Rotational Image Deblurring with Sparse Matrices

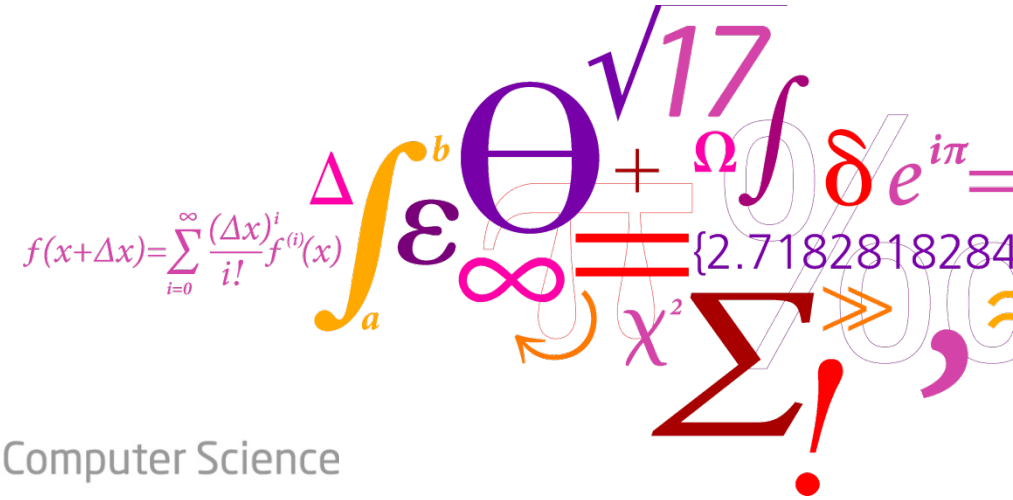
Per Christian Hansen
Technical University of Denmark

Joint work with

- James G. Nagy, Emory University
- Konstantinos Tigkos, Fraunhofer Inst., Erlangen, Germany

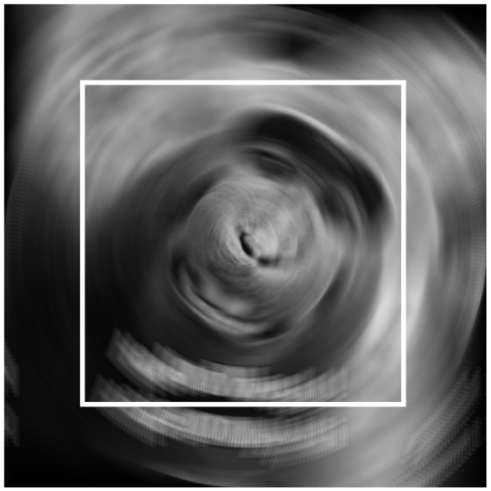
With thanks to H. Aanæs and J. Bardsley

BIT Numerical Mathematics
DOI: 10.1007/s10543-013-0464-y



DTU Compute
Department of Applied Mathematics and Computer Science

Motivation: Motion Blur



Motion blur arises frequently in several applications.

Our interest here is rotational blurring which arises, e.g., when a satellite navigates in space.

This is an example of *spatially variant blur*, where the blurring depends on the location in the image.

Different Types of Rotational Blur

Pure rotation. The rotation axis points toward the camera (i.e., the scene rotates around its center) and is orthogonal to the image plane.

Pure tilt. The rotation axis is orthogonal to the direction of the scene (i.e., the scene moves on a circle round the focal point at a fixed distance).

Tilted rotation. Similar to the pure rotation, except that the rotation axis is not orthogonal to the image plane.

Pure rotation

Pure tilt

Tilted rotation

Previous Work Related to Rotational Blur

The underlying problem (the fortward model):

- $A\,x = b$ – but A is ill conditioned and b is noisy so don't solve this system!

Transform to polar coordinates where the motion is linear:

- Estatico & Di Bendetto (2013); Sawchuk (1974).
- Interpolation errors – works only for pure rotation.

Section the image in patches with approximately invariant blur:

- Nagy & O'Leary (1998); Trussell & Fogel (1992).
- Works best when the blur is smoothly varying over the image.

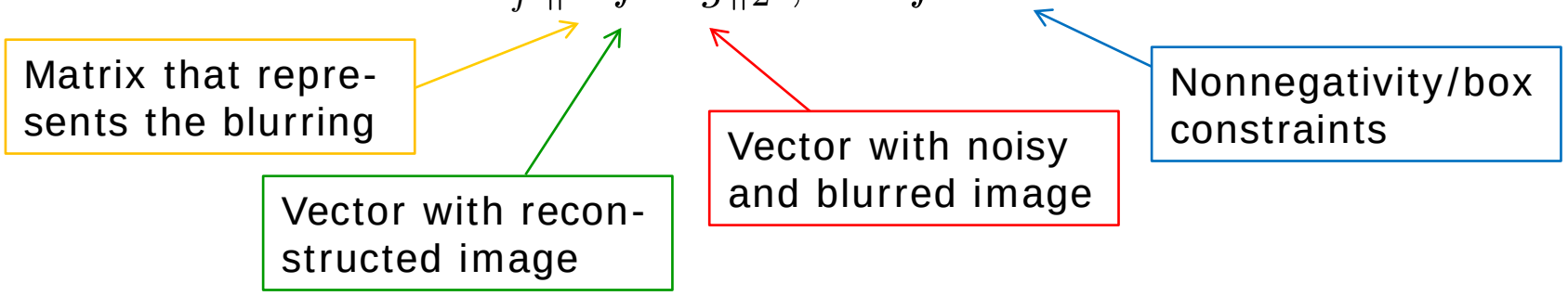
Model the motion blur by summing a sequence of rotated/translated images:

- Tai, Tan & Brown (2011).
- Matrix-free; uses the Richardson-Lucy algorithm.
- Inspired this work.

The Deblurring Problem

Our deblurring problem is a constrained linear least squares problem:

$$\min_f \|A\,f - g\|_2 \;, \qquad f \in \mathcal{C}$$



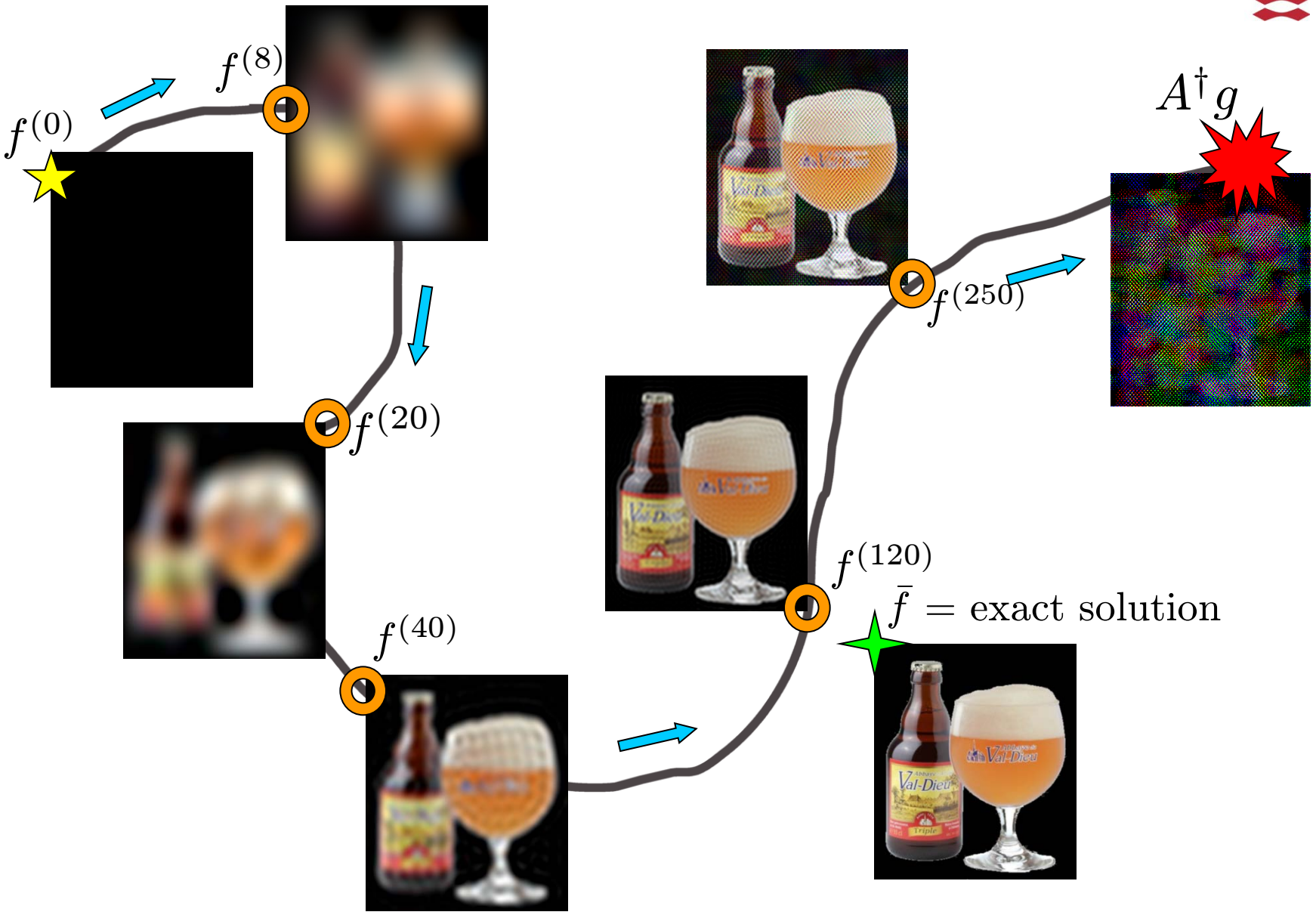
The inverse problem is *ill posed*, and hence A is *ill conditioned* $\rightarrow$ we need *regularization* to stably compute a useful approximate solution.

For spatially variant blurring, there are usually no "fast algorithms" – i.e., no matrix structure to exploit, except perhaps sparsity.

Hence it is natural to use iterative algorithms.

We use *regularizing iterations* where regularization/stabilization is achieved by early termination of the iterations (semi-convergence).

Illustration of Semi-Convergence



The Forward Model

Forward model: the blurred image is created as a sum of images that undergo a sequence of small rotations:

The standard way to implement this is via a matrix-free approach that uses interpolation from one rotated image to the next.

\+ Simple to implement. Requires little extra storage.
\− Slow!

We choose to explicitly create a sparse matrix A that represents the rotation sequence.

\+ Fast, once the matrix is created.
\+ Easy to incorporate boundary conditions.
\− Initialization and storage of the sparse matrix.

Computation of *A*

The motion-blurred image G is the scaled sum of images at incremental times, or positions, during acquisition:

$$G = \frac{1}{M} \sum_{i=0}^{M-1} F(\vec{\xi}_i).$$

- Each $F(\vec{\xi}_i)$ represents the ith snapshot of the object,
- $\vec{\xi}_0$ is the position vector corresponding to the original image,
- $\vec{\xi}_i$, $i = 1, 2, \ldots, M-1$ are position vectors of the ith transformed images, where $\vec{\xi}_i = H(\alpha_i)\vec{\xi}_0$ and H is a *homography matrix*.

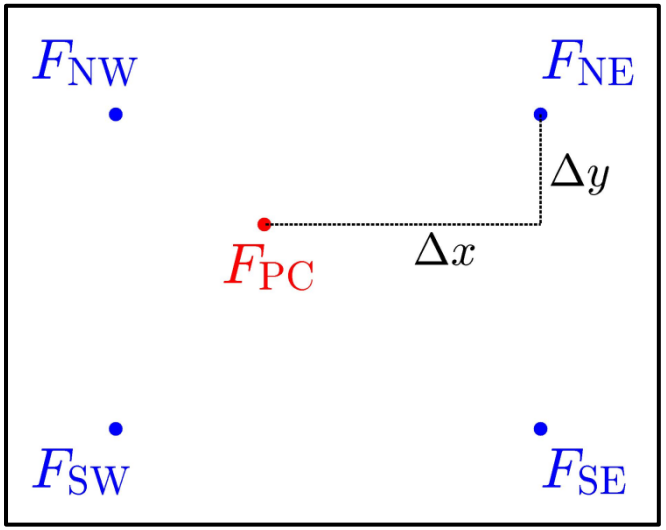
We thus obtain

$$g = \frac{1}{M} \sum_{i=0}^{M-1} \mathrm{vec}(F(\vec{\xi}_i)) = \frac{1}{M} \sum_{i=0}^{M-1} A_i f = Af \ .$$

More About the Computation of *A*

Sample each image $F(\vec{\xi}_i)$ at the pixel centers via bilinear interpolation.

The pixel value at each center is approximated by a weighted average of the four transformed values that surround it; the weights are proportional to the distance from the pixel center.



$F_{\rm PC}$ = pixel value at a pixel center.

$F_{\rm NW}, F_{\rm NE}, F_{\rm SW}, F_{\rm SE}$ = surrounding pixel values after motion transformation.

Then bilinear interpolation approximates $F_{\rm PC}$ as

$$\begin{aligned} F_{\rm PC} \;\approx\; & (1-\Delta x)(1-\Delta y)\, F_{\rm SW} + \Delta x \Delta y\, F_{\rm NE} + \\ & (1-\Delta x)\Delta y\, F_{\rm NW} + \;\Delta x(1-\Delta y)\, F_{\rm SE}\;. \end{aligned}$$

So each row of A has 4 nonzero elements: $(1-\Delta x)(1-\Delta y)$, $\Delta x \Delta y$, etc.

Boundary Conditions (Important)

Recall the deblurring problem: $\min_f \|A\,f - g\|_2,\ f \in \mathcal{C}$. Here:

$$A = A_{\mathrm{F}} + A_{\mathrm{BC}}$$

A_{F} = simple forward model, A_{BC} = correction for boundary constraints.

- **Zero:** the scene outside the initial image is black.
- **Replicate:** each pixel at the border of the initial image is infinitely replicated outside the image.
- **Periodic:** the scene outside the initial image is a periodic replication of the image.
- **Reflective:** the scene outside the initial image is a reflection of the image along the image border.

The matrices A_i in $A = \frac{1}{M}\sum_{i=0}^{M-1} A_i$ are modified accordingly.

Iterative Reconstruction Methods

A variety of iterative algorithms are available:

- ART (Kaczmarz) – slow for image deblurring.
- CGLS – cannot incorporate nonnegativity or box constraints.
- FISTA (Fast Iterative Shrinkage-Thresholding Algorithm) – slow.
- MRNSD (Mod. Residual Norm Steepest Descent) – works well.
- PL (Projected Landweber) – works well for us.
- Richardson-Lucy – slow for our problems.

PL and MRNSD have iterates of the form:

$$f^{(k+1)} = f^{(k)} + \tau_k \, d^{(k)} \ ,$$

where $d^{(k)}$ = direction vector and τ_k = step length.

Projected Landweber

We used a fixed step $\tau_k = \tau$ (found by experiments) and the direction is

$$d^{(k)} = A^T(g - A\,f^{(k)}) \ .$$

We incorporate box constraints via the projection

$$f^{(k+1)} = \mathcal{P}\big(f^{(k)} + \tau\,A^T(g - A\,f^{(k)})\big) \ .$$

The semi-convergence property is well established:

- van der Sluis & van der Vorst (1990); Elfving, Nikazad & H (2010).
- With projection: Elfving, H & Nikazad (2012).

The noise-error behaves as

$$\|f^{(k)} - \bar{f}^{(k)}\| = O(\tau\,k\,\epsilon)$$

where ϵ = norm of data errors and $\bar{f}^{(k)}$ = iterates for noise-free data.

MRNSD

Here τ_k is chosen to ensure nonnegativity, i.e.,

$$f^{(k+1)} = f^{(k)} + \tau_k \, d^{(k)} \geq 0 \ ,$$

and the step direction is

$$d^{(k)} = D_k \, A^T (g - A \, f^{(k)}) \ , \qquad D_k = \mathrm{diag}(f^{(k)}) \ .$$

Derivation and analysis: Nagy & Strakoš (2000); Bardsley & Nagy (2006).

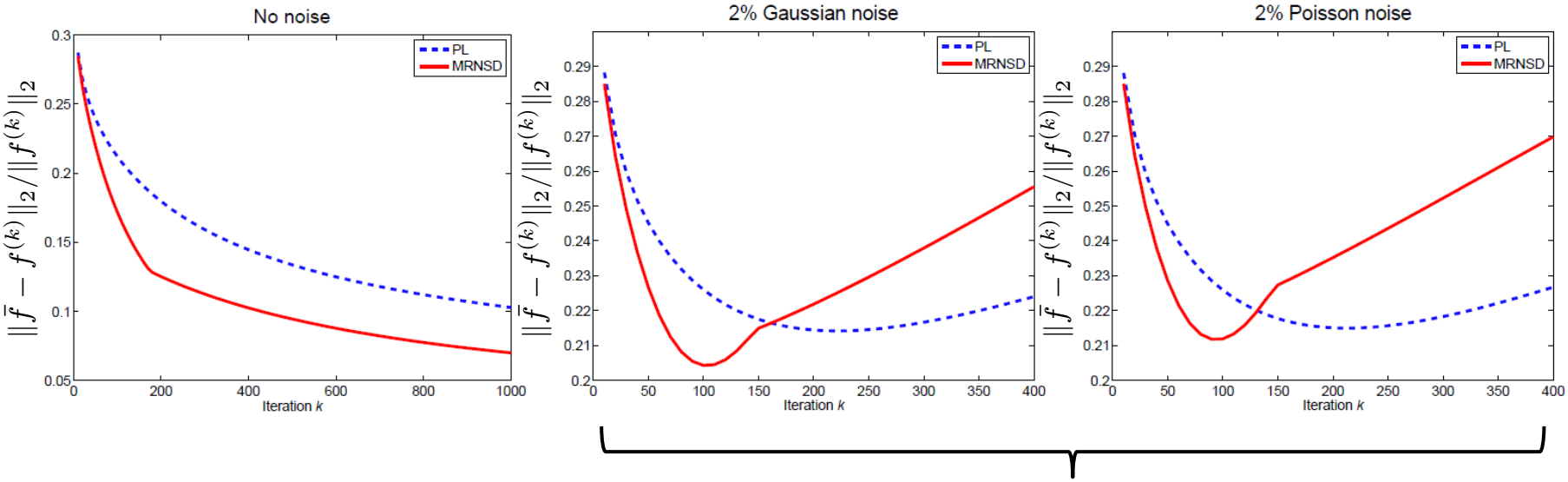
Richardson-Lucy can be considered a simpler and slower version of MRNSD.

The semi-convergence property is observed experimentally – but a rigorous analysis is unfortunately lacking.

Perhaps the analysis of semi-convergence in ART – Elfving, H & Nikazad (2014) – carries over?

Illustration of Performance I

Example: pure rotation by 20° - no noise, Gaussian noise, Poisson noise.



The semi-convergence is evident.

Illustration of Performance II

(a) Pure rotation (b) PL (c) MRNSD

(d) Pure tilt (e) PL (f) MRNSD

(g) Tilted rotation (h) PL (i) MRNSD

Rotation	PL		MRNSD	
	ρ	S	ρ	S
Pure rotation	0.2149	0.7328	0.2117	0.7448
Pure tilt	0.2320	0.6635	0.2273	0.6761
Tilted rotation	0.2218	0.7234	0.2240	0.7300

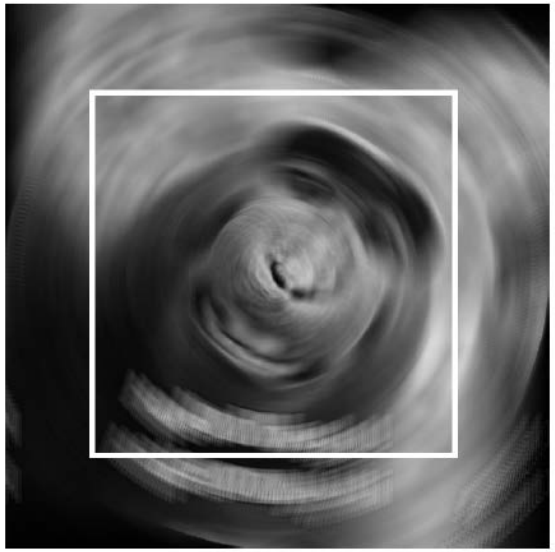
ρ = relative 2-norm error.

S = SSIM (struct. sim. index).

Notice the "ghosts" – see our paper for explanation.

Illustration of Performance III

(a) Sharp padded image



(b) Blurred padded image

(c) Cropped

(d) Rec., reflexive BC

(e) Rec., zero BC

The role of boundary conditions:

- Inverse crime: the blurred image is created with reflective padding.
- Thus reflective BC give perfect reconstruction.
- Zero BC give severe artifacts.

Stopping Criterion

We tested two different stopping criteria.

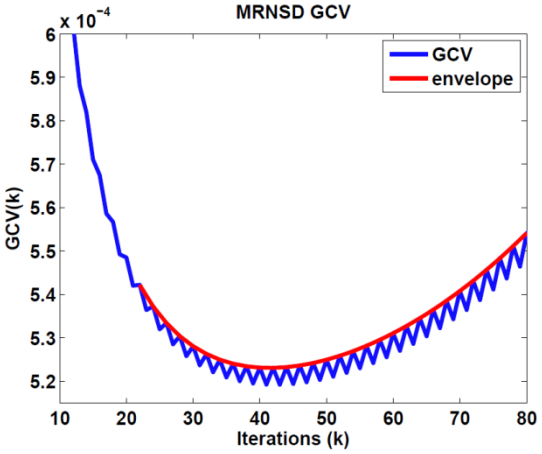
NCP Criterion
Rust (1998); H, Kilmer & Kjeldsen (2006); Rust & O'Leary (2008).

Uses the normalized cumulative periodogram (cumulative power spectrum) to stop when the residual vector $g - A\,f^{(k)}$ becomes noise-like.

Monte-Carlo GCV
Girard (1989); Hutchinson (1989); Perry & Reeves (1994); Bardsley (2008).

Minimizes an estimate of the prediction error $\|\bar{g} - A\,f^{(k)}\|_2$, $\bar{g}$ = pure data.

As noted by Perry & Reeves and Bardsley, the GCV function can exhibit oscillations, and we must smooth it.



Performance Results I

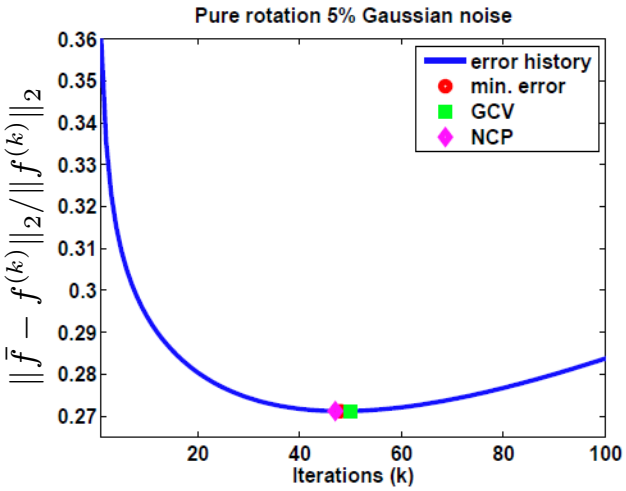
Projected Landweber

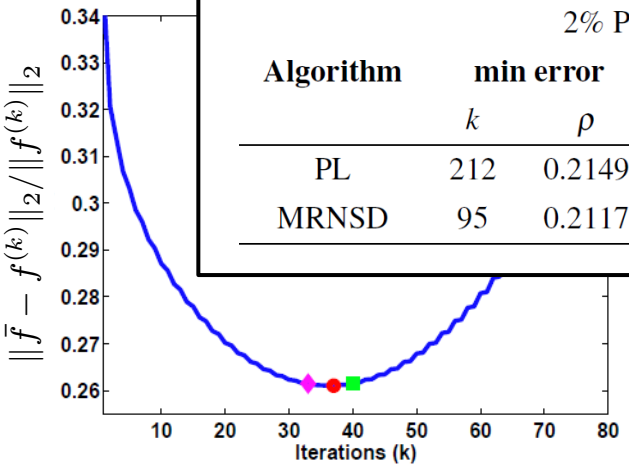
5% Poisson noise

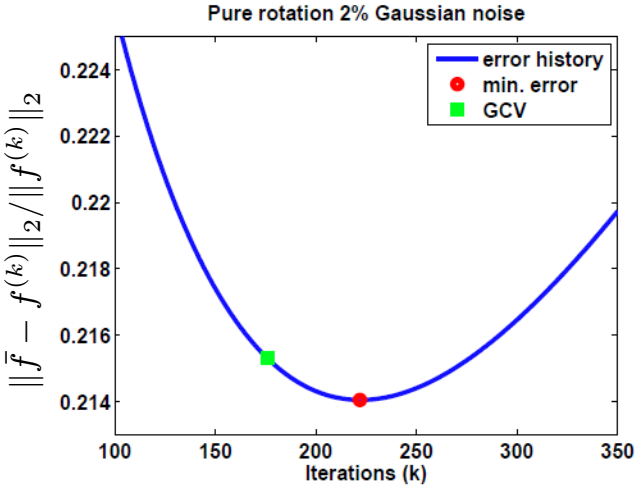
Algorithm	min error		NCP		GCV	
	k	ρ	k	ρ	k	ρ
PL	46	0.2716	46	0.2716	57	0.2729
MRNSD	33	0.2681	35	0.2682	64	0.2960

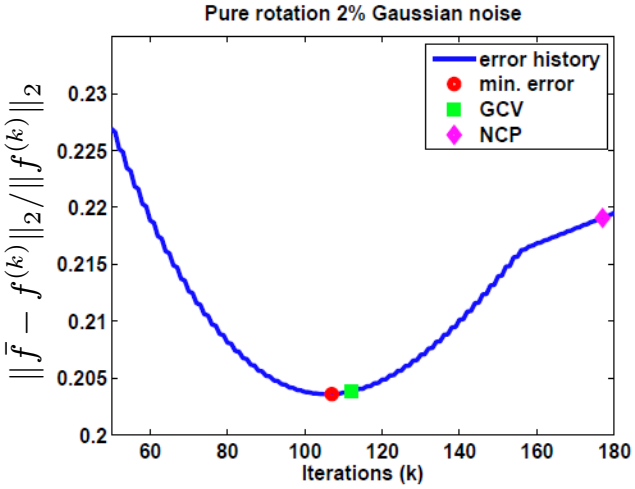
2% Poisson noise

Algorithm	min error		NCP		GCV	
	k	ρ	k	ρ	k	ρ
PL	212	0.2149	492	0.2359	200	0.2150
MRNSD	95	0.2117	221	0.2385	130	0.2193









No inverse crime:

We use high-resolution images to compute the blurred image.

Performance Results II

The best results are obtained with MRNSD and reflective BC.

(a) Exact image

(b) Blurred image

(c) Gaussian noise
$\rho = 0.2602, S = 0.7194$

(d) Poisson noise
$\rho = 0.2623, S = 0.7082$

Conclusions

- Our approach with a sparse coefficient matrix allows us to treat rotation by an arbitrary axis.
- We can easily incorporate boundary conditions in the reconstruction algorithm.
- We can also incorporate nonnegativity and box constraints.
- We implemented and tested robust stopping rules (Monte-Carlo GCV worked best).
- Future work: faster algorithms, total variation, …