Preconditioning for Saddle Point Problems and Image Processing Applications

Michael K. Ng
Department of Mathematics
Hong Kong Baptist University

Regularization, regularized least squares

- Consider an error contaminated ill-conditioned linear system of equations

$$A\mathbf{x} \approx \mathbf{b}$$

- Tikhonov's regularization

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

L – regularization operator

λ – regularization parameter

Assumptions behind regularized least squares solution

- The vector b is related to the unknown parameter vector x by a linear relation: $Ax = b + n$
- The vector n consists of white Gaussian noise
- The unknown vector x satisfies a Gaussian prior distribution.

In reality

- The prior distribution of the unknown vector rarely satisfies the Gaussian assumption, very often the additive noise does not satisfy the Gaussian assumption either.
  - Least Mixed Norm (LMN) solution, Least Absolute Deviation (LAD) solution

Least mixed norm, least absolute deviation

- Least squares

$$\min_{\mathbf{f}} \|\mathbf{g} - H\mathbf{f}\|_2^2 + \alpha \|R\mathbf{f}\|_2^2.$$

- Least mixed norm, nonnegative least mixed norm

$$\min_{\mathbf{f}} \|\mathbf{g} - H\mathbf{f}\|_2^2 + \alpha \|R\mathbf{f}\|_1$$

- Least absolute deviation, nonnegative least absolute deviation

$$\min_{\mathbf{f}} \|\mathbf{g} - H\mathbf{f}\|_1 + \alpha \|R\mathbf{f}\|_1$$

Formulating the Nonnegative LAD problem as a linear programming problem

- Linear programming:

$$\mathbf{u} = H\mathbf{f} - \mathbf{g} \qquad \mathbf{v} = \alpha R\mathbf{f}$$

$$\mathbf{u}^+ = \max(\mathbf{u}, 0) \qquad \mathbf{u}^- = \max(-\mathbf{u}, 0)$$

$$\mathbf{v}^+ = \max(\mathbf{v}, 0) \qquad \mathbf{v}^- = \max(-\mathbf{v}, 0)$$

Then the nonnegative LAD problem can be stated as:

$$\min_{\mathbf{f},\mathbf{u}^+,\mathbf{u}^-,\mathbf{v}^+,\mathbf{v}^-} \mathbf{1}^T\mathbf{u}^+ + \mathbf{1}^T\mathbf{u}^- + \mathbf{1}^T\mathbf{v}^+ + \mathbf{1}^T\mathbf{v}^-$$

subject to

$$H\mathbf{f} - \mathbf{g} = \mathbf{u}^+ - \mathbf{u}^-$$

$$\alpha R\mathbf{f} = \mathbf{v}^+ - \mathbf{v}^-$$

$$\mathbf{u}^+,\ \mathbf{u}^-,\ \mathbf{v}^+,\ \mathbf{v}^-,\ \mathbf{f} \geq 0$$

The linear programming problem restated

- Le $A = \begin{bmatrix} H & -I & I & 0 & 0 \\ \alpha R & 0 & 0 & -I & I \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \mathbf{g} \\ 0 \end{bmatrix},$

$$\mathbf{x} = \begin{bmatrix} \mathbf{f} \\ \mathbf{u}^+ \\ \mathbf{u}^- \\ \mathbf{v}^+ \\ \mathbf{v}^- \end{bmatrix} \quad \text{and} \quad \mathbf{c} = \begin{bmatrix} 0 \\ \mathbf{1} \\ \mathbf{1} \\ \mathbf{1} \\ \mathbf{1} \end{bmatrix}.$$

Then the linear programming problem can be written as

$$\min_{\mathbf{x}} \mathbf{c}^T \mathbf{x} \quad \text{subject to} \quad A\mathbf{x} = \mathbf{b}, \quad \mathbf{x} \geq 0$$

Lagrangian function, optimality conditions

- The Lagrangian function:

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}, \mathbf{s}) = \mathbf{c}^T\mathbf{x} - \boldsymbol{\lambda}^T(A\mathbf{x} - \mathbf{b}) - \mathbf{s}^T\mathbf{x}$$

- The optimality conditions:

$$F(\mathbf{x}, \boldsymbol{\lambda}, \mathbf{s}) = \begin{bmatrix} A^T\boldsymbol{\lambda} + \mathbf{s} - \mathbf{c} \\ A\mathbf{x} - \mathbf{b} \\ XS\mathbf{1} \end{bmatrix} = 0 \qquad \mathbf{x} \geq 0, \quad \mathbf{s} \geq 0$$

$$S = \operatorname{diag}(\mathbf{s}) \qquad X = \operatorname{diag}(\mathbf{x})$$

Interior point methods

- Interior point methods apply variants of Newton's method for nonlinear system of equations.
- The basic Newton step is modified such that the search directions are aimed at points on the central

$$F(\mathbf{x}_\tau, \boldsymbol{\lambda}_\tau, \mathbf{s}_\tau) = \begin{bmatrix} 0 \\ 0 \\ \tau \mathbf{1} \end{bmatrix}, \quad \mathbf{x}_\tau > 0, \quad \mathbf{s}_\tau > 0$$

$$\tau = \sigma \mu$$

$\sigma \in [0, 1]$ is a centering parameter,

$\mu = \frac{1}{n} \sum_{i=1}^{n} x_i s_i = \frac{\mathbf{x}^T \mathbf{s}}{n}$ is the duality measure.

The Newton step

- The Newton step is computed by solving the linear system:

$$\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \triangle \mathbf{x} \\ \triangle \boldsymbol{\lambda} \\ \triangle \mathbf{s} \end{bmatrix} = \begin{bmatrix} -\mathbf{r}_c \\ -\mathbf{r}_b \\ -\mathbf{r}_a \end{bmatrix}$$

where $\mathbf{r}_a = XS\mathbf{1} - \sigma\mu\mathbf{1}$ $\mathbf{r}_b = A\mathbf{x} - \mathbf{b}$ $\mathbf{r}_c = A^T\boldsymbol{\lambda} + \mathbf{s} - \mathbf{c}$

- The linear system can be reduced to the following form:

$$\left[D_1^{-2} + H^T(D_2^2 + D_3^2)^{-1}H + \alpha^2 R^T(D_4^2 + D_5^2)^{-1}R\right] \triangle \mathbf{f} = \tilde{\mathbf{r}}_{c1}$$

Saddle Point Systems

By eliminating Δs

$$\begin{bmatrix} -X^{-1}S & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \end{bmatrix} = \begin{bmatrix} -\hat{r}_c \\ -r_b \end{bmatrix}, \tag{2.10}$$

where $\hat{r}_c = r_c - X^{-1} r_a$. Let $D = S^{-1/2} X^{1/2}$; then (2.10) can be written as

$$\begin{bmatrix} -D^{-2} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \end{bmatrix} = \begin{bmatrix} -\hat{r}_c \\ -r_b \end{bmatrix}.$$

Formulating the nonnegative LMN problem as a quadratic programming problem

- Quadratic programming:

  Let $\mathbf{v} = \alpha R\mathbf{f}$

  $$\mathbf{v}^+ = \max(\mathbf{v}, 0)$$
  $$\mathbf{v}^- = \max(-\mathbf{v}, 0)$$

  Then the nonnegative LMN problem can be stated as:

  $$\min_{\mathbf{f},\mathbf{v}^+,\mathbf{v}^-} \mathbf{1}^T\mathbf{v}^+ + \mathbf{1}^T\mathbf{v}^- + \|H\mathbf{f} - \mathbf{g}\|_2^2$$

  subject to

  $$\alpha R\mathbf{f} = \mathbf{v}^+ - \mathbf{v}^-,$$
  $$\mathbf{v}^+, \mathbf{v}^-, \mathbf{f} \geq 0.$$

The quadratic programming problem restated

- Let

$$G = \begin{bmatrix} 2H^T H & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad A = \begin{bmatrix} \alpha R & -I & I \end{bmatrix},$$

$$\mathbf{b} = \mathbf{0}, \quad \mathbf{x} = \begin{bmatrix} \mathbf{f} \\ \mathbf{v}^+ \\ \mathbf{v}^- \end{bmatrix} \quad \text{and} \quad \mathbf{c} = \begin{bmatrix} -2H^T\mathbf{g} \\ \mathbf{1} \\ \mathbf{1} \end{bmatrix}.$$

Then the quadratic programming problem can be written as

$$\min_{\mathbf{x}} \frac{1}{2}\mathbf{x}^T G\mathbf{x} + \mathbf{c}^T\mathbf{x} \quad \text{subject to} \quad A\mathbf{x} = \mathbf{b}, \quad \mathbf{x} \geq 0$$

Lagrangian function, optimality conditions

- The Lagrangian function:

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}, \mathbf{s}) = \frac{1}{2}\mathbf{x}^T G \mathbf{x} + \mathbf{c}^T \mathbf{x} - \boldsymbol{\lambda}^T (A\mathbf{x} - \mathbf{b}) - \mathbf{s}^T \mathbf{x},$$

- The optimality conditions:

$$F(\mathbf{x}, \boldsymbol{\lambda}, \mathbf{s}) = \begin{bmatrix} G\mathbf{x} + \mathbf{c} - A^T \boldsymbol{\lambda} - \mathbf{s} \\ A\mathbf{x} - \mathbf{b} \\ XS\mathbf{1} \end{bmatrix} = 0 \qquad \mathbf{x} \geq 0, \quad \mathbf{s} \geq 0$$

$$S = \operatorname{diag}(\mathbf{s}) \qquad X = \operatorname{diag}(\mathbf{x})$$

Interior point method for quadratic programming

- Interior point methods can also be used to solve quadratic programming problems.
- The Newton step is computed by solving the linear system:

$$\begin{bmatrix} G & -A^T & -I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta \mathbf{x} \\ \Delta \boldsymbol{\lambda} \\ \Delta \mathbf{s} \end{bmatrix} = \begin{bmatrix} -\mathbf{r}_c \\ -\mathbf{r}_b \\ -\mathbf{r}_a \end{bmatrix}$$

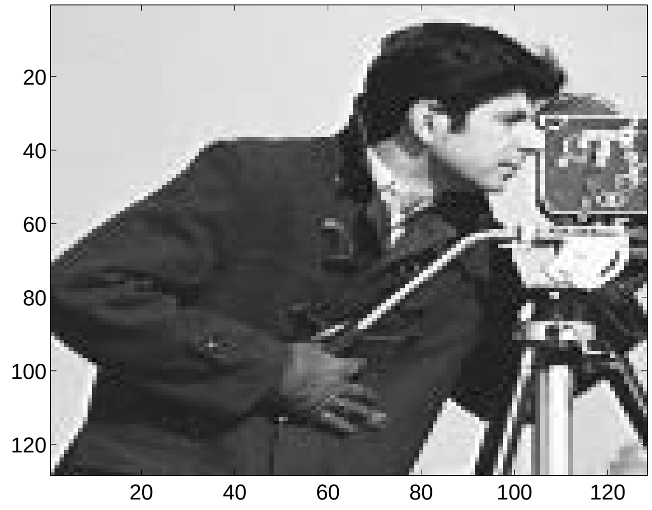
- The linear system can be reduced to the following form:

$$\left[2H^T H + D_1^{-2} + \alpha^2 R^T (D_2^2 + D_3^2)^{-1} R\right] \Delta \mathbf{f} = -\tilde{\mathbf{r}}_{c1}$$

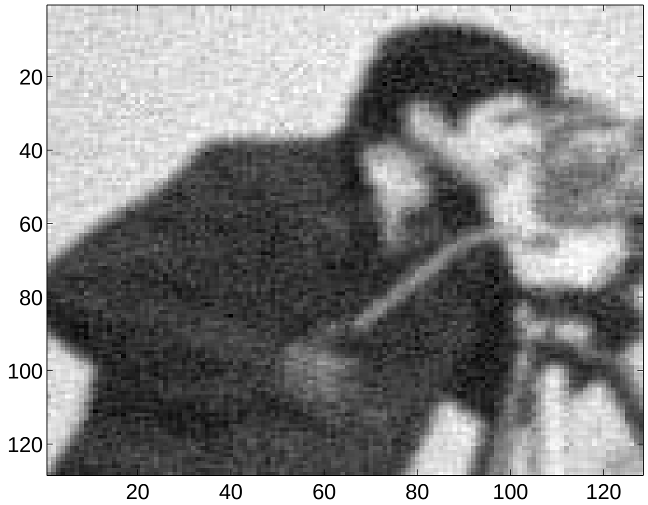
Preconditioning the inner systems

- The inner systems are symmetric positive definite, and can be solved by Conjugate Gradient type method.
- The inner systems get ill-conditioned as the iterates get close to the solution, preconditioners are needed to accelerate convergence.
- Factorized Sparse Inverse Preconditioners (FSIP)
  - Let A be a SPD matrix, let $A=C^TC$ be its Cholesky factorizatio $\|I - CL\|_F$ P is a lower triangular matrix L with certain sparsity pattern such that is minimized.
- Factorized Banded Inverse Preconditioners (FBIP)
  - The preconditioner is banded

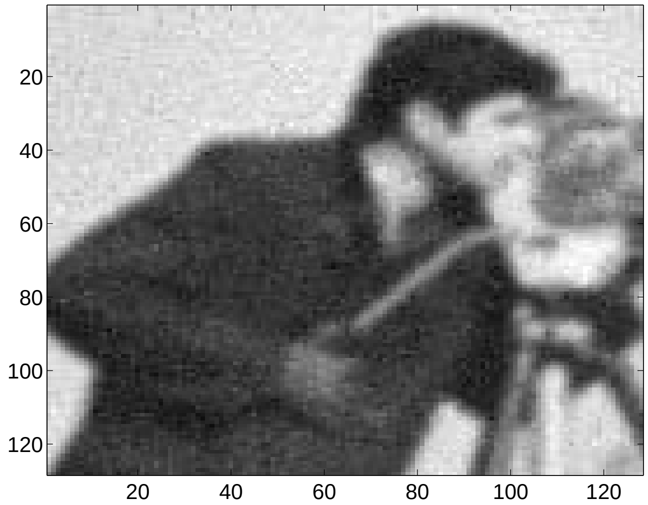
A computational example – the original image



A computational example – the observed images

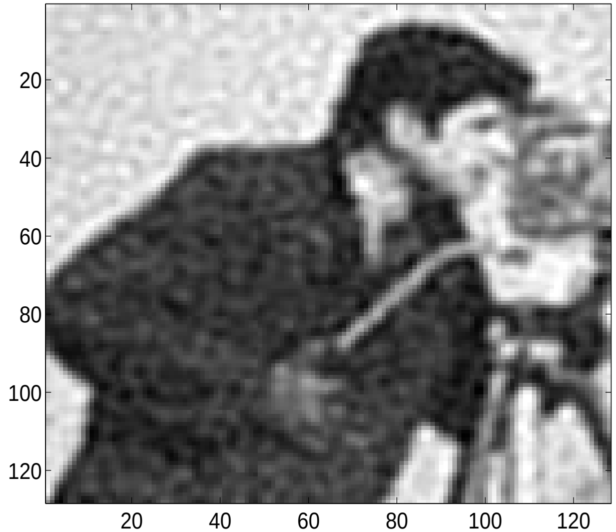


Observed image with white Gaussian noise

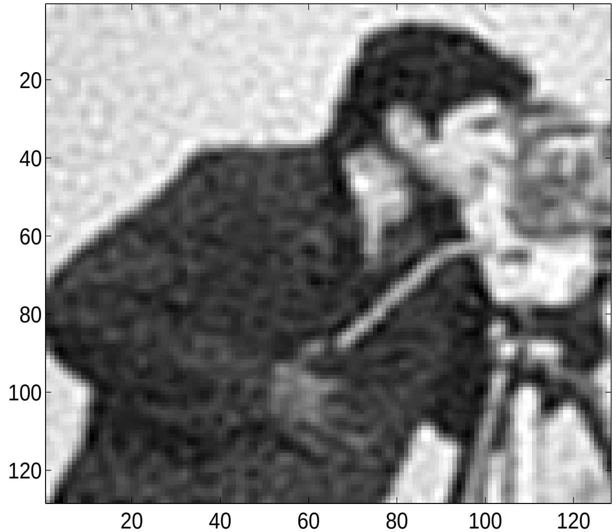


Observed image with 50% of the pixels contaminated by white Gaussian noise

A computational example – the least squares restorations

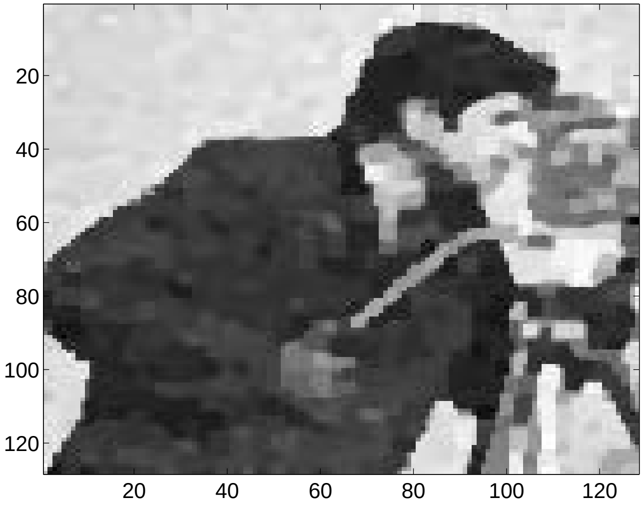


Least squares restoration of the image that is contaminated by white Gaussian noise
PSNR = 20.46

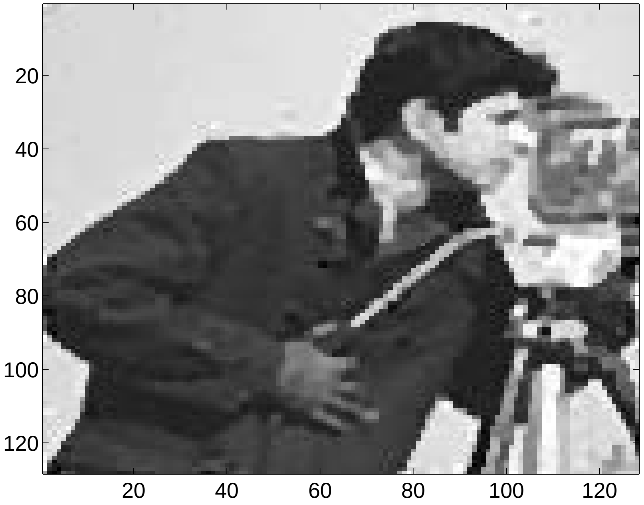


Least squares restoration of the image that only 50% of the pixels are contaminated by noise
PSNR = 20.87

A computational example – the LMN and LAD solutions

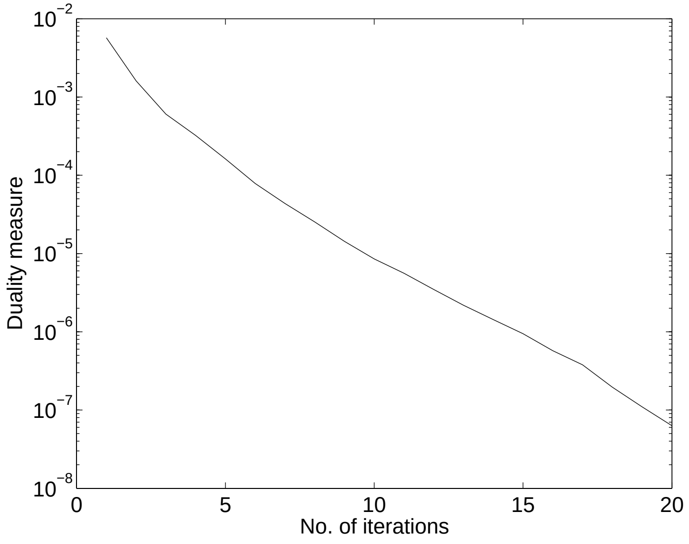


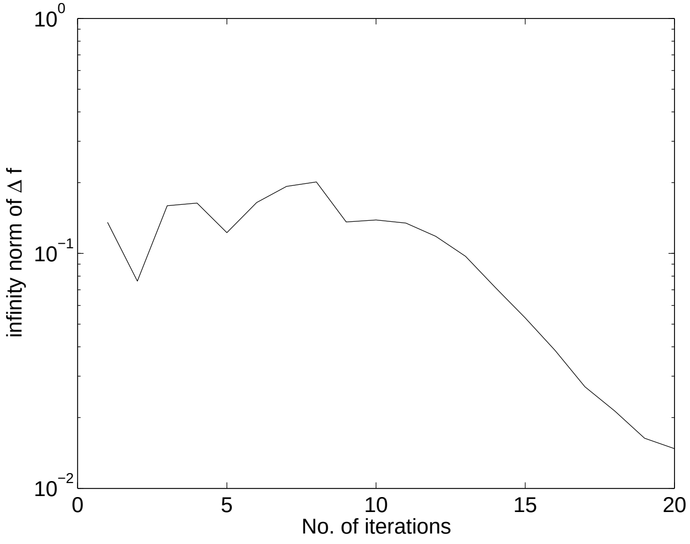
LMN solution of the image that is contaminated by white Gaussian noise
PSNR = 20.78



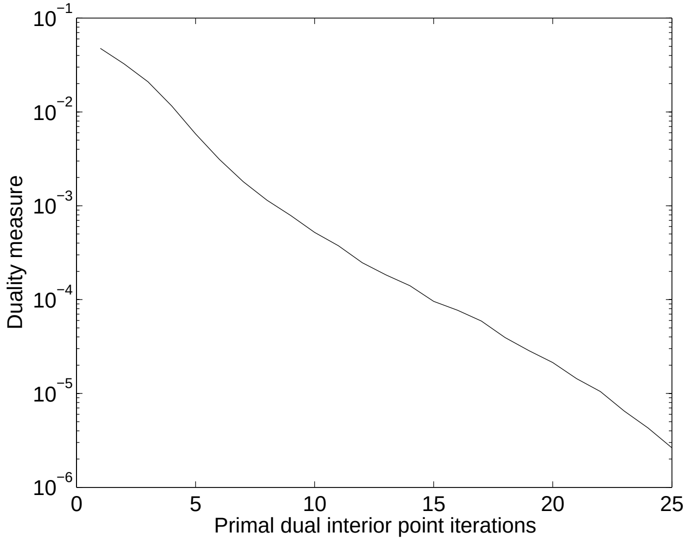
LAD solution of the image that only 50% of the pixels are contaminated by noise
PSNR = 22.82

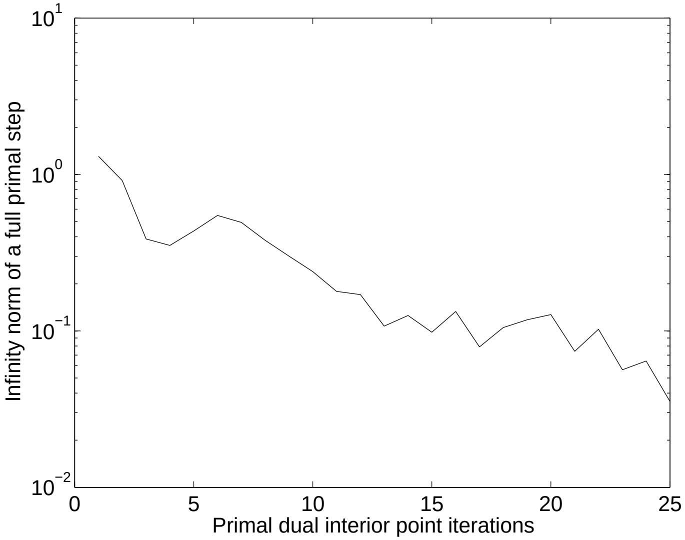
A computational example – convergence of the interior point method for LMN solution





A computational example – convergence of the interior point method for LAD solution





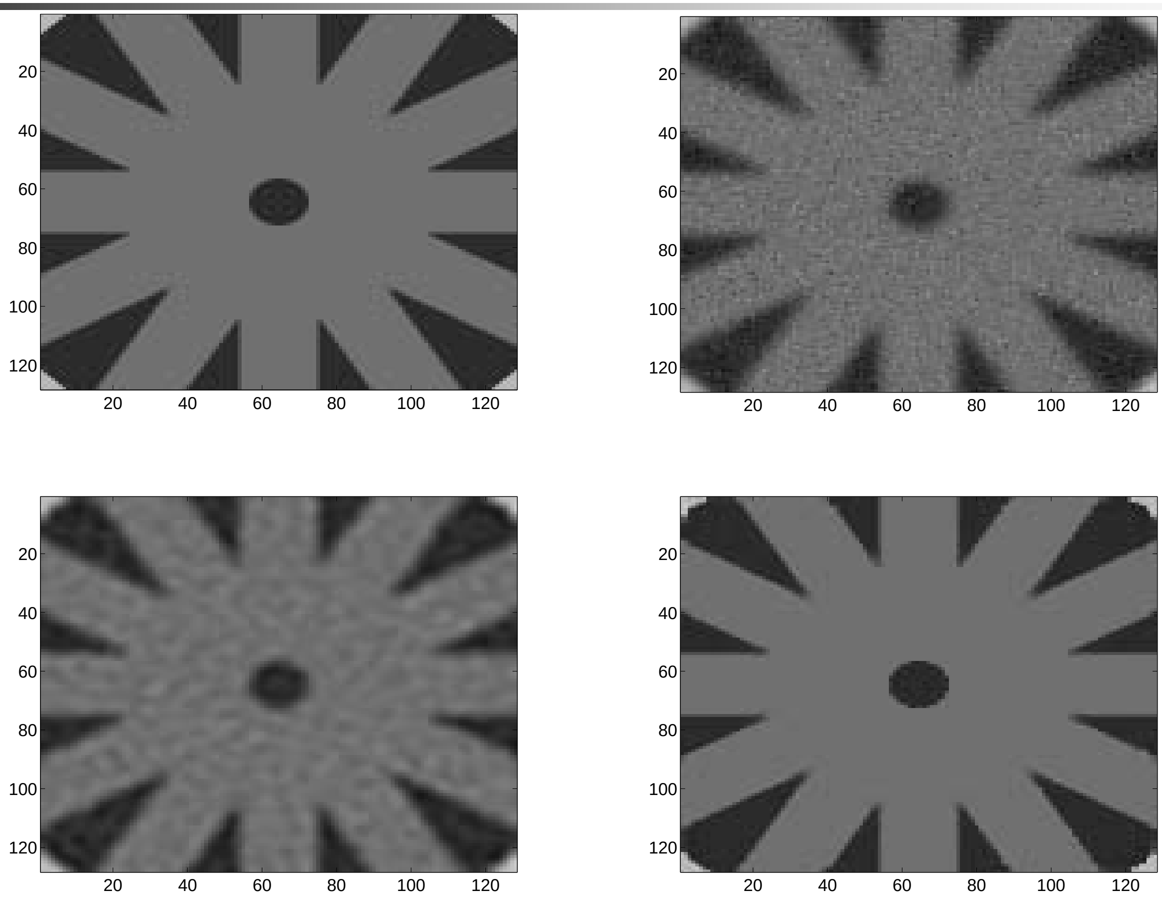
A computational example – effectiveness of the FBIP

PD Itn	No Pre	Diag Pre	FSIP2	FSIP3	FSIP4
10	368 45.01	180 22.94	31 **6.15**	21 6.89	15 10.52
12	618 76.27	281 35.27	51 9.71	32 **9.24**	22 12.72
14	908 112.37	438 55.28	85 16.12	48 **13.35**	32 15.86
16	1391 171.36	626 78.27	139 25.95	71 **18.31**	49 20.84
18	2202 265.25	989 119.56	225 40.32	123 29.99	67 **26.55**
20	>3000 -	1814 222.30	408 72.91	228 54.15	120 **41.62**

Computational Results

PD Itn	No Pre	Diag Pre	FSIP2	FSIP3	FSIP4
10	839 79.10	669 64.95	57 11.40	24 **9.39**	14 11.76
12	1406 130.74	1140 108.45	96 17.16	38 **11.94**	21 13.40
14	2199 208.98	1672 164.74	129 22.31	52 **14.61**	28 15.54
16	>3000 –	2296 222.31	175 29.55	73 19.45	39 **18.64**
18	>3000 –	>3000 –	266 43.23	106 26.57	54 **22.80**
20	>3000 –	>3000 –	378 59.50	142 32.89	74 **28.19**

Another computational example



Results

PD Itn	NOR	BENZI	No Pre	Diag Pre	FSIP2	FSIP3	FSIP4
10	>3000 –	295 161.78	644 78.68	544 68.72	48 15.24	22 **13.32**	18 17.08
12	>3000 –	483 237.70	982 125.92	804 105.53	51 15.75	26 **14.14**	21 18.01
14	>3000 –	1047 589.68	1603 207.05	1334 175.88	60 17.42	31 **15.31**	25 18.99
16	>3000 –	1791 969.84	2452 305.14	1934 242.51	77 20.90	43 **18.47**	34 22.14
18	>3000 –	>2000 –	>3000 –	2607 339.06	106 25.85	54 **20.87**	41 25.11
20	>3000 –	>2000 –	>3000 –	>3000 –	170 37.71	85 **28.18**	58 28.91

Application: Sparse Fisher Discriminant Method

selection in microarray data. In the new algorithm, we calculate a weight for each gene and use the weight values as an indicator to identify the subsets of relevant genes that categorize patient and normal samples in two-class classification problems. This is achieved by including the weight sparsity term in the Fisher objective function that is minimized in the discriminant process:

$$\|S_w u - z\|_2^2 + \alpha \|u\|_1.$$

Here S_w is the within-class scatter matrix of the samples in a microarray data and z comes from the between-class scatter matrix, u is the projection vector and α is the regularization parameter to control the sparsity of u. Each entry of u represents a weight for each gene. An efficient l_2-l_1 norm minimization method is implemented to the above discriminant model to automatically compute the weights of all genes in the samples

Nonconvex and Nonsmooth Regularization

$$J(\mathbf{f}) = \Theta(H\mathbf{f} - \mathbf{g}) + \beta\Phi(\mathbf{f}), \quad \mathbf{f} \geq 0, \qquad \Phi(\mathbf{f}) = \sum_{i=1}^{r} \varphi(\mathbf{d}_i^T \mathbf{f})$$

Convex PFs	
(f1)	$\varphi(t) = \|t\|$
Non-convex PFs	
(f2)	$\varphi(t) = \|t\|^{\alpha}, \ 0 < \alpha < 1$
(f3)	$\varphi(t) = \dfrac{\alpha\|t\|}{1 + \alpha\|t\|}$
(f4)	$\varphi(t) = \log\left(\alpha\|t\| + 1\right)$
(f5)	$\varphi(0)=0, \ \varphi(t)=1 \text{ if } t \neq 0$

TABLE 1.1

Non-smooth at zero PFs φ where $\alpha > 0$ is a parameter. Some references are [7, 55, 27, 28, 56, 39]

Property

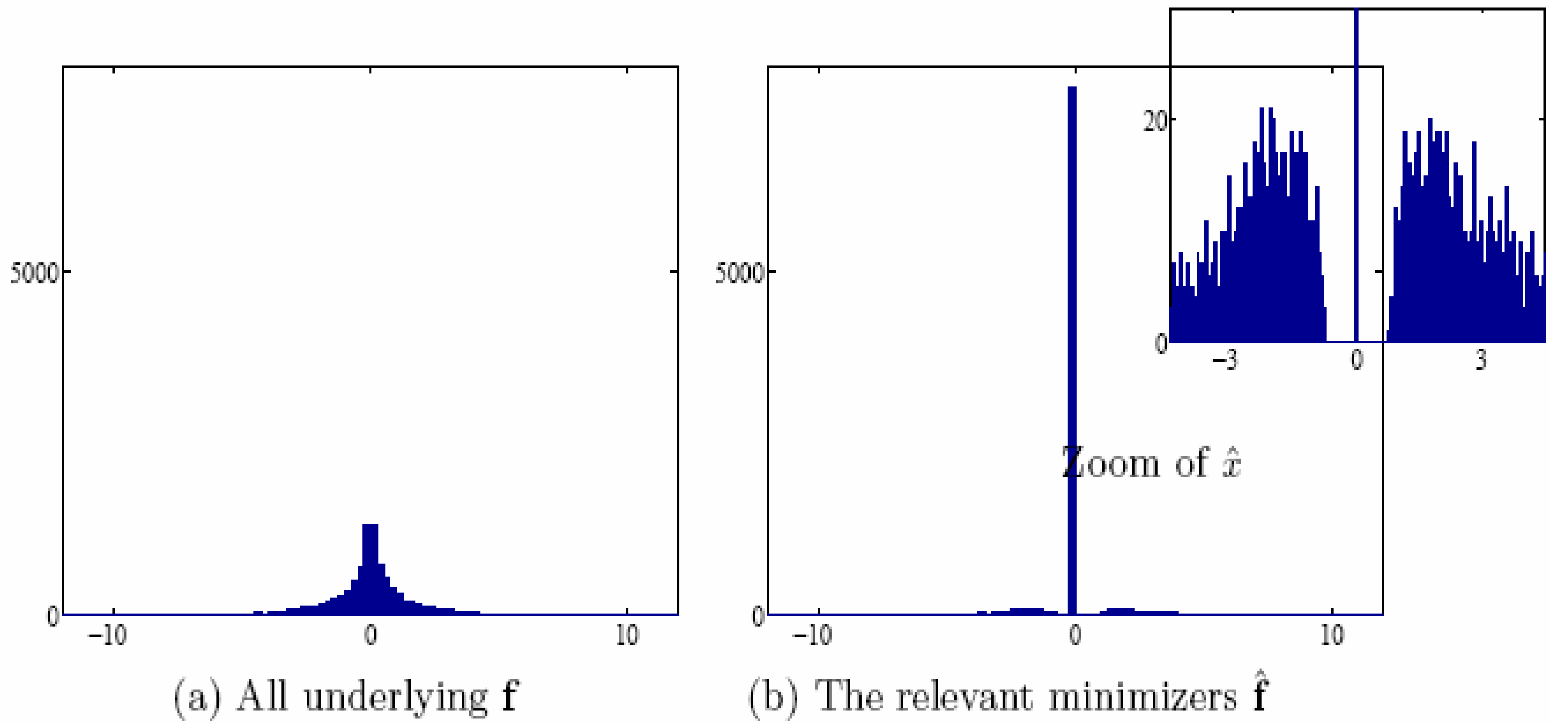
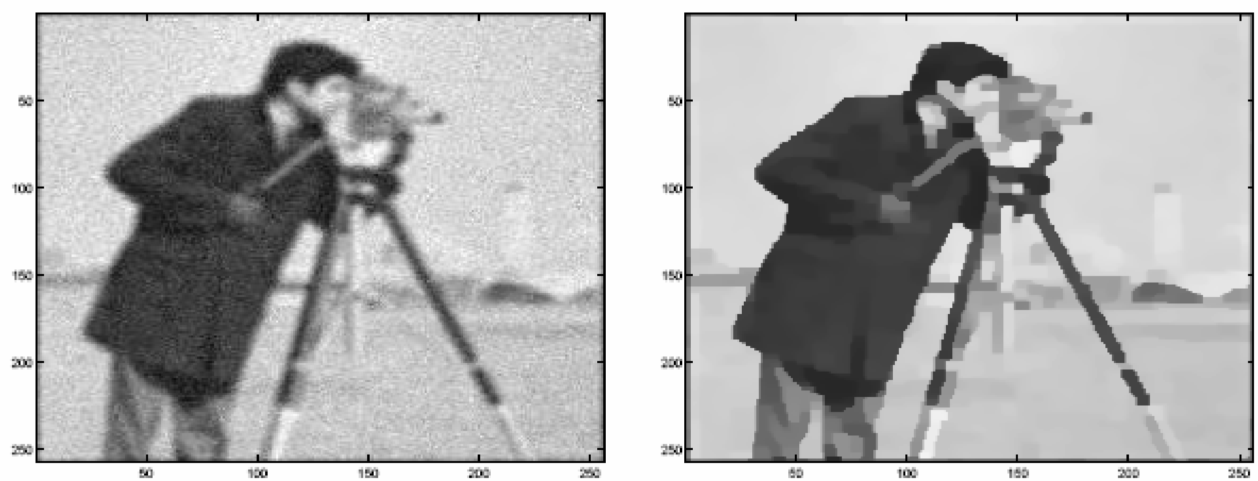


FIG. 1.1. *Histograms for 10 000 independent trials:* $\mathbf{f}$ *and* $\hat{\mathbf{f}}$ *for* $\varphi(t) = \sqrt{|t|}$.

FIG. 6.15. *The original cameraman image*



(a) (b)

FIG. 6.16. *(a) Observed image; (b) The restored image.*

Continuation Method

The Newton search direction $(\Delta\mathbf{y}, \Delta\boldsymbol{\lambda}, \Delta\mathbf{s})$ is computed by solving the system:

$$\begin{bmatrix} M + \nabla^2\Psi_\varepsilon(\mathbf{y}) & -B^T & -I \\ B & 0 & 0 \\ S & 0 & Y \end{bmatrix} \begin{bmatrix} \Delta\mathbf{y} \\ \Delta\boldsymbol{\lambda} \\ \Delta\mathbf{s} \end{bmatrix} = \begin{bmatrix} -\mathbf{r}_c \\ -\mathbf{r}_b \\ -\mathbf{r}_a \end{bmatrix}$$

where

$$\mathbf{r}_c = M\mathbf{y} + c + \nabla\Psi_\varepsilon(\mathbf{y}) - B^T\boldsymbol{\lambda} - \mathbf{s},$$

$$\mathbf{r}_b = B\mathbf{y}, \quad \mathbf{r}_a = YS\mathbf{1} - \sigma\mu\mathbf{1}.$$

By eliminating Δs from the above equation, we obtain

$$\begin{bmatrix} M + \nabla^2\Psi_\varepsilon + Y^{-1}S & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} \Delta\mathbf{y} \\ -\Delta\lambda \end{bmatrix} = \begin{bmatrix} -\hat{\mathbf{r}}_c \\ -\mathbf{r}_b \end{bmatrix}$$

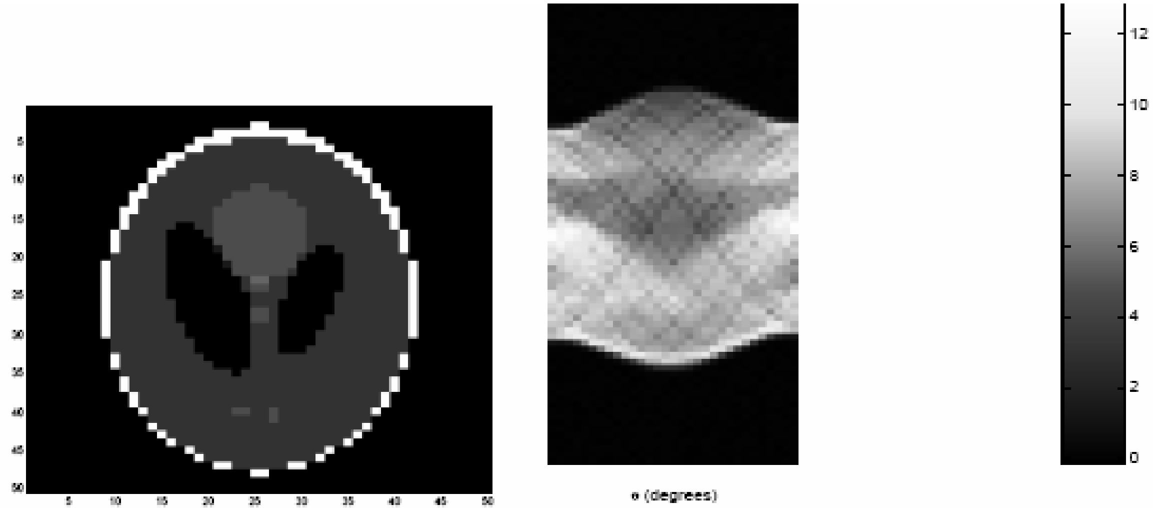
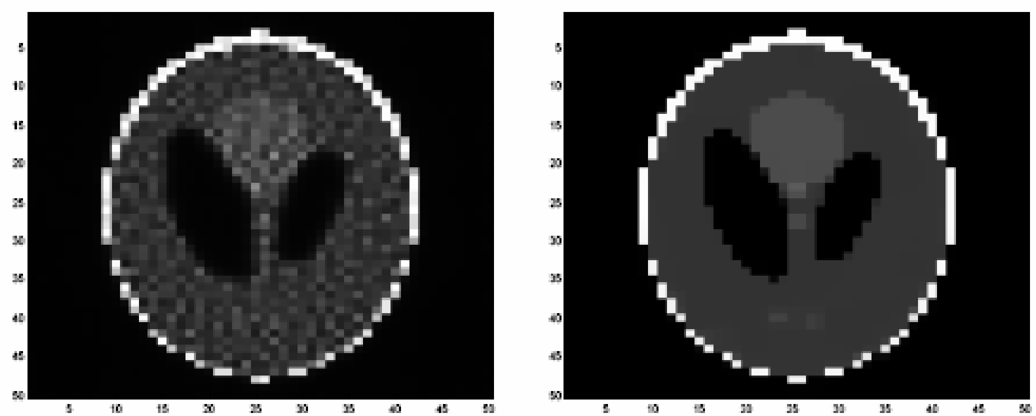
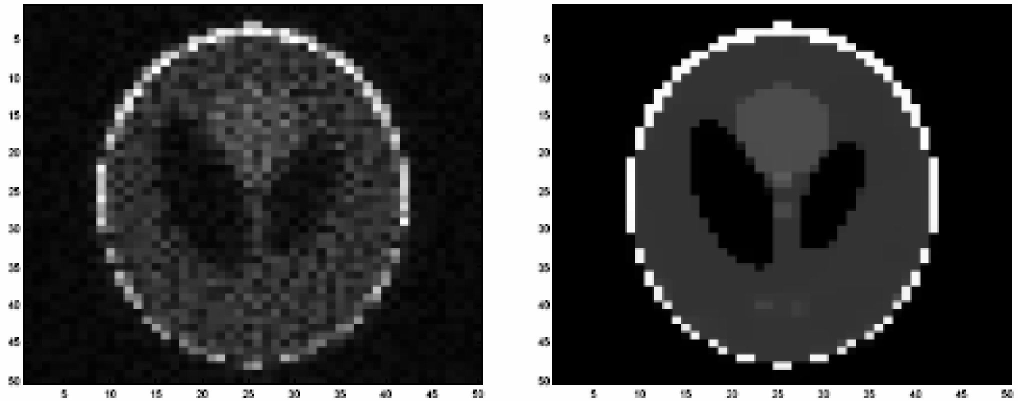


FIG. 6.9. *(a) The original modified Shepp-Logan image with size* 50×50*; (b) the obtained image after radon transform along the angles from 0 to 180 with the increasing of 6. The noise follows the normal distribution with mean zero and deviation 0.05*



(a) (b)

FIG. 6.10. *Restored Shepp-Logan images for PF (6.1) when the initial guess is a flat image:* $0.5 \times ones(50, 50)$ *(a) the restored image when* $\varepsilon = 1$*, PSNR = 25.82; (b) the restored image when* $\varepsilon = 0 \rightarrow 1$*, PSNR = 41.96.*



(a) (b)

FIG. 6.11. *Restored Shepp-Logan images for PF (6.1) when the initial guess is a random image (a) the restored image when $\varepsilon = 1$, PSNR = 20.23. (b) the restored image when $\varepsilon = 0 \longrightarrow 1$, PSNR = 41.96.*

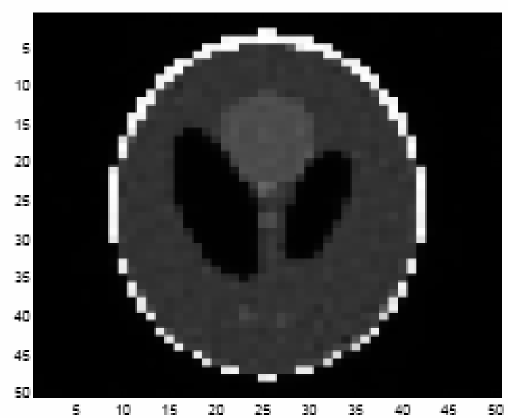


FIG. 6.12. *The restored Shepp-Logan image with PF $\psi(t) = |t|$. The initial guess is a flat image, PSNR = 34.71.*

Concluding Remarks

- Saddle Point System Solvers
- Optimization Problems
- Existing preconditioners does not perform quite well
- Future research area