**Introduction to Medical Imaging**

**Iterative Reconstruction Methods**

Klaus Mueller

Computer Science Department
Stony Brook University

---

**Ideal Assumptions**

- Dense and regular sampling of the Fourier domain → many projections
- Noise free projections
- Straight rays

---

**Non-Ideal Scenarios**

- Projections might be:
  - sparse
  - acquired over less than 180°
  - noisy

- Rays might be non-linear (curved, refracted, scattered, …)
  - for example: refraction in ultrasound imaging

---

**Dealing With Non-Ideal Scenarios**

Iterative methods are advantageous in these cases

They can handle:

- limited number of projections
- irregularly-spaced and -angled projections
- non-straight ray paths (example: refraction in ultrasound imaging)
- corrective measures during reconstruction (example: metal artifacts)
- presence of statistical (Poisson) noise and scatter (mainly in functional imaging: SPECT, PET)

20 projections
SNR=10
low-dose CT
high-dose CT

low-dose CT
high-dose CT
**Specifics**

In medical imaging:
- $M$ unknown voxels (depending on desired object resolution)
- $N$ known measurements (pixels in the projection images)
- represent voxels and pixels as vectors $V$ and $P$, respectively

\[
\begin{align*}
W_{11}v_1 + W_{12}v_2 + \ldots + W_{1M}v_M &= p_1 \\
W_{21}v_1 + W_{22}v_2 + \ldots + W_{2M}v_M &= p_2 \\
\vdots & \\
W_{N1}v_1 + W_{N2}v_2 + \ldots + W_{NM}v_M &= p_N
\end{align*}
\]

- this gives rise to a system $WV=P$

---

**Solving for $V$**

The obvious solution is then:
- compute $V = W^{-1} \cdot P$

The main problem with this direct approach:
- $P$ is not be consistent due to noise $\rightarrow$ lines do not intersect in solution
- This turns $WV=P$ into an optimization problem

2D case

\[
\begin{align*}
W_{11}v_1 + W_{12}v_2 &= p_1 \\
W_{21}v_1 + W_{22}v_2 &= p_2 \\
W_{31}v_1 + W_{32}v_2 &= p_3
\end{align*}
\]

---

**Optimization Algorithms**

**Algebraic methods**
- Algebraic Reconstruction Technique (ART), SART, SIRT
- Projection Onto Convex Sets (POCS)

**Sparse system solvers**
- Gradient Descent (GD), Conjugate Gradients (CG)
- Gauss-Seidel

**Statistical methods**
- Expectation Maximization (EM)
- Maximum Likelihood Estimation (MLE)

All of these are *iterative* methods:
- predict $\rightarrow$ compare $\rightarrow$ correct $\rightarrow$ predict $\rightarrow$ compare $\rightarrow$ correct …

---

**Big Picture: Iterative Reconstruction**

Before delving into details, let’s see an iterative scheme at work.
Iterative Reconstruction Demonstration: SART

Foundations: Vectors

Consider two vectors, \( a \) and \( b \)

\[
\begin{align*}
\mathbf{a} &= [a_1, a_2], & |\mathbf{a}| &= \sqrt{a_1^2 + a_2^2} \\
\mathbf{b} &= [b_1, b_2], & |\mathbf{b}| &= \sqrt{b_1^2 + b_2^2}
\end{align*}
\]

Foundations: Scalar Projection

Scalar projection of \( \mathbf{a} \) onto \( \mathbf{b} \):

\[
|\mathbf{a}| \cos \alpha = \frac{\mathbf{a} \cdot \mathbf{b}}{|\mathbf{b}|}
\]

The dot product:

\[
\mathbf{a} \cdot \mathbf{b} = \mathbf{a} \cdot \mathbf{b}^T = [a_1, a_2] \cdot [b_1, b_2]^T = a_1b_1 + a_2b_2 = |\mathbf{a}| |\mathbf{b}| \cos \alpha
\]

→ the scalar projection is the dot product with \( |\mathbf{b}| = 1 \) (unit vector)

\[
|\mathbf{b}| = \sqrt{b_1^2 + b_2^2} = 1
\]
Foundations: Line Equation

\[ a_1 x_1 + a_2 x_2 = y \]
\[ \|a\| = \sqrt{a_1^2 + a_2^2} = 1 \]

The vector \( a \) is the unit vector normal to the line \( l_a \).
The length \( y \) is the perpendicular distance of \( l_a \) to the origin.

For any point \( x \):
- If \( x \) is on \( l_a \) then the scalar projection of \( x \) onto \( a \) will be:
  \[ x \cdot a = y \]

Foundations: Distance From Line

For any other point \( x' \) not on \( l_a \) the scalar projection of \( x' \) onto \( a \) will be:
\[ x'^\cdot a = y' = y + \Delta y \]

Foundations: Closest Point

The closest point to \( x' \) on \( l_a \) is \( x'' \), computed by:
\[ x'' = x' - \Delta y \]
\[ = x' - (x'^\cdot a - y) \]
\[ = x' + (y - x'^\cdot a) \]

Foundations: Solving an Equation System

Assume you have two equations to solve for solution point \( x_s = (x_1, x_2) \):
- The intersection of the two lines

\[ a_{11} x_1 + a_{12} x_2 = y_1 \]
\[ a_{21} x_1 + a_{22} x_2 = y_2 \]
Foundations: Iterating to Solution
Of course, you could solve this equation via Gaussian elimination
• we shall take an iterative approach instead
Start with some point \(x^{(0)} = (x_1, x_2)\)

Pick an equation (line, say \(l_2\)) and find the closest point to \(x^{(0)}\)
• use the approach outlined before
• this gives a new point \(x^{(1)}\)

Iteratively
• pick alternate equations (lines) and project
• the solution will converge towards \(x_s\)
• the more iterations the closer the convergence

Foundations: Extension to Higher Dimensions
Three dimensions:
• 3 equations with 3 unknowns

\(N\) dimensions:
• \(N\) equations with \(M\) unknowns
• \(M\) can be less or greater than \(N\)
• inconsistent (most often) or not
Specifics to Medical Imaging

In medical imaging:
- $M$ unknown voxels (depending on desired object resolution)
- $N$ known measurements (pixels in the projection images)
- represent voxels and pixels as vectors $V$ and $P$, respectively

\[
w_{11}v_1 + w_{12}v_2 + ... + w_{1M}v_M = p_1 \\
w_{21}v_1 + w_{22}v_2 + ... + w_{2M}v_M = p_2 \\
... \\
w_{N1}v_1 + w_{N2}v_2 + ... + w_{NM}v_M = p_N
\]

- this gives rise to a system $WV = P$

Iterate either by
- ray by ray (Algebraic Reconstruction Technique, ART)
- image by image (Simultaneous ART, SART)
- all data at once (SIRT)

Iterative Update Schedule: ART

- one pixel at a time
- Project
- Correct
- Backproject

Iterative Update Schedule: SART

- one projection at a time
- Project
- Correct
- Backproject

Iterative Update Schedule: SIRT

- all projections
- Project
- Correct
- Backproject
Iterative Reconstruction Demonstration: SART

Iteratively solves $W \cdot V = P$

$$p_i = \frac{\sum_j v_j^k w_{ij}}{\sum_j w_{ij}}$$

$$v_j^{k+1} = v_j^k + \lambda \frac{\sum_i p_i - \sum_l v_l^k w_{il}}{\sum_i w_{ij}}$$

Projection

Projection (into pixel)
Correction factor computation

Projection (into pixel)

Normalized at pixel $i$

$$v_j^{k+1} = v_j^k + \lambda \sum_i w_{ij}$$

Scanned pixel

Voxel normalization

Backprojection (into voxel)

Scanned pixel

Backprojection (into voxel)

Normalization at pixel $i$

$$v_j^{k+1} = v_j^k + \lambda \sum_i w_{ij}$$

New $(k+1)$ and previous $(k)$ values of voxel $j$

Normalization at voxel $j$

Voxel update

Scanned pixel

Normalization at voxel $j$

$$v_j^{k+1} = v_j^k + \lambda \sum_i w_{ij}$$

Voxel normalization
SART

Next projection

\[ \sum_i p_i \sum_l \frac{v_i^k w_{il}}{w_{ij}} \]

\[ v_j^{k+1} = v_j^k + \lambda \sum_i w_{ij} \]

Gradient Descent

Quadratic form of a vector:

\[ f(x) = \frac{1}{2} x^T A x - b^T x + c \]

- this equation is minimized when \( A x = b \)
- this occurs when \( f'(x) = 0 \)
- thus, minimizing the quadratic form will solve the reconstruction problem

Graph plot
Contour plot
Gradient plot

Steepest Descent

Start at an arbitrary point and slide down to the bottom of the parabola
- in practice this will be a hyper-parabola since \( x, b \) are high-dimensional
- choose the direction in which \( f \) decreases most quickly

\[ -f'(x^{(i)}) = b - A x^{(i)} \]

where \( x^{(i)} \) is the current (predicted) solution

- similar to ART but now looks at all equations simultaneously

Some basics:
- error: how far are we away from the solution
  \[ e^{(i)} = x^{(i)} - x \]
- residual: how far are we away from the correct value of \( b \)
  \[ r^{(i)} = b - A x^{(i)} \]
  \[ r^{(i)} = A e^{(i)} \]
  \[ r^{(i)} = -f'(x^{(i)}) \]

Figures from J. Shewchuk, UC Berkeley
**Steepest Descent**

Finding the right place to turn directions is called *line search*

\[ x_{(1)} = x_{(0)} + \alpha r_{(0)} \]

To find \( \alpha \) we can use the following requirements:

1. the new direction of \( r \) must be orthogonal to the previous:
   \[ r_{(1)}^T r_{(0)} = 0 \]
2. the residual at \( x_{(1)} \):
   \[ f'(x_{(1)}) = -r_{(1)} \]
3. after some math:
   \[ \alpha = \frac{r_{(0)}^T r_{(0)}}{r_{(0)}^T A r_{(0)}} \]

**Steepest Descent: Summary**

\[ r_{(i)} = b - Ax_{(i)} \]
\[ \alpha = \frac{r_{(i)}^T r_{(i)}}{r_{(i)}^T A r_{(i)}} \]
\[ x_{(i+1)} = x_{(i)} + \alpha r_{(i)} \]

**Shortcoming:**

- sub-optimal since some directions might be taken more than once
- this can be fixed by the method of Conjugant Gradients

**Conjugant Gradients**

Picks a set of *orthogonal* search directions \( d_{(0)}, d_{(1)}, d_{(2)}, \ldots \)

1. take exactly one step along each
2. stop at exactly the right length for each to line up evenly with \( x \)

\[ x_{(i+1)} = x_{(i)} + \alpha_{(i)} d_{(i)} \]

- to determine \( \alpha_{(i)} \) use the fact that \( e_{(i+1)} \) should be orthogonal to \( d_{(i)} \)

\[ d_{(i)}^T e_{(i+1)} = 0 \]
\[ d_{(i)}^T (e_{(i)} + \alpha d_{(i)}) = 0 \]
\[ \alpha_{(i)} = \frac{d_{(i)}^T e_{(i)}}{d_{(i)}^T d_{(i)}} \]

- however, this requires knowledge of \( e_{(i)} \) which we do not have

**Conjugant Gradients**

Solution:

- make the search direction \( A \)-orthogonal (or, *conjugate*)

\[ \alpha_{(i)} = \frac{d_{(i)}^T A e_{(i)}}{d_{(i)}^T A d_{(i)}} = \frac{d_{(i)}^T r_{(i)}}{d_{(i)}^T A d_{(i)}} \]

- \( A \) transforms a coordinate system such that two vectors are orthogonal

\[ d_{(i)}^T A d_{(j)} = 0 \quad i \neq j \]
**Conjugant Gradients**

All directions taken are mutually orthogonal
- each new residual is orthogonal to all the previous residuals and search directions
- each new search direction is constructed (from the residual) to be $A$-orthogonal to all the previous residuals and search directions

Each new search direction adds a new dimension to the traversed sub-space
- the solution is a projection into the sub-space explored so far
- so after $n$ steps the full space is built and the solution has been reached

\[
\begin{align*}
    d^{(0)} &= r^{(0)} = b - Ax^{(0)}, \\
    \alpha(i) &= \frac{r_i^T r_i}{d_i^T Ad_i}, \\
    x^{(i+1)} &= x^{(i)} + \alpha(i) d^{(i)}, \\
    r^{(i+1)} &= r^{(i)} - \alpha(i) Ad^{(i)}, \\
    \beta(i+1) &= \frac{r^{(i+1)}_i r^{(i+1)}_i}{r^{(i)}_i r^{(i)}_i}, \\
    d^{(i+1)} &= r^{(i+1)} + \beta(i+1) d^{(i)}.
\end{align*}
\]

**Statistical Techniques**

Algebraic/gradient methods do not model statistical effects in the underlying data
- this is OK for CT (within reason)

However, the emission of radiation from radionuclides is highly statistical
- the direction is chosen at random
- similar metabolic activities may not emit the same radiation
- not all radiation is actually collected (collimators reject many photons)
- in low-dose CT, noise is also a significant problem

Need a reconstruction method that can accounts for these statistical effects
- Maximum Likelihood – Expectation Maximization (ML-EM) is one such method

Also called the law of rare events
- it is the binomial distribution of $k$ as the number of trials $n$ goes to infinity

\[
\lim_{n \to \infty} P(X = k) = \lim_{n \to \infty} \binom{n}{k} p^k (1-p)^{n-k}
\]

- with $p = \lambda / n$

\[
f(k; \lambda) = \frac{e^{-\lambda} \lambda^k}{k!}
\]

$\lambda$: expected number of events (the mean) in a given time interval

Some examples for Poisson-distributed events:
- the number of phone calls at a call center per minute
- the number of spelling errors a secretary makes while typing a single page
- the number of soldiers killed by horse-kicks each year in each corps in the Prussian cavalry
- the number of positron emissions in a radio nucleotide in PET and SPECT
- the number of annihilation events in PET and SPECT
Overall Concept of ML-EM

There are three types of variables

#1: The observed data $y(d)$:
- the detector readings

#2: The unobserved (latent) data $x(b)$:
- the photon emission activities in the pixels (the tissue), $x(b)$
- these give rise to the detector readings
- they follow a Poisson distribution

#3: The model parameters $\lambda(b)$:
- these cause the emissions
- they are the metabolic activities (state) of interest
- the emissions only approximate those
  → they represent the expectations (means, $\lambda$) of the resulting Poisson distribution causing the readings at the detectors

Overall Concept of ML-EM

There is a many-to-one mapping of parameters $\rightarrow$ data

Since there is a many-to-one mapping, many objects are probable to have produced the observed data
- the object reconstruction (the image) having the highest such probability is the maximum likelihood estimate of the original object

Goal:
- estimate the model parameters using the observed data

Solution:
- EM will converge to a solution of maximum likelihood (but not necessarily the global maximum)

Maximum Likelihood Expectation Maximization (ML-EM)

Initialization step: choose an initial setting of the model parameters

Then proceed to EM, which has two steps, executed iteratively:
- E (expectation) step: estimate the unobserved data from the current estimate of the model parameters and the observed data
- M (maximization) step: compute the maximum-likelihood estimate of the model parameters using the estimated unobserved data

Stop when converged

Initialize model parameters $p$

E-Step: estimate unobserved data $x$ using $p$ and observed data $y$

M-Step: compute ML-estimate of $p$ using $x$

return if converged

After combining the E-step and the ML-step:

$$v_{jk}^{k+1} = \frac{v_{jk}^k}{\sum_i w_{ij}} \sum_j \frac{p_i}{\sum_j v_{ij}^k w_{ij}}$$
Maximum Likelihood Expectation Maximization (ML-EM)

Maximizes the likelihood of the values of (object) voxels $j$, given values at (detector) pixels $i$

New $(k+1)$ and previous $(k)$ values of voxel $j$

Backprojection (into voxel $j$)

Normalization at voxel $j$

Projection (into pixel $i$)

Algorithm Comparison

SART:
- projection ordering important
- ensure that consecutively selected projections are approximately orthogonal
- random selection works well in practice

CG:
- much depends on the condition number of the (system) matrix $A$
- various pre-conditioning methods exist in the literature
- also, line search can be expensive and inaccurate
- various methods and heuristics for line search have been described in the literature

EM:
- convergence slow if all projections are applied before voxel update
- use OS-EM (Ordered Subsets EM): only a subset of projections are applied per iteration

Inconsistent Equations

Real life data (as mentioned earlier)
- typically equations (the data) are not consistent
- you may have more equations (data) than unknowns or not enough
- solution falls within a convex shape spanned by the intersection set
- need further criteria to determine the true solution (some prior model)

Determining the True Solution

Need further criteria to determine the true solution

Use some prior model
- smoothness, approximate shape, sharp edges, …
- incorporate this model into the reconstruction procedure

Example:
- enforce smoothness by intermittent blurring
- but at the same time preserve edges

streak artifacts, good edges
smooth, good edges