Metodi Matematici nel trattamento delle immagini

Roma, 15-16 gennaio 2013

Spatially-Adaptive Methods for Image Deblurring and Denoising

Alessandro Lanza, Serena Morigi, Fiorella Sgallari

University of Bologna, Dept. Math, Italy

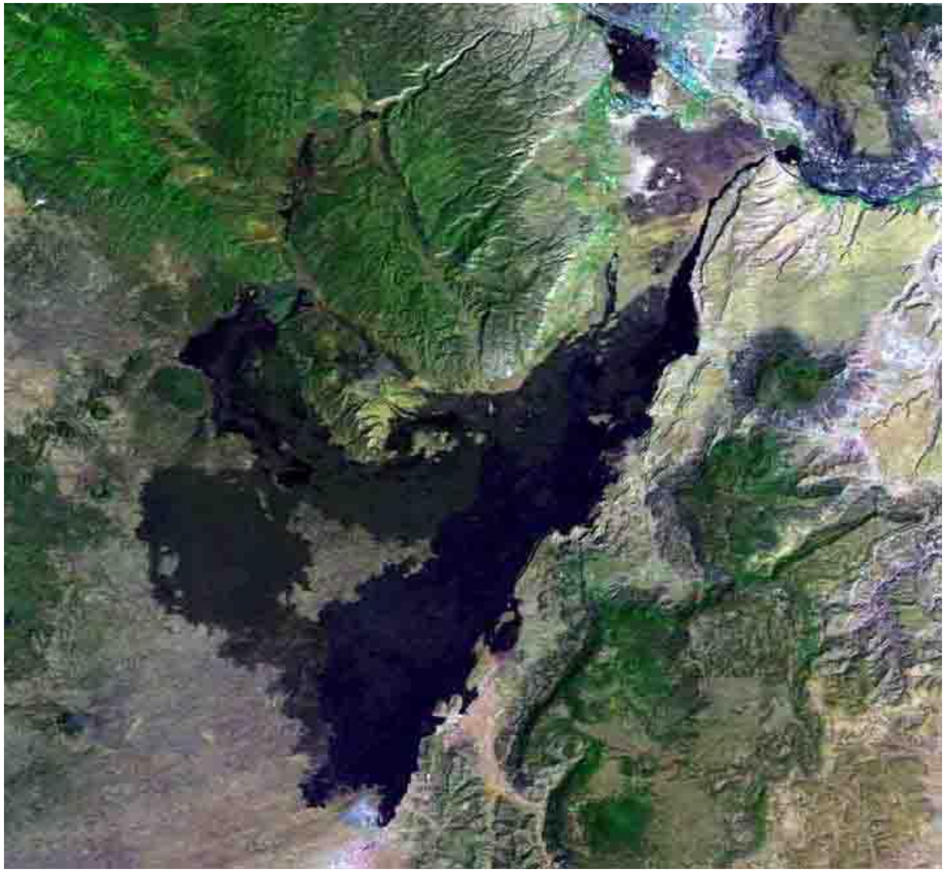
Raymond H. Chan

The Chinese University of Hong Kong, Hong Kong, China

Outline

- Image restoration
- Texture-preserving: the regularization operator is constructed by using fractional order derivatives
  - Model and Numerical Algorithm
  - Fractional-order derivatives
  - Numerical Examples
- Edge-preserving: norm adapted to the image features
- Simple iterative alternating algorithms based on the half-quadratic strategy.

Image Recovery



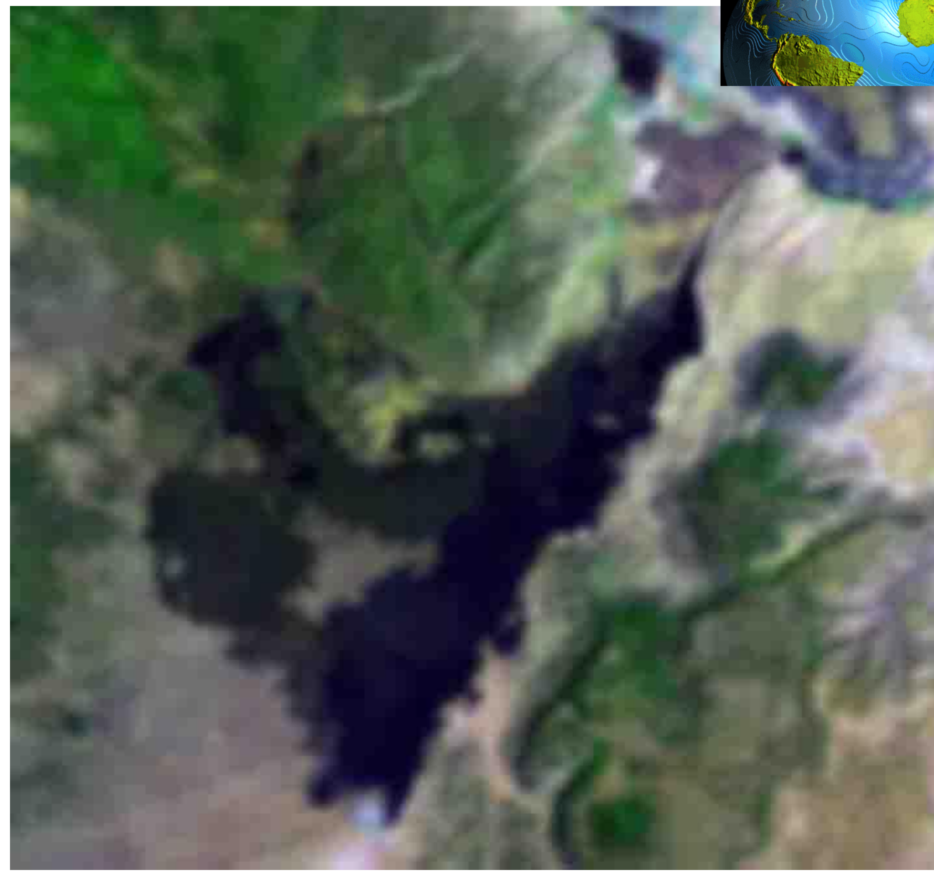
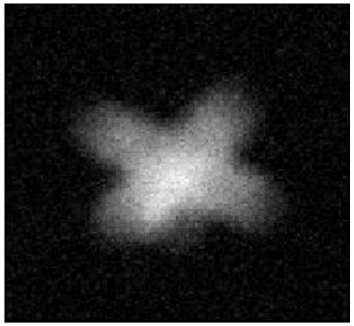
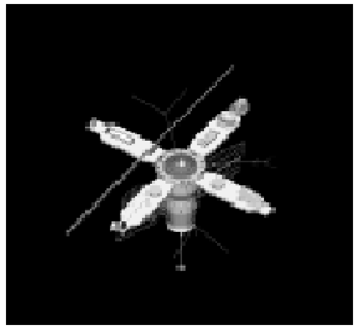


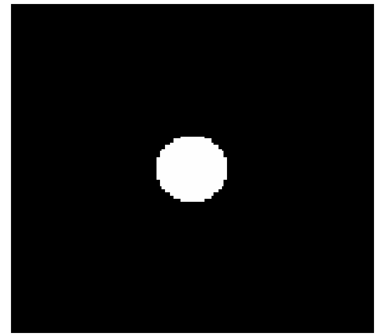
Image Restoration Problem



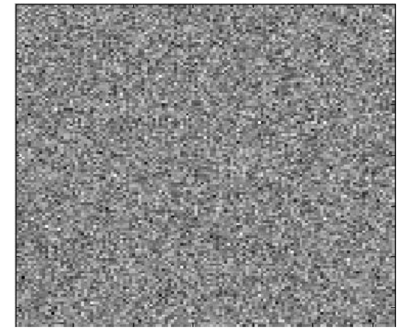
=



*



+



Observed image	Unknown true image	Known Point Spread Function	Unknown noise
f	u	k	e

Goal: Given ***f***, recover **u**

Degradation model

Continuous degradation model:

$$f(x) = \iint_{\Omega} k(x, y) u(y) dy + e(x) \qquad x \in \Omega$$

Perturbed observed image

Point Spread Function

Blur and noise-free image

Data noise

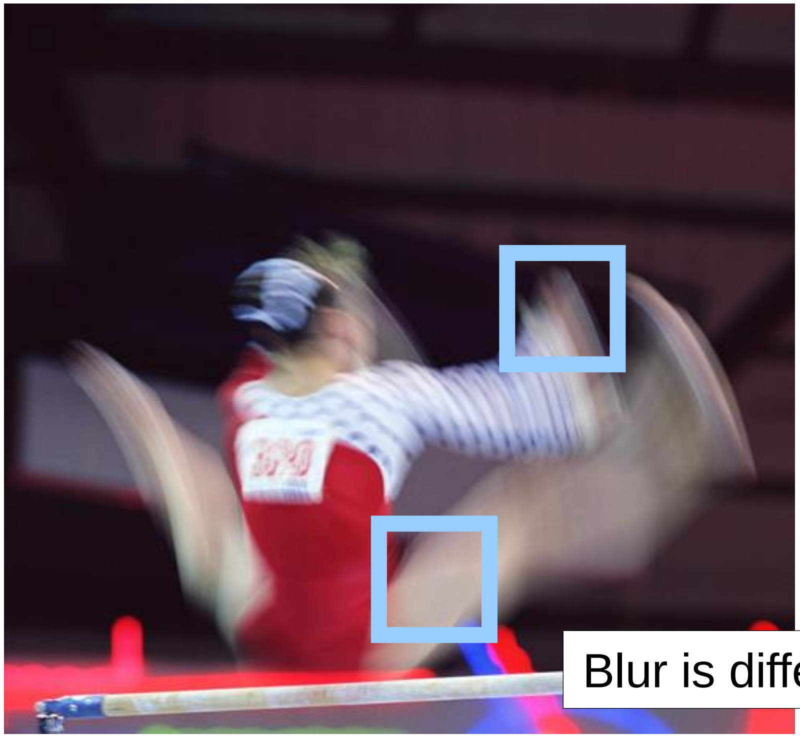
Integral equation can be expressed as

$$f = k * u + e$$

Space variant - space invariant blur

Two causes for motion blur

Hand shaking



Object motion

Degradation model

Continuous degradation model:

$$f(x) = \iint_{\Omega} k(x,y)u(y)dy + e(x) \qquad x \in \Omega$$

Perturbed observed image

Point Spread Function

Blur and noise-free image

Data noise

Integral equation can be expressed as

$$f = k * u + \cancel{e}$$

Discretization yields

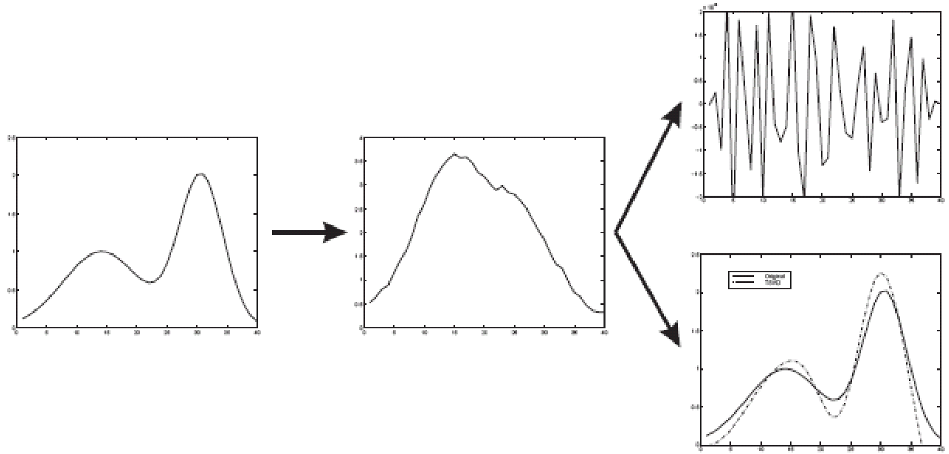
$$\mathbf{f} = \mathbf{K}\mathbf{u}$$

with matrix K block Toeplitz with Toeplitz blocks

P. C. Hansen, *Regularization tools version 4.0 for Matlab 7.3*, Numer. Algorithms, Vol. 46, 2007.

Solution **Ku=f**: add 0.1% noise to rhs

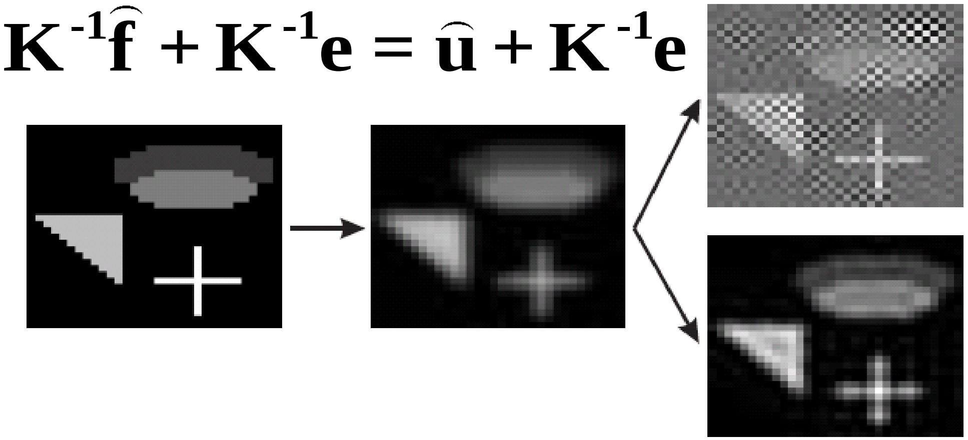
Shaw.m



u=K^{-1}f

$$\mathbf{f} = \widehat{\mathbf{f}} + \mathbf{e}$$

$$\mathbf{u} = \mathbf{K}^{-1}(\widehat{\mathbf{f}} + \mathbf{e}) = \mathbf{K}^{-1}\widehat{\mathbf{f}} + \mathbf{K}^{-1}\mathbf{e} = \widehat{\mathbf{u}} + \mathbf{K}^{-1}\mathbf{e}$$



u=K^{-1}f

Regularization

- Minimize the energy functional

$$E(u) = \left\{ \int_\Omega \Phi((k*u - f)^2) + \lambda R(|\nabla u|^2) dx \right\}$$

Data term: enforces the match between the sought image and the observed image via the blur model

Smoothness term: brings in regularity assumptions about the unknown image

$$\Phi(s^2) = s^2 \qquad R(s^2) = s^2 \qquad \textbf{Tikhonov}$$

$$R(s^2) = \sqrt{s^2 + \varepsilon^2} \qquad \textbf{TV}$$

$$R(s^2) = \rho^2 \ln(1 + s^2 / \rho^2) \qquad \textit{Perona} - \textit{Malik}$$

Regularization: discrete setting

Solve the minimization problem $$\min_{u} \left\{ \| \mathbf{Ku} - \mathbf{f} \|_p^p + \frac{\lambda}{q} \| A(\mathbf{u}) \|_q^q \right\},$$

- **A** is a regularization operator, λ is a positive regularization parameter that controls the trade-off between the data fitting term and the regularization term.

- **p = 2, q = 2**, **Tikhonov regularization**
- **p = 2, q = 1,** TV regularization (***ℓ_2-TV***) ***A(u) the gradient magnitude of u***.
- **p = 1, q = 1,** TV regularization (***ℓ_1-TV***) ***A(u) the gradient magnitude of u***.

Regularization: TV

ℓ_1-TV **regularization** $\min_{u} \left\{ \|\mathbf{Ku} - \mathbf{f}\|_1 + \lambda \|\mathbf{u}\|_{TV} \right\},$

-

$$\|u\|_{TV} = \|A(u)\|_1 := \sum_{i=1}^{n^2} \sqrt{\left(G_{x,i}u\right)^2 + \left(G_{y,i}u\right)^2}$$

$$\nabla u_i := \left(G_{x,i}u, G_{y,i}u\right)^T$$

ℓ_1-TV regularization

has problems in preserving textures

blocky smoothed image

Adaptive Fractional (AF) Variational model

Replace the TV regularization term $\|u\|_{TV}$ ***with a spatially adaptive fractional order TV regularization term***.

- **fractional order a of derivatives** to better preserve textures,
- spatial **adaptivity of** α in order to allow flexibility in choosing the correct regularizing operator,
- spatial **adaptivity of λ** in order to locally control the extent of restoration over image regions according to their content,
- **effective texture detection methodology** based on the noise auto-correlation energy which makes no assumption about the noise level of the image.

Adaptive Fractional Variational model

$$\min_{u} \left\{ \|\mathbf{Ku} - \mathbf{f}\|_1 + \|\Lambda A_\alpha(u)\|_1 \right\},$$

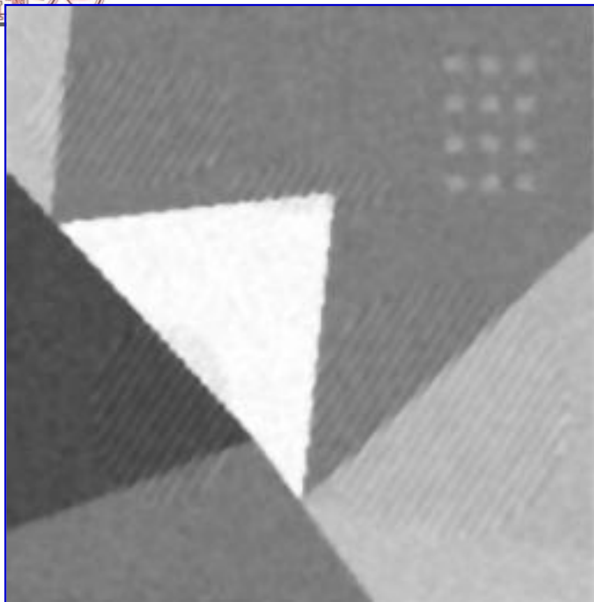
where

$\Lambda = diag(\lambda_1, ..., \lambda_{n^2})$ $n^2 \times n^2$ diagonal matrix λ_i representing the regularization parameter for the ith pixel,

$A_\alpha(u_i) = \|\nabla^{\alpha_i} u_i\|$ α_i represents the fractional order of differentiation for the ith pixel,

$\nabla^{\alpha_i} u_i := \left(G^{\alpha_i}_{x,i} u, G^{\alpha_i}_{y,i} u\right)^T$ is the fractional-order discrete gradient operator, with components representing the x and y-directional fractional finite difference operators.

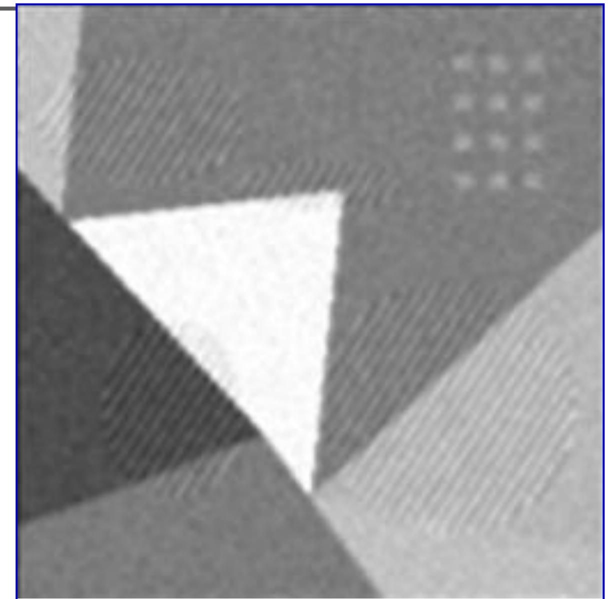
Fractional derivatives for texture preserving

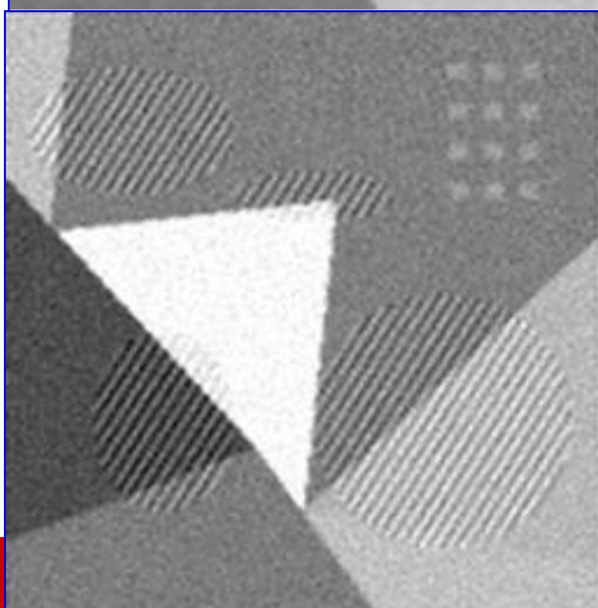


$\alpha=1.0$

ℓ_1-TV model preserves edges but fails to preserve fine scale features such as textures

$\alpha=1.5$

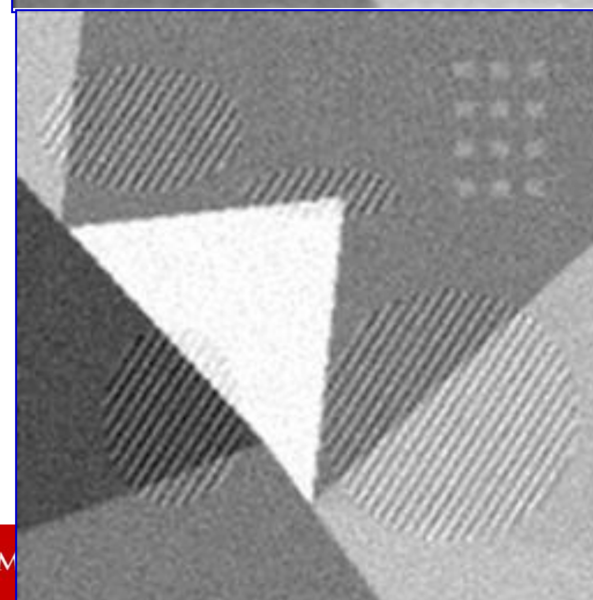




$\alpha=1.8$

The high-pass capability becomes stronger with larger α

$\alpha=2.0$



Adaptive Fractional Variational Algorithm

First phase: apply the **texture detector** to the observed image **f** to obtain a texture map.

The texture map is partitioned into ***C subclasses*** according to the texture measure.

$$\alpha_i = \begin{cases} \mathbf{1} & \text{if the ith pixel belongs to the non-texture class} \\ \{\hat{\alpha}_1, \ldots, \hat{\alpha}_C\} & \text{if the ith pixel belongs to one of the C texture subclasses} \end{cases}$$

The regularization parameters λ_i in the diagonal matrix Λ are then chosen according to α_i's;

Non-texture class has $\lambda = 1.0$

Second phase: apply TV regularization (ℓ_1-TV) to the non-texture regions apply a fractional order TV regularization (ℓ_1-TV$^\alpha$) in the texture classes.

The numerical algorithm

Minimize the functional

$$\|v\|_{1,\beta} := \sum_i |v_i|_\beta$$

$$|v_i|_\beta := \sqrt{v_i^2 + \beta}$$

$$\Phi(u) = \|Ku - f\|_{1,\gamma} + \|\Lambda A_\alpha(u)\|_{1,\beta}$$

$$= \sum_{i=1}^{n^2} |K_i u - f_i|_\gamma + \lambda_i |\nabla^{\alpha_i}(u_i)|_\beta \qquad (**)$$

$$= \sum_{i=1}^{n^2} \sqrt{(K_i u - f)^2 + \gamma} + \lambda_i \sqrt{(G_{x,i}^{\alpha_i} u)^2 + (G_{y,i}^{\alpha_i} u)^2 + \beta}$$

where

K_i is the i-th row of K,

f_i is the intensity of the i-th pixel of the observed image

The numerical algorithm

Half-quadratic regularization

$$|x| = \min_{v>0}\left\{ v x^2 + \frac{1}{4v} \right\} \quad \text{minimum at } v = \frac{1}{2|x|}$$

quadratic in x but not in v

$$\min_u \Phi(u) := \min_{u,v>0,w>0} \mathcal{L}(u,v,w)$$

$$\mathcal{L}(u,v,w) = \sum_{i=1}^{n^2}\left[w_i \left|K_i u - f_i\right|_\gamma^2 + \frac{1}{4w_i} + \lambda_i\left(v_i \left|\nabla^{\alpha_i} u_i\right|_\beta^2 + \frac{1}{4v_i}\right)\right]$$

[a] M. Nikolova and R. Chan, The equivalence of half-quadratic minimization and the gradient linearization iteration, IEEE Trans. Image Proc.,vol. 16, pp. 1623–1627, 2007.

[b] D. Geman and C. Yang, Nonlinear image recovery with half-quadratic regularization and FFTs, IEEE Trans. Image Proc., vol. 4, pp. 932–946, 1995.

Alternating minimization procedure

For each iteration step k = 0, 1, . . ., we solve successively

$$\min_{u,v>0,w>0} \mathcal{L}(u,v,w)$$

$$v^{(k+1)} = \arg\min_{v>0} \mathcal{L}(u^{(k)},v,w^{(k)})$$

$$w^{(k+1)} = \arg\min_{w>0} \mathcal{L}(u^{(k)},v^{(k+1)},w)$$

$$u^{(k+1)} = \arg\min_{u} \mathcal{L}(u,v^{(k+1)},w^{(k+1)})$$

For each iteration step k:

1. Explicit solution: $$v_i^{(k+1)} = \frac{1}{2}\left|\nabla^{\alpha_i} u_i^{(k)}\right|_\beta^{-1}$$

2. Explicit solution $$w_i^{(k+1)} = \frac{1}{2}\left|K_i u^{(k)} - f_i\right|_\gamma^{-1}$$

3. Compute u by imposing $$0 = \nabla_u \mathcal{L}(u,v^{(k+1)},w^{(k+1)}) = \left(G^\alpha\right)^T \hat{\Lambda}\hat{D}_\beta\left(u^{(k)}\right)G^\alpha + K^T D_\gamma\left(u^{(k)}\right)(Ku - f)$$

Alternating minimization procedure

3. Compute u by solving

$$u^{(k+1)} = \arg\min_u \mathcal{L}(u, v^{(k+1)}, w^{(k+1)})$$

$$\left[\left(G^{\alpha}\right)^T \hat{\Lambda}\hat{D}_{\beta}\left(u^{(k)}\right)G^{\alpha} + K^T D_{\gamma}\left(u^{(k)}\right)K\right]u^{(k+1)} = K^T D_{\gamma}\left(u^{(k)}\right) f$$

$G^{\alpha} := \left[G_x^{\alpha} \;;\; G_y^{\alpha}\right] \in \mathbb{R}^{n^2 \times n^2}$ discretization matrix of the adaptive fractional gradient operator

$\hat{\Lambda} := diag(\Lambda, \Lambda) \in \mathbb{R}^{2n^2 \times 2n^2}$ diagonal matrix of the adaptive regularization parameters

$\hat{D}_{\beta}\left(u^{(k)}\right) := diag(D_{\beta}\left(u^{(k)}\right), D_{\beta}\left(u^{(k)}\right)) \in \mathbb{R}^{2n^2 \times 2n^2}$ $\quad \hat{D}_{\gamma}\left(u^{(k)}\right) \in \mathbb{R}^{2n^2 \times 2n^2}$

$$\left(D_{\beta}\left(u^{(k)}\right)\right)_i = 2v_i^{(k+1)} = \frac{1}{\left|\nabla^{\alpha_i} u_i^{(k)}\right|_{\beta}}$$

$$\left(D_{\gamma}\left(u^{(k)}\right)\right)_i = 2w_i^{(k+1)} = \frac{1}{\left|K_i u^{(k)} - f_i\right|_{\gamma}}$$

Adaptive-Fractional (AF) Algorithm

Input: **degraded image f** ***, number of texture classes C;***

Output: **approximate solution $u^{(k)}$ of (**);**

1. $\{\lambda_i, \alpha_i, i = 1,..,n^2\} = \mathbf{TD}(f;C)$ compute the texture-adaptive parameters on the degraded image ***f***;

2. Initialize the iterative process by setting $\mathbf{u}^{(0)} = \mathbf{f}$;

3. **For** k = 1, 2, . . . until convergent, solve

$$\left[\left(G^{\alpha}\right)^{T}\hat{\Lambda}\hat{D}_{\beta}\left(u^{(k)}\right)G^{\alpha}+K^{T}D_{\gamma}\left(u^{(k)}\right)K\right]u^{(k+1)}=K^{T}D_{\gamma}\left(u^{(k)}\right)f$$

endfor

$$\left[\left(G^{\alpha}\right)^{T}\hat{\Lambda}\hat{D}_{\beta}\left(u^{(k)}\right)G^{\alpha}+K^{T}D_{\gamma}\left(u^{(k)}\right)K\right]u^{(k+1)}=K^{T}D_{\gamma}\left(u^{(k)}\right)f$$

- Solver: the conjugate gradient method
- Stopping criterium: norm of the residual is less than or equal to 10^{-4}.
- No storage problems for large dimension matrices **K** and $\mathbf{G^a}$
- *the only requirement is matrix-vector products.*
- The product which involves matrix K makes use of FFT convolution.

How to compute the matrix-vector product $\left((G^{\alpha})^{T}\hat{\Lambda}\hat{D}_{\beta}G^{\alpha}\right)u^{(k+1)}$

Convergence

Theorem

For the sequence $u^{(k)}$ generated by the half-quadratic **AF Algorithm**, if

$$\ker\left((G^{\alpha})^T G^{\alpha}\right) \cap \ker\left(K^T K\right) = \{0\} \qquad (*)$$

we have:

- $\left\{\Phi\left(u^{(k)}\right)\right\}$ is monotonic decreasing and convergent;
- $\lim_{k\to\infty} \left\| u^{(k)} - u^{(k+1)} \right\|_2 = 0$
- $\left\{\Phi\left(u^{(k)}\right)\right\}$ converges to the unique minimizer u^* of $\Phi(u)$ from any initial guess $u^{(0)}$

Remark: in our case, for $\alpha \in [1, 2]$, $\ker((G^{\alpha})^T G^{\alpha})$ is spanned at most by the two vectors: $\mathbf{1}_{n^2}$, a n^2 vector of ones, and $(1, 2, \ldots, n^2)$, while the blurring matrix K is a low-pass filter.

Grunwald-Letnikov Fractional-order derivatives

The discrete fractional-order gradient at a pixel (i, j) is defined as

$$\left(\nabla^{\alpha_{i,j}} u\right)_{i,j} = \left(\left(\Delta_x^{\alpha_{i,j}} u\right)_{i,j}, \left(\Delta_y^{\alpha_{i,j}} u\right)_{i,j}\right)$$

$$\left(\Delta_x^{\alpha_{i,j}} u\right)_{i,j} = \sum_{s=0}^{L-1} \omega_s^{\alpha_{i,j}} u_{i-s,j} \qquad \alpha_{i,j} \in \mathbb{R}^+$$

$$\left(\Delta_y^{\alpha_{i,j}} u\right)_{i,j} = \sum_{s=0}^{L-1} \omega_s^{\alpha_{i,j}} u_{i,j-s}$$

where L > 0 is the number of pixels used for the approximation, and ω^{α}_s , for a generic $\alpha = \alpha_{i,j}$, are the real coefficients defined as

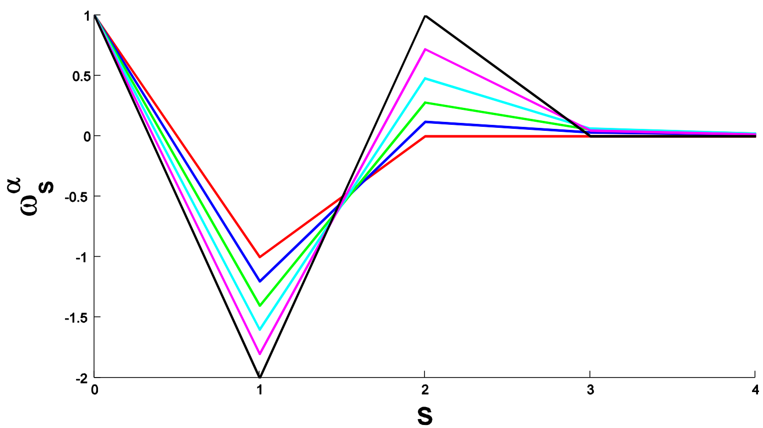
$$\omega_s^{\alpha} = (-1)^s \binom{\alpha}{s} = (-1)^s \frac{\Gamma(\alpha+1)}{\Gamma(s+1)\Gamma(\alpha-s+1)}, \qquad \alpha \in \mathbb{R}^+, \;\; s \in \mathbb{N}$$

The generalized binomial coefficients $\binom{\alpha}{s}$ are computed by the following recurrence relationships

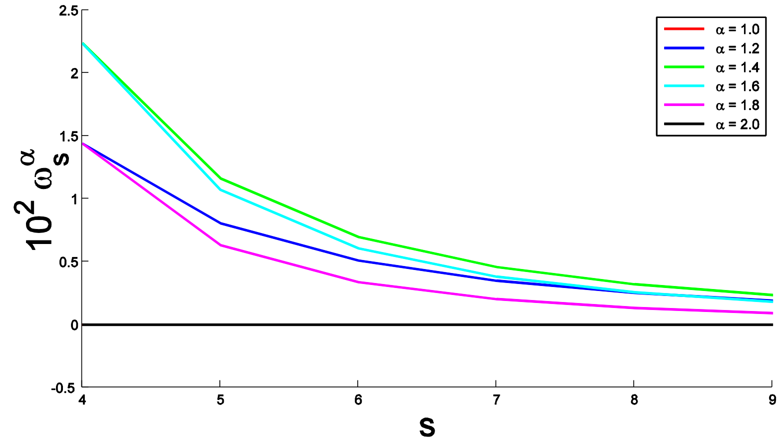
$$\binom{\alpha}{0} = 1; \quad \binom{\alpha}{s} = \binom{\alpha}{s-1} \bullet \left(1 - \frac{\alpha+1}{s}\right) \qquad \alpha \in \mathbb{R}^+, \;\; s = 1, 2, ...$$

Grunwald-Letnikov Fractional-order derivatives

$$\omega_s^{\alpha} = (-1)^s \binom{\alpha}{s} = (-1)^s \frac{\Gamma(\alpha+1)}{\Gamma(s+1)\Gamma(\alpha-s+1)}, \qquad \alpha \in \mathbb{R}^+, \;\; s \in \mathbb{N}$$



the first fourth s values



the remaining coefficients in a different plot scale

Coefficients ω_s^α for different values of $\alpha \in [1, 2]$ and increasing s values.
Only the first ten coefficients $\omega_0^\alpha, \ldots, \omega_9^\alpha$ are reported since they vanish to zero very fast.

For $\alpha = 1$ and $\alpha = 2$ the coefficients exactly vanish for $s > 1$ and $s > 2$, respectively, and in fact they reduce to the discretizations of the first and second order derivatives respectively.

Finally, we point out that the coefficients sum up to zero independently on $\alpha \in [1, 2]$.

Fractional-order Gradient operator

$$G^{\alpha} = \left(G_x^{\alpha}, G_y^{\alpha}\right) \quad 2n^2 \times n^2 \quad \text{matrix}$$

Non-adaptive α, Assuming Dirichlet homogeneous boundary conditions

$$G_x^{\alpha} = I_n \otimes U^{\alpha} \qquad \text{block Toeplitz withToeplitz blocks}$$

$$G_y^{\alpha} = U^{\alpha} \otimes I_n$$

$\otimes$ denotes the Kronecker product,

I_n is the n-order identity matrix

U_{α} is the n×n Toeplitz lower triangular banded matrix whose first column is $(\omega^{\alpha}_0, \omega^{\alpha}_1, \ldots, \omega^{\alpha}_{L-1})$

Adaptive α, the two matrices retain the same sparsity structure but are no longer Toeplitz: each row will contain different coefficients depending on the fractional-order of differentiation selected for the corresponding pixel

Numerical Experiments

ℓ_2-TV method *Rudin-Osher-Fatemi model*

ℓ_1-TV method

$$\min_{u} \left\{ \|\mathbf{Ku} - \mathbf{f}\|_1 + \lambda \|\mathbf{u}\|_{TV} \right\}, \qquad \lambda = 0.1$$

$\alpha_i = 1, \quad \lambda_i = 0.1,$ for all i in the difference matrix G^{α} and in the diagonal matrix Λ

AF algorithm

Neumann homogeneous boundary conditions for the difference matrix G^{α}

$\beta = 10^{-3}$, $\gamma = 10^{-6}$, the number of nodes **L = 8**

Signal-to-Noise Ratio (SNR)

$$SNR(u, \hat{u}) := 10 \log_{10} \frac{\|u - E(\hat{u})\|}{\|u - \hat{u}\|} dB$$

u available approximation of the desired **blur- and noise-free image û**

E(û) mean gray-level value of the uncorrupted image

Numerical Experiments

The **matrix K** represents a **Gaussian blurring operator** generated by the Matlab function **`blur.m`** in Regularization Tools [P.C.Hansen].

Band specifies the half-bandwidth of the Toeplitz blocks

Sigma is the variance of the Gaussian point spread function.

The larger sigma, the more blurring.

Enlarging band increases the storage requirement, the arithmetic work required for the evaluation of matrix-vector products with K, and to some extent the blurring.

Numerical Experiments

f contaminated either by **additive Gaussian noise** or by **salt-and pepper** noise.

In the case of **Gaussian noise**,

$$\tilde{f} \in \mathbb{R}^{n^2}$$ blurred image

$$f = \tilde{f} + e$$ e represents the noise.

noise-level $$\nu = \frac{\|e\|}{\|\tilde{f}\|}$$

In the **salt-and-pepper noise** white and black pixels randomly occur, while unaffected pixels always remain unchanged.

The salt-and-pepper noise is quantified by the percentage of corrupted pixels

Numerical Experiments

We partitioned the **texture-map** only into **four classes**,
three texture classes and one non-texture class,
associated fractional order and regularization parameter values

$\alpha_1 = 1.9, \ \lambda_1 = 0.05, \quad \alpha_2 = 1.8, \ \lambda_2 = 0.05,$
$\alpha_3 = 1.7, \ \lambda_3 = 0.05 \quad \alpha_4 = 1.0, \ \lambda_4 = 1.0,$

For this particular case, the diagonal entries of the matrix Λ may assume one of the four different values $\lambda_1, \lambda_2, \lambda_3$ and λ_4.

The core of the algorithm:
outer iteration loop (step 3) : **at most 10 outer iterations**
inner iteration loop required by CG for the linear system:
with 10^{-4} *as stopping tolerance* - **an average of 18 inner iterations**.

Example 1

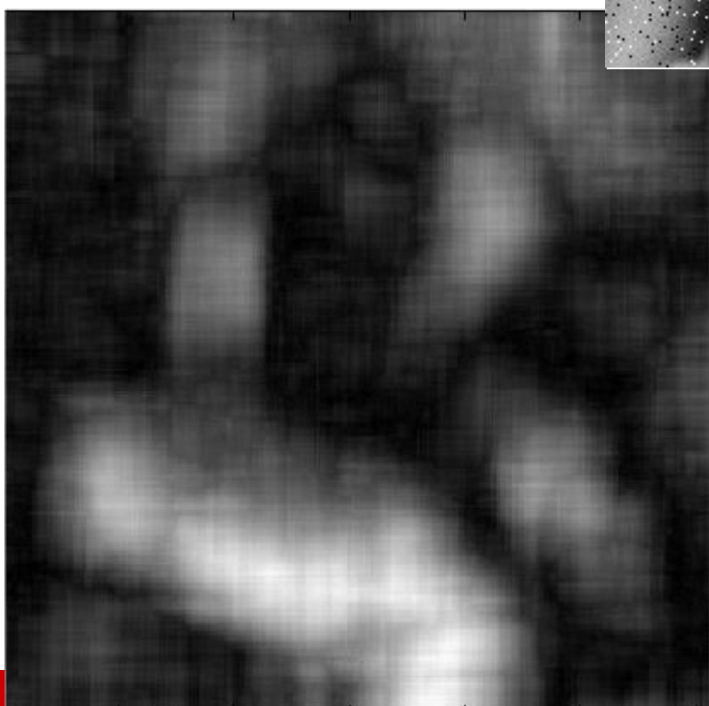
%5 salt-and-pepper noise

Gaussian blur,
band = 3 , sigma = 1*.5*

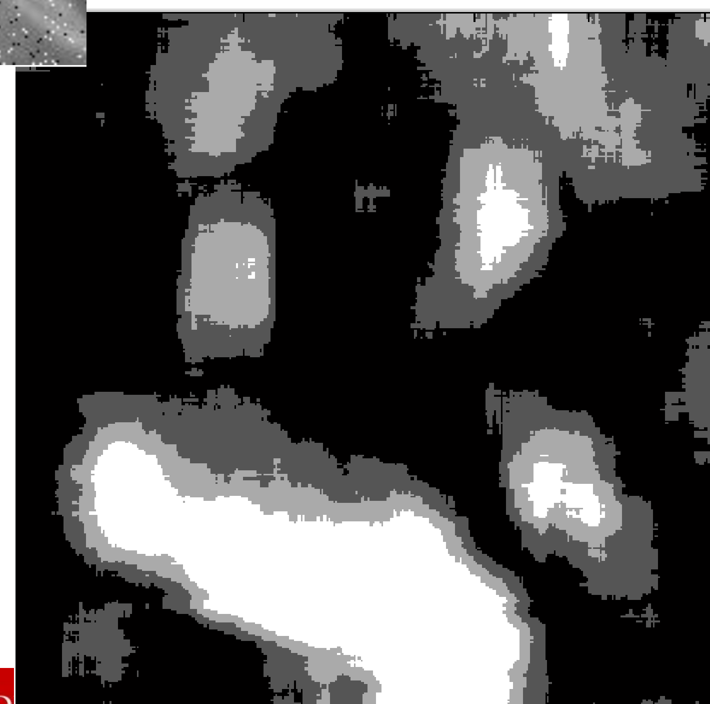
AF

Example 1

texture map



texture classes



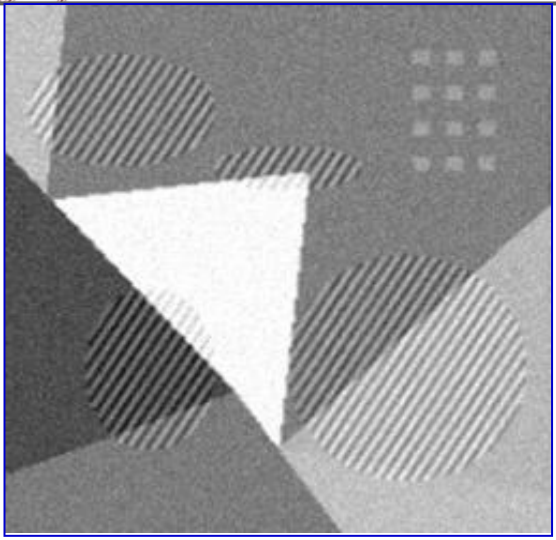
Example 1

ℓ_2-TV

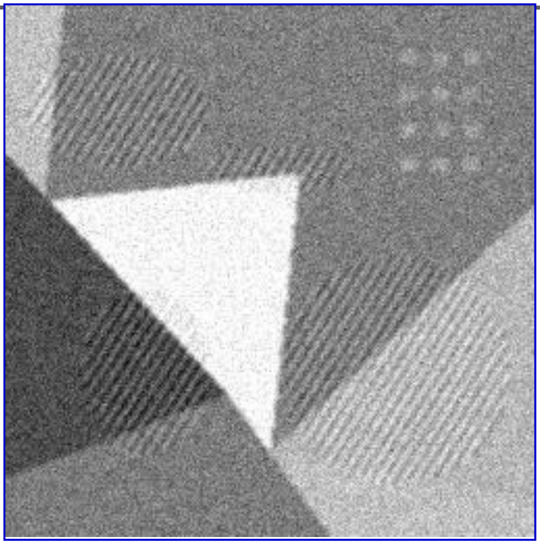
AF

n	SNR_i	SNR_{AF}	$\text{SNR}_{\ell 1-\text{TV}}$	$\text{SNR}_{\ell 2-\text{TV}}$
5%	3.89dB	14.12	13.39	7.56
10%	1.42dB	13.17	12.93	6.61

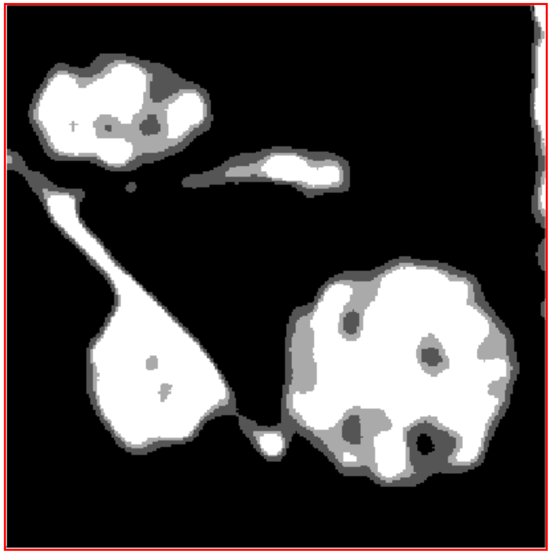
Example 2



true image
255x255

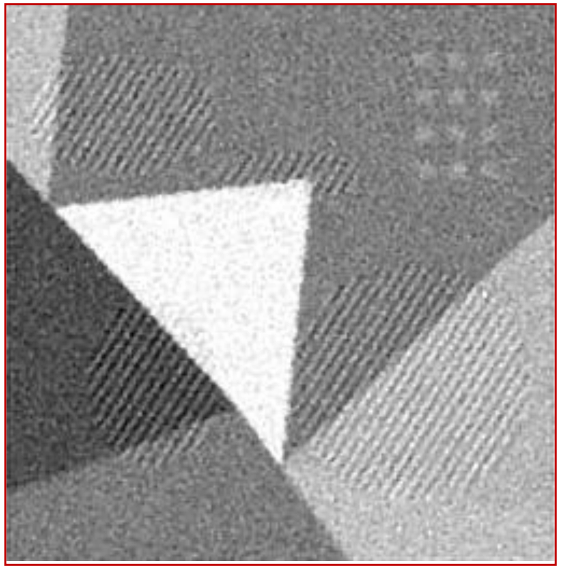


Observed image
spatially-invariant Guassian blur
band = 3
sigma = 1.5,
10% noise

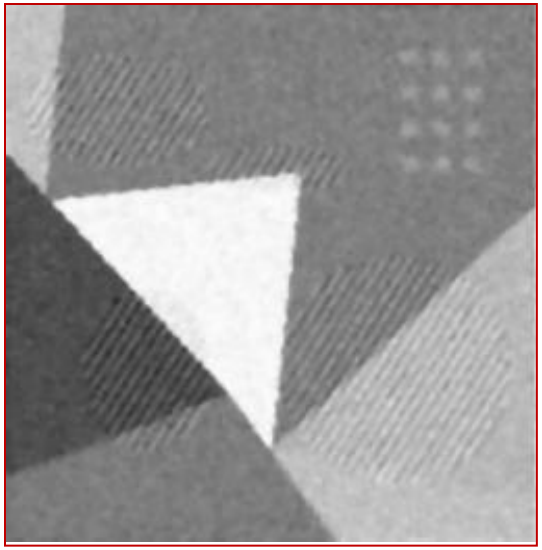


texture map

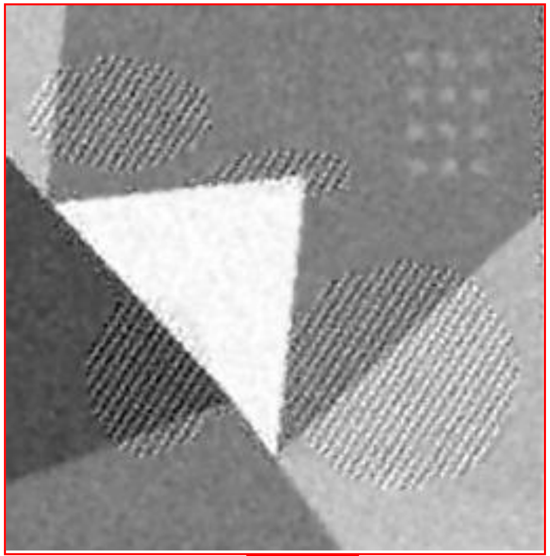
Example 2



ℓ_1-TV



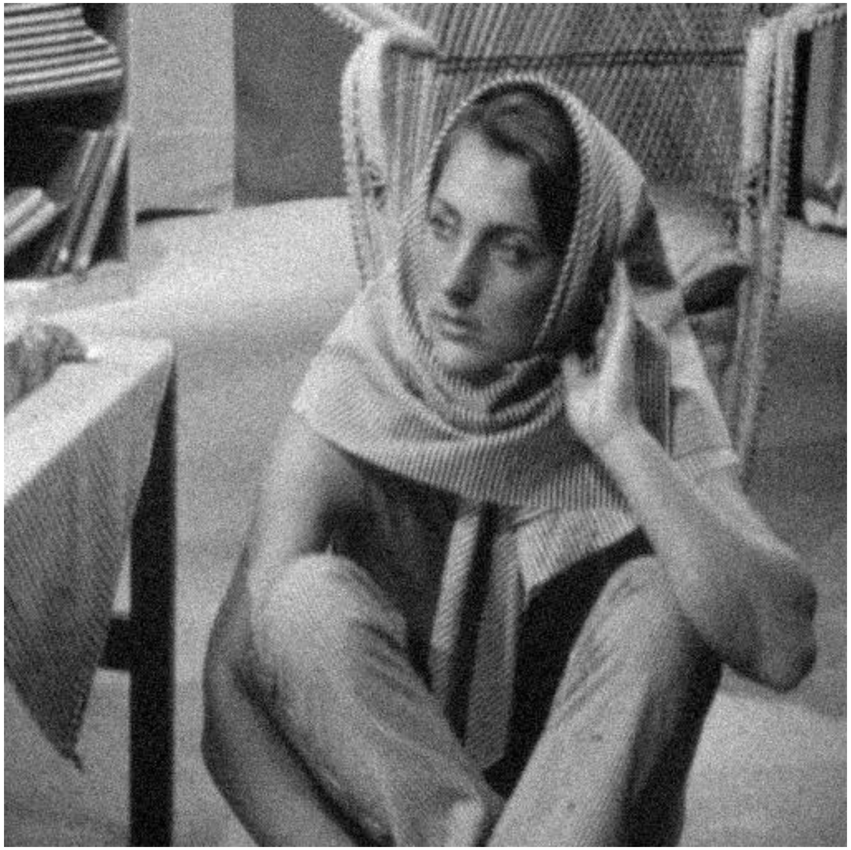
ℓ_2-TV



AF

ν	SNR_i	SNR_{AF}	$\text{SNR}_{\ell 1-\text{TV}}$	$\text{SNR}_{\ell 2-\text{TV}}$
0.01	15.98	20.46	20.00	18.22
0.05	13.29	15.86	15.06	15.48

Example 3



true image 510x510

observed image

Gaussian blur, band = 3. sigma =1*.5, 10% Gaussian noise*

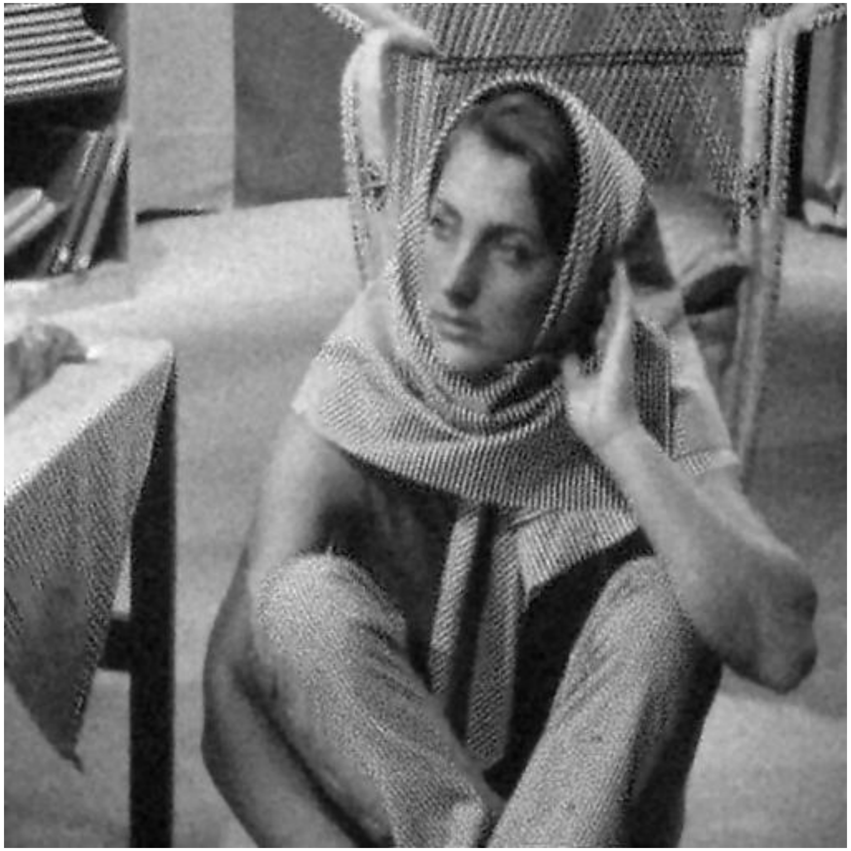
Example 3

texture map from original

texture map from corrupted

Example 3

ℓ_1-TV



AF

ν	SNR_i	$\mathrm{SNR}_{\mathrm{AF}}$	$\mathrm{SNR}_{\ell 1-\mathrm{TV}}$	$\mathrm{SNR}_{\ell 2-\mathrm{TV}}$
0.01	11.41	14.00	13.53	12.14
0.05	10.71	11.94	10.89	11.64
0.10	9.05	10.59	9.73	10.26

Example 4

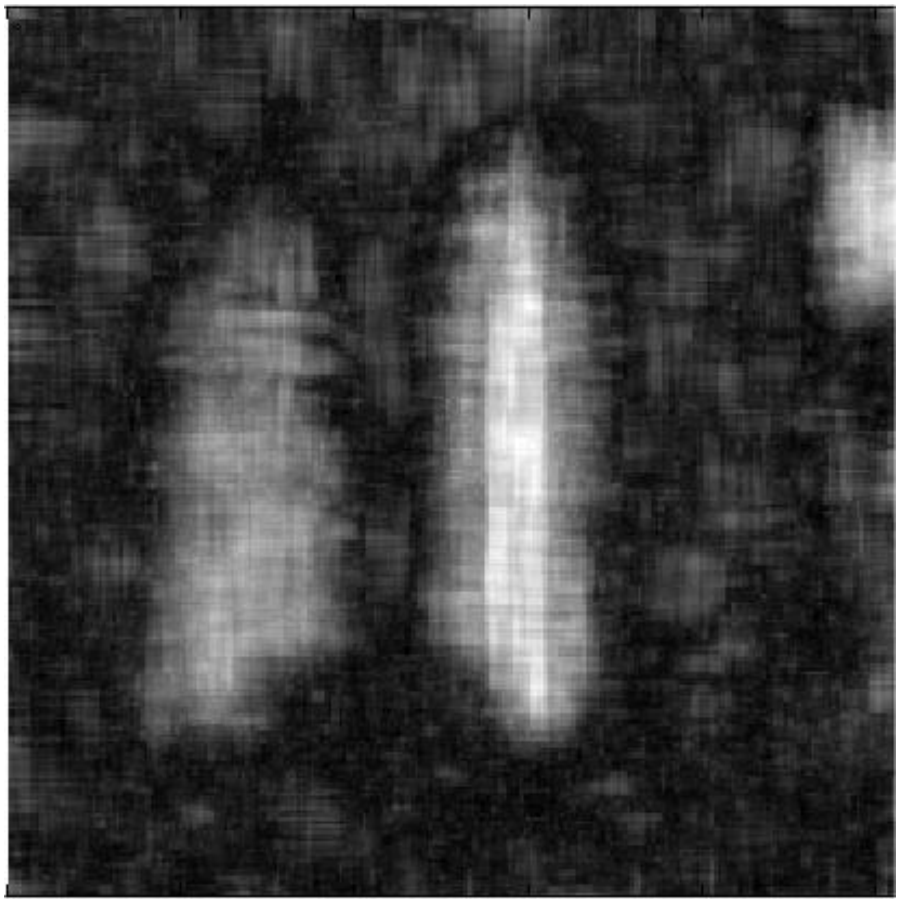
True image
256x256

Observed Image
10% Gaussian noise
Gaussian blur
band = 3 , sigma = 1.5

SNR=8.17

Example 4

texture map



texture classes

Example 4

AF

SNR=9.79

%10 Gaussian noise
Gaussian blur
band = 3 , sigma = 1.*5*

SNR=8.17

Example 4

AF

SNR=9.79

ℓ_1-TV

SNR=8.44

Regularization: adaptive norm

Solve the minimization problem

$$\min_{u} \left\{ \|\mathbf{Ku} - \mathbf{f}\|_p^p + \frac{\lambda}{q} \|A(\mathbf{u})\|_q^q \right\},$$

A is a regularization operator, λ is a positive regularization parameter that controls the trade-off between the data fitting term and the regularization term.

- **p = 2, q = 2**, **Tikhonov regularization** Gaussian noise, oversmoothed
- **p = 2, q = 1,** TV regularization (*ℓ_2-TV*)
- **p = 1, q = 1,** TV regularization (*ℓ_1-TV*) *I*mpulse noise, blocky restored images)

Main goal: **adaptively consider a suitable norm (q = 1 or q = 2) driven by a coherence map of the image structures (smooth regions or edges).**

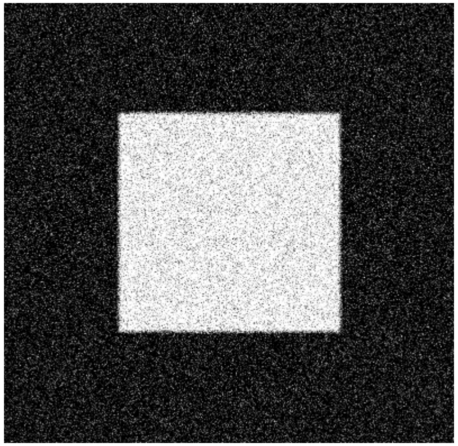
Adaptive Norm (AN) image restoration model

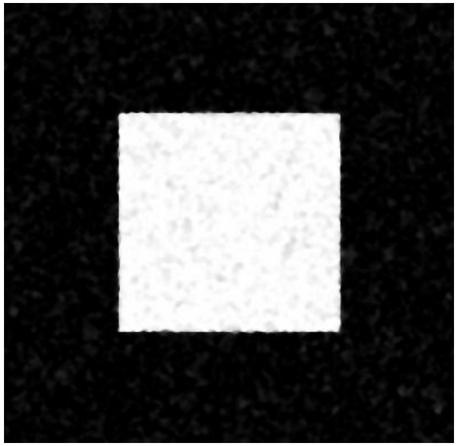
Gaussian noise
band = 5, sigma = 3
noise 5%

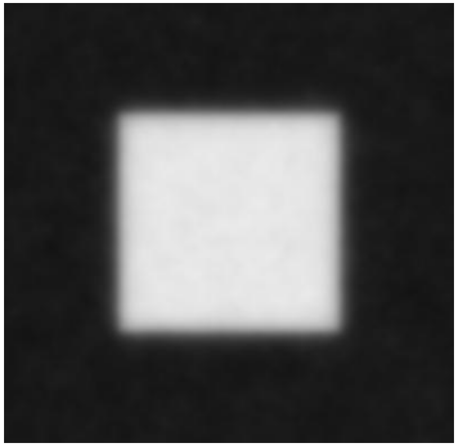
L1-TV p = 1, q = 1
SNR = 20.30

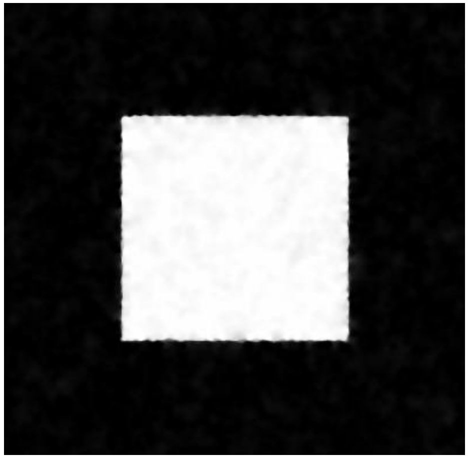
p = 2, q = 2,
SNR = 9.62

adaptive-norm p = 1
SNR = 20.93









Coherence matrix construction

1. Compute the tensor matrix $S_\delta(\nabla u_\sigma) := \left(K_\delta * \left(\nabla u_\sigma \otimes \nabla u_\sigma\right)\right)$

   $\mathbf{K}_\delta$ is a Gaussian kernel

2. Compute λ_1 λ_2 eigenvalues of $\boldsymbol{S}_\delta$

The matrix $\boldsymbol{S}_\delta$ is symmetric positive semi-definite and its eigenvalues $\boldsymbol{\lambda_1}$ $\boldsymbol{\lambda_2}$ integrate the variation of the gray values within a neighborhood of size O(δ).

$\boldsymbol{\lambda_1 = \lambda_2 = 0}$ constant areas,

$\boldsymbol{\lambda_1 >> \lambda_2 = 0}$ straight edges.

Coherence matrix construction

3. Compute normalized coherence value at pixel i-th

$$c_i = \frac{(\lambda_1 - \lambda_2)^2}{max\{(\lambda_1 - \lambda_2)^2\}}$$

4. Construct the diagonal matrix C

$$C_{ii} = \begin{cases} 0 & c_i < \tau \quad \textit{homogenous regions} \\ 1 & c_i \geq \tau \quad \textit{edges} \end{cases}$$

Adaptive Norm (AN) image restoration model

Solve the minimization problem

$$\min_{\mathbf{u}} \left\{ \|\mathbf{Ku} - \mathbf{f}\|_p^p + \frac{\lambda}{q} \|A(\mathbf{u})\|_q^q \right\},$$

$$\min_u \Phi(u) \qquad \Phi(u) = \|Ku - f\|_1^1 + \mu_1 \|CAu\|_1^1 + \mu_2 \|(I - C)Lu\|_2^2$$

C diagonal coherence matrix

Regularization operator

Alternating minimization procedure

For each iteration step k = 0, 1, . . ., we solve successively

$$\min_{u,v>0,w>0} \mathcal{L}(u,v,w)$$

$$v^{(k+1)} = \arg\min_{v>0} \mathcal{L}(u^{(k)}, v, w^{(k)})$$

$$w^{(k+1)} = \arg\min_{w>0} \mathcal{L}(u^{(k)}, v^{(k+1)}, w)$$

$$u^{(k+1)} = \arg\min_{u} \mathcal{L}(u, v^{(k+1)}, w^{(k+1)})$$

For each iteration step k:

1. Explicit solution:

$$v_i^{(k+1)} = \frac{1}{2}\left|\nabla^{\alpha_i} u_i^{(k)}\right|_{\beta}^{-1}$$

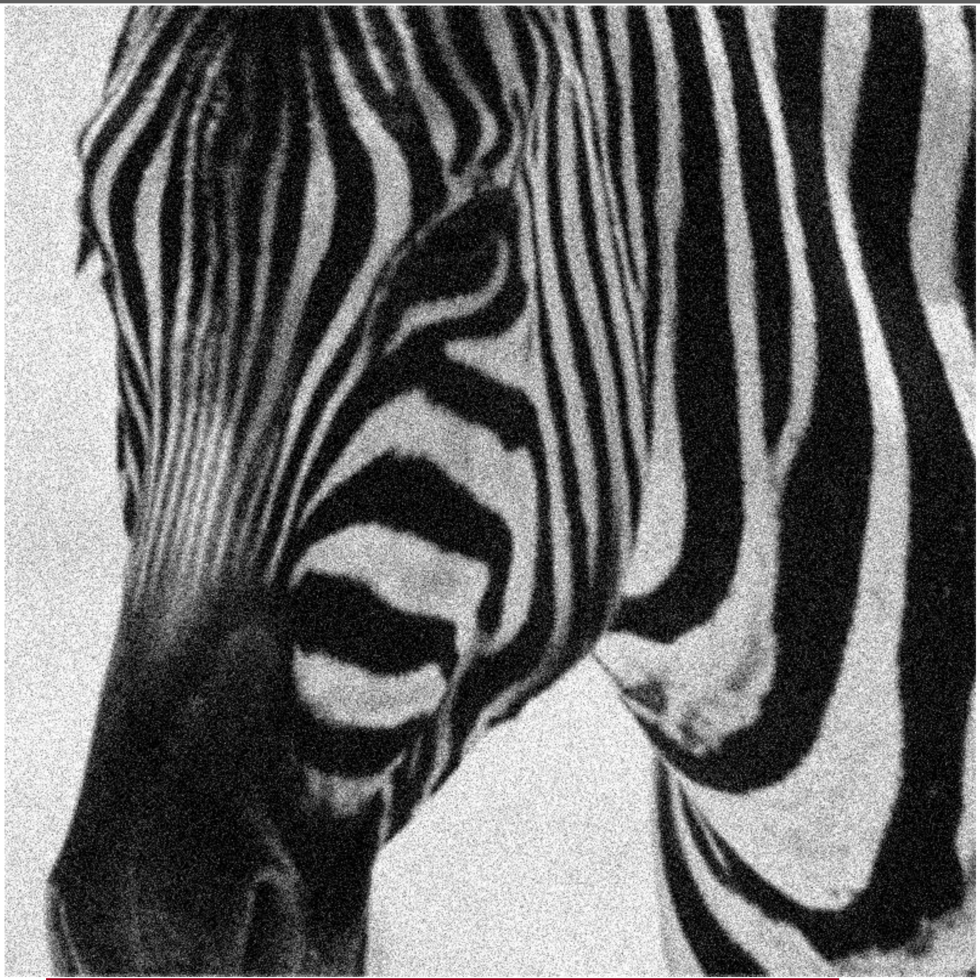
2. Explicit solution

$$w_i^{(k+1)} = \frac{1}{2}\left|K_i u^{(k)} - f_i\right|_{\gamma}^{-1}$$

3. Compute u by solving

$$\left[\mu_1 A^T C \hat{D}_\beta(u^{(k)}) C A + \mu_2 L^T (I - C) L + K^T D_\gamma(u^{(k)}) K\right] u^{(k+1)} = K^T D_\gamma(u^{(k)}) f$$

Example 1



corrupted image (SNR = 9.43)
Band=5, sigma=3 Noise 2%

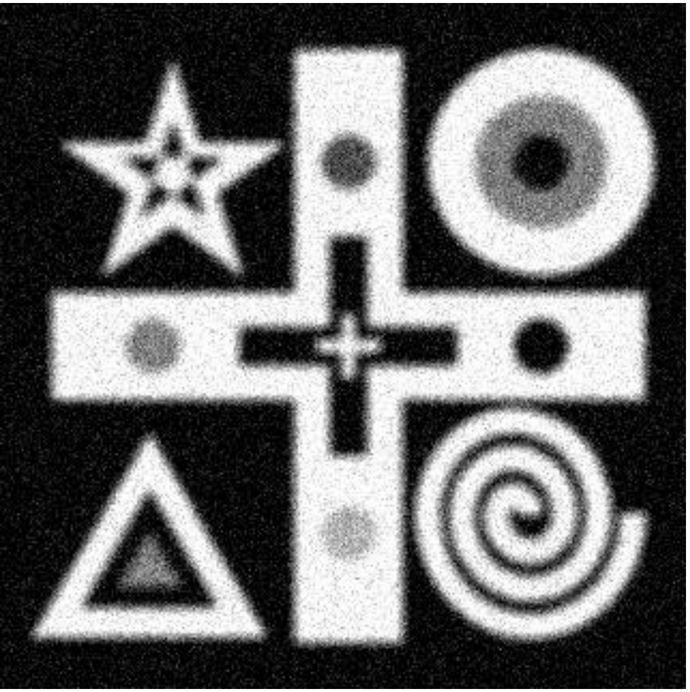
coherence map

Example 1

Restoration by L1-TV, p = 1,q = 1
μ= 0.5 (SNR = 17.15)

Restoration by AN p = 1,
μ_1= 0.5, μ_2 = 80 (SNR=17.47) k = 10

Example 2



Gaussian noise
band = 7, sigma = 5
Noise 2%

L2-TV , p = 2, q = 1
μ= 10
SNR = 10.68

AN , p = 1,
μ_1= 0.2, μ_2 = 10
SNR=16.76.

Example 2

band	sigma	%noise	SNR (L1-TV)	SNR (AN)
7	5	1%	22.09	22.90
7	5	2%	20.08	20.95
7	5	5%	16.74	17.61
7	5	10%	13.69	15.20
5	3	1%	23.63	24.53
5	3	2%	21.07	22.16
5	3	5%	18.02	18.76
5	3	10%	15.10	15.68
3	1	1%	26.35	26.78
3	1	2%	23.20	23.98
3	1	5%	18.69	19.38
3	1	10%	14.62	15.35

Conclusion

- Spatially-Adaptive Methods for image deblurring and denoising.
- Texture-preserving: the regularization operator is constructed by using fractional order derivatives
  - The choice of the fractional order for each pixel in the image is driven by the texture map of the image.
  - The regularization parameters are also chosen adaptively according to the texture map.
- Edge-preserving: norm adapted to the image features
- Simple iterative alternating algorithms to solve the models based on the half-quadratic strategy.

Thanks for your attention !