^t, \quad n_s^t \sim \mathcal{N}(0, \sigma_s)$$

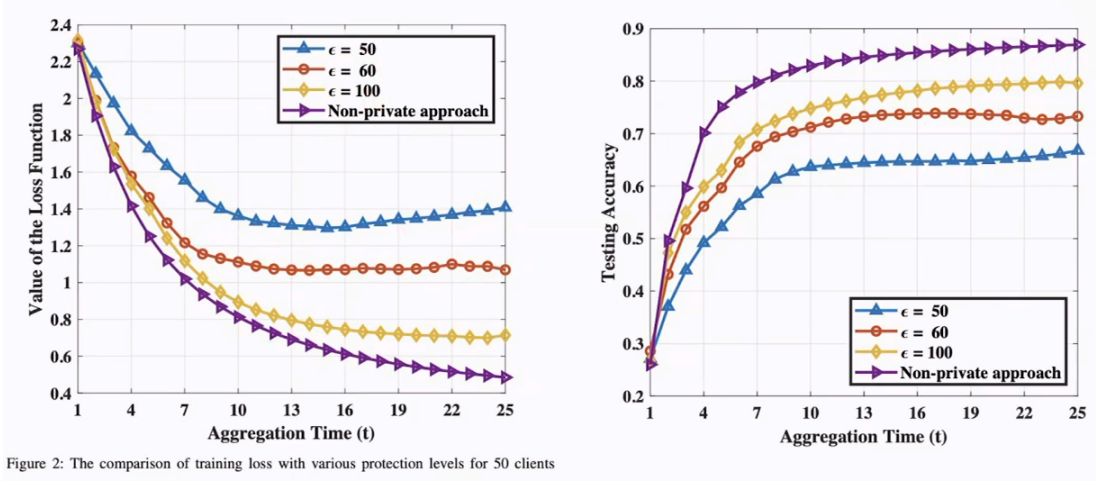


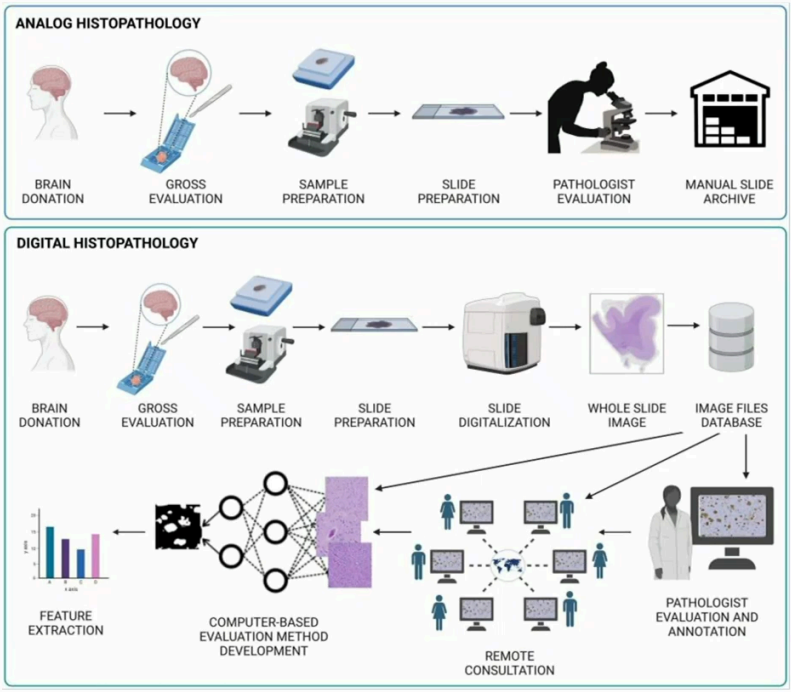
Figure 2: The comparison of training loss with various protection levels for 50 clients

Microscopy

Why Microscopy matters in Medicine?

Microscopy reveals structure and function at the cellular and tissue level, which is critical for diagnosis, research, and therapy decisions.

Key medical applications:
- Histopathology: For example, cancer diagnosis through tissue examination.
- Hematology: Analysis of blood smears.
- Infectious disease identification: Detecting pathogens.
- Cell biology & drug discovery: Understanding cellular mechanisms.



Why Machine Learning?

Traditional manual microscopy analysis is:
- Time-intensive: Pathologists must manually scan large slides.
- Subjective: High variability between different practitioners.
- Hard to scale: Difficult to handle large datasets of high-resolution slides.

Machine Learning (ML) Advantages:
- Automates repetitive tasks.
- Delivers quantitative measures (e.g., cell counts, morphology).

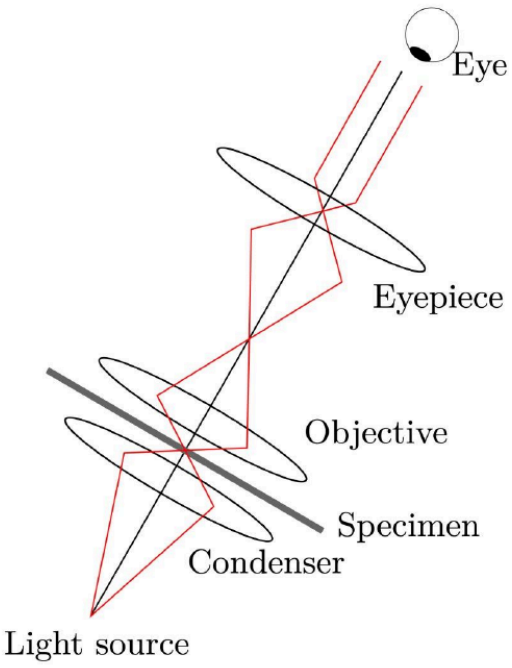
- Enables pattern discovery beyond human perception.

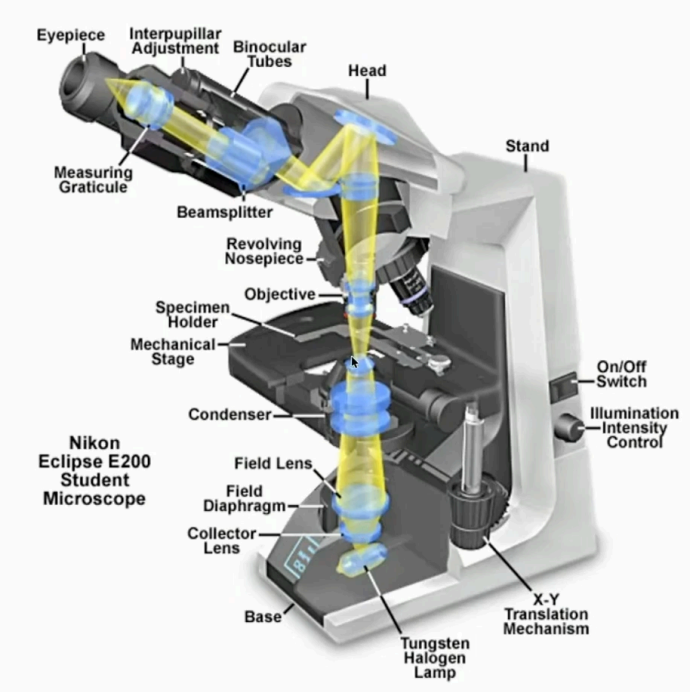
Microscopy Modalities Overview

Modality	Contrast Mechanism	Typical Data Type	Main Applications	Advantages	Limitations	Typical Spatial Resolution
Brightfield Microscopy	Absorption of light by stains (e.g., H&E, IHC)	2D images, Whole Slide Images (WSI)	Histopathology, cancer diagnosis, tissue morphology	Cheap, standardized, clinically established	Requires staining, limited molecular specificity	0.2–0.5 µm (light diffraction limit)
Phase Contrast Microscopy	Optical phase shifts from refractive index differences	2D time-lapse images	Live cell imaging, cell motility, cell division	No staining, suitable for living cells	Limited molecular specificity	0.2–0.5 µm
Fluorescence Microscopy (Widefield)	Fluorophore excitation/emission	2D multi-channel images	Protein localization, biomarker detection	Molecular specificity, multichannel	Out-of-focus blur, photobleaching	0.2–0.3 µm (lateral), 0.5–0.7 µm (axial)
Confocal Microscopy	Optical sectioning via pinhole rejection	2D slices, 3D stacks	3D tissue/cell imaging, morphology	High contrast, true 3D imaging	Slower, phototoxicity	0.18–0.25 µm (lateral), 0.5 µm (axial)
Electron Microscopy (TEM)	Electron transmission and scattering	2D grayscale images	Subcellular ultrastructure, organelles	Extremely high resolution	Expensive, destructive, grayscale	< 1 nm ($\approx$0.1–0.5 nm)
Electron Microscopy (SEM)	Electron surface scattering	2D surface topology	Cell surfaces, materials, morphology	3D-like surface detail	Limited internal structure	1–10 nm

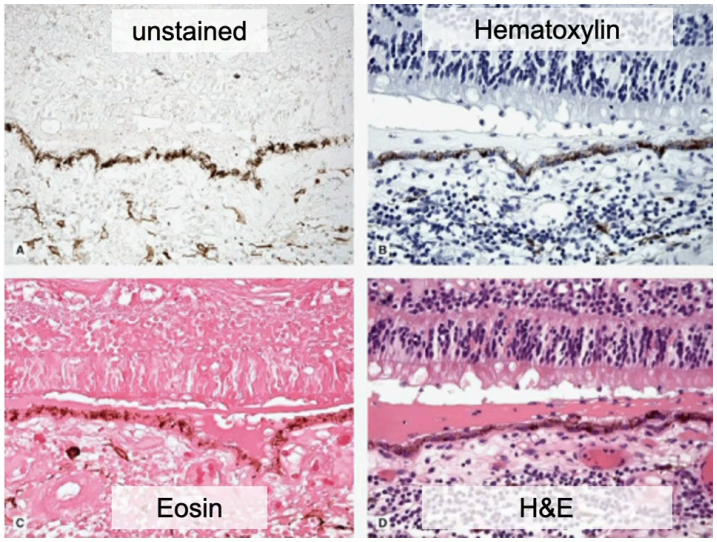
Brightfield Microscopy

White light passes through the sample, and the image is based on absorption by stains. This is the most used method in standard histology.



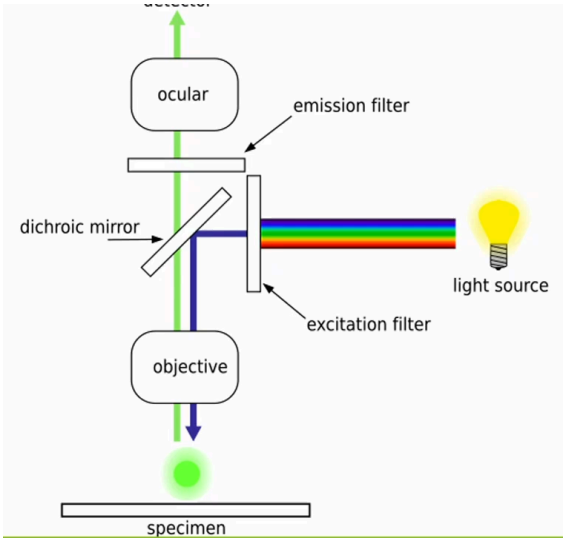


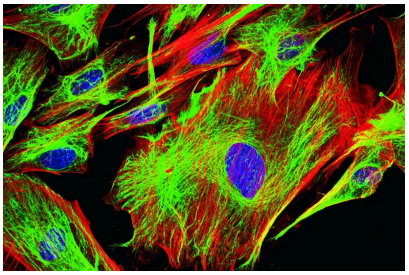
Definition 25 (Staining) . Biological tissues are largely transparent. Stains (like Hematoxylin & Eosin / H&E) bind selectively to cellular components (e.g., nuclei vs. cytoplasm) to convert biochemical differences into visible intensity differences.



Fluorescence microscopy

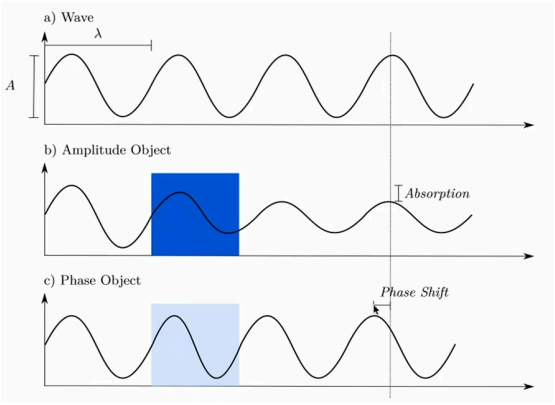
Uses fluorophores that absorb excitation light and emit light at a longer wavelength. We can prepare the sample with dye by fluroscent stain or use fluorescent proteins. With that we can get multiple images with different channels that all come from different stains. With that we can get highly specific imaging.

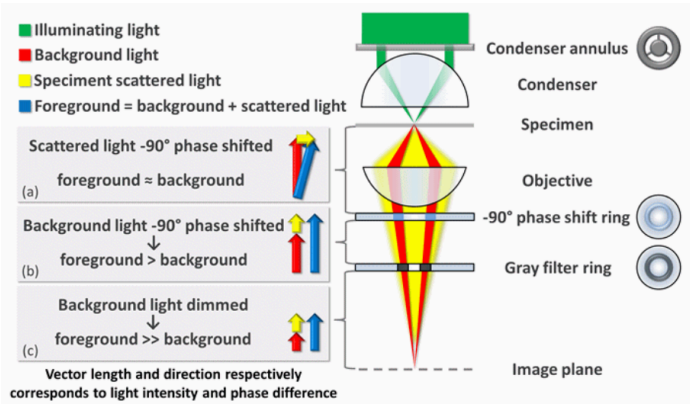




Phase Contrast

Here we use the refrection index of the materials which accounts for different phase shifts

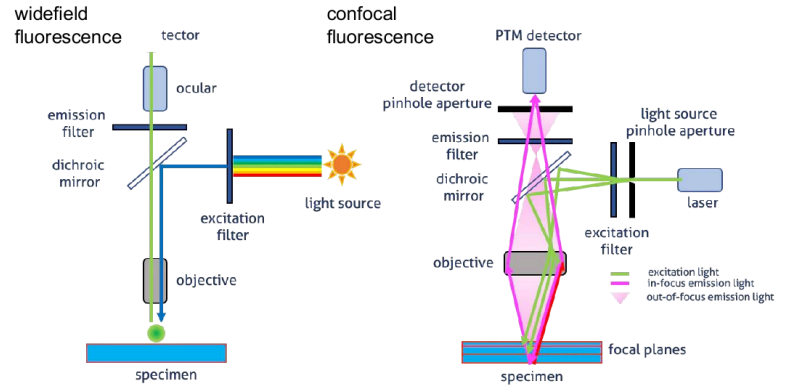


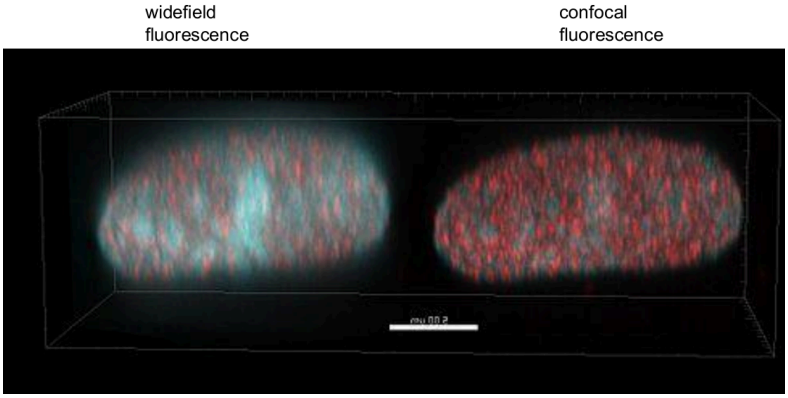


With this you can also investigate still living entities.

Confocal Microscopy

A technique using a pinhole to reject out-of-focus light, allowing for 3D "optical sectioning". Only the things that are on the same height appear in-focus.

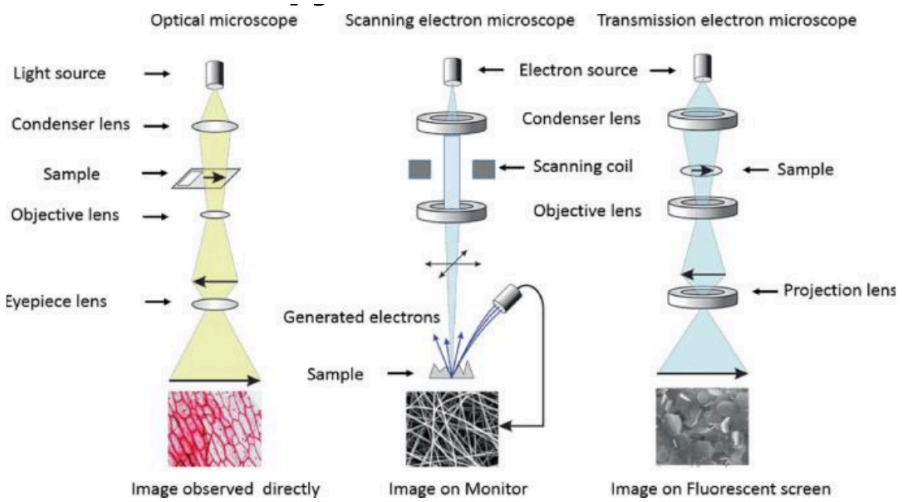


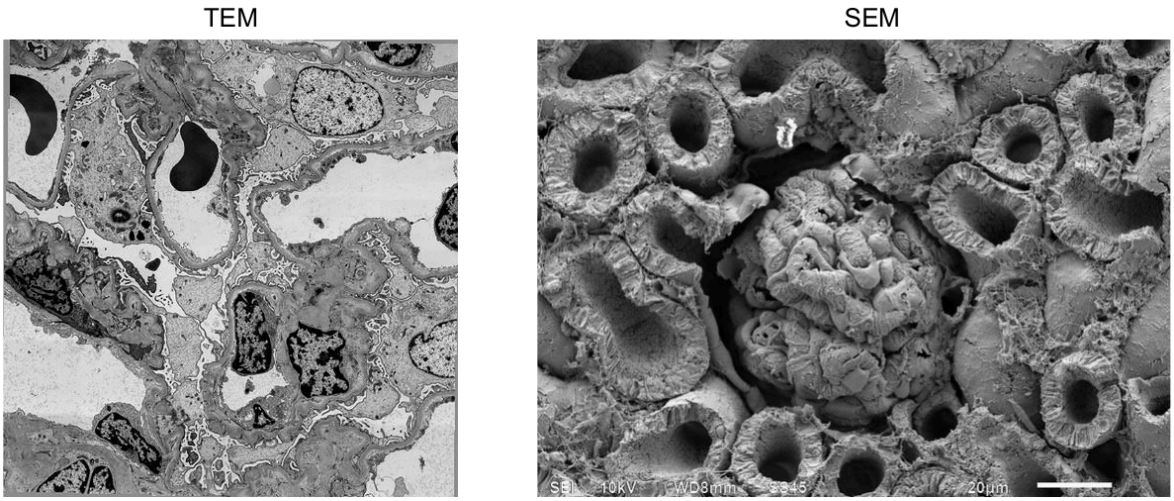


Electron microscopy

Uses electrons instead of photons for resolution up to 1,000,000x (optical microscopy up to 1,500x). It acchievs that by using electromagnetic lenses instead of glass. The sample preparation requires lot of work, no live imaging possible. The sample must be within vacuum. There are two types:

- Transmission EM: internal ultrastructure
- Sampling EM: surface morphology and topology

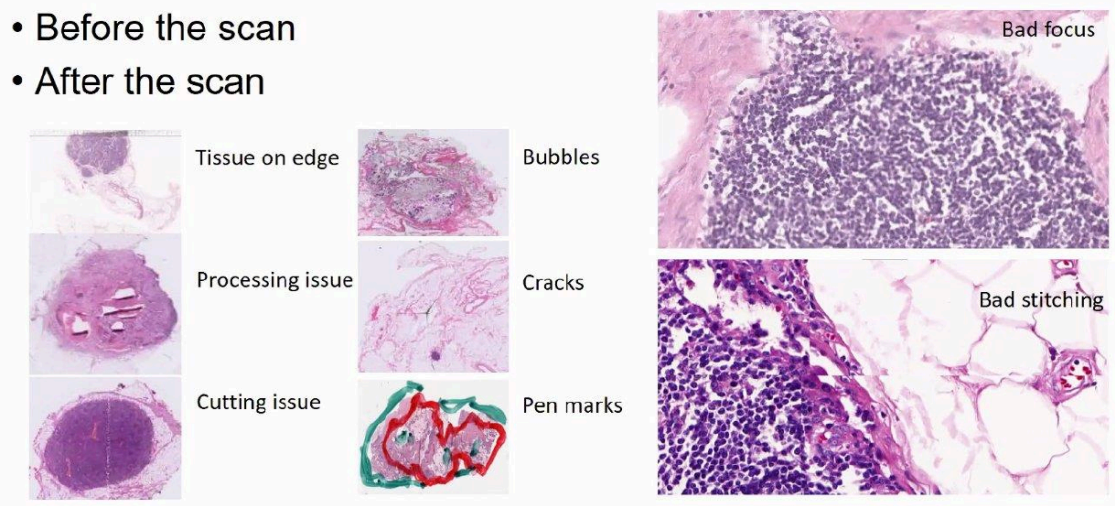




Key Challenges in Medical Imaging

1. Limited labeled data Expert annotations require
   - Pathologists
   - Biologists
   - Hours per slide
2. Class imbalance
   Many tasks involve rare events (Mitoses, ...)
3. Whole slide images (WSI) can be
   - 100,000 x 100,000 pixels
   - 10GB per image
   - Multiple channels (fluorescence)
4. Domain shifts
   - Scanner types
   - Straining protocol
   - population
5. Explainability
   - Interpretable predictions
   - Visual explanations

Image Level Challenges



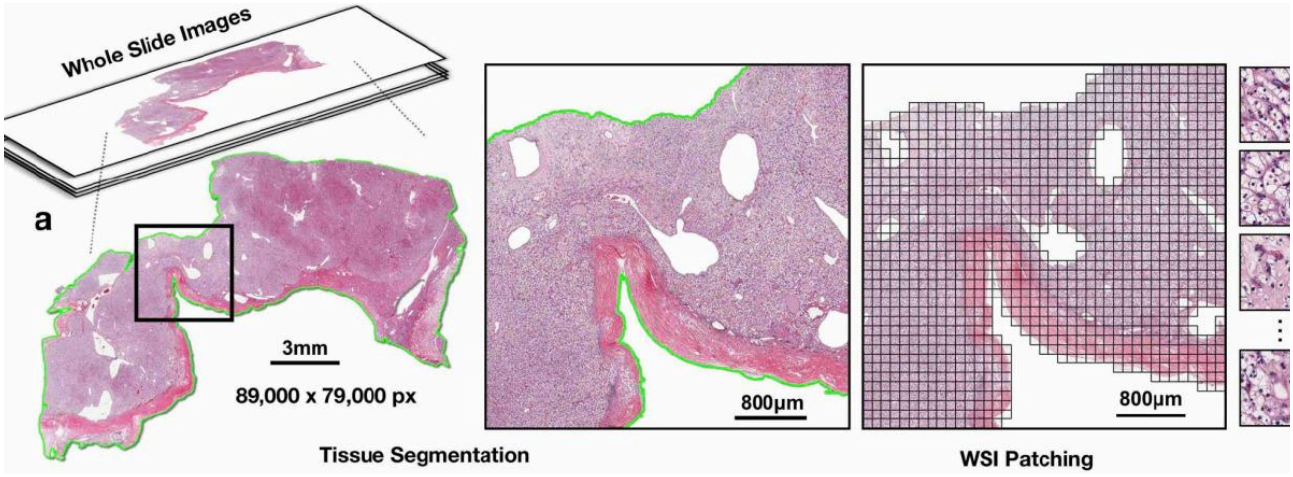
Annotation Challenges: Weak labels

Truth labels are less accurate than required

- Case-level annotation
- Slide-level annotation
- Region-level annotation
- Tile-level annotation
- (Pixel-level annotation)

From WSI to Tiles

On the size of the tiles which are usually 224x224 the image classification happens.



Multiple Instance Learning (MIL)

Due to the size of WSIs and the lack of pixel-level labels, we often use Weakly Supervised Learning through MIL.

Definition 26 (Multiple Instance Learning (MIL)) . Instead of individual labeled samples, we have bags of instances/samples $X_j = \{x_{\{j1\}}, x_{\{j2\}}, ..., x_{\{jK\}}\}$.
- A bag is labeled $Y = 0$ if all instances are negative.
- A bag is labeled $Y = 1$ if at least one instance is positive.

Theorem 1 (Permutation Invariance) . A MIL scoring function $S(X)$ must be symmetric (invariant to the order of instances). It can be decomposed as:

$$S(X) = g\left(\sum_{x \in X} f(x)\right)$$

where f and g are sutiable transformations.

Theorem 2 () . For any $\varepsilon > 0$, a Hausdorff continuous symmetric $S : \mathbb{R}^{K \times D} \to \mathbb{R}$ can be arbitrarily approximated by

$$|S(\boldsymbol{X}) - g\Big(\max_{\boldsymbol{x} \in \boldsymbol{X}} f(\boldsymbol{x})\Big)| < \varepsilon.$$

Deep MIL Approaches

1. Transform each instance $\boldsymbol{x}$ using the transformation f_θ
2. Combine transformed instances using a symmetric (permutation-invariant) function σ (also called MIL **pooling** function)
3. Transform combined instances using a function g_ψ to obtain

Two main approaches:
- **Instance-level approach**
  The transformation f is an instance-level classifier. The instance-level scores are aggregated by MIL pooling σ. The function g is the identity.
- **Embedding-level approach**
  The transformation f maps instances to low-dimensional embeddings. MIL pooling σ used to obtain bag representation. g becomes a bag-level classifier.

Attention-based MIL pooling

$$\sigma(\boldsymbol{h}_1, ..., \boldsymbol{h}_K) = \sum_{k=1}^{K} a_k \boldsymbol{h}_k,$$

where $\boldsymbol{h}_k = f(\boldsymbol{x}_k) \in \mathbb{R}^M$ and the attention weights are computed as

$$a_k = \frac{\exp(w^\top \tanh(V\boldsymbol{h}_k))}{\sum_{j=1}^{K} \exp\big(w^\top \tanh\big(V\boldsymbol{h}_j\big)\big)}$$

$w \in \mathbb{R}^{L \times 1}$ and $V \in \mathbb{R}^{L \times M}$ are learnable weight.

Gated attention variant

$$a_k = \frac{\exp\{w^\top(\tanh(V\boldsymbol{h}_k) \odot \sigma(U\boldsymbol{h}_k))\}}{\sum_{j=1}^{K} \exp\{w^\top(\tanh(V\boldsymbol{h}_j) \odot \sigma(U\boldsymbol{h}_j))\}}$$

where $U \in \mathbb{R}^{L \times M}$ is also learnable.