Lake Como School for Advanced Studies

Computational Methods for Inverse Problems and Applications in Image Processing

Software for Iterative Solvers

James Nagy

Emory University
Atlanta, GA USA

Review: Regularization by SVD Filtering

Suppose $\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$, and $\mathbf{b} = \mathbf{A}\mathbf{x}_{\text{true}} + \boldsymbol{\eta} = \mathbf{b}_{\text{true}} + \boldsymbol{\eta}$.

Naive inverse solution, $\mathbf{x}_{\text{inv}} = \mathbf{A}^{-1}\mathbf{b} = \mathbf{V}\boldsymbol{\Sigma}^{-1}\mathbf{U}^T\mathbf{b}$

$$\mathbf{x}_{\text{inv}} = \sum_{i=1}^{n} \frac{\mathbf{u}_i^T\mathbf{b}}{\sigma_i}\mathbf{v}_i = \underbrace{\sum_{i=1}^{n} \frac{\mathbf{u}_i^T\mathbf{b}_{\text{true}}}{\sigma_i}\mathbf{v}_i}_{\mathbf{x}_{\text{true}}} + \underbrace{\sum_{i=1}^{n} \frac{\mathbf{u}_i^T\boldsymbol{\eta}}{\sigma_i}\mathbf{v}_i}_{\text{error}}$$

The goal is to balance:

- reconstructing "good" SVD components: $\dfrac{\mathbf{u}_i^T\mathbf{b}_{\text{true}}}{\sigma_i}$ (large σ_i)
- avoid reconstructing "bad" SVD components $\dfrac{\mathbf{u}_i^T\boldsymbol{\eta}}{\sigma_i}$ (small σ_i)

SVD Filtering

$$\mathbf{x}_{\text{filt}} = \sum_{i=1}^{n} \phi_i \frac{\mathbf{u}_i^T\mathbf{b}}{\sigma_i}\mathbf{v}_i, \quad \text{where} \quad \phi_i \approx \begin{cases} 1 & \text{for "large" } \sigma_i \\ 0 & \text{for "small" } \sigma_i \end{cases}$$

Examples: TSVD, Tikhonov, Exponential

Review: Filtering and Large Scale Problems

- Computing the SVD for large problems can be very expensive.
- Exceptions for some structured matrices:
  - Kronecker products: If $\mathbf{A} = \mathbf{A}_r \otimes \mathbf{A}_c$, then

$$\begin{aligned} \mathbf{A} &= \left(\mathbf{U}_r \boldsymbol{\Sigma}_r \mathbf{V}_r^T\right) \otimes \left(\mathbf{U}_c \boldsymbol{\Sigma}_c \mathbf{V}_c^T\right) \\ &= \left(\mathbf{U}_r \otimes \mathbf{U}_c\right) \left(\boldsymbol{\Sigma}_r \otimes \boldsymbol{\Sigma}_v\right) \left(\mathbf{V}_r \otimes \mathbf{V}_c\right) \end{aligned}$$

  - Circulant matrices: Instead of using singular value decomposition, use spectral value decomposition:

$$\mathbf{A} = \mathcal{F}^* \boldsymbol{\Lambda} \mathcal{F}$$

    where $\mathcal{F}$ is the Fourier transform matrix, and $\mathcal{F}^* \mathcal{F} = \mathbf{I}$.
  - Other structures that admit efficient spectral decompositions.

Iterative Methods: Introduction

Remarks:

- If $\mathbf{A}$ is not too large, we can use SVD filtering methods.
- In case of large scale problems, filtering methods can be used if $\mathbf{A}$ has exploitable structure (e.g., Kronecker product, Circulant)
- We need to use iterative methods when $\mathbf{A}$ is very large, and
  - We cannot efficiently compute SVD of $\mathbf{A}$.
  - We want to enforce constraints, such as nonnegativity.
  - We want to incorporate (non-periodic) boundary conditions.

Iterative Methods: Introduction

- Consider the general inverse problem: $\mathbf{b} = \mathbf{A}\mathbf{x} + \mathbf{e}$
- We know that

$$\mathbf{x}_{\text{inv}} = \mathbf{A}^{-1}\mathbf{b} \quad \text{or} \quad \mathbf{x}_{\text{LS}} = \arg\min_{\mathbf{x}} \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2$$

  is a poor solution.

- Assume we cannot efficiently use FFT based, SVD, or other transform based filtering.
- However, we can efficiently compute multiplications:

$$\mathbf{A}\mathbf{z} \quad \text{and} \quad \mathbf{A}^T\mathbf{z}$$

  for example, if $\mathbf{A}$ is sparse and/or structured.

- Then we can use iterative methods.

Iterative Methods: Introduction

Some approaches to using iterative methods:

1. Apply iterative method to variational form of regularization:

   $$\min_{\mathbf{x}} \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2 + \alpha^2 \|\mathbf{L}\mathbf{x}\|_2^2$$

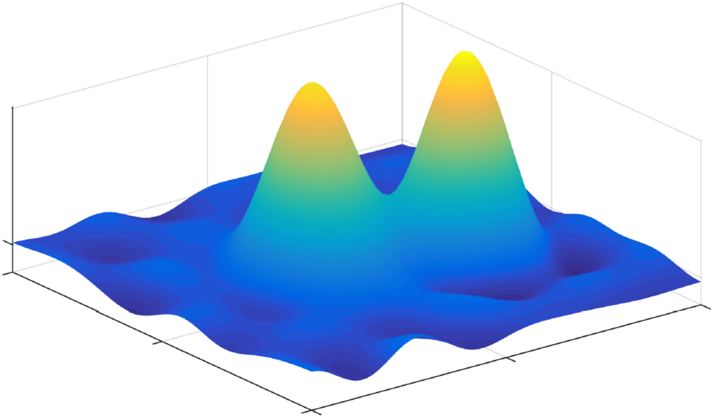
2. Apply iterative method directly to

   $$\min_{\mathbf{x}} \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2$$

   enforce regularization by stopping iteration early.

3. Combine the two approaches $\Rightarrow$ Hybrid Method

IR Tools



Silvia Gazzola University of Bath, UK S.Gazzola@bath.ac.uk	Per Christian Hansen Tech. Univ. of Denmark pcha@dtu.dk	James Nagy Emory University, USA jnagy@emory.edu

Motivation for *IR Tools*

Provide easy to use MATLAB codes for

- Methods to solve large-scale, linear ill-posed inverse problems.
- Test problems (large scale) to evaluate new algorithms.

Package can be used in many ways:

- Use our implementations to solve your problems.
- Experiment with different regularization approaches, constraints, etc.
- Use our test problems to evaluate your new algorithms.
- Compare your best/new algorithms with our implementations.

IR Tools: Test Problems

Test problem type	**Functions**
Image deblurring	`PRblur` (generic function)
– spatially invariant blur	`PRblurdefocus`, `PRdeblurgauss`, `PRdeblurmotion`, `PRdelburshake`, `PRdeblurspeckle`
– spatially variant blur	`PRblurrotation`
Inverse diffusion	`PRdiffusion`
Inverse interpolation	`PRinvinterp2`
NMR relaxometry	`PRnmr`
Tomography	
– seismic travel-time tomography	`PRseismic`
– spherical means tomography	`PRspherical`
– X-ray computed tomography	`PRtomo`
Add noise to the exact data: Gauss, Laplace, multiplicative	`PRnoise`
Visualize the data b and the solution x	`PRshowb`, `PRshowx`

Current Solvers in IRtools

Problem type	**Functions**
$\min_x \|\|Ax - b\|\|_2$ + semi-convergence	`IRart`, `IRcgls`, `IRenrich`, `IRsirt`, `IRrrgmres`
$\min_x \|\|Ax - b\|\|_2$ s.t. $x \geq 0$ + semi-convergence	`IRmrnsd`, `IRnnfcgls`
$\min_x \|\|Ax - b\|\|_2$ s.t. $x \in \mathcal{C}$ + semi-convergence	`IRconstr_ls`, `IRfista`
$\min_x \|\|Ax - b\|\|_2 + \alpha \|\|Lx\|\|_2$	`IRcgls`, `IRhybrid_lsqr`, `IRhybrid_gmres`
$\min_x \|\|Ax - b\|\|_2 + \alpha \|\|Lx\|\|_2$ s.t. $x \in \mathcal{C}$	`IRconstr_ls`, `IRfista`
$\min_x \|\|Ax - b\|\|_2 + \alpha \|\|x\|\|_1$	`IRell1`, `IRhybrid_fgmres`, `IRirn`
$\min_x \|\|Ax - b\|\|_2 + \alpha \|\|x\|\|_1$ s.t. $x \geq 0$	`IRirn`
$\min_x \|\|Ax - b\|\|_2 + \alpha \mathrm{TV}(x)$ with or without constraint $x \geq 0$	`IRhtv`

Current Solvers in IRtools

Each iterative method can be used in one of the following ways:

```
[x, IterInfo] = IRxxxx(A, b);
[x, IterInfo] = IRxxxx(A, b, K, options);
```

where

- `A` can be a sparse matrix, user defined object, or function handle.
- b is a vector.
- `K` specifies iterations to be returned in `X`.
- can change default `options` using `IRset`.

Example

First setup tomography test problem:

```
spine = double(imread('spine.tif'));
options = PRset('phantomImage', spine);
[A, b_true, x_true, ProbInfo] = PRtomo(options);
[b, NoiseInfo] = PRnoise(b_true);
```

Use `IRcgls` to attempt to solve, with all default algorithm parameters:

```
x = IRcgls(A, b);
```

Solution is over-fitted (too noisy). Investigate further using true solution:

```
options = IRcgls('defaults');
options = IRset(options, 'x_true', x_true);
[x, IterInfo] = IRcgls(A, b, options);
```

Plot `IterInfo.Enrm` to see semi-convergence behavior, and display solution where error is minimized:

```
PRshowx(IterInfo.BestReg.X, ProbInfo)
```

LSQR Hybrid Method

Based on Golub-Kahan (Lanczos) Bidiagonalization (GKBD):

Given $\mathbf{A}$ and $\mathbf{b}$, for $k = 1, 2, \ldots$, compute

- $\mathbf{W}_k = \begin{bmatrix} \mathbf{w}_1 & \mathbf{w}_2 & \cdots & \mathbf{w}_k & \mathbf{w}_{k+1} \end{bmatrix}, \quad \mathbf{w}_1 = \mathbf{b}/||\mathbf{b}||$
- $\mathbf{Z}_k = \begin{bmatrix} \mathbf{z}_1 & \mathbf{z}_2 & \cdots & \mathbf{z}_k \end{bmatrix}$
- $\mathbf{B}_k = \begin{bmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_k & \alpha_k \\ & & & \beta_{k+1} \end{bmatrix}$

where $\mathbf{W}_k$ and $\mathbf{Z}_k$ have orthonormal columns, and

$$\begin{aligned} \mathbf{A}^T\mathbf{W}_k &= \mathbf{Z}_k\mathbf{B}_k^T + \alpha_{k+1}\mathbf{z}_{k+1}\mathbf{e}_{k+1}^T \\ \mathbf{A}\mathbf{Z}_k &= \mathbf{W}_k\mathbf{B}_k \end{aligned}$$

GKBD and LSQR

At kth GKBD iteration, use QR to solve *projected* LS problem:

$$\min_{\mathbf{x}\in R(\mathbf{Z}_k)} \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2^2 = \min_{\mathbf{x}} \|\mathbf{W}_k^T\mathbf{b} - \mathbf{B}_k\mathbf{x}\|_2^2 = \min_{\mathbf{x}} \|\beta\mathbf{e}_1 - \mathbf{B}_k\mathbf{x}\|_2^2$$

where $\mathbf{x}_k = \mathbf{Z}_k\mathbf{x}$

For our ill-posed inverse problems:

- Singular values of $\mathbf{B}_k$ converge to k largest sing. values of $\mathbf{A}$.
- Thus, $\mathbf{x}_k$ is in a subspace that approximates a subspace spanned by the large singular components of $\mathbf{A}$.
  - For $k < n$, $\mathbf{x}_k$ is a regularized solution.
  - $\mathbf{x}_n = \mathbf{x}_{\text{inv}} = \mathbf{A}^{-1}\mathbf{b}$ (bad approximation)

Example: Inverse Heat Equation

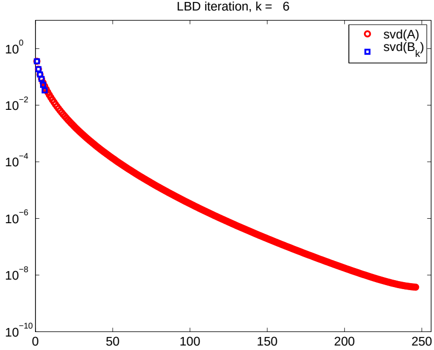
Singular values of $\mathbf{B}_k$ converge to large singular values of $\mathbf{A}$.
Thus, for early iterations k: $\mathbf{x} = \mathbf{B}_k \setminus \mathbf{W}_k \mathbf{b}$

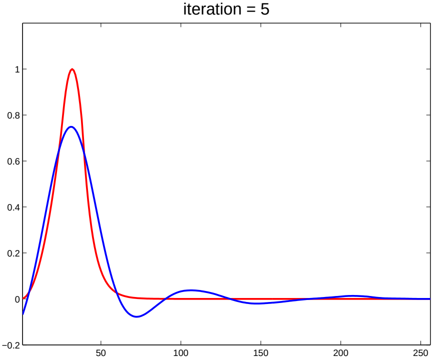
$$\mathbf{x}_k = \mathbf{Z}_k \mathbf{x}$$

is a regularized reconstruction. Thus, for later iterations k:
$\mathbf{x} = \mathbf{B}_k \setminus \mathbf{W}_k \mathbf{b}$

$$\mathbf{x}_k = \mathbf{Z}_k \mathbf{x}$$

is a noisy reconstruction.





Example: Inverse Heat Equation

Singular values of $\mathbf{B}_k$ converge to large singular values of $\mathbf{A}$.

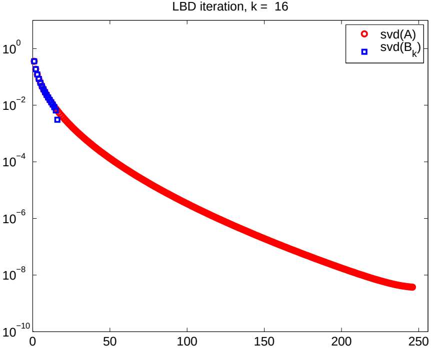
Thus, for early iterations k:

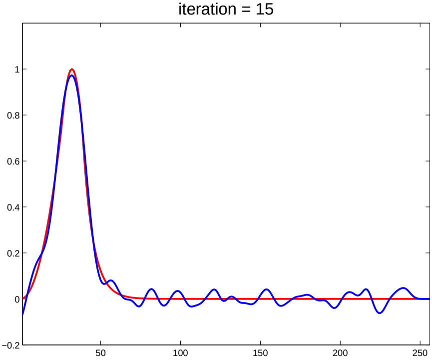
$$\mathbf{x} = \mathbf{B}_k \setminus \mathbf{W}_k \mathbf{b}$$
$$\mathbf{x}_k = \mathbf{Z}_k \mathbf{x}$$

is a regularized reconstruction. Thus, for later iterations k:

$$\mathbf{x} = \mathbf{B}_k \setminus \mathbf{W}_k \mathbf{b}$$
$$\mathbf{x}_k = \mathbf{Z}_k \mathbf{x}$$

is a noisy reconstruction.





Example: Inverse Heat Equation

Singular values of $\mathbf{B}_k$ converge to large singular values of $\mathbf{A}$.

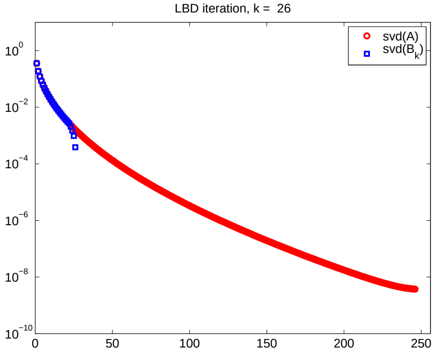
Thus, for early iterations k:

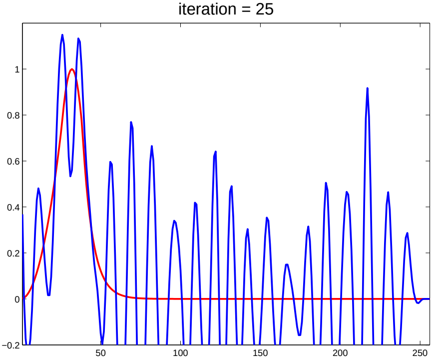
$$\mathbf{x} = \mathbf{B}_k \backslash \mathbf{W}_k \mathbf{b}$$
$$\mathbf{x}_k = \mathbf{Z}_k \mathbf{x}$$

is a regularized reconstruction. Thus, for later iterations k:

$$\mathbf{x} = \mathbf{B}_k \backslash \mathbf{W}_k \mathbf{b}$$
$$\mathbf{x}_k = \mathbf{Z}_k \mathbf{x}$$

is a noisy reconstruction.





Example: Inverse Heat Equation

Singular values of $\mathbf{B}_k$ converge to large singular values of $\mathbf{A}$.

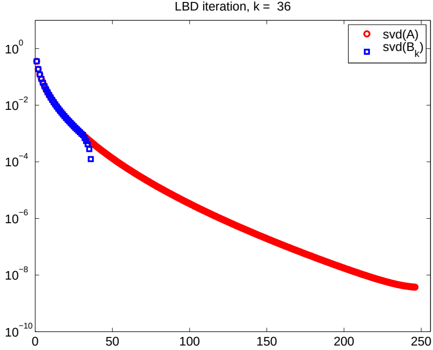
Thus, for early iterations k:

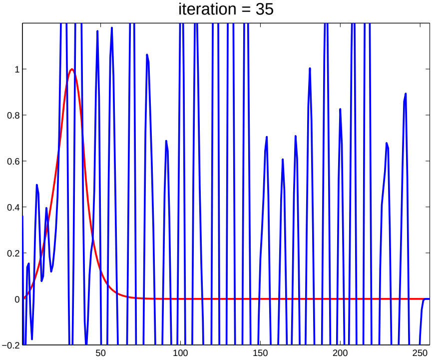
$$\mathbf{x} = \mathbf{B}_k \setminus \mathbf{W}_k \mathbf{b}$$
$$\mathbf{x}_k = \mathbf{Z}_k \mathbf{x}$$

is a regularized reconstruction. Thus, for later iterations k:

$$\mathbf{x} = \mathbf{B}_k \setminus \mathbf{W}_k \mathbf{b}$$
$$\mathbf{x}_k = \mathbf{Z}_k \mathbf{x}$$

is a noisy reconstruction.





Golub-Kahan Based Hybrid Methods

To avoid noisy reconstructions, embed regularization in LBD:

- O'Leary and Simmons, SISSC, 1981.
- Björck, BIT 1988.
- Björck, Grimme, and Van Dooren, BIT, 1994.
- Larsen, PhD Thesis, 1998.
- Hanke, BIT 2001.
- Kilmer and O'Leary, SIMAX, 2001.
- Kilmer, Hansen, Español, SISC 2007.
- Chung, N, O'Leary, ETNA 2007 (`IRhybrid_lsqr` Implementation)

Note: There is also a lot of work on Arnoldi-Tikhonov hybrid methods; see work by Reichel (and collaborators), and Gazzola and Novati.

Regularize the Projected Least Squares Problem

To stabilize convergence, regularize the projected problem:

$$\min_{\mathbf{x}} \left\| \begin{bmatrix} \beta \mathbf{e}_1 \\ \mathbf{0} \end{bmatrix} - \begin{bmatrix} \mathbf{B}_k \\ \lambda \mathbf{I} \end{bmatrix} \mathbf{x} \right\|_2^2$$

Note: $\mathbf{B}_k$ is very small compared to $\mathbf{A}$, so

- Can use "expensive" methods to choose λ (e.g., GCV)
- Can also use GCV information to estimate stopping iteration (Björck, Grimme, and Van Dooren, BIT, 1994).

Example: Inverse Heat Equation

LSQR (no regularization)

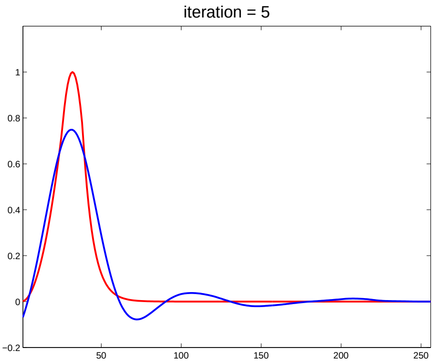
$$\mathbf{x} = \mathbf{B}_k \backslash \mathbf{W}_k \mathbf{b}$$

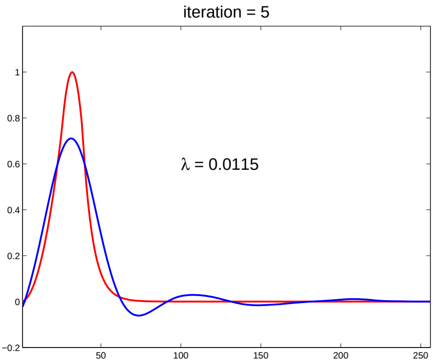
$$\mathbf{x}_k = \mathbf{Z}_k \mathbf{x}$$

HyBR (Tikhonov regularization)

$$\mathbf{x} = \begin{bmatrix} \mathbf{B}_k \\ \lambda_k \mathbf{I} \end{bmatrix} \Big\backslash \begin{bmatrix} \mathbf{W}_k \mathbf{b} \\ \mathbf{0} \end{bmatrix}$$

$$\mathbf{x}_k = \mathbf{Z}_k \mathbf{x}$$





Example: Inverse Heat Equation

LSQR (no regularization)

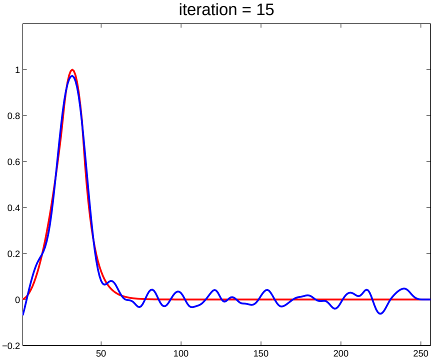
$$\mathbf{x} = \mathbf{B}_k \setminus \mathbf{W}_k \mathbf{b}$$

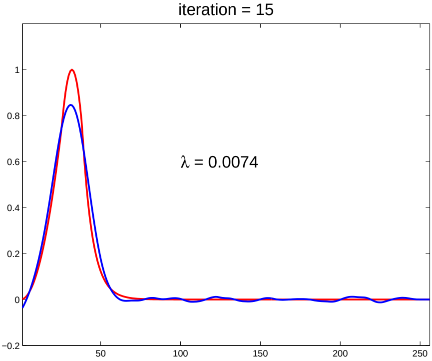
$$\mathbf{x}_k = \mathbf{Z}_k \mathbf{x}$$

HyBR (Tikhonov regularization)

$$\mathbf{x} = \begin{bmatrix} \mathbf{B}_k \\ \lambda_k \mathbf{I} \end{bmatrix} \setminus \begin{bmatrix} \mathbf{W}_k \mathbf{b} \\ \mathbf{0} \end{bmatrix}$$

$$\mathbf{x}_k = \mathbf{Z}_k \mathbf{x}$$





Example: Inverse Heat Equation

LSQR (no regularization)

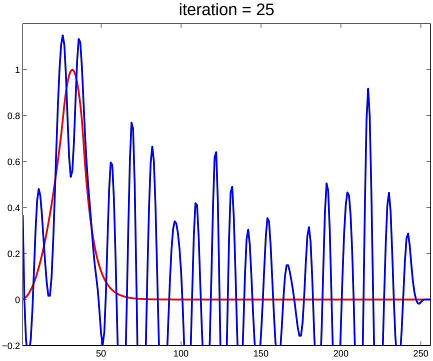
$$\mathbf{x} = \mathbf{B}_k \backslash \mathbf{W}_k \mathbf{b}$$

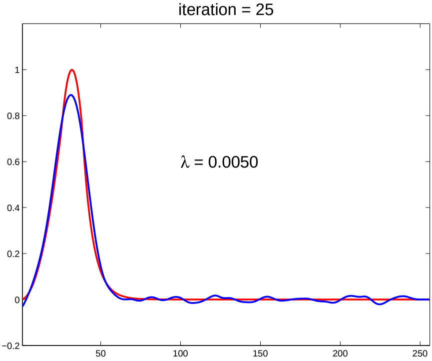
$$\mathbf{x}_k = \mathbf{Z}_k \mathbf{x}$$

HyBR (Tikhonov regularization)

$$\mathbf{x} = \begin{bmatrix} \mathbf{B}_k \\ \lambda_k \mathbf{I} \end{bmatrix} \Big\backslash \begin{bmatrix} \mathbf{W}_k \mathbf{b} \\ \mathbf{0} \end{bmatrix}$$

$$\mathbf{x}_k = \mathbf{Z}_k \mathbf{x}$$





Example: Inverse Heat Equation

LSQR (no regularization)

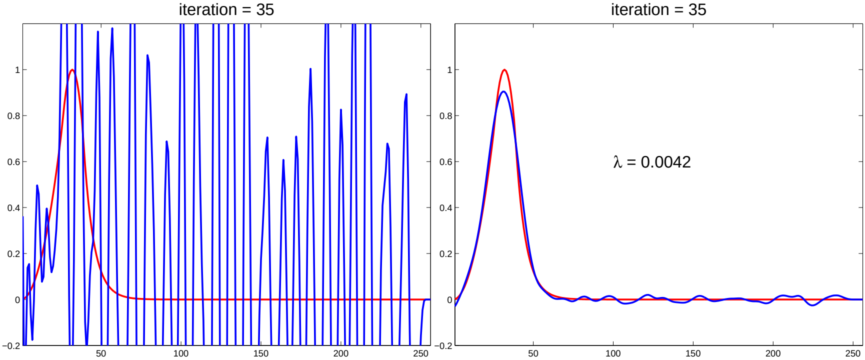
$$\mathbf{x} = \mathbf{B}_k \backslash \mathbf{W}_k \mathbf{b}$$

$$\mathbf{x}_k = \mathbf{Z}_k \mathbf{x}$$

HyBR (Tikhonov regularization)

$$\mathbf{x} = \begin{bmatrix} \mathbf{B}_k \\ \lambda_k \mathbf{I} \end{bmatrix} \Big\backslash \begin{bmatrix} \mathbf{W}_k \mathbf{b} \\ \mathbf{0} \end{bmatrix}$$

$$\mathbf{x}_k = \mathbf{Z}_k \mathbf{x}$$



Regularize the Projected Least Squares Problem

To stabilize convergence, regularize the projected problem:

$$\min_{\mathbf{x}} \left\| \begin{bmatrix} \beta \mathbf{e}_1 \\ \mathbf{0} \end{bmatrix} - \begin{bmatrix} \mathbf{B}_k \\ \lambda \mathbf{I} \end{bmatrix} \mathbf{x} \right\|_2^2$$

Problems choosing regularization parameters:

- Very little regularization is needed in early iterations.
- GCV tends to choose too large λ for bidiagonal system. Our remedy: Use a weighted GCV (Chung, N, O'Leary, 2007)
- Can also use WGCV information to estimate stopping iteration (approach similar to Björck, Grimme, and Van Dooren, BIT, 1994).

Weighted GCV

If GCV tends to over or under smooth for class of problems, use:

$$GCV(\lambda) = \frac{n\|(I - AA_\lambda^\dagger)\mathbf{b}\|^2}{\left[\text{trace}(I - \omega AA_\lambda^\dagger)\right]^2}$$

- $\omega = 1 \Rightarrow$ standard GCV
- $\omega > 1 \Rightarrow$ smoother solutions
- $\omega < 1 \Rightarrow$ less smooth solutions

Weighted GCV used in:

Friedman, Silverman (Technometrics, 1989)

Nychka, et al. (FUNFITS statistical toolbox, 1998)

Cummins, Filloon, Nychka (J. Am. Stat. Assoc., 2001)

Kim, Gu (Royal Stat. Soc. B, 2004)

Interpretations of Modified GCV

- Weighted "leave-one-out" prediction method.

- $\text{trace}\Big(I - \omega A A_\lambda^\dagger\Big) = \sum_{i=1}^{n}(1-\phi_i) + (1-\omega)\sum_{i=1}^{n}\phi_i,$

  where $\phi_i = \dfrac{\sigma_i^2}{\sigma_i^2 + \lambda^2}$ (Tikhonov SVD filter factors)

- If $\omega > 1$, modified GCV function has poles when $\displaystyle\sum_{i=1}^{n}\phi_i = \frac{n}{\omega}$

How to choose ω?

- GCV chooses too large λ_k at each iteration.
- If we know $\lambda_{k,opt}$, find ω by solving

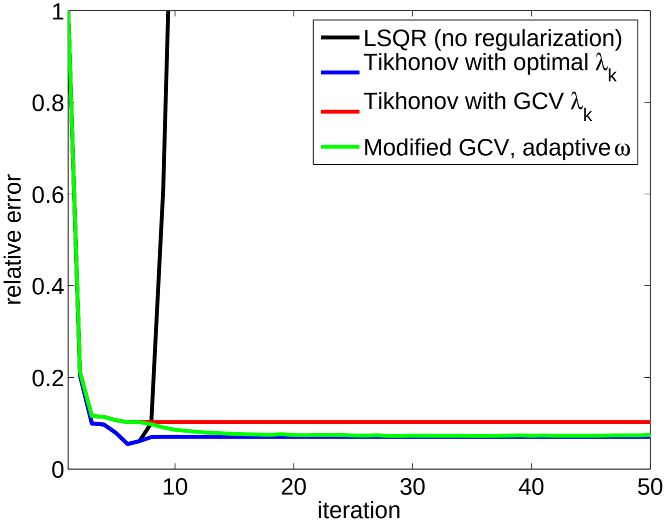
$$\frac{\partial}{\partial \lambda} \left[G(\omega, \lambda) \right] \Big|_{\lambda = \lambda_{k,opt}} = 0$$

- At early iterations, we need little or no regularization, so

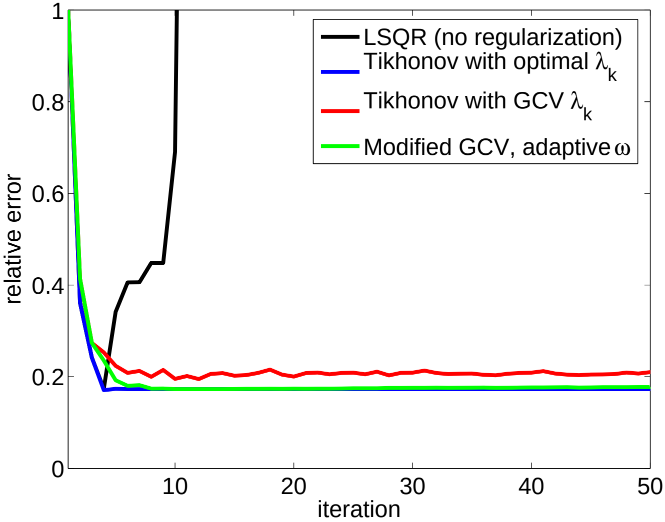
$$0 \le \lambda_{k,opt} \le \sigma_{min} \left(\mathbf{B}_k \right)$$

- Adaptive approach:
  - Find $\hat{\omega}_k$ corresponding to $\lambda_{k,opt} = \sigma_{min} \left(\mathbf{B}_k \right)$
  - Use ω_k =mean$\{\hat{\omega}_1, \hat{\omega}_2, \ldots, \hat{\omega}_k\}$

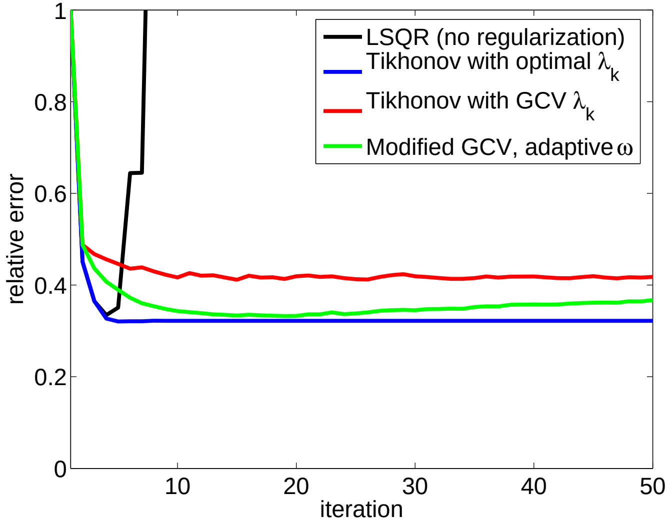
Examples: Regularization Tools, phillips



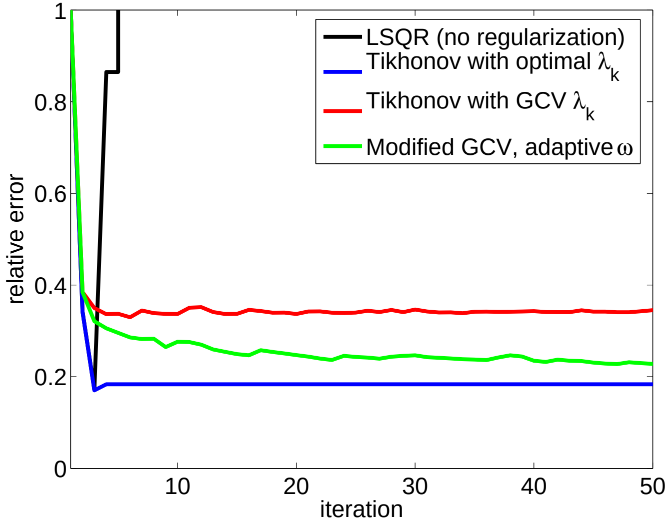
Examples: Regularization Tools, `shaw`



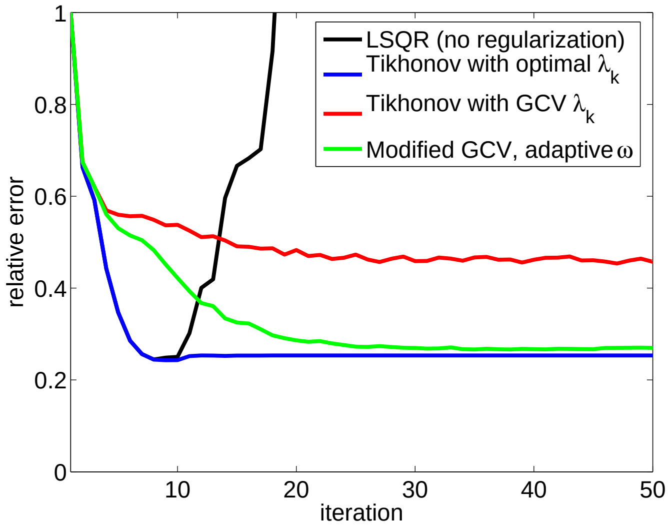
Examples: Regularization Tools, `deriv2`



Examples: Regularization Tools, baart



Examples: Regularization Tools, heat



Example

- Recall our previous example with the tomography test problem.
- We used `IRcgls`, which didn't enforce any regularization, and wasn't able to determine a good stopping iteration without any additional information.
- Now try a hybrid method, which will use Tikhonov regularization, and GCV to stop iterations, as described on the previous slides:

```
[x, IterInfo] = IRhybrid_lsqr(A, b);
PRshowx(x, ProbInfo)
```

Exercise: See if you can reconstruct problems created by someone else:

- Go to `https://www.fips.fi/dataset.php`
- Scroll to nearly bottom of page, and grab mat files:

```
DataFull_128x15.mat
DataFull_128x45.mat
DataFull_128x180.mat
```

- Load one of the data sets, say:

```
load DataFull_128x15.mat
```

- For convenience manually set `ProbInfo` structure:

```
ProbInfo.problemType = 'tomography';
ProbInfo.xType = 'image2D';
ProbInfo.xType = 'image2D';
ProbInfo.xSize = [128, 128];
ProbInfo.bSize = size(m);
```

- Reshape sinogram data `m` as a vector `b`:

```
b = m(:);
```

- Solve using some of the `IRxxxx` methods. For example,

```
[x, IterInfo] = IRhybrid_lsqr(A, b);
```

- Display using `PRshowx`:

```
PRshowx(x, ProbInfo)
```