ART Performance

Per Christian Hansen

joint work with, among others

Tommy Elfving

Touraj Nikazad

Hans Henrik B. Sørensen

HDtomo
HIGH - DEFINITION TOMOGRAPHY

erc European Research Council

DTU Compute

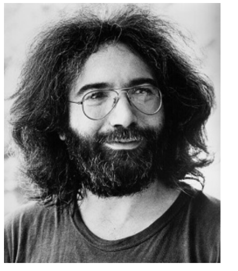
Department of Applied Mathematics and Computer Science

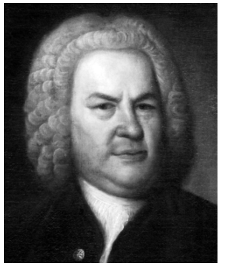
ART = Algebraic Reconstruction Technique
= A Classical Algorithm

Perspective:

Listen to Grateful Dead (1965–1995) $\rightarrow$ old fashioned.
Listen to Mozart (1756–91) or Bach (1685–28) $\rightarrow$ the classics!

Talk about total variation (1992) $\rightarrow$ old stuff.
Talk about ART (1937) $\rightarrow$ classical algorithm.





Our motivation: solve linear systems of equations $\boxed{A\,x = b}$ derived from discretization of an underlying tomography problem.

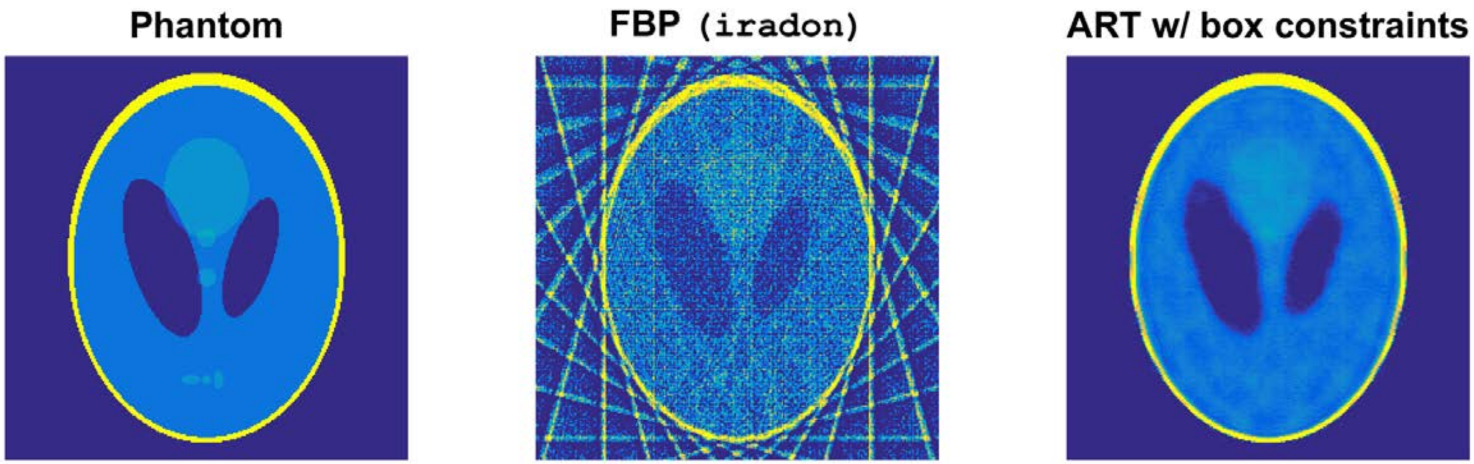
In this talk we do not pay attention to the discretization method.

ART is a simple iterative method for solving $A\,x = b$ where each iteration updates x via sweeps over the rows a_i^T of the matrix $A \in \mathbb{R}^{m\times n}$.

Filtered Back Projection (FBP) versus ART

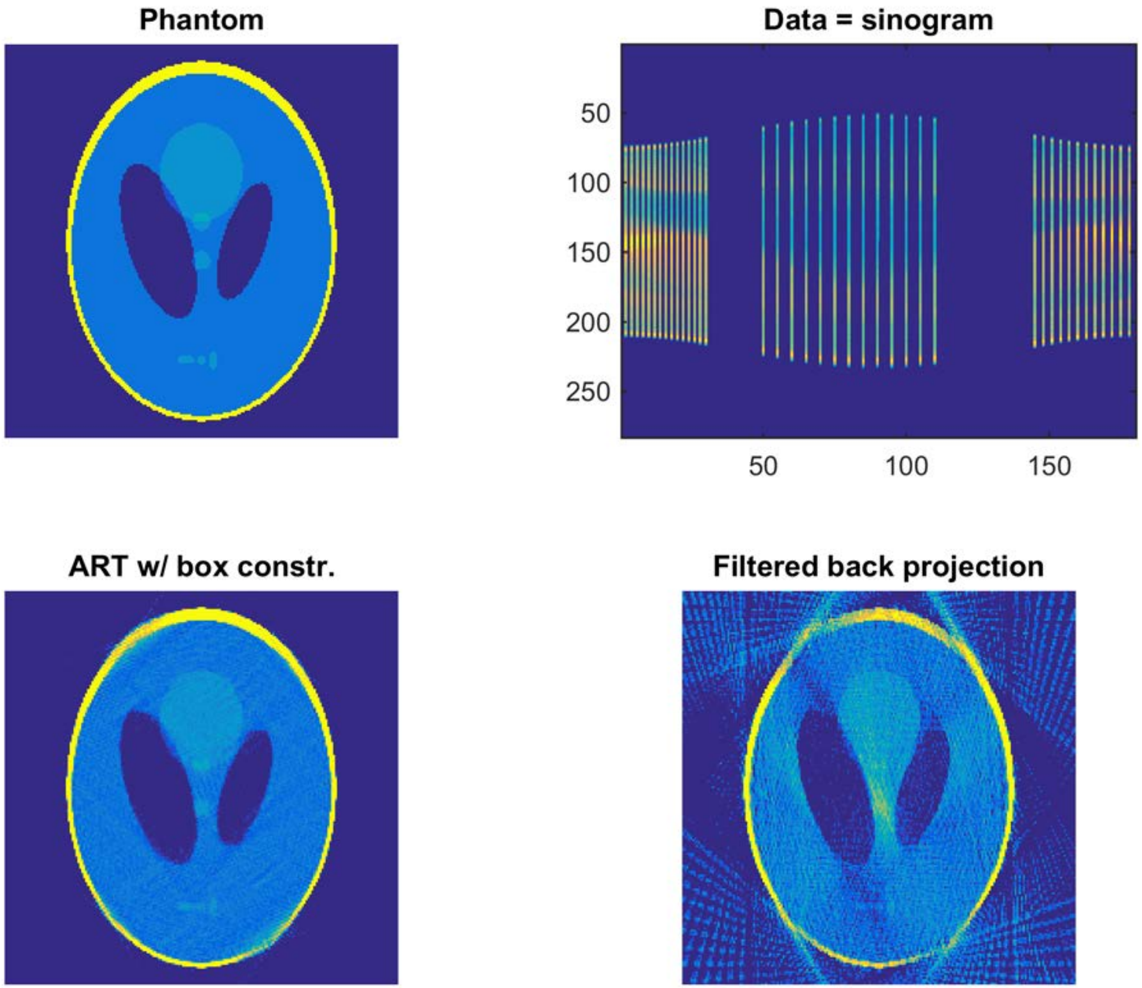
- FBP: low memory, works really well with many data.
- But *artifacts* appear with limited data, or nonuniform distribution of projection angles or ray.
- Difficult to incorporate constraints (e.g., nonnegativity) in FBP
- ART and other algebraic methods are more flexible and adaptive.

Example with 3% noise and projection angles $15°, 30°, \ldots, 180°$.



FBP versus ART – Limited Data

Irregularly spaced angles / “missing” angles also cause difficulties for FBP



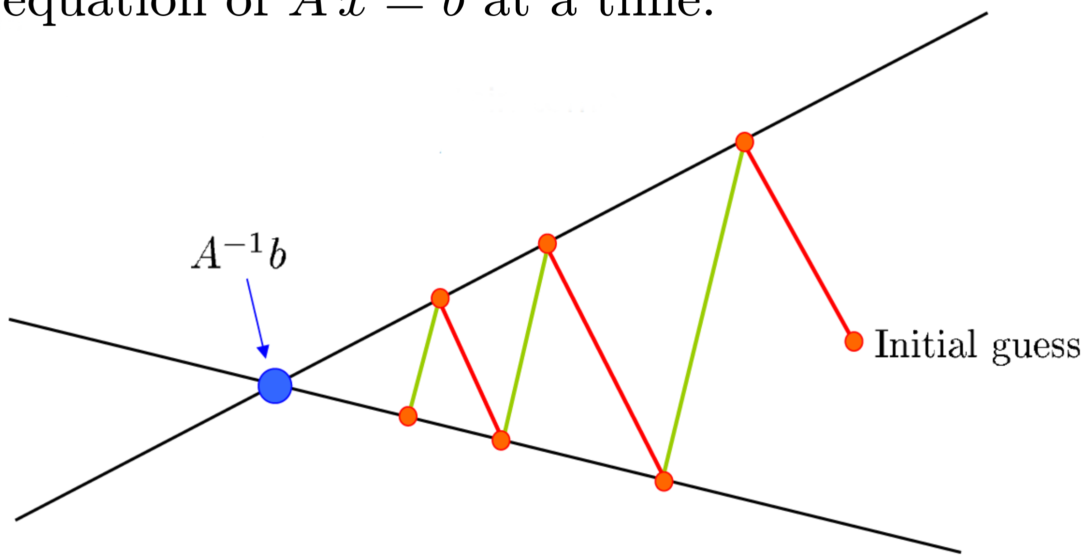
ART History

Kaczmarz (1937): orthogonally project x on the hyperplane defined by the ith row a_i^T and the corresponding element b_i of the right-hand side:

$$x \leftarrow \mathcal{P}_i\, x = x + \frac{b_i - a_i^T x}{\|a_i\|_2^2}\, a_i \ , \qquad i = 1, 2, \ldots, m \ .$$

Gordon, Bender, Herman (1970): coined the term "ART" and also introduced a

Satisfy one equation of $A\,x = b$ at a time:



x … n .

Herman, L… …eters $\omega_k < 2$:

Today ART … convex set:

$$x \leftarrow \mathcal{P}_{\mathcal{C}}\left(x + \omega_k\, \frac{b_i - a_i^T x}{\|a_i\|_2^2}\, a_i \right) \ , \qquad i = 1, 2, \ldots, m \ .$$

The Optimization Viewpoint

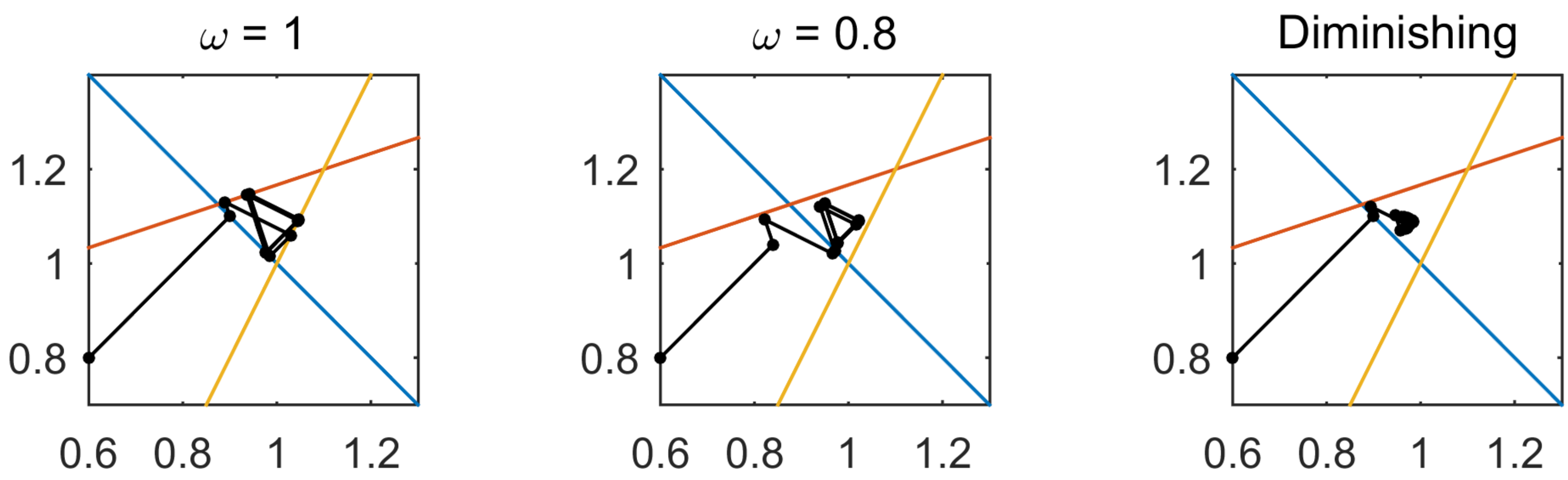
ART is usually considered as a solver for $A\,x = b$; but it is often more convenient to consider it as an **optimization method**.

- We can introduce a *relaxation parameter* – or step length parameter – in the algorithm which controls the "size" of the updating and, as a consequence, the convergence of the method:
  - a constant ω, or
  - a parameter ω_k that changes with the iterations.
- In each updating step we can incorporate a *projection* $\mathcal{P}_\mathcal{C}$ on a suitably chosen convex set $\mathcal{C}$ that reflects prior knowledge.
- We can view it as a projected incremental gradient optimization method, which opens for further extensions and careful convergence analysis.

Iteration-Dependent Relax. Parameter

For inconsistent systems, basic ART with a fixed relaxation parameter ω gives cyclic and non-convergent behavior.

With the *diminishing relaxation parameter* $\omega_k = 1/\sqrt{k} \to 0$ as $k \to \infty$ the iterates converge to a weighted least squares solution.



There is also a *column version* of ART which always converges to the standard least squares solution.

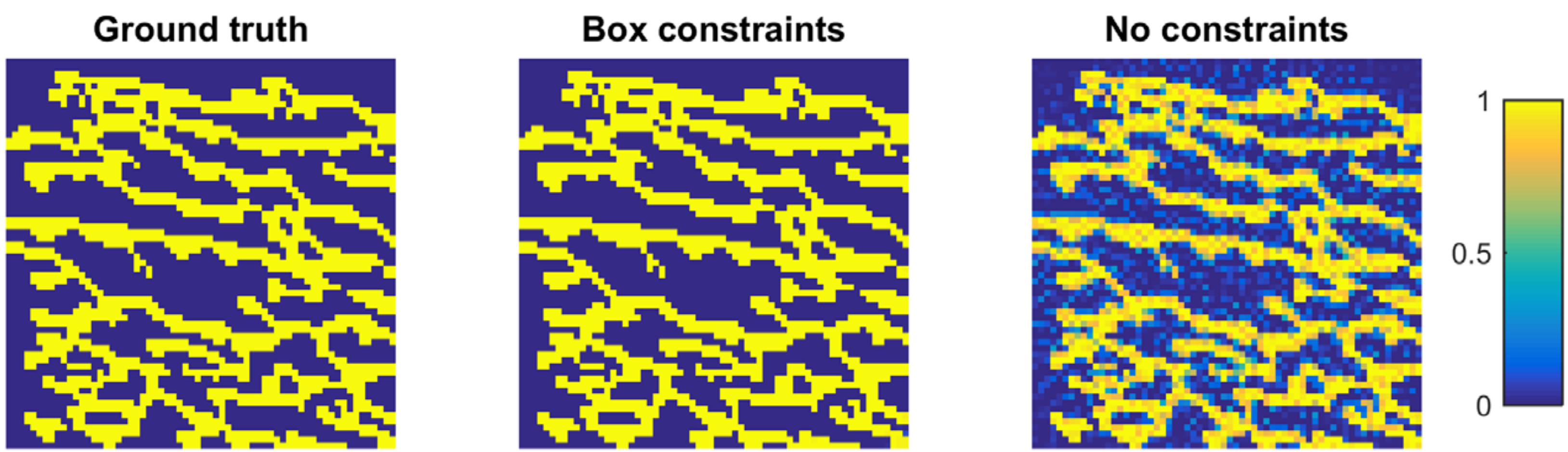
Simple Constraints

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

$$x_i \geq 0, \qquad i = 1, 2, \ldots, n.$$

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

$$0 \leq x_i \leq 1, \qquad i = 1, 2, \ldots, n.$$



A Projected Incremental Gradient Method

Consider the constrained least squares problem

$$\min_x \, {}^1\!/\!_2 \|b - A\,x\|_2^2 \qquad \text{subject to} \qquad x \in \mathcal{C}$$

and write the objective function as ${}^1\!/\!_2 \|b - A\,x\|_2^2 = \sum_{i=1}^n f_i(x)$ with

$$f_i(x) = {}^1\!/\!_2 \frac{(b_i - a_i^T x)^2}{\|a_i\|_2^2} \qquad \Rightarrow \qquad \nabla f_i(x) = -\frac{b_i - a_i^T x}{\|a_i\|_2^2}$$

Incremental gradient methods use only the gradient of a singe term $f_i(x)$ in each iteration, leading to the ART update:

$$x \leftarrow \mathcal{P}_{\mathcal{C}}\left(x + \omega_k \, \frac{b_i - a_i^T x}{\|a_i\|_2^2} \, a_i \right) \ , \qquad i = 1, 2, \ldots, m \ .$$

Software for ART

- **SNARK09**: C++ package from NYU; 2D reconstructions.
  `www.dig.cs.gc.cuny.edu/software/snark09`
- **ASTRA**: MATLAB package with GPU accelleration and interfact to Python from Univ. of Antwerp + CWI, Amsterdam; 2D and 3D reconstructions.
  `sourceforge.net/p/astra-toolbox/wiki/Home`
- **Image reconstruction toolbox**: MATLAB package from Univ. of Michigan; 2D reconstructions.
  `web.eecs.umich.edu/~fessler/code`
- **AIR Tools**: MATLAB package from DTU; 2D reconstructions.
  `www.compute.dtu.dk/~pcha/AIRtools`
- **Xmipp**: C++ package from the Spanish National Biotechnology Centre; 3D electron microscopy.
  `xmipp.cnb.csic.es/twiki/bin/view/Xmipp/WebHome`

ART Academy

ART is a rich source for research problems!

This list is quite biased towards my own work with AIR Tools.

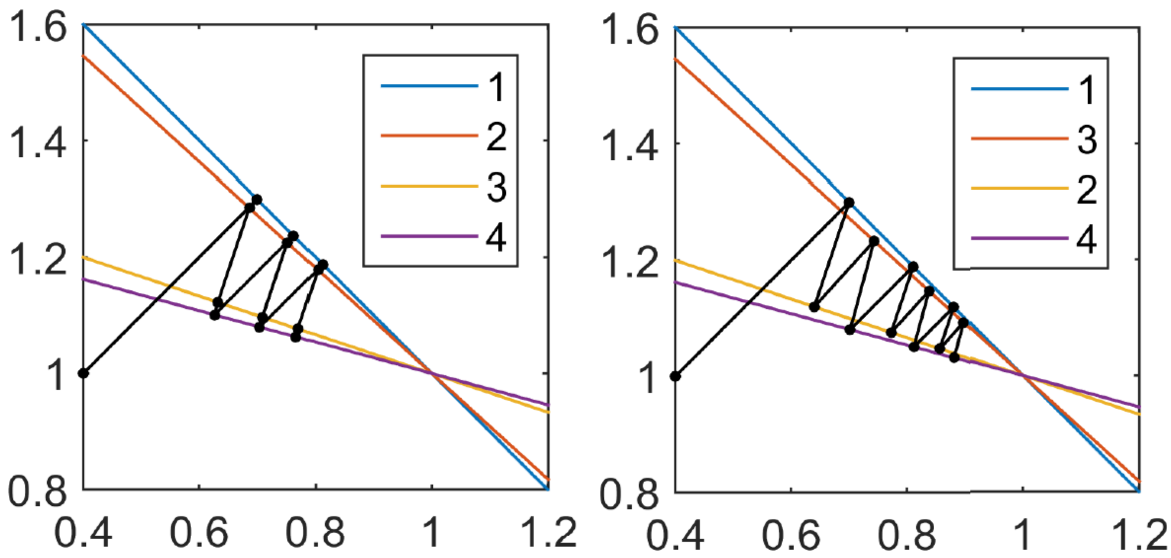
- Convergence and semi-convergence.
- Block algorithms.
- Efficient implementation.
- Choice of relaxation parameter.
- Stopping rules.
- Variations and extensions ART, e.g., for Poisson noise.
- Column version of ART.

This presentation:
ART performance

Convergence Issues

The convergence of ART is quite obvious from the graph on slide 5 – but can we say more?

Difficulty: the *ordering* of the rows of A influence the convergence:



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

The ordering 1–3–2–4 er preferable and almost twice as fast.

Convergence of ART – No Noise

Assume that we *select the rows randomly*, that A is invertible, and that all rows of A are scaled to unit 2-norm. Then the expected value $\mathcal{E}(\cdot)$ of the error norm satisfies:

Strohmer & Vershynin, 2009

$$\mathcal{E}\left(\|x_k - \bar{x}\|_2^2\right) \square \left(1 - \frac{1}{n\,\kappa^2}\right)^k \|x_0 - \bar{x}\|_2^2, \quad k = 1, 2, \ldots$$

where $\bar{x} = A^{-1}b$ and $\kappa = \|A\|_2\,\|A^{-1}\|_2$. **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, corresp. to one sweep over all the rows of A, the factor is $1 - 1/\kappa^2$.

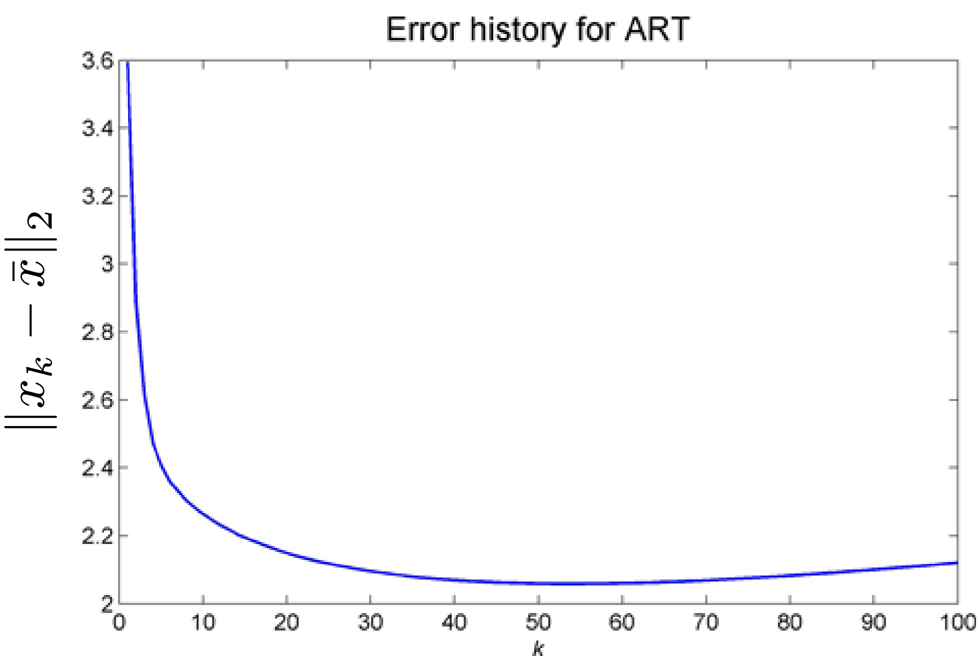
Note: there are often orderings for which the convergence is faster!

Semi-Convergence of ART – with Noise

Notation: $b = A\,\bar{x} + e$, $\bar{x}$ = exact solution, e = noise.

Initial iterations: the error $\|x_k - \bar{x}\|_2$ decreases.

Later: the error increases as $x_k \to A^{-1}b$.



A few references:

- F. Natterer, *The Mathematics of Computerized Tomography* (1986)
- A. van der Sluis & H. van der Vorst, *SIRT- and CG-type methods for the iterative solution of sparse linear least-squares problems* (1990)
- M. Bertero & P. Boccacci, *Inverse Problems in Imaging* (1998)
- M. Kilmer & G. W. Stewart, *Iterative Regularization And Minres* (1999)
- H. W. Engl, M. Hanke & A. Neubauer, *Regularization of Inverse Problems* (2000)

Part I: Semi-Convergence for ART

Elfving, H, Nikazad, *Semi-convergence properties of Kaczmarzs method*, Inverse Problems, 30 (2014), DOI: 10.1088/0266-5611/30/5/055007.

Recall that $\bar{x}$ = solution to noise-free problem, and let x_k and $\bar{x}_k$ denote the iterates when applying ART to b and $\bar{b} = A\,\bar{x}$:

$$\|x_k - \bar{x}\|_2 \;\square\; \|x_k - \bar{x}_k\|_2 + \|\bar{x}_k - \bar{x}\|_2 \ .$$

Noise error

Iteration error

Convergence theory for ART for noise-free data is well established and ensures that the iteration error $\bar{x}_k - \bar{x}$ goes to zero.

Our concern here is the noise error $e_k^{\mathrm{N}} = x_k - \bar{x}_k$. We wish to establish that it increases, and how fast.

Noise Error for ART – No Projection

Successive Over-Relaxation

ART is equivalent to applying SOR to $A\,A^T y = b$, $x = A^T y$. Splitting:

$$A A^T = L + D + L^T, \qquad M = (D + \omega L)^{-1},$$

where L is strictly lower triangular and $D = \mathrm{diag}(\|a_i\|_2^2)$. Then:

$$\boxed{x_{k+1} = x_k + \omega A^T M\,(b - A\,x_k)\ .}$$

We introduce: $\quad e = b - \bar{b} = \text{noise in data}, \quad Q = I - \omega A^T M\,A.$

Then simple manipulations show that the noise error is given by

$$e_k^{\mathrm{N}} = x_k - \bar{x}_k = Q\ e_{k-1}^{\mathrm{N}} + \omega A^T M\,e = \omega \sum_{j=1}^{k-1} Q^j A^T M\,e\ .$$

After some work (see the paper) we obtain the bound

$$\boxed{\|e_k^{\mathrm{N}}\|_2 \approx k\,\omega\,\|A^T M\,e\|_2.}$$

Noise Error Analysis – A Tighter Bound

Further analysis (see the paper) shows that the noise error in ART is bounded above as:

$$
\begin{aligned}
\|e_k^{\mathrm{N}}\|_2 \quad &\square \quad \frac{1-(1-\omega\sigma_{\min}^2)^k}{\sigma_{\min}} \, \frac{\|A^T M\, e\|_2}{\sigma_{\min}} + \mathcal{O}(\sigma_{\min}^2), \\
\sigma_{\min} \quad &= \quad \text{smallest singular value of } A.
\end{aligned}
$$

As long as $\omega\sigma_{\min}^2 < 1$ we have

$$
\frac{1-(1-\omega\sigma_{\min}^2)^k}{\sigma_{\min}} \ \square \ \sqrt{k}\,\sqrt{\omega}
$$

and thus

$$
\boxed{\|e_k^{\mathrm{N}}\|_2 \ \square \ \sqrt{k}\, \frac{\sqrt{\omega}\,\|A^T M e\|_2}{\sigma_{\min}} + \mathcal{O}(\sigma_{\min}^2).}
$$

This also holds for *projected* ART provided that A and $\mathcal{P}_{\mathcal{C}}$ satisfy

$$
y \in \mathcal{R}(A^T) \Rightarrow \mathcal{P}_{\mathcal{C}}\, y \in \mathcal{R}(A^T).
$$

Numerical Results (`paralleltomo` from AIR Tools)

The point of **semi-convergence** arises when noise error ≈ iteration error.

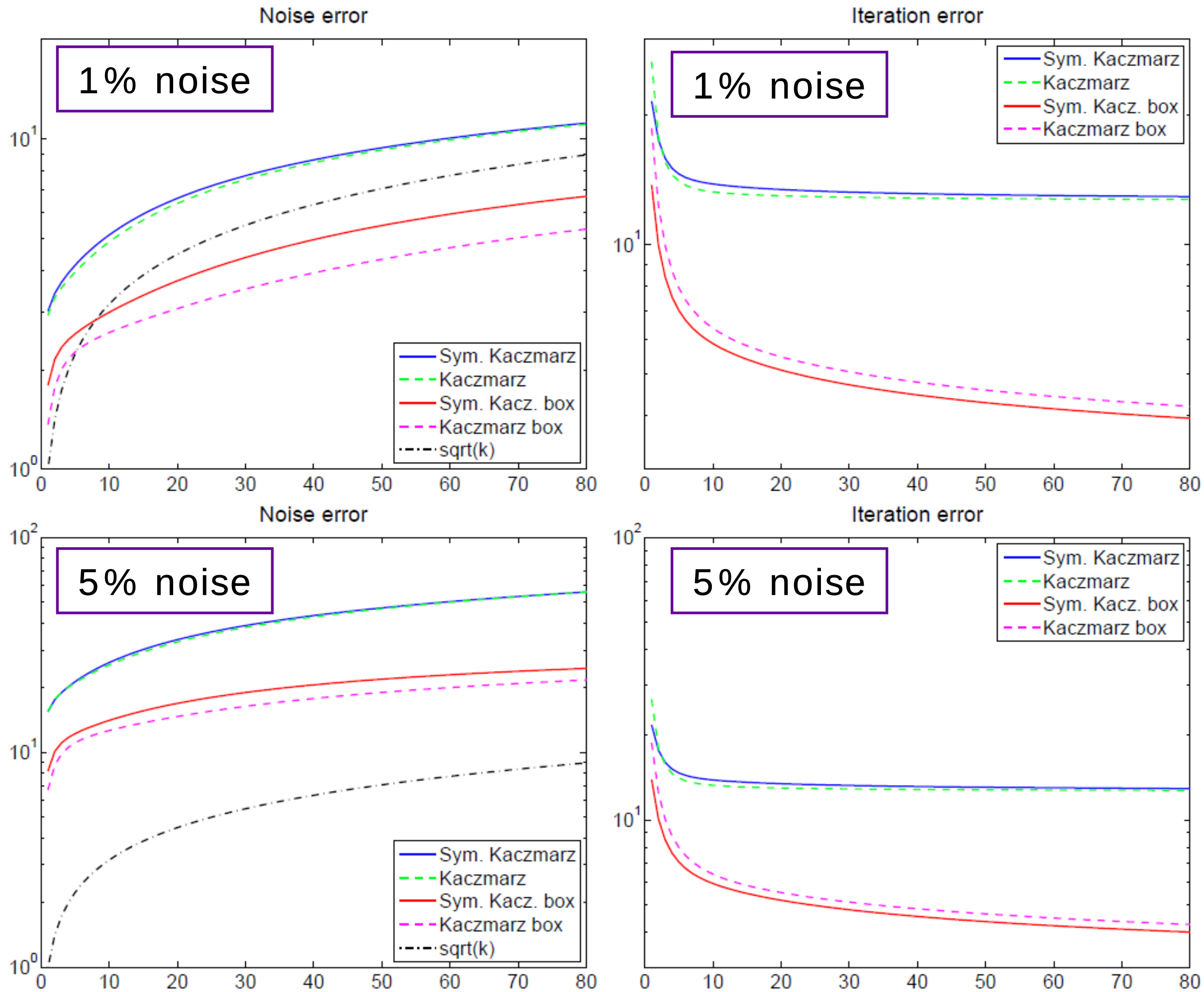
Test problem:

- 200×200 phantom,
- 60 projections at
- 3°,6°,9°,...,180°,
- *m* = 15,232,
- *n* = 40,000.

We estimate

$$\frac{\sqrt{\omega}\|A^T M e\|_2}{\sigma_{\min}} \approx 10^7.$$

Hence our bound is a wild over-estimate but it correctly *tracks* the noise error.



Conclusion so Far

We derived an upper bound for the noise error that supports the observed semi-convergence of ART.

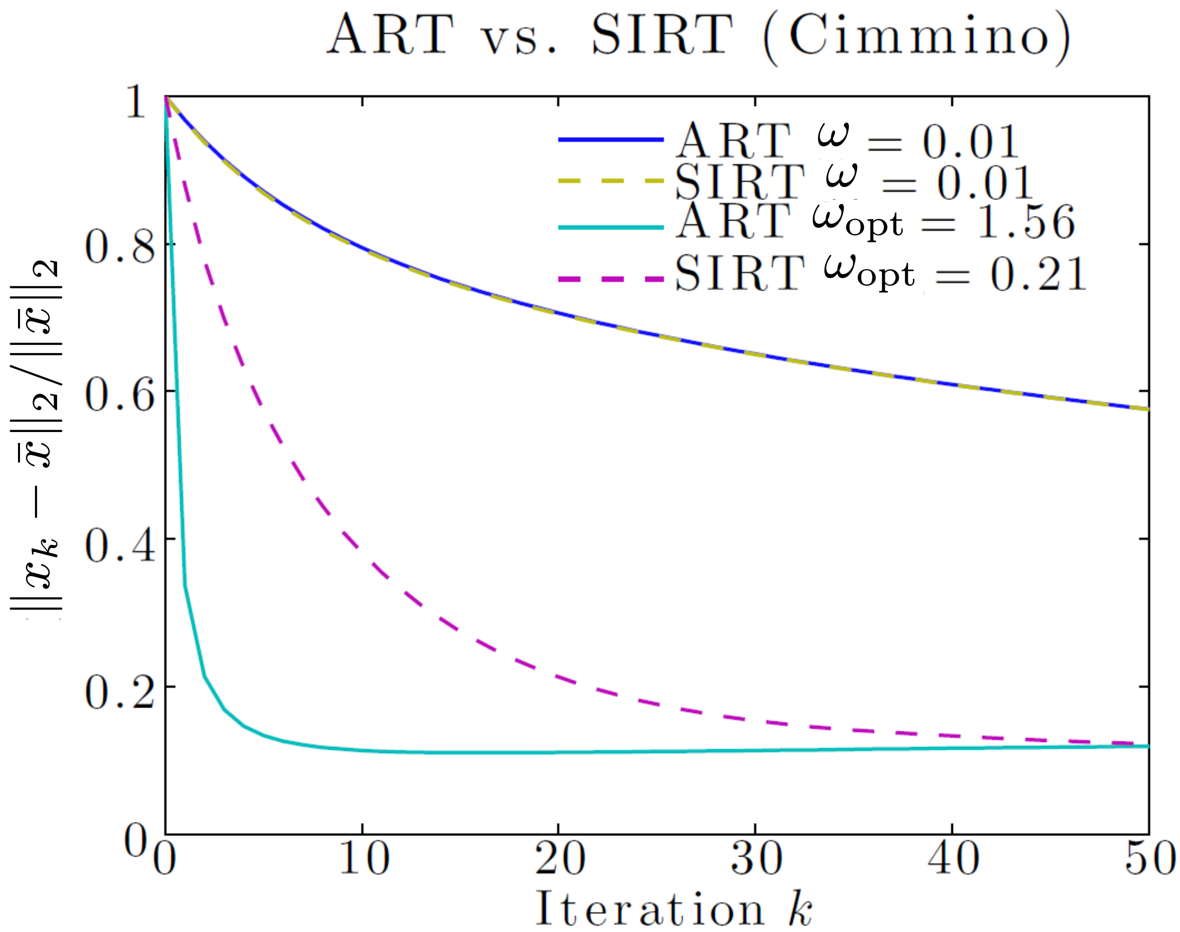
The bound includes a crazy large factor – how to get rid of it?

We would also like to have a lower bound – statisticial analysis?

Part II: Performance Issues

In these numerical experiments we compute and store A explicitly!

SIRT (Cimmino): $x \leftarrow \mathcal{P}_{\mathcal{C}}\big(x + \omega\, A^T D^{-2}(b - A\,x)\big), \quad D = \mathrm{diag}(\|a_i\|_2)$



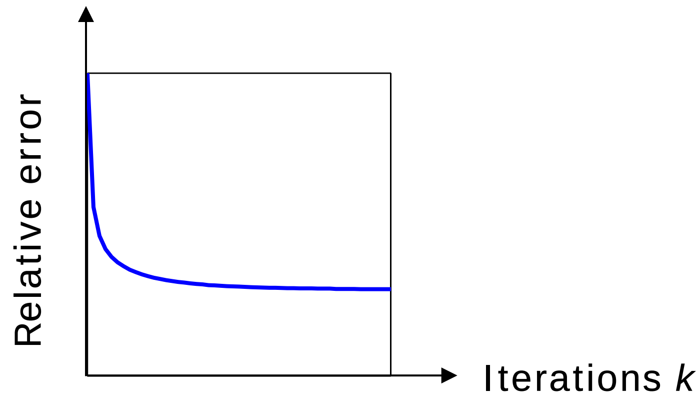
$\omega_{\rm opt}$ gives fastest semi-convergence.

SIRT: slow convergence.

ART can converge a lot *faster* than SIRT.

Performance
1 core

$$\|x_k - \bar{x}\|_2 / \|\bar{x}\|_2$$



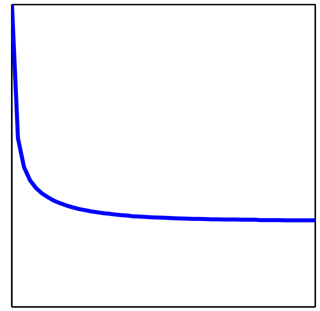
$m \times n$	ART t/iter
$13 \cdot 128^2 \times 64^3$	0.08 s
$13 \cdot 256^2 \times 128^3$	0.93 s
$13 \cdot 512^2 \times 256^3$	10.8 s

Intel Xeon E5620
2.40 GHz (1 core)

Test Problem:

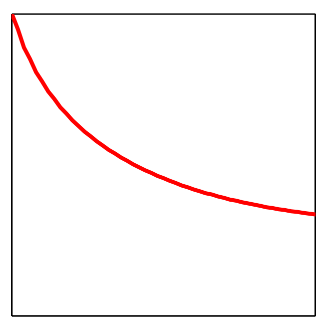
- Parallel-beam tomography.
- 13 projections.
- 3D Shepp-Logan phantom, Schabel (2006).

Performance
1 core



$m \times n$	ART t/iter	SIRT t/iter
$13 \cdot 128^2 \times 64^3$	0.08 s	0.08 s
$13 \cdot 256^2 \times 128^3$	0.93 s	1.02 s
$13 \cdot 512^2 \times 256^3$	10.8 s	14.7 s

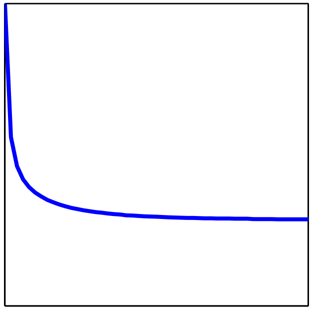
Intel Xeon E5620
2.40 GHz (1 core)



Same number of flops!

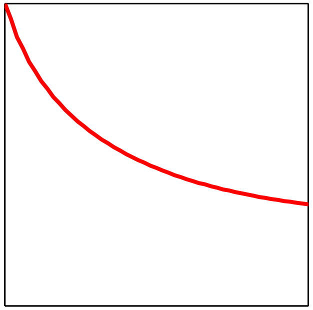
The difference is due to the cache: ART uses row a_i twice once it is loaded.

Performance
4 cores



$m \times n$	ART t/iter	SIRT 1 core t/iter	SIRT 4 cores t/iter
$13 \cdot 128^2 \times 64^3$	0.08 s	0.08 s	0.04 s
$13 \cdot 256^2 \times 128^3$	0.93 s	1.02 s	0.41 s
$13 \cdot 512^2 \times 256^3$	10.8 s	14.7 s	4.12 s

Intel Xeon E5620
2.40 GHz (4 cores)



Four cores are better suited for **block** matrix-vector operations.

Conclusion so Far

Our dilemma:

ART has *faster convergence* than SIRT – i.e., more reduction of the error per iteration.

SIRT can better take advantage of *multi-core architecture* than ART.

How to achieve the ”best of both worlds?” $\rightarrow$ Block methods!

- Y. Censor, *Parallel application of block-iterative methods in medical imaging and radiation therapy*, Math. Programming, 42 (1988), pp. 307–325.
- T. Elfving and T. Nikazad, *Properties of a class of block-iterative methods*, Inverse Problems, 25 (2009), 115011.
- M. Jiang and G. Wang, *Convergence studies on iterative algorithms for image reconstruction*, IEEE Trans. Medical Imaging, 22 (2003), pp. 569–579.
- H. H. B. Sørensen and P. C. Hansen, *Multi-core performance of block algebraic iterative reconstruction methods*, SIAM J. Sci. Comp., 36 (2014), pp. C524–C546.

Part III: Block Methods

$$A = \begin{pmatrix} A_1 \\ A_2 \\ \vdots \\ A_p \end{pmatrix}, \quad b = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_p \end{pmatrix}, \qquad A_\ell \in \mathbb{R}^{m_\ell \times n}, \quad \ell = 1, \ldots, p,$$

In each iteration we can:

- Treat *all blocks* sequentially or simultaneously (i.e., in parallel).
- Treat *each block* by an iterative method or by a direct computation.

We obtain several methods:

- ~~Sequential processing + ART on each block → classical ART~~
- Sequential processing + SIRT on each block
- Sequential processing + pseudoinverse of A_ℓ
- Parallel processing + ART on each block
- ~~Parallel processing + SIRT on each block → classical SIRT~~
- Parallel processing + pseudoinverse of A_ℓ

Block-*Sequential* Methods

SART: Andersen, Kak (1984)
Block-Iteration: Censor (1988)

Algorithm: Block-Sequential

Initialization: choose an arbitrary $x^0 \in \mathbb{R}^n$

Iteration: for $k = 0, 1, 2, \ldots$

$$x_k^0 = x_{k-1}$$
$$x_k^\ell = P\Big(x^{k,\ell-1} + \omega\, A_\ell^T\, M_\ell\, (b_\ell - A_\ell\, x_k^{\ell-1}) \Big)\,, \qquad \ell = 1, 2, \ldots, p$$
$$x_k = x_{k-1}^p$$

The convergence depends on the number of blocks *p*:

- If *p* = 1, we recover SIRT
- If *p* = *m*, we recover ART

Parallelism within each block of m/p rows

Variant by **Elfving** (1980): $M_\ell = (A_\ell A_\ell^T)^\dagger \;\Rightarrow\; A_\ell^T M_\ell = A_\ell^\dagger$

Block-*Parallel* Methods

Algorithm: Block-Parallel

Initialization: choose an arbitrary $x_0 \in \mathbb{R}^n$

Iteration: for $k = 0, 1, 2, \ldots$

for $\ell = 1, \ldots, p$ execute in parallel

$$x_k^\ell = \text{ART-sweep}(\omega, A_\ell, b_\ell, x_{k-1})$$

$$x_k = {}^1\!/_p \sum_{\ell=1}^p x_k^\ell .$$

String-Averaging:
Censor, Elfving, Herman (2001)

The convergence depends on *p*:

- If *p* = 1, we recover ART
- If *p* = *m*, we recover SIRT

Parallelism over the p blocks

Variants:

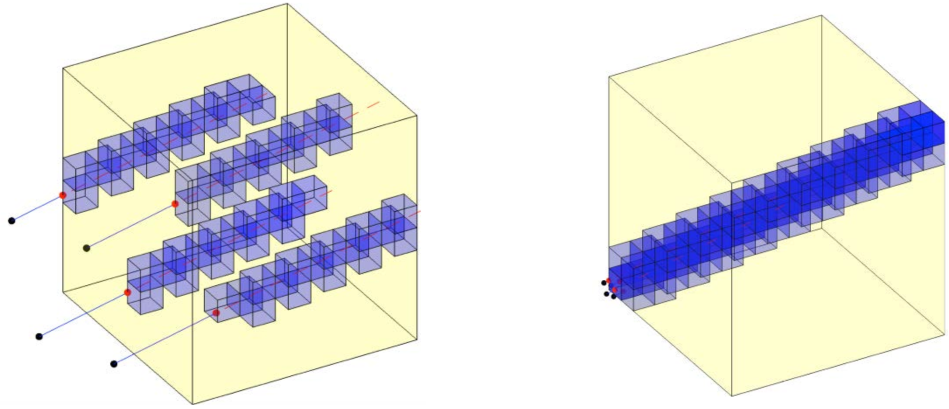
- **Elfving** (1980) – inner step: $x_k^\ell = P\big(x_{k-1}^\ell + \omega\, A_\ell^\dagger(b_\ell - A_\ell\, x_{k-}^\ell)\big)$
- **CARP** algorithm, Gordon & Gordon (2005):

$$x_k = \sum_{\ell=1}^p D_\ell\, x_k^{ell}, \qquad D_\ell \text{ depends on sparsity structure}$$

Blocks of Structurally Orthogonal Rows

Especially in *3D problems*, it is easy to find sets of rows that are orthogonal due to the structure of zeros/nonzeros.

Thus, a re-ordering of the rows can produce blocks with mutually orthogonal rows (= the traces of rays are non-overlapping).

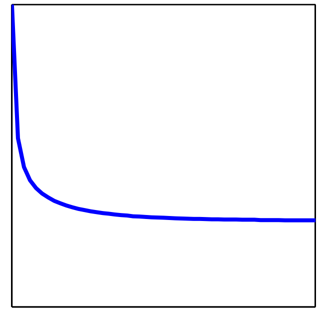


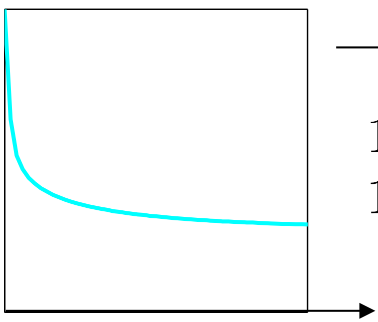
When a block has structurally orthogonal rows then ART, SIRT, and "Elfving" are *equivalent*. It is worthwhile to utilize this!

- **PART** algorithm, Gordon (2006)

Block *Sequential*

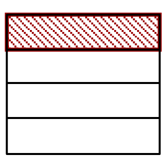
Intel Xeon
E5620
2.40 GHz
(4 cores)

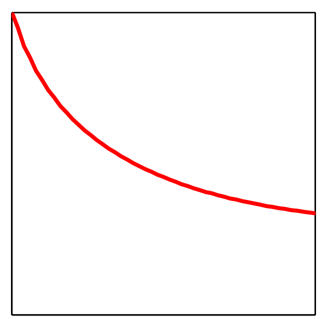




$m \times n$	ART t/iter	SIRT t/iter	Block Seq. t/iter
$13 \cdot 128^2 \times 64^3$	0.08 s	0.04 s	0.05 s
$13 \cdot 256^2 \times 128^3$	0.93 s	0.41 s	0.48 s
$13 \cdot 512^2 \times 256^3$	10.8 s	4.12 s	4.36 s

4 blocks

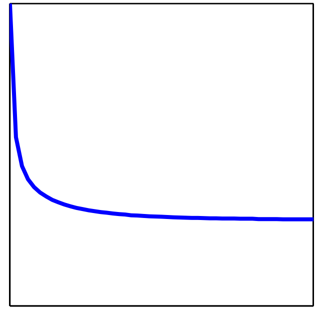




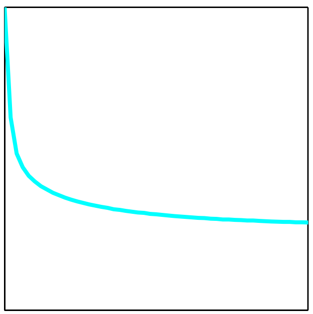
- The ”building blocks” are SIRT iterations, suited for multicore.
- The error reduction per iteration is close to that of ART.

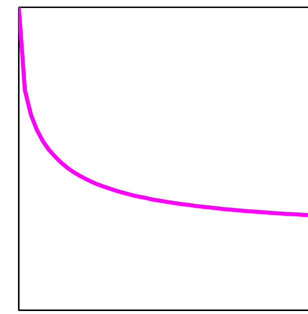
Block *Parallel*

Intel Xeon
E5620
2.40 GHz
(4 cores)

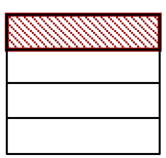


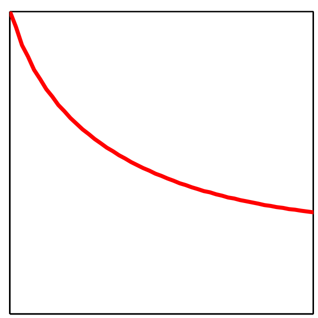
$m \times n$	ART t/iter	SIRT t/iter	Block Seq. t/iter	Block Par. t/iter
$13 \cdot 128^2 \times 64^3$	0.08 s	0.04 s	0.05 s	0.10 s
$13 \cdot 256^2 \times 128^3$	0.93 s	0.41 s	0.48 s	0.37 s
$13 \cdot 512^2 \times 256^3$	10.8 s	4.12 s	4.36 s	5.41 s





4 blocks

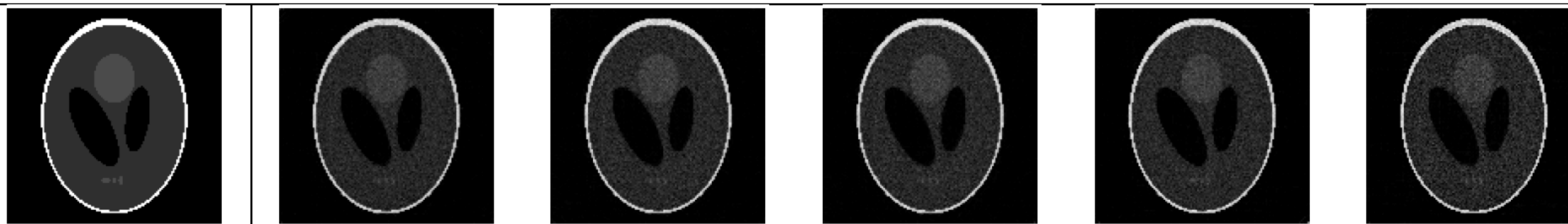




- The ”building blocks” are ART iterations, *not* suited for multicore.
- The error reduction per iteration is close to that of SIRT.

Multi-Core Results – 32 Cores

Multi-core: 32 cores



Method	Block-Seq	Block-Par	CARP	PART	ART
Blocks	64	2	4	460	115
Iterations	2	2	3	2	2
Time (s)	4.77	5.98	7.60	**2.50**	11.29

4 socket AMD Opteron 6282 SE
2.60 GHz (32 cores)

With many cores, and *A* stored explicitly, PART is a clear winner.

Block-Seq: block-sequential-SIRT
Block-Par: block-parallel-ART (Censor, Elfving, Herman)
CARP: block-parallel-ART (Gordon, Gordon)
PART – utilizes struct. orthog.
ART (1 thread)

256^3 voxels
133 projections of
256×256 pixels

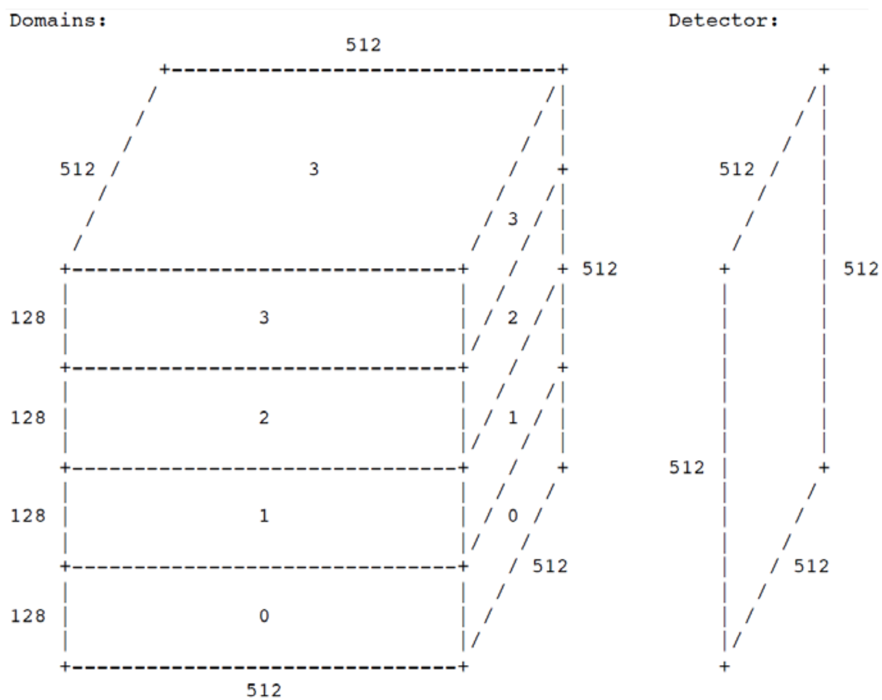
Conclusions so Far

- Block algorithms achieve *semi-convergence rate* similar to that of ART,
- with the *smaller computing time* of SIRT, because we can utilize the multicore architecture.
- With a suitable row ordering and choice of blocks, we can produce blocks of structurally orthogonal rows.
- PART has identical convergence to ART and very good scaling properties in practice.

Matrix-Free GPU Implementation

Size of 3D solution: $N \times N \times N$. Projections: $400 \times N \times N$.

Domain decomposition (example with N = 512):



One domain for each GPU.

GPU Results

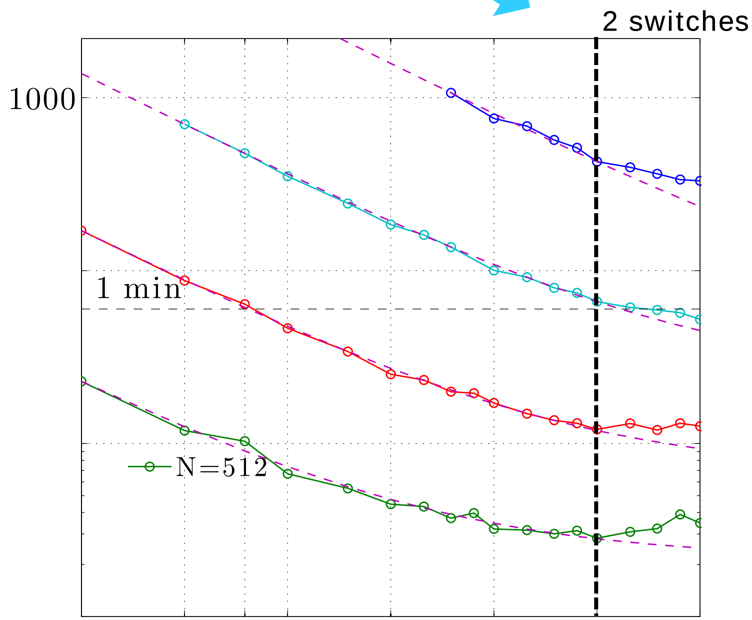
Multi-GPU – K80

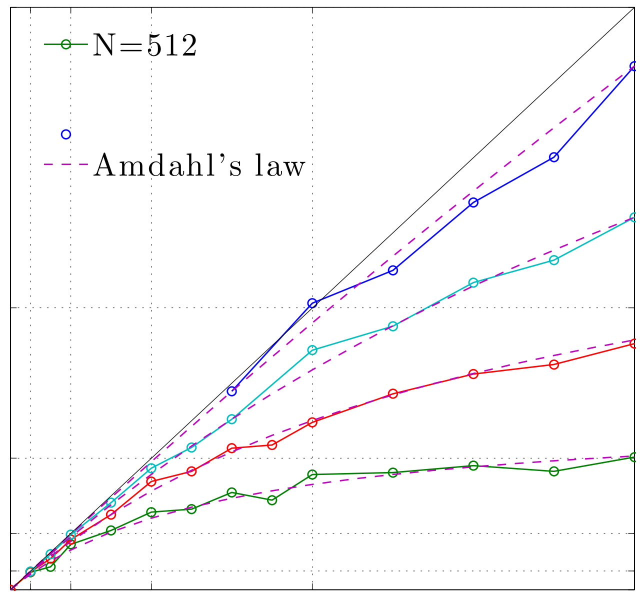
Computing times in seconds

N	256	512	1024	2048
1 GPU	3.33	22.8	181	-
2 GPU	2.31	12.6	95.6	798
4 GPU	2.08	7.68	51.1	398
8 GPU	2.00	4.65	26.4	211

Solution: $N \times N \times N$. Projections: $400 \times N \times N$.

GPU Cluster – K40





Final Conclusions

- We have some theory for ART that supports the observed performance for noisy systems: *semi-convergence*.

- Block-sequential methods – preferably with structurally orthogonal rows in each block – have *very fast* performance.

- Block algorithms targeting *GPUs* (joint work with ASTRA group at CWI, Amsterdam) are under development.