Hybrid Enriched Bidiagonalization for Discrete Ill-Posed Problems

Per Christian Hansen
Technical University of Denmark

Joint work with

Kuniyoshi Abe - Gifu Shotoku Gakuen University

Yiqiu Dong - Technical University of Denmark

With thanks to
Henrik Garde and Nao Kuroiwa

DTU Compute
Department of Applied Mathematics and Computer Science

Setting the Stage – Overview of Talk

Forward problem:
$b = Ax$

Inverse problem:
solve $Ax = b$

1. Iterative Krylov-subspace methods – regularizing iterations.
2. Enrichment: augmenting the Krylov subspace.
3. Golub-Kahan bidiagonalization algorithm with augmented subspace.
4. A hybrid version with Tikhonov regularization.
5. Numerical examples that illustrate the advantage of this idea.

Inverse Problems → Ill-Conditioning

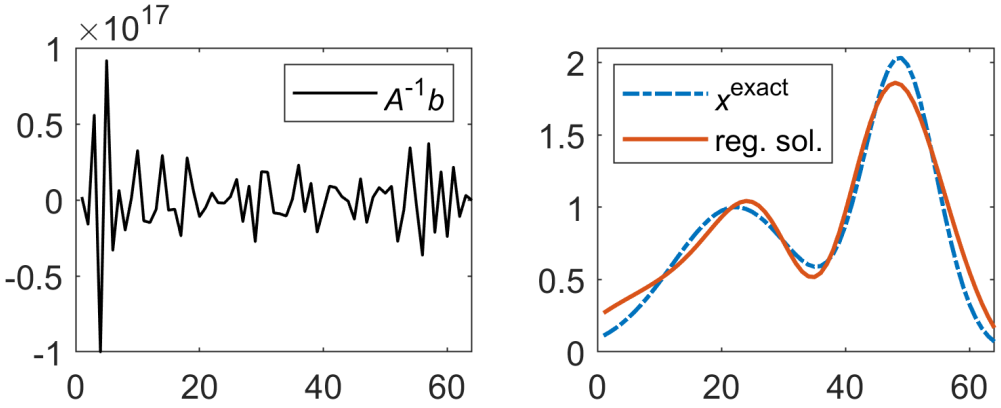
The underlying problem $\boxed{\mathcal{A}\, f = g}$ is ill posed:

Arbitrarily small perturbations of g can produce arbitrarily large perturbations of the solution f.

The discretized problem $\boxed{A\, x = b}$ has an ill-conditioned coefficient matrix A, and the "naive solution" $x = A^{-1} b$ is useless:

$$x = A^{-1}(b^{\text{exact}} + e) = A^{-1}(A\, x^{\text{exact}} + e) = x^{\text{exact}} + A^{-1} e \ .$$

We can *approximate* the exact solution by means of **regularization**.



Regularization Algorithms

Variational formulations take the form

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

where $\mathcal{R}(x)$ is a *regularization term* that penalizes unwanted features in the solution, and λ is a user-chosen regularization parameter.

Projection formulations take the form

$$\min_x \|A\,x - b\|_2^2 \qquad \text{s.t.} \quad x \in \mathcal{S}_k \ ,$$

where the *"signal subspace"* $\mathcal{S}_k$ is a linear subspace of dimension k.

If $\mathcal{S}_k$ is chosen such that it captures the main features in the solution, then this approach is well suited for large-scale problems.

Hybrid methods are iterative methods that combine regularization and projection $\rightarrow$ this talk.

Krylov Subspaces and Semi-Convergence

In some applications we can use a *pre-determined subspace*, e.g., spanned by the Fourier basis, the discrete cosine basis, a wavelet basis, etc.

An example: truncated SVD

$$\mathcal{S}_k = \text{span}\{v_1,\, v_2,\, \ldots,\, v_k\}.$$

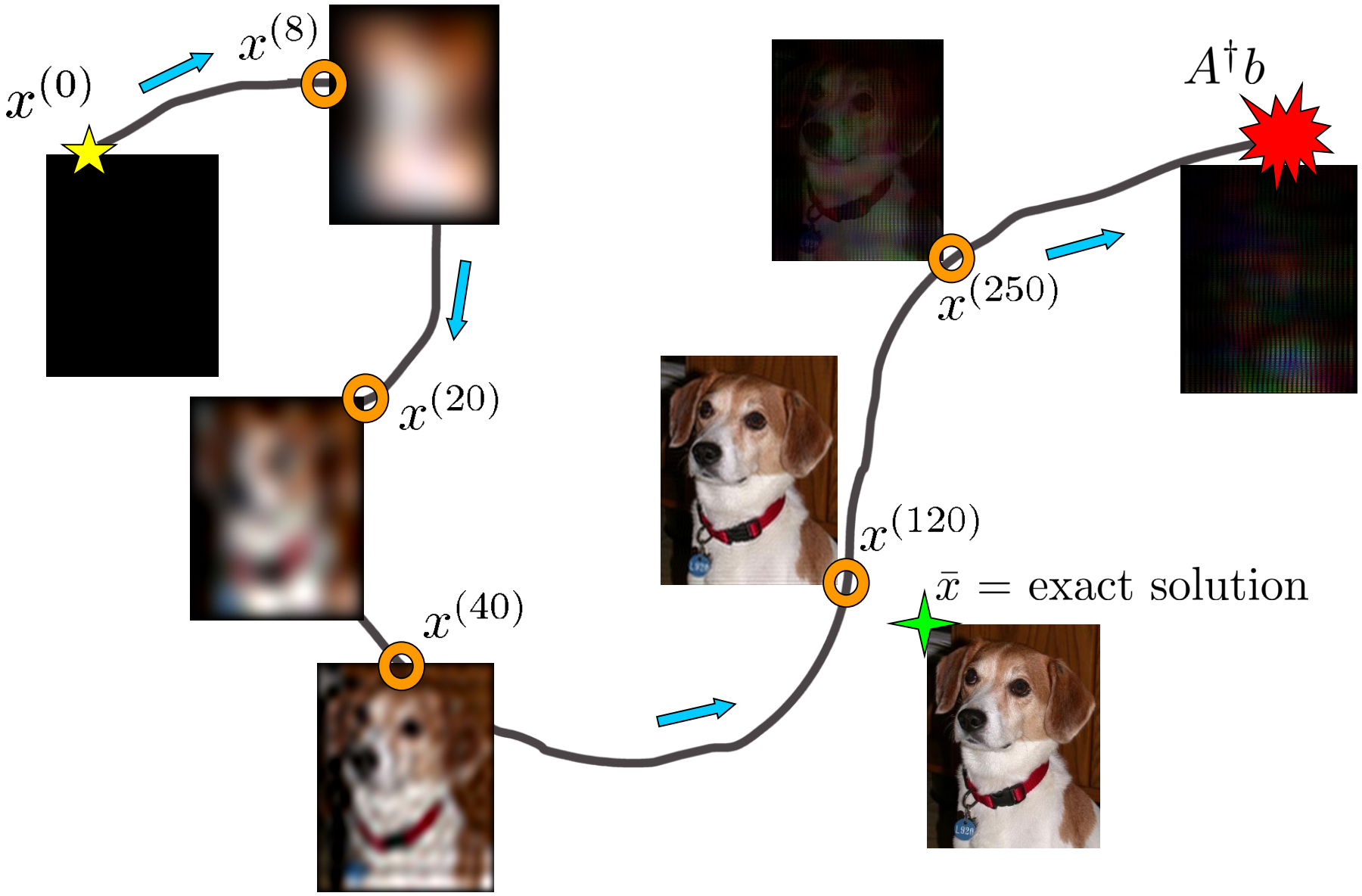
Alternatively we can use a subspace determined by the given problem, e.g., the *Krylov subspace* $\mathcal{K}_k$ associated with a specific iterative method

$$\begin{aligned} \text{CGLS/LSQR} &: \text{span}\{A^T b, (A^T A)\, A^T b, (A^T A)^2 A^T b, \ldots\}\ , \\ \text{GMRES} &: \text{span}\{b, A\, b, A^2\, b, \ldots\}\ , \\ \text{RRGMRES} &: \text{span}\{A\, b, A^2 b, A^3 b, \ldots\}\ . \end{aligned}$$

As we take more iterations – and increase the dimension of the Krylov subspace – we encounter **semi-convergence:**

- first the iterates approach the desired solution,
- later they approache the undesired "naive solution" $A^{-1}b$ or $A^\dagger b$.

Illustration of Semi-Convergence



Hybrid Methods

Recall that a Krylov subspace method, associated with the space $\mathcal{K}_k$, computes the solution

$$x^{(k)} = \text{argmin}_x \|A\,x - b\|_2^2 \qquad \text{s.t.} \qquad x \in \mathcal{K}_k \ .$$

If $\mathcal{K}_k = \text{range}(V_k)$ then

$$x^{(k)} = V_k\,z_k \ , \qquad \boxed{z_k = \text{argmin}_z \|(AV_k)\,z - b\|_2^2 \ .}$$

Projected problem

The Krylov subspace $\mathcal{K}_k$ may pick up unwanted basis vectors *before* all the desired ones have emerged. Then $\mathcal{K}_k$ is not a good signal subspace.

The solution is to add regularization to the projected problem:

$$x^{(k)} = V_k\,z_k \ , \qquad \boxed{z_k = \text{argmin}_z \left\{ \|(AV_k)\,z - b\|_2^2 + \lambda_k \|z\|_2^2 \right\} \ .}$$

Regularized projected problem

The power of this approach is that λ_k is chosen in each iteration to imposed just the right amount of regularization.

Augmented Krylov Subspace

Let $\mathcal{W}_p$ denote a linear subspace – defined by the user – that captures additional specific components of the desired solution.

Assume that $\dim(\mathcal{W}_p) = p \ll k = \text{no. its.}$

Then it can be advantageous to use an *augmented* linear subspace

$$\mathcal{S}_{p,k} = \mathcal{W}_p + \mathcal{K}_k, \qquad \mathcal{W}_p = \mathcal{R}(W_p) = \text{span}\{w_1, \ldots, w_p\} \ .$$

Thus we want an efficient iterative algorithm to solve the problem

$$\min_x \|A\,x - b\|_2^2 \quad \text{s.t.} \quad x \in \mathcal{S}_{p,j} \ .$$

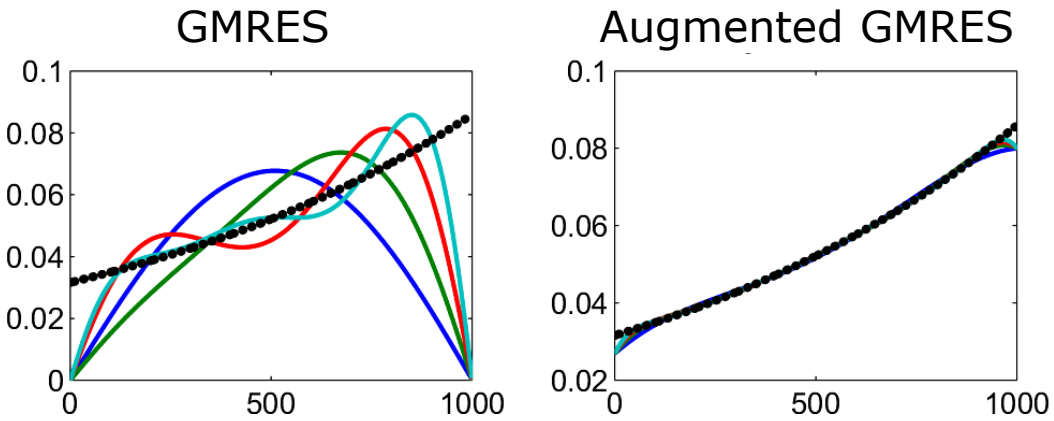
Example: Augmented GMRES

Baglama & Reichel (2007): GMRES-based methods that leave the component of x in $\mathcal{W}_p$ unchanged and build a Krylov subspace from that.

Example: `deriv2`

All vectors in the Krylov subspace $\to 0$ at end points. Now use

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



Building on this, we developed the algorithm *Regularized RRGMRES*, or R^{3}GMRES, that solves

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

Overview of Methods

Square matrix $A \in \mathbb{R}^{n \times n}$

- Augmented GRMES and RRGMRES – Baglama, Reichel (2007),
- R^{3}GMRES – Dong, Garde, H (2014).

Rectangular matrix $A \in \mathbb{R}^{m \times n}$

- In some problems (tomography, inverse heat equation) the Arnoldi subspace underlying GMRES is not suited.
- In many problems the matrix A is rectangular.
- Enriched/augmented CGNR – Calvetti, Reichel, Shuibi (2003).
- Combining enrichment with a hybrid method $\rightarrow$ our approach.

P. C. Hansen, Y. Dong, and K. Abe, *Hybrid enriched bidiagonalization for discrete ill-posed problems*, Numerical Linear Algebra Appl., 26 (2019), e2230, doi: 10.1002/nla.2230.

Enriched CGNR

We want to solve

$$\min_x \|A\,x - b\|_2^2 \quad \text{s.t.} \quad x \in \mathcal{W}_p + \mathcal{K}_k(A^TA, A^Tb) \ .$$

Standard CGNR = CGLS: $x^{(k+1)} = x^{(k)} + \alpha_k\,p_k$ where the search direction p_k is conjugate to all previous search directions.

Enriched CGNR: $x^{(k+1)} = x^{(k)} + \alpha_k\,p_k + q_k$ where q_k solves

$$\min_q \|A\,q - (b - A\,x^{(k)})\|_2 \quad \text{s.t.} \quad q \in \mathcal{W}_p \backslash \mathcal{K}_k(A^TA, A^Tb) \ .$$

Straightforward to replace $\|A\,x - b\|_2$ with the Tikhonov problem

$$\left\| \begin{bmatrix} A \\ \lambda I \end{bmatrix} x - \begin{bmatrix} b \\ 0 \end{bmatrix} \right\|_2^2$$

with a fixed λ.

Towards our Algorithm

We want an algorithm that allows a different regularization parameter λ_k in each iteration – still based on the problem

$$\min_x \|A\,x - b\|_2^2 \quad \text{s.t.} \quad x \in \mathcal{W}_p + \mathcal{K}_k(A^TA, A^Tb) \ .$$

We prefer to use a stable and efficient "standard" algorithm.

Run the *bidiagonalization* algorithm to compute an orthonormal basis of $\mathcal{K}_k(A^TA, A^Tb)$, and augment it by $\mathcal{W}_p$ in each step of the algorithm.

This seems cumbersome – but the overhead is favorably small!

Setting the Stage for Our Algorithm

At step k we have the decomposition

$$A\,[\,V_k\,,\,W_p\,] = \left[\,U_{k+1}\,,\,\widetilde{U}_k\,\right] \begin{bmatrix} B_k & G_k \\ 0 & F_k \end{bmatrix}$$

where

- $A\,V_k = U_{k+1} B_k$ is obtained after k steps of the bidiag. process.
- $V_k \in \mathbb{R}^{n\times k}$ has orthonormal columns that span $\mathcal{K}_j(A^TA, A^Tb)$.
- $U_{k+1} \in \mathbb{R}^{m\times(k+1)}$ has orthonormal columns, $u_1 = b/\|b\|_2$.
- $\widetilde{U}_k \in \mathbb{R}^{m\times p}$: $\mathcal{R}(A\,W_p) = \mathcal{R}(U_{k+1}G_k + \widetilde{U}_k F_k)$ and $\widetilde{U}_k^T U_{k+1} = 0$.
- $B_k \in \mathbb{R}^{(k+1)\times k}$ is a lower bidiagonal matrix.
- $F_k \in \mathbb{R}^{p\times p}$ and *changes in every iteration.*
- G_k is $(k+1)\times p$ and is *updated* along with B_k.

The columns of $[\,V_j\,,\,W_p\,]$ form a basis for $\mathcal{S}_{p,j}$.

More Details

Recall that

$$A\,[\,V_k\,,\,W_p\,] = \left[\,U_{k+1}\,,\,\widetilde{U}_k\,\right] \begin{bmatrix} B_k & G_k \\ 0 & F_k \end{bmatrix} .$$

The matrices $G_k \in \mathbb{R}^{(k+1)\times p}$ and $F_k \in \mathbb{R}^{p\times p}$ are composed of the coefficients of $A\,W_p$ with respect to basis of $\mathcal{R}(U_{k+1})$ and $\mathcal{R}(\widetilde{U}_k)$, respectively:

$$G_k = U_{k+1}^T A W_p, \qquad F_k = \widetilde{U}_k^T A W_p \ .$$

Then the iterate $x^{(k)} \in \mathcal{S}_{p,k}$ is given by $x^{(k)} = [\,V_k\,,\,W_p\,]\,y^{(k)}$, where

$$y^{(k)} = \operatorname{argmin}_y \left\| \begin{bmatrix} B_k & G_k \\ 0 & F_k \end{bmatrix} y - \begin{bmatrix} U_{k+1}^T \\ \widetilde{U}_j^T \end{bmatrix} b \right\|_2^2 .$$

Projected problem

Basic Enriched Bidiagonalization

1. Set $U_1 = b/\|b\|_2$, $V_0 = [\,]$, $B_0 = [\,]$, $G_0 = U_1^T A W_p$, and $k = 1$.
2. Use the bidiag. process to obtain v_k, u_{k+1} such that $A V_k = U_{k+1} B_k$, where

$$V_k = [V_{k-1}, v_k],\ U_{k+1} = [\,U_k, u_{k+1}\,],\ B_k = \begin{bmatrix} B_{k-1} & 0 \\ & \times \\ 0 & \times \end{bmatrix}.$$

3. Update $G_k = \begin{bmatrix} G_{k-1} \\ u_{k+1}^T A W_p \end{bmatrix} \in \mathbb{R}^{(k+1)\times p}$.
4. Orthonormalize AW_p with respect to U_{k+1} to obtain $\widetilde{U}_k \in \mathbb{R}^{m\times p}$.
5. Compute $F_k = \widetilde{U}_k^T A W_p \in \mathbb{R}^{p\times p}$.
6. Solve $\min_y \left\| \begin{bmatrix} B_k & G_k \\ 0 & F_k \end{bmatrix} y - \begin{bmatrix} U_{k+1}^T \\ \widetilde{U}_k^T \end{bmatrix} b \right\|_2^2$ to obtain $y^{(k)}$.
7. Then $x^{(k)} = [\,V_k\,,\, W_p\,]\, y^{(k)}$.
8. Stop, or set $k := k+1$ and return to step 2.

Recomputation of $\widetilde{U}_k$ and F_k in each step; but p is small!

Efficient and Stable Implementation

In each step we update the orthogonal factorization:

$$\begin{bmatrix} B_k & G_k \\ 0 & F_k \end{bmatrix} = Q \begin{bmatrix} T_k^{(11)} & T_k^{(12)} \\ 0 & T_k^{(22)} \\ 0 & 0 \end{bmatrix},$$

$T_k^{(11)} \in \mathbb{R}^{k \times k}$ and $T_k^{(22)} \in \mathbb{R}^{p \times p}$ are upper triangular, Q is orthogonal.

Update $T_k^{(11)}$ via Givens rotations that are also applied to G_k and $U_{k+1}^T b$.

$\widetilde{U}_k \in \mathbb{R}^{m \times p}$ is already orthogonal to U_k, hence we can perform the update

$$\widetilde{U}_{k+1} = (I_m - u_{k+1} u_{k+1}^T)\, \widetilde{U}_k.$$

For numerical stability: must reorthogonalize the columns of V_k, U_{k+1}, and $\widetilde{U}_k$. Consider the use of partial reorthogonalization.

Algorithm HYBR (Chung, Nagy, O'Leary 2008) also uses full reorthogonalization.

And Now: a Hybrid Algorithm

Recall that we want a hybrid algorithm with regularization added to the projected problem. An *equivalent* formulation:

$$x^{(k)} = \text{argmin}_x \left\{ \|A\,x - b\|_2^2 + \lambda_k^2 \, \|x\|_2^2 \right\} \qquad \text{s.t.} \quad x \in \mathcal{K}_k \ .$$

where λ_k is chosen in each iteration. This has two advantages:

1. The amount of regularization adapts to each iteration.
2. Can be used as a stopping rule, when λ_k or $x^{(k)}$ settles.

This is not possible with the enriched CGNR algorithm.

But it is possible with the enriched bidiagonalization algorithm.

HEB: Hybrid Enriched Bidiagonalization

1. Set $U_1 = b/\|b\|_2$, $V_0 = [\,]$, $B_0 = [\,]$, $G_0 = U_1^T A W_p$, and $k = 1$.
2. Use the bidiag. process to obtain v_k, u_{k+1} such that $A\,V_k = U_{k+1} B_k$, where

$$V_k = [V_{k-1}, v_k],\; U_{k+1} = [\,U_k, u_{k+1}\,],\; B_k = \begin{bmatrix} B_{k-1} & 0 \\ 0 & \times \end{bmatrix}.$$

3. Compute $G_k = \begin{bmatrix} G_{k-1} \\ u_{k+1}^T A W_p \end{bmatrix} \in \mathbb{R}^{(k+1)\times p}$.
4. Orthonormalize AW_p with respect to U_{k+1} to obtain $\widetilde{U}_k \in \mathbb{R}^{m\times p}$.
5. Compute $F_k = \widetilde{U}_k^T A W_p \in \mathbb{R}^{p\times p}$.
6. Choose λ_k and solve

Regularized projected problem

$$\min_y \left\| \begin{bmatrix} B_k & G_k \\ 0 & F_k \\ \lambda_k V_k & \lambda_k W_p \end{bmatrix} y - \begin{bmatrix} U_{k+1}^T \\ \widetilde{U}_k^T \\ 0 \end{bmatrix} b \right\|_2^2 \quad \text{for } y_{\lambda_k}^{(k)}.$$

7. Then $x^{(k)} = [\,V_k\,,\, W_p\,]\, y_{\lambda_k}^{(k)}$.
8. Stop, or set $k := k+1$ and return to step 2.

Terrible Computational Details of Step 6

Use any parameter-choice rule (GCV, discrep. principle, L-curve, ...) to choose λ_k.

Already described how to update a QR factorization of the top 2×2 block matrix.

To treat the bottom block $\lambda_k \, [\, V_k \, , \, W_p \,] \in \mathbb{R}^{n \times (k+p)}$, we multiply from the left with the orthogonal matrix $[\, V_k \, , \, V_o \,]^T$. The bottom block then takes the form

$$[\, V_k \, , \, V_o \,]^T \, \lambda_k \, [\, V_k \, , \, W_p \,] = \lambda_k \begin{bmatrix} I_k & V_k^T W_p \\ 0 & V_o^T W_p \end{bmatrix} , \qquad V_k^T W_p = \begin{bmatrix} V_{k-1}^T W_p \\ v_k^T W_p \end{bmatrix} ,$$

where $V_k = [\, V_{k-1} \, , \, v_k \,]$. Since the matrix V_o is not explicitly available, we consider the Cholesky factorization of the symmetric and positive definite $p \times p$ matrix

$$(V_o^T W_p)^T V_o^T W_p = W_p^T V_o \, V_o^T W_p = W_p^T (I_n - V_k \, V_k^T) \, W_p = R_k^T R_k \ ,$$

where $R_k \in \mathbb{R}^{p \times p}$ is the Cholesky factor. It follows immediately that

$$R_k^T R_k = W_p^T (I_n - V_{k-1} \, V_{k-1}^T) \, W_p - (W_p^T v_k) \, (W_p^T v_k)^T = R_{k-1}^T R_{k-1} - (W_p^T v_k) \, (W_p^T v_k)^T .$$

Hence we can compute R_k from R_{k-1} using techniques that *downdate* a Cholesky factor due to a rank-one change.

Numerical Examples

Setting up the test problems:

1. Generate noise-free system: $A\,x_{\text{exact}} = b_{\text{exact}}$.
2. Add noise: $b = b_{\text{exact}} + e$ where e is a random vector of Gaussian white noise scaled such that $\|e\|_2 / \|b_{\text{exact}}\|_2 = \eta$.
3. Show the following results:
   - the best solution within the iterations,
   - the relative error $\|x_{\text{exact}} - x^{(k)}\|_2 / \|x_{\text{exact}}\|_2$,
   - the residual norm $\|b - A\,x^{(k)}\|_2$.

We compare the following algorithms:

- LSQR is the implementation from Regularization Tools.
- HEB with a fixed λ (identical to Enriched CGNR).
- HEB + GCV with λ_k chosen by generalized cross validation (GCV) applied to projected problem.

Large Component in Augment. Subspace

Test problem `deriv2(n,2)`, $n = 32$, relative noise level $\eta = 10^{-6}$.

$$\mathcal{W}_2 = \text{span}\{w_1, w_2\}, \quad w_1 = (1, 1, \ldots, 1)^T, \quad w_2 = (1, 2, \ldots, n)^T.$$

For this problem

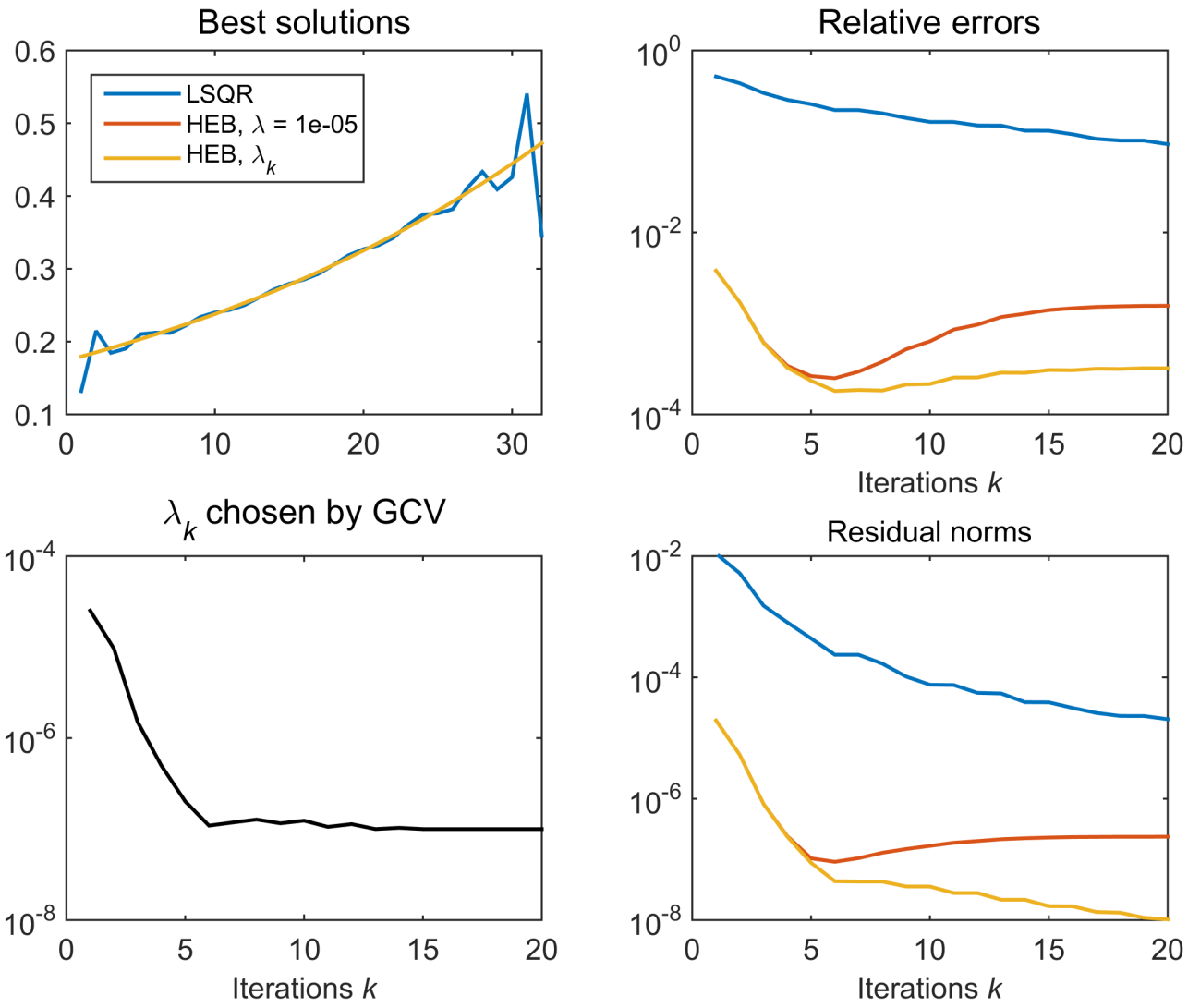
$$\|W_2 W_2^T x_{\text{exact}}\|_2 / \|x_{\text{exact}}\|_2 = 0.99 \ ,$$
$$\|(I - W_2 W_2^T) x_{\text{exact}}\|_2 / \|x_{\text{exact}}\|_2 = 0.035 \ ;$$

we only need to spend effort in capturing the small component in $\mathcal{W}_2^{\perp}$.

Results next page →

- We need augmentation to suppress oscillations towards the ends.
- HEB is sensitive to λ, and produces a good result only if we know a good value of λ.
- HEB + GCV performs very well and is able to choose a good λ_k.

Large Component in Augment. Subspace



1D Deconvolution and "Inpainting"

1. Create an $n \times n$ Toeplitz matrix A_{full} with $n = 216$, similar to the test problem `phillips` from Regularization Tools.

2. The exact solution x_{exact} has elements $\sin(1.5\pi i/n) + \cos(0.1\pi i/n)$.

3. Remove rows 71–126 of A_{full} to obtain (using MATLAB notation) $A = A_{\text{full}}([1{:}70, 127{:}216], :) \in \mathbb{R}^{160 \times 216}$.

4. Then $b_{\text{exact}} = A\, x_{\text{exact}}$ misses the middle 56 elements.

5. Use the augmentation subspace $\mathcal{W}_3 = \text{span}\{w_1, w_2, w_3\}$ with $w_1 = (1, 1, \ldots, 1)^T$, $w_2 = (1, 2, \ldots, n)^T$, $w_3 = (1, 4, \ldots, n^2)^T$.

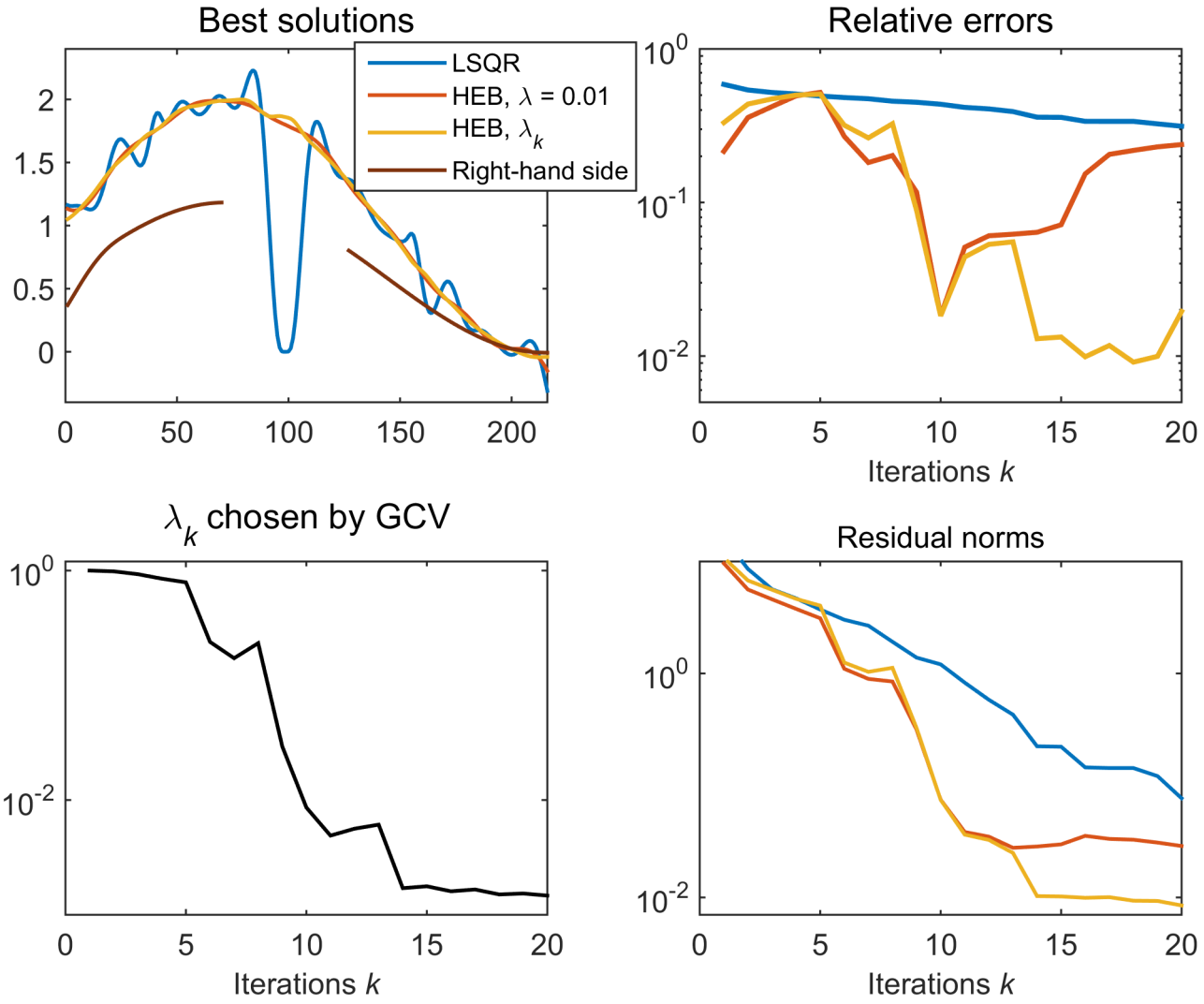
Results next page →

LSQR approximates a minimum-norm solution and therefore even the best solution has a large error in the middle.

HEB solutions are much better: the augmentation subspace W_3 provides basis vectors that "fill the gap" – but at good λ is needed.

HEB + GCV works very well.

1D Deconvolution and "Inpainting"



2D Image Deblurring and Inpainting

1. Create smooth $N \times N$ image with $N = 80$ and pixels given by $\sin(\pi\, i/(N-1))\, \sin(\pi\, j/(N-1))$.

2. Remove the rows of the blur matrix that corresponds to a 16×16 region in the middle of the image $\rightarrow$ A is $(N^2 - 256) \times N^2$.

3. Use a 4-dimensional augmentation subspace $\mathcal{W}_4$ whose basis vectors are vectorized versions of four simple arrays (using some MATLAB notation):

$$w_1 = \mathrm{vec}(\texttt{ones(N,N)}), \qquad w_2 = \mathrm{vec}(\texttt{ones(N,1)} * \texttt{(1:N)})$$

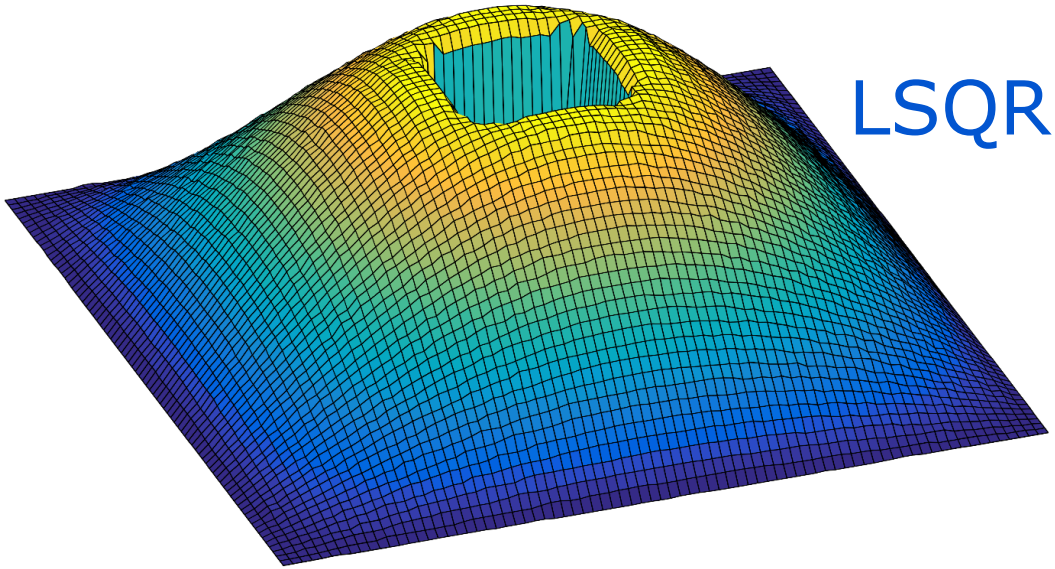
$$w_3 = \mathrm{vec}(\texttt{(1:N)}' * \texttt{ones(1:N)}), \qquad w_4 = \mathrm{vec}(\texttt{(1:N)}' * \texttt{(1:N)})\ .$$

Results next page →

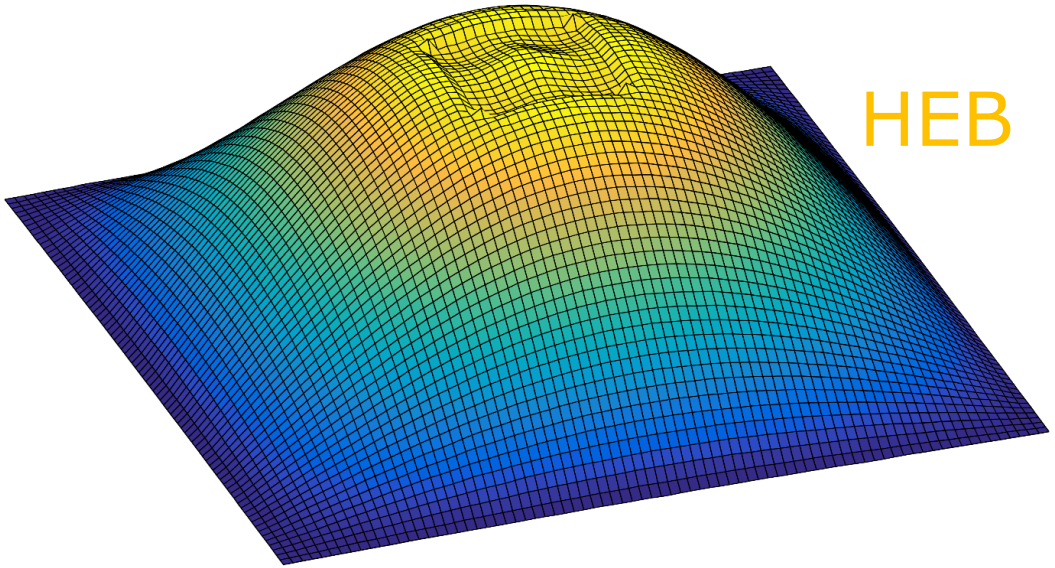
LSQR is not able to produce smooth inpainting; it leads to a smooth reconstruction but with a central region with small pixel values.

HEB inpaints the missing pixels in a smooth fashion as dictated by our augmentation subspace.

2D Image Deblurring and Inpainting



LSQR



HEB

Conclusions

- We augment the *bidiagonalization* algorithm underlying LSQR.
- Our algorithm uses an *enriched* subspace:
  the Krylov subspace plus a low-dimensional linear subspace.
- We add standard-form Tikhonov regularization, thus arriving at a *hybrid* enriched bidiagonalization algorithm.
- We choose the regularization parameter adaptively in each iteration, e.g., by means of GCV.
- Possible extension: use general-form Tikhonov regularization.