Algorithms for Noise Removal and the Bilateral Filter

Michael Elad

Scientific Computing and Computational Mathematics (SCCM) Program

Stanford University

May 6th, 2002

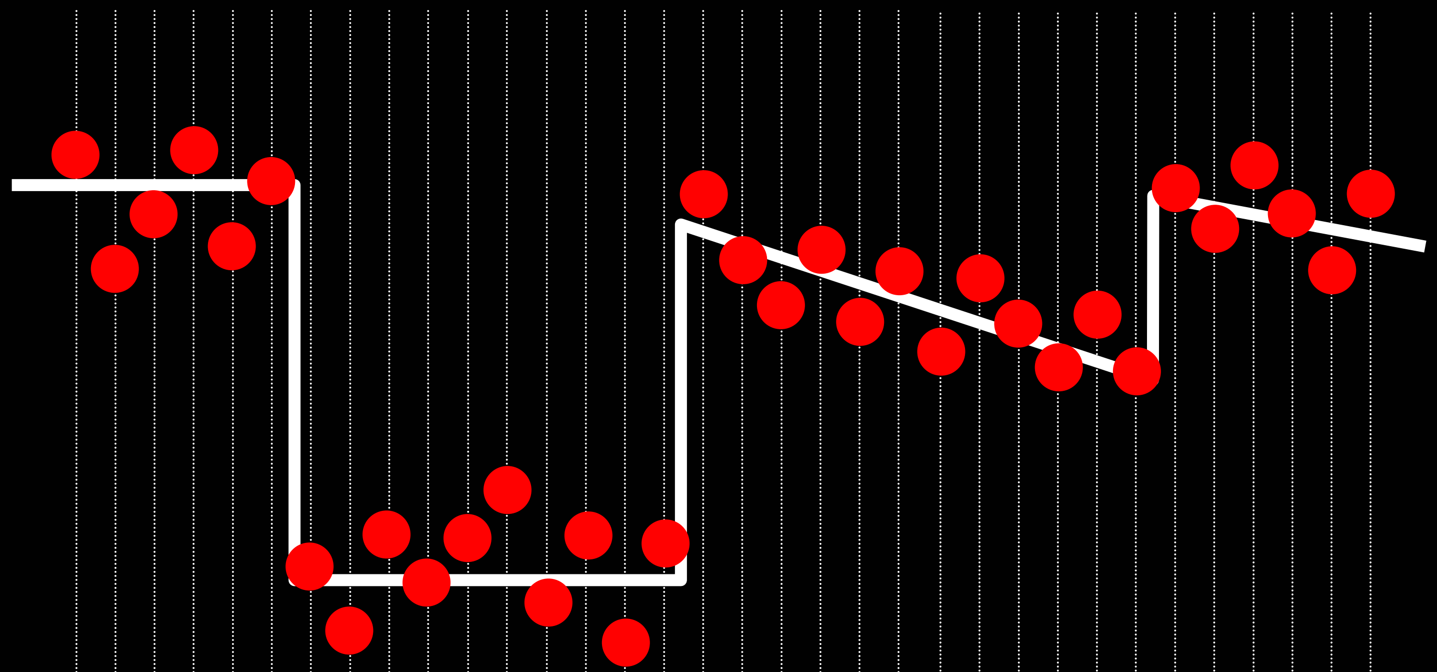
1.1 General

- This work deals with state-of-the-art algorithms for noise suppression.
- The basic model assumption is

$$\underline{Y}=\underline{X}+\underline{V}$$

where $\underline{X}$ – Unknown signal to be recovered,
$\underline{V}$ – Zero-mean white Gaussian noise,
$\underline{Y}$ – Measured signal.

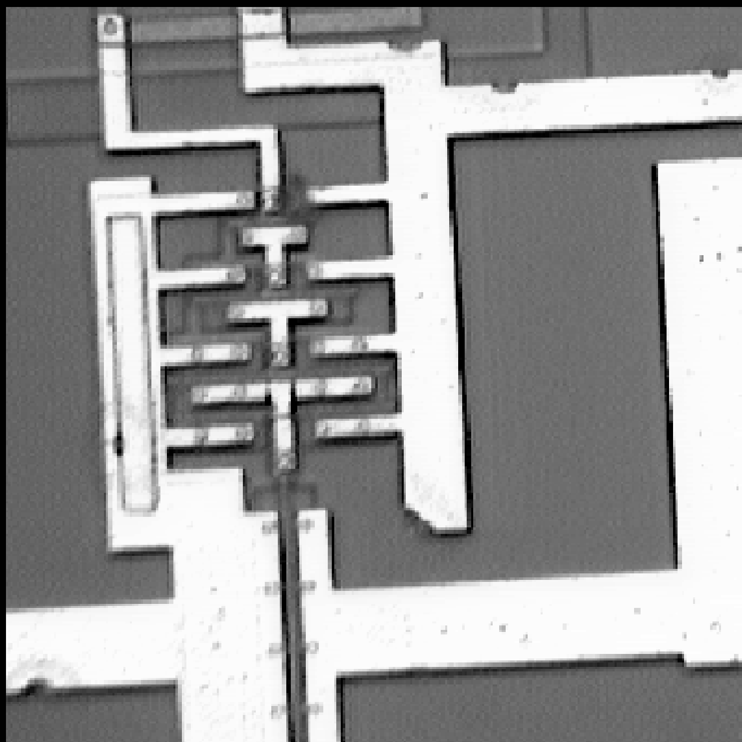
1.2 Graphically ...

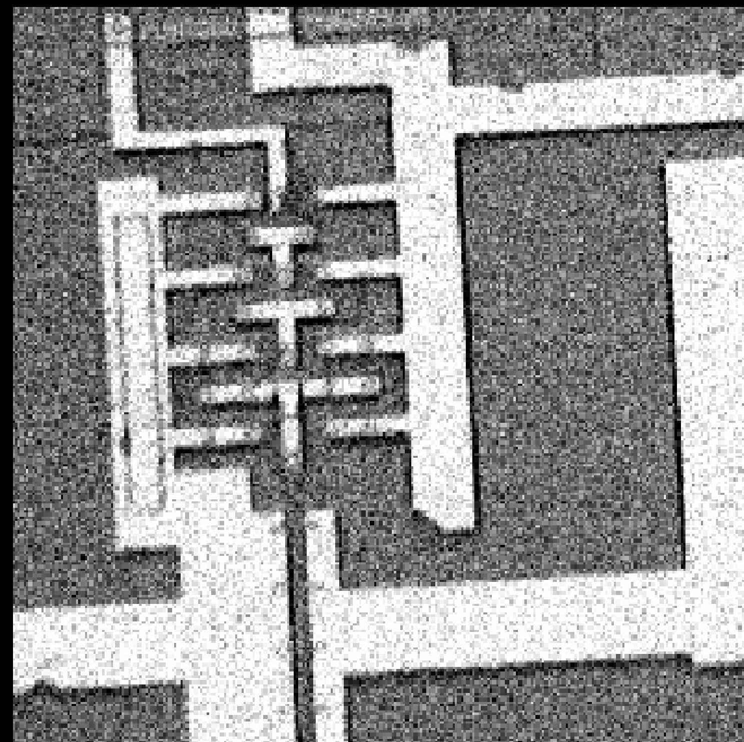


White - Ideal continuous signal

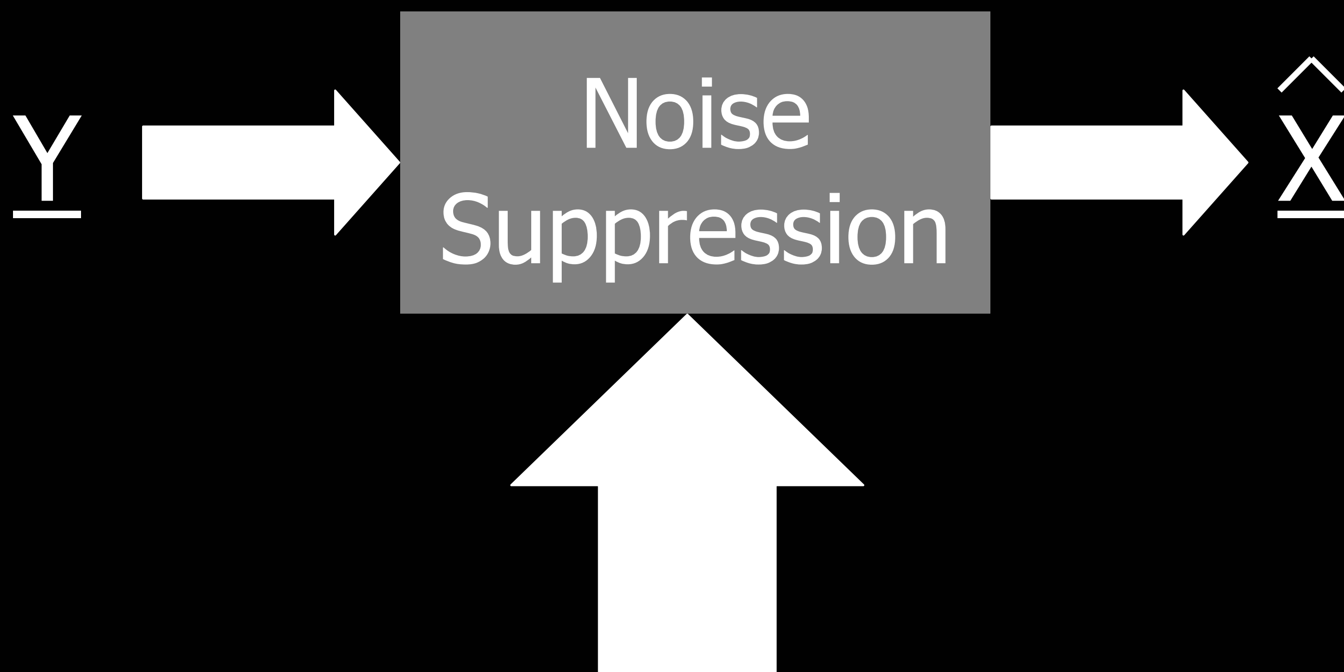
Red – Sampled (discrete) noisy signal

1.3 Example





1.4 Noise Suppression



Assumptions on the noise
and the desired signal $\underline{X}$

1.5 Background

- There are numerous methods to suppress noise,
- We are focusing on the family of methods based on
  - Piece-wise smoothness assumption
  - Maximum A-posteriori Probability Estimation (Bayesian)
- State-of-the-art methods from this family:
  - WLS - Weighted Least Squares,
  - RE - Robust Estimator,
  - AD - Anisotropic diffusion,
  - Bilateral filter

1.6 In Focus – Bilateral Filter

- The bilateral filter was originally proposed by Tomasi and Manduchi in 1998 (ICCV) as a heuristic tool for noise removal,
- A similar filter (Digital-TV) was proposed by Chan, Osher and Chen in February 2001 (IEEE Trans. On Image Proc.),
- In this talk we:
  - Analyze the bilateral filter and relate it to the WLS/RE/AD algorithms,
  - Demonstrate its behavior,
  - Suggest further improvements to this filter.

Chapter 2

The WLS, RE and AD Filters

2.1 MAP Based Filters

- We would like the filter to produce a signal that
  - Is close to the measured signal,
  - Is smooth function, and
  - Preserves edges.
- Using Maximum-Aposteriori-Probability formulation, we can write a penalty function which, when minimized, results with the signal we desire.
- Main Question: How to formulate the above requirements?

2.2 Least Squares

$$\varepsilon_{LS}\{\underline{X}\} = \frac{1}{2}\left[\underline{X}-\underline{Y}\right]^T\left[\underline{X}-\underline{Y}\right] + \frac{\lambda}{2}\left[\underline{X}-\mathbf{D}\underline{X}\right]^T\left[\underline{X}-\mathbf{D}\underline{X}\right]$$

Proximity to the measurements

Spatial smoothness

D - A one-sample shift operator. Thus, ($\underline{X}$-**D**$\underline{X}$) is merely a discrete one-sided derivative.

$$\mathbf{D} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

2.3 Weighted Least Squares

$$\varepsilon_{\mathrm{WLS}}\left\{\underline{X}\right\} = \frac{1}{2}\left[\underline{X}-\underline{Y}\right]^T\left[\underline{X}-\underline{Y}\right] + \frac{\lambda}{2}\left[\underline{X}-\mathbf{D}\underline{X}\right]^T \mathbf{W}(\underline{Y})\left[\underline{X}-\mathbf{D}\underline{X}\right]$$

Proximity to the measurements

Spatially smooth

Edge Preserving by diagonal weight matrix

Based on Y:

Samples belonging to smooth regions are assigned with large weight ($\rightarrow$1).

Samples suspected of being edge points are assigned with low weight ($\rightarrow$0).

2.4 WLS Solution

The penalty derivative:

$$\frac{\partial \varepsilon_{WLS}\{\underline{X}\}}{\partial \underline{X}} = \left[\underline{X} - \underline{Y}\right] + \lambda\left[\mathbf{I} - \mathbf{D}\right]^T \mathbf{W}(\underline{Y})\left[\mathbf{I} - \mathbf{D}\right]\underline{X}$$

A single SD Iteration with $\underline{Y}$ as initialization gives:

$$\underline{\hat{X}}_1^{WLS} = \underline{\hat{X}}_0^{WLS} - \mu \left.\frac{\partial \varepsilon_{WLS}\{\underline{X}\}}{\partial \underline{X}}\right|_{\underline{X}=\underline{\hat{X}}_0^{WLS}}$$

$$= \underline{Y} - \mu\lambda\left(\mathbf{I} - \mathbf{D}\right)^T \mathbf{W}(\underline{Y})\left(\mathbf{I} - \mathbf{D}\right)\underline{Y}$$

What about updating the weights after each iteration ?

2.5 Robust Estimation

$$\varepsilon_{\mathrm{RE}}\{\underline{X}\} = \frac{1}{2}\left[\underline{X}-\underline{Y}\right]^{\mathrm{T}}\left[\underline{X}-\underline{Y}\right] + \frac{\lambda}{2}\rho\{\underline{X}-\mathbf{D}\underline{X}\}$$

Proximity to the measurements

Spatially smooth and edge preserving

$\rho(\alpha)$ - A symmetric non-negative function, e.g. $\rho(\alpha)=\alpha^2$ or $\rho(\alpha)=|\alpha|$, etc.

2.6 RE Solution

The penalty derivative:

$$\frac{\partial \varepsilon_{\mathrm{RE}}\left\{\underline{\mathrm{X}}\right\}}{\partial \underline{\mathrm{X}}} = \left[\underline{\mathrm{X}} - \underline{\mathrm{Y}}\right] + \lambda\left[\mathbf{I} - \mathbf{D}\right]^{\mathrm{T}} \rho'\left\{\underline{\mathrm{X}} - \mathbf{D}\underline{\mathrm{X}}\right\}$$

A single SD Iteration with $\underline{\mathrm{Y}}$ as initialization gives:

$$\hat{\underline{\mathrm{X}}}_1^{\mathrm{RE}} = \hat{\underline{\mathrm{X}}}_0^{\mathrm{RE}} - \mu \left.\frac{\partial \varepsilon_{\mathrm{RE}}\left\{\underline{\mathrm{X}}\right\}}{\partial \underline{\mathrm{X}}}\right|_{\underline{\mathrm{X}} = \hat{\underline{\mathrm{X}}}_0^{\mathrm{RE}}}$$

$$= \underline{\mathrm{Y}} - \mu\lambda\left(\mathbf{I} - \mathbf{D}\right)^{\mathrm{T}} \rho'\left\{\left(\mathbf{I} - \mathbf{D}\right)\underline{\mathrm{Y}}\right\}$$

2.7 WLS and RE Equivalence

$$\hat{\underline{X}}_1^{RE} = \underline{Y} - \mu\lambda\left(\mathbf{I}-\mathbf{D}\right)^T \rho'\left\{\left(\mathbf{I}-\mathbf{D}\right)\underline{Y}\right\}$$

$$\hat{\underline{X}}_1^{WLS} = \underline{Y} - \mu\lambda\left(\mathbf{I}-\mathbf{D}\right)^T \mathbf{W}(\underline{Y})\left(\mathbf{I}-\mathbf{D}\right)\underline{Y}$$

For equivalence, require

$$\forall\underline{Y},\ \mathbf{W}(\underline{Y})\left(\mathbf{I}-\mathbf{D}\right)\underline{Y} = \rho'\left\{\left(\mathbf{I}-\mathbf{D}\right)\underline{Y}\right\}$$

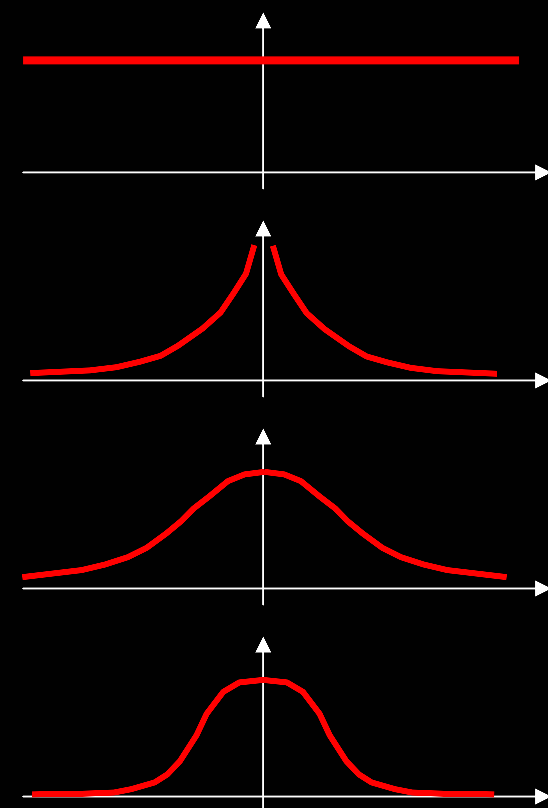
$$\Rightarrow\ \mathbf{W}(\underline{Y}) = \frac{\rho'\left\{\left(\mathbf{I}-\mathbf{D}\right)\underline{Y}\right\}}{\left(\mathbf{I}-\mathbf{D}\right)\underline{Y}}$$

$$W[k] = \frac{\rho'\left\{Y[k]-Y[k-1]\right\}}{Y[k]-Y[k-1]}$$

2.8 WLS and RE Examples

$\rho(\alpha)$	$\rho'(\alpha)$	$w(\alpha)$
$\frac{1}{2}\alpha^2$	α	1
$\lvert\alpha\rvert$	$\mathrm{sign}(\alpha)$	$\frac{\mathrm{sign}(\alpha)}{\alpha}$
$\sqrt{\alpha^2+\varepsilon^2}$	$\alpha / \sqrt{\alpha^2+\varepsilon^2}$	$1 / \sqrt{\alpha^2+\varepsilon^2}$
$\alpha^2 / \left(\alpha^2+\varepsilon^2\right)$	$2\alpha\varepsilon^2 / \left(\alpha^2+\varepsilon^2\right)^2$	$2\varepsilon^2 / \left(\alpha^2+\varepsilon^2\right)^2$

The weight as a function of the derivative



2.9 RE as a Bootstrap-WLS

$$\underline{X}_{k+1}^{WLS} = \underline{X}_k^{WLS} - \mu\left[\left(\underline{X}_k^{WLS} - \underline{Y}\right) + \lambda\left(\mathbf{I} - \mathbf{D}\right)^T \mathbf{W}\left\{\left(\mathbf{I} - \mathbf{D}\right)\underline{X}_k^{WLS}\right\}\right]$$

$$\underline{X}_{k+1}^{RE} = \underline{X}_k^{RE} - \mu\left[\left(\underline{X}_k^{RE} - \underline{Y}\right) + \lambda\left(\mathbf{I} - \mathbf{D}\right)^T \boxed{\rho'\left\{\left(\mathbf{I} - \mathbf{D}\right)\underline{X}_k^{RE}\right\}}\right]$$

$$\boxed{\rho'\left\{\left(\mathbf{I} - \mathbf{D}\right)\underline{X}_k^{RE}\right\}} = \mathbf{W}\left\{\underline{X}_k^{RE}\right\}\left(\mathbf{I} - \mathbf{D}\right)\underline{X}_k^{RE}$$

This way the RE actually applies an update of the weights after each iteration

2.10 Anisotropic Diffusion

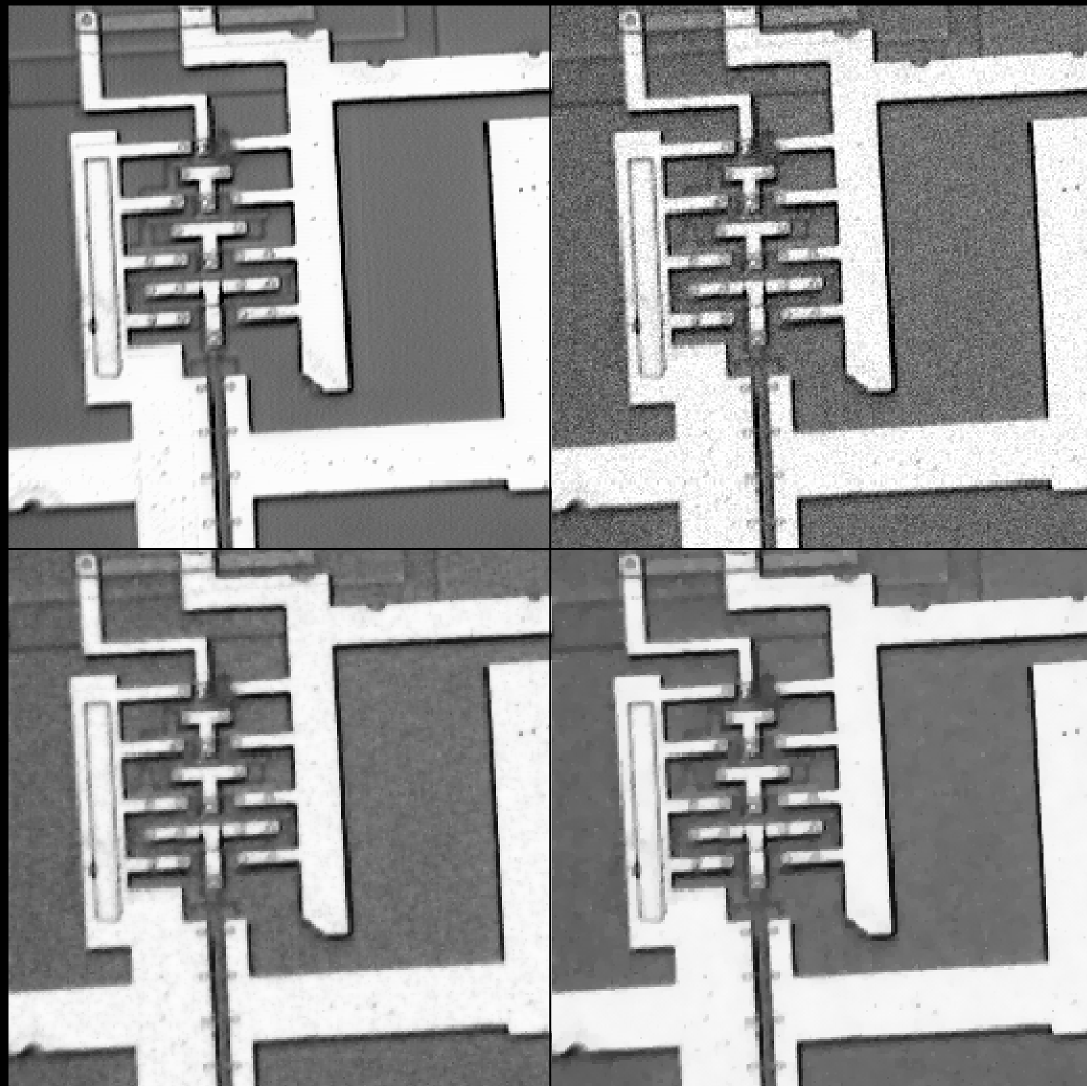
- Anisotropic diffusion filter was presented originally by Perona & Malik on 1987
- The proposed filter is formulated as a Partial Differential Equation,

$$\partial_t X = -\nabla\left\{g\left(\left|\nabla X\right|^2\right)\cdot\nabla X\right\}$$

- When discretized, the AD turns out to be the same as the Robust Estimation and the line-process techniques (see – Black and Rangarajan – 96` and Black and Sapiro – 99’).

2.11 Example

Original image



Noisy image

Var=15

WLS – 100 Iterations

RE – 100 Iterations

Chapter 3

The Bilateral Filter

3.1 General Idea

- Every sample is replaced by a weighted average of its neighbors (as in the WLS),
- These weights reflect two forces
  - How close are the neighbor and the center sample, so that larger weight to closer samples,
  - How similar are the neighbor and the center sample – larger weight to similar samples.
- All the weights should be normalized to preserve the local mean.

3.2 In an Equation

Averaging over the 2N+1 neighborhood

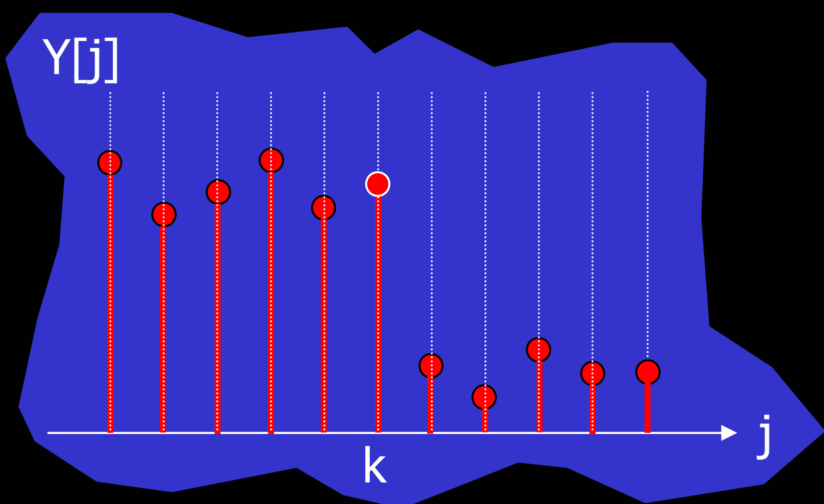
The weight

The neighbor sample

The result at the k^{th} sample

$$\hat{X}[k] = \frac{\sum_{n=-N}^{N} W[k,n]\, Y[k-n]}{\sum_{n=-N}^{N} W[k,n]}$$

Normalization of the weighting



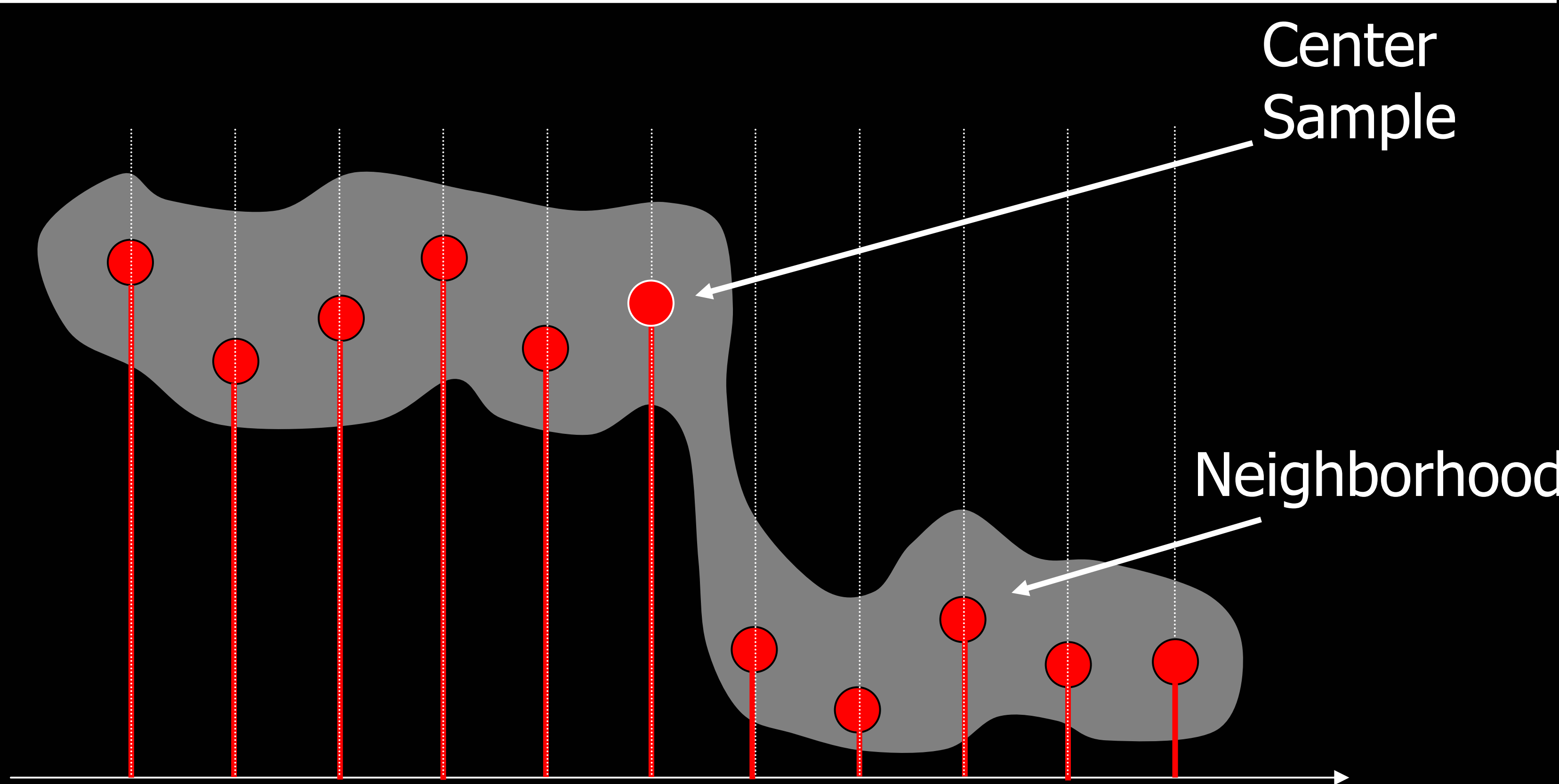
3.3 The Weights

$$W_S[k,n] = \exp\left\{ -\frac{d_S^2\left\{[k],[k-n]\right\}}{2\sigma_S^2} \right\} = \exp\left\{ -\frac{n^2}{2\sigma_S^2} \right\}$$

$$W_R[k,n] = \exp\left\{ -\frac{d_R^2\left\{Y[k],Y[k-n]\right\}}{2\sigma_R^2} \right\} = \exp\left\{ -\frac{\left[Y[k]-Y[k-n]\right]^2}{2\sigma_R^2} \right\}$$

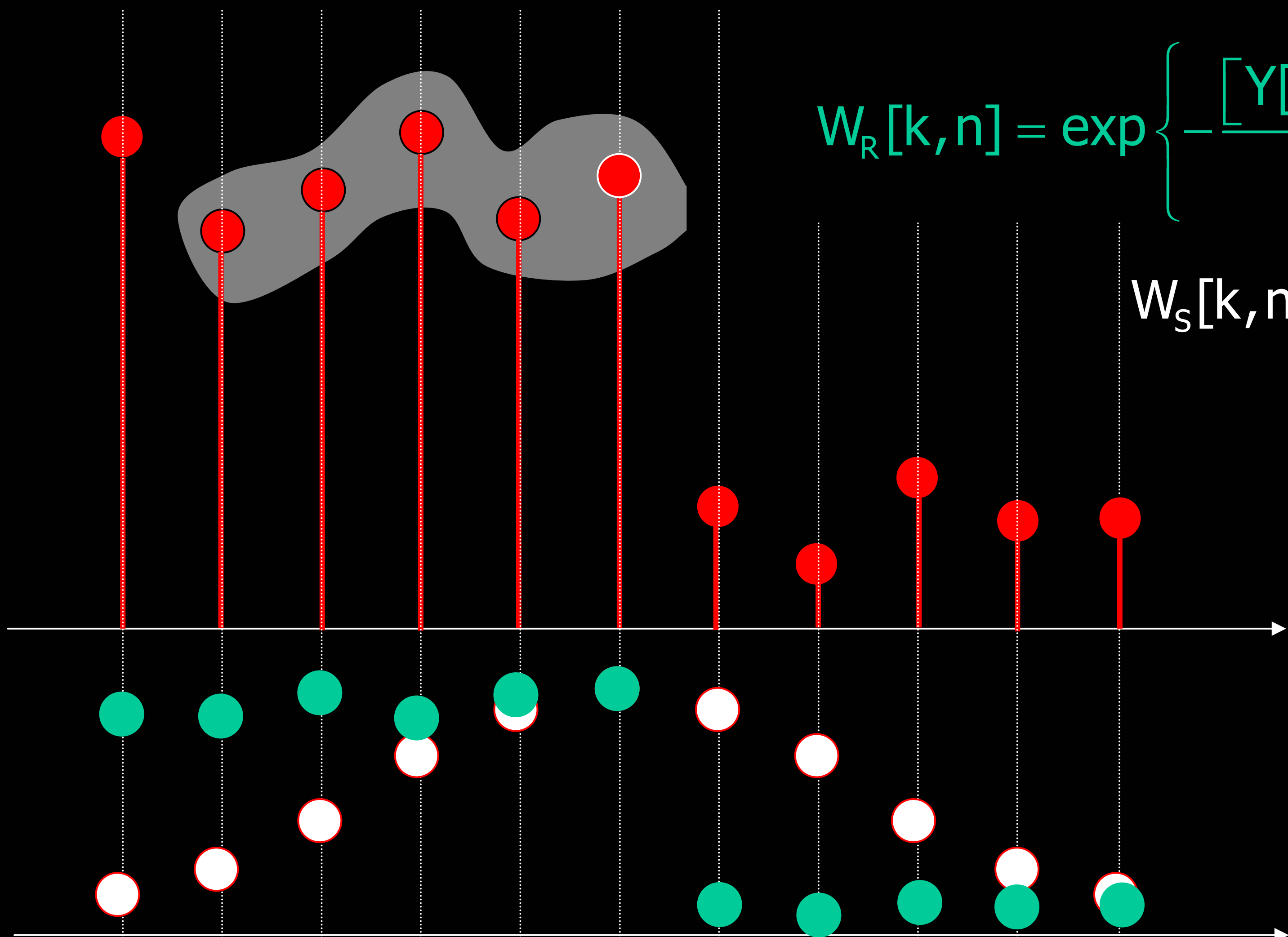
$$W[k,n] = W_S[k,n] \cdot W_R[k,n]$$

3.4 Graphical Example



It is clear that in weighting this neighborhood, we would like to preserve the step

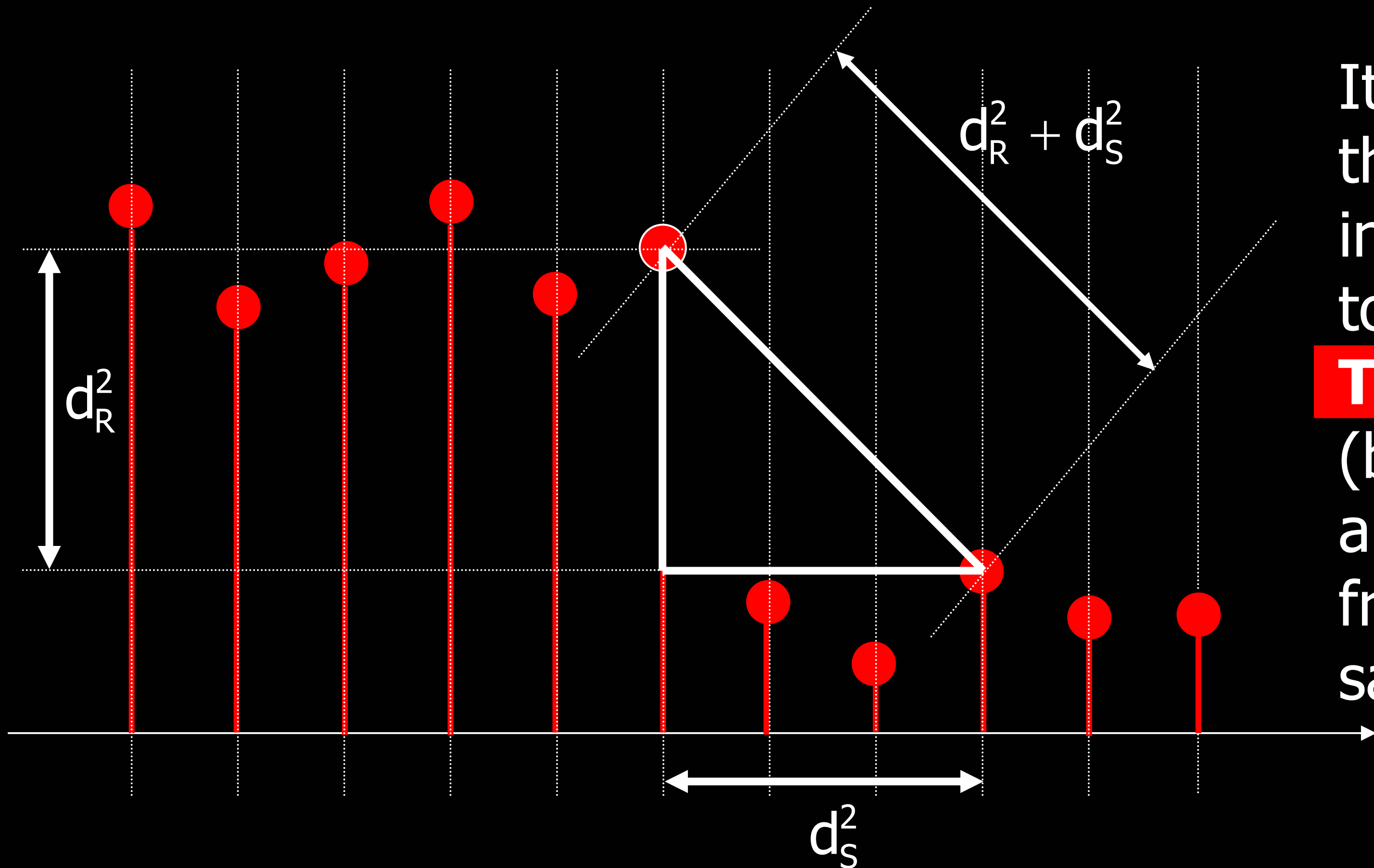
3.5 The Weights



$$W_R[k,n]=\exp\left\{-\frac{\left[Y[k]-Y[k-n]\right]^2}{2\sigma_R^2}\right\}$$

$$W_S[k,n]=\exp\left\{-\frac{n^2}{2\sigma_S^2}\right\}$$

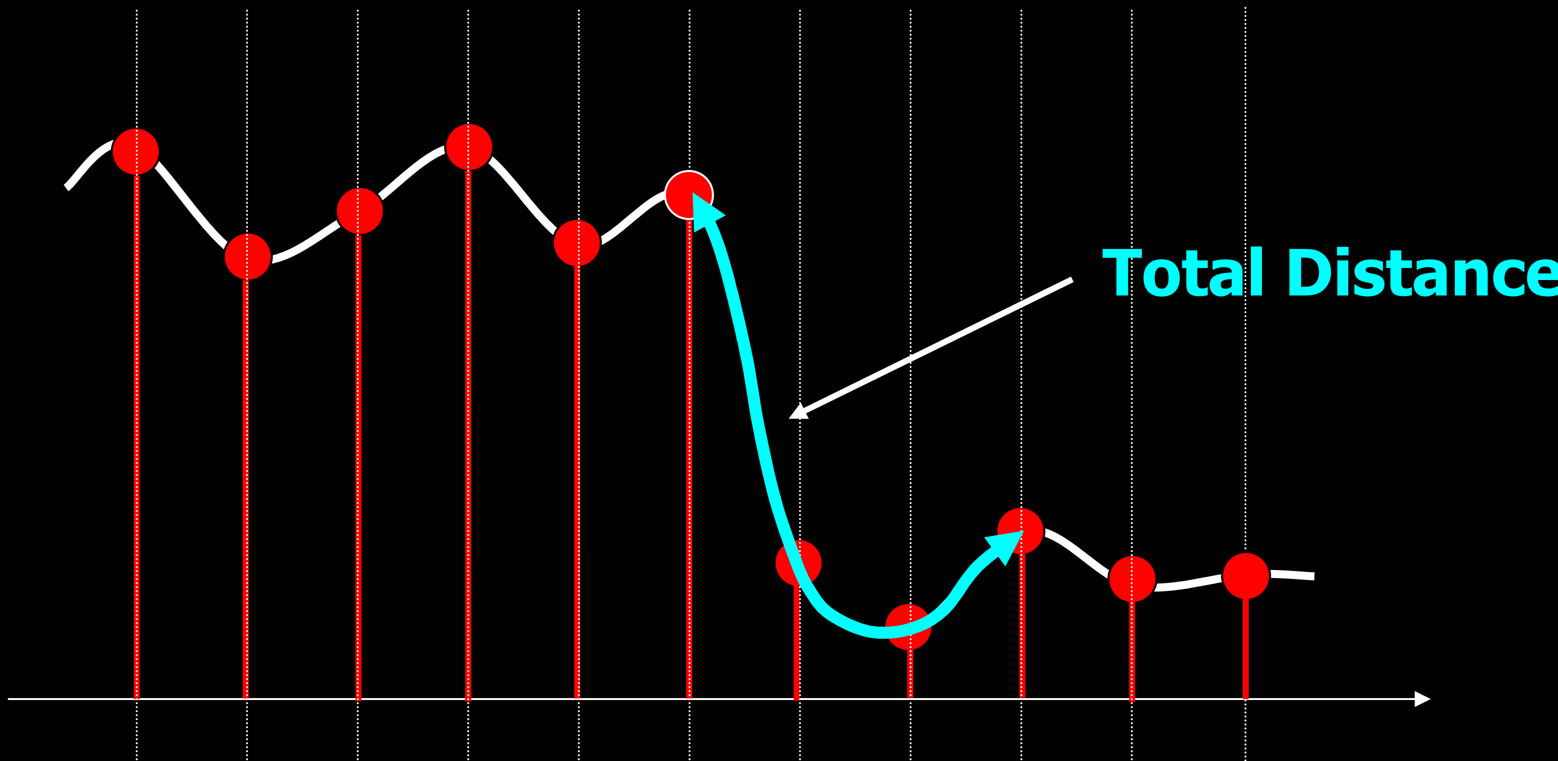
3.6 Total-Distance



It appears that the weight is inversely prop. to the **Total-Distance** (both horizontal and vertical) from the center sample.

$$W[k,n] = \exp\left\{-\frac{\sigma_R^2 d_S^2\{[k],[k-n]\} + \sigma_S^2 d_R^2\{Y[k],Y[k-n]\}}{2\sigma_S^2\sigma_R^2}\right\}$$

3.7 Discrete Beltrami Flow?



This idea is similar in spirit to the ‘Beltrami Flow’ proposed by Sochen, Kimmel and Maladi (1998). There, the effective weight is the ‘Geodesic Distance’ between the samples.

3.8 Kernel Properties

- Per each sample, we can define a 'Kernel' that averages its neighborhood

$$\frac{\left[W[k,-N],....W[k,-1],\ W[k,0],W[k,+1],W[k,+N]\right]}{\sum_{n=-N}^{N} W[k,n]}$$

- This kernel changes from sample to sample!
- The sum of the kernel entries is 1 due to the normalization,
- The center entry in the kernel is the largest,
- Subject to the above, the kernel can take any form (as opposed to filters which are monotonically decreasing).

3.9 Filter Parameters

As proposed by Tomasi and Manduchi, the filter is controlled by 3 parameters:

N – The size of the filter support,

σ_S – The variance of the spatial distances,

σ_R – The variance of the spatial distances, and

It – The filter can be applied for several iterations in order to further strengthen its edge-preserving smoothing.

3.10 Additional Comments

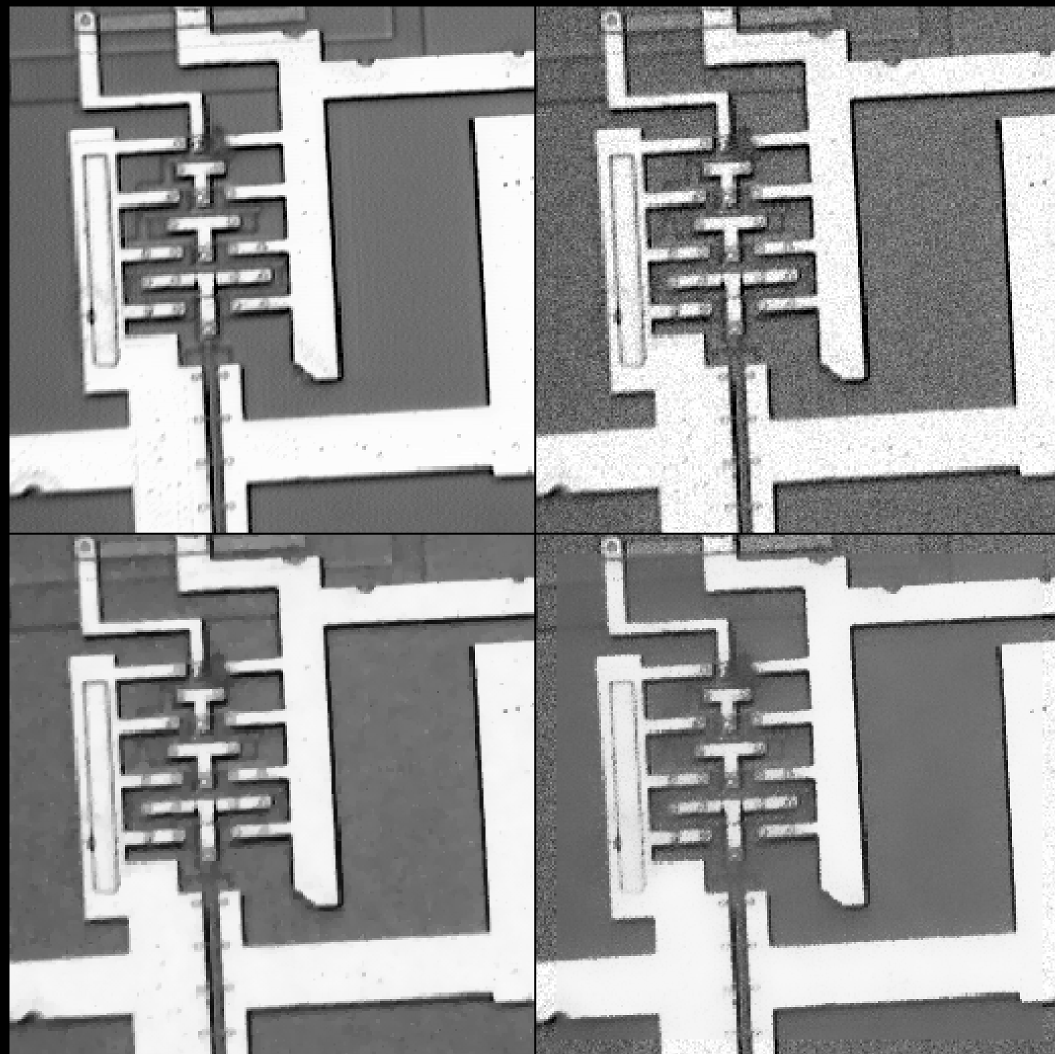
The bilateral filter is a powerful filter:

- One application of it gives the effect of numerous iterations using traditional local filters,
- Can work with any reasonable distances d_s and d_R definitions,
- Easily extended to higher dimension signals, e.g. Images, video, etc.
- Easily extended to vectored-signals, e.g. Color images, etc.

3.11 Example

Original image

Noisy image Var=15



RE – 100 Iterations

Bilateral (N=10, …)

3.12 To Summarize

Feature	Bilateral filter	WLS/RE/AD
Behavior	Edge preserve	Edge preserve
Support size	May be large	Very small
Iterations	Possible	Must
Origin	Heuristic	MAP-based

What is the connection between the bilateral and the WLS/RE/AD filters ?

Chapter 4

The Bilateral Filter Origin

4.1 General Idea

In what follows we shall show that:

- We can propose a novel penalty function $\varepsilon\{\underline{X}\}$, extending the ones presented before,
- The bilateral filter emerges as a single Jacobi iteration minimizing $\varepsilon\{\underline{X}\}$, if Y is used for initialization,
- We can get the WLS/RE filters as special cases of this more general formulation.

4.2 New Penalty Function

Proximity to the measurements

Spatially smooth and edge preservation

$$\varepsilon\{\underline{X}\} = \frac{1}{2}\left[\underline{X}-\underline{Y}\right]^T\left[\underline{X}-\underline{Y}\right]+$$

$$+\frac{\lambda}{2}\sum_{n=1}^{N}\left[\underline{X}-\mathbf{D}^n\underline{X}\right]^T\mathbf{W}(\underline{Y},n)\boxed{\left[\underline{X}-\mathbf{D}^n\underline{X}\right]}$$

X[k]-x[k-n]

4.3 Penalty Minimization

$$\frac{\partial \varepsilon\{\underline{X}\}}{\partial \underline{X}} = \left[\mathbf{I} + \lambda \sum_{n=1}^{N} \left(\mathbf{I} - \mathbf{D}^{n}\right)^{T} \mathbf{W}(\underline{Y}, n)\left(\mathbf{I} - \mathbf{D}^{n}\right)\right]\underline{X} - \underline{Y}$$

A single Steepest-Descent iteration with $\underline{Y}$ as initialization gives

$$\hat{\underline{X}}_1 = \left[\mathbf{I} - \mu\lambda \sum_{n=1}^{N} \left(\mathbf{I} - \mathbf{D}^{-n}\right) \mathbf{W}(\underline{Y}, n)\left(\mathbf{I} - \mathbf{D}^{n}\right)\right]\underline{Y}$$

4.4 Algorithm Speed-Up

Instead the SD, we use a Jacobi iteration, where μ is replaced by the inverse of the Hessian Matrix main diagonal

$$\frac{\partial^2 \varepsilon}{\partial \underline{X}^2} = \mathbf{H}(\underline{Y}) = \left[\mathbf{I} + \lambda \sum_{n=1}^{N} \left(\mathbf{I} - \mathbf{D}^{-n} \right) \mathbf{W}(\underline{Y}, n) \left(\mathbf{I} - \mathbf{D}^{n} \right) \right]$$

$$\Rightarrow \mathbf{M}(\underline{Y}) = \left[\mathrm{diag}\{\mathbf{H}(\underline{Y})\} + \boxed{\xi \mathbf{I}} \right]^{-1}$$

Relaxation

$$\hat{\underline{X}}_1 = \left[\mathbf{I} - \lambda \mathbf{M}(\underline{Y}) \sum_{n=1}^{N} \left(\mathbf{I} - \mathbf{D}^{-n} \right) \mathbf{W}(\underline{Y}, n) \left(\mathbf{I} - \mathbf{D}^{n} \right) \right] \underline{Y}$$

4.5 Choice of Weights

Let us choose the weights in the diagonal matrix $\mathbf{W}(\underline{Y},n)$ as

$$\mathbf{W}(\underline{Y},n)=\frac{\rho'\left\{\left(\mathbf{I}-\mathbf{D}^{n}\right)\underline{Y}\right\}}{\left(\mathbf{I}-\mathbf{D}^{n}\right)\underline{Y}}\cdot V(n)$$

Where: $\rho(x)$ - Some robust function (non-negative, symmetric penalty function, e.g. $\rho(x)=|x|$.

$V(n)$ - Symmetric, monotonically decreasing weight, e.g. $V(n)=\alpha^{|n|}$, where $0<\alpha<1$.

4.6 The Obtained Filter

$$\hat{\underline{X}}_1 = \boxed{\left[\mathbf{I} - \lambda \mathbf{M}(\underline{Y}) \sum_{n=1}^{N} \left(\mathbf{I} - \mathbf{D}^{-n}\right) \mathbf{W}(\underline{Y}, n) \left(\mathbf{I} - \mathbf{D}^{n}\right)\right]} \underline{Y}$$

This entire operation can be viewed as a weighted average of samples in Y, where the weights themselves are dependent on Y !

We can write

$$\hat{X}_1[k] = \sum_{n=-N}^{N} f\left[\ell, k\right] \cdot Y\left[k - \ell\right]$$

4.7 The Filter Coefficients

$$
f[\ell,k]=\begin{cases}
\dfrac{\lambda V(\ell)\cdot\dfrac{\rho'\{Y[k]-Y[k-\ell]\}}{\left(Y[k]-Y[k-\ell]\right)}}{\xi+1+\lambda\sum\limits_{n=-N}^{N}V(n)\cdot\dfrac{\rho'\{Y[k]-Y[k-n]\}}{Y[k]-Y[k-n]}} & -\left[\begin{array}{c}N\le\ell\le N,\\ \ell\neq 0\end{array}\right]\\[2em]
\dfrac{\xi+1}{\xi+1+\lambda\sum\limits_{n=-N}^{N}V(n)\cdot\dfrac{\rho'\{Y[k]-Y[k-n]\}}{Y[k]-Y[k-n]}} & \left[\ell=0\right].
\end{cases}
$$

4.8 The Filter Properties

- If we choose

$$\rho(\alpha)=\sigma_R^2\left[1-\exp\left\{-\frac{\alpha^2}{2\sigma_R^2}\right\}\right],\quad V(\ell)=\exp\left\{-\frac{\ell^2}{2\sigma_S^2}\right\}$$

  we get an exact equivalence to the bilateral filter.

- The values of ξ and λ enable a control over the uniformity of the filter kernel.
- The sum of all the coefficients is 1, and all are non-negative.

4.9 To Recap

A new penalty function was defined

We used a single Jacobi iteration

We assumed a specific weight form

We got the bilateral filter

Chapter 5

Improving The Bilateral Filter

No time ? press HERE

5.1 What can be Achieved?

- Why one iteration? We can apply several iterations of the Jacobi algorithm.
- Speed-up the algorithm effect by a Gauss-Siedel (GS) behavior.
- Speed-up the algorithm effect by updating the output using sub-gradients.
- Extend to treat piece-wise linear signals by referring to 2^{nd} derivatives.

5.2 GS Acceleration

For a function of the form: $\varepsilon\{\underline{X}\} = \frac{1}{2}\underline{X}^T \mathbf{Q}\underline{X} - \underline{P}^T\underline{X} + C$

The SD iteration: $\hat{\underline{X}}_1 = \hat{\underline{X}}_0 + \mu\left(\underline{P} - \mathbf{Q}\hat{\underline{X}}_0\right)$

The Jacobi iteration: $\hat{\underline{X}}_1 = \hat{\underline{X}}_0 + \left(I + \mu \mathrm{diag}\{\mathbf{Q}\}\right)^{-1}\left(\underline{P} - \mathbf{Q}\hat{\underline{X}}_0\right)$

The GS iteration: $\hat{\underline{X}}_1 = \hat{\underline{X}}_0 + \left(I + \mu \cdot \mathrm{updiag}\{\mathbf{Q}\}\right)^{-1}\left(\underline{P} - \mathbf{Q}\hat{\underline{X}}_0\right)$

The GS intuition – Compute the output sample by sample, and use the already computed values whenever possible.

5.3 Sub-Gradients

The function we have has the form

$$\varepsilon\{\underline{X}\} = \sum_{j=1}^{J}\left[\frac{1}{2}\underline{X}^T Q_j \underline{X} - \underline{P}_j^T \underline{X} + C_j\right]$$

One SD iteration:

$$\underline{\hat{X}}_1 = \underline{\hat{X}}_0 - \mu \sum_{j=1}^{J}\left[Q_j \underline{\hat{X}}_0 - \underline{P}_j\right]$$

$$\underline{\hat{X}}_1 = \underline{\hat{X}}_0 - \mu\left[Q_1 \underline{\hat{X}}_0 - \underline{P}_1\right]$$

$$\underline{\hat{X}}_2 = \underline{\hat{X}}_1 - \mu\left[Q_2 \underline{\hat{X}}_1 - \underline{P}_2\right]$$

$$\vdots$$

$$\underline{\hat{X}}_J = \underline{\hat{X}}_{J-1} - \mu\left[Q_J \underline{\hat{X}}_{J-1} - \underline{P}_J\right]$$

5.4 Piecewise Linear Signals

Similar penalty term using 2nd derivatives for the smoothness term

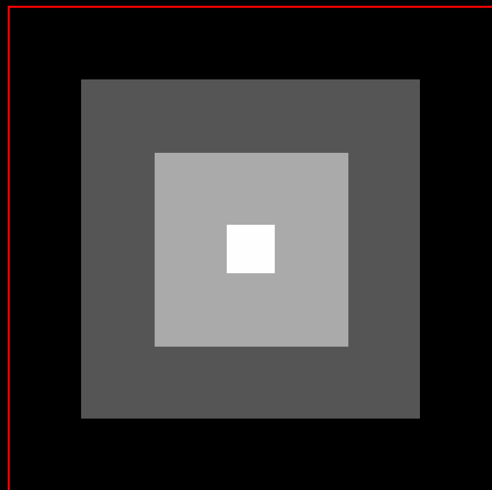
$$\varepsilon\{\underline{X}\} = \frac{1}{2}\|\underline{X}-\underline{Y}\|^2 + \frac{\lambda}{2}\sum_{n=1}^{N}\left[\underline{X}-\frac{\mathbf{D}^n\underline{X}+\mathbf{D}^{-n}\underline{X}}{2}\right]^T \mathbf{W}(\underline{Y},n)\left[\underline{X}-\frac{\mathbf{D}^n\underline{X}+\mathbf{D}^{-n}\underline{X}}{2}\right]$$

This way we do not penalize linear signals !

Chapter 6

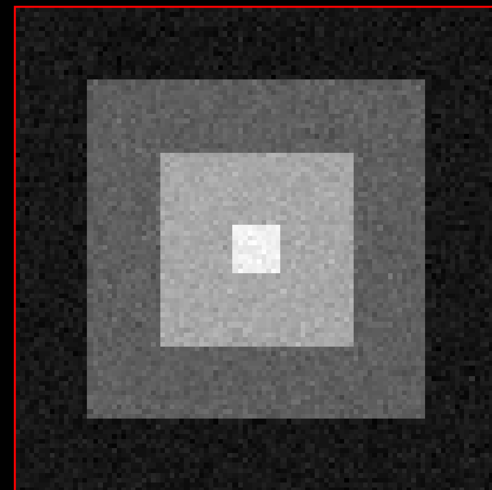
Results

6.1 General Comparison



Original image

Values in the range [1,7]



Noisy image

Gaussian Noise - σ=0.2

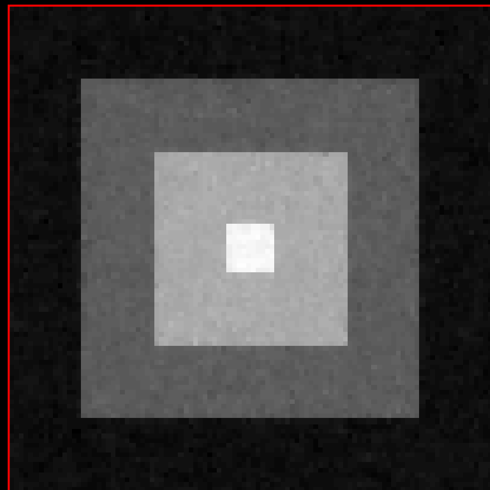
$$\text{Noise Gain} = \frac{\text{Mean-Squared-Error before the filter}}{\text{Mean-Squared-Error after the filter}}$$

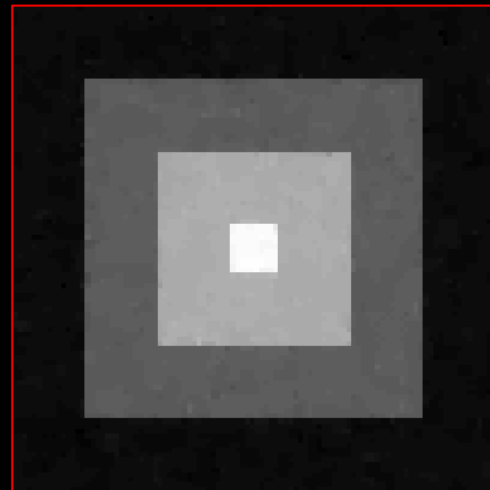
6.2 Results

WLS

50 iterations

Gain: 3.90





RE ($\rho(\alpha) = |\alpha|$)

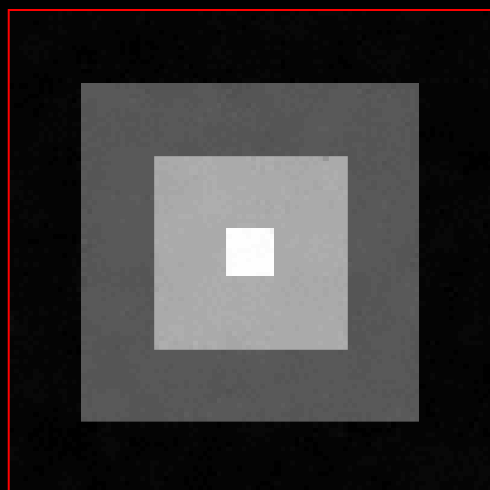
50 iterations

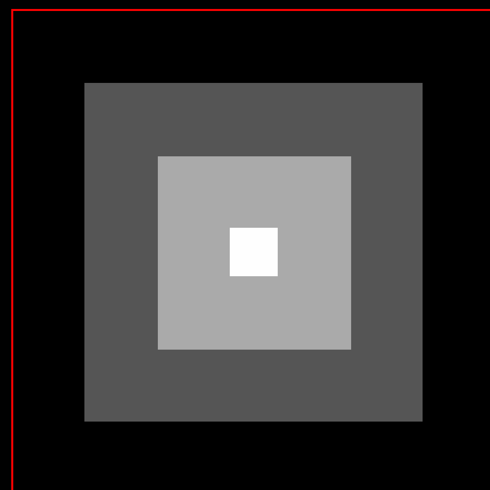
Gain: 10.99

Bilateral (N=6)

1 iteration

Gain: 23.50





Bilateral

10 iterations

Gain: 318.90

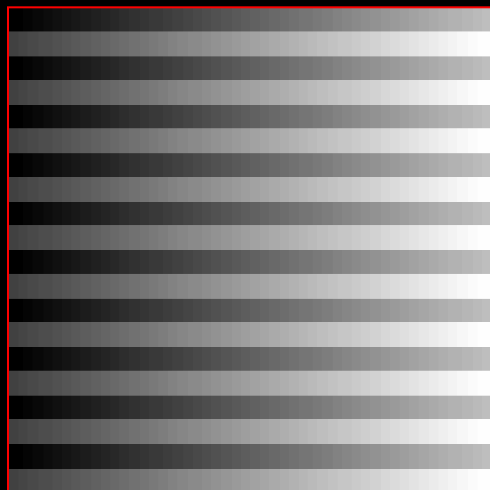
6.3 Speedup Results

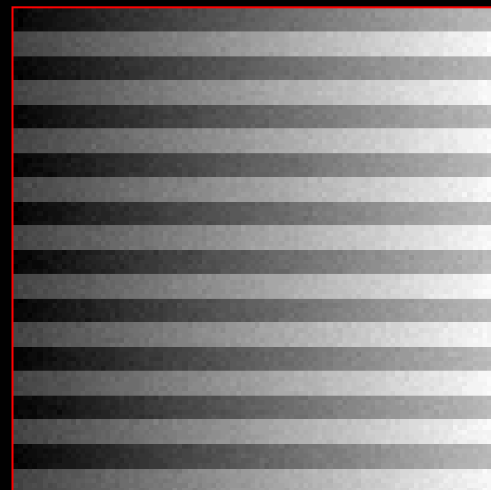
- Regular bilateral filter gave Gain=23.50.
- Using the Gauss-Siedel version of the filter we got Gain=39.44.
- Using the sub-gradient approach we got Gain=197.26! The filter size is 13 by 13, which means that we have 169 sub-iterations instead of a single large one.

6.3 Piecewise linear Image

Original image

Values in the range [0,16]



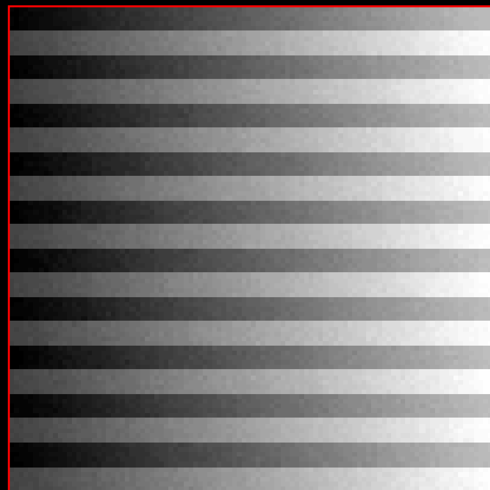


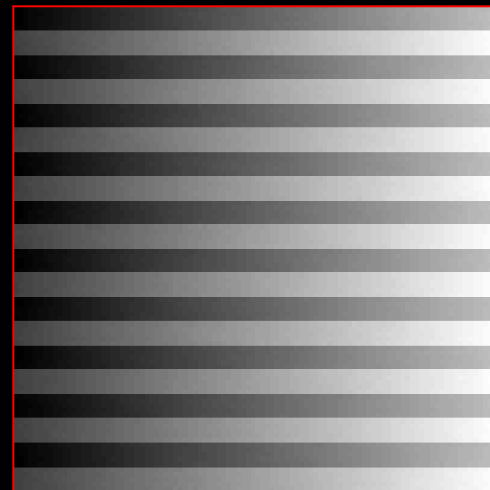
Noisy image

Gaussian noise with σ=0.2

6.4 Results

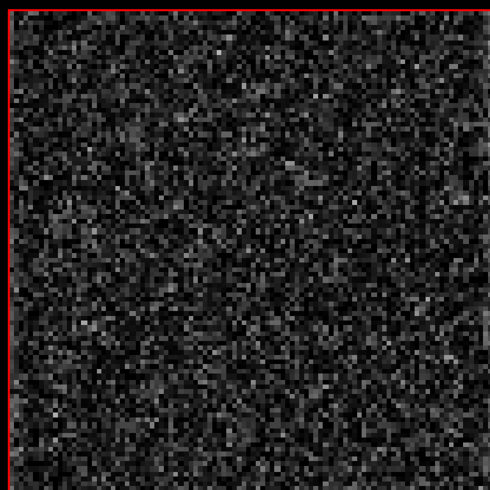
Regular
bilateral filter
Gain: 1.53





Piecewise lin.
bilateral filter
Gain: 12.91

Regular BL
Filter error
(mul. By 80)

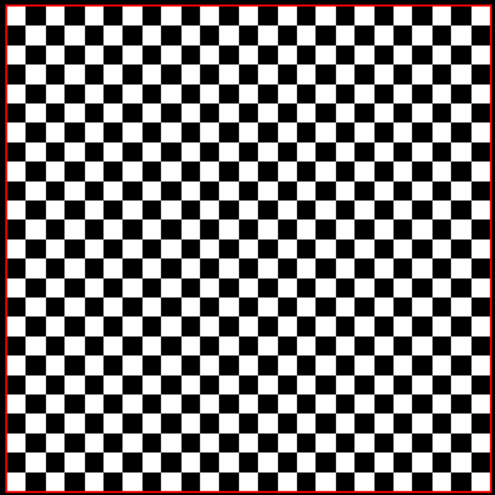


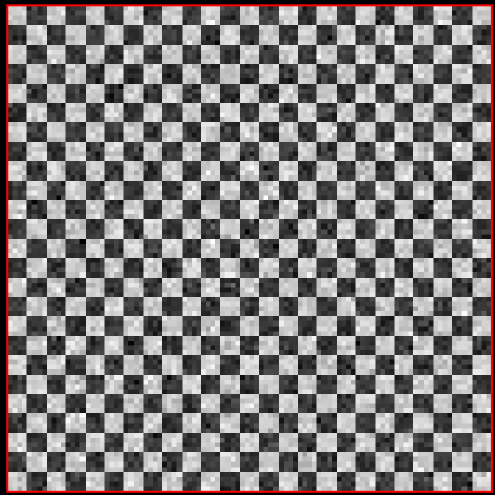
Regular BL
Filter error
(mul. By 80)

6.5 The Filter's Kernel

Original image

Values in the range [0,4]





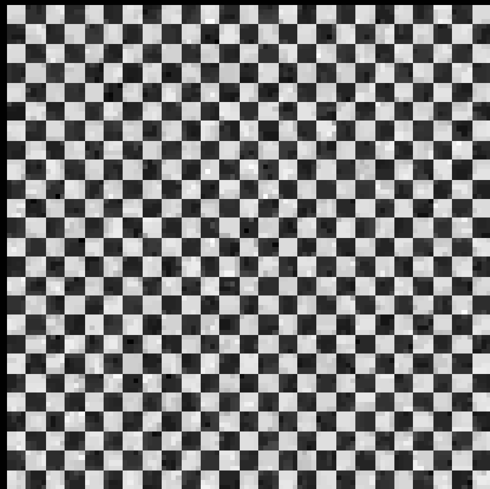
Noisy image

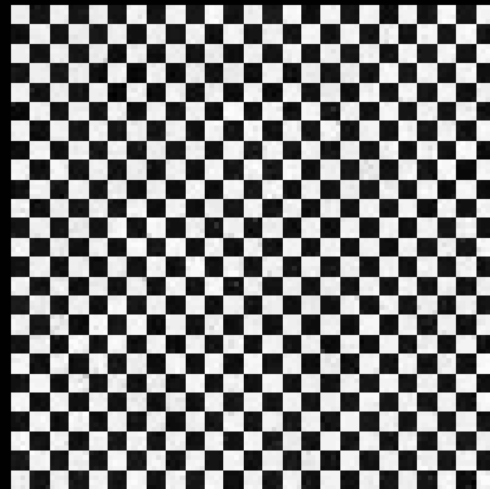
Gaussian noise with σ=0.2

6.6 RE & Bilateral Results

RE with 1500 iterations

Gain: 2.32



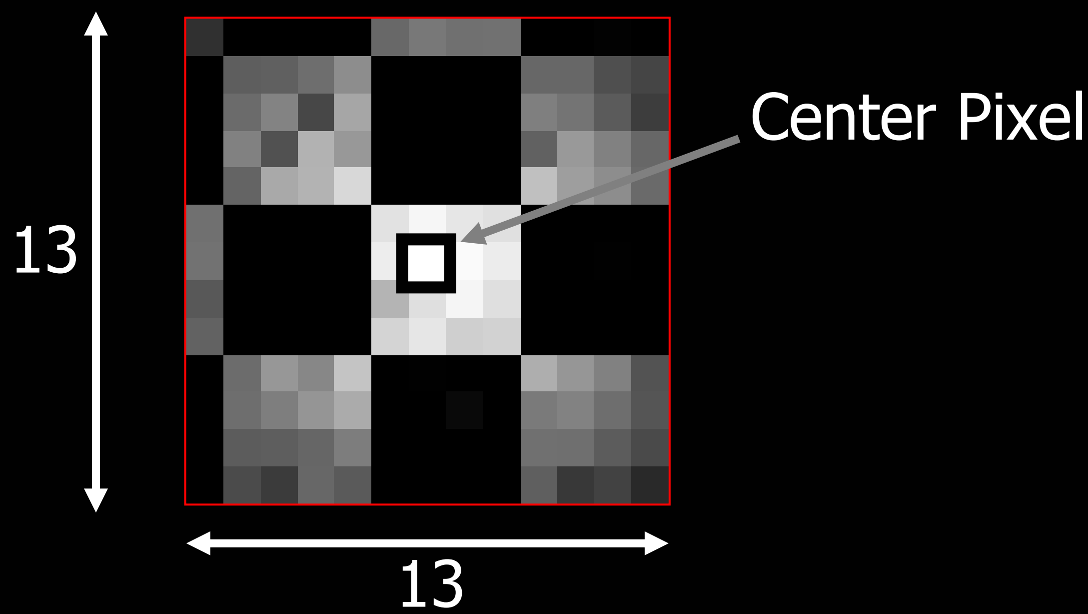


Bilateral Filter

Gain: 19.97

- The bilateral filter uses a 13 by 13 filter support, and exploits every possible pixel in this neighborhood in order to average the noise.
- The RE effective support cannot propagate across edges! Thus, at most 4 by 4 pixels (the size of the squares in the checkerboard) are averaged.

6.7 Bilateral Kernel Shape



Important Property: As opposed to the WLS, RE, and AD filters, the bilateral filter may give non-monotonically varying weights.

Chapter 7

Conclusions and Further Work

7.1 Conclusions

- The bilateral filter is a powerful alternative to the iteration-based (WLS,RE,AD) filters for noise removal.
- We have shown that this filter emerges as a single Jacobi iteration of a novel penalty term that uses 'long-distance' derivative.
- We can further speed the bilateral filter using either the GS or the sub-gradient approaches.
- We have generalized the bilateral filter for treating piece-wise linear signals.

7.2 What Next ?

- Convergence proofs for the regular bilateral filter if applied iteratively, and its speed-up variations,
- Relation to Wavelet-based (Donoho and Johnston) and other de-noising algorithms,
- Approximated bilateral filter - Enjoying the good de-noising performance while reducing complexity.