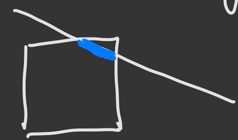
ALGEBRAIC RECONSTRUCTION METHODS

lets assume, for one X-Ray, a constant attenuation (x_j) in pixel j. In that case, we have

$$b_i = \sum_{j \in \text{ray}_i} a_{ij} \cdot x_j$$

where a_{ij} is the length of the i-th ray in pixel j.



the sum is over all pixel that intersect with ray i.

Alternatively, we can write

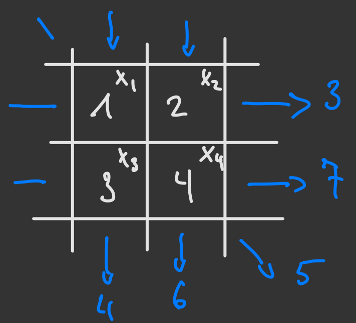
$$b_i = \sum_{i=1}^{n} a_{ij} \cdot x_j$$

if $a_{ij} = 0$ when the i-th ray does not intersect pixel j.

n ... # pixel

Given $i \in \{1, \dots, m\}$ (i.e., m rays), we get a system of linear equations: $Ax = b$

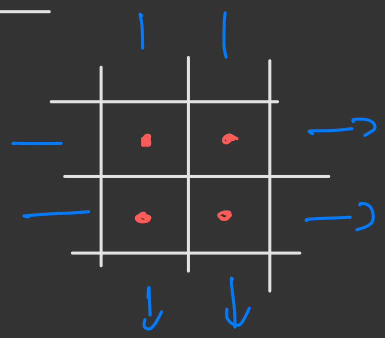
Example:



$$\underbrace{\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}}_{x} = \underbrace{\begin{pmatrix} 3 \\ 7 \\ 5 \\ 6 \\ 4 \end{pmatrix}}_{b}$$

This becomes large pretty quickly!

Another example:



$\Rightarrow$ Four unknowns, four rays!

With respect to the previous example, we get infinitely many solutions, for $k \in \mathbb{R}$

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} + k \cdot \begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix}$$

With enough rays, we have a unique solution.

⚠ In CT, we rather use <u>iterative</u> solvers/approaches!

<u>One approach:</u> KACZMARZ method

$\langle r_1, x \rangle = b_1$

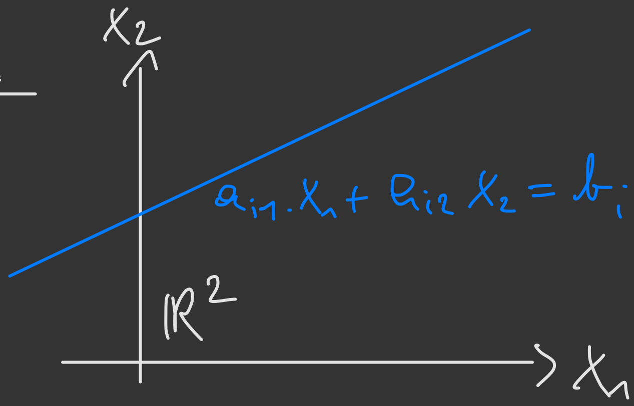
Geometric perspective!

$$\begin{cases} a_{11} \cdot x_1 + a_{12} x_2 + \dots & a_{1n} x_n = b_1 \\ a_{21} \; x_1 + a_{22} x_2 + \dots & a_{2n} \; x_n = b_2 \\ \vdots & \\ a_{m1} x_1 + a_{m2} x_2 + \dots & a_{mn} \; x_n = b_m \end{cases}$$

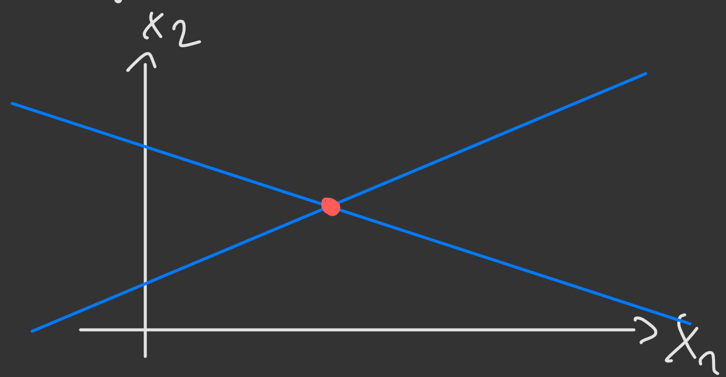
$\langle r_m, x \rangle = b_m$

Each of <u>these equations</u> defines an (affine) hyperplane.

Example:

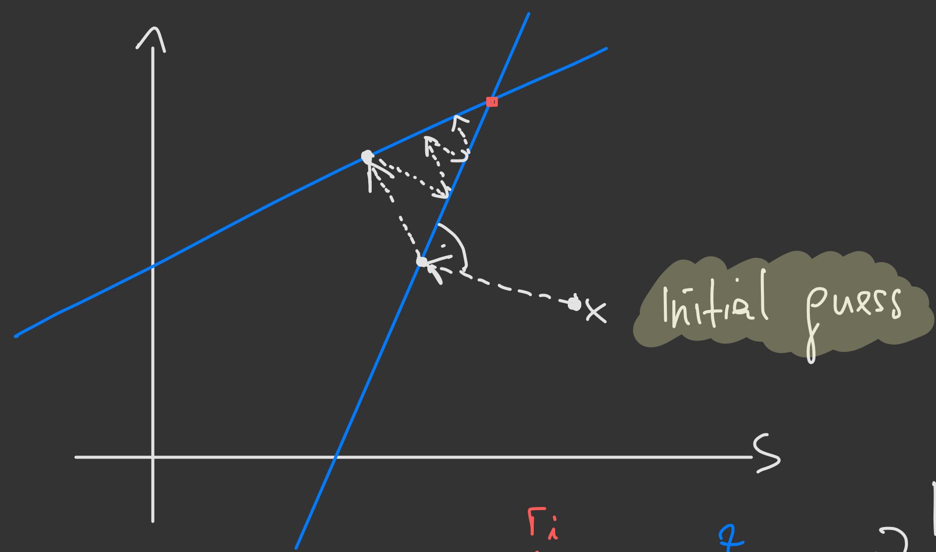


Assuming $Ax = b$ has a unique solution, then this solution is the point $x \in \mathbb{R}^n$ where the hyperplanes intersect.

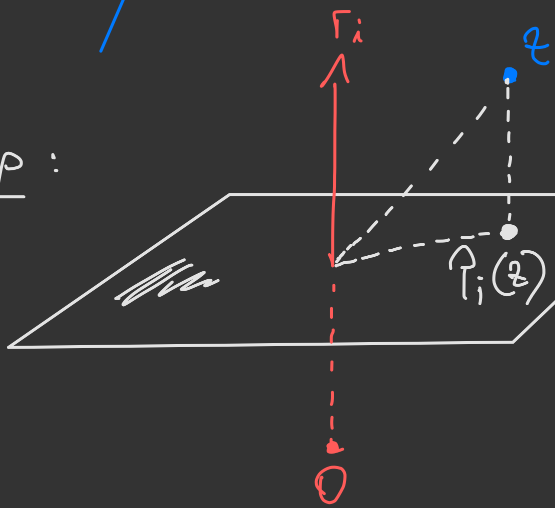


Idea of Kaczmarz's method:

In each iteration, compute a new iteration vector such that one of the equations is satisfied. How? By projecting the current estimate (current x) onto one of the hyperplanes: $\langle r_i, x \rangle = b_i$ for $i = 1, \dots, m, 1, \dots, m, 1, \dots$ (until convergence).



Projection step:



Hyperplane

$$H_i = \{ x \in \mathbb{R}^n : \langle r_i, x \rangle = b_i \}$$

$$P_i(z) = z + \frac{b_i - \langle r_i, z \rangle}{\|r_i\|^2} \cdot r_i$$

<u>Algorithm:</u> $x^{(0)}$ ... initial guess

for $k = 0, 1, \ldots$

$i = k \pmod m$ → <u>Cycle</u> through the hyperplanes

$$x^{(k+1)} = P_i(x^{(k)}) = x^{(k)} + \frac{b_i - \langle r_i, x^{(k)} \rangle}{\|r_i\|^2} \cdot r_i$$

check convergence → break if converged

end for

Alternatively, one could select i randomly, or cycle symmetrically ($i = 1, 2, \ldots m-1, m, m-1, \ldots 2, 1, \ldots$)

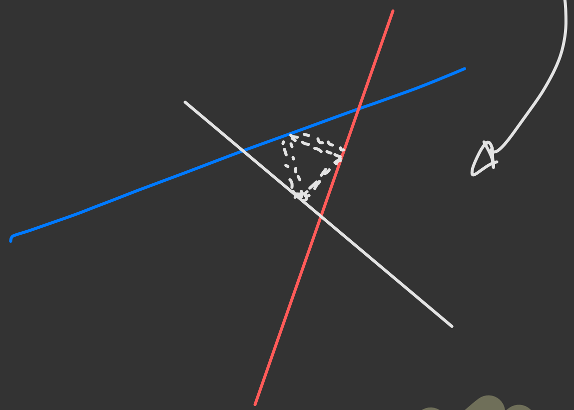
<u>Note</u>: the ordering of the rows of A is important! (it influences convergence speed!).

(One approach to check convergence: $\|Ax - b\|^2 <$ tolerance (e.g. $1e^{-8}$))

The whole algorithm is also known as ART (Algebraic reconstruction technique).

Note: we have assumed that a unique solution exists for $Ax = b$ $\Rightarrow$ convergence ✓
(all affine hyperplanes intersect at a single point).

If not: we get cyclic convergence



Solution: introduce a relaxation parameter (ω_k) with $\omega_k = \frac{1}{\gamma_k}$!

Advantages of this approach (e.g. vs. filtered backprojection):

- It's easy to include constraints, e.g., positivity $x_i \geq 0$, or $\underbrace{0 \leq x_i \leq 1}_{\text{Box constraints}}$ for $i = 1, \dots, n$