

Algebraic Iterative Reconstruction Methods for Computed Tomography

Course 22485

Per Christian Hansen

DTU Compute
Department of Applied Mathematics and Computer Science
Technical University of Denmark

Plan for Today's Lectures

- 1 A bit of motivation.
- 2 The algebraic formulation; matrix notation and interpretation.
- 3 Kaczmarz's method (also known as ART) – fully sequential.
- 4 Cimmino's method and variants – fully simultaneous.
- 5 Extensions of the methods.

Plan for Today's Lectures

- 1 A bit of motivation.
- 2 The algebraic formulation; matrix notation and interpretation.
- 3 Kaczmarz's method (also known as ART) – fully sequential.
- 4 Cimmino's method and variants – fully simultaneous.
- 5 Extensions of the methods.

Points to take home today:

- Algebraic formulations provide more flexibility than formulations based on inversion of the Radon transform.
- Linear algebra provides a concise framework for formulating the associated algorithms for algebraic formulations.
- Convergence analysis of iterative algebraic methods is well understood.

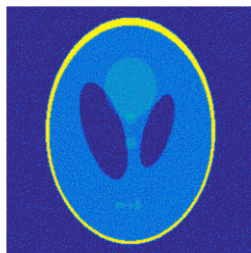
FBP: Filtered Back Projection

- This is *the classical* method for 2D parallel-beam reconstructions.
- There are similar (approximate) methods for 3D, such as FDK.
- Many year of use \rightarrow lots of *practical experience*.
- The FBP method is *very fast* (it uses the Fast Fourier Transform)!
- The FBP method has *low memory requirements*.
- With many data, FBP gives very good results.
- Example with 3% noise:

Phantom



FBP 180 projections



FBP 1000 projections



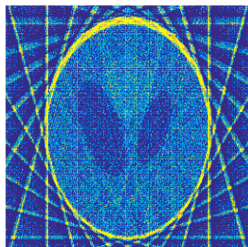
FBP Versus Algebraic Methods

- Limited data, or nonuniform distribution of projection angles or rays \rightarrow *artifacts* appear in FBP reconstructions.
- Difficult to incorporate constraints (e.g., nonnegativity) in FBP.
- Algebraic methods are more flexible and adaptive.
- Same example with 3% noise and projection angles $15^\circ, 30^\circ, \dots, 180^\circ$:

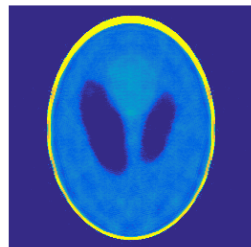
Phantom



FBP (iradon)



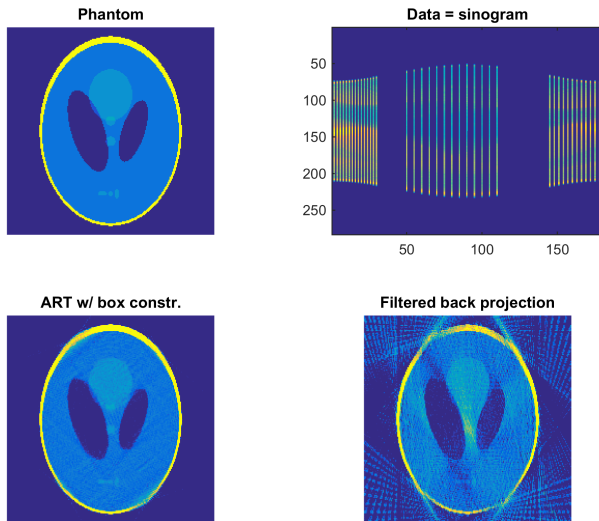
ART w/ box constraints



Algebraic Reconstruction Technique, box constraints (pixel values $\in [0,1]$).

Another Motivating Example: Missing Data

Irregularly spaced angles & “missing” angles also cause difficulties for FBP:



The Lambert-Beer Law – For Monochromatic X-Rays

The *change* dI in intensity is proportional to the intensity $I(\ell)$ and the slab thickness $d\ell$. The proportionality constant is called the *linear attenuation coefficient* $\mu(\ell)$. Hence:

$$dI = -\mu(\ell) I(\ell) d\ell$$

which in the limit of $d\ell \rightarrow 0$ results in

$$\frac{dI}{d\ell} = -\mu(\ell) I(\ell) .$$

This is a linear first-order differential equation with the general solution

$$I(\ell) = C \exp \left\{ - \int \mu(\ell) d\ell \right\} .$$

The initial condition $I(0) = I_0$ determines the constant and we arrive at the usual form of the Lambert-Beer law

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

Setting Up the Algebraic Model

Taking the “log” on both sides of the Lambert-Beer law we see that the attenuation $I - I_0$ of the i th X-ray through the object is a line integral of the attenuation coefficient μ along the ray:

$$b_i = \int_{\text{ray}_i} \mu(\xi_1, \xi_2) d\ell, \quad i = 1, 2, \dots, m.$$

Assume that μ is a constant x_j in pixel j . This leads to:

$$b_i = \sum_{j \sim \text{ray}_i} a_{ij} x_j, \quad a_{ij} = \text{length of ray}_i \text{ in pixel } j,$$

where the sum is over those pixels j that are intersected by ray_i .

Define $a_{ij} = 0$ for those pixels *not* intersected by ray_i . Then we have a simple sum

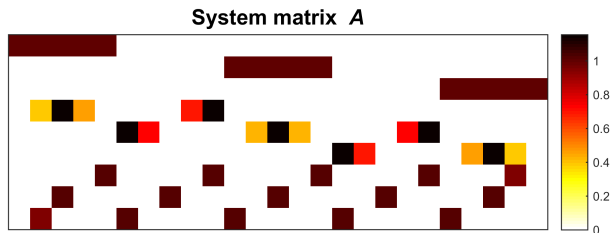
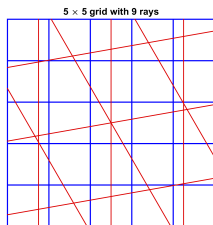
$$b_i = \sum_{j=1}^n a_{ij} x_j, \quad n = \text{number of pixels.}$$

A Big and Sparse System

If we collect all m equations then we arrive at a system of linear equations

$$Ax = b$$

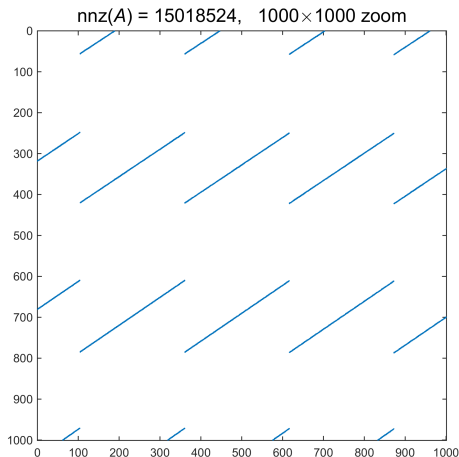
with a very sparse system matrix A . Example: 5×5 pixels and 9 rays:



Flexibility: We only set up equations for the data that we actually have. In case of missing data, e.g., for certain projection angles or certain rays in a projection, we just omit those from the linear system.

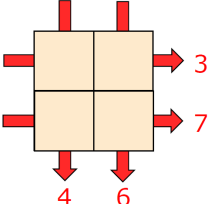
The System Matrix is Very Sparse

Another example: 256×256 pixels and 180 projections with 362 rays each.
The system matrix \mathbf{A} is $65,160 \times 65,536$ and has $\approx 4.27 \cdot 10^9$ elements.
There are 15,018,524 nonzero elements corresponding to a fill of 0.35%.



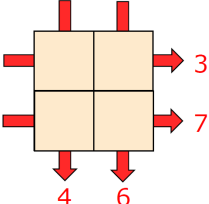
A "Sudoku" Problem

Four unknowns, four rays \rightarrow system of linear equations $\mathbf{A} \mathbf{x} = \mathbf{b}$:

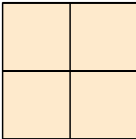

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 3 \\ 7 \\ 4 \\ 6 \end{pmatrix}$$

A "Sudoku" Problem

Four unknowns, four rays \rightarrow system of linear equations $\mathbf{A} \mathbf{x} = \mathbf{b}$:


$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 3 \\ 7 \\ 4 \\ 6 \end{pmatrix}$$

Unfortunately there are infinitely many solutions, with $k \in \mathbb{R}$:

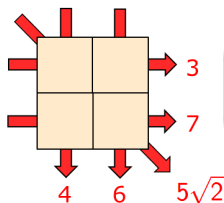

$$= \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} + k \times \begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix}$$

(There is an arbitrary component in the null space of the matrix \mathbf{A} .)

More Data Gives a Unique Solution

With *enough* rays the problem has a unique solution.

Here, one more ray is enough to ensure a full-rank matrix:


$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ \sqrt{2} & 0 & 0 & \sqrt{2} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 3 \\ 7 \\ 4 \\ 6 \\ 5\sqrt{2} \end{pmatrix}$$

In practise, algebraic iterative reconstruction methods do not care if the matrix \mathbf{A} is rank deficient.

Algebraic Reconstruction Methods

- In principle, all we need to do in the algebraic formulation is to solve the large sparse linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$:

$$\text{Math: } \mathbf{x} = \mathbf{A}^{-1}\mathbf{b},$$

$$\text{MATLAB: } \mathbf{x} = \mathbf{A} \setminus \mathbf{b}.$$

How hard can that be?

- Actually, this can be a formidable task if we try to use a traditional approach such as Gaussian elimination.
- Researchers in tomography have therefore focused on the use of *iterative solvers* – and they have rediscovered many methods developed by mathematicians . . .
- In tomography they are called **algebraic reconstruction methods**. They are much more flexible than FBP with respect to measurement geometry and constraints, but at a higher computational cost!

Some Algebraic Reconstruction Methods

Fully Sequential Methods

- Kaczmarz's method + variants.
- These are row-action methods: they update the solution using one row of \mathbf{A} at a time (there are also column versions, which we do not cover here).
- “Fast” convergence.

Fully Simultaneous Methods

- Landweber, Cimmino, CAV, DROP, SART, SIRT, ...
- These methods use all the rows of \mathbf{A} simultaneously in one iteration (i.e., they are based on matrix multiplications).
- Slower convergence.

Block Methods

- These methods combine the best properties of the above two classes.
- They are not covered today, but quite popular in applications.

Matrix Notation & Interpretation

Notation:

$$\mathbf{A} = \left(\begin{array}{c|c|c|c} | & | & \cdots & | \\ \mathbf{c}_1 & \mathbf{c}_2 & & \mathbf{c}_n \\ | & | & & | \end{array} \right) = \left(\begin{array}{c|c|c} \text{---} & \mathbf{r}_1^T & \text{---} \\ & \vdots & \\ \text{---} & \mathbf{r}_m^T & \text{---} \end{array} \right),$$

The matrix \mathbf{A} maps the discretized absorption coefficients (the vector \mathbf{x}) to the data in the detector pixels (the elements of the vector \mathbf{b}) via:

$$\mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix} = \mathbf{A} \mathbf{x} = \underbrace{x_1 \mathbf{c}_1 + x_2 \mathbf{c}_2 + \cdots + x_n \mathbf{c}_n}_{\text{linear combination of columns}} = \begin{pmatrix} \mathbf{r}_1^T \mathbf{x} \\ \mathbf{r}_2^T \mathbf{x} \\ \vdots \\ \mathbf{r}_m^T \mathbf{x} \end{pmatrix}.$$

The i th row of \mathbf{A} maps \mathbf{x} to detector element i (through the i th ray) via the **inner product**:

$$b_i = \mathbf{r}_i^T \mathbf{x} = \sum_{j=1}^n a_{ij} x_j, \quad i = 1, 2, \dots, m.$$

Example of Column Interpretation

A 32×32 image has four nonzero pixels with intensities 1, 0.8, 0.6, 0.4. In the vector \mathbf{x} these four pixels correspond to entries 468, 618, 206, 793. Hence the sinogram, represented as a vector \mathbf{b} , takes the form

$$\mathbf{b} = 0.6 \mathbf{c}_{206} + 1.0 \mathbf{c}_{468} + 0.8 \mathbf{c}_{618} + 0.4 \mathbf{c}_{793}.$$



Geometric Interpretation of $\mathbf{Ax} = \mathbf{b}$

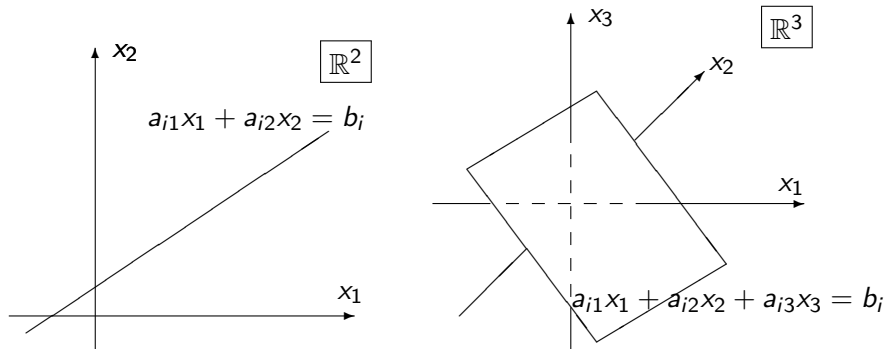
$$\mathbf{r}_1^T \mathbf{x} = a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1$$

$$\mathbf{r}_2^T \mathbf{x} = a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2$$

$$\vdots$$

$$\mathbf{r}_m^T \mathbf{x} = a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m.$$

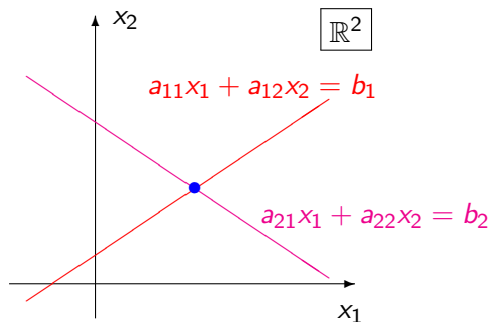
Each equation $\mathbf{r}_i^T \mathbf{x} = b_i$ defines an *affine hyperplane* in \mathbb{R}^n :



Geometric Interpretation of the Solution

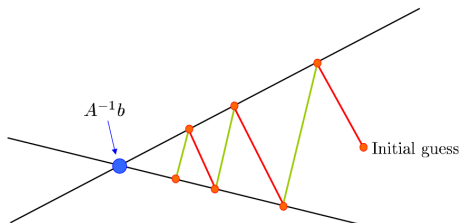
Assuming that the solution to $\mathbf{A}\mathbf{x} = \mathbf{b}$ is unique, it is the point $\mathbf{x} \in \mathbb{R}^m$ where all the m affine hyperplanes intersect.

Example with $m = n = 2$:



Kaczmarz's Method = Algebraic Reconstruction Technique

A simple iterative method based on the geometric interpretation.

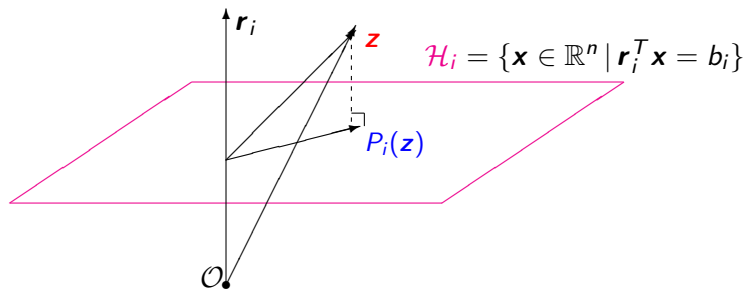


In each iteration, and in a *cyclic fashion*, compute the new iteration vector such that precisely one of the equations is satisfied.

This is achieved by projecting the current iteration vector \mathbf{x} on one of the hyperplanes $\mathbf{r}_i^T \mathbf{x} = b_i$ for $i = 1, 2, \dots, m, 1, 2, \dots, m, 1, 2, \dots$

Originally proposed in 1937, and independently suggested under the name **ART** by Gordon, Bender, Herman in 1970 for tomographic reconstruction.

Orthogonal Projection on Affine Hyperplane



The orthogonal projection $P_i(z)$ of an arbitrary point z on the affine hyperplane \mathcal{H}_i defined by $r_i^T x = b_i$ is given by:

$$P_i(z) = z + \frac{b_i - r_i^T z}{\|r_i\|_2^2} r_i, \quad \|r_i\|_2^2 = r_i^T r_i.$$

In words, we scale the row vector r_i by $(b_i - r_i^T z)/\|r_i\|_2^2$ and add it to z .

Kaczmarz's Method

We thus obtain the following algebraic formulation:

Basic Kaczmarz algorithm

$\mathbf{x}^{(0)}$ = initial vector

for $k = 0, 1, 2, \dots$

$i = k \pmod{m}$

$$\mathbf{x}^{(k+1)} = P_i(\mathbf{x}^{(k)}) = \mathbf{x}^{(k)} + \frac{b_i - \mathbf{r}_i^T \mathbf{x}^{(k)}}{\|\mathbf{r}_i\|_2^2} \mathbf{r}_i$$

end

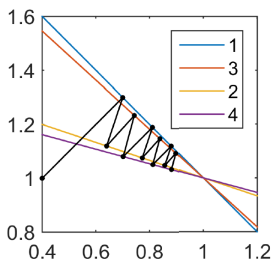
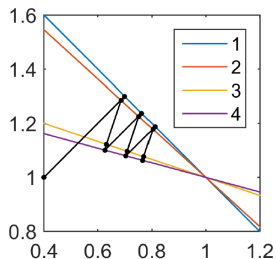
Each time we have performed m iterations of this algorithm, we have performed one *sweep* over the rows of \mathbf{A} . Other choices of sweeps:

- **Symmetric Kaczmarz:** $i = 1, 2, \dots, m-1, m, m-1, \dots, 3, 2$.
- **Randomized Kaczmarz:** select row i randomly.

Convergence Issues

The convergence of Kaczmarz's method is quite obvious from the graph on slide 18 – but can we say more?

Difficulty: the *ordering* of the rows of \mathbf{A} influences the *convergence rate*.



$$\begin{pmatrix} 1.0 & 1.0 \\ 1.0 & 1.1 \\ 1.0 & 3.0 \\ 1.0 & 3.7 \end{pmatrix} \mathbf{x} = \begin{pmatrix} 2.0 \\ 2.1 \\ 4.0 \\ 4.7 \end{pmatrix}$$

The ordering 1–3–2–4 is preferable: almost twice as fast.

Convergence of Kaczmarz's Method

One way to avoid the difficulty associated with influence of the ordering of the rows is to assume that we *select the rows randomly*.

For simplicity, assume that \mathbf{A} is invertible and that all rows of \mathbf{A} are scaled to unit 2-norm. Then the expected value $\mathcal{E}(\cdot)$ of the error norm satisfies:

$$\mathcal{E}\left(\|\mathbf{x}^{(k)} - \bar{\mathbf{x}}\|_2^2\right) \leq \left(1 - \frac{1}{n\kappa^2}\right)^k \|\mathbf{x}^{(0)} - \bar{\mathbf{x}}\|_2^2, \quad k = 1, 2, \dots,$$

where $\bar{\mathbf{x}} = \mathbf{A}^{-1}\mathbf{b}$ and $\kappa = \|\mathbf{A}\|_2 \|\mathbf{A}^{-1}\|_2$. This is **linear convergence**.

When κ is large we have

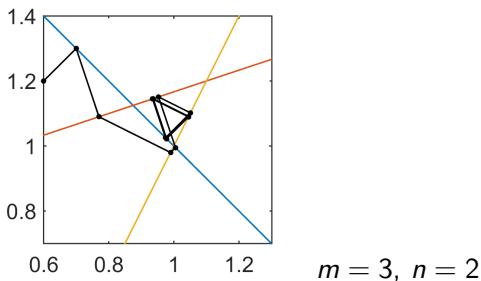
$$\left(1 - \frac{1}{n\kappa^2}\right)^k \approx 1 - \frac{k}{n\kappa^2}.$$

After $k = n$ steps, one *sweep* of the rows, the reduction factor is $1 - 1/\kappa^2$. Note that there are often orderings for which the convergence is faster!

Cyclic Convergence

So far we have assumed that there is a unique solution $\bar{\mathbf{x}} = \mathbf{A}^{-1}\mathbf{b}$ that satisfies $\mathbf{A}\mathbf{x} = \mathbf{b}$, i.e., all the affine hyperplanes associated with the rows of \mathbf{A} intersect in a single point.

What happens when this is not true? \rightarrow *cyclic convergence*:

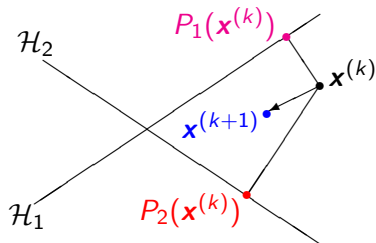


Kaczmarz's method can be brought to converge to a unique point, and we will discuss the modified algorithm shortly.

From Sequential to Simultaneous Updates

Karzmarz's method accesses the rows sequentially. **Cimmino's method** accesses the rows *simultaneously* and computes the **next iteration vector** as the average of the all the projections of the previous iteration vector:

$$\begin{aligned}\mathbf{x}^{(k+1)} &= \frac{1}{m} \sum_{i=1}^m P_i(\mathbf{x}^{(k)}) = \frac{1}{m} \sum_{i=1}^m \left(\mathbf{x}^{(k)} + \frac{b_i - \mathbf{r}_i^T \mathbf{x}^{(k)}}{\|\mathbf{r}_i\|_2^2} \mathbf{r}_i \right) \\ &= \mathbf{x}^{(k)} + \frac{1}{m} \sum_{i=1}^m \frac{b_i - \mathbf{r}_i^T \mathbf{x}^{(k)}}{\|\mathbf{r}_i\|_2^2} \mathbf{r}_i.\end{aligned}$$



Matrix formulation of Cimmino's Method

We can write the updating in our matrix-vector formalism as follows

$$\begin{aligned}\mathbf{x}^{(k+1)} &= \mathbf{x}^{(k)} + \frac{1}{m} \sum_{i=1}^m \frac{b_i - \mathbf{r}_i^T \mathbf{x}^{(k)}}{\|\mathbf{r}_i\|_2^2} \mathbf{r}_i \\ &= \mathbf{x}^{(k)} + \frac{1}{m} \begin{pmatrix} \frac{\mathbf{r}_1}{\|\mathbf{r}_1\|_2^2} & \cdots & \frac{\mathbf{r}_m}{\|\mathbf{r}_m\|_2^2} \end{pmatrix} \begin{pmatrix} b_1 - \mathbf{r}_1^T \mathbf{x}^{(k)} \\ \vdots \\ b_m - \mathbf{r}_m^T \mathbf{x}^{(k)} \end{pmatrix} \\ &= \mathbf{x}^{(k)} + \frac{1}{m} \begin{pmatrix} \mathbf{r}_1^T \\ \vdots \\ \mathbf{r}_m^T \end{pmatrix}^T \begin{pmatrix} \|\mathbf{r}_1\|_2^{-2} & & \\ & \ddots & \\ & & \|\mathbf{r}_m\|_2^{-2} \end{pmatrix} \left(\mathbf{b} - \begin{pmatrix} \mathbf{r}_1^T \\ \vdots \\ \mathbf{r}_m^T \end{pmatrix} \mathbf{x}^{(k)} \right) \\ &= \mathbf{x}^{(k)} + \mathbf{A}^T \mathbf{M} (\mathbf{b} - \mathbf{A} \mathbf{x}^{(k)}),\end{aligned}$$

where we introduced the diagonal matrix $\mathbf{M} = \text{diag}\left(\frac{1}{m\|\mathbf{r}_i\|_2^2}\right)$.

Cimmino's Method

We thus obtain the following formulation:

Basic Cimmino algorithm

$\mathbf{x}^{(0)}$ = initial vector

for $k = 0, 1, 2, \dots$

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \mathbf{A}^T \mathbf{M}(\mathbf{b} - \mathbf{A} \mathbf{x}^{(k)})$$

end

Note that one iteration here involves all the rows of \mathbf{A} , while one iteration in Kaczmarz's method involves a single row.

Therefore, the computational work in one Cimmino iteration is equivalent to m iterations (a sweep over all the rows) in Kaczmarz's basic algorithm.

The issue of finding a good row ordering is, of course, absent from Cimmino's method.

Convergence Study

Assume $\mathbf{x}^{(0)} = \mathbf{0}$ and let \mathbf{I} denote the $n \times n$ identity matrix; then:

$$\begin{aligned}\mathbf{x}^{(k+1)} &= \sum_{j=0}^k (\mathbf{I} - \mathbf{A}^T \mathbf{M} \mathbf{A})^j \mathbf{A}^T \mathbf{M} \mathbf{b} \\ &= \left(\mathbf{I} - (\mathbf{I} - \mathbf{A}^T \mathbf{M} \mathbf{A})^{k+1} \right) (\mathbf{A}^T \mathbf{M} \mathbf{A})^{-1} \mathbf{A}^T \mathbf{M} \mathbf{b}.\end{aligned}$$

If \mathbf{A} is invertible then

$$(\mathbf{A}^T \mathbf{M} \mathbf{A})^{-1} \mathbf{A}^T \mathbf{M} \mathbf{b} = \mathbf{A}^{-1} \mathbf{M}^{-1} \mathbf{A}^{-T} \mathbf{A}^T \mathbf{M} \mathbf{b} = \mathbf{A}^{-1} \mathbf{b}.$$

Moreover, the largest eigenvalue of the symmetric matrix $\mathbf{I} - \mathbf{A}^T \mathbf{M} \mathbf{A}$ is strictly smaller than one, and therefore

$$\left(\mathbf{I} - (\mathbf{I} - \mathbf{A}^T \mathbf{M} \mathbf{A})^{k+1} \right) \rightarrow \mathbf{I} \quad \text{for} \quad k \rightarrow \infty.$$

Hence the iterates $\mathbf{x}^{(k)}$ converge to the solution $\bar{\mathbf{x}} = \mathbf{A}^{-1} \mathbf{b}$.

Convergence of Cimmino's Method

To simplify the result, assume that \mathbf{A} is invertible and that the rows of \mathbf{A} are scaled such that $\|\mathbf{A}\|_2^2 = m$. Then

$$\|\mathbf{x}^{(k)} - \bar{\mathbf{x}}\|_2^2 \leq \left(1 - \frac{2}{1 + \kappa^2}\right)^k \|\mathbf{x}^{(0)} - \bar{\mathbf{x}}\|_2^2$$

where $\bar{\mathbf{x}} = \mathbf{A}^{-1}\mathbf{b}$, $\kappa = \|\mathbf{A}\|_2 \|\mathbf{A}^{-1}\|_2$, and we have **linear convergence**.

When $\kappa \gg 1$ then we have the approximate upper bound

$$\|\mathbf{x}^{(k)} - \bar{\mathbf{x}}\|_2^2 \lesssim (1 - 2/\kappa^2)^k \|\mathbf{x}^{(0)} - \bar{\mathbf{x}}\|_2^2,$$

showing that in each iteration the error is reduced by a factor $1 - 2/\kappa^2$.

This is almost the same factor as in one sweep through the rows of \mathbf{A} in Kaczmarz's method.

Incorporating Simple Constraints

We can include constraints on the elements of the reconstructed image.

Assume that we can write the constraint as $\mathbf{x} \in \mathcal{C}$, where \mathcal{C} is a convex set; this includes two very common special cases:

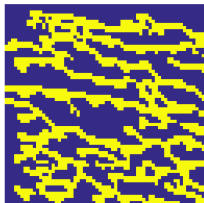
Non-negativity constraints. The set $\mathcal{C} = \mathbb{R}_+^n$ corresponds to

$$x_i \geq 0, \quad i = 1, 2, \dots, n.$$

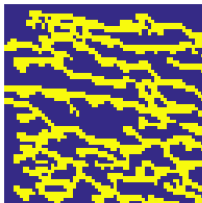
Box constraints. The set $\mathcal{C} = [0, 1]^n$ (n -dimensional box) corresponds to

$$0 \leq x_i \leq 1, \quad i = 1, 2, \dots, n.$$

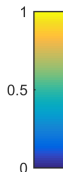
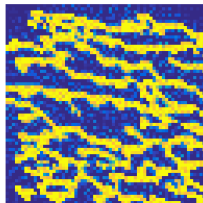
Ground truth



Box constraints

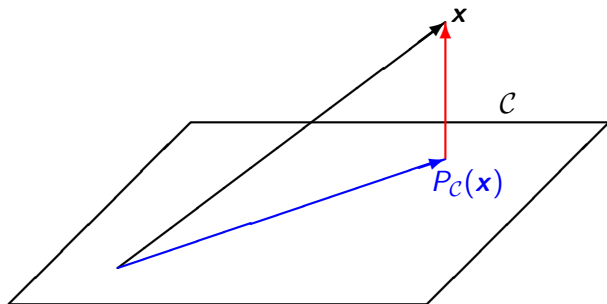


No constraints



Orthogonal Projections

Given a set \mathcal{C} , the orthogonal projection $P_{\mathcal{C}}(\mathbf{x})$ of an arbitrary vector $\mathbf{x} \in \mathbb{R}^n$ on \mathcal{C} is the unique vector that satisfies: $P_{\mathcal{C}}(\mathbf{x}) \perp (\mathbf{x} - P_{\mathcal{C}}(\mathbf{x}))$.



If $\mathcal{C} = \mathbb{R}_+^n$ (non-negativity constraints) then, in MATLAB, we compute the corresponding projection of \mathbf{x} as $\max(\mathbf{x}, 0)$.

The Projected Algorithms

Both algorithms below solve $\min_{\mathbf{x} \in \mathcal{C}} \|\mathbf{M}^{1/2}(\mathbf{b} - \mathbf{A}\mathbf{x})\|_2$.

Projected gradient algorithm ($\omega_k < 2/\|\mathbf{A}^T \mathbf{M} \mathbf{A}\|_2$)

$\mathbf{x}^{(0)}$ = initial vector

for $k = 0, 1, 2, \dots$

$$\mathbf{x}^{(k+1)} = P_{\mathcal{C}}(\mathbf{x}^{(k)} + \omega_k \mathbf{A}^T \mathbf{M}(\mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}))$$

end

Projected incremental gradient (Kaczmarz) algorithm ($\omega_k < 2$)

$\mathbf{x}^{(0)}$ = initial vector

for $k = 0, 1, 2, \dots$

$$i = k \pmod{m}$$

$$\mathbf{x}^{(k+1)} = P_{\mathcal{C}}\left(\mathbf{x}^{(k)} + \omega_k \frac{b_i - \mathbf{r}_i^T \mathbf{x}}{\|\mathbf{r}_i\|_2^2} \mathbf{r}_i\right)$$

end

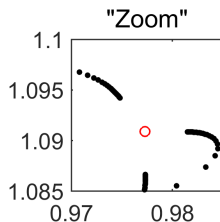
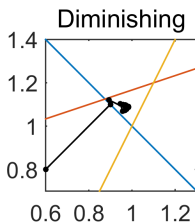
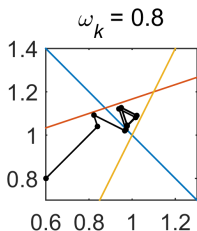
Iteration-Dependent Relaxation Parameter ω_k

To avoid cyclic and non-convergent behavior, we introduce relaxation:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \omega_k \frac{b_i - \mathbf{r}_i^T \mathbf{x}^{(k)}}{\|\mathbf{r}_i\|_2^2}.$$

Consider the example from slide 23 with:

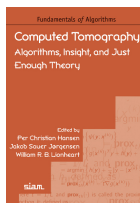
$$\omega_k = 0.8 \text{ (independent of } k) \quad \text{and} \quad \omega_k = 1/\sqrt{k}, \quad k = 0, 1, 2, \dots$$



- With a fixed $\omega_k < 1$ we still have a cyclic non-convergent behavior.
- With the *diminishing relaxation parameter* $\omega_k = 1/\sqrt{k} \rightarrow 0$ as $k \rightarrow \infty$ the iterates converge to the weighted least squares solution $\mathbf{x}_{LS,M}$.

A Few References

- P. C. Hansen and J. S. Jørgensen, *AIR Tools II: algebraic iterative reconstruction methods, improved implementation*, Numer. Algo., 79 (2018), pp. 107–137. github.com/jakobsj/AIRToolsII
- P. C. Hansen, J. S. Jørgensen and W. R. B. Lionheart (Eds.), *Computed Tomography: Algorithms, Insight, and Just Enough Theory*, SIAM, Philadelphia, 2021.



- G. T. Herman, *Fundamentals of Computerized Tomography – Image Reconstruction from Projections*, Springer, New York, 2009.
- A. C. Kak and M. Slaney, *Principles of Computerized Tomographic Imaging*, IEEE Press, 1988.