Tomographic reconstruction
Lectures 2 and 3: Iterative reconstruction methods

Camille Pouchol[1]
pouchol@kth.se

Department of Mathematics
KTH, Stockholm

HL2027: 3D Image Reconstruction and Analysis in Medicine
January/Februrary 2019

[1]based on a previous lecture by Sebastian Banert, Ozan Öktem

Review from last lecture

The fully discretised reconstruction problem

Image reconstruction

Recover the digital image $\boldsymbol{\alpha}_{\text{true}} \in \mathbb{R}^n$ from measured data $g \in \mathbb{R}^m$ assuming

$$g = \mathbf{A} \cdot \boldsymbol{\alpha}_{\text{true}} + g_{\text{noise}}. \tag{1}$$

Here, $\mathbf{A}$ is the $(m \times n)$-measurement matrix and $g_{\text{noise}} \in \mathbb{R}^m$ is the noise component of data, $\boldsymbol{\alpha}_{\text{true}} = \langle f, \phi_j \rangle_X$.

Measurement matrix:

$$\mathbf{A} := \begin{pmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,n} \\ a_{2,1} & a_{2,2} & \dots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \dots & a_{m,n} \end{pmatrix}.$$

Transmission tomography and using the voxel basis:

$$a_{i,j} = \text{length of } j\text{:th line through } i\text{:th voxel.}$$

One can show that $\boldsymbol{\alpha}^{\dagger} \in \mathbb{R}^n$ solves

$$\min_{\boldsymbol{\alpha}} \|\mathbf{A} \cdot \boldsymbol{\alpha} - g\|_2^2 \tag{2}$$

if it solves the normal equations: $\mathbf{A}^{\mathrm{t}} \cdot (\mathbf{A} \cdot \boldsymbol{\alpha}^{\dagger} - g) = \mathbf{0}$.

- If $\mathbf{A}$ is the matrix representing the discretised ray transform, then $\mathbf{A}^{\mathrm{t}}$ (the transpose of $\mathbf{A}$) is the matrix representing the discretised backprojection.
- $\mathbf{A}^{\mathrm{t}} \cdot \mathbf{A}$ is not necessarily sparse even though $\mathbf{A}$ is.
- If there are infinitely many solutions to (2), then it is common to choose the one with least 2-norm.
- If columns of $\mathbf{A}$ are linearly independent, then $\mathbf{A}^{\mathrm{t}} \cdot \mathbf{A}$ is invertible and (2) has a unique solution.
- A solution to (2), there may be infinitely many, is often undesirable (overfitting).

Problem: A is huge and cannot be stored.

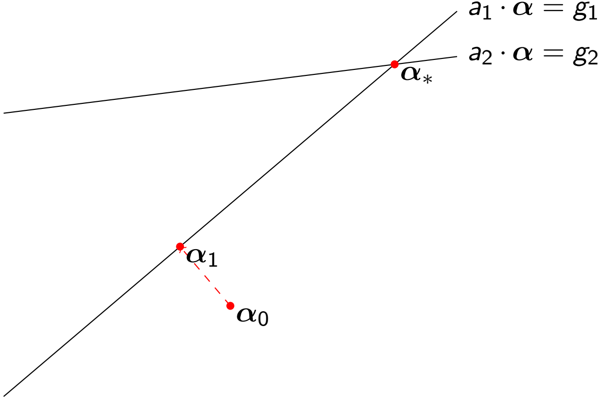
Stefan Kaczmarz (1895–1939), Polish mathematician active at University of Lviv (now in Ukraine).

Kaczmarz method: Solving linear systems of equations without needing to store the matrix.

- Introduced by Stefan Kaczmarz in 1937.
- Rediscovered in 1970 by Richard Gordon, Robert Bender, and Gabor Herman, then under the name algebraic reconstruction technique (ART).

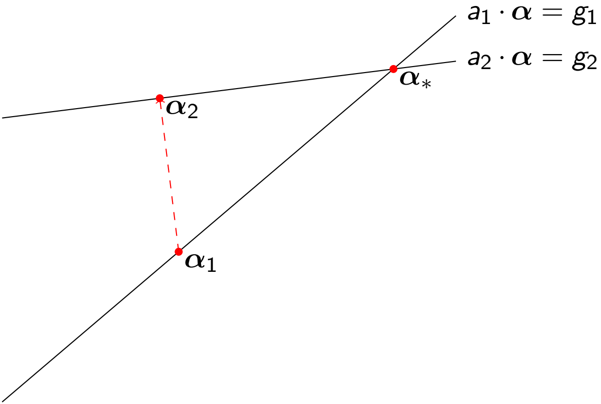
Reconstruction methods

Kaczmarz method: The case $n = m = 2$



$\boldsymbol{\alpha}_1 :=$ projection of $\boldsymbol{\alpha}_0$ into hyperplane $a_1 \cdot \boldsymbol{\alpha} = g_1$.

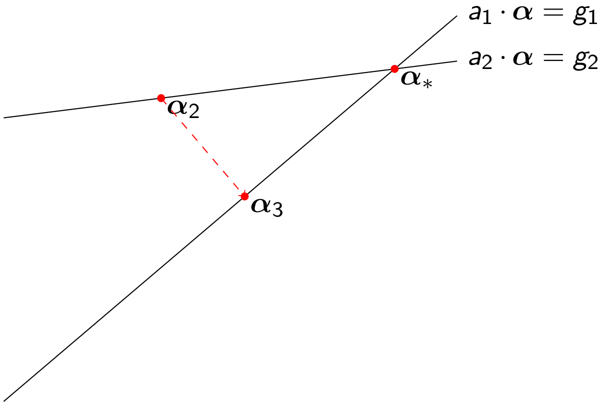
Reconstruction methods
Kaczmarz method: The case $n = m = 2$



$\boldsymbol{\alpha}_2 :=$ projection of $\boldsymbol{\alpha}_1$ into hyperplane $a_2 \cdot \boldsymbol{\alpha} = g_2$.

Reconstruction methods

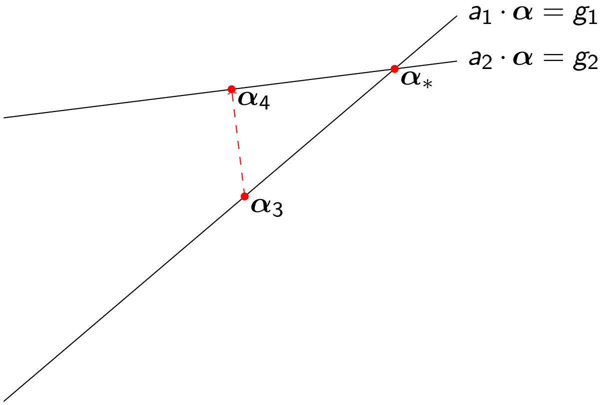
Kaczmarz method: The case $n = m = 2$



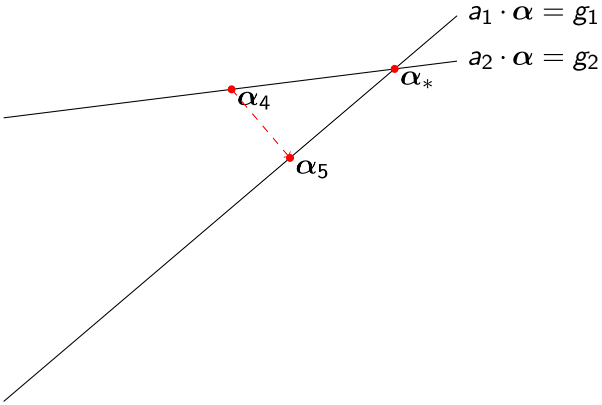
$\boldsymbol{\alpha}_3 :=$ projection of $\boldsymbol{\alpha}_2$ into hyperplane $a_1 \cdot \boldsymbol{\alpha} = g_1$.

Reconstruction methods

Kaczmarz method: The case $n = m = 2$



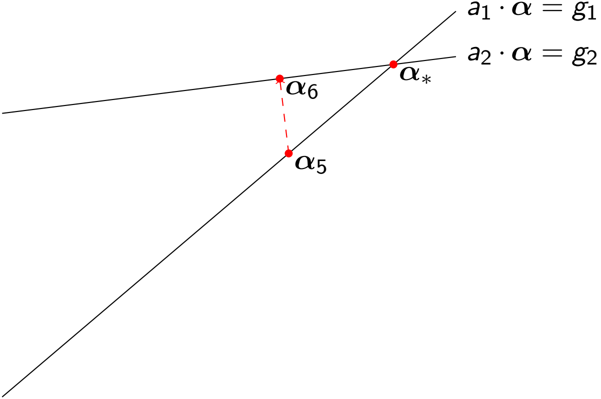
$\boldsymbol{\alpha}_4 :=$ projection of $\boldsymbol{\alpha}_3$ into hyperplane $a_2 \cdot \boldsymbol{\alpha} = g_2$.



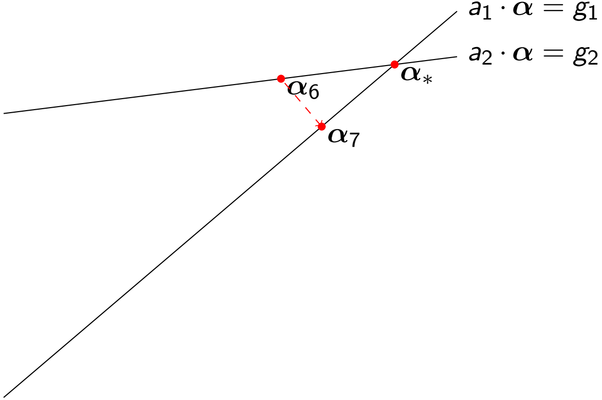
$\boldsymbol{\alpha}_5 :=$ projection of $\boldsymbol{\alpha}_4$ into hyperplane $a_1 \cdot \boldsymbol{\alpha} = g_1$.

Reconstruction methods

Kaczmarz method: The case $n = m = 2$



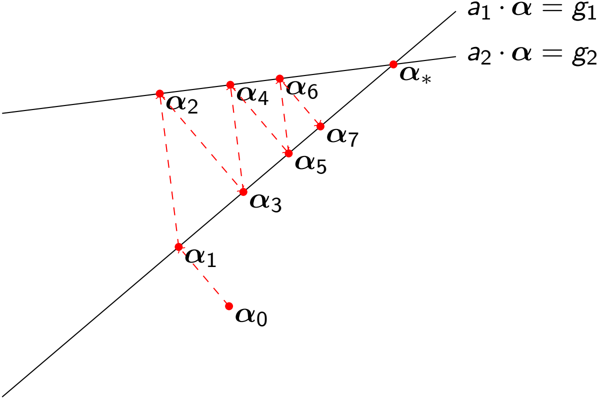
$\boldsymbol{\alpha}_6 :=$ projection of $\boldsymbol{\alpha}_5$ into hyperplane $a_2 \cdot \boldsymbol{\alpha} = g_2$.



$\boldsymbol{\alpha}_7 :=$ projection of $\boldsymbol{\alpha}_6$ into hyperplane $a_1 \cdot \boldsymbol{\alpha} = g_1$.

Reconstruction methods

Kaczmarz method: The case $n = m = 2$

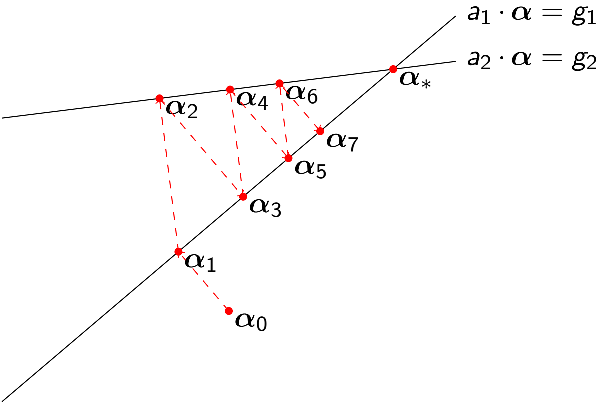


Iterative scheme: If $\pi_i(\boldsymbol{\alpha})$ denotes the projection of $\boldsymbol{\alpha}$ into the i:th hyperplane $a_i \cdot \boldsymbol{\alpha} = g_i$ ($i = 1, \ldots, m$), then

$$\boldsymbol{\alpha}_{k+1} := \pi_i(\boldsymbol{\alpha}_k) \quad \text{where } k \text{ sweeps through } 1, \ldots, m.$$

Reconstruction methods

Kaczmarz method: The case $n = m = 2$



Iterative scheme:

$$\boldsymbol{\alpha}_{k+1} := \boldsymbol{\alpha}_k + \frac{g_i - a_i \cdot \boldsymbol{\alpha}_k}{\|a_i\|_2^2} a_i, \quad \text{where } k \text{ sweeps through } 1, \ldots, m.$$

Orthogonal projection of a vector into a hyperplane.

- The vector a_i is orthogonal to the hyperplane $a_i \cdot \boldsymbol{\alpha} = g_i$.

  The equation $a_i \cdot \boldsymbol{\alpha} = g_i$ is the scalar equation of a hyperplane with a_i as the normal, so a_i is orthogonal to all vectors in the hyperplane $a_i \cdot \boldsymbol{\alpha} = g_i$.

- The orthogonal projection $\pi(x)$ of a vector $x \in \mathbb{R}^n$ onto $a_i \cdot \boldsymbol{\alpha} = g_i$ is found by subtracting a multiple of a_i from x:

$$\pi(x) = x - \gamma a_i \quad \text{for some } \gamma \in \mathbb{R}.$$

- γ must satisfy

$$g_i = \pi(x) \cdot a_i = (x - \gamma a_i) \cdot a_i = x \cdot a_i - \gamma a_i \cdot a_i.$$

- Solving this equation for γ gives

$$\gamma = \frac{a_i \cdot \boldsymbol{\alpha} - g_i}{a_i \cdot a_i} \quad \Longrightarrow \quad \pi(x) = x + \frac{g_i - a_i \cdot \boldsymbol{\alpha}}{\|a_i\|_2^2} a_i$$

- Convergence: let $\boldsymbol{\alpha}_k$ denotes the iterates generated by the Kaczmarz method for solving $\mathbf{A} \cdot \boldsymbol{\alpha} = g$.
  - If the linear system has a unique solution, then Kaczmarz iterates converge towards this solution.
  - For overdetermined systems, Kaczmarz method with initial vector $\boldsymbol{\alpha}_0 = \mathbf{0}$ converges to the least-squares solution.
  - For underdetermined systems, $\boldsymbol{\alpha}_k$ converges to the least-squares solution closest to the initial vector $\boldsymbol{\alpha}_0$ (Tanabe, 1971).
- Convergence rate: faster convergence when the angle between the hyperplanes is large. Extreme case is when hyperplanes are orthogonal, convergence in $m + 1$ steps.
- Speed-up: often not worth doing Gram-Schmidt orthogonalization. Instead, most efficient approach is to project on random hyperplanes.
- Need a balance between accurately computing $\mathbf{A}$ and the inconsistencies that result from crude approximations.

Examples of reconstructions

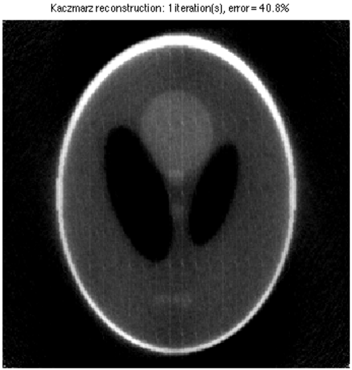
- Simulated parallel beam data of 2D phantom (true image).
- Relative error (in %) $:= 100 \cdot \dfrac{\|\boldsymbol{\alpha} - \boldsymbol{\alpha}_{\mathrm{true}}\|_2}{\|\boldsymbol{\alpha}_{\mathrm{true}}\|_2}$

Simulation protocol

- Phantom: 256×256 pixel 2D Shepp-Logan
  $n = 256 \cdot 256 = 65\,536$
- Data: Full angular range $[0^\circ, 180^\circ]$ with 1° step (180 directions) and 400 detector elements.
  $m = 180 \cdot 400 = 72\,000$.
- Noise component in data: Additive Gaussian noise with relative noise level 5%.

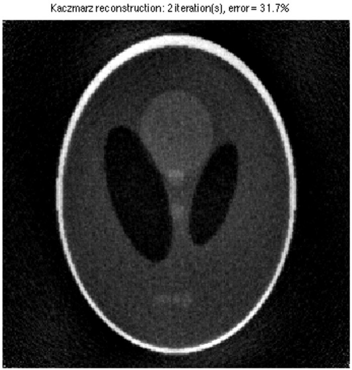
Overdetermined problem since $m > n$.

Reconstruction methods
Kaczmarz method: impact of number of iterations



$\#$ of iterations $= 1$
Relative error $= 40.8\%$

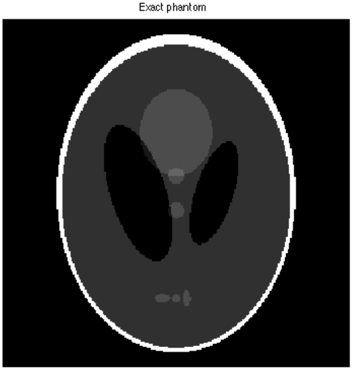
Reconstruction methods
Kaczmarz method: impact of number of iterations

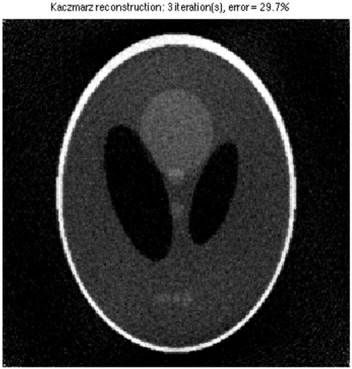


\# of iterations = 2
Relative error = 31.7%

Reconstruction methods

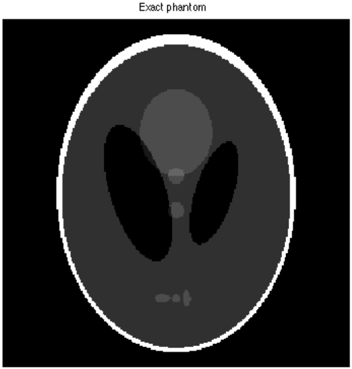
Kaczmarz method: impact of number of iterations

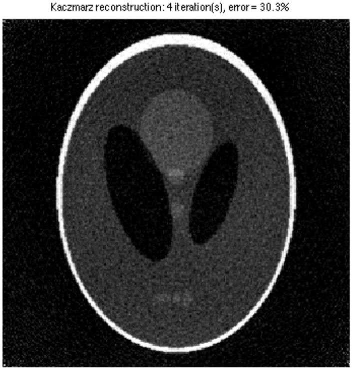




$\#$ of iterations $= 3$
Relative error $= 29.7\%$

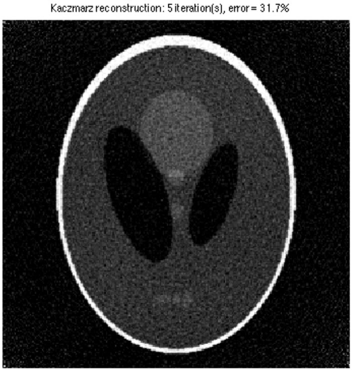
Reconstruction methods
Kaczmarz method: impact of number of iterations





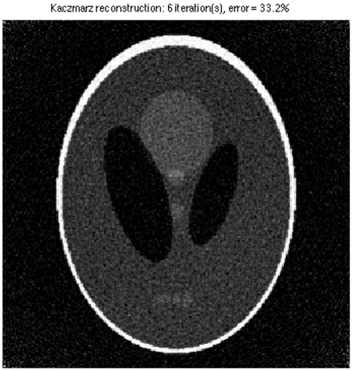
\# of iterations = 4
Relative error = 30.3%

Reconstruction methods
Kaczmarz method: impact of number of iterations



\# of iterations = 5
Relative error = 31.7%

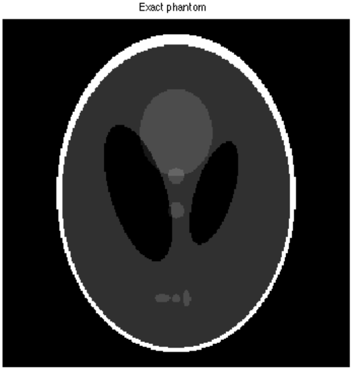
Reconstruction methods
Kaczmarz method: impact of number of iterations

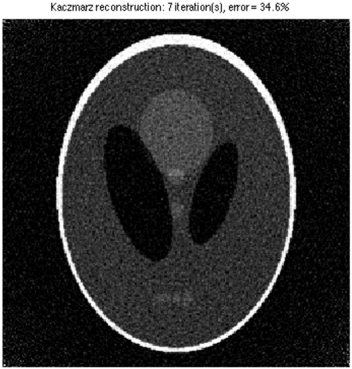


\# of iterations = 6
Relative error = 33.2%

Reconstruction methods

Kaczmarz method: impact of number of iterations

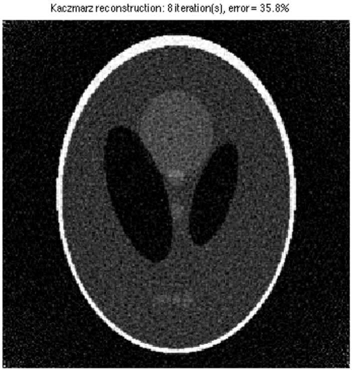




\# of iterations = 7
Relative error = 34.6%

Reconstruction methods

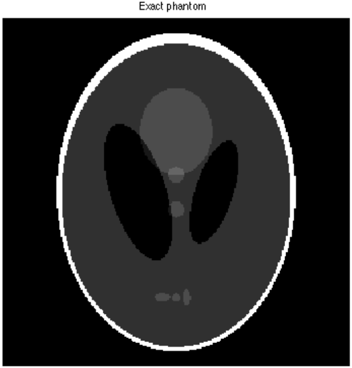
Kaczmarz method: impact of number of iterations

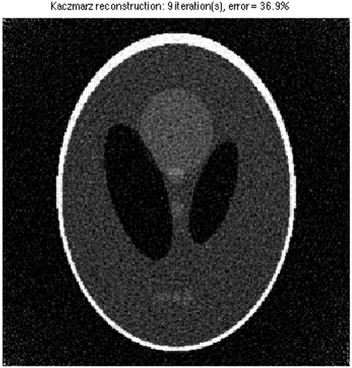


\# of iterations $= 8$
Relative error $= 35.8\%$

Reconstruction methods

Kaczmarz method: impact of number of iterations

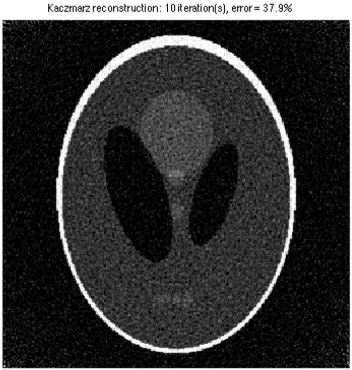




\# of iterations = 9
Relative error = 36.9%

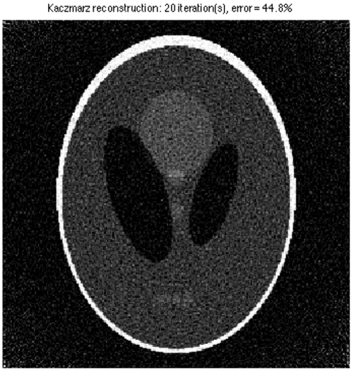
Reconstruction methods

Kaczmarz method: impact of number of iterations



\# of iterations $= 10$
Relative error $= 37.9\%$

Reconstruction methods
Kaczmarz method: impact of number of iterations



\# of iterations = 20
Relative error = 44.8%

Reconstruction methods

Kaczmarz method: impact of number of iterations

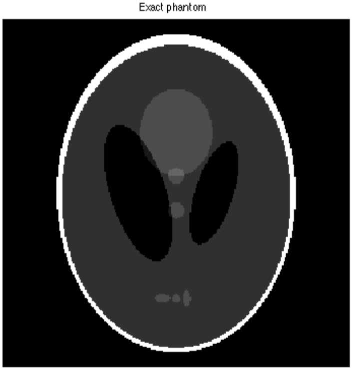
Exact phantom

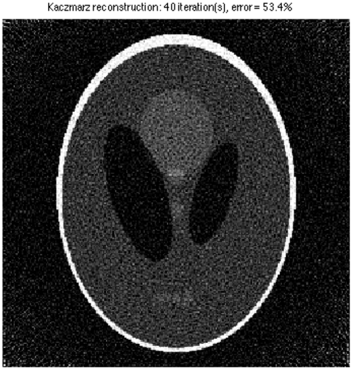
Kaczmarz reconstruction: 30 iteration(s), error = 49.5%

$\#$ of iterations $= 30$
Relative error $= 49.5\%$

Reconstruction methods

Kaczmarz method: impact of number of iterations

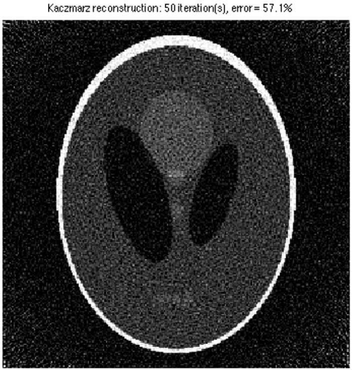




$\#$ of iterations $= 40$
Relative error $= 53.4\%$

Reconstruction methods

Kaczmarz method: impact of number of iterations



\# of iterations = 50
Relative error = 57.1%

Reconstruction methods

Kaczmarz method: impact of number of iterations

$\#$ of iterations $= 60$
Relative error $= 60.4\%$

Reconstruction methods

Kaczmarz method: impact of number of iterations

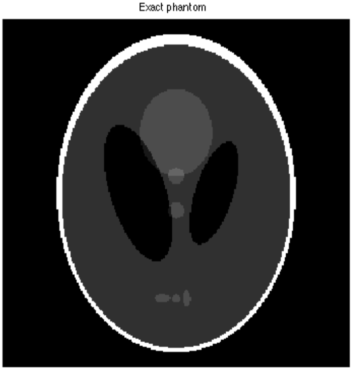
Exact phantom

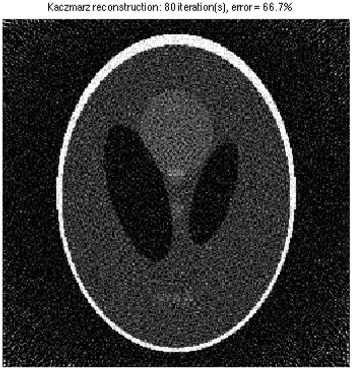
Kaczmarz reconstruction: 70 iteration(s), error = 63.6%

$\#$ of iterations $= 70$
Relative error $= 63.6\%$

Reconstruction methods

Kaczmarz method: impact of number of iterations

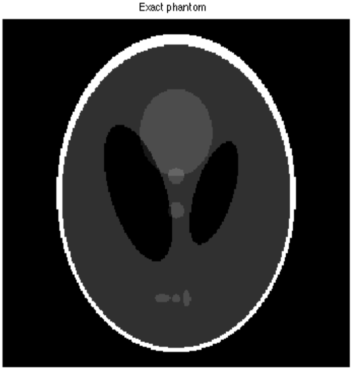


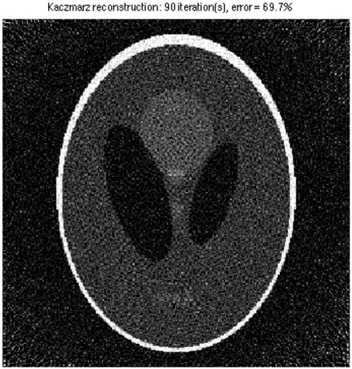


$\#$ of iterations $= 80$
Relative error $= 66.7\%$

Reconstruction methods

Kaczmarz method: impact of number of iterations





$\#$ of iterations = 90
Relative error = 69.7%

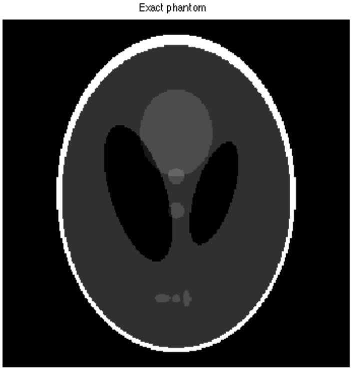
Reconstruction methods

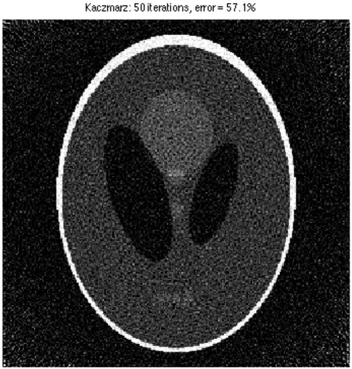
Kaczmarz method: impact of number of iterations

# of iterations = 100
Relative error = 72.5%

Reconstruction methods

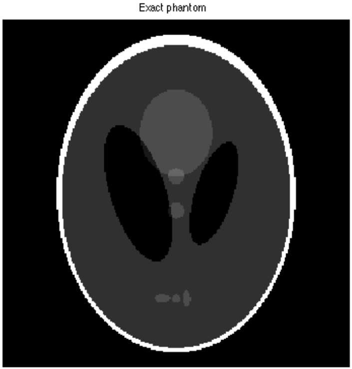
Kaczmarz method: impact of enforcing non-negativity

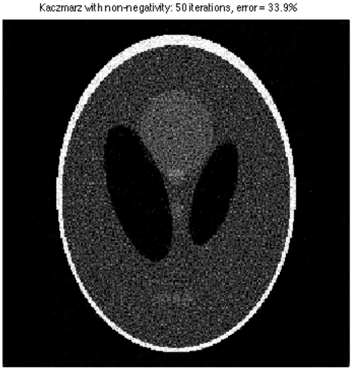




\# of iterations = 50
Non-negativity not enforced
Relative error = 57.1%

Reconstruction methods
Kaczmarz method: impact of enforcing non-negativity





\# of iterations $= 50$
Non-negativity enforced
Relative error $= 33.9\%$

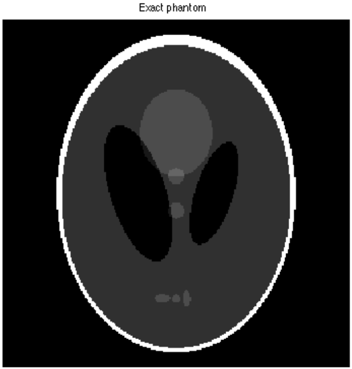
Simulation protocol

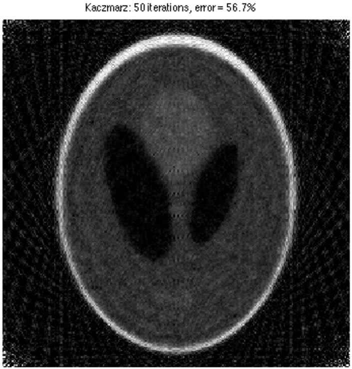
- Phantom: 256×256 pixel 2D Shepp-Logan
  $n = 256 \cdot 256 = 65\,536$
- Data: Full angular range $[0^\circ, 180^\circ]$ with 5° step (36 directions) and 256 detector elements.
  $m = 36 \cdot 256 = 9216$.
- Noise component in data: Additive Gaussian noise with relative noise level 5%.

Underdetermined problem since $m < n$.

Reconstruction methods

Kaczmarz method: impact of enforcing non-negativity

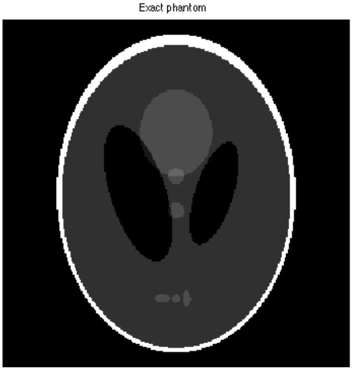


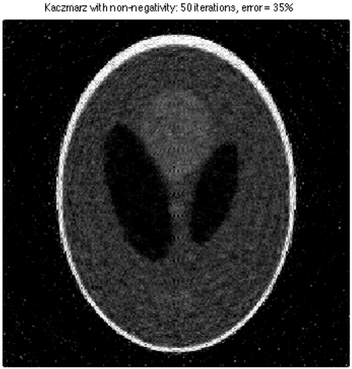


\# of iterations $= 50$
Non-negativity not enforced
Relative error $= 56.7\%$

Reconstruction methods

Kaczmarz method: impact of enforcing non-negativity





\# of iterations = 50
Non-negativity enforced
Relative error = 35.0%

Reconstruction methods

Kaczmarz method: regularised variants

- Constraints: Enforce bound constraints, such as positivity, by projection techniques.
- Semi-convergence: Reconstruction improves during the first few iterates, then it begins to deteriorate (semi-convergence). In medical imaging only a few iterations are used.
  For ART: Empirical observation, no theoretical backing:
  - T. Elfving, P. C. Hansen, and T. Nikazad. *Semi-convergence properties of Kaczmarz's method*, Inverse Problems, vol. 30, no. 5 (055007), 2014.

  Stopping rule: Rule for choosing the number of iterations.
- Regularisation by relaxation: Modification of iterates

  $$\alpha_{k+1} := \alpha_k + \lambda_k \frac{g_i - a_i \cdot \alpha_k}{\|a_i\|_2^2} a_i \quad \text{where } i = k \mod m+1.$$

  $0 < \lambda_k < 2$ is a relaxation parameter.

Regularisation parameters: Number of iterates and relaxation parameter.

- Constraints: Enforce bound constraints, such as positivity, by projection techniques.
- Semi-convergence: Reconstruction improves during the first few iterates, then it begins to deteriorate (semi-convergence). In medical imaging only a few iterations are used.
  For ART: Empirical observation, no theoretical backing:
  - T. Elfving, P. C. Hansen, and T. Nikazad. *Semi-convergence properties of Kaczmarz's method*, Inverse Problems, vol. 30, no. 5 (055007), 2014.

  Stopping rule: Rule for choosing the number of iterations.
- Regularisation by relaxation: Modification of iterates

$$\alpha_{k+1} := \alpha_k + \lambda_k \frac{g_i - a_i \cdot \alpha_k}{\|a_i\|_2^2} a_i \quad \text{where } i = k \mod m+1.$$

  $0 < \lambda_k < 2$ is a relaxation parameter.

Regularisation parameters: Number of iterates and relaxation parameter.

Reconstruction methods

Kaczmarz method: regularised variants

- Constraints: Enforce bound constraints, such as positivity, by projection techniques.
- Semi-convergence: Reconstruction improves during the first few iterates, then it begins to deteriorate (semi-convergence). In medical imaging only a few iterations are used.
  For ART: Empirical observation, no theoretical backing:
  - T. Elfving, P. C. Hansen, and T. Nikazad. *Semi-convergence properties of Kaczmarz's method*, Inverse Problems, vol. 30, no. 5 (055007), 2014.

  Stopping rule: Rule for choosing the number of iterations.
- Regularisation by relaxation: Modification of iterates

$$\boldsymbol{\alpha}_{k+1} := \boldsymbol{\alpha}_k + \lambda_k \frac{g_i - a_i \cdot \boldsymbol{\alpha}_k}{\|a_i\|_2^2} a_i \quad \text{where } i = k \mod m+1.$$

  $0 < \lambda_k < 2$ is a relaxation parameter.

Regularisation parameters: Number of iterates and relaxation parameter.

- Constraints: Enforce bound constraints, such as positivity, by projection techniques.
- Semi-convergence: Reconstruction improves during the first few iterates, then it begins to deteriorate (semi-convergence). In medical imaging only a few iterations are used.
  For ART: Empirical observation, no theoretical backing:
  - T. Elfving, P. C. Hansen, and T. Nikazad. *Semi-convergence properties of Kaczmarz's method*, Inverse Problems, vol. 30, no. 5 (055007), 2014.

  Stopping rule: Rule for choosing the number of iterations.
- Regularisation by relaxation: Modification of iterates

$$\boldsymbol{\alpha}_{k+1} := \boldsymbol{\alpha}_k + \lambda_k \frac{g_i - a_i \cdot \boldsymbol{\alpha}_k}{\|a_i\|_2^2} a_i \quad \text{where } i = k \mod m+1.$$

  $0 < \lambda_k < 2$ is a relaxation parameter.

Regularisation parameters: Number of iterates and relaxation parameter.

- Algebraic reconstruction technique (ART)
  - Based on vector multiplications: A single iterative update makes use of a single row of $\mathbf{A}$.
  - Good semi-convergence observed (no theory explaining this).

  Algorithms: Kaczmarz's method and variants of it.

- Simultaneous iterative reconstruction technique (SIRT)
  - Based on matrix multiplications: A single iterative update uses all the rows of $\mathbf{A}$ simultaneously.
  - Slower semi-convergence, but otherwise good understanding of convergence theory.

  Algorithms: Landweber, Cimmino, CAV, DROP, and SART.

- Krylov subspace methods: Class of iterative methods based on matrix multiplications where iterates are given as

$$\alpha_k = \alpha_{k-1} + \mathbf{K}^{-1} \cdot (g - \mathbf{A} \cdot \alpha_{k-1}).$$

  $\mathbf{K}$ is here a simple invertible matrix that is "close" to $\mathbf{A}$.
  Algorithms: CGLS, LSQR, GMRES, ...

Iterative reconstruction methods

Row-action methods

- Algebraic reconstruction technique (ART)
  - Based on vector multiplications: A single iterative update makes use of a single row of **A**.
  - Good semi-convergence observed (no theory explaining this).

  Algorithms: Kaczmarz's method and variants of it.
- Simultaneous iterative reconstruction technique (SIRT)
  - Based on matrix multiplications: A single iterative update uses all the rows of **A** simultaneously.
  - Slower semi-convergence, but otherwise good understanding of convergence theory.

  Algorithms: Landweber, Cimmino, CAV, DROP, and SART.
- Krylov subspace methods: Class of iterative methods based on matrix multiplications where iterates are given as

$$\alpha_k = \alpha_{k-1} + \mathbf{K}^{-1} \cdot (g - \mathbf{A} \cdot \alpha_{k-1}).$$

  K is here a simple invertible matrix that is "close" to **A**.

  Algorithms: CGLS, LSQR, GMRES, ...

- Algebraic reconstruction technique (ART)
  - Based on vector multiplications: A single iterative update makes use of a single row of $\mathbf{A}$.
  - Good semi-convergence observed (no theory explaining this).

  Algorithms: Kaczmarz's method and variants of it.
- Simultaneous iterative reconstruction technique (SIRT)
  - Based on matrix multiplications: A single iterative update uses all the rows of $\mathbf{A}$ simultaneously.
  - Slower semi-convergence, but otherwise good understanding of convergence theory.

  Algorithms: Landweber, Cimmino, CAV, DROP, and SART.
- Krylov subspace methods: Class of iterative methods based on matrix multiplications where iterates are given as

  $$\boldsymbol{\alpha}_k = \boldsymbol{\alpha}_{k-1} + \mathbf{K}^{-1} \cdot (g - \mathbf{A} \cdot \boldsymbol{\alpha}_{k-1}).$$

  $\mathbf{K}$ is here a simple invertible matrix that is "close" to $\mathbf{A}$.

  Algorithms: CGLS, LSQR, GMRES, ...

Typical step: Update of iterate involves a single row of $\mathbf{A}$:

$$\boldsymbol{\alpha}_{k+1} := \boldsymbol{\alpha}_k + \lambda_k \frac{g_{i_k} - a_{i_k} \cdot \boldsymbol{\alpha}_k}{\|a_{i_k}\|_2^2} a_{i_k} \quad \text{where } k = 1, 2 \dots$$

and $i_k \in \{1, \dots, m\}$ is given by the row ordering.

- Relaxation parameter: $0 < \lambda_k < 2$, setting $\lambda_k = 1$ implies projecting $\boldsymbol{\alpha}_k$ onto hyperplane $g_i = a_i \cdot \boldsymbol{\alpha}$ (original un-regularised Kaczmarz's method).
- Row ordering: How $i_k \in \{1, \dots, m\}$ depends on k, *i.e.*, how the iterates sweep through the m rows $a_1, \dots, a_m$:
  - Classical Kaczmarz: $i_k = 1, 2, \dots, m, 1, 2, \dots, m, \dots$
  - Symmetric Kaczmarz: $i_k = 1, 2, \dots, m-1, m, m-1, \dots, 2, 1, \dots$
  - Randomized Kaczmarz: At each k, let i_k be the i:th row a_i randomly with probability proportional to the row norm $\|a_i\|_2$.

- Semi-convergence: Not formally proved, only empirically observed
  - Fast initial convergence $\Longrightarrow$ method of choice when only a few iterations can be afforded.
  - After some initial iterations the convergence can be very slow.
- Convergence rate:
  - Estimates of convergence rates are based on quantities of $\mathbf{A}$ that are hard to compute and difficult to compare with convergence estimates of other iterative methods.
  - Need to exploit the analytic structure associated with the forward operator.
  - Rate of convergence depends on the ordering of the equations.

- Choosing relaxation parameter
  For $\lambda_k = 1$ the high-frequency components (such as noise) show up early in the iteration, while overall features are determined later.
  For $\lambda_k \ll 1$, say 0.1, the iterations first determine the smooth parts of the image and small details in later iterates
  $\Longrightarrow$ suprprisingly small values of λ_k (*e.g.*, $\lambda_k = 0.05$) are quite common in ART.
- Stopping rules
  - The discrepancy principle
  - The L-curve
  - Generalised cross-validation (GCV)
  - Normalised cumulative periodogram (NCP)
  - ...

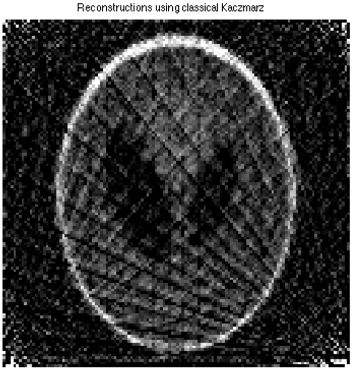
Iterative reconstruction methods

Influence of relaxation

Problem size & noise
n=16384
m=5400
noise level=10%

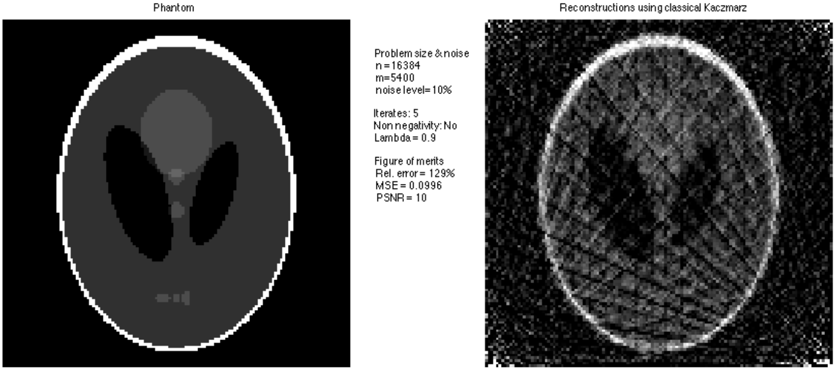
Iterates: 5
Non negativity: No
Lambda = 1

Figure of merits
Rel. error = 145%
MSE = 0.126
PSNR = 9



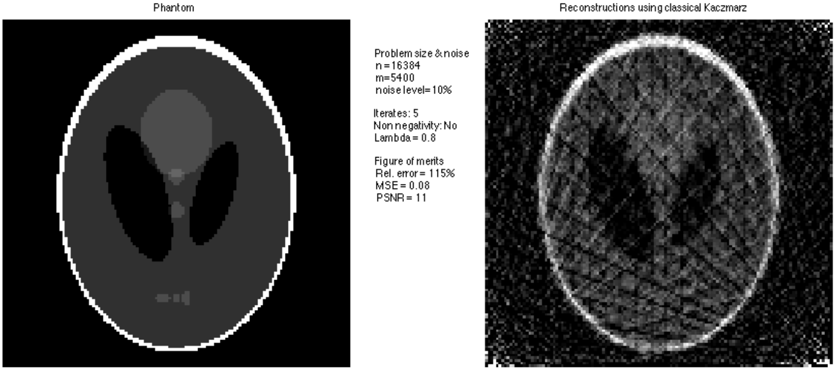
$$\lambda_k = 1.0$$

Iterative reconstruction methods
Influence of relaxation



$$\lambda_k = 0.9$$

Iterative reconstruction methods
Influence of relaxation



$$\lambda_k = 0.8$$

Iterative reconstruction methods
Influence of relaxation

Problem size & noise
n = 16384
m=5400
noise level= 10%

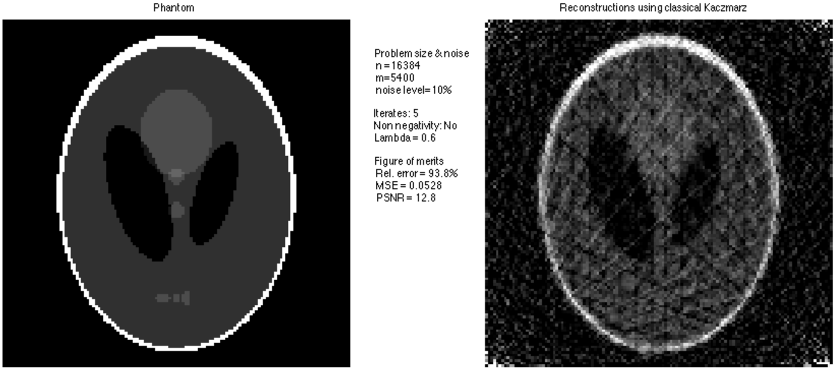
Iterates: 5
Non negativity: No
Lambda = 0.7

Figure of merits
Rel. error = 104%
MSE = 0.0648
PSNR = 11.9

$$\lambda_k = 0.7$$

Iterative reconstruction methods

Influence of relaxation



$$\lambda_k = 0.6$$

Iterative reconstruction methods
Influence of relaxation

Phantom

Problem size & noise
n = 16384
m=5400
noise level= 10%

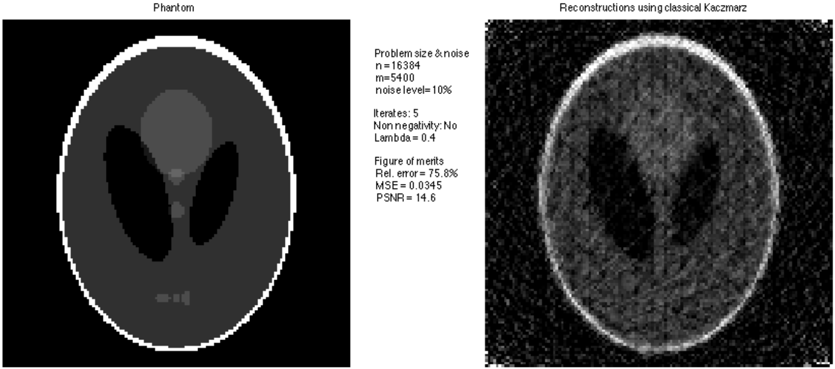
Iterates: 5
Non negativity: No
Lambda = 0.5

Figure of merits
Rel. error = 84.5%
MSE = 0.0428
PSNR = 13.7

Reconstructions using classical Kaczmarz

$$\lambda_k = 0.5$$

Iterative reconstruction methods
Influence of relaxation



$$\lambda_k = 0.4$$

Phantom

Problem size & noise
n = 16384
m=5400
noise level= 10%

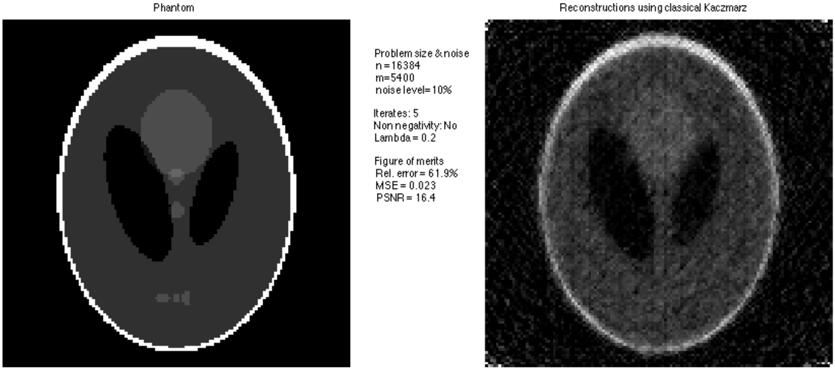
Iterates: 5
Non negativity: No
Lambda = 0.3

Figure of merits
Rel. error = 68%
MSE = 0.0277
PSNR = 15.6

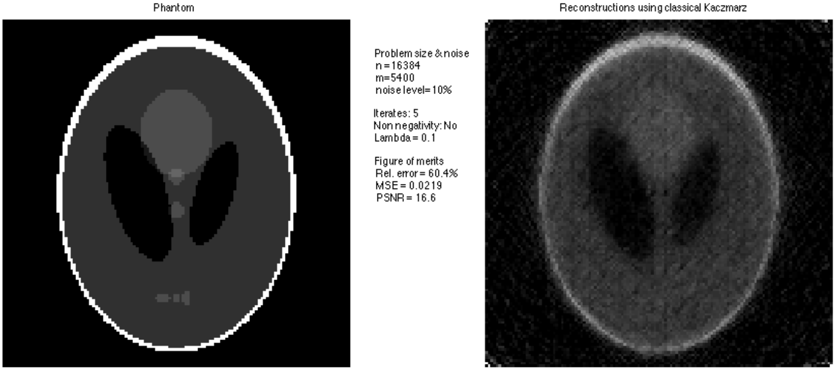
Reconstructions using classical Kaczmarz

$\lambda_k = 0.3$

Iterative reconstruction methods
Influence of relaxation



$$\lambda_k = 0.2$$



Phantom
Reconstructions using classical Kaczmarz
Problem size & noise
n = 16384
m=5400
noise level= 10%
Iterates: 5
Non negativity: No
Lambda = 0.1
Figure of merits
Rel. error = 60.4%
MSE = 0.0219
PSNR = 16.6

$$\lambda_k = 0.1$$

Iterative reconstruction methods
Influence of relaxation

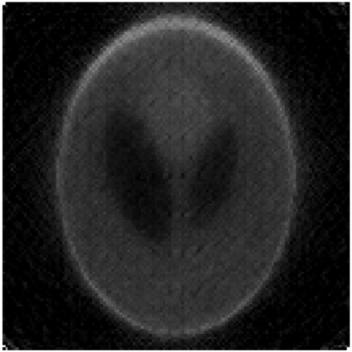
Phantom

Problem size & noise
n = 16384
m=5400
noise level= 10%

Iterates: 5
Non negativity: No
Lambda = 0.05

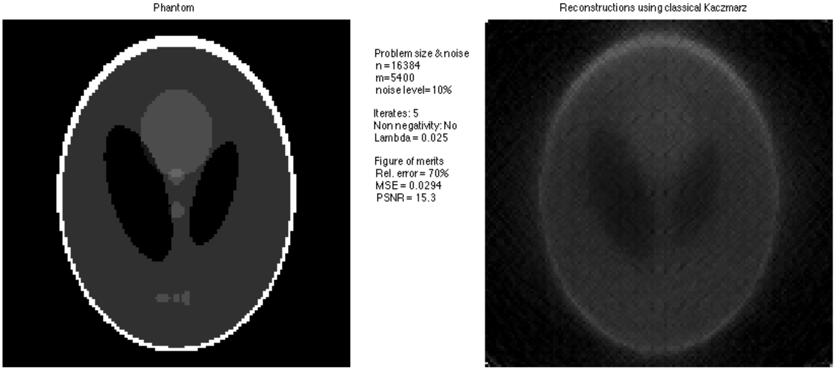
Figure of merits
Rel. error = 64.4%
MSE = 0.0249
PSNR = 16

Reconstructions using classical Kaczmarz



$$\lambda_k = 0.05$$

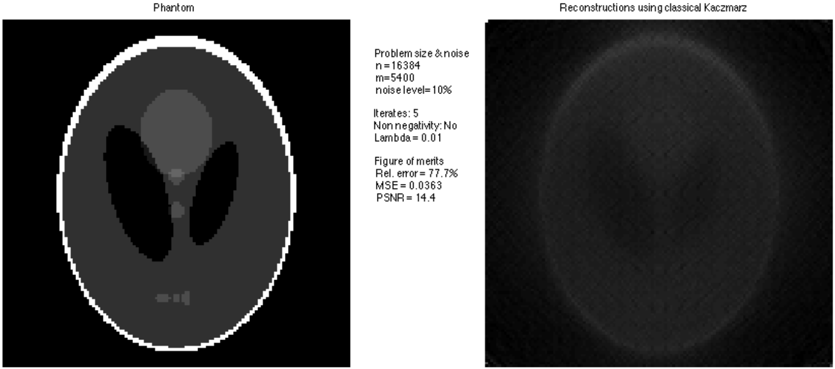
Iterative reconstruction methods
Influence of relaxation



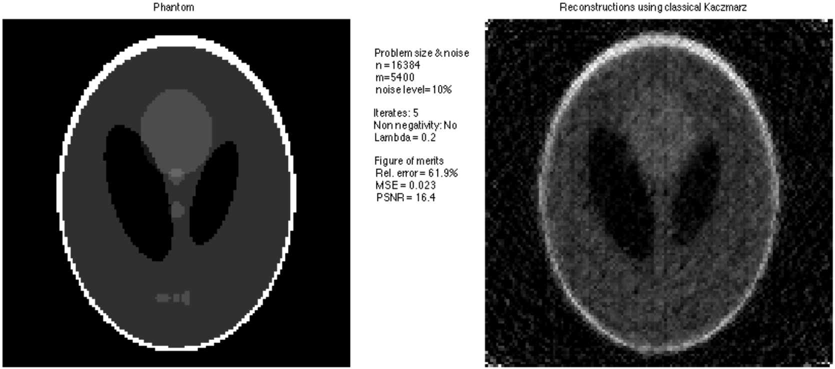
$$\lambda_k = 0.025$$

Iterative reconstruction methods

Influence of relaxation

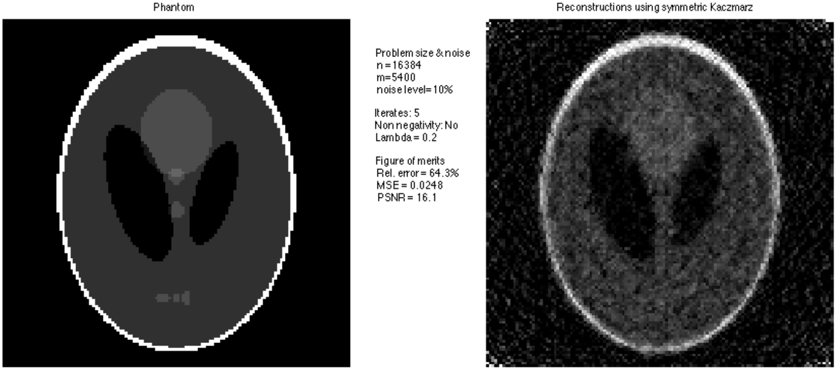


$$\lambda_k = 0.01$$



Phantom
Reconstructions using classical Kaczmarz
Problem size & noise
n = 16384
m=5400
noise level= 10%
Iterates: 5
Non negativity: No
Lambda = 0.2
Figure of merits
Rel. error = 61.9%
MSE = 0.023
PSNR = 16.4

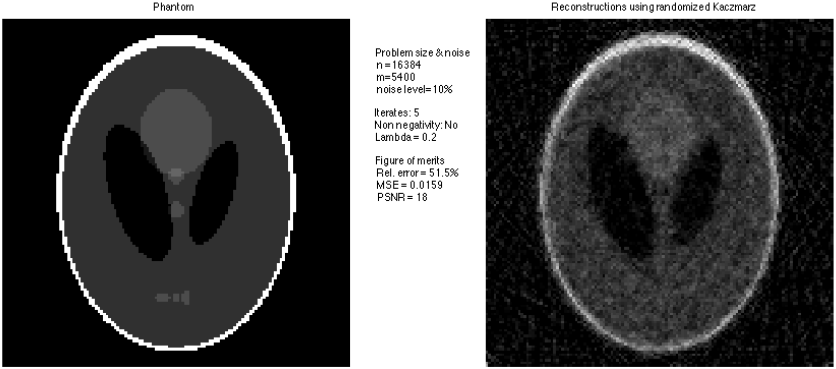
Classical



Phantom
Problem size & noise
n = 16384
m=5400
noise level= 10%
Iterates: 5
Non negativity: No
Lambda = 0.2
Figure of merits
Rel. error = 64.3%
MSE = 0.0248
PSNR = 16.1
Reconstructions using symmetric Kaczmarz

Symmetric

Iterative reconstruction methods
Influence of row ordering



Random

Least-squares problem for the reconstruction problem:

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^n} Q(\boldsymbol{\alpha}) \quad \text{where} \quad Q(\boldsymbol{\alpha}) := \frac{1}{2}\|\mathbf{A}\cdot\boldsymbol{\alpha} - g\|_2^2. \tag{3}$$

Gradient-descent method: Solve (3) by the iterative scheme

$$\boldsymbol{\alpha}_{k+1} := \boldsymbol{\alpha}_k - \lambda_{k+1}\nabla Q(\boldsymbol{\alpha}_k) \quad \text{for } k = 0, 1, 2, \ldots.$$

- The gradient of the quadratic form $Q\colon \mathbb{R}^n \to \mathbb{R}_+$ is

$$\nabla Q(\alpha) = \mathbf{A}^{\mathrm{t}}\cdot\big(\mathbf{A}\cdot\alpha - g\big).$$

- Value of step size λ_k may change at every iteration.
- Convergence to a local minima of (3) can be guaranteed for certain methods for choosing λ_k.
- Serves as a basis for simultaneous iterative reconstruction technique (SIRT) methods.

Least-squares problem for the reconstruction problem:

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^n} Q(\boldsymbol{\alpha}) \quad \text{where} \quad Q(\boldsymbol{\alpha}) := \frac{1}{2}\|\mathbf{A} \cdot \boldsymbol{\alpha} - g\|_2^2. \tag{3}$$

Gradient-descent method: Solve (3) by the iterative scheme

$$\boldsymbol{\alpha}_{k+1} := \boldsymbol{\alpha}_k - \lambda_{k+1} \nabla Q(\boldsymbol{\alpha}_k) \quad \text{for } k = 0, 1, 2, \ldots.$$

- The gradient of the quadratic form $Q \colon \mathbb{R}^n \to \mathbb{R}_+$ is

$$\nabla Q(\boldsymbol{\alpha}) = \mathbf{A}^{\mathrm{t}} \cdot \big(\mathbf{A} \cdot \boldsymbol{\alpha} - g\big).$$

- Value of step size λ_k may change at every iteration.
- Convergence to a local minima of (3) can be guaranteed for certain methods for choosing λ_k.
- Serves as a basis for simultaneous iterative reconstruction technique (SIRT) methods.

General form of iterates:

$$\boldsymbol{\alpha}_{k+1} := \boldsymbol{\alpha}_k - \lambda_k \mathbf{T} \cdot \mathbf{A}^{\mathrm{t}} \cdot \mathbf{M} \cdot \big(\mathbf{A} \cdot \boldsymbol{\alpha}_k - g\big) \quad \text{for } k = 0, 1, 2, \ldots$$

Iterates are stopped early and $\lambda_k > 0$ is a relaxation parameter.

Methods vary depending on choices of the diagonal $(n \times n)$-matrix $\mathbf{T}$ and the diagonal $(m \times m)$-matrix $\mathbf{M}$.

- Classical Landweber: $\mathbf{T} = \mathbf{I}_n$ and $\mathbf{M} = \mathbf{I}_m$.
- Cimmino: $\mathbf{T} = \mathbf{I}_n$ and $\mathbf{M} = \mathbf{D}$.
- CAV: $\mathbf{T} = \mathbf{I}_n$ and $\mathbf{M} = \mathbf{D}_S$.
- DROP: $\mathbf{T} = \mathrm{diag}(1/s_j)$ and $\mathbf{M} = m\mathbf{D}$.
- SART: $\mathbf{T} = \mathrm{diag}\Big(1/\sum_{i=1}^{m} a_{i,j}\Big)$ and $\mathbf{M} = \mathrm{diag}\Big(1/\sum_{j=1}^{n} a_{i,j}\Big)$.

General form of iterates:

$$\boldsymbol{\alpha}_{k+1} := \boldsymbol{\alpha}_k - \lambda_k \mathbf{T} \cdot \mathbf{A}^{\mathrm{t}} \cdot \mathbf{M} \cdot \big(\mathbf{A} \cdot \boldsymbol{\alpha}_k - g\big) \quad \text{for } k = 0, 1, 2, \dots$$

Iterates are stopped early and $\lambda_k > 0$ is a relaxation parameter.

Methods vary depending on choices of the diagonal $(n \times n)$-matrix $\mathbf{T}$ and the diagonal $(m \times m)$-matrix $\mathbf{M}$.

- Classical Landweber: $\mathbf{T} = \mathbf{I}_n$ and $\mathbf{M} = \mathbf{I}_m$.
- Cimmino: $\mathbf{T} = \mathbf{I}_n$ and $\mathbf{M} = \mathbf{D}$.
- CAV: $\mathbf{T} = \mathbf{I}_n$ and $\mathbf{M} = \mathbf{D}_S$.
- DROP: $\mathbf{T} = \operatorname{diag}(1/s_j)$ and $\mathbf{M} = m\mathbf{D}$.
- SART: $\mathbf{T} = \operatorname{diag}\Big(1/\sum_{i=1}^{m} a_{i,j}\Big)$ and $\mathbf{M} = \operatorname{diag}\Big(1/\sum_{j=1}^{n} a_{i,j}\Big)$.

Here

$$\mathbf{D} = \frac{1}{m} \operatorname{diag}\Big(\frac{1}{\|a_i\|_2^2}\Big)$$

General form of iterates:

$$\boldsymbol{\alpha}_{k+1} := \boldsymbol{\alpha}_k - \lambda_k \mathbf{T} \cdot \mathbf{A}^{\mathrm{t}} \cdot \mathbf{M} \cdot \bigl(\mathbf{A} \cdot \boldsymbol{\alpha}_k - g\bigr) \quad \text{for } k = 0, 1, 2, \ldots$$

Iterates are stopped early and $\lambda_k > 0$ is a relaxation parameter.

Methods vary depending on choices of the diagonal $(n \times n)$-matrix $\mathbf{T}$ and the diagonal $(m \times m)$-matrix $\mathbf{M}$.

- Classical Landweber: $\mathbf{T} = \mathbf{I}_n$ and $\mathbf{M} = \mathbf{I}_m$.
- Cimmino: $\mathbf{T} = \mathbf{I}_n$ and $\mathbf{M} = \mathbf{D}$.
- CAV: $\mathbf{T} = \mathbf{I}_n$ and $\mathbf{M} = \mathbf{D}_S$.
- DROP: $\mathbf{T} = \operatorname{diag}(1/s_j)$ and $\mathbf{M} = m\mathbf{D}$.
- SART: $\mathbf{T} = \operatorname{diag}\Bigl(1/\sum_{i=1}^{m} a_{i,j}\Bigr)$ and $\mathbf{M} = \operatorname{diag}\Bigl(1/\sum_{j=1}^{n} a_{i,j}\Bigr)$.

Here

$$\mathbf{D} = \frac{1}{m}\operatorname{diag}\Bigl(\frac{1}{\|a_i\|_2^2}\Bigr) \quad \text{and} \quad \mathbf{D}_S := \operatorname{diag}\Bigl(\frac{1}{\sum_{j=1}^{n} s_j a_{i,j}^2}\Bigr)$$

$s_j =$ number of non-zero elements in j:th column $\mathbf{A}$

General form of iterates:

$$\boldsymbol{\alpha}_{k+1} := \boldsymbol{\alpha}_k - \lambda_k \mathbf{T} \cdot \mathbf{A}^{\mathrm{t}} \cdot \mathbf{M} \cdot \big(\mathbf{A} \cdot \boldsymbol{\alpha}_k - g\big) \quad \text{for } k = 0, 1, 2, \ldots$$

Iterates are stopped early and $\lambda_k > 0$ is a relaxation parameter.

Methods vary depending on choices of the diagonal $(n \times n)$-matrix $\mathbf{T}$ and the diagonal $(m \times m)$-matrix $\mathbf{M}$.

- Classical Landweber: $\mathbf{T} = \mathbf{I}_n$ and $\mathbf{M} = \mathbf{I}_m$.
- Cimmino: $\mathbf{T} = \mathbf{I}_n$ and $\mathbf{M} = \mathbf{D}$.
- CAV: $\mathbf{T} = \mathbf{I}_n$ and $\mathbf{M} = \mathbf{D}_S$.
- DROP: $\mathbf{T} = \operatorname{diag}(1/s_j)$ and $\mathbf{M} = m\mathbf{D}$.
- SART: $\mathbf{T} = \operatorname{diag}\Big(1/\sum_{i=1}^{m} a_{i,j}\Big)$ and $\mathbf{M} = \operatorname{diag}\Big(1/\sum_{j=1}^{n} a_{i,j}\Big)$.

Here

$$\mathbf{D} = \frac{1}{m} \operatorname{diag}\Big(\frac{1}{\|a_i\|_2^2}\Big)$$

$$s_j = \text{ number of non-zero elements in } j\text{:th column } \mathbf{A}$$

General form of iterates:

$$\boldsymbol{\alpha}_{k+1} := \boldsymbol{\alpha}_k - \lambda_k \mathbf{T} \cdot \mathbf{A}^{\mathrm{t}} \cdot \mathbf{M} \cdot \left(\mathbf{A} \cdot \boldsymbol{\alpha}_k - g\right) \quad \text{for } k = 0, 1, 2, \dots$$

Iterates are stopped early and $\lambda_k > 0$ is a relaxation parameter.

Methods vary depending on choices of the diagonal $(n \times n)$-matrix $\mathbf{T}$ and the diagonal $(m \times m)$-matrix $\mathbf{M}$.

- Classical Landweber: $\mathbf{T} = \mathbf{I}_n$ and $\mathbf{M} = \mathbf{I}_m$.
- Cimmino: $\mathbf{T} = \mathbf{I}_n$ and $\mathbf{M} = \mathbf{D}$.
- CAV: $\mathbf{T} = \mathbf{I}_n$ and $\mathbf{M} = \mathbf{D}_S$.
- DROP: $\mathbf{T} = \mathrm{diag}(1/s_j)$ and $\mathbf{M} = m\mathbf{D}$.
- SART: $\mathbf{T} = \mathrm{diag}\Big(1/\sum_{i=1}^{m} a_{i,j}\Big)$ and $\mathbf{M} = \mathrm{diag}\Big(1/\sum_{j=1}^{n} a_{i,j}\Big)$.

General form of iterates:

$$\boldsymbol{\alpha}_{k+1} := \boldsymbol{\alpha}_k - \lambda_k \mathbf{T} \cdot \mathbf{A}^{\mathrm{t}} \cdot \mathbf{M} \cdot \big(\mathbf{A} \cdot \boldsymbol{\alpha}_k - g\big) \quad \text{for } k = 0, 1, 2, \ldots$$

Iterates are stopped early and $\lambda_k > 0$ is a relaxation parameter.

Methods vary depending on choices of the diagonal $(n \times n)$-matrix $\mathbf{T}$ and the diagonal $(m \times m)$-matrix $\mathbf{M}$.

- Classical Landweber: $\mathbf{T} = \mathbf{I}_n$ and $\mathbf{M} = \mathbf{I}_m$.
- Cimmino: $\mathbf{T} = \mathbf{I}_n$ and $\mathbf{M} = \mathbf{D}$.
- CAV: $\mathbf{T} = \mathbf{I}_n$ and $\mathbf{M} = \mathbf{D}_S$.
- DROP: $\mathbf{T} = \operatorname{diag}(1/s_j)$ and $\mathbf{M} = m\mathbf{D}$.
- SART: $\mathbf{T} = \operatorname{diag}\Big(1/\sum_{i=1}^{m} a_{i,j}\Big)$ and $\mathbf{M} = \operatorname{diag}\Big(1/\sum_{j=1}^{n} a_{i,j}\Big)$.

Note: $\mathbf{T}$ and $\mathbf{M}$ are diagonal matrices $\Rightarrow$ can be stored.

Iterative reconstruction methods
SIRT

General form of iterates:

$$\boldsymbol{\alpha}_{k+1} := \boldsymbol{\alpha}_k - \lambda_k \mathbf{T} \cdot \mathbf{A}^{\mathrm{t}} \cdot \mathbf{M} \cdot \big(\mathbf{A} \cdot \boldsymbol{\alpha}_k - g\big) \quad \text{for } k = 0, 1, 2, \ldots$$

Iterates are stopped early and $\lambda_k > 0$ is a relaxation parameter.

Methods vary depending on choices of the diagonal $(n \times n)$-matrix $\mathbf{T}$ and the diagonal $(m \times m)$-matrix $\mathbf{M}$.

- Classical Landweber: $\mathbf{T} = \mathbf{I}_n$ and $\mathbf{M} = \mathbf{I}_m$.
- Cimmino: $\mathbf{T} = \mathbf{I}_n$ and $\mathbf{M} = \mathbf{D}$.
- CAV: $\mathbf{T} = \mathbf{I}_n$ and $\mathbf{M} = \mathbf{D}_S$.
- DROP: $\mathbf{T} = \operatorname{diag}(1/s_j)$ and $\mathbf{M} = m\mathbf{D}$.
- SART: $\mathbf{T} = \operatorname{diag}\Big(1/\sum_{i=1}^{m} a_{i,j}\Big)$ and $\mathbf{M} = \operatorname{diag}\Big(1/\sum_{j=1}^{n} a_{i,j}\Big)$.

Several stopping rules and approaches to choose the relaxation parameter λ_k.

Iterates:

$$\boldsymbol{\alpha}_{k+1} := \boldsymbol{\alpha}_k - \lambda_k \cdot \mathbf{A}^{\mathsf{t}} \cdot \big(\mathbf{A} \cdot \boldsymbol{\alpha} - g\big) = \boldsymbol{\alpha}_k - \lambda_k \nabla Q(\boldsymbol{\alpha}_k)$$

where $Q(\boldsymbol{\alpha}) := \dfrac{1}{2}\|\mathbf{A} \cdot \boldsymbol{\alpha} - g\|_2^2$.

- Each step in Landweber's method is a step in the direction of steepest descent.
- Semi-convergence property well-established.
- $0 < \lambda_k < 2/\sigma^2$ where σ is an estimate of the largest singular value of $\mathbf{A}$.

Iterates:

$$\boldsymbol{\alpha}_{k+1} := \boldsymbol{\alpha}_k - \lambda_k \cdot \mathbf{A}^{\mathrm{t}} \cdot \mathbf{D} \cdot \big(\mathbf{A} \cdot \boldsymbol{\alpha} - g\big) \quad \text{for } k = 0, 1, 2, \ldots$$

where $\mathbf{D} = \dfrac{1}{m} \operatorname{diag}\Big(\dfrac{1}{\|a_i\|_2^2}\Big)$, so

$$\boldsymbol{\alpha}_{k+1} := \boldsymbol{\alpha}_k + \frac{\lambda_k}{m} \sum_{i=1}^{m} \frac{g_i - a_i \cdot \boldsymbol{\alpha}_k}{\|a_i\|_2^2} a_i.$$

- Each step in Cimmino's method is the average of the projections onto the m hyperplanes $a_i \cdot \boldsymbol{\alpha} = g_i$ with $i = 1, \ldots, m$ (compare with ART).

Iterative reconstruction methods
Non-negativity and box constraints

Assume one knows beforehand that $\boldsymbol{\alpha}_{\text{true}} \in C \subset \mathbb{R}^n$ where C is a known convex set. One can incorporate such a constraint into ART and SIRT iterations by projecting iterates onto C:

- Projected ART

$$\boldsymbol{\alpha}_{k+1} := \mathcal{P}_C\Bigl(\boldsymbol{\alpha}_k + \lambda_k \frac{g_{i_k} - a_{i_k} \cdot \boldsymbol{\alpha}_k}{\|a_{i_k}\|_2^2} a_{i_k}\Bigr)$$

  with $i_k \in \{1, \dots, m\}$ given by the row-ordering scheme.

- Projected SIRT

$$\boldsymbol{\alpha}_{k+1} := \mathcal{P}_C\Bigl(\boldsymbol{\alpha}_k - \lambda_k \mathbf{T} \cdot \mathbf{A}^{\mathrm{t}} \cdot \mathbf{M} \cdot \bigl(\mathbf{A} \cdot \boldsymbol{\alpha} - g\bigr)\Bigr)$$

Assume one knows beforehand that $\boldsymbol{\alpha}_{\text{true}} \in C \subset \mathbb{R}^n$ where C is a known convex set. One can incorporate such a constraint into ART and SIRT iterations by projecting iterates onto C:

- Projected ART

$$\boldsymbol{\alpha}_{k+1} := \mathcal{P}_C\Big(\boldsymbol{\alpha}_k + \lambda_k \frac{g_{i_k} - a_{i_k} \cdot \boldsymbol{\alpha}_k}{\|a_{i_k}\|_2^2} a_{i_k}\Big)$$

  with $i_k \in \{1, \dots, m\}$ given by the row-ordering scheme.
- Projected SIRT

$$\boldsymbol{\alpha}_{k+1} := \mathcal{P}_C\Big(\boldsymbol{\alpha}_k - \lambda_k \mathbf{T} \cdot \mathbf{A}^{\mathrm{t}} \cdot \mathbf{M} \cdot \big(\mathbf{A} \cdot \boldsymbol{\alpha} - g\big)\Big)$$

C can represent box-constraints $a \leq \alpha_i \leq b$ in which case

$$\mathcal{P}_C(\boldsymbol{\alpha})_i = \begin{cases} \alpha_i & \text{if } a < \alpha_i < b \\ a & \text{if } \alpha_i \leq a \\ b & \text{if } \alpha_i \geq b \end{cases} \qquad \text{for } i = 1, \dots, n.$$

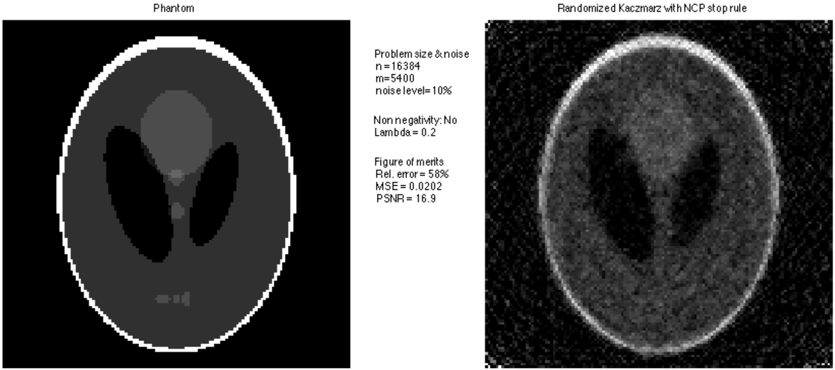
Assume one knows beforehand that $\boldsymbol{\alpha}_{\mathrm{true}} \in C \subset \mathbb{R}^n$ where C is a known convex set. One can incorporate such a constraint into ART and SIRT iterations by projecting iterates onto C:

- Projected ART
$$\boldsymbol{\alpha}_{k+1} := \mathcal{P}_C\Bigl(\boldsymbol{\alpha}_k + \lambda_k \frac{g_{i_k} - a_{i_k}\cdot\boldsymbol{\alpha}_k}{\|a_{i_k}\|_2^2} a_{i_k}\Bigr)$$
with $i_k \in \{1,\dots,m\}$ given by the row-ordering scheme.
- Projected SIRT
$$\boldsymbol{\alpha}_{k+1} := \mathcal{P}_C\Bigl(\boldsymbol{\alpha}_k - \lambda_k \mathbf{T}\cdot\mathbf{A}^{\mathrm{t}}\cdot\mathbf{M}\cdot\bigl(\mathbf{A}\cdot\boldsymbol{\alpha} - g\bigr)\Bigr)$$

Projected iterates should converge to a least-squares solution in C, *i.e.*, to a solution of
$$\min_{\boldsymbol{\alpha}\in C} \|\mathbf{A}\cdot\boldsymbol{\alpha} - g\|_2^2.$$

Projected SIRT converges to a solution of the above problem.

Iterative reconstruction methods
SIRT: examples without enforcing non-negativity

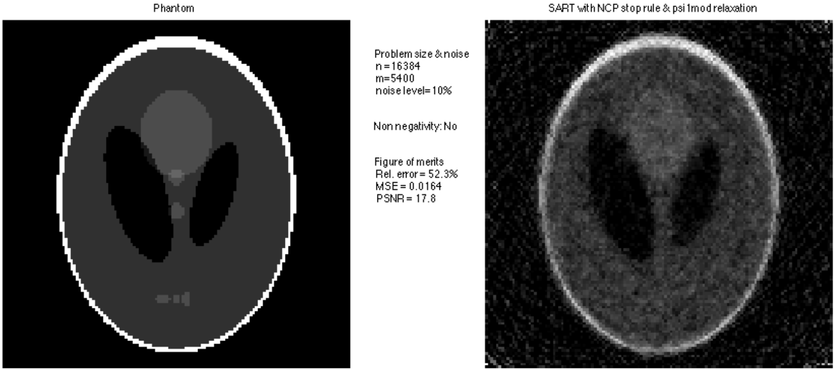


Randomized ART

All iterative schemes make use of the same stopping rule.

Iterative reconstruction methods

SIRT: examples without enforcing non-negativity

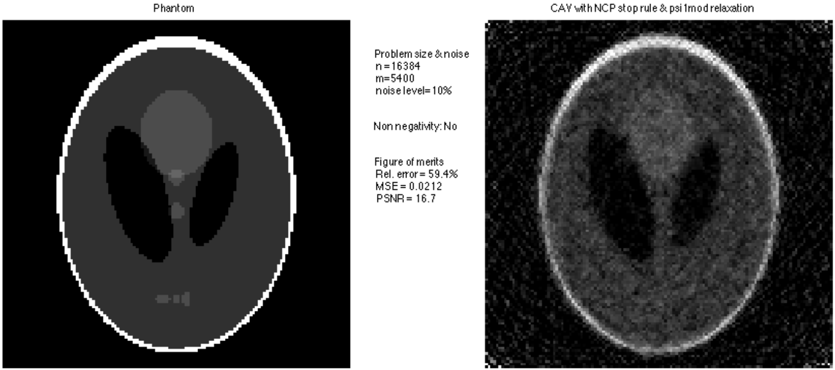


SART

All iterative schemes make use of the same stopping rule.

Iterative reconstruction methods

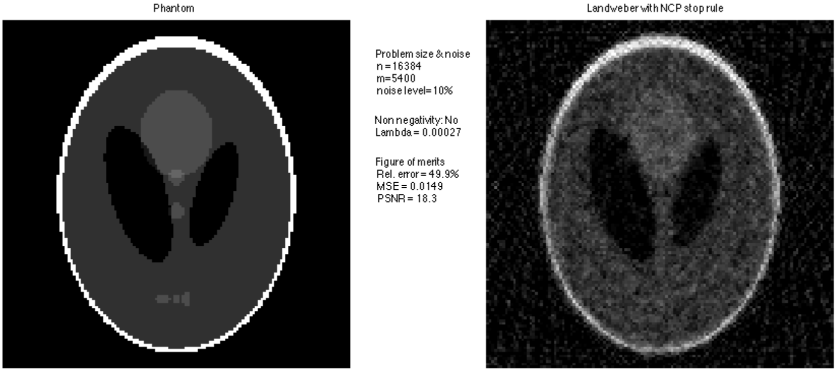
SIRT: examples without enforcing non-negativity



CAV

All iterative schemes make use of the same stopping rule.

Iterative reconstruction methods
SIRT: examples without enforcing non-negativity

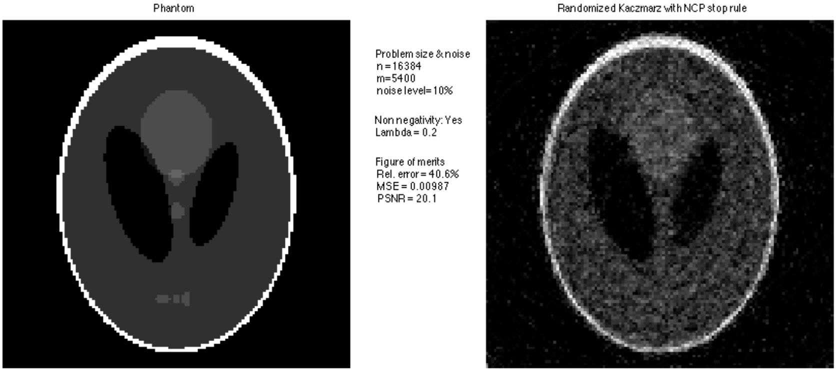


Landweber

All iterative schemes make use of the same stopping rule.

Iterative reconstruction methods

SIRT: examples enforcing non-negativity



Randomized ART

All iterative schemes make use of the same stopping rule.

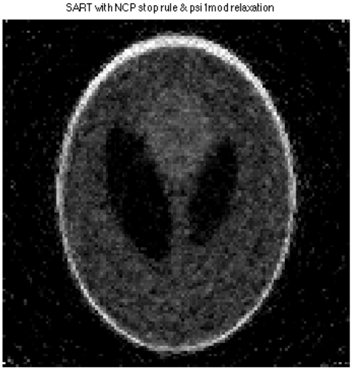
Iterative reconstruction methods

SIRT: examples enforcing non-negativity

Problem size & noise
n = 16384
m=5400
noise level=10%

Non negativity: Yes

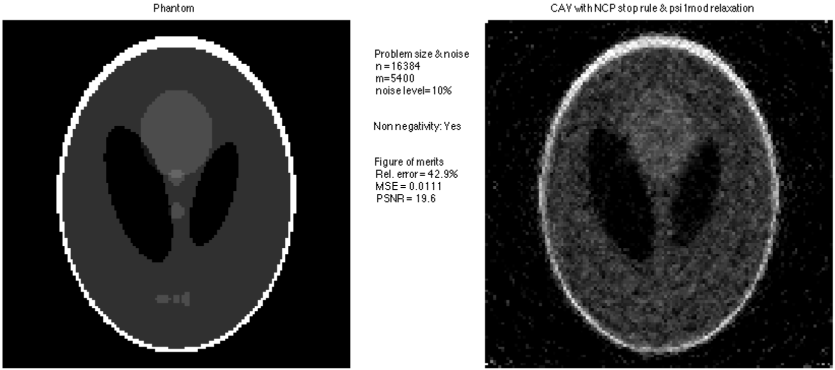
Figure of merits
Rel. error = 40.9%
MSE = 0.01
PSNR = 20



SART

All iterative schemes make use of the same stopping rule.

Iterative reconstruction methods
SIRT: examples enforcing non-negativity

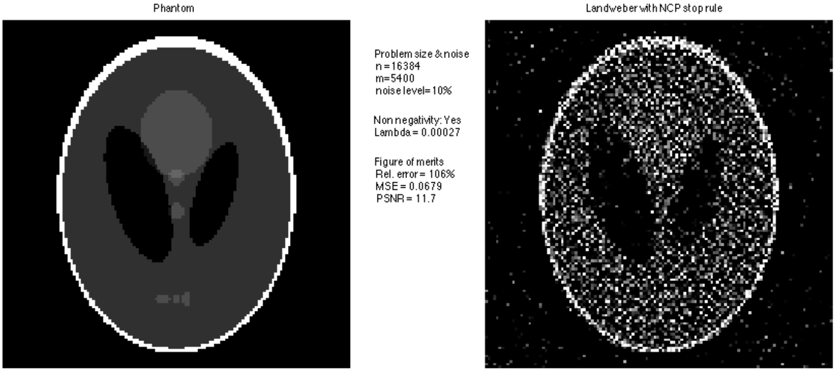


CAV

All iterative schemes make use of the same stopping rule.

Iterative reconstruction methods

SIRT: examples enforcing non-negativity



Landweber

All iterative schemes make use of the same stopping rule.

A Krylov subspace method – iterative scheme for finding a least-squares solution to the reconstruction problem:

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^n} Q(\boldsymbol{\alpha}) \quad \text{where} \quad Q(\boldsymbol{\alpha}) := \frac{1}{2} \|\mathbf{A} \cdot \boldsymbol{\alpha} - g\|_2^2. \tag{4}$$

Basic CG algorithm for minimising a non-linear function Q

1: $\boldsymbol{\alpha}_0$ arbitrary, $r_0 := \nabla Q(\boldsymbol{\alpha}_0)$, $d_0 := -r_0$.
2: **for** $k := 0, 1, \ldots$ **do**
3: $\quad \boldsymbol{\alpha}_{k+1} :=$ minima of Q on half-line $t \mapsto \boldsymbol{\alpha}_k + t d_k$
4: $\quad r_{k+1} := \nabla Q(\boldsymbol{\alpha}_{k+1})$
5: $\quad \beta_{k+1} := \|r_{k+1}\|_2^2 / \|r_k\|_2^2$
6: $\quad d_{k+1} := -r_{k+1} + \beta_{k+1} d_k$
7: **end for**

The conjugate gradient (CG) method

The basic algorithm

A Krylov subspace method – iterative scheme for finding a least-squares solution to the reconstruction problem:

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^n} Q(\boldsymbol{\alpha}) \quad \text{where} \quad Q(\boldsymbol{\alpha}) := \frac{1}{2}\|\mathbf{A}\cdot\boldsymbol{\alpha} - g\|_2^2. \tag{4}$$

Basic CG algorithm for minimising a non-linear function Q

1: $\boldsymbol{\alpha}_0$ arbitrary, $r_0 := \nabla Q(\boldsymbol{\alpha}_0)$, $d_0 := -r_0$.
2: **for** $k := 0, 1, \ldots$ **do**
3: $\quad\boldsymbol{\alpha}_{k+1} :=$ minima of Q on half-line $t \mapsto \boldsymbol{\alpha}_k + t d_k$
4: $\quad r_{k+1} := \nabla Q(\boldsymbol{\alpha}_{k+1})$
5: $\quad\beta_{k+1} := \|r_{k+1}\|_2^2 / \|r_k\|_2^2$
6: $\quad d_{k+1} := -r_{k+1} + \beta_{k+1} d_k$
7: **end for**

Gradient of Q: $\nabla Q(\boldsymbol{\alpha}) = \mathbf{A}^{\mathrm{t}} \cdot \big(\mathbf{A}\cdot\boldsymbol{\alpha} - g\big)$.

The conjugate gradient (CG) method
The basic algorithm

A Krylov subspace method – iterative scheme for finding a least-squares solution to the reconstruction problem:

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^n} Q(\boldsymbol{\alpha}) \quad \text{where} \quad Q(\boldsymbol{\alpha}) := \frac{1}{2}\|\mathbf{A}\cdot\boldsymbol{\alpha} - g\|_2^2. \tag{4}$$

CGLS algorithm for solving (4)

$\boldsymbol{\alpha}_0$ arbitrary, $r_0 := \mathbf{A}^{\mathrm{t}} \cdot \big(\mathbf{A}\cdot\boldsymbol{\alpha}_0 - g\big)$, $d_0 := -r_0$.
for $k := 0, 1, \ldots$ **do**
  $t_{k+1} := -\langle r_k, d_k\rangle / \|\mathbf{A}\cdot d_k\|_2^2$
  $\boldsymbol{\alpha}_{k+1} := \boldsymbol{\alpha}_k + t_{k+1} d_k$
  $r_{k+1} := \mathbf{A}^{\mathrm{t}} \cdot \big(\mathbf{A}\cdot\boldsymbol{\alpha}_{k+1} - g\big)$
  $\beta_{k+1} := \|r_{k+1}\|_2^2 / \|r_k\|_2^2$
  $d_{k+1} := -r_{k+1} + \beta_{k+1} d_k$
end for

The conjugate gradient (CG) method

The basic algorithm

A Krylov subspace method – iterative scheme for finding a least-squares solution to the reconstruction problem:

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^n} Q(\boldsymbol{\alpha}) \quad \text{where} \quad Q(\boldsymbol{\alpha}) := \frac{1}{2}\|\mathbf{A}\cdot\boldsymbol{\alpha} - g\|_2^2. \tag{4}$$

CGLS algorithm for solving (4)

```
α_0 arbitrary, r_0 := Aᵗ·(A·α_0 − g), d_0 := −r_0.
for k := 0, 1, ... do
    t_{k+1} := −⟨r_k, d_k⟩/‖A·d_k‖_2^2
    α_{k+1} := α_k + t_{k+1} d_k
    r_{k+1} := Aᵗ·(A·α_{k+1} − g)
    β_{k+1} := ‖r_{k+1}‖_2^2/‖r_k‖_2^2
    d_{k+1} := −r_{k+1} + β_{k+1} d_k
end for
```

$\boldsymbol{\alpha}_0$ arbitrary, $r_0 := \mathbf{A}^{\mathrm{t}}\cdot\big(\mathbf{A}\cdot\boldsymbol{\alpha}_0 - g\big)$, $d_0 := -r_0$.

for $k := 0, 1, \ldots$ **do**

$\quad t_{k+1} := -\langle r_k, d_k\rangle/\|\mathbf{A}\cdot d_k\|_2^2$

$\quad \boldsymbol{\alpha}_{k+1} := \boldsymbol{\alpha}_k + t_{k+1} d_k$

$\quad r_{k+1} := \mathbf{A}^{\mathrm{t}}\cdot\big(\mathbf{A}\cdot\boldsymbol{\alpha}_{k+1} - g\big)$

$\quad \beta_{k+1} := \|r_{k+1}\|_2^2/\|r_k\|_2^2$

$\quad d_{k+1} := -r_{k+1} + \beta_{k+1} d_k$

end for

All operations involve multiplication with $\mathbf{A}$ or $\mathbf{A}^{\mathrm{t}}$.

The conjugate gradient (CG) method

The basic algorithm

A Krylov subspace method – iterative scheme for finding a least-squares solution to the reconstruction problem:

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^n} Q(\boldsymbol{\alpha}) \quad \text{where} \quad Q(\boldsymbol{\alpha}) := \frac{1}{2}\|\mathbf{A}\cdot\boldsymbol{\alpha} - g\|_2^2. \tag{4}$$

CGLS algorithm for solving (4)

1: $\boldsymbol{\alpha}_0$ arbitrary, $r_0 := \mathbf{A}^{\mathrm{t}}\cdot\big(\mathbf{A}\cdot\boldsymbol{\alpha}_0 - g\big)$, $d_0 := -r_0$.
2: **for** $k := 0, 1, \ldots$ **do**
3: $\quad t_{k+1} := -\langle r_k, d_k\rangle / \|\mathbf{A}\cdot d_k\|_2^2$
4: $\quad \boldsymbol{\alpha}_{k+1} := \boldsymbol{\alpha}_k + t_{k+1} d_k$
5: $\quad r_{k+1} := \mathbf{A}^{\mathrm{t}}\cdot\big(\mathbf{A}\cdot\boldsymbol{\alpha}_{k+1} - g\big)$
6: $\quad \beta_{k+1} := \|r_{k+1}\|_2^2 / \|r_k\|_2^2$
7: $\quad d_{k+1} := -r_{k+1} + \beta_{k+1} d_k$
8: **end for**

Need to regularise by early stopping.

Choosing the regularisation parameter(s)

Recover a least-squares solution to the ill-posed problem

$$g = \mathbf{A} \cdot \alpha_{\text{true}} + g_{\text{noise}}.$$

- ART and SIRT methods involve two regularisation parameters, the relaxation parameter and the number of iterates.
- CG methods have one regularisation parameter, the number of iterates.

Choosing the regularisation parameter(s)

Recover a least-squares solution to the ill-posed problem

$$g = \mathbf{A} \cdot \alpha_{\text{true}} + g_{\text{noise}}.$$

- ART and SIRT methods involve two regularisation parameters, the relaxation parameter and the number of iterates.
- CG methods have one regularisation parameter, the number of iterates.

Two types of errors in regularisation of ill-posed problems:

Noise (perturbation) error: error in iterates due to "inverting" the noise g_{noise} in data.

Iteration (regularisation) error: errors in iterates that are due to semi-convergence (errors get amplified one iterates progress too far).

Both noise and iteration errors are always present in a regularised solution.

$\Longrightarrow$ their size depends on the regularisation parameter(s).

Choosing the regularisation parameter(s)

Recover a least-squares solution to the ill-posed problem

$$g = \mathbf{A} \cdot \alpha_{\text{true}} + g_{\text{noise}}.$$

- ART and SIRT methods involve two regularisation parameters, the relaxation parameter and the number of iterates.
- CG methods have one regularisation parameter, the number of iterates.

For iterative methods:

- The choice of the relaxation parameter mainly seeks to limit the noise error.
- The choice of number of iterates (stopping rule) mainly seeks to limit the iteration error.

Choosing the regularisation parameter(s)
The relaxation parameter

- ART and SIRT methods: Possible to estimate the noise and iteration errors for a fixed relaxation parameter. Albeit pessimistic, these estimates correctly describe the evolution of these errors as iterations progress.
- Choice of relaxation parameter $\lambda_k \Longrightarrow$ limit the noise error

One possible strategy: Choose $\lambda_0 = \lambda_1 = \sqrt{2}/\sigma^2$ and

$$\lambda_k = \frac{2}{\sigma^2}(1 - \zeta_k) \quad \text{or} \quad \lambda_k = \frac{2}{\sigma^2}\frac{1 - \zeta_k}{(1 - \zeta_k^k)^2} \quad \text{for } k = 2, 3, \ldots$$

Here, σ is an estimate of the largest singular value of $\mathbf{A}$ and $0 < \zeta_k < 1$ is the unique root of the polynomial

$$p_{k-1}(\zeta) := (2k-1)\zeta^{k-1} - (\zeta^{k-2} + \ldots + \zeta + 1).$$

Leads to diminishing step-size.

Choosing the regularisation parameter(s)

The relaxation parameter

- ART and SIRT methods: Possible to estimate the noise and iteration errors for a fixed relaxation parameter. Albeit pessimistic, these estimates correctly describe the evolution of these errors as iterations progress.
- Choice of relaxation parameter $\lambda_k \Longrightarrow$ limit the noise error

One possible strategy: Choose $\lambda_0 = \lambda_1 = \sqrt{2}/\sigma^2$ and

$$\lambda_k = \frac{2}{\sigma^2}(1-\zeta_k) \quad \text{or} \quad \lambda_k = \frac{2}{\sigma^2}\frac{1-\zeta_k}{(1-\zeta_k^k)^2} \quad \text{for } k = 2, 3, \ldots$$

Here, σ is an estimate of the largest singular value of $\mathbf{A}$ and $0 < \zeta_k < 1$ is the unique root of the polynomial

$$p_{k-1}(\zeta) := (2k-1)\zeta^{k-1} - (\zeta^{k-2} + \ldots + \zeta + 1).$$

Leads to diminishing step-size.

- Common to stop the iterations in iterative methods when the residual norm $\|\mathbf{A} \cdot \boldsymbol{\alpha}_k - g\|_2$ is "sufficiently small" since this may imply that $\boldsymbol{\alpha}_k$ is close to a least-squares solution.
- Not good for ill-posed problems since a least-squares solution is probably useless as it has overfitting artefacts.
- Stopping rules regulate the iteration error.

Principle: Assume the residual norm decreases monotonically with the iterates. Then, stop iterates when the difference to data is smaller than size of the data noise.

- An estimate $\delta > 0$ of the size of the noise component in data, *i.e.*, $\|g_{\text{noise}}\|_2 < \delta$.
- $\tau > 1$, a safety factor.

Find k such that

$$\left\|\mathbf{A} \cdot \alpha_{k+1} - g\right\|_2 \leq \tau\delta \leq \left\|\mathbf{A} \cdot \alpha_k - g\right\|_2.$$

Properties:

- Unique solution for regularisation parameter k since residual norm varies monotonically with the iterates.
- Relies on a good estimate δ of the size of the noise in data, which may be difficult to obtain in practice.
- Computed regularisation parameter k is very sensitive to the accuracy of the estimate δ. A too small estimate can lead to dramatic under-smoothing (because k is chosen too large).

Choosing the Regularisation Parameter
Stopping rules: The discrepancy criterion

Principle: Assume the residual norm decreases monotonically with the iterates. Then, stop iterates when the difference to data is smaller than size of the data noise.

- An estimate $\delta > 0$ of the size of the noise component in data, *i.e.*, $\|g_{\text{noise}}\|_2 < \delta$.
- $\tau > 1$, a safety factor.

Find k such that

$$\left\|\mathbf{A} \cdot \boldsymbol{\alpha}_{k+1} - g\right\|_2 \leq \tau\delta \leq \left\|\mathbf{A} \cdot \boldsymbol{\alpha}_k - g\right\|_2.$$

Properties:

- Unique solution for regularisation parameter k since residual norm varies monotonically with the iterates.
- Relies on a good estimate δ of the size of the noise in data, which may be difficult to obtain in practice.
- Computed regularisation parameter k is very sensitive to the accuracy of the estimate δ. A too small estimate can lead to dramatic under-smoothing (because k is chosen too large).

Choosing the Regularisation Parameter

Stopping rules: The discrepancy criterion

Principle: Assume the residual norm decreases monotonically with the iterates. Then, stop iterates when the difference to data is smaller than size of the data noise.

- An estimate $\delta > 0$ of the size of the noise component in data, *i.e.*, $\|g_{\text{noise}}\|_2 < \delta$.
- $\tau > 1$, a safety factor.

Find k such that

$$\left\|\mathbf{A} \cdot \boldsymbol{\alpha}_{k+1} - g\right\|_2 \leq \tau\delta \leq \left\|\mathbf{A} \cdot \boldsymbol{\alpha}_k - g\right\|_2.$$

Properties:

- Unique solution for regularisation parameter k since residual norm varies monotonically with the iterates.
- Relies on a good estimate δ of the size of the noise in data, which may be difficult to obtain in practice.
- Computed regularisation parameter k is very sensitive to the accuracy of the estimate δ. A too small estimate can lead to dramatic under-smoothing (because k is chosen too large).

Principle: For small k smaller than some threshold, the iteration error dominates in $\boldsymbol{\alpha}_k$ so the 2-norm $\|\boldsymbol{\alpha}_k\|_2$ is expected to be small while the residual norm $\left\|\mathbf{A}\cdot\boldsymbol{\alpha}_k - g\right\|_2$ is large.

- $\|\boldsymbol{\alpha}_k\|_2$ is almost a constant given by $\|\boldsymbol{\alpha}_{\mathrm{true}}\|_2$ except for very small k where $\|\boldsymbol{\alpha}_k\|_2$ gets smaller as $k \to 0$.
- The residual norm $\left\|\mathbf{A}\cdot\boldsymbol{\alpha}_k - g\right\|_2$ increases as $k \to 0$, until it reaches its maximum value at $k = 0$.

For k larger than some threshold the noise error dominates $\boldsymbol{\alpha}_k$ leading to the following k dependency:

- $\|\boldsymbol{\alpha}_k\|_2$ increases as k increases (overfitting).
- The residual norm $\left\|\mathbf{A}\cdot\boldsymbol{\alpha}_k - g\right\|_2$ stays almost constant at the noise level in data.

The curve

$$k \mapsto \Big(\|\boldsymbol{\alpha}_k\|_2, \big\| \mathbf{A} \cdot \boldsymbol{\alpha}_k - g \big\|_2 \Big)$$

is "L"-formed with two distinctly different parts:

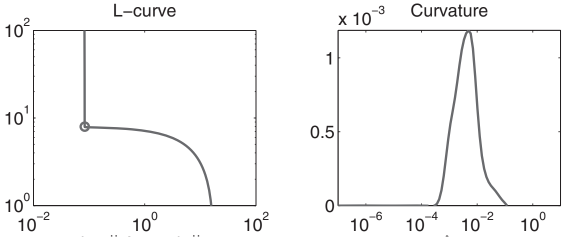
- Part where it is quite flat (when the noise error dominates)
- Part that is more vertical (when the iteration error dominates)

A log-log scale emphasizes the different characteristics of these two parts leading to the definition of the L-curve:

The L-curve:

$$k \mapsto \left(\log\big(\|\boldsymbol{\alpha}_k\|_2\big), \log\Big(\big\| \mathbf{A} \cdot \boldsymbol{\alpha}_k - g \big\|_2 \Big) \right)$$

L-curve criterion: Choose k that corresponds to the corner (point with highest curvature) of the L-curve.



Properties:

- Heuristic criteria with no guarantee that it will always produce a good regularisation parameter.
- Typically fails when the change in the residual and solution norms is small for two consecutive values of k.
- Computing the corner can be challenging, there may be many small local corners.

Choosing the Regularisation Parameter

Stopping rules: the generalised cross-validation criterion

Principle: Remove data and select the value for the regularisation parameter that minimises the error in predicting the removed data. Consider the reduced problem:

$$\mathbf{A}^i \cdot \boldsymbol{\alpha}_k^i = g^i \quad \text{for fixed } i = 1, \ldots, m.$$

- $g^i \in \mathbb{R}^{m-1}$ is the data after we leave out the i:th data point.
- $\mathbf{A}^i$ is the $\mathbf{A}$ with the i:th row left out
- $\boldsymbol{\alpha}_k^i \in \mathbb{R}^n$ is the reconstruction obtained after k iterates when solving the reduced problem above.

Use $\boldsymbol{\alpha}_k^i \in \mathbb{R}^n$ and the i:th row a_i of $\mathbf{A}$ to predict i:th data element:

$$g_i^{\text{predict}} := a_i \cdot \boldsymbol{\alpha}_k^i.$$

Prediction error $= \left| g_i^{\text{predict}} - g_i \right| = \left| a_i \cdot \boldsymbol{\alpha}_k^i - g_i \right|$.

Choosing the Regularisation Parameter

Stopping rules: the generalised cross-validation criterion

Principle: Remove data and select the value for the regularisation parameter that minimises the error in predicting the removed data. Consider the reduced problem:

$$\mathbf{A}^i \cdot \boldsymbol{\alpha}_k^i = g^i \quad \text{for fixed } i = 1, \ldots, m.$$

- $g^i \in \mathbb{R}^{m-1}$ is the data after we leave out the i:th data point.
- $\mathbf{A}^i$ is the $\mathbf{A}$ with the i:th row left out
- $\boldsymbol{\alpha}_k^i \in \mathbb{R}^n$ is the reconstruction obtained after k iterates when solving the reduced problem above.

Choose the regularisation parameter k (number of iterations) such that the total prediction error is minimised, *i.e.*, solve

$$\min_k \sum_{i=1}^{m} \Big(g_i^{\text{predict}} - g_i \Big)^2 = \min_k \sum_{i=1}^{m} \Big(a_i \cdot \boldsymbol{\alpha}_k^i - g_i \Big)^2.$$

Computationally unfeasible since m different reconstruction problems are involved $\Longrightarrow$ need to simplify above minimisation.

Choosing the Regularisation Parameter

Stopping rules: the generalised cross-validation criterion

Principle: Remove data and select the value for the regularisation parameter that minimises the error in predicting the removed data. Consider the reduced problem:

$$\mathbf{A}^i \cdot \boldsymbol{\alpha}_k^i = g^i \quad \text{for fixed } i = 1, \dots, m.$$

- $g^i \in \mathbb{R}^{m-1}$ is the data after we leave out the i:th data point.
- $\mathbf{A}^i$ is the $\mathbf{A}$ with the i:th row left out
- $\boldsymbol{\alpha}_k^i \in \mathbb{R}^n$ is the reconstruction obtained after k iterates when solving the reduced problem above.

Properties:

- Quite robust and accurate, as long as the noise is white.
- Occasional failure of GCV is well understood, and it often reveals itself by the ridiculous under-smoothing it leads to.
- Statistical and asymptotic properties is very well understood.
- Computationally demanding.

Methods for choosing the regularisation parameter
Summary

- The discrepancy principle is a simple method that seeks to reveal when the residual vector is noise-only. It relies on a good estimate of the size of the noise in data which may be difficult to obtain in practice.
- The L-curve criterion is based on an intuitive heuristic and seeks to balance the two error components via inspection (manually or automated) of the L-curve. This method fails when the solution is very smooth.
- The GCV criterion seeks to minimise the prediction error, and it is often a very robust method – with occasional failure, often leading to ridiculous under-smoothing that reveals itself.

Methods for choosing the regularisation parameter

Summary

- The discrepancy principle is a simple method that seeks to reveal when the residual vector is noise-only. It relies on a good estimate of the size of the noise in data which may be difficult to obtain in practice.
- The L-curve criterion is based on an intuitive heuristic and seeks to balance the two error components via inspection (manually or automated) of the L-curve. This method fails when the solution is very smooth.
- The GCV criterion seeks to minimise the prediction error, and it is often a very robust method – with occasional failure, often leading to ridiculous under-smoothing that reveals itself.

- The discrepancy principle is a simple method that seeks to reveal when the residual vector is noise-only. It relies on a good estimate of the size of the noise in data which may be difficult to obtain in practice.
- The L-curve criterion is based on an intuitive heuristic and seeks to balance the two error components via inspection (manually or automated) of the L-curve. This method fails when the solution is very smooth.
- The GCV criterion seeks to minimise the prediction error, and it is often a very robust method – with occasional failure, often leading to ridiculous under-smoothing that reveals itself.

- Imaging, in particular 3D tomography, often leads to solving large-scale linear inverse problems.
- A useful reconstruction method must avoid factorisation of the measurement matrix:
  - The main "building blocks" must be matrix-vector multiplications, avoiding any factorization of the measurement matrix.
  - Allow the user to select regularisation parameter(s) via a parameter-choice method that does not require solving the reconstruction problem from scratch for each new parameter.

- Imaging, in particular 3D tomography, often leads to solving large-scale linear inverse problems.
- A useful reconstruction method must avoid factorisation of the measurement matrix:
  - The main “building blocks” must be matrix-vector multiplications, avoiding any factorization of the measurement matrix.
  - Allow the user to select regularisation parameter(s) via a parameter-choice method that does not require solving the reconstruction problem from scratch for each new parameter.

- Focus on a single application, or a specific and narrow class of applications; no reconstruction method is guaranteed to work for a broad class of problems.
- When implementing the reconstruction method, focus on modularity and clarity of the computer code; it is guaranteed that you need to go back and modify/expand the software at some point in time.
- Make sure you understand the performance of the implementation, including computing times, storage requirements, etc.

- When testing the reconstruction method, make sure to generate test problems that reflect as many aspects as possible of real, measured data.
- When testing, also make sure to model the noise as realistically as possible, and use realistic noise levels.
- Be aware of the concept of "inverse crime" (same ingredients are used to create synthetic data and to recover image):
  1. As a "proof-of-concept" first use tests that commit inverse crime; if the reconstruction method does not work under such circumstances, it can never work.
  2. Next, in order to check the robustness to model errors, test the reconstruction method without committing inverse crime.

- Carefully evaluate the regularised solutions; consider which characteristics are important, and use the appropriate measure of the error (the 2-norm between the exact and regularised solutions is not always the optimal measure).
- Using the same exact data, create many realizations of the noise and perform a systematic study the robustness of the reconstruction method. Use histograms or other tools to investigate if the distribution of the errors has an undesired tail.

Iterative methods produce a sequence of digital images in $\mathbb{R}^n$

$$\boldsymbol{\alpha}_0 \to \boldsymbol{\alpha}_1 \to \boldsymbol{\alpha}_2 \to \dots$$

Important properties

- Iterates designed to converge to a least-squares solution of $\mathbf{A} \cdot \boldsymbol{\alpha} = g$.
- Semi-convergence, so initial convergence towards $\boldsymbol{\alpha}_{\text{true}}$ followed by (slow) convergence to least-squares solution.
  $\Longrightarrow$ Iteration number is a regularisation parameter.

Iterative methods produce a sequence of digital images in $\mathbb{R}^n$

$$\boldsymbol{\alpha}_0 \to \boldsymbol{\alpha}_1 \to \boldsymbol{\alpha}_2 \to \dots$$

Advantages

- Works with any linear forward problem.
- Only uses matrix-vector multiplications, so the matrix $\mathbf{A}$ is only accessed via matrix-vector multiplications and not explicitly required and never altered.
  $\Longrightarrow$ Enough to have a "black box" software component for computing the action of $\mathbf{A}$ and $\mathbf{A}^{\mathrm{t}}$.
- Atomic operations in iterative methods (mat-vec product, norm) suited for high-performance computing.
- Often produce a natural sequence of regularised solutions; stop when the solution is "satisfactory" (parameter choice).