Tutorial:
Krylov Subspace Methods

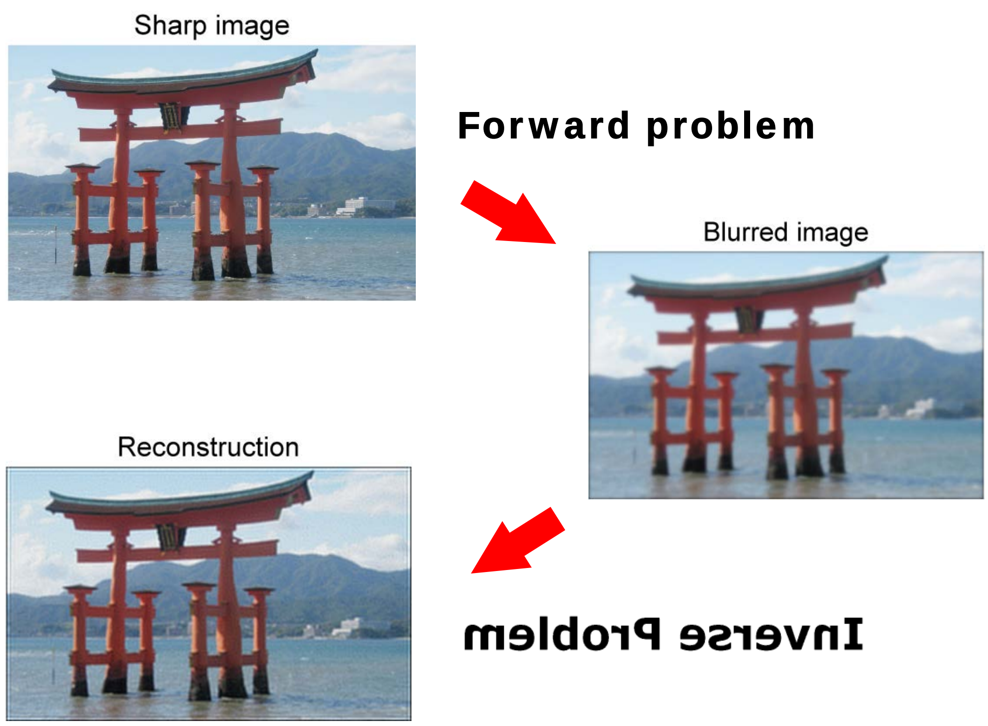
Per Christian Hansen
Technical University of Denmark

DTU Compute
Department of Applied Mathematics and Computer Science

Image Reconstruction

Test case:

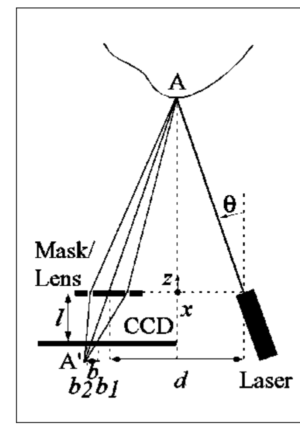
- Image deblurring

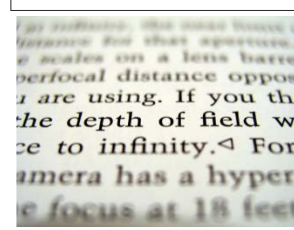


This talk:

- Blurring
- Regularization
- Projection
- CGLS
- Other iterations
- Noise propagation
- Augmentation
- Preconditioning

Sources of Blurred Images





Some Types of Blur and Distortion

From the camera:

- the lens is out of focus,
- imperfections in the lens, and
- noise in the CCD and the analog/digital converter.

From the environments:

- motion of the object (or camera),
- fluctuations in the light's path (turbulence), and
- false light, cosmic radiation (in astronomical images).

Given a mathematical/statistical ***model*** of the blur/distortion, we can ***deblur*** the image and compute a sharper reconstruction (as apposed to ”cosmetic improvements” by PhotoShop etc).

Top 10 Algorithms

J. J. Dongarra, F. Sullivan et al., *The Top 10 Algorithms*, IEEE Computing in Science and Engineering, 2 (2000), pp. 22-79.

1946: The Monte Carlo method (Metropolis Algorithm).

1947: The Simplex Method for Linear Programming.

1950: Krylov Subspace Methods (CG, CGLS, Arnoldi, etc.).

1951: Decomposition Approach to matrix computations.

1957: The Fortran Optimizing Compiler.

1961: The QR Algorithm for computing eigenvalues and –vectors.

1962: The Quicksort Algorithm.

1965: The Fast Fourier Transform algorithm.

1977: The Integer Relation Detection Algorithm.

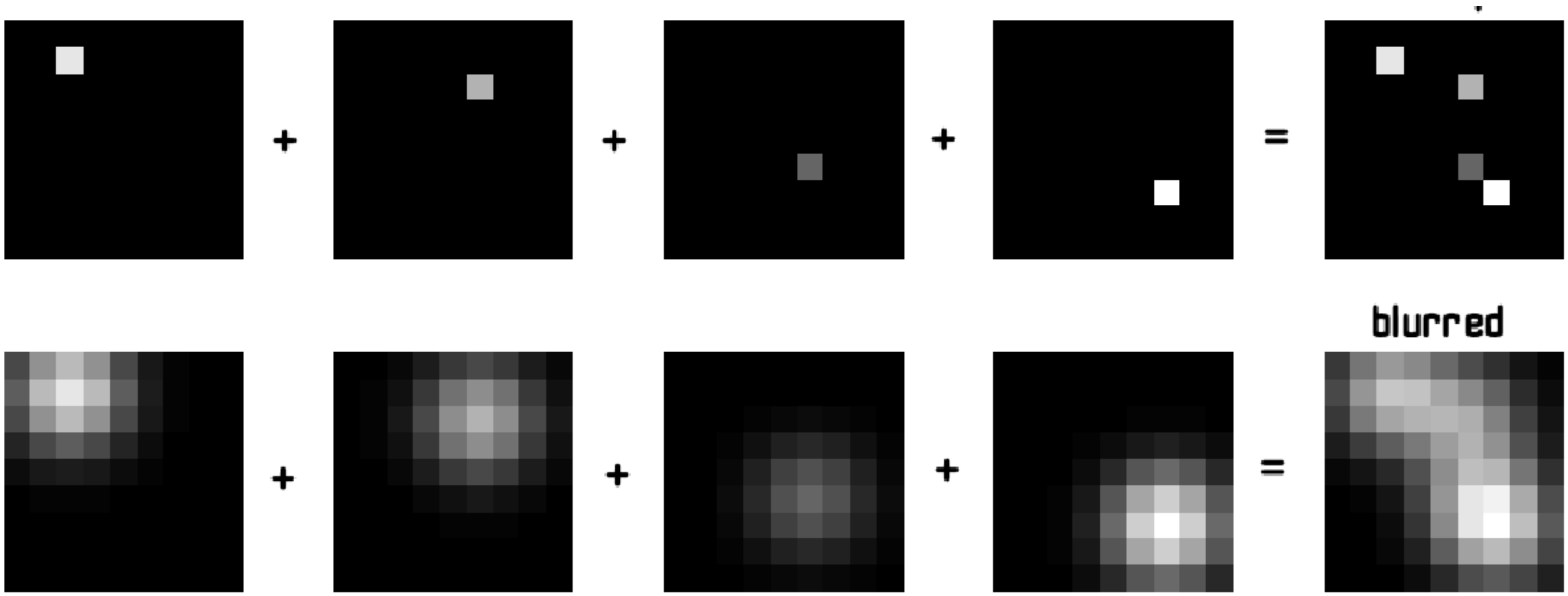
1987: The Fast Multipole Algorithm for *N*-body simulations.

Key algorithms in image deblurring.

The Point Spread Function – Linearity

The point spread function is the image of a single bright pixel.

The blurred image is the sum of all the blurred pixels.



The Deblurring Problem

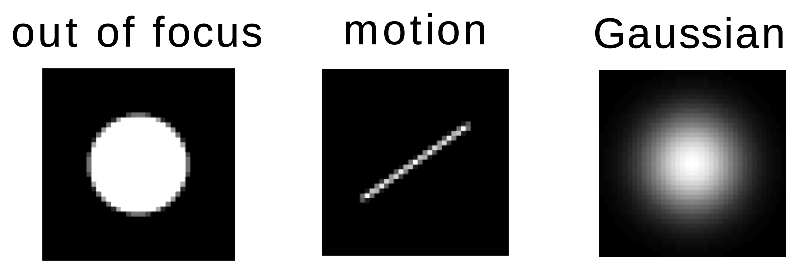
Fredholm integral equation of the first kind:

$$\int_0^1 \int_0^1 K(x, y; x', y')\, f(x, y)\, dx\, dy = g(x', y') , \qquad 0 \le x', y' \le 1.$$

Think of f as an unknown sharp image, and g as the blurred version.

Think of K as a model for the point spread function.

Examples of point spread functions →



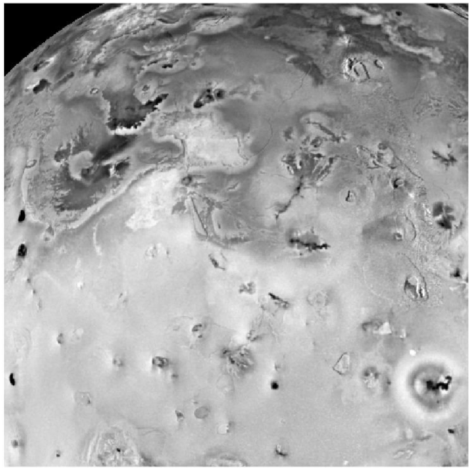
Discretization yields a LARGE system of linear equations: $A\,x = b$.

Two important aspects related to this system:

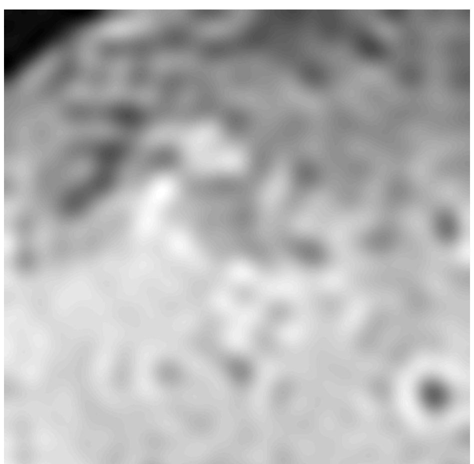
- Use the right boundary conditions.
- The matrix A is very ill conditioned → **Do not solve $A\,x = b$!**

Boundary Conditions (BC)

Sharp image



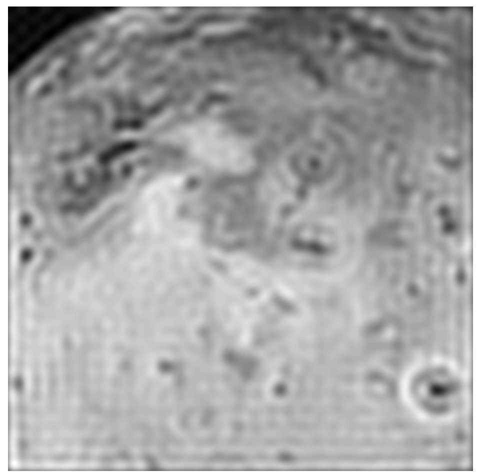
Blurred image



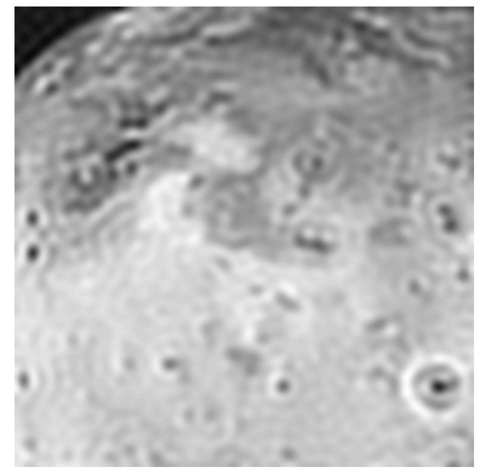
• Zero and periodic BC lead to artifacts at the boundaries.

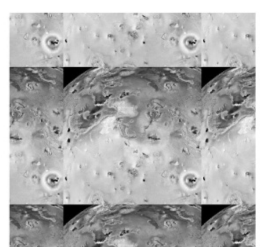
• Reflexive BC can lead to better images.

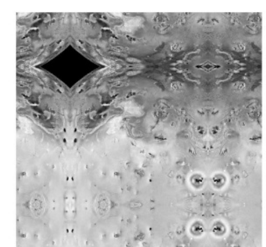
Periodic BC



Reflexive BC







Inverse Problem: Regularization is Needed!

The inverse problem of image deblurring is an *ill-posed problem*, i.e, it violates one or more of the three Hadamard conditions for a well-posed problem:

- the solution exists,
- the solution is unique,
- the solution is stable with respect to perturbations of data.

$$\text{Algebraic model:} \qquad A\,x = b, \qquad b = A\,x^{\text{exact}} + e.$$

In the algebraic model, the matrix *A* is very ill conditioned, and we do **not** want to compute the "naive solution":

$$x^{\text{naive}} = A^{-1}b = x^{\text{exact}} + A^{-1}e, \qquad \|A^{-1}e\| \gg \|x^{\text{exact}}\|$$

We must use *regularization* to compute a stable solution.

Setting the Stage for Regularization

We need the SVD of the matrix A:

$$A = U\,\Sigma\,V^T = \sum_{i=1}^{\min(m,n)} u_i\,\sigma_i\,v_i^T .$$

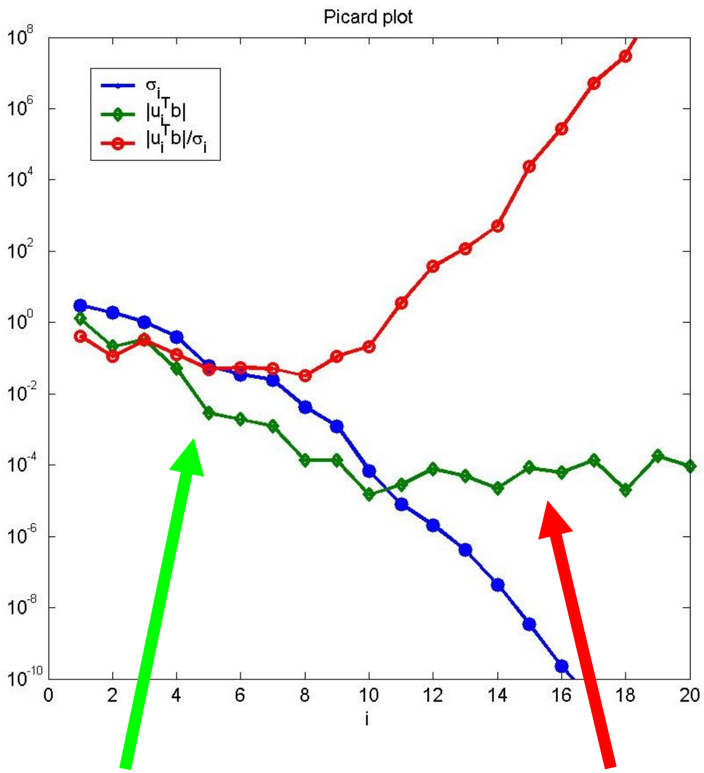
The (minimum norm) least squares least squares solution is:

$$x_{\mathrm{LS}} = A^\dagger b = \sum_{i=1}^{\mathrm{rank}(A)} \frac{u_i^T b}{\sigma_i}\,v_i .$$

Regularized solutions (obtained by "spectral filtering") are:

$$x_{\mathrm{reg}} = \sum_{i=1}^{n} \varphi_i\,\frac{u_i^T b}{\sigma_i}\,v_i , \qquad \varphi_i = \text{filter factors.}$$

The Need for Regularization



Assume Gaussian noise:

$$b = b^{\text{exact}} + e \ , \quad e \sim \mathcal{N}\left(0, \sigma_{\text{noise}}^2 I\right).$$

Then

$$x_{\text{naive}} \equiv A^{-1} b = x^{\text{exact}} + A^{-1} e,$$

and using the SVD we see that

$$x_{\text{naive}} = \sum_{i=1}^{n} \frac{u_i^T b}{\sigma_i} v_i$$

$$= \sum_{i=1}^{n} \frac{u_i^T b^{\text{exact}}}{\sigma_i} v_i + \sum_{i=1}^{n} \frac{u_i^T e}{\sigma_i} v_i .$$

"inverted noise"

<u>Picard condition:</u>

$|u_i^T b|$ decays faster than σ_i for small i.

<u>Noise:</u>

$|u_i^T b|$ levels off for larger i.

Regularization:

keep the "good" SVD components and discard the noisy ones!

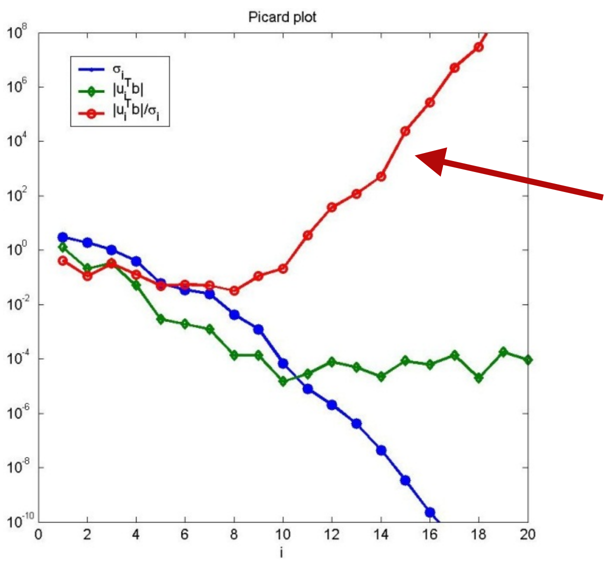
Regularize!

We must apply regularization in order to deal with the ill conditioning of the problem and suppress the influence of the noise in the data.

The previous slide suggest a “brute force” approach – chop off the most troublesome components in the SVD expansion of the (least squars) solution.

Truncated SVD:

$$x_k = \sum_{i=1}^{k} \frac{u_i^T b}{\sigma_i} v_i \ .$$



The truncation parameter k should be selected to discard those SVD components that are dominated by the noise in the right-hand side.

Note that k is determined from the behavor of the right-hand side's SVD coefficients $u_i^T b$ – and not from the size of the singular values σ_i.

A Systematic View of Regularization

We must apply regularization in order to deal with the ill conditioning of the problem and suppress the influence of the noise in the data.

Tikhonov regularization:

$$\min_x \left\{ \|A\,x - b\|_2^2 + \lambda^2 \, \|L\,x\|_2^2 \right\}$$

The choice of smoothing norm, together with the choice of λ, forces x to be effectively dominated by components in a low-dimensional subspace, determined by the GSVD of (A, L) – or the SVD of A if $L = I$.

Regularization by projection:

$$\min_x \|A\,x - b\|_2 \quad \text{subject to} \quad x \in \mathcal{W}_k$$

where $\mathcal{W}_k$ is a k-dimensional subspace.

This works well if "most of" $x^{\rm exact}$ lies in a low-dimensional subspace; hence $\mathcal{W}_k$ must be spanned by desirable basis vectors. Think of Truncated SVD: $\mathcal{W}_k = \text{span}\{v_1, v_2, \ldots, v_k\}$, $v_i =$ right singular vectors.

The Projection Method

A more practical formulation of regularization by projection.

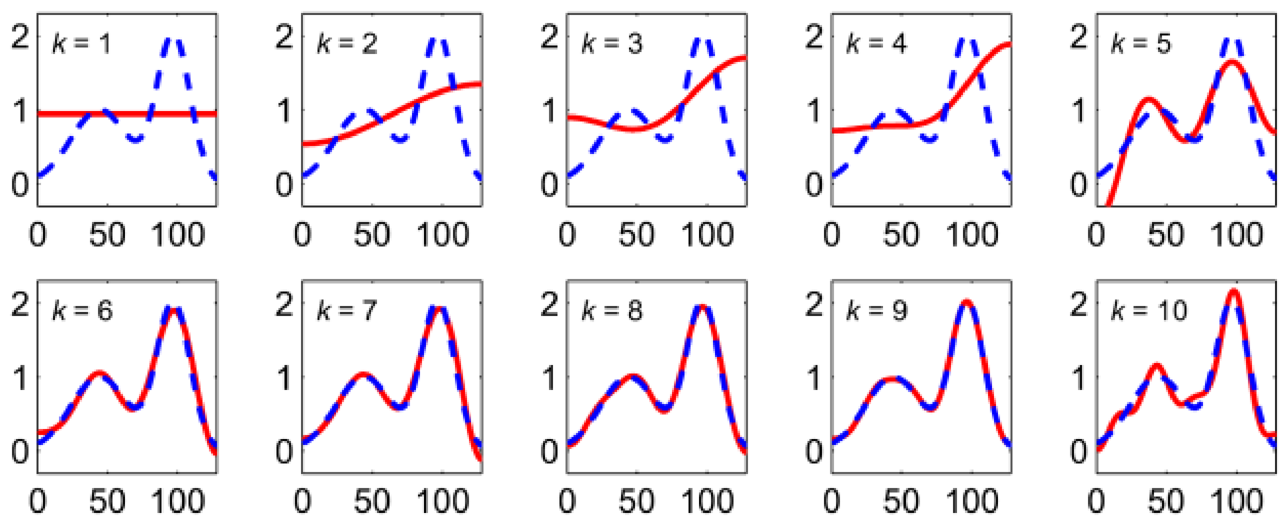
We are given the matrix $W_k = (w_1, \ldots, w_k) \in \mathbb{R}^{n \times k}$ such that $\mathcal{W}_k = \mathcal{R}(W_k)$.

We can write the requirement as $x = W_k\, y$, leading to the formulation

$$x^{(k)} = W_k\, y^{(k)}, \qquad y^{(k)} = \text{argmin}_y \, \|(A\, W_k)\, y - b\|_2 .$$

Projected problem

Example:
DCT basis



Operations often do not require W_k explicitly.

Some Thought on the Basis Vectors

The DCT basis – and similar bases that define fast transforms:

- computationally convenient (fast) to work with, but
- may not be well suited for the particular problem.

The SVD basis – or GSVD basis if $L \neq I$ – gives an "optimal" basis for representation of the matrix A, but ...

- it is computationally expensive (slow), and
- it does not involve information about the right-hand side b.

Is there a basis that is computationally attractive and also involves information about both A and b, and thus the complete given problem?

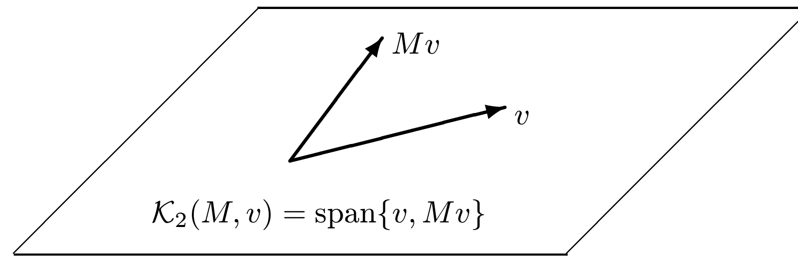
→ Krylov subspaces!

Krylov Subspaces

Given a square matrix M and a vector v, the associated Krylov subspace is defined by

$$\mathcal{K}_k(M,v) \equiv \text{span}\{v, Mv, M^2 v, \ldots, M^{k-1}v\}, \qquad k = 1, 2, \ldots$$

with $\dim(\mathcal{K}_k(M,v))$ □ k.



Krylov subspaces have many important applications in scientific computing:

- solving large systems of linear equations,
- computing eigenvalues,
- solving algebraic Riccati equations, and
- determining controllability in a control system.

They are also important tools for regularization of large-scale discretizations of inverse problems, which is the topic of this talk.

More about the Krylov Subspace

The Krylov subspace, defined as

$$\mathcal{K}_k \equiv \text{span}\{A^T b, A^T A\, A^T b, (A^T A)^2 A^T b, \ldots, (A^T A)^{k-1} A^T b\},$$

always *adapts* itself to the problem at hand! But the "naive" basis,

$$p_i = (A^T A)^{i-1} A^T b \,/\, \|(A^T A)^{i-1} A^T b\|_2, \qquad i = 1, 2, \ldots$$

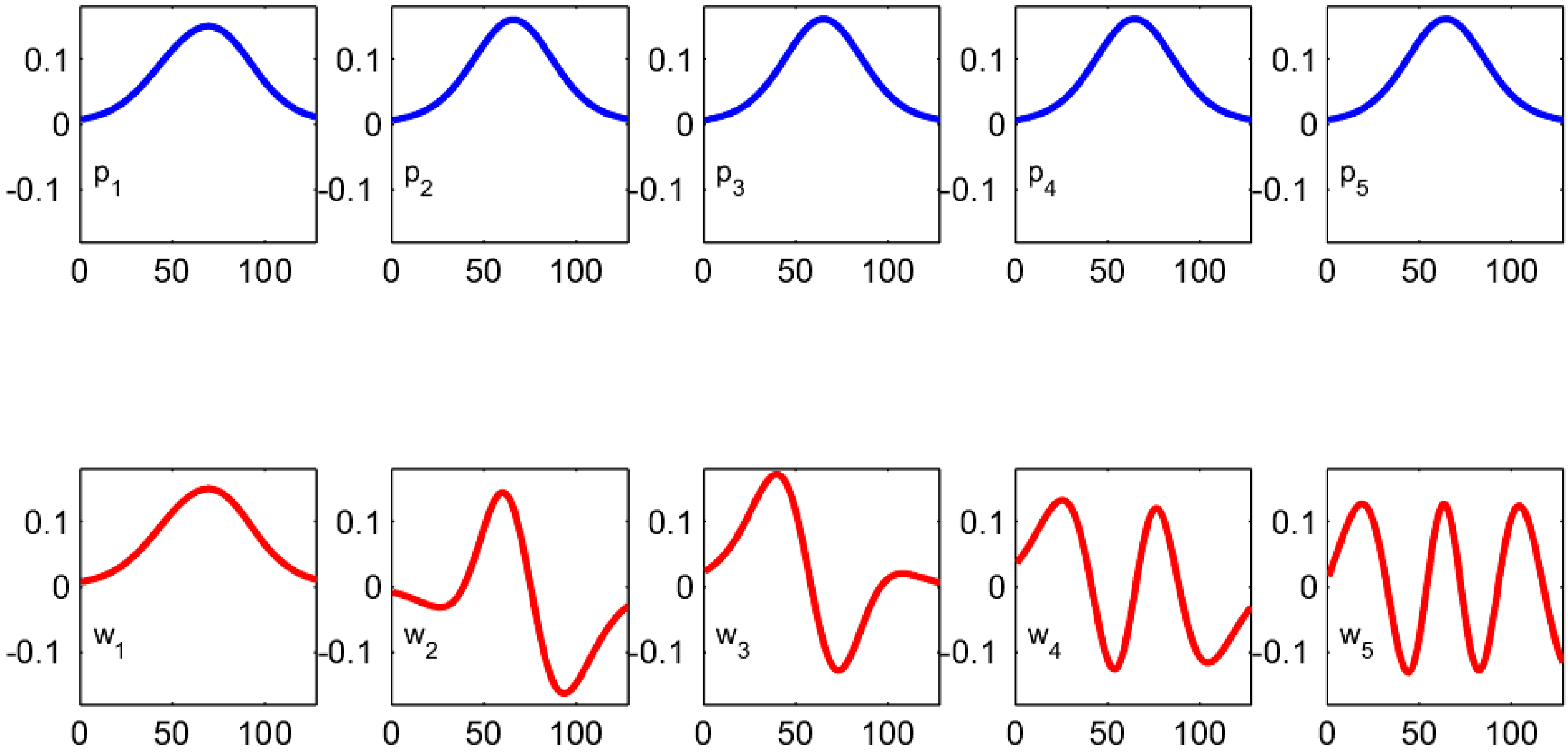
are NOT useful: $p_i \to v_1$ as $i \to \infty$.

Can use modified Gram-Schmidt:

$$\begin{array}{lll}
w_1 \leftarrow A^T b; & w_1 \leftarrow w_1 / \|w_1\|_2 & \\
w_2 \leftarrow A^T A\, w_2; & w_2 \leftarrow w_2 - w_1^T w_2\, w_1; & w_2 \leftarrow w_2 / \|w_2\|_2 \\
w_3 \leftarrow A^T A\, w_2; & w_3 \leftarrow w_3 - w_1^T w_3\, w_1; & \\
 & w_3 \leftarrow w_3 - w_2^T w_3\, w_2; & w_3 \leftarrow w_3 / \|w_3\|_2
\end{array}$$

The Krylov Subspace – Example

Normalized basis vectors p_i (blue) and orthonormal basis w_i (red).



Regularizing Iterations

Can we compute $x^{(k)}$ without forming and storing the Krylov basis in W_k?

Apply CG to the normal equations for the least squares problem

$$\min \|A\,x - b\|_2 \qquad \Leftrightarrow \qquad A^T A\,x = A^T b\ .$$

This stable stable and efficient implementation of this algorithm is called CGLS, and it produces a sequence of iterates $x^{(k)}$ which solve

$$\min \|A\,x - b\|_2 \qquad \text{subject to} \qquad x \in \mathcal{K}_k\ .$$

This use of CGLS to compute regularized solutions in the Krylov subspace $\mathcal{K}_k$ is referred to as *regularizing iterations*.

Iterative methods are based on multiplications with A and A^T (blurring).

How come repeated blurings can lead to reconstruction?

$\rightarrow$ CGLS constructs a polynomial approximation to $A^\dagger = (A^T A)^{-1} A^T$.

The CGLS Algorithm

$$
\begin{aligned}
x^{(0)} &= \text{starting vector (e.g., zero)} \\
r^{(0)} &= b - A\,x^{(0)} \\
d^{(0)} &= A^T r^{(0)}
\end{aligned}
$$

Initialization

for $k = 1, 2, \ldots$

$$
\begin{aligned}
\bar{\alpha}_k &= \|A^T r^{(k-1)}\|_2^2 / \|\underbrace{A\,d^{(k-1)}}\|_2^2 \\
x^{(k)} &= x^{(k-1)} + \bar{\alpha}_k\, d^{(k-1)} \\
r^{(k)} &= r^{(k-1)} - \bar{\alpha}_k\, A\, d^{(k-1)} \\
\bar{\beta}_k &= \|A^T r^{(k)}\|_2^2 / \|A^T r^{(k-1)}\|_2^2 \\
d^{(k)} &= \underbrace{A^T r^{(k)}} + \bar{\beta}_k\, d^{(k-1)}
\end{aligned}
$$

end

Mult. with A^T Mult. with A

The Behavior of CGLS

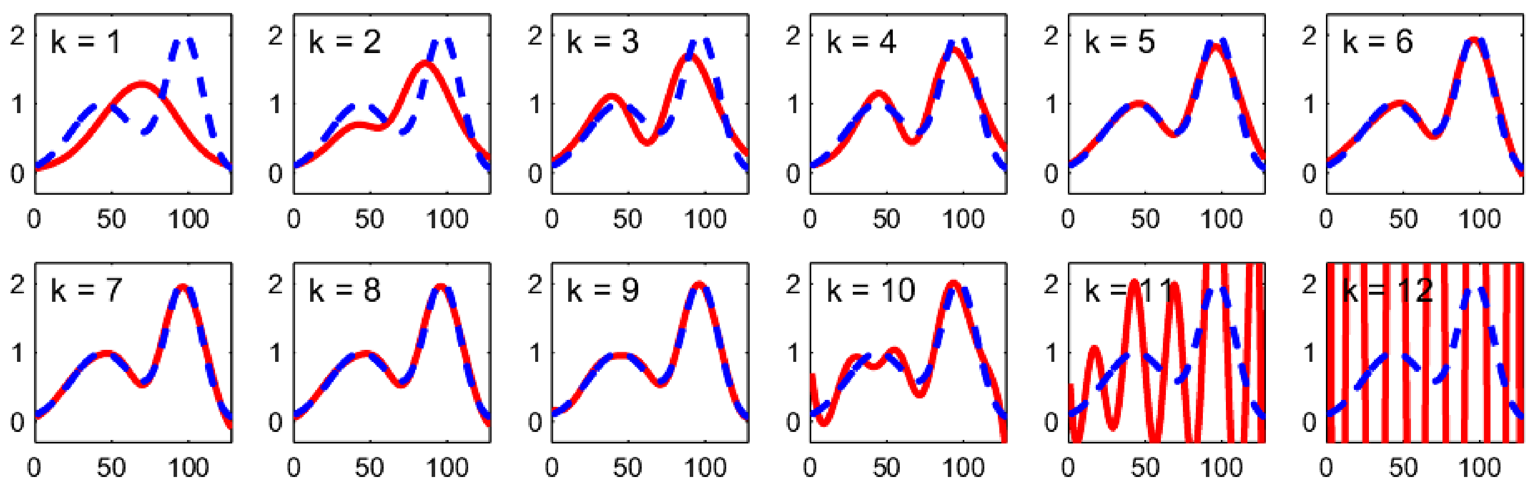
CGLS algorithm solves the problem without forming the Krylov basis explicitly.

Finite precision: convergence slows down, but no deterioration of the solution.

The solution and residual norms are monotone functions of k:

$$\|x^{(k)}\|_2 \geq \|x^{(k-1)}\|_2, \qquad \|A\,x^{(k)} - b\|_2 \;\square\; \|A\,x^{(k-1)} - b\|_2, \qquad k = 1, 2, \ldots$$

Same example as before: CGLS iterates



The CGLS Polynomials

CGLS implicitly constructs a polynomial $\mathcal{P}_k$ such that

$$x^{(k)} = \mathcal{P}_k(A^T A) A^T b \ .$$

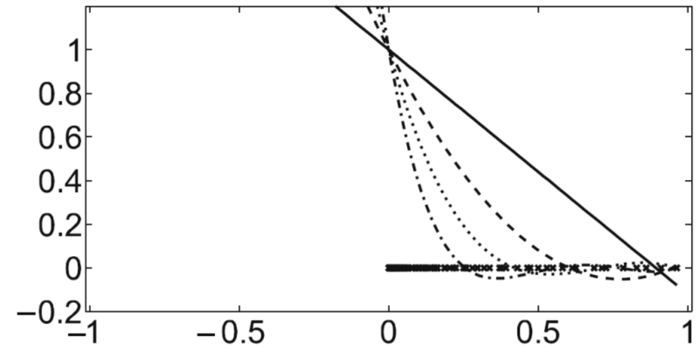
But how is $\mathcal{P}_k$ constructed? Consider the residual

$$\begin{aligned} r^{(k)} &= b - A\,x^{(k)} = \big(I - A\,\mathcal{P}_k(A^T A)\,A^T\big)\,b \\ \|r^{(k)}\|_2^2 &= \big\|\big(I - \Sigma\,\mathcal{P}_k(\Sigma^2)\Sigma\big)\,U^T b\big\|_2^2 \\ &= \sum_{i=1}^n \big(1 - \sigma_i^2 \mathcal{P}_k(\sigma_i^2)\big)^2 (u_i^T b)^2 = \sum_{i=1}^n \mathcal{Q}_k(\sigma_i^2)(u_i^T b)^2 \end{aligned}$$

To minimize residual norm $\|r^{(k)}\|_2$:

→ make $\mathcal{Q}_k(\sigma_i^2)$ small where $(u_i^T b)^2$ is large

→ force $\mathcal{Q}_k(\sigma_i^2)$ to have roots
near σ_i that corresp. to large $(u_i^T b)^2$.



Semi-Convergence

During the first iterations, the Krylov subspace $\mathcal{K}_k$ captures the "important" information in the noisy right-hand side b.

- In this phase, the CGLS iterate $x^{(k)}$ approaches the exact solution.

At later stages, the Krylov subspace $\mathcal{K}_k$ starts to capture undesired noise components in b.

- Now the CGLS iterate $x^{(k)}$ diverges from the exact solution and approach the undesired solution $A^\dagger b$ to the least squares problem.

The iteration number k (= the dimension of the Krylov subspace $\mathcal{K}_k$) plays the role of the regularization parameter.

This behavior is called *semi-convergence*.

Illustration of Semi-Convergence

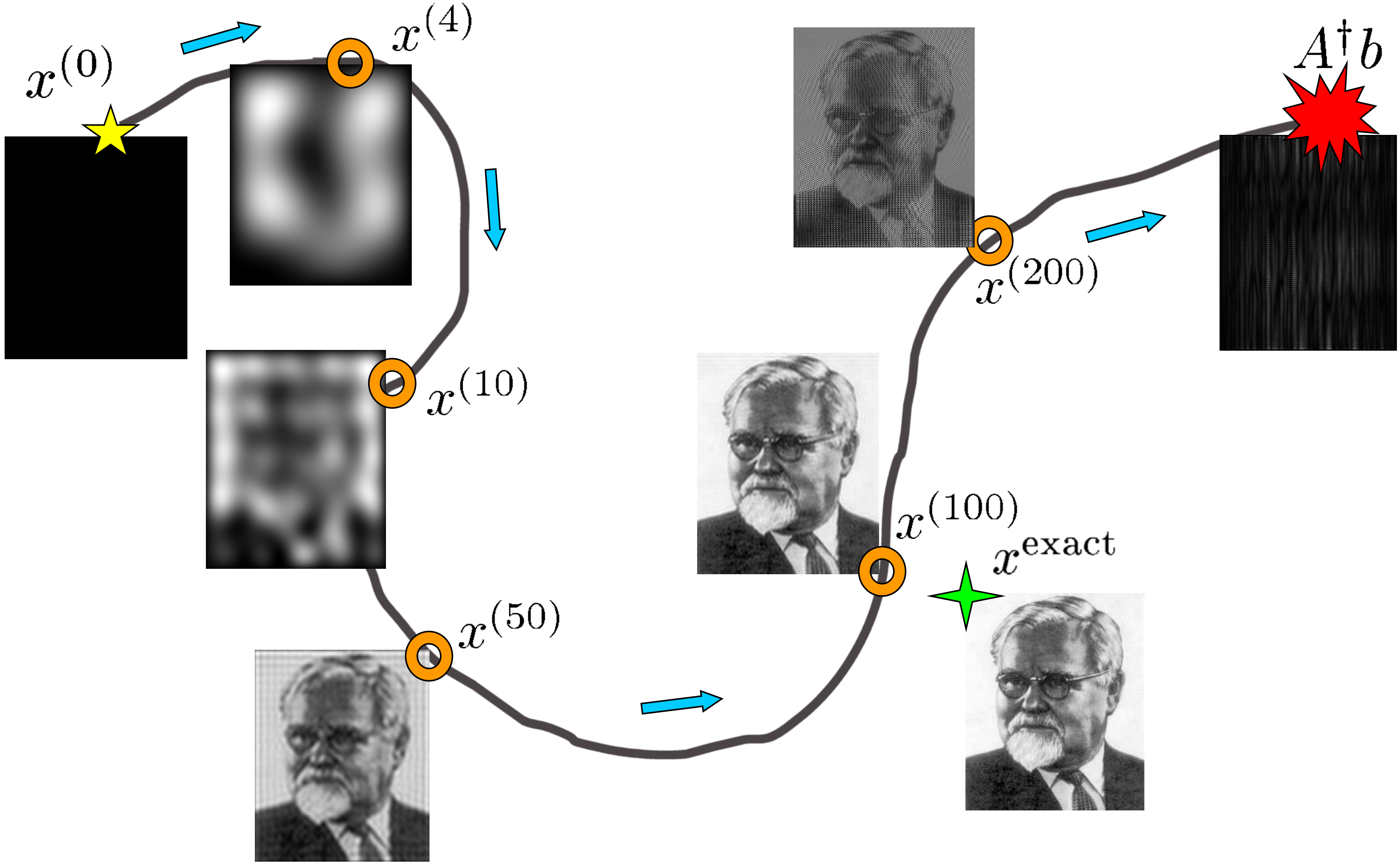
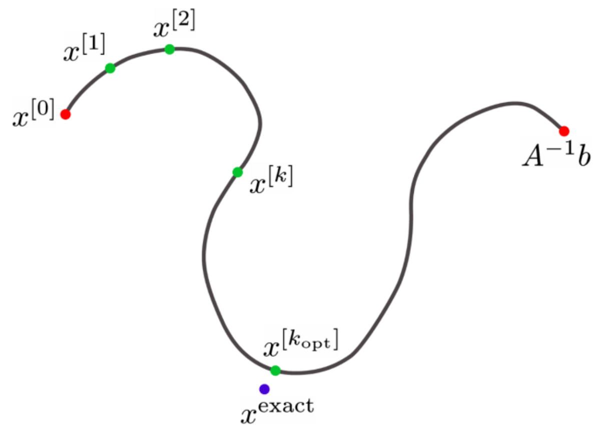
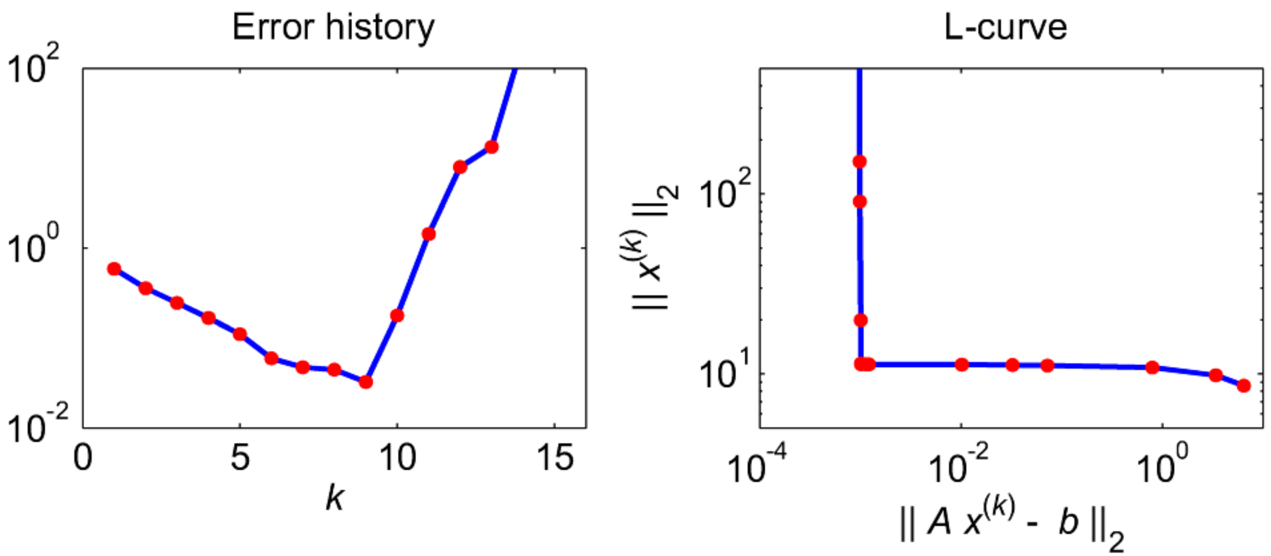


Illustration of Semi-Convergence

Recall this illustration:



The "ideal" behavior of the error $\| x^{(k)} - x^{\rm exact} \|_2$ and the associated L-curve:



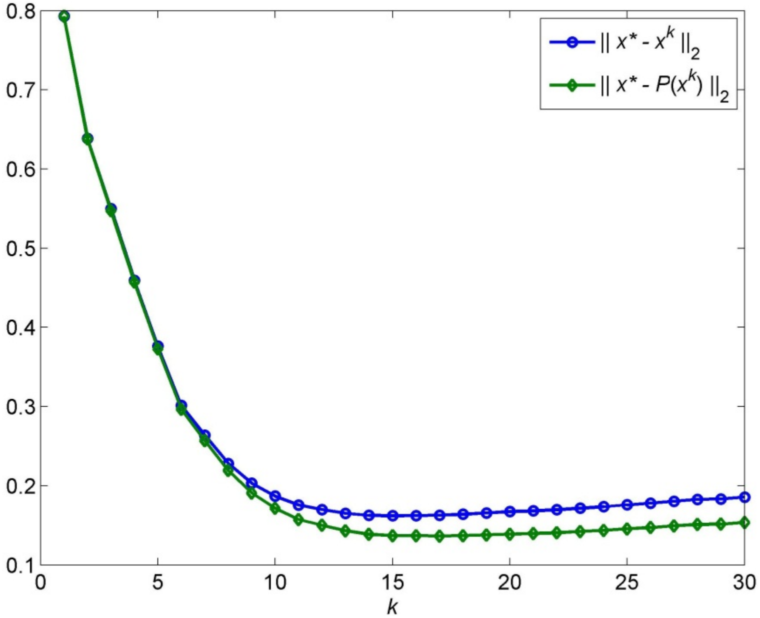
Matlab Example – AIR Tools

```
N = 128;        % Image size.
eta = 0.04;     % Rel. noise.
kmax = 30;      % No. Iterations.

% Test problem from AIR Tools.
[A,bex,xex] = fanbeamtomo(N);
e = randn(size(bex));
e = eta*norm(bex)*e/norm(e);
b = bex + e;
nex = norm(xex);

X = cglsAIR(A,b,1:kmax);
Xp = X;
Xp(Xp<0) = 0;
Xp(Xp>1) = 1;
for k=1:kmax
    err(k,1) = norm(xex - X(:,k))/nex;
    errp(k,1) = norm(xex - Xp(:,k))/nex;
end
```

Exact | k = 15 | k = 30



Advantages of the Krylov Subspace

The SVD basis vectors $v_1, v_2, \ldots$ are well suited for representation of A.

But this basis "does not know all there is to know" about the given problem; it can not utilize information about the right-hand side b.

The Krylov subspace $\mathcal{K}_k$ "knows" about the right-hand side and therefore adapts itself to the given problem, through the starting vector

$$A^T b = A^T A\, x^{\text{exact}} + A^T e = \textstyle\sum_{i=1}^n \sigma_i^2\, (v_i^T x^{\text{exact}})\, v_i + \sum_{i=1}^n \sigma_i\, (u_i^T e)\, v_i .$$

Hence the Krylov basis vectors are rich in those directions that are needed.

SVD analysis

$$x^{(k)} = \textstyle\sum_{i=1}^n \phi_i^{(k)} \frac{u_i^T b}{\sigma_i}\, v_i , \qquad \phi_i^{(k)} = 1 - \prod_{j=1}^k \frac{\theta_j^{(k)} - \sigma_i^2}{\theta_j^{(k)}}$$

Here $\theta_j^{(k)}$ are the Ritz values, i.e., the eigenvalues of the projection of $A^T A$ on the Krylov subspace $\mathcal{K}_k$. They converge to those σ_i^2 whose corresponding SVD components $u_i^T b$ are large.

The CGLS Filter Factors

A closer look at the filter factors $\phi_i^{(k)}$ in the filtered SVD expansion

$$x^{(k)} = \sum_{i=1}^{n} \phi_i^{(k)} \, \frac{u_i^T b}{\sigma_i} \, v_i$$

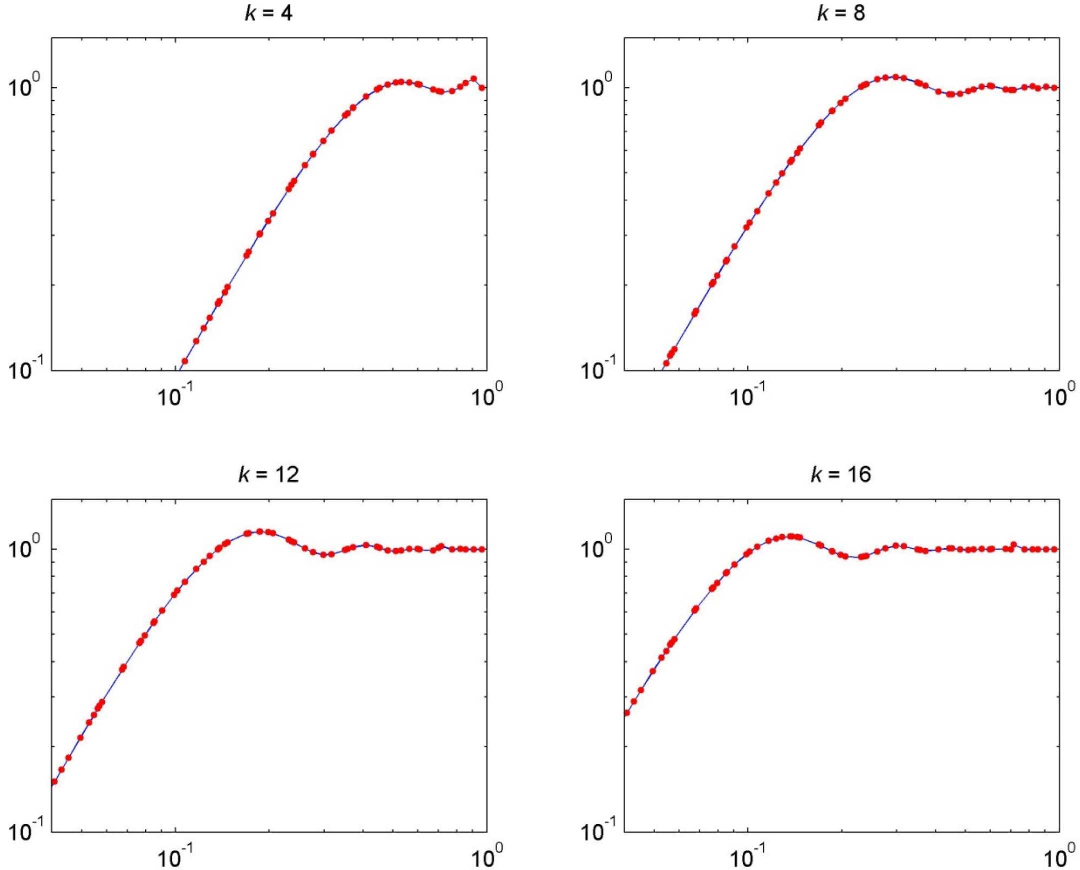
$$= V \, \Phi_k \, \Sigma^\dagger U^T b$$

$$\Phi_k = \mathcal{P}_k(\Sigma^2) \, \Sigma^2$$

Here $\mathcal{P}_k$ is a unique polynomial such that

$$x^{(k)} = \mathcal{P}_k(A^T A) \, A^T b.$$

Filter factors $\phi_i^{(k)}$

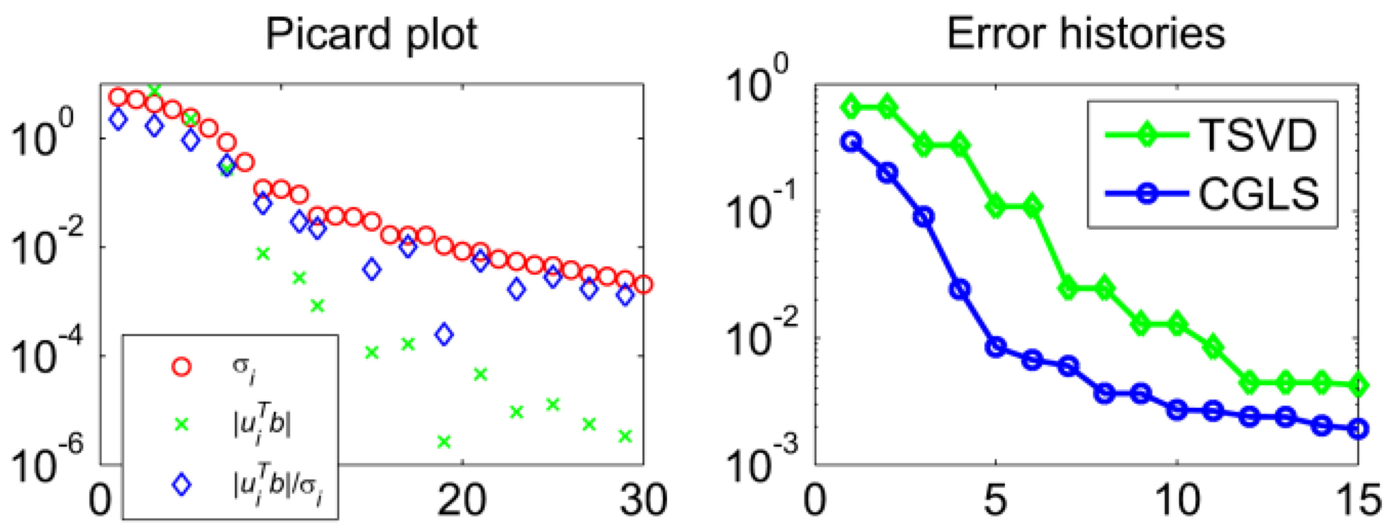


CGLS Focuses on Significant Components

Example: `phillips` from Regularization Tools.

Exact solution has many zero SVD coefficients.

- The TSVD solution x_k includes all coefficients from 1 thru k.
- The CGLS solution $x^{(k)}$ includes only those coefs. we need.



CGLS suppresses noise better than TSVD in this case.

Another Story: CGLS for Tikhonov

One could also use CGLS to solve the Tikhonov problem in the form

$$\min_x \left\| \begin{pmatrix} A \\ \lambda L \end{pmatrix} - \begin{pmatrix} b \\ 0 \end{pmatrix} \right\|_2^2 .$$

But this approach typically requires that the system is solved many times, for many diffrent values of λ.

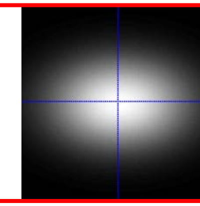
Also, preconditioning is often necessary – but it can be difficult to design a good preconditioner for the Tikhonov problem.

We shall not pursue this aspect further in this talk.

Other Krylov Subspace Methods

Sometimes it is impractical to use methods – such as CGLS – that need A^T, e.g, if $A = A^T$ or if we have a black-box function that computes $A\,x$.

A is symmetric, e.g., if the PSF is "doubly symmetric."



MINRES and GMRES come to mind if the matrix A is square – these methods are based on the Krylov subspace:

$$\mathcal{K}_k = \text{span}\{b, Ab, A^2b, \ldots, A^{k-1}b\}.$$

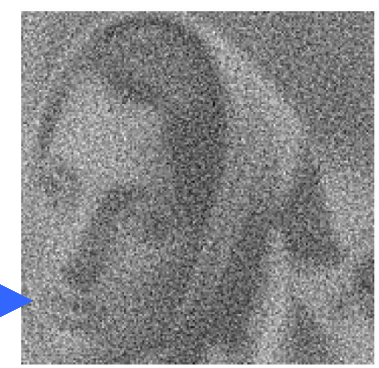
Unfortunately it is a bad idea to include the noisy vector b in the subspace.

Fewer GMRES than CGLS iterations before noise enters.

GMRES tends to give noisier images!

$k_{\text{CGLS}} = 70$

$k_{\text{GMRES}} = 15$

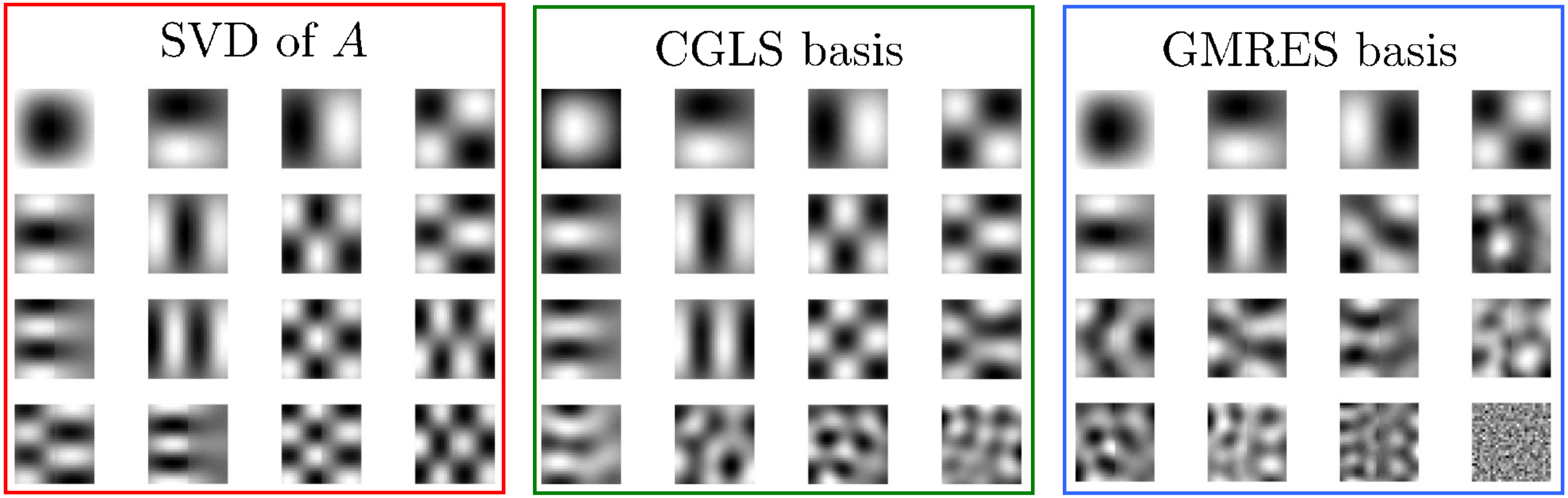


GMRES and CGLS Basis Vectors

Truncated SVD subspace = $\text{span}\{v_1, v_2, v_3, \ldots\}$.

CGLS subspace = $\text{span}\{A^T b, (A^T A) A^T b, (A^T A)^2 A^T b, \ldots\}$.

GMRES subspace = $\text{span}\{b, Ab, A^2 b, \ldots\}$.



The GMRES basis always includes a "noisy" basis vector, due to the presence of b in the Krylov subspace.

Other Krylov Subspace Methods Continued

A better choice is the "shiftet" Krylov subspace:

$$\vec{\mathcal{K}}_k = \operatorname{span}\{Ab, A^2b, \ldots, A^kb\}.$$

The corresponding methods are called MR-II and RRGMRES (both are now included in Regularization Tools).

Examples on next slides

Comparing Krylov Methods: MINRES, MR-II

♥ The presence of *b* in the MINRES Krylov subspace gives very noisy solutions.

♠ The absence of *b* in the MR-II Krylov subspace is essential for the noise reduction.

♣ MR-II computes a filtered SVD solution:

$$x^{(k)} = V\,\Phi_k\,\Sigma^\dagger V^T b$$

$$\Phi_k = \mathcal{P}_k(\Omega\,\Sigma)\,\Omega\,\Sigma$$

$$\Lambda = \Omega\,\Sigma, \quad \Omega = \mathrm{diag}(\pm 1)$$

♦ Negative eigenvalues of *A* do not inhibit the regulalarizing effect of MR-II, but they can slow down the convergence.

Comparing: GMRES, RRGMRES

♥ The presence of b in the GMRES Krylov subspace gives very noisy solutions.

♠ The absence of b in the RRGMRES Krylov subspace is essential for the noise reduction.

♣ RRGMRES *mixes* the SVD components in each iteration and $x^{(k)}$ is not a filtered SVD solution:

$$x^{(k)} = V\,\Phi_k\,\Sigma^\dagger U^T b$$

$$\Phi_k = \mathcal{P}_k(C\,\Sigma)\,C\,\Sigma$$

$$C = V^T U$$

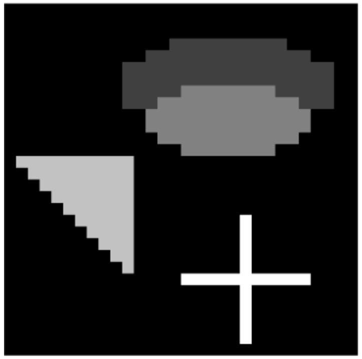
♦ RRGMRES works well if the mixing is weak (e.g., if $A \approx A^T$), or if the Krylov basis vectors are well suited for the problem.

MINRES / MR-II Case Study

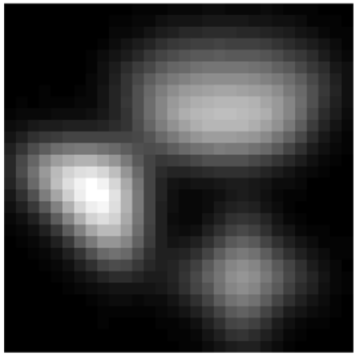
$$A = A^T \quad \text{and} \quad PA = (PA)^T, \quad P = \text{reversal matrix.}$$

Use CGLS, MINRES and MR-II to solve the two problems

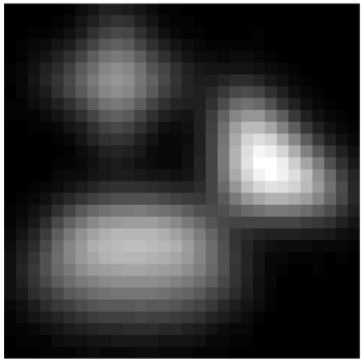
$$A\,x = b \quad \text{and} \quad P\,A\,x = P\,b.$$



$x = \mathrm{vec}(X)$



$b = \mathrm{vec}(B)$

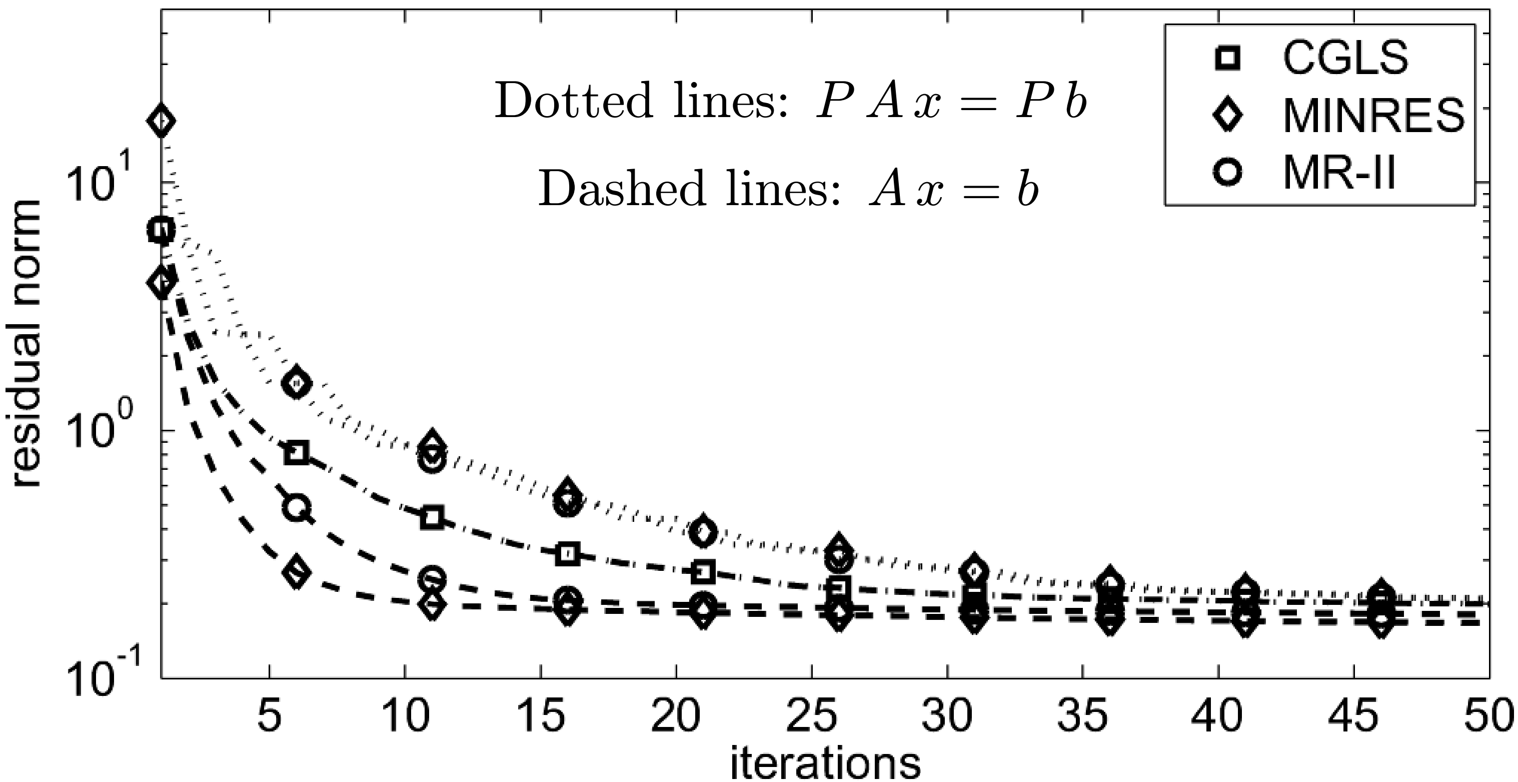


$P\,b$

CGLS behaves identically on both problems because $(PA)^T(PA) = A^T A$.

MINRES and MR-II have different Krylov subspaces = signal subspaces and, therefore, different convergence histories.

MINRES / MR-II Case Study – Results



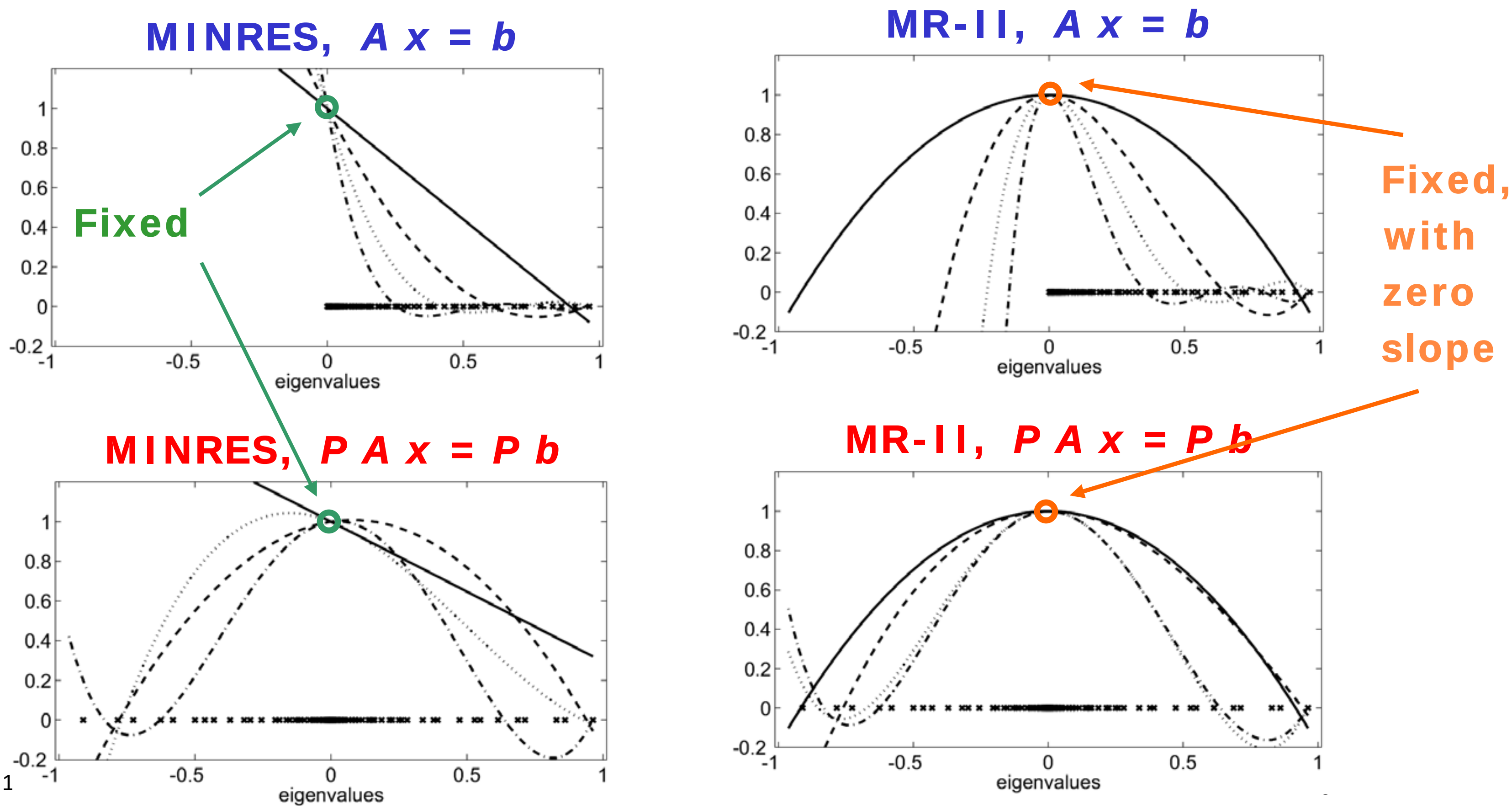
- Permuted problem: approx. same convergence of MINRES and MR-II; slower than CGLS.
- Original problem: MINRES converges faster than MR-II; both are faster han CGLS.

MINRES / MR-II Case Study – Insight

Residual polynomials for MINRES solution $x^{(k)}$ and MR-II solution $\bar{x}^{(k)}$:

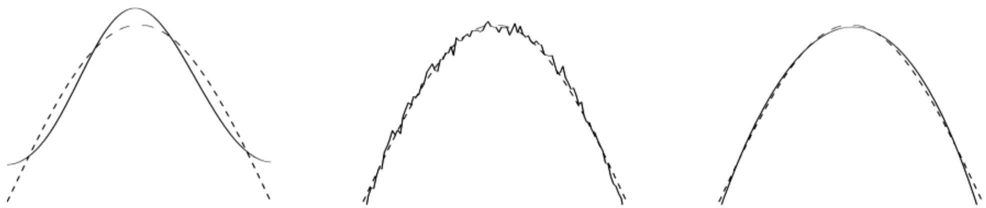
$$b - A\,x^{(k)} = \mathcal{Q}_k(A)\,b, \qquad b - A\,\bar{x}^{(k)} = \bar{\mathcal{Q}}_k(A)\,b$$

Must "kill" residual components corresp. to largest (in magnitude) eigenvalues



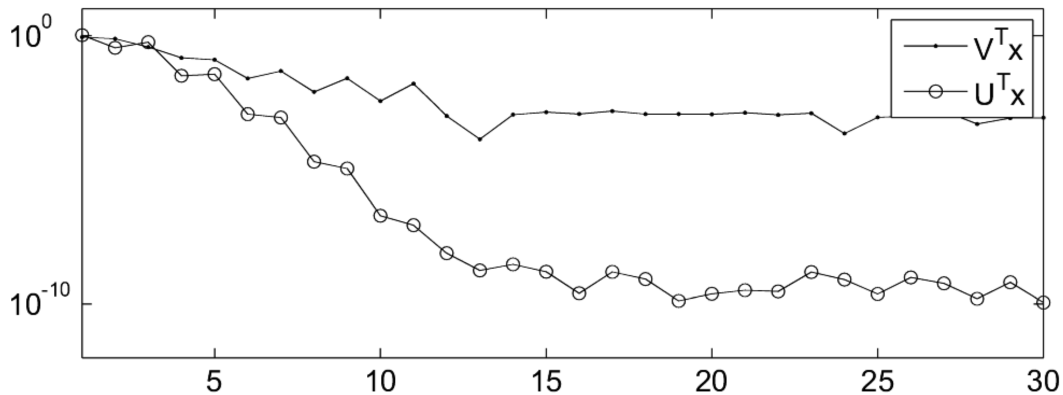
GMRES and RRGMRES

Test problem **baart** from REGULARIZATION TOOLS – nonsymmetric A.



CGLS $k = 4$ GMRES $k = 3$ RRGMRES $k = 3$

RRGMRES provides a better solution subspace than GMRES, because the noisy b is not included in the Krylov subspace!



Solution coefs. in left and right SVD basis.

The SVD's U basis gives a faster expansion of x than the V basis for *this problem*. Hence RRGMRES produces better iterates than CGLS.

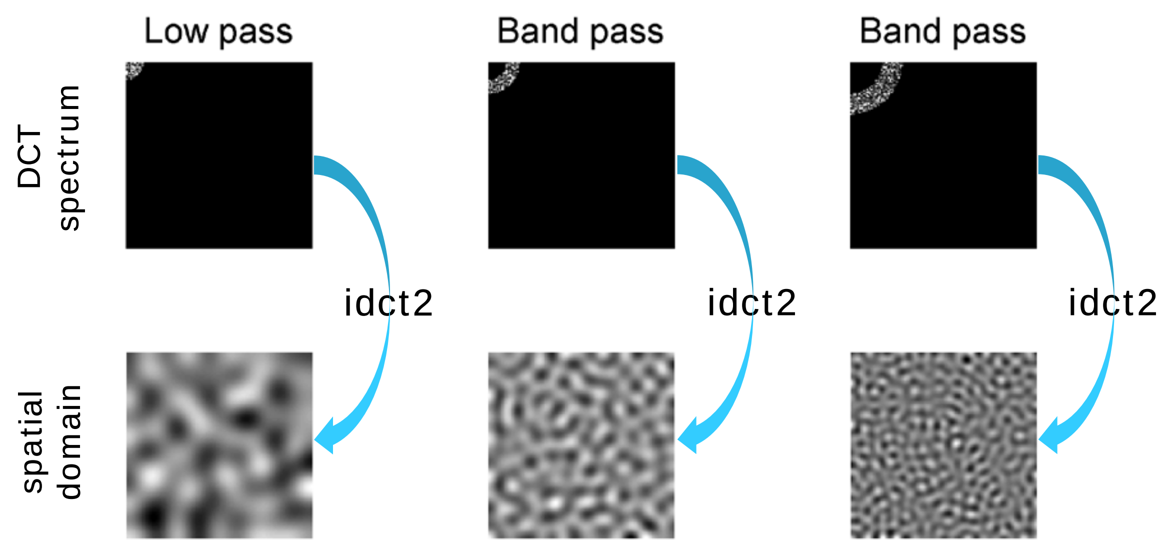
Back to CGLS: The "Freckles"

CGLS:
k = 4, 10
and 25
iterations

Initially, the image gets sharper – then "freckles" start to appear.

Low frequencies carry the main information.

"Freckles" are band-pass filtered noise.



Noise Propagation

Recall once again that we can write the CGLS solution as:

$$x^{(k)} = \mathcal{P}_k(A^T A)\, A^T b,$$

where $\mathcal{P}_k$ is the polynomium associated with the Krylov subspace $\mathcal{K}_k(A^T b, A^T A)$.

Thus $\mathcal{P}_k$ is fixed by A and b, and if $b = b^{\mathrm{exact}} + e$ then

$$x^{(k)} = \mathcal{P}_k(A^T A)\, A^T b^{\mathrm{exact}} + \mathcal{P}_k(A^T A)\, A^T e \equiv x^{(k)}_{b^{\mathrm{exact}}} + x^{(k)}_e .$$

Similarly for the other iterative methods.

Signal component

Noise component

Note that signal component $x^{(k)}_{b^{\mathrm{exact}}}$ depends on the noise e via $\mathcal{P}_k$.

Signal and Noise Components

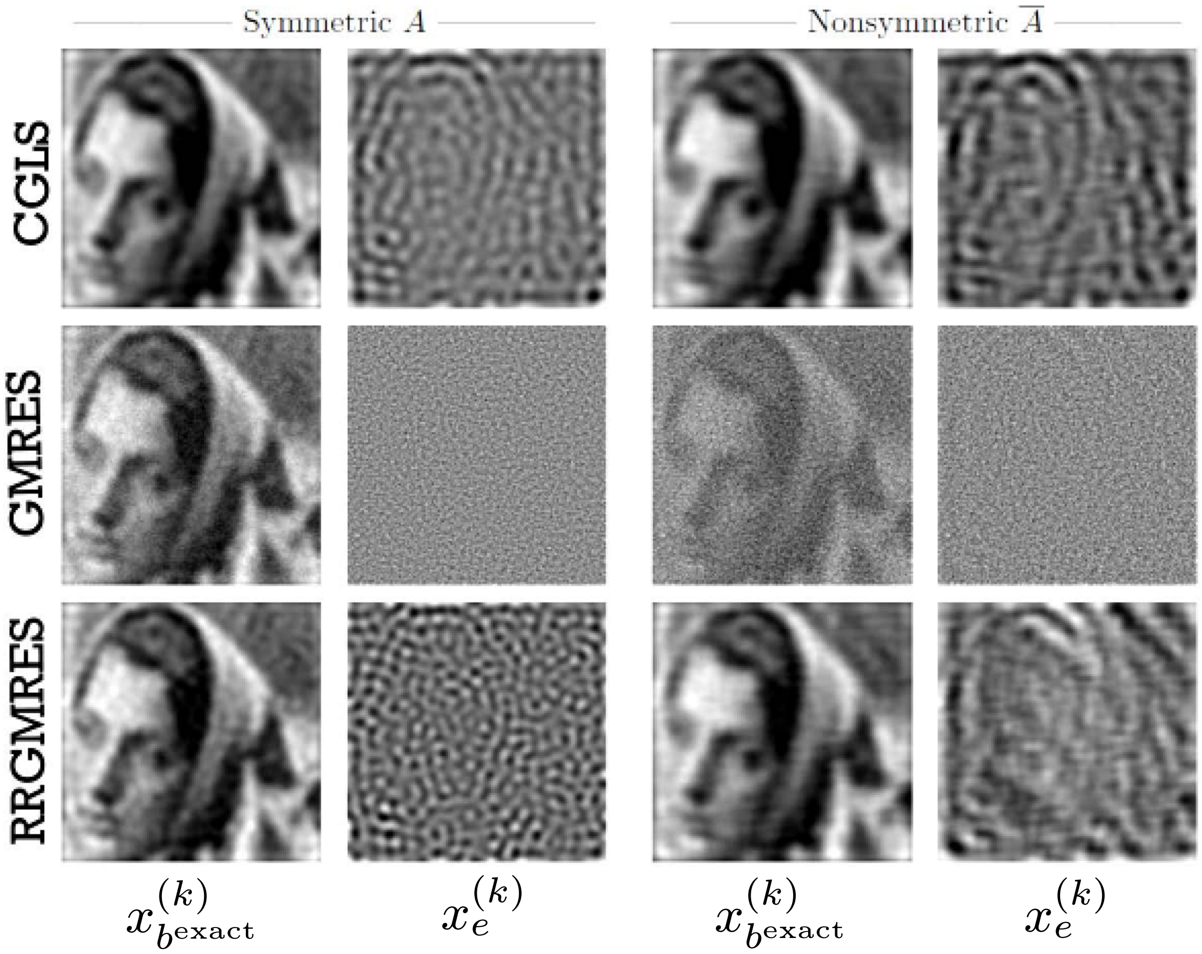
$$x^{(k)} = \mathcal{P}_k(A^TA)\,A^Tb = \underbrace{\mathcal{P}_k(A^TA)\,A^Tb^{\mathrm{exact}}}_{\text{signal}} + \underbrace{\mathcal{P}_k(A^TA)\,A^Te}_{\text{noise}} \ .$$

Note that the noise components (the freckles) are *correlated* with structures in the image!

Tends to *mask* the appearance of the noise!!

Two different matrices *A*

Same Behavior in All Methods



The noise components are always correlated with the image!

Yet Another Krylov Subspace Method

If certain components (or features) are missing from the Krylov subspace, then it makes good sense to *augment* the subspace with these components.

Augmented (RR)GMRES does precisely that:

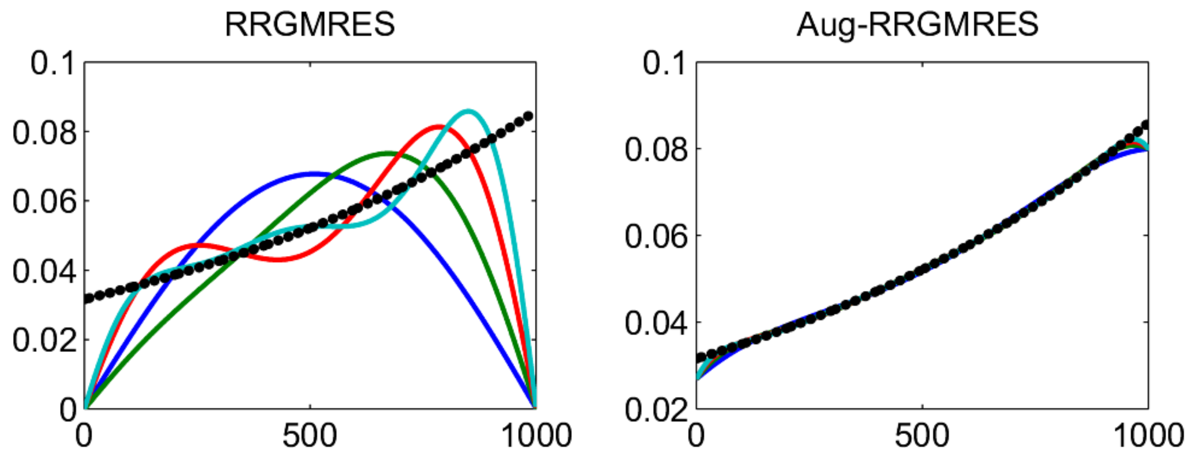
$$\mathcal{S}_k = \mathrm{span}\{w_1, \ldots, w_p\} + \mathrm{span}\{b, Ab, A^2 b, \ldots, A^{k-1} b\}.$$

$$\vec{\mathcal{S}}_k = \mathrm{span}\{w_1, \ldots, w_p\} + \mathrm{span}\{Ab, A^2 b, A^3 b, \ldots, A^k b\}.$$

Example: deriv2.

All vectors in the Krylov subspace $\to 0$ at the ends.

$w_1 = (1,1,\ldots,1)^T$
$w_2 = (1,2,\ldots,n)^T$



Implementation Aspects, RRGMRES

Baglama & Reichel (2007) proposed algorithm AugRRGMRES that uses the simple formulation

$$A\,W_p = V_p\,H_0 \quad \to \quad A\,\big[\,W_p\,,\,V_k\,\big] = \big[\,V_p\,,\,V_{k+1}\,\big]\,H_k\ .$$

But their algorithm actually solves the problem

$$\min_x \|A\,x - b\|_2^2 \quad \text{s.t.} \quad x \in \mathcal{W}_p + \mathcal{K}_j\big((I - V_p V_p^T)A, (I - V_p V_p^T)Ab\big)\ .$$

Dong, Garde & H recently proposed an alternative algorithm $\mathrm{R^3GMRES}$ (Regularized RRGMRES) that uses the desired subspace

$$\mathcal{W}_p + \mathcal{K}_j(A, Ab)\ .$$

Their algorithm is a bit more complicated, but has the same complexity as RRGMRES and AugRRGMRES.

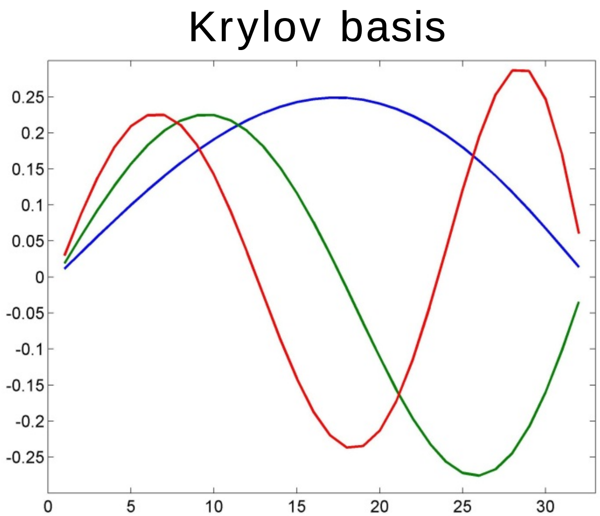
Test Problem: "deriv2" (Reg. Tools)

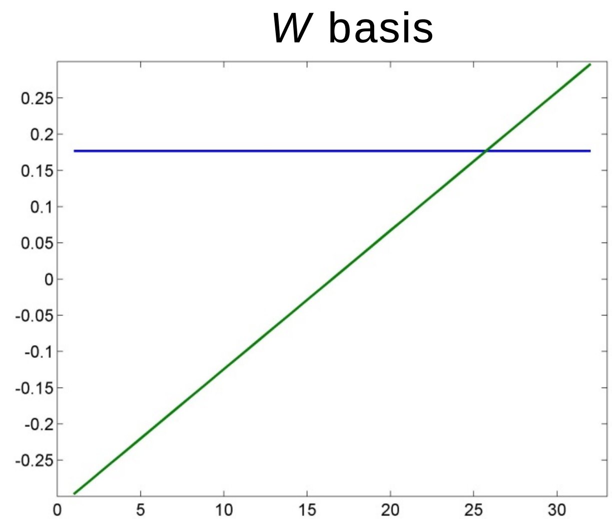
First-kind Fredholm integral equation with kernel

$$K(s,t) = \begin{cases} s(t-1)\ , & s < t \\ t(s-1)\ , & s \geq t \end{cases}$$

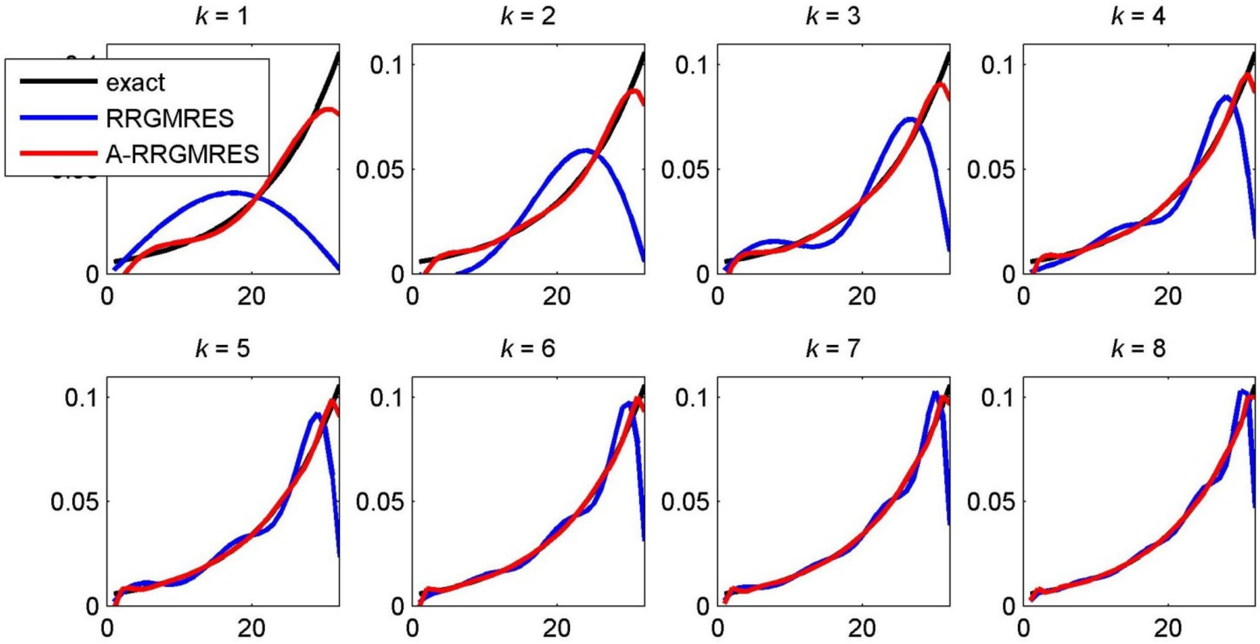
Augmentation basis – does not approach 0 at the ends of the interval:

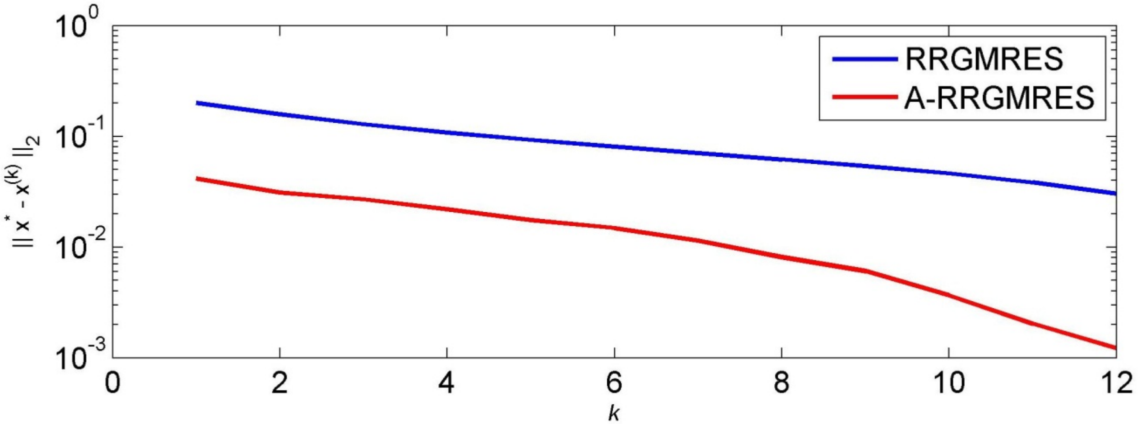
$$w_1 = (\,1, 1, \ldots, 1\,)^T, \qquad w_2 = (\,1, 2, \ldots, n\,)^T.$$





Numerical Results





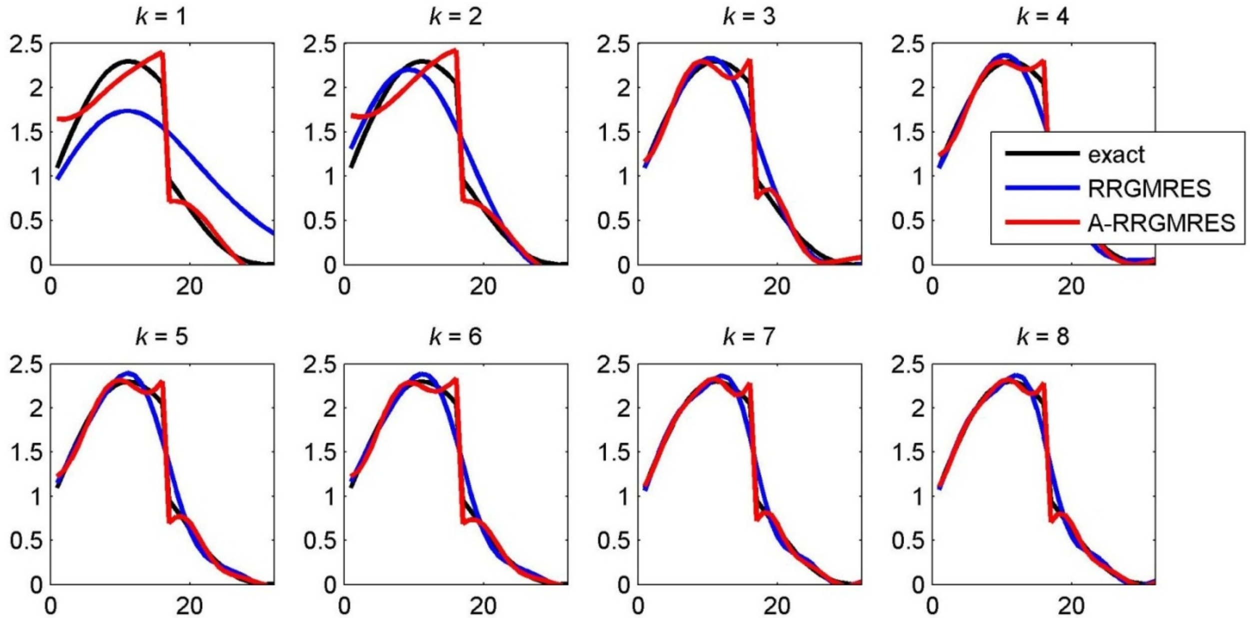
Test Problem: “gravity” with Discontinuity

First-kind Fredholm integral equation with kernel

$$K(s,t) = d\big(d^2 + (s-t)^2\big)^{-3/2}.$$

Augmentation basis – allows a discontinuity at a known position:

$$w_1 = (\,1,\ldots,1,0\ldots,0\,)^T, \qquad w_2 = (\,0,\ldots,0,1,\ldots,1\,)^T.$$



General-Form Tikhonov Regularization

CGLS is linked to the SVD of A and thru the Krylov subspace, the Ritz polynomium, and the convergence of the Ritz values.

Thus CGLS is also related to Tikhonov regularization in standard form

$$\min_x \left\{ \|A\,x - b\|_2^2 + \lambda^2\, \|x\|_2^2 \right\}$$

But occationally we prefer the *general* formulation

$$\min_x \left\{ \|A\,x - b\|_2^2 + \lambda^2\, \|L\,x\|_2^2 \right\}, \qquad L \neq I.$$

But CGLS can only see the LS problem $\|A\,x - b\|_2^2$ with no regularization term.

How do we modify CGLS such that it can incorporate the matrix L?

We must *modify* the Krylov subspace underlying the method!

Standard-Form Transformation

We are given:

$$\min_x \left\{ \|A\,x - b\|_2^2 + \lambda^2\,\|L\,x\|_2^2 \right\}, \qquad L \neq I.$$

If L is invertible, we can rewrite the above as:

$$\min_{\bar{x}} \|(A\,L^{-1})\,\bar{x} - b\|_2^2 + \lambda^2 \|\bar{x}\|_2^2 \qquad \text{with} \qquad \bar{x} = L\,x \quad \Leftrightarrow \quad x = L^{-1}\bar{x}.$$

In the general case, use the *standard-form transformation:*

$$\min_{\bar{x}} \|\bar{A}\,\bar{x} - b\|_2^2 + \lambda^2 \|\bar{x}\|_2^2 \qquad \text{with} \qquad \bar{A} = A\,L^{\#} \quad \text{and} \quad x = L^{\#}\bar{x} + x_{\mathcal{N}},$$

where $L^{\#}$ = oblique pseudoinverse of L and $x_{\mathcal{N}} \in \mathcal{N}(L)$.

Subspace Preconditioning

If we apply CGLS to the standard-form problem

$$\min_{\bar{x}} \|\bar{A}\,\bar{x} - b\|_2^2 + \lambda^2 \|\bar{x}\|_2^2,$$

then the iterates, when transformed back via $L^{\#}$, lie in the affine space

$$\mathrm{span}\{M A^T b, (M A^T A)\, M A^T b, (M A^T A)^2 M A^T b, \ldots\} + x_{\mathcal{N}} \ ,$$

where $M = L^{\#}(L^{\#})^T$.

Hence L is a preconditioner for CGLS that provides a better suited subspace.

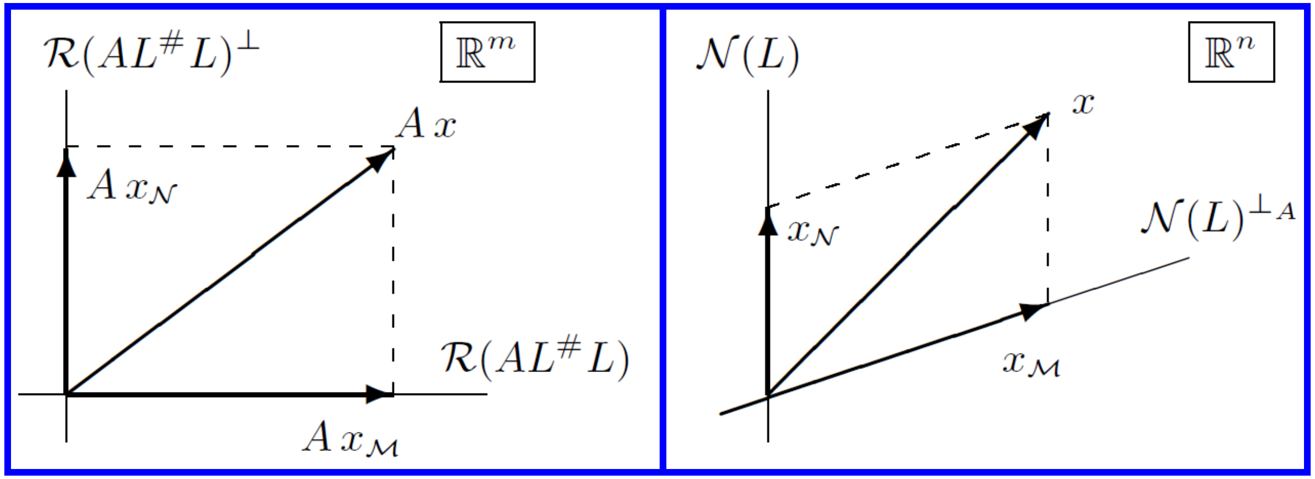
The Krylov subspace methods are implemented such that $\bar{A}$ is never formed.

How is the oblique pseudoinverse $L^{\#}$ defined? And why this particular matrix?

Outside scope of this talk.

Next slide please …

Splitting!



Write $x = x_{\mathcal{M}} + x_{\mathcal{N}}$ with $x_{\mathcal{N}} \in \mathcal{N}(L)$ and $x_{\mathcal{M}}$ being $A^T A$–orthogonal to $x_{\mathcal{N}}$.

This corresponds to an *oblique* splitting of the subspace $\mathbb{R}^n$.

Then the vector $A\,x = A\,x_{\mathcal{M}} + A\,x_{\mathcal{N}}$ splits into two *orthogonal* components.

The Tikhonov problem reduces to two independent problems for $x_{\mathcal{M}}$ and $x_{\mathcal{N}}$:

$$\min \|A\,x_{\mathcal{M}} - b\|_2^2 + \lambda^2 \|x_{\mathcal{M}}\|_2^2 \qquad \text{and} \qquad \min \|A\,x_{\mathcal{N}} - b\|_2^2 .$$

Since $x_{\mathcal{M}} = L^{\#} L\,x$ we get $A\,x_{\mathcal{M}} = (A\,L^{\#})\,(L\,x) \to$ the standard-form problem.

More About Subspace Preconditioning

To summarize the subspace preconditinong idea:

$$x = L^{\#}\,\bar{x} + x_{\mathcal{N}}, \qquad \text{solve} \quad \|(A\,L^{\#})\,\bar{x} - b\|_2$$

where $x_{\mathcal{N}} \in \text{null}(L)$ and $L^{\#}$ = weighted pseudoinverse of L.

Compute $\bar{x}^{(k)}$ via regularizing iterations ($\mathcal{P}_k$ = polynomial):

$$\bar{x}^{(k)} = \mathcal{P}_k\big((A\,L^{\#})^T(A\,L^{\#})\big)\,(A\,L_A^{\dagger})^T b.$$

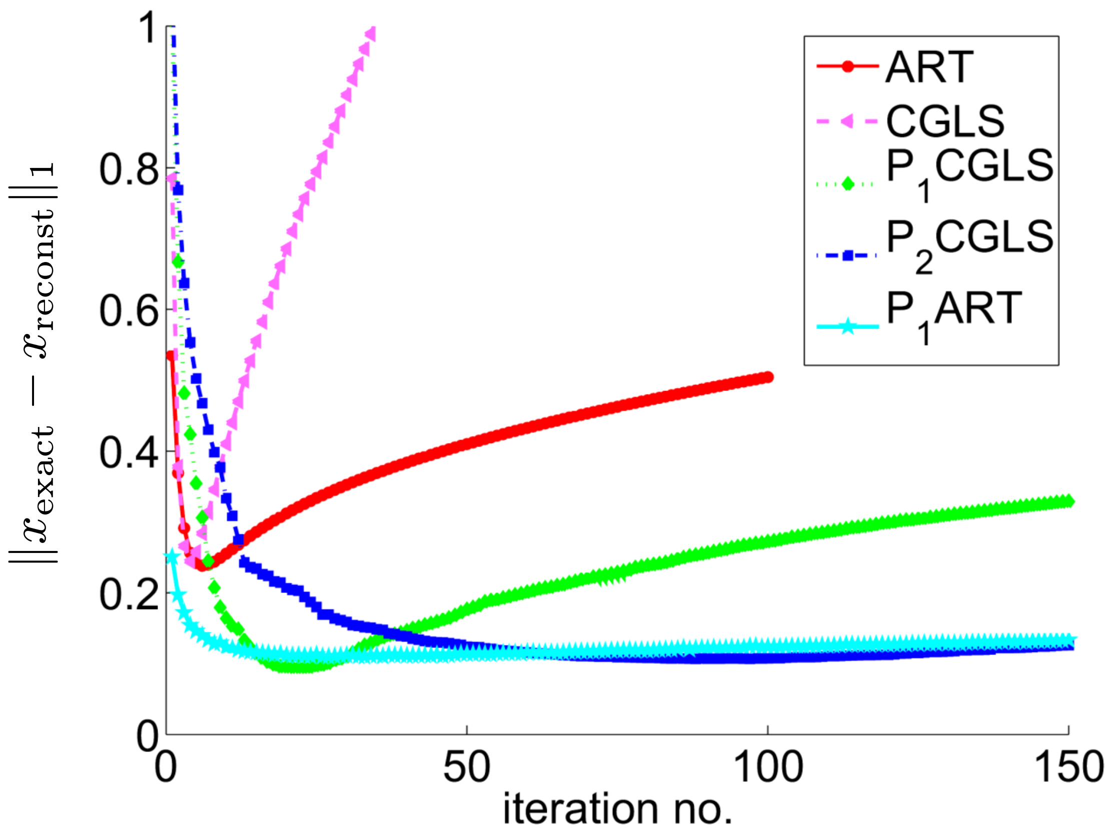
Insertion shows that

$$x^{(k)} = L^{\#}\bar{x}^{(k)} + x_{\mathcal{N}} = \mathcal{P}_k\big(M\,A^T A\big)\,M\,A^T b + x_{\mathcal{N}},$$

where $M = L^{\#}(L^{\#})^T$ acts as a preconditioner that ensures a solution in the desired subspace. See Reg. Tools for implementation details.

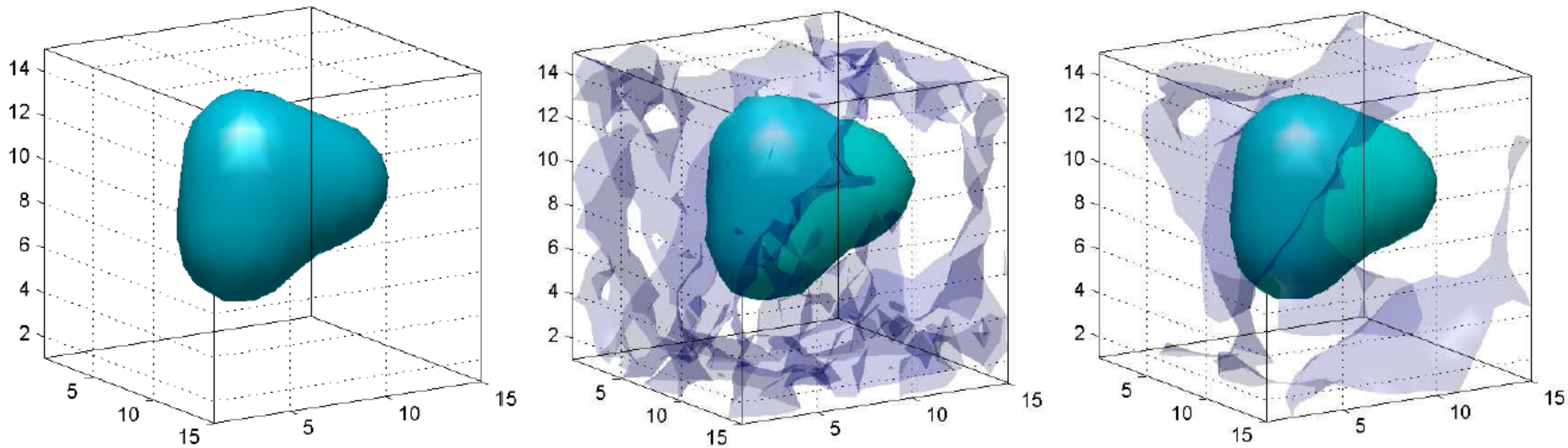
Convergence Histories

Example from tomography (reconstruction of smooth function).



Standard methods are inferior to preconditioned versions!

Why Preconditioning is Necessary



Exact smooth solution.

Reconstruction with $L = I$, very noisy near boundaries.

Smooth reconstruction with $L \neq I$.

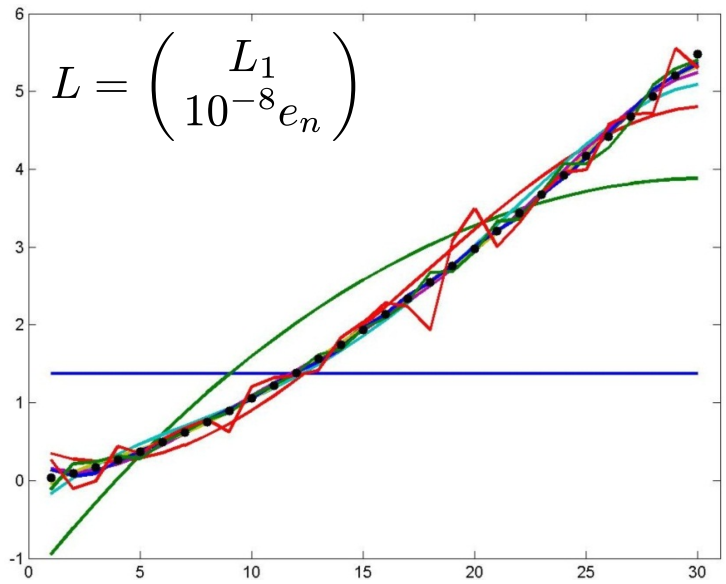
Much fewer rays penetrate the domain near the boundaries.

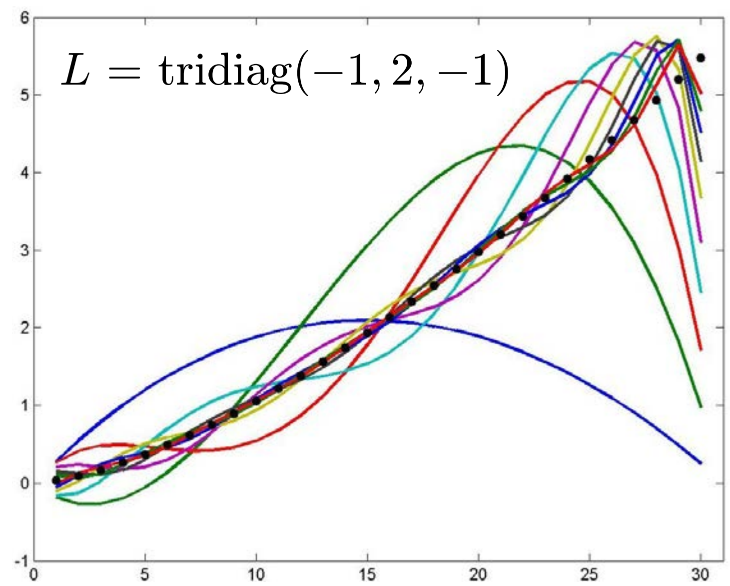
Preconditioning for GMRES

Preconditioning is easy if L is invertible:

Use GMRES to solve $(A\,L^{-1})\,z = b \quad \rightarrow \quad$ then set $x = L^{-1}z$.

A rectangular L can be augmented – but be careful!





Both choices of L have severe difficulites at the right end of the interval.

A Better Approach: Use a Square matrix

1. Write

$$x = L_A^\dagger \, y + N \, z = \left(L_A^\dagger \, , \, N \right) \begin{pmatrix} y \\ z \end{pmatrix}, \qquad \text{range}(N) = \text{null}(L).$$

2. Consider the square system

$$\left(L_A^\dagger \, , \, N \right)^T A \left(L_A^\dagger \, , \, N \right) \begin{pmatrix} y \\ z \end{pmatrix} = \left(L_A^\dagger \, , \, N \right)^T b.$$

3. Precompute $x_0 = N \, z$.

4. Use GMRES to solve the Schur complement system

$$(L_A^\dagger)^T E \, A \, L_A^\dagger \, y = (L_A^\dagger)^T E \, b$$

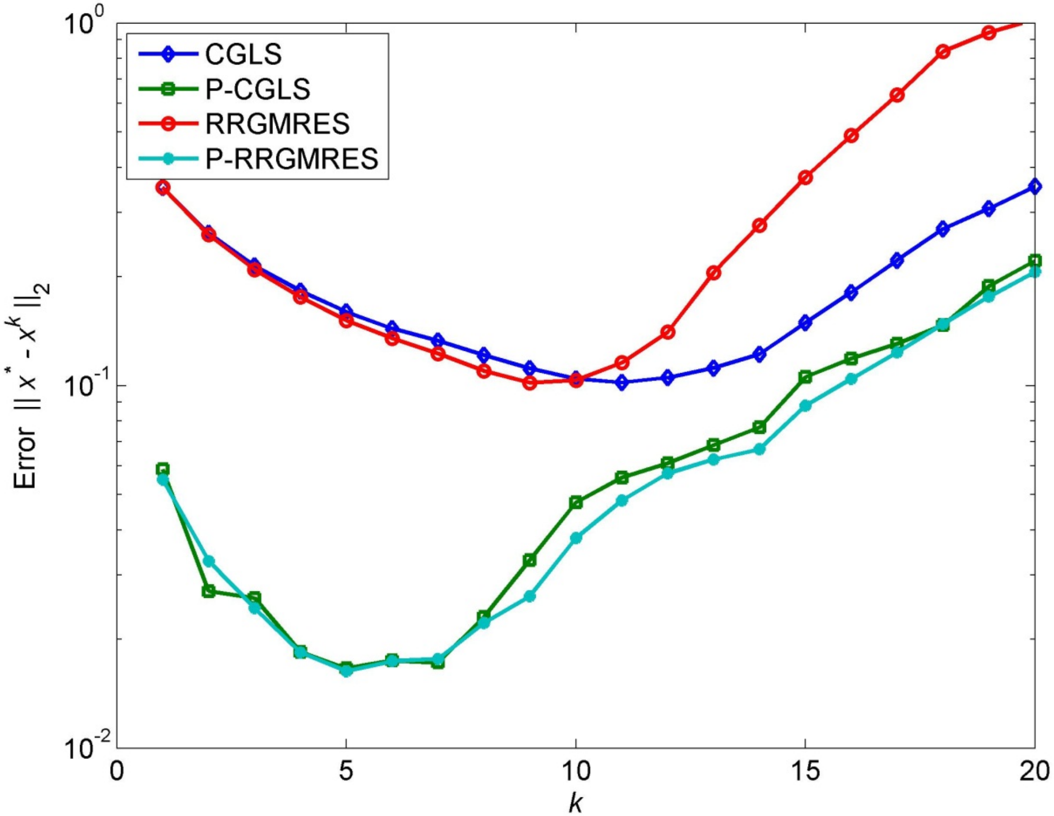
where E is the oblique projection on $\mathcal{R}(AN)$ along $\mathcal{R}(L^T)$.

All operations with $L_A^\dagger$ are done as for CGLS, cf. (H & Jensen, 2005).

(P)CGLS and (P)RRGMRES

Test problem deriv2 from REGULARIZATION TOOLS with $L = L_1$.

P-CGLS and P-RRGMRES have similar convergence; they are faster than CGLS and RRGMRES and give more accurate results.



(P)CGLS and (P)RRGMRES – Matlab Code

```
n = 64;  eta = 0.001;  k = 20;  reorth = 1;  % Set parameters.

[A,bex,xex] = deriv2(n);  % Define the noisy test problem.
e = randn(n,1); e = norm(bex)*eta*e/norm(e);
b = bex + e;

[L,W] = get_l(n,1);  % First derivative smoothing.

Xcgls = cgls(A,b,k,reorth);          % (P)CGLS solutions.
Xpcgls = pcgls(A,L,W,b,k,reorth);

Xrrgmres = rrgmres(A,b,k);           % (P)RRGMRES solutions.
Xprrgmres = prrgmres(A,L,W,b,k);

for i=1:k   % Compute the errors.
    ecgls(i,1) = norm(xex-Xcgls(:,i));
    epcgls(i,1) = norm(xex-Xpcgls(:,i));
    errgmres(i,1) = norm(xex-Xrrgmres(:,i));
    eprrgmres(i,1) = norm(xex-Xprrgmres(:,i));
end
```

Stopping Rules = Reg. Param. Choice

The classical stopping rule for iterative methods is:

- Stop when the residual norm $\| b - A\, x^{(k)} \|_2$ is ”small.”

It does not work for ill-posed problems: a small residual norm does not imply that $x^{(k)}$ is close to the exact solution!

Must stop when all available information has been extracted from the right-hand side b, just before the noise start to dominate $x^{(k)}$.

- discrepancy principle,
- generalized cross validation (GCV),
- L-curve criterion (?),
- normalized cumulative periodogram (NCP),
- and probably perhaps others …

Conclusion

- Deblurring is an ill-posed problem
- Regularization by projection is suited for large-scale problems
- CGLS = projection on Krylov subspace
- CGLS = spectral filtering method (SVD basis)
- Another Krylov subspace: span{Ab,A2b,A3b,...}
- The noise component is correlated with the signal component
- Augmentation $\rightarrow$ improved subspace
- Subspace preconditioning $\rightarrow$ improved subspace.