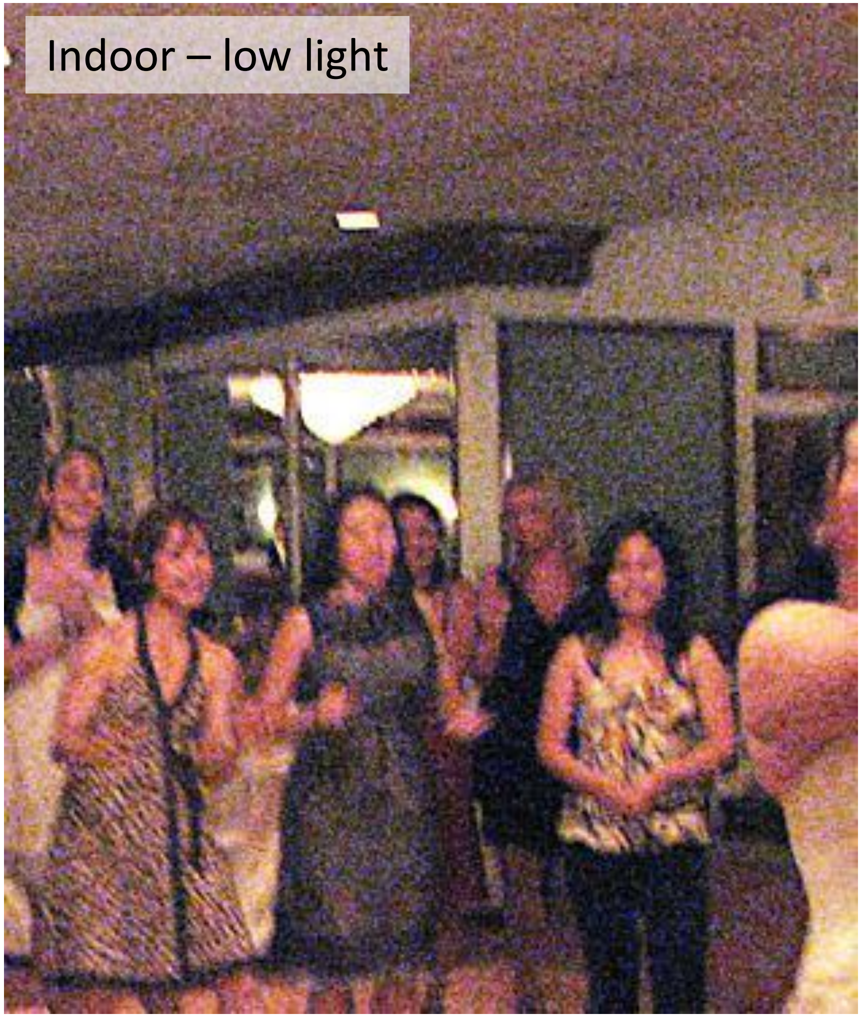
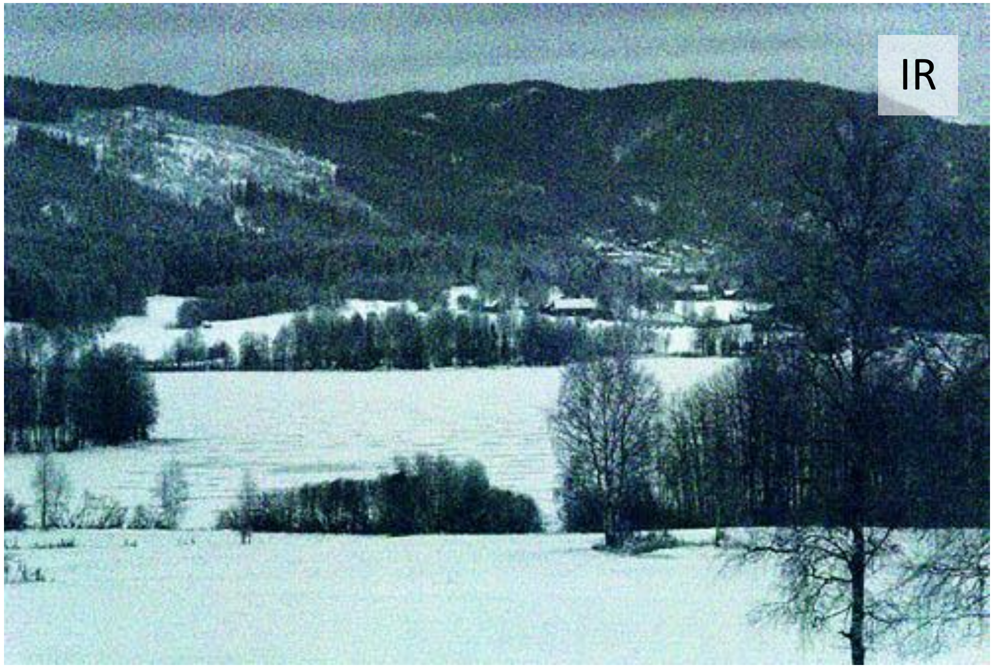
Image Denoising

Jianping Fan
Dept of Computer Science
UNC-Charlotte

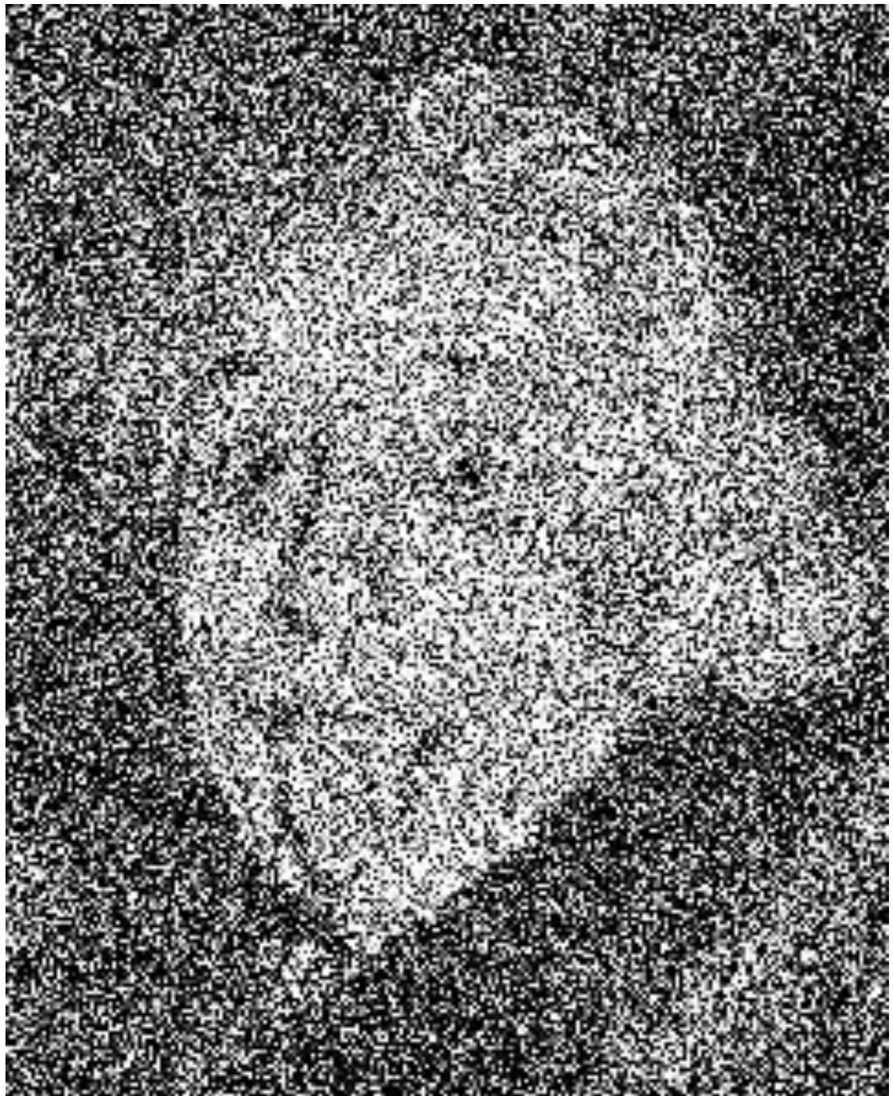
Course Website:
http://webpages.uncc.edu/jfan/itcs5152.html


Indoor – low light

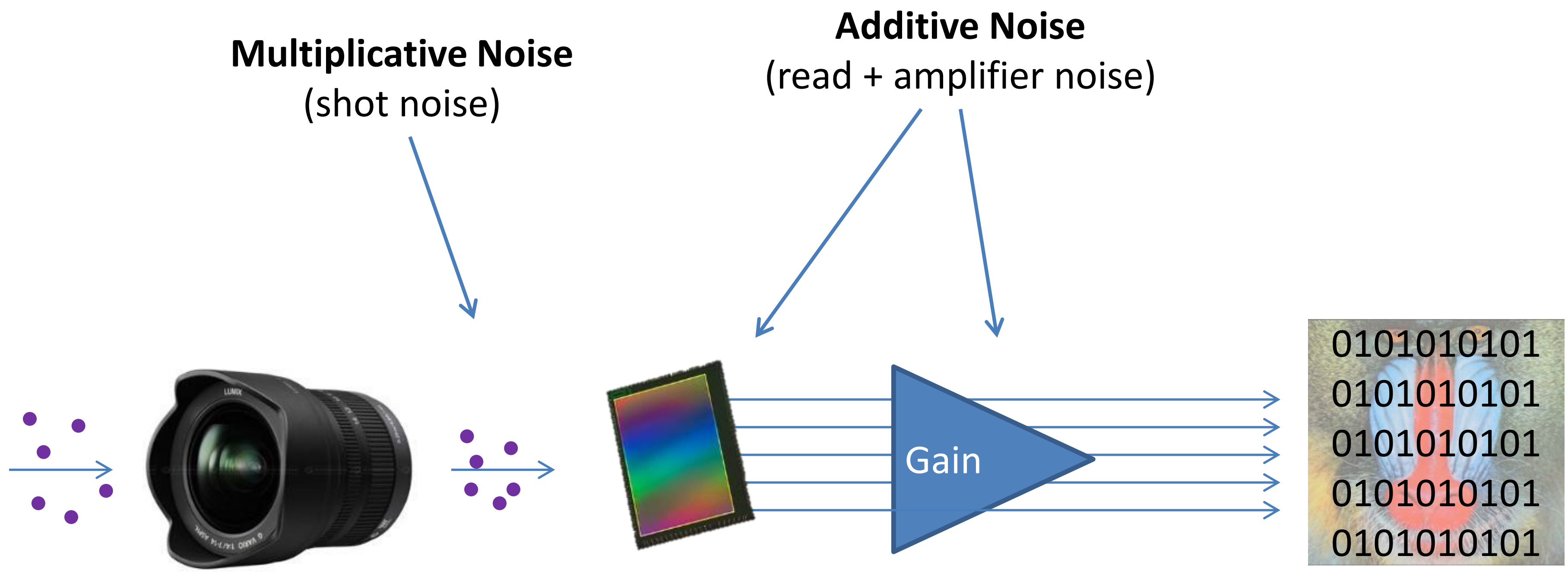

IR


US

Can we (humans) denoise?



Sources of Noise

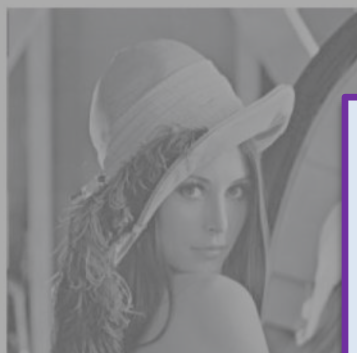


Problem Definition

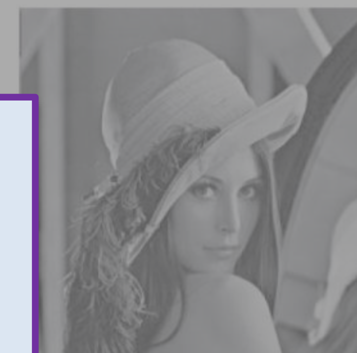
Visual Quality

MSE / PSNR

x



$\hat{x}$

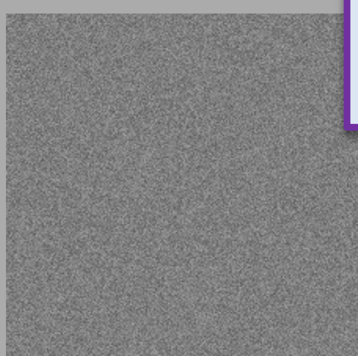


Mean Square Error

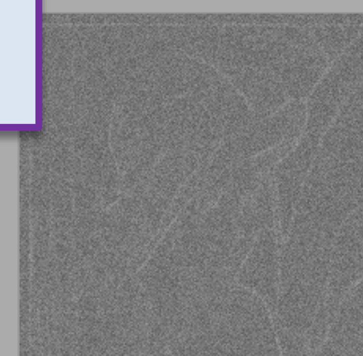
$$\text{MSE} = \|\hat{x} - x\|_2^2$$

Peak Signal-to-Noise ratio

$$\text{PSNR} = 20log_{10}\frac{255}{\sqrt{\text{MSE}}}$$



$$y = x + n$$



n

Noise? → $\hat{n} = y - \hat{x}$

Outline

- **Classical Denoising**
  - Spatial Methods
  - Transform Methods

- **State-of-the-art Methods**
  - GSM – Gaussian Scale Mixture
  - NLM – Non-local means
  - BM3D – Block Matching 3D collaborative filtering

- **Learning-based Methods**

Denoising in the Spatial Domain

- The "classical" assumption:
  Images are piecewise constant

- Neighboring pixels are highly correlated

⇒ Denoise = "Average nearby pixels" (filtering)

Denoising in the Spatial Domain

a	b	c
d	e	f
g	h	i

H

$*$

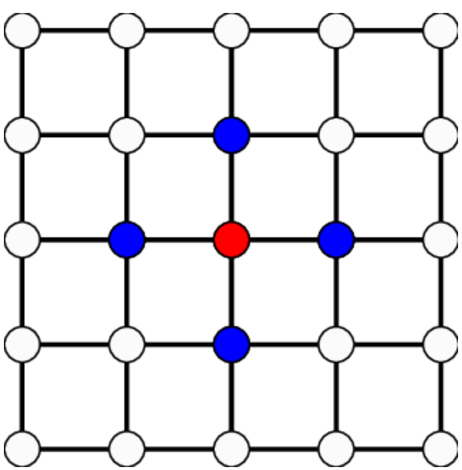
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

I

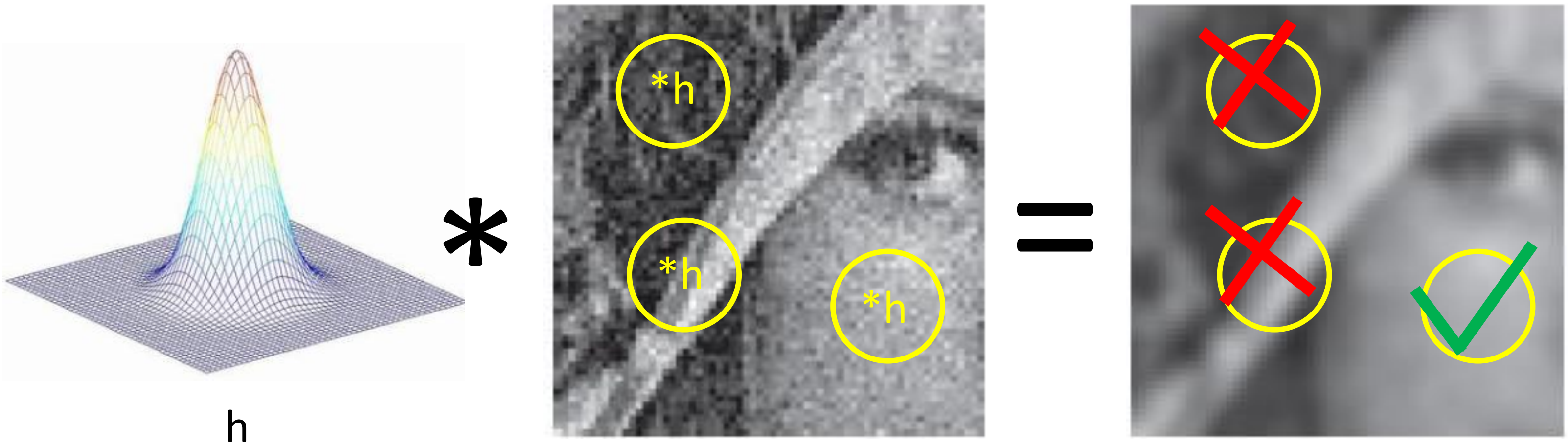
$=$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

I'



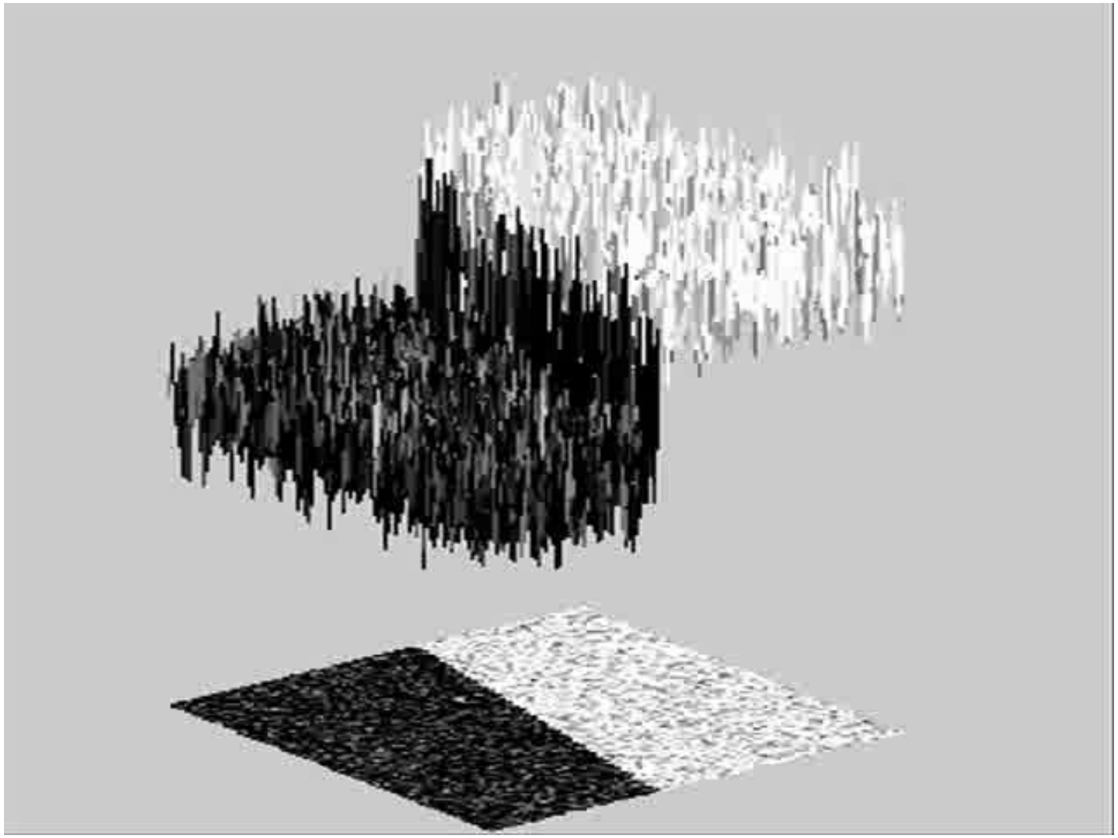
Gaussian Smoothing



h

$$\hat{x}(i) = \frac{1}{C_i} \sum_j y(j) e^{-\frac{\|i-j\|^2}{2\sigma^2}}$$

Toy Example



How can we preserve the fine details?

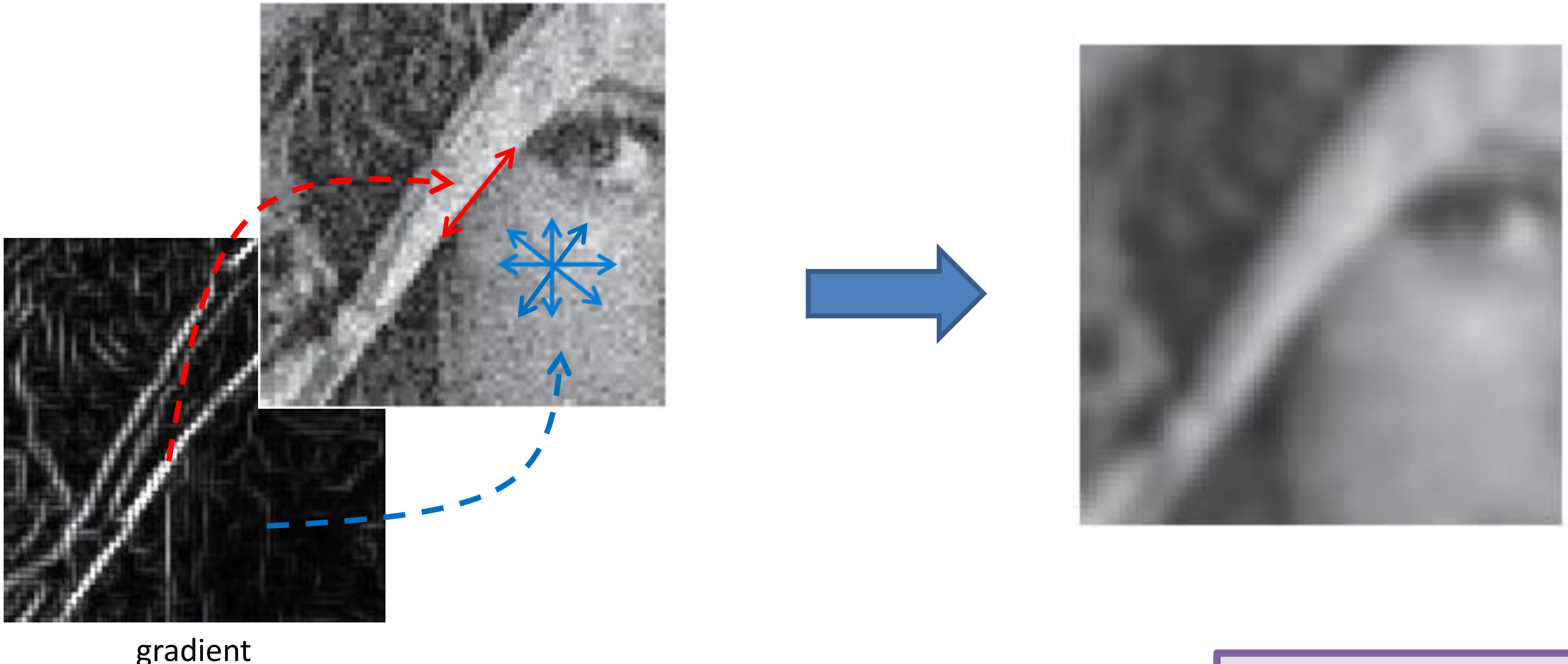
Local adaptive smoothing

- Non uniform smoothing
  Depending on image content:
  - Smooth where possible
  - Preserve fine details

How?

Anisotropic Filtering

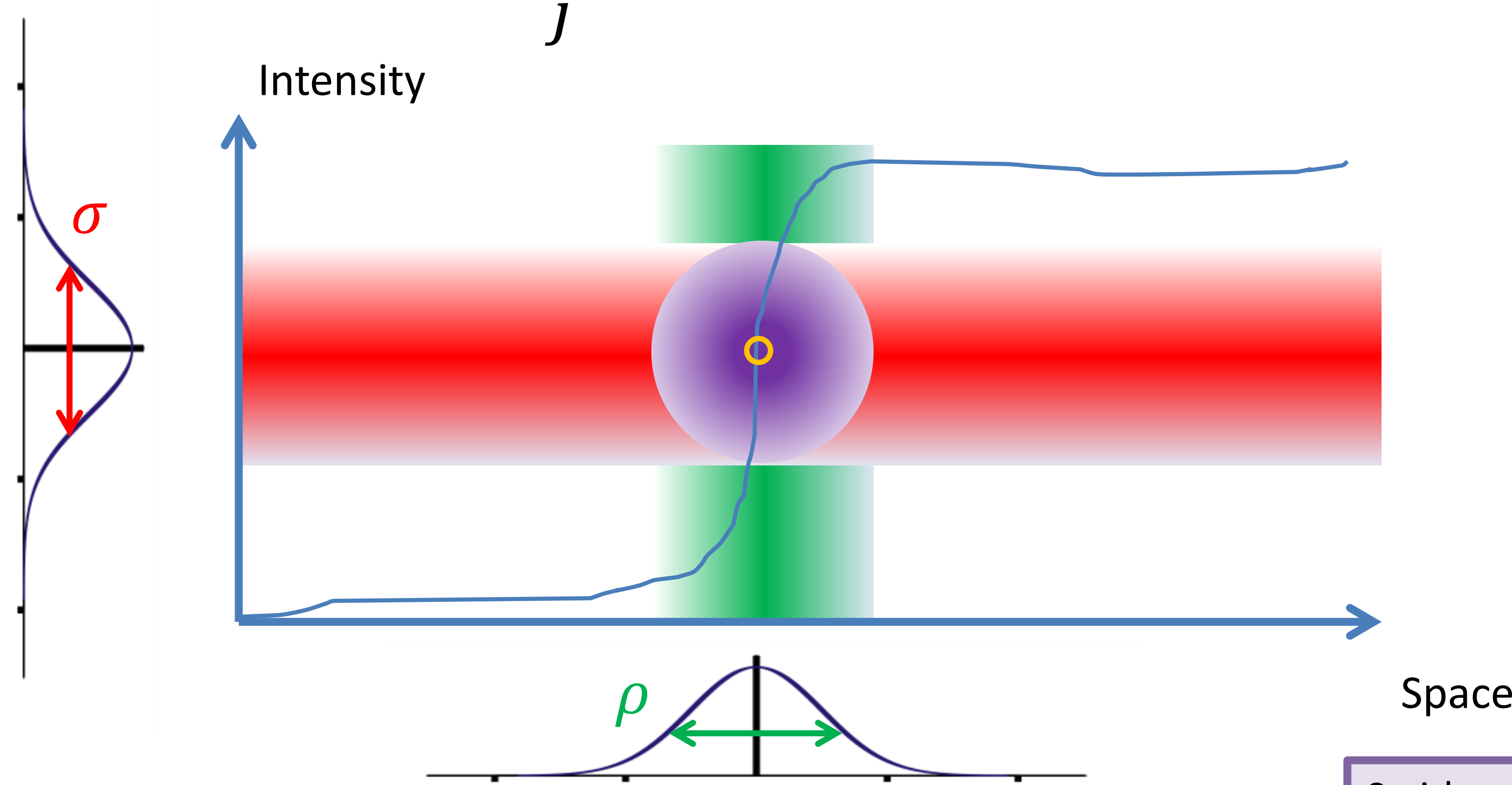
- Edges ⇒ smooth only along edges
- "Smooth" regions ⇒ smooth isotropically



Perona and Malik (1990)

Bilateral Filtering

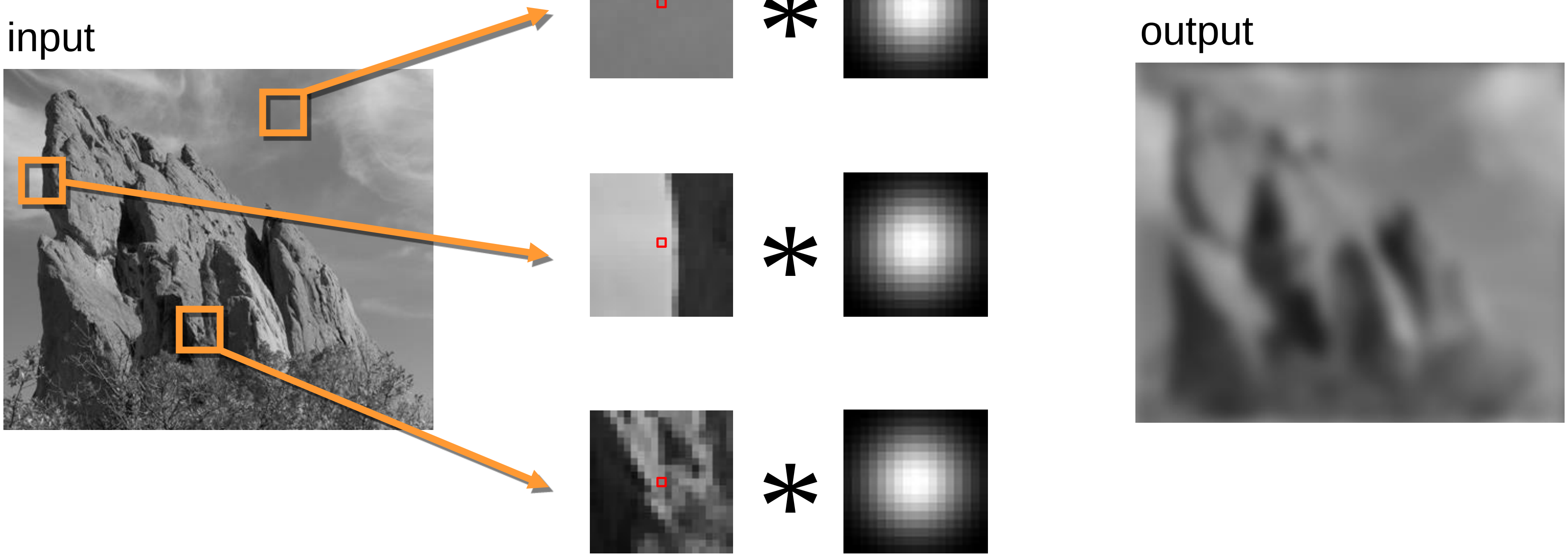
$$\hat{x}(i) = \frac{1}{C_i}\sum_j y(j)e^{-\frac{\|i-j\|^2}{2\rho^2}}$$



Smith and Brady (1997)

Slides taken from Sylvain Paris, Siggraph 2007

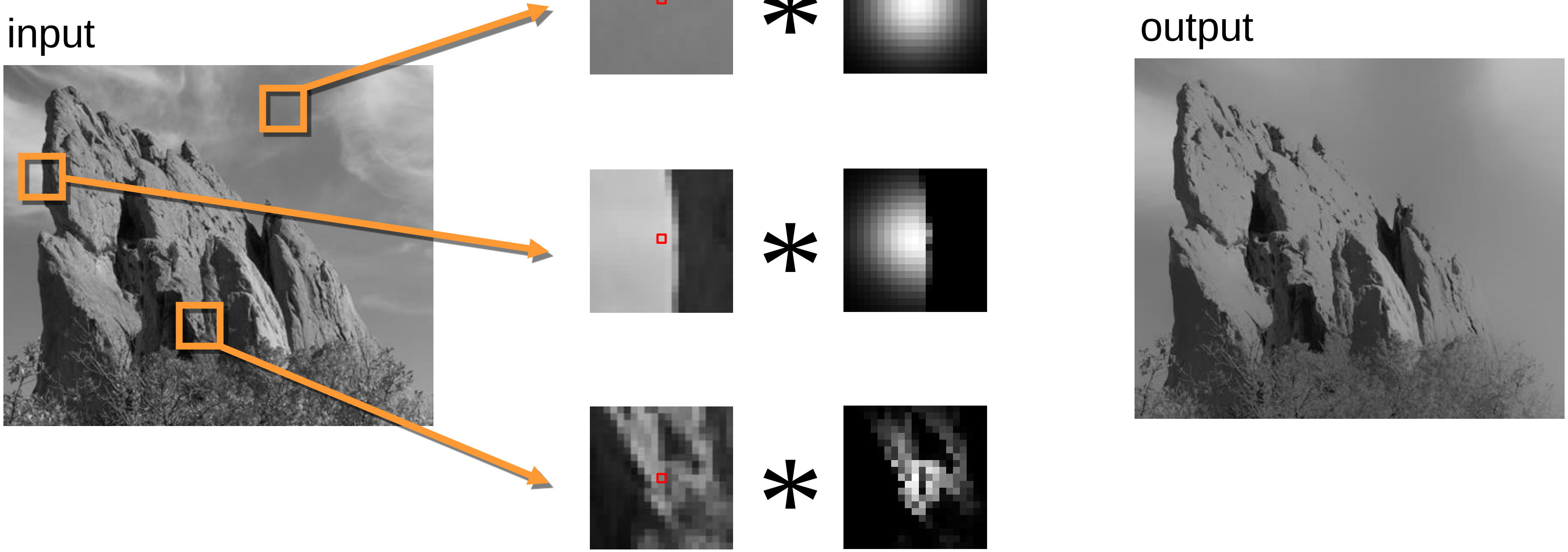
Gaussian Smoothing



Same Gaussian kernel everywhere

Averages across edges ⇒ blur

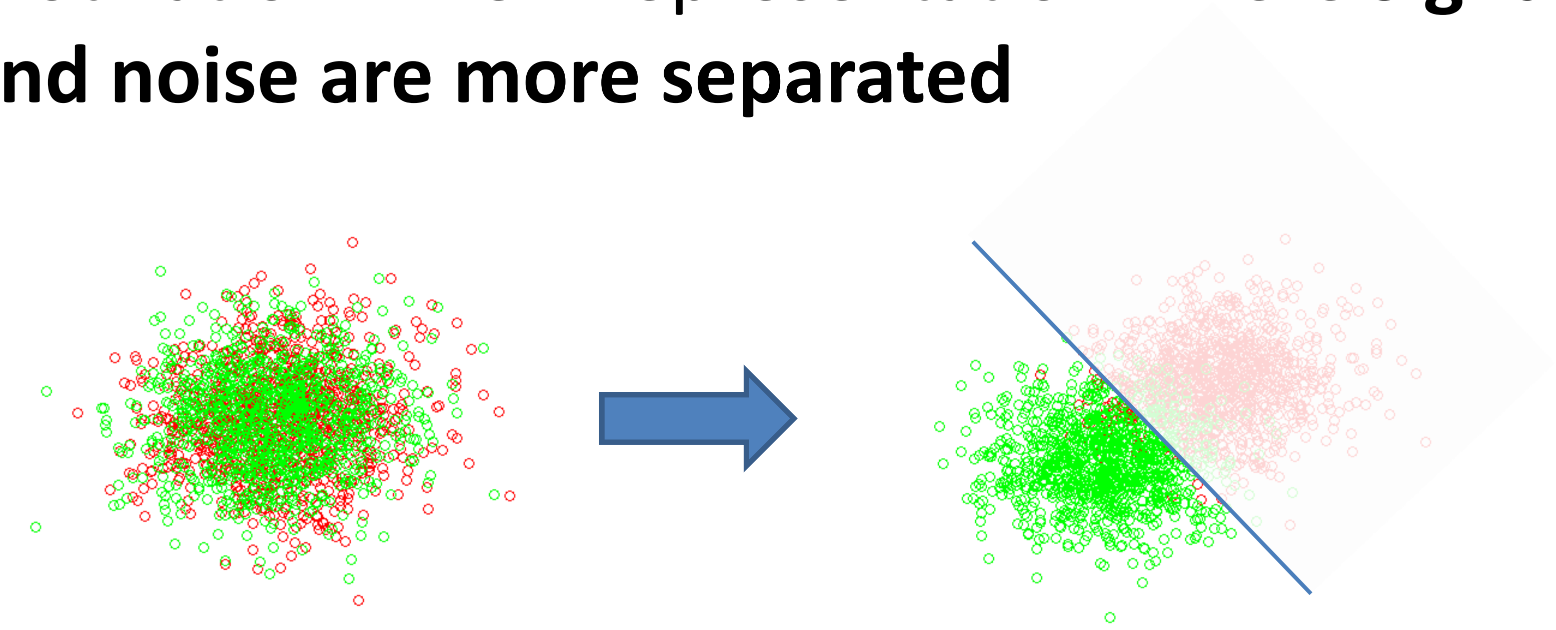
Bilateral Filtering



Kernel shape depends on image content
Avoids averaging across edges

Denoising in the Transform Domain

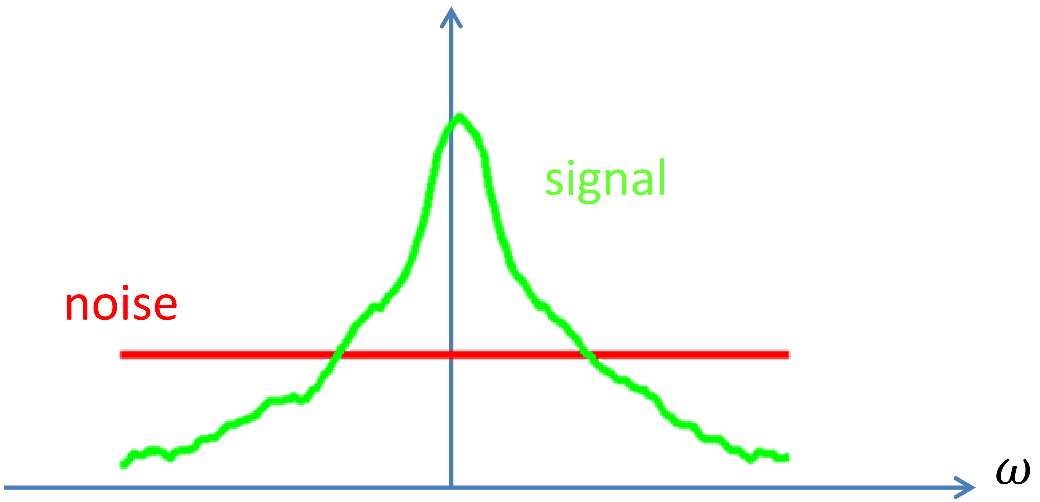
- Motivation – New representation where **signal and noise are more separated**



- Denoise = "Suppress noise coefficients while preserving the signal coefficients"

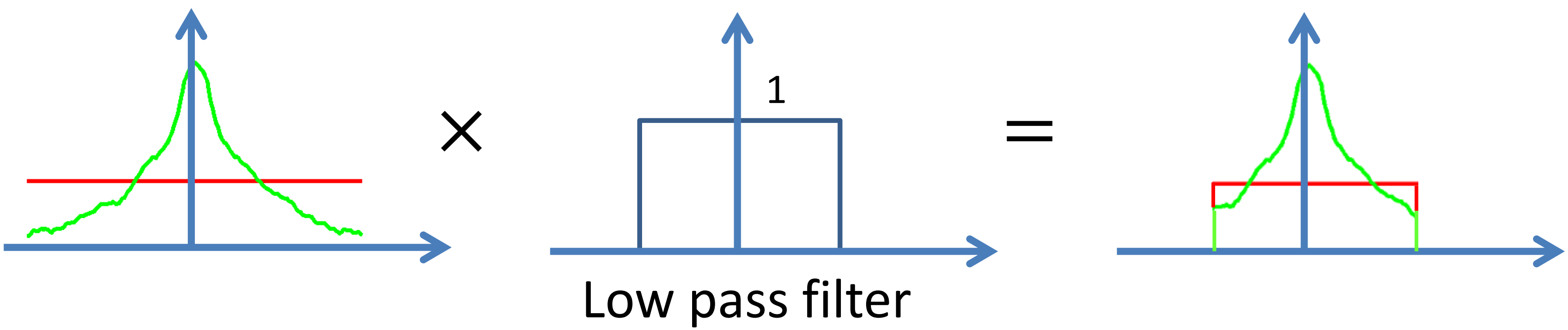
Fourier Domain

- <u>Noise</u>
  White ⇒ spread uniformly in Fourier domain
- <u>Signal</u>
  Spread non-uniformly in the Fourier domain



Low-Pass Filtering

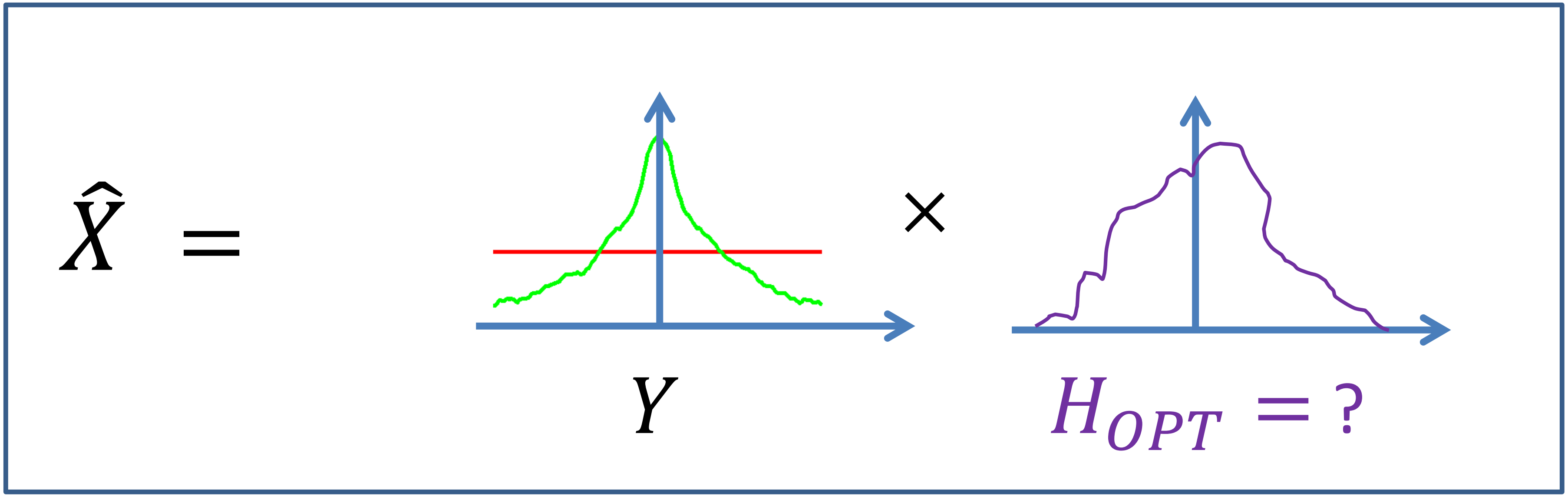
- Low pass with some cut-off frequency
- Keeps most of the signal energy



Equivalent to Global Smoothing

Looking for an Optimal Filter

$$\hat{X}(\omega) = Y(\omega)H(\omega)$$



Assumption: Signal and Noise are Stationary independent random processes

The Fourier Wiener Filter

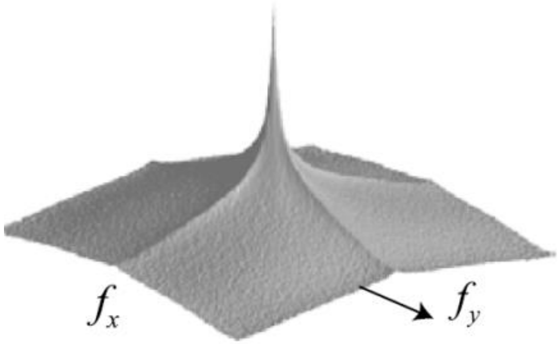
$$\hat{X}(\omega) = \underbrace{\frac{|X(\omega)|^2}{|X(\omega)|^2 + |N(\omega)|^2}}_{H_{OPT}(\omega)} Y(\omega)$$

- $|X(\omega)| \gg |N(\omega)| \Rightarrow H(\omega) \approx 1$ **= Keep**
- $|X(\omega)| \ll |N(\omega)| \Rightarrow H(\omega) \approx 0$ **= Suppress**
- Soft and adaptive thresholding

Optimal **Linear** Mean Square Error Estimator

Fourier Wiener Filter in Practice

- Use a model for $|X(\omega)|^2$ - for **example**:



or $$|X(\omega)|^2 = \frac{A}{(\alpha^2 + \|\omega\|^2)^{1+\eta}}$$

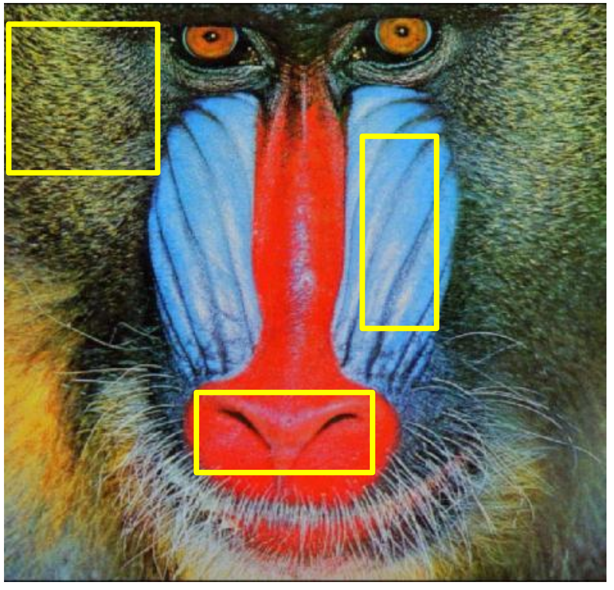
- Use $|Y(\omega)|^2$ instead (empirical approach)

$$|X(\omega)|^2 = g[|Y(\omega)|^2]$$

Ramani *et al*. (2006)
"… the Stochastic Matern Model"

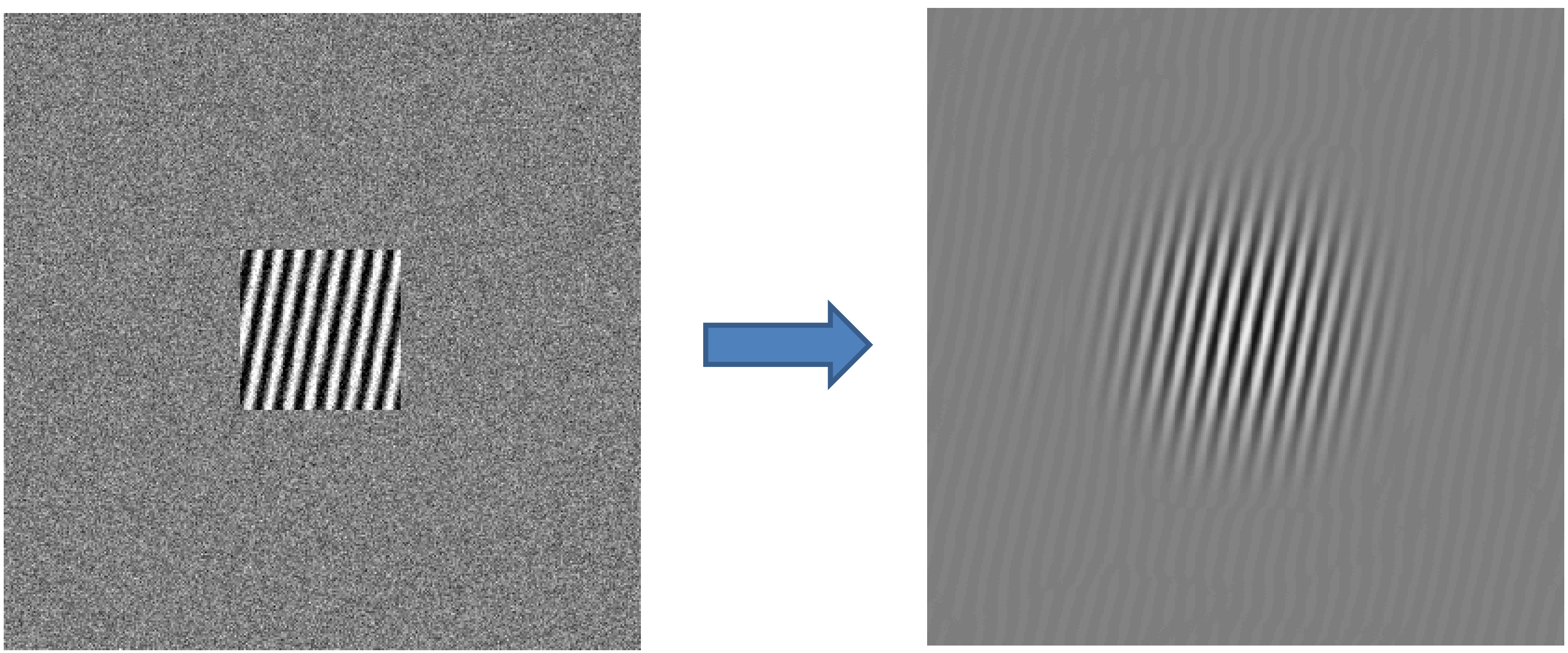
Why isn't it enough?

- We assumed stationarity:
  "statistics of all image windows is the same"
- But natural images are not stationary



Why isn't it enough?

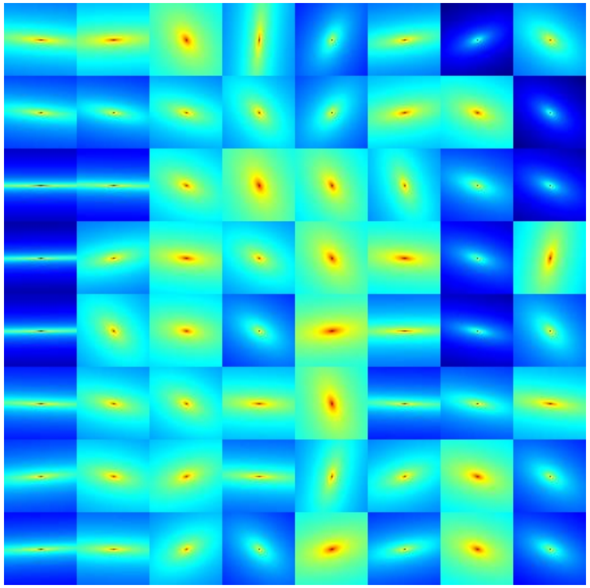
- Mismatches and errors ⇒ global artifacts



The Windowed Fourier Wiener Filter

- Image has a local structure

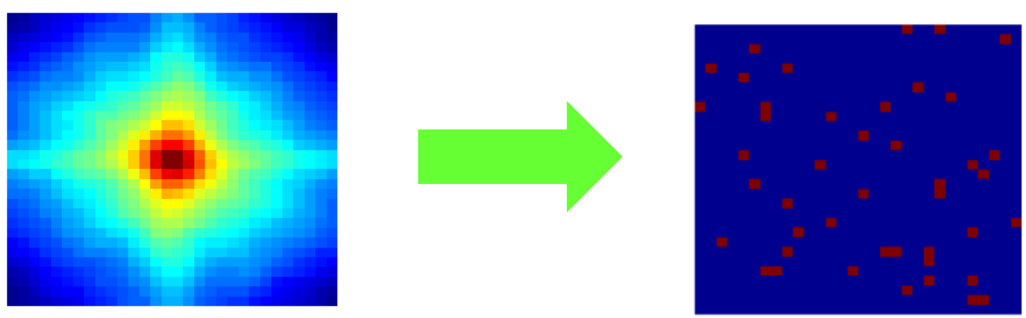
⇒ Denoise each region based on its own statistics



Perform Wiener filtering in image windows

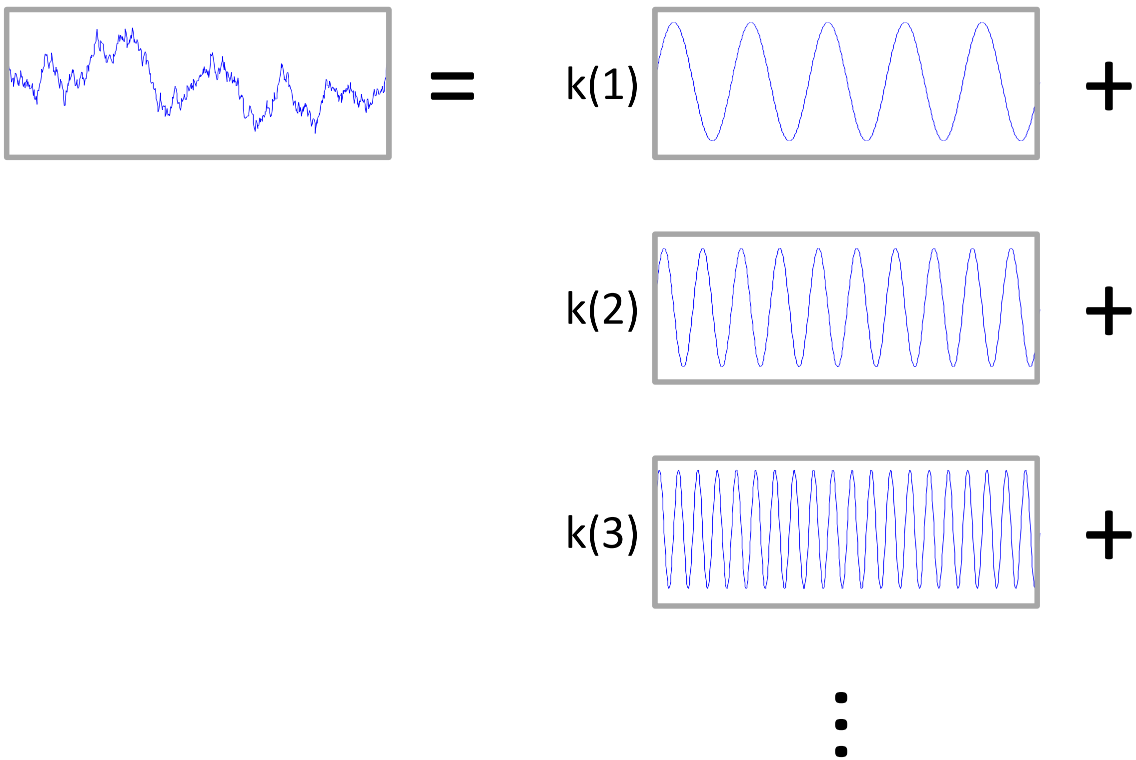
Can we do better?

- Why restrict ourselves to a Fourier basis?
- Other representations can be better:
  - Sparsity ⇒ Signal/Noise separation
  - Localization of image details

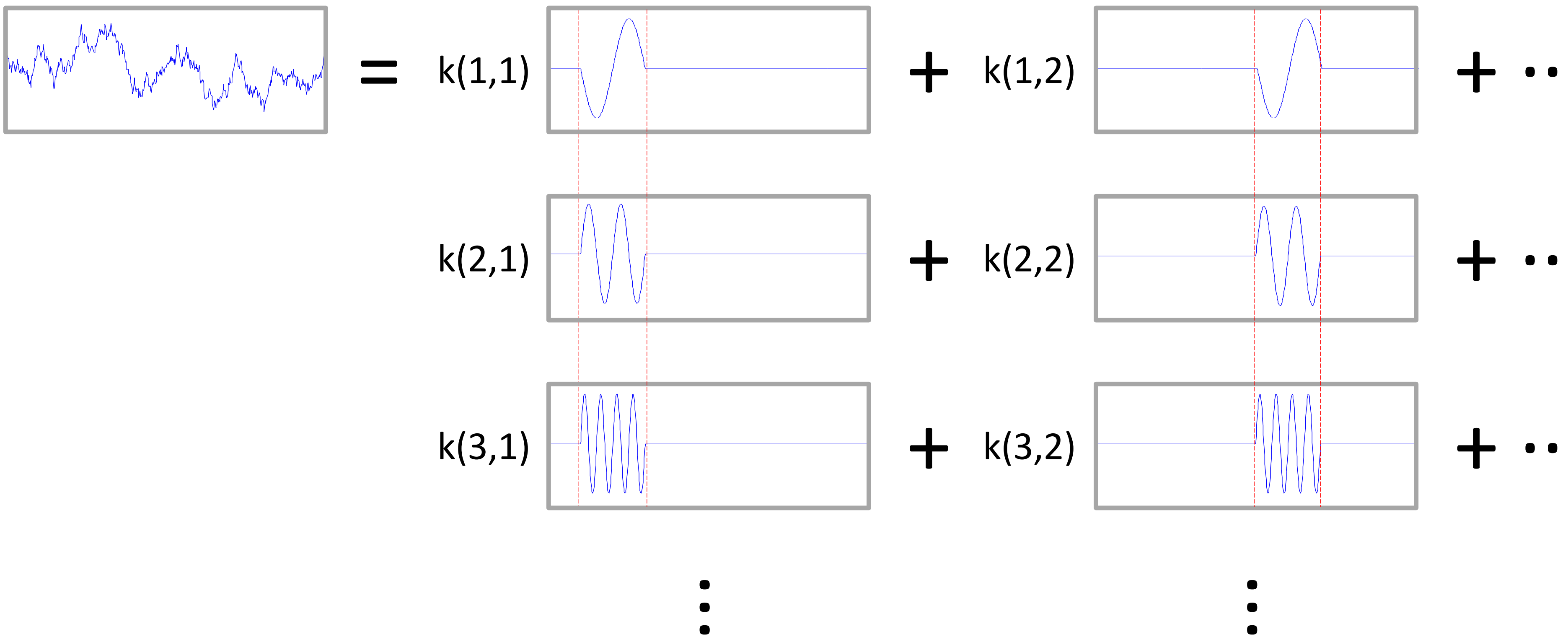


Wavelets

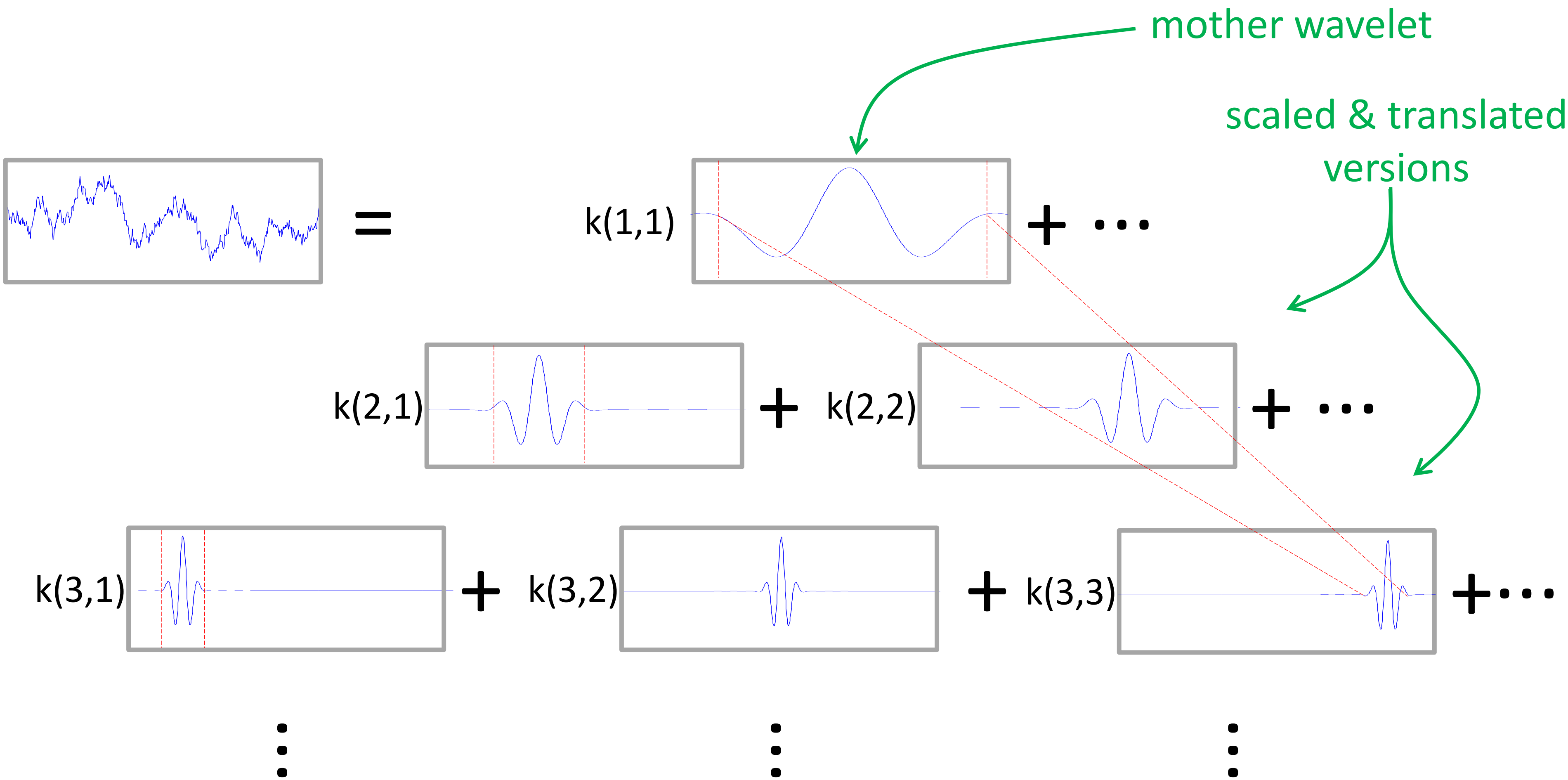
Fourier Decomposition



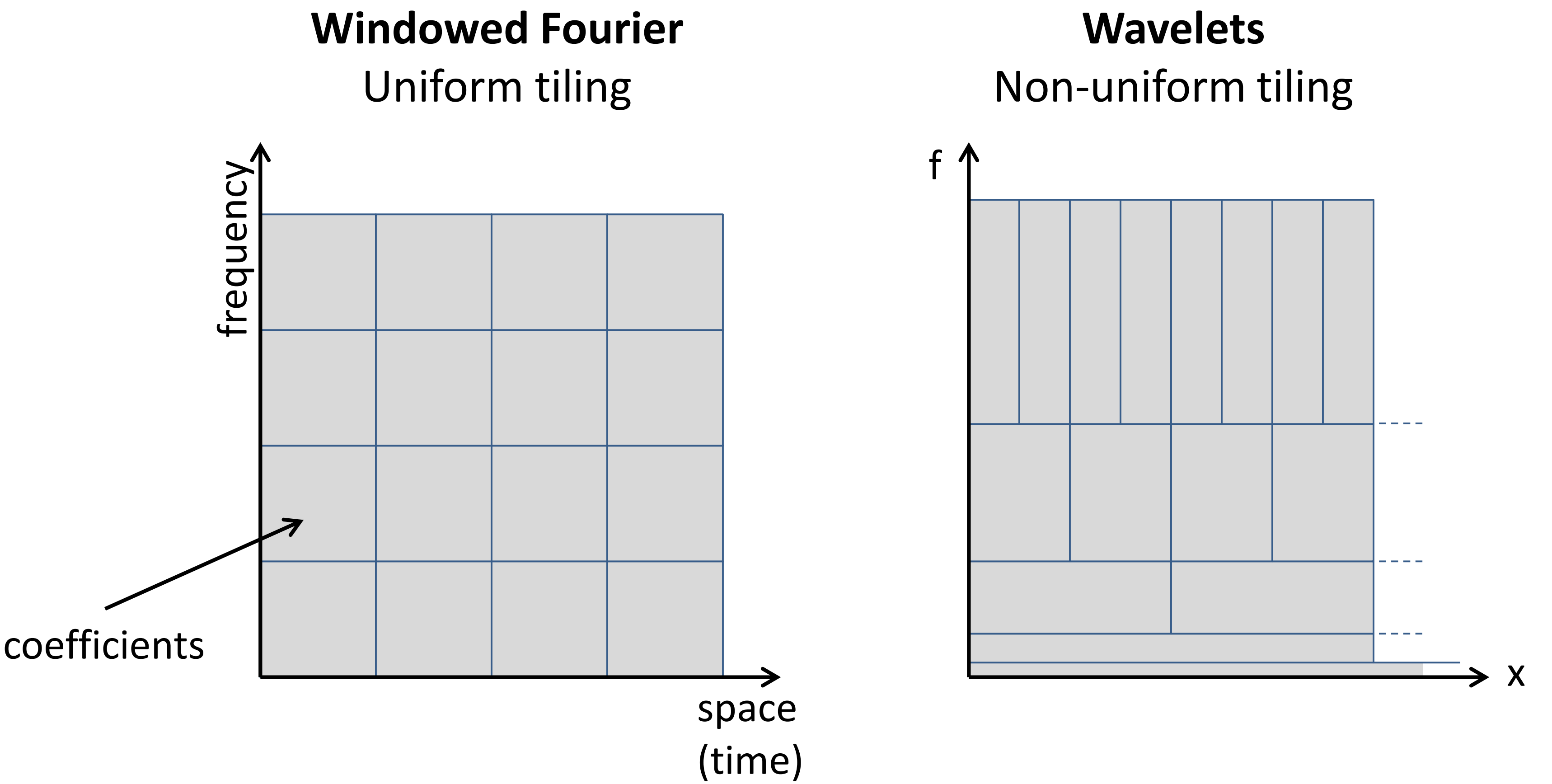
Windowed Fourier Decomposition



Wavelet Decomposition



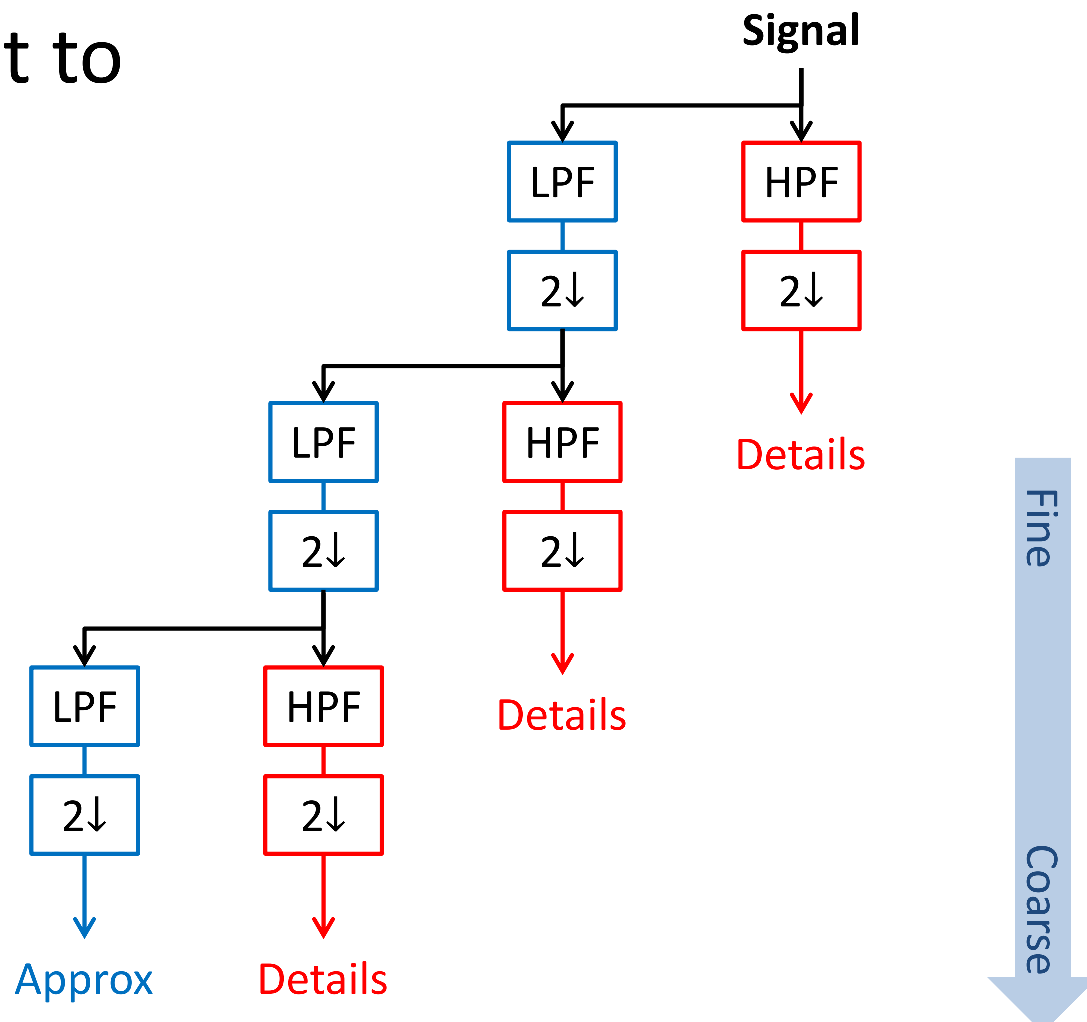
Space-Frequency Localization



Better distribution of the “Coefficient Budget”

Discrete Wavelet Transform (DWT)

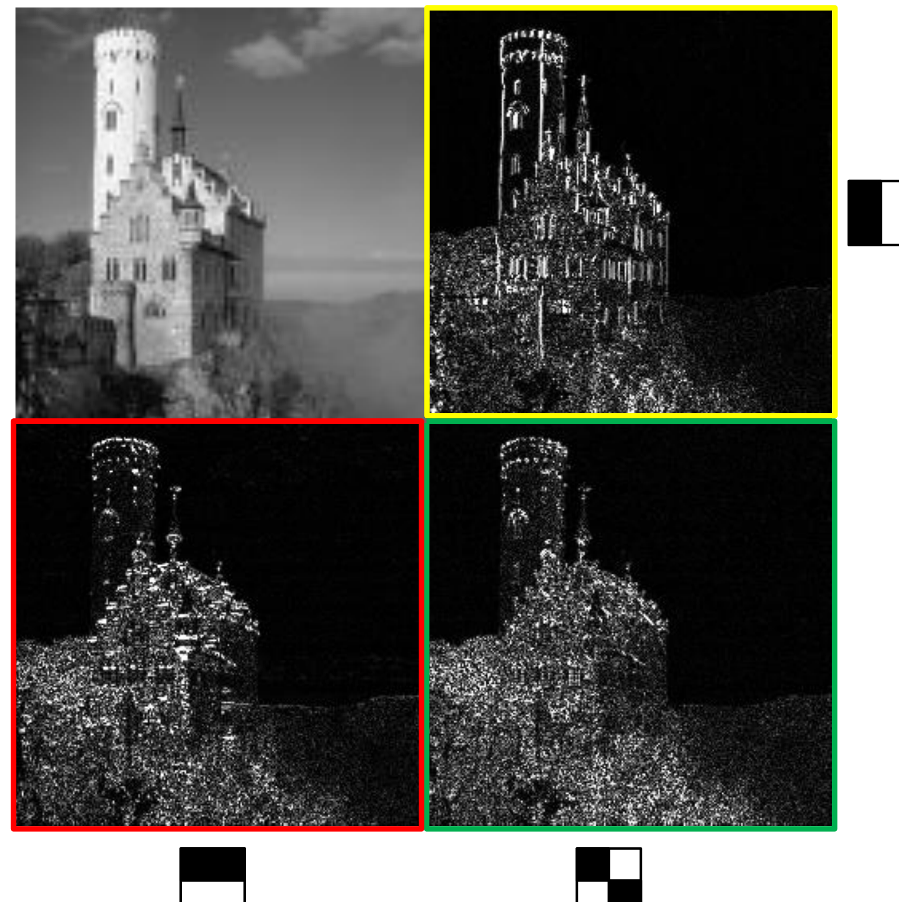
- Recursively, split to
  - Approximation
  - Details



Wavelet Transform - Example

Original image

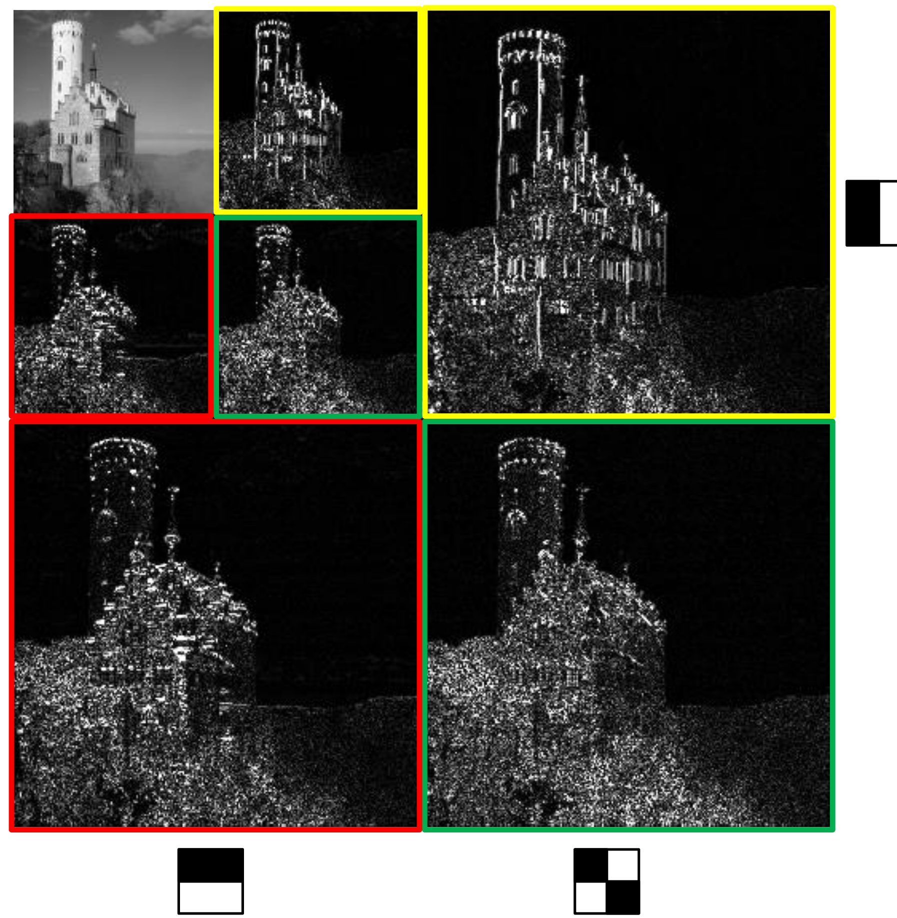
1 level DWT



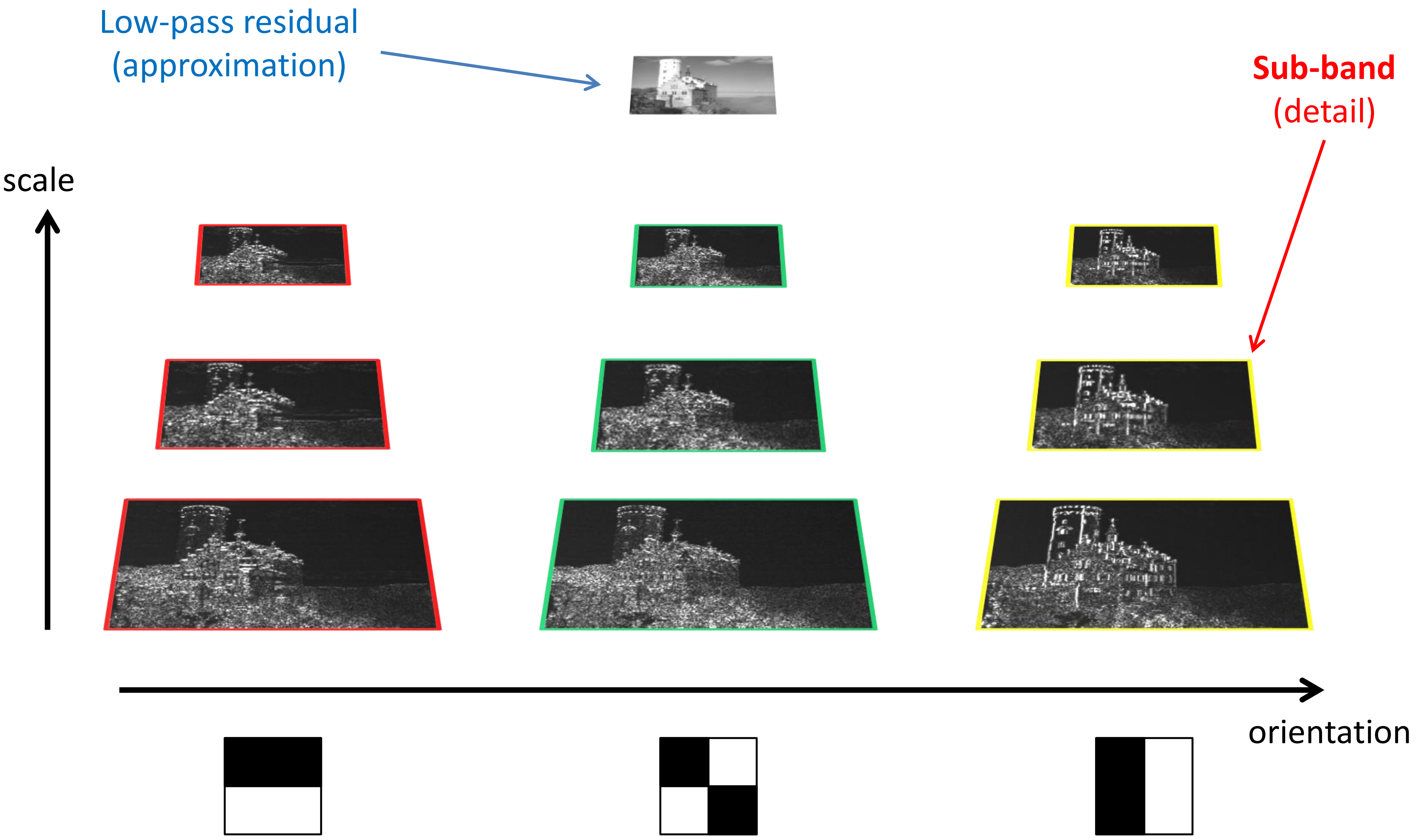
Wavelet Transform - Example

Original image

2 level DWT

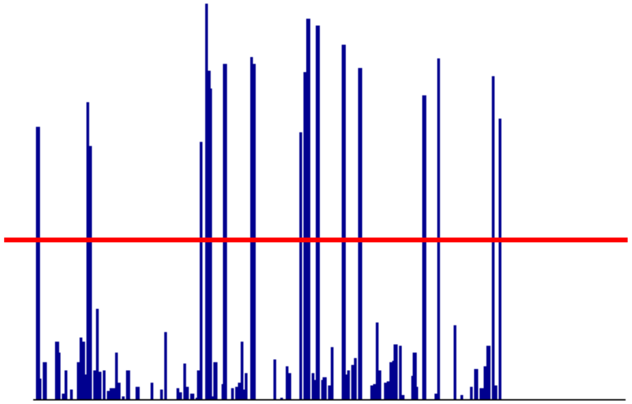


Wavelet Pyramid



Wavelet Thresholding (WT)

- Wavelet ⇒ Sparser Representation
- Improved separation between signal and noise at different scales and orientations



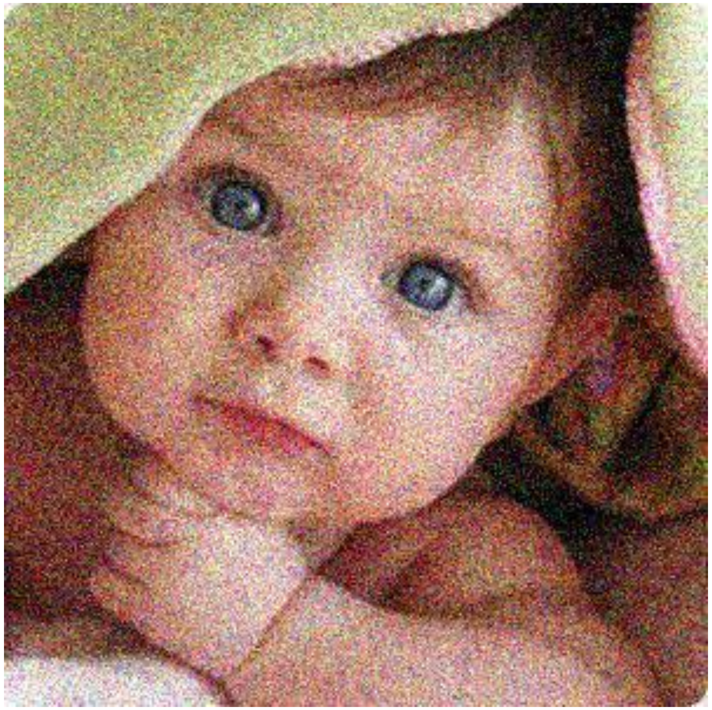
Thresholding (hard/soft) is more meaningful

A Probabilistic Perspective

- Learn or assume statistical model of image and noise - $p(x), p(n)$

- Use Bayesian inference to obtain $\hat{x}$

Which image do you prefer?



A Probabilistic Perspective

- With some prior knowledge about images
- Denoise = “find an optimal explanation”:

  - **MAP** – Maximum a posterior

$$\hat{x} = \text{argmax}_x p(x|y)$$

  - **MMSE** – Minimum Mean Square Error

$$\hat{x} = \text{argmin}_{\hat{x}} E\{(\hat{x}(y) - x)^2\} = E(x|y)$$

Performance Evaluation
Denoised Images

Original
$\sigma = 20$

Gaussian
Smoothing

Anisotropic
Filtering

Bilateral
Filtering

Windowed
Weiner

Hard WT

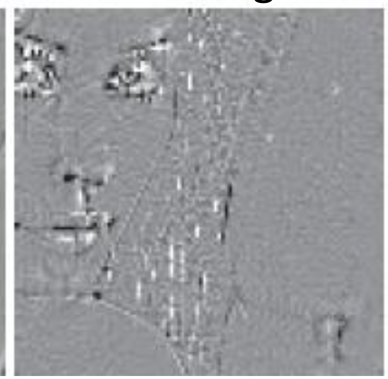
Soft WT

Buades *et al*. (2005)

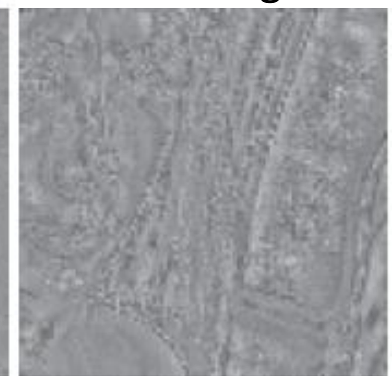
Performance Evaluation
Method Noise

Gaussian Smoothing

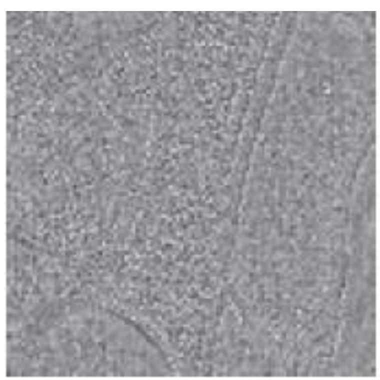
Anisotropic Filtering



Bilateral Filtering



Windowed Weiner

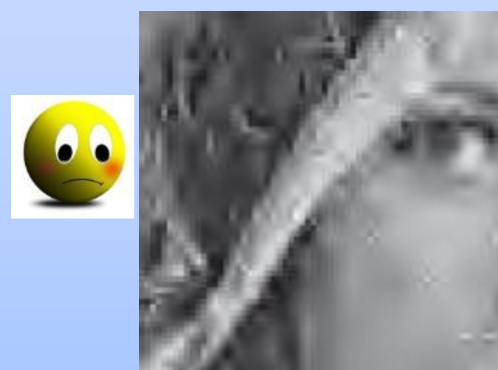
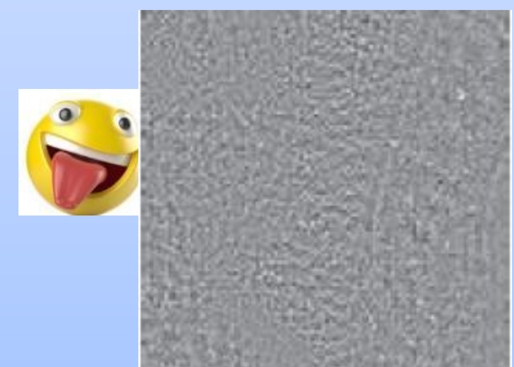


Hard WT

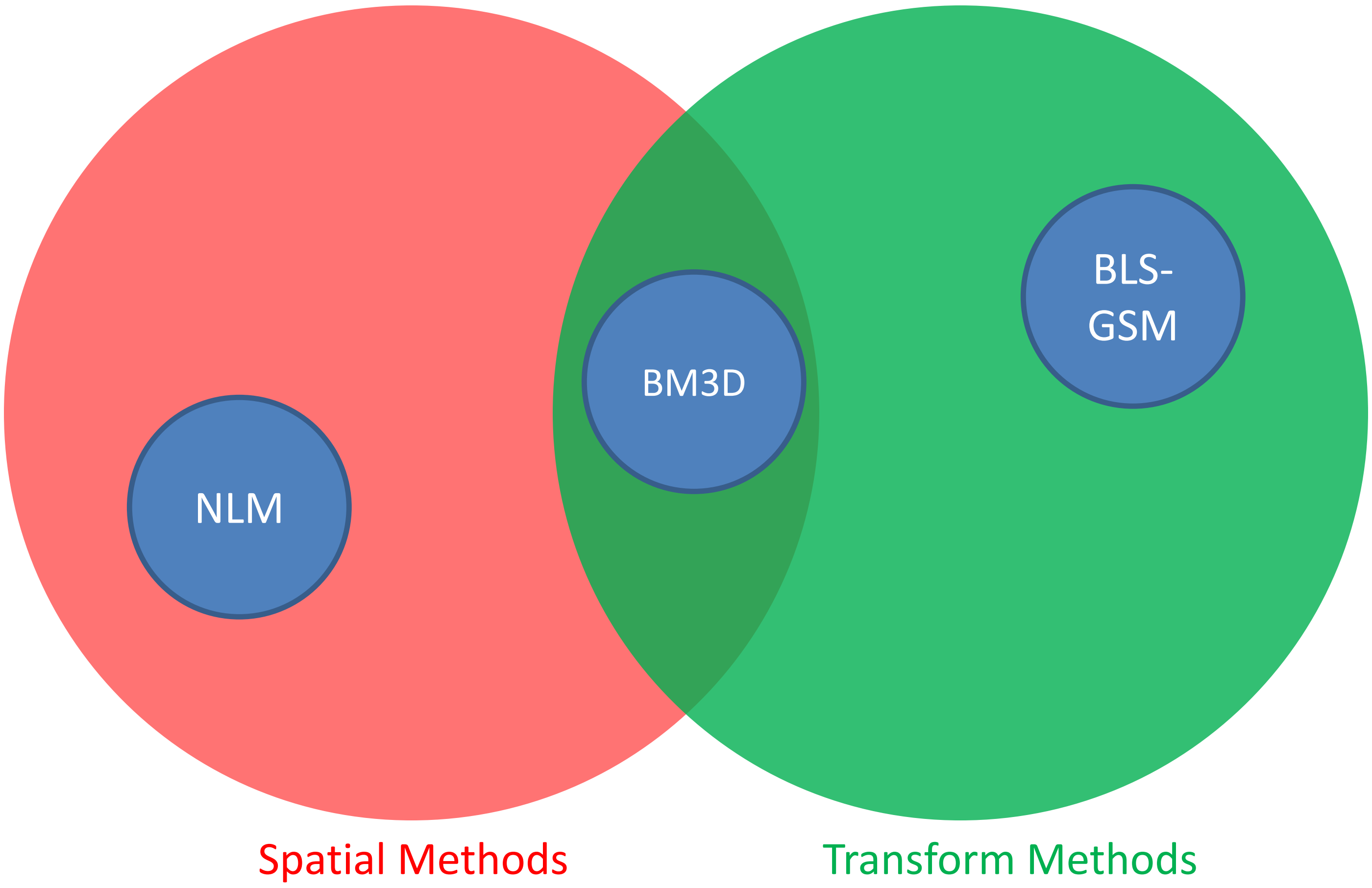
Soft WT

Buades *et al*. (2005)

Conclusions

	Denoised Image	Method Noise
Spatial Methods		
Transform Methods		

State of the art Methods



BLS-GSM-Wavelet Denoising

Bayes Least Squares Gaussian Scale Mixture Wavelet Denoising

(Portilla *et al*. 2003)

- Transform to Wavelet domain
- Assume GSM model on neighborhoods
- Denoise using BLS estimation

Over Complete Wavelets

- BLS-GSM uses over-complete wavelets

Classical (orthogonal) Wavelets:
#coefficients **=** #pixels

Over-complete Wavelets:
#coefficients **>** #pixels

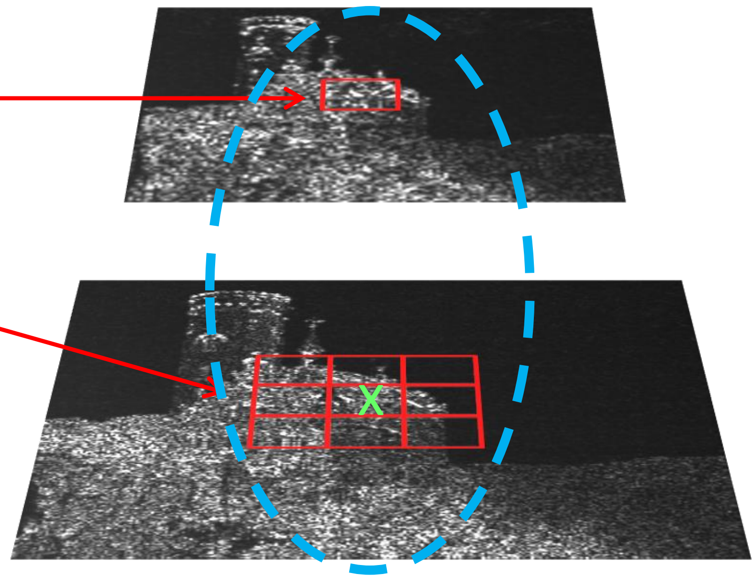
Representation is redundant

⇒ Combined estimates may improve denoising

Local Neighborhoods

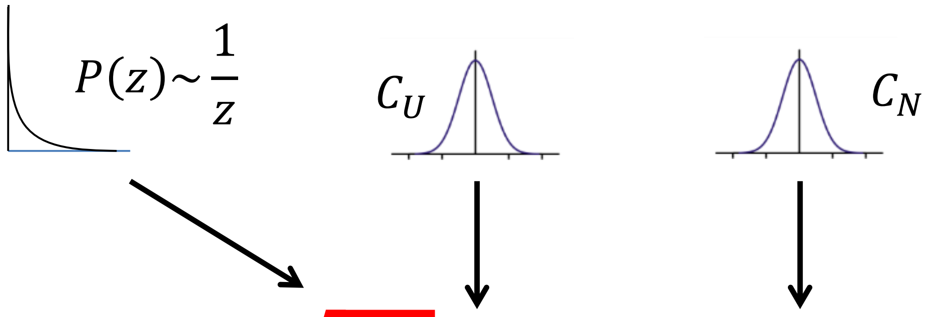
- Spatial-Scale Neighborhood

  For example:
  - Scale parent of central coef
  - 3x3 in space



- Such neighborhoods are highly structured

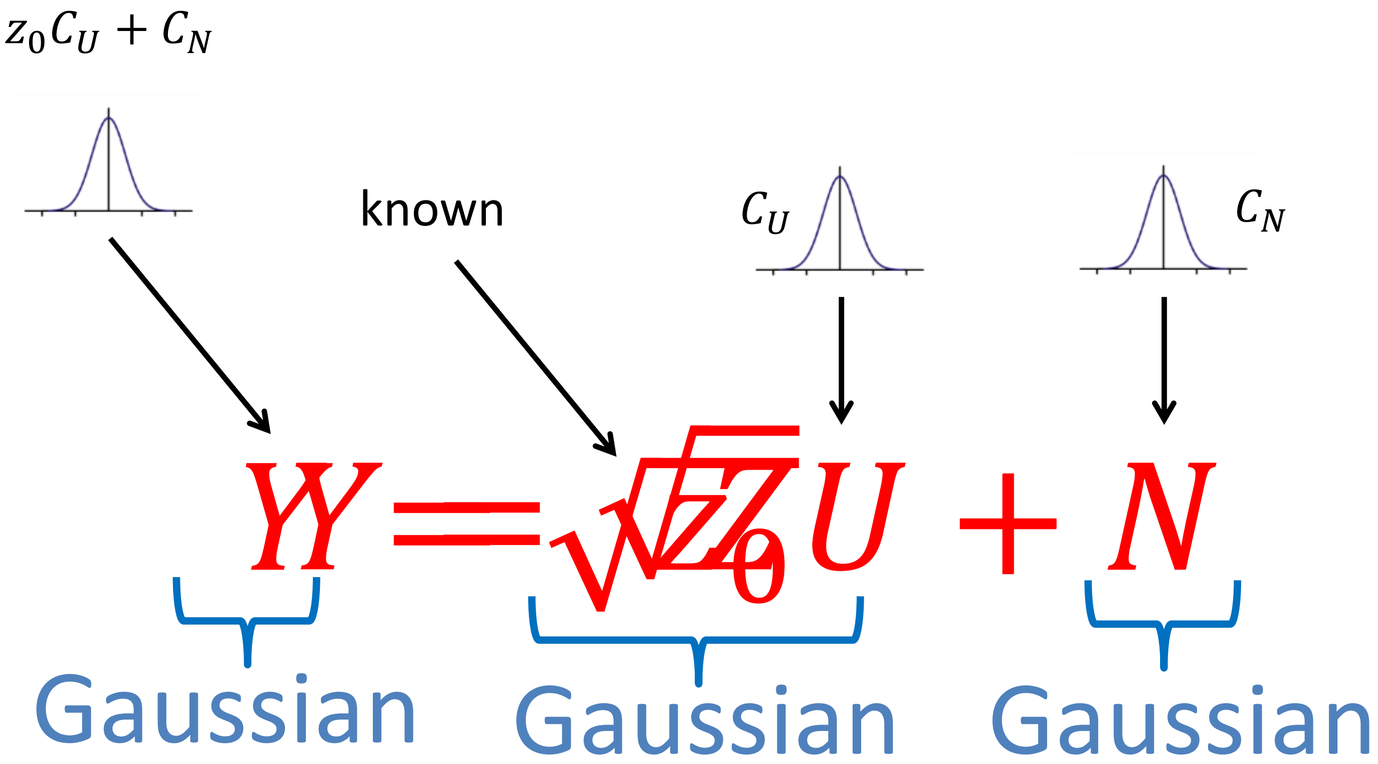
The GSM model



$$Y = \underbrace{\sqrt{Z}U}_{X} + \underbrace{N}$$

GSM Gaussian

The GSM model



Everything is Gaussian!

2-Step GSM Denoising

- The Naive approach
  For each neighborhood Y:
  1. Estimate Z => X, Y are jointly Gaussian
  2. Denoise= optimal estimation of $X|\mathrm{Y}, \mathrm{Z} = z_0$

In MMSE sense:

$$\hat{X}(Y) = E(X|Y, Z = z_0) = z_0 C_U (z_0 C_U + C_N)^{-1} Y$$

This is the Wiener estimate of X

Joint GSM Denoising

- 2-Step is sub-optimal…
- For each neighborhood Y:
  Find the MMSE estimator -> $E(X|Y)$

$$E(X|Y) = \int p(z|Y) E(X|Y,z) dz$$

?

The local Wiener estimate
(shown last slide)

Posterior distribution of multiplier

- Bayes' rule:

$$p(Z|Y) = \frac{p(Y|Z)p_z(Z)}{\int p(Y|\alpha)p_z(\alpha)d\alpha}$$

Gaussian (given Z)

Known (Prior)

Weighted sum of Wiener estimates

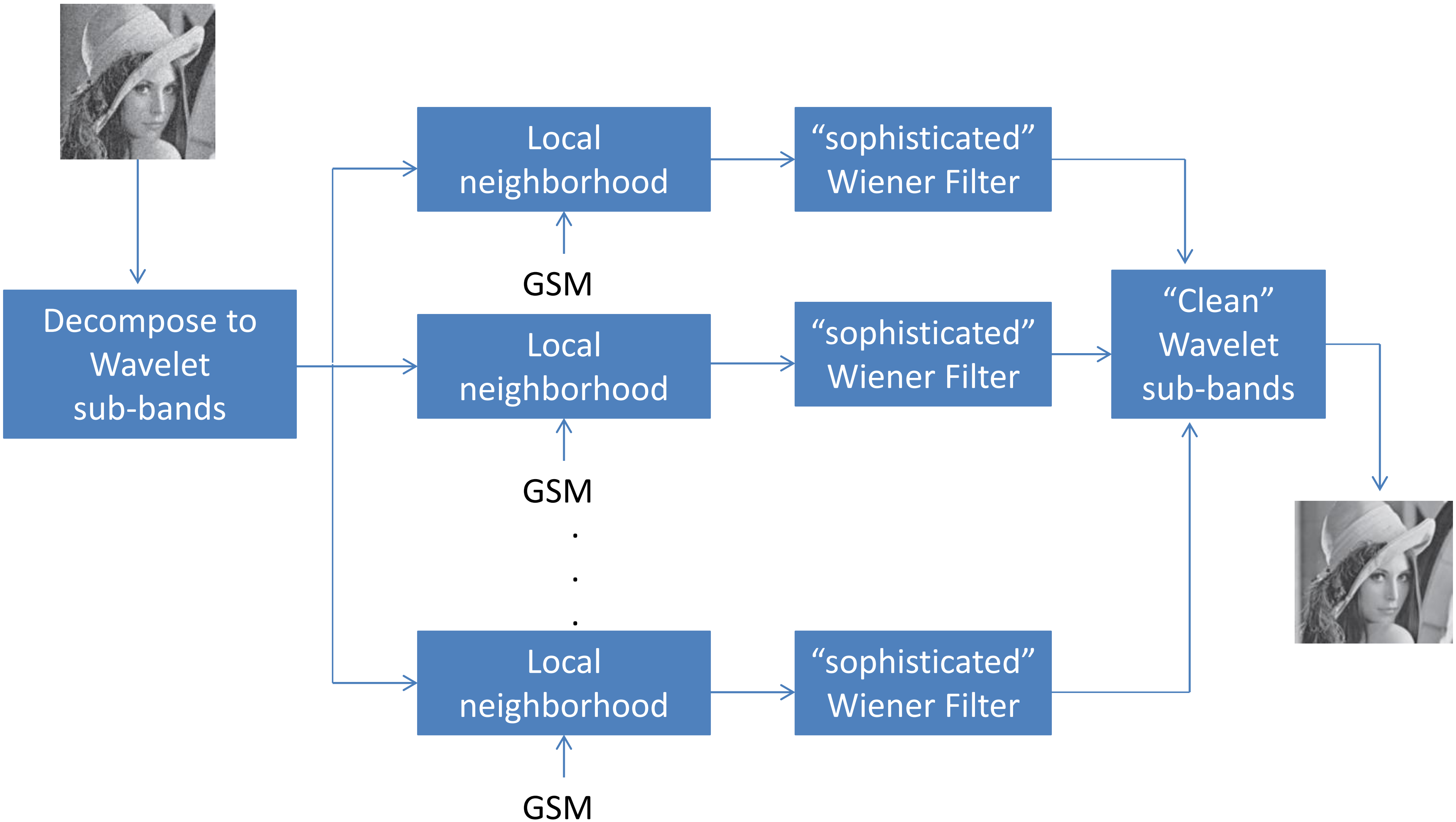
$$\hat{X} = E(X|Y) = \int p(z|Y)\, E(X|Y,z)\, dz$$

$$\hat{X} = \int w_z(Y)\, \mathrm{Wiener}_z(Y)\, dz$$

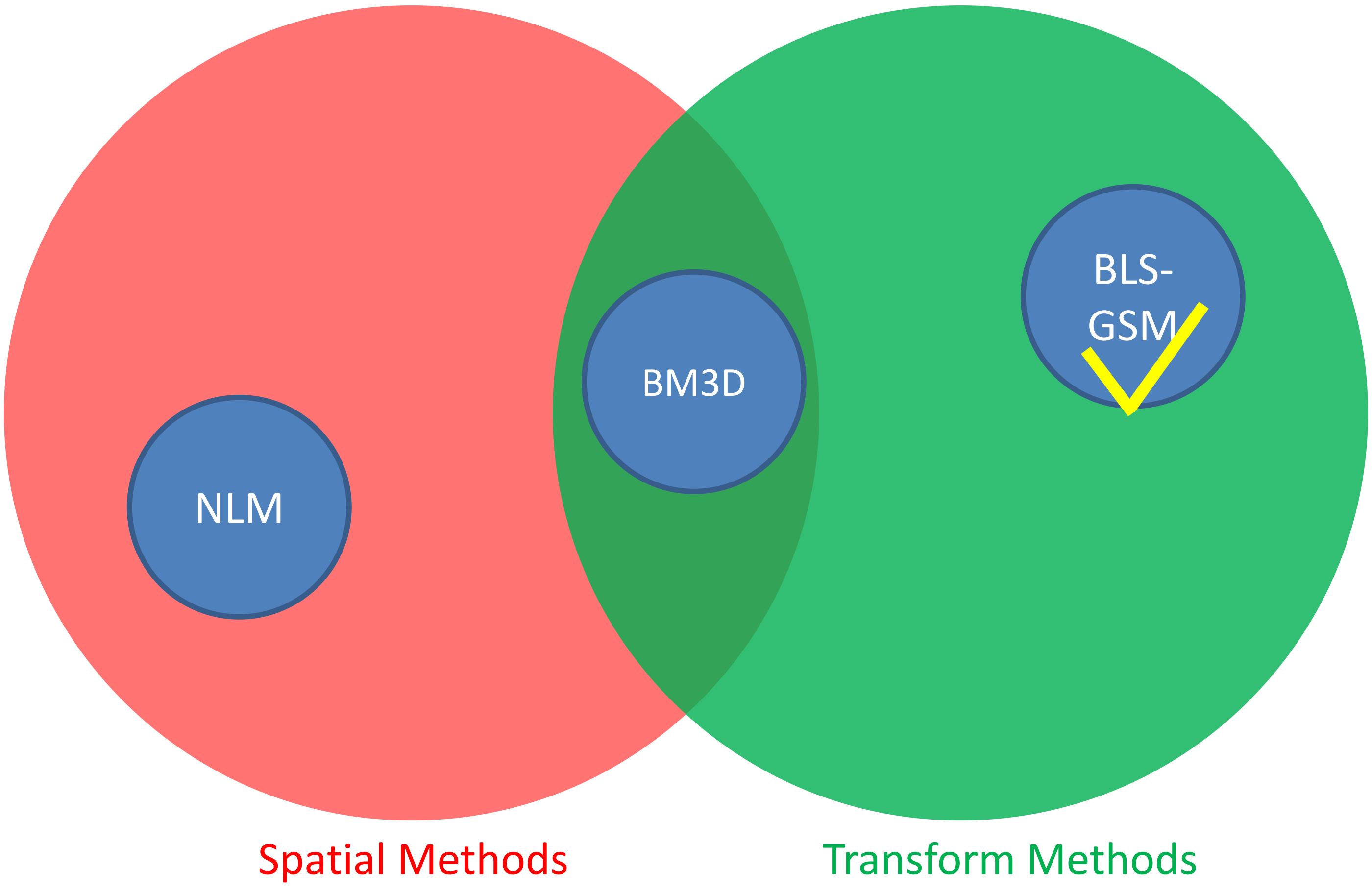
Weighted sum of local Wiener estimates

All z-explanations contribute to the estimate!

BLS-GSM-Wavelet Denoising



State of the art Methods



Motivation - Drawback of Locality

- Previous methods perform some local filtering

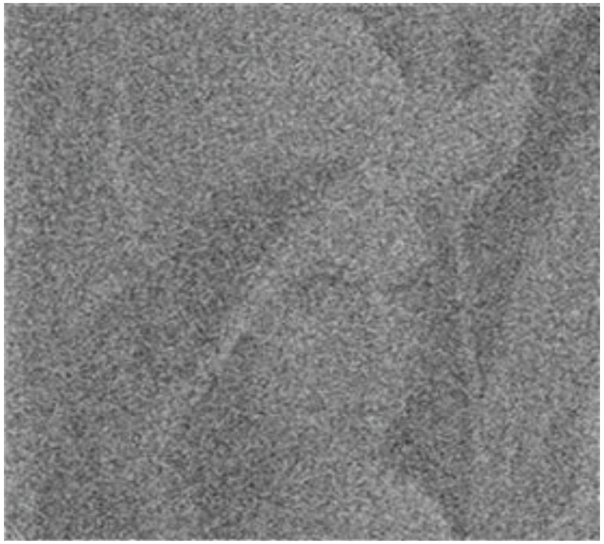
$\Rightarrow$ mixing of pixels from different statistics

$\Rightarrow$ blur

- Goal:
  Reduce the mixing $\Leftrightarrow$ "smarter" localization

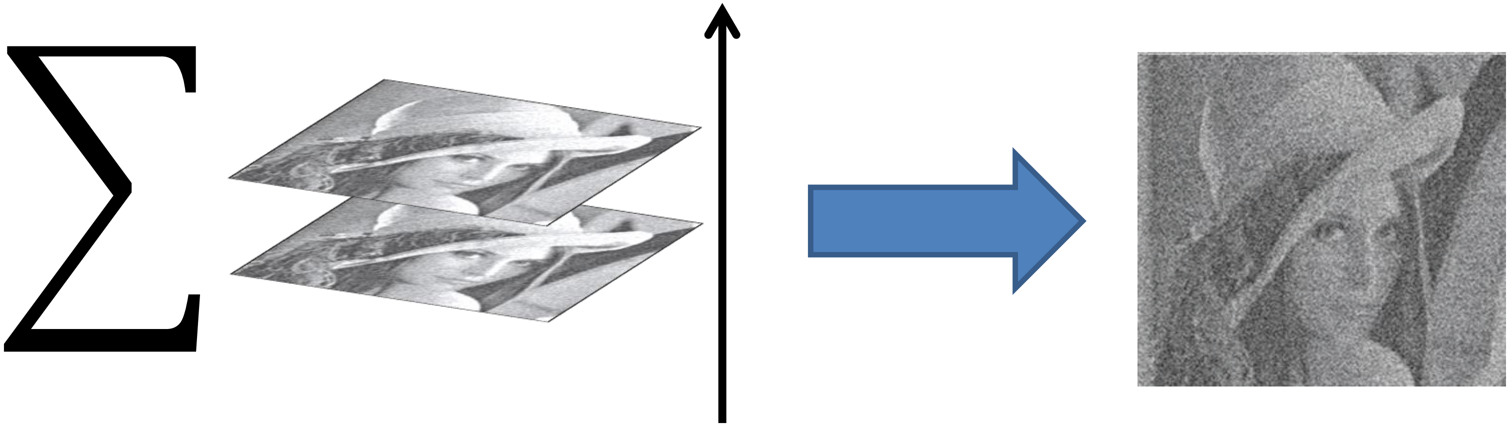
Motivation - Temporal perspective

- Assume a static scene
- Consider multiple images $y(t)$ at different times
- The signal $x(t)$ remains constant
- $n(t)$ varies over time with zero mean



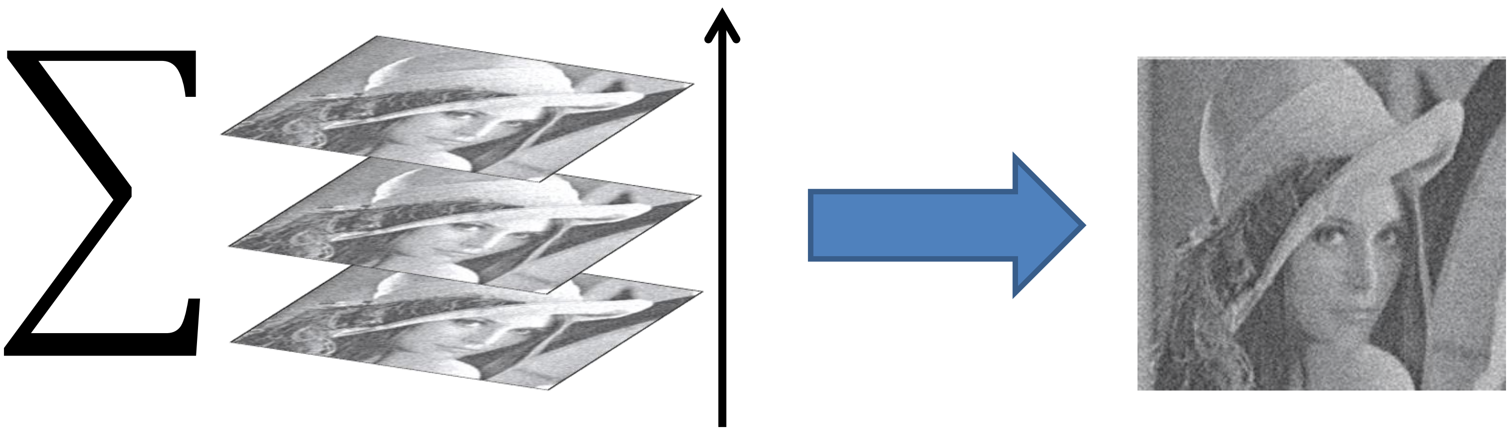
“Temporal Denoising”

Average multiple images over time



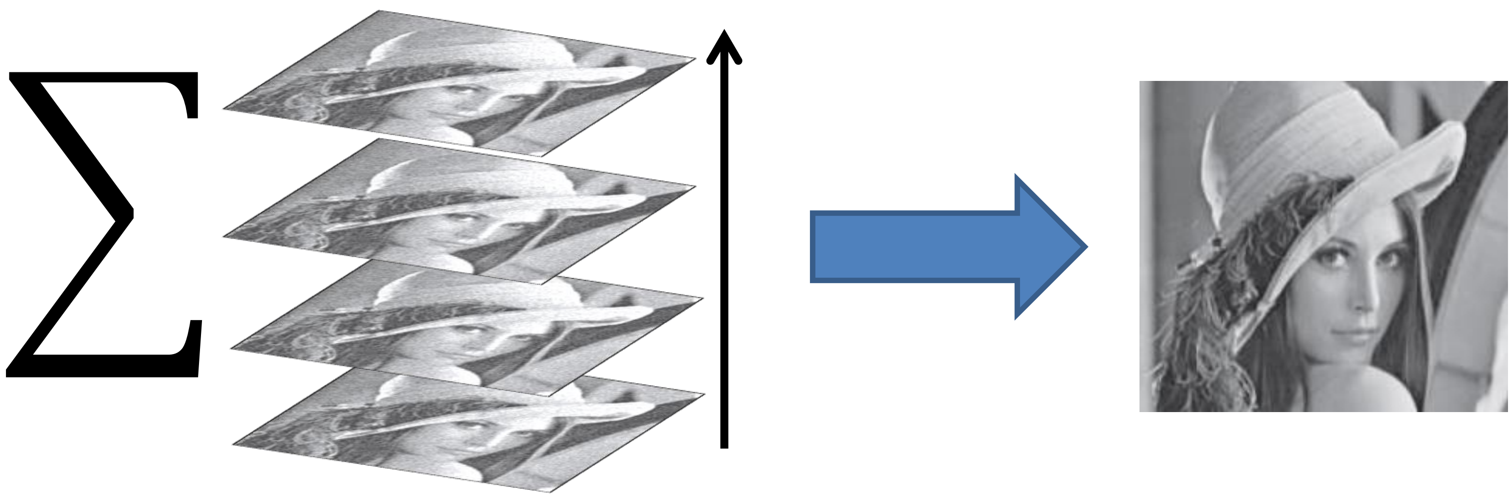
“Temporal Denoising”

Average multiple images over time

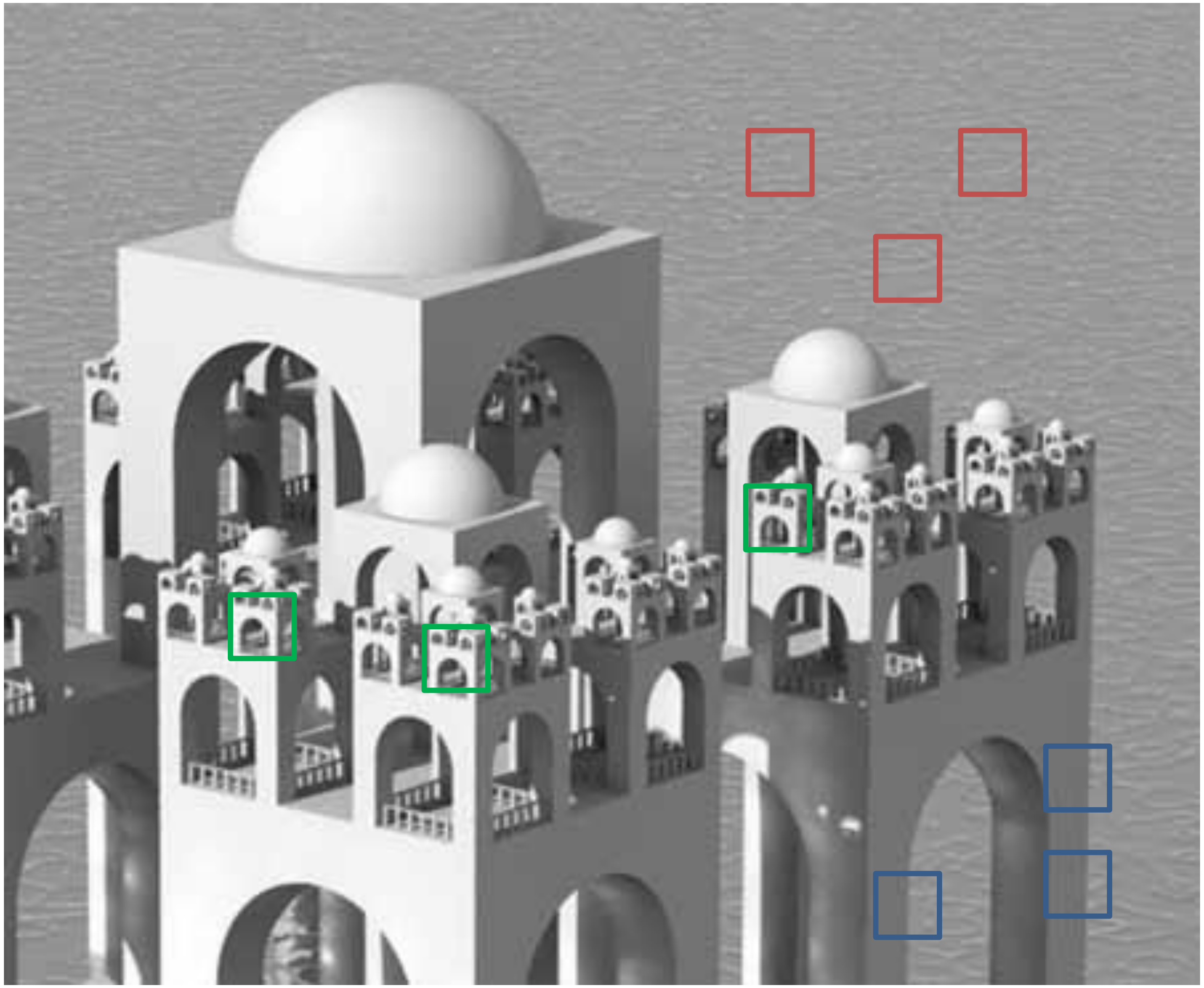


“Temporal Denoising”

Average multiple images over time

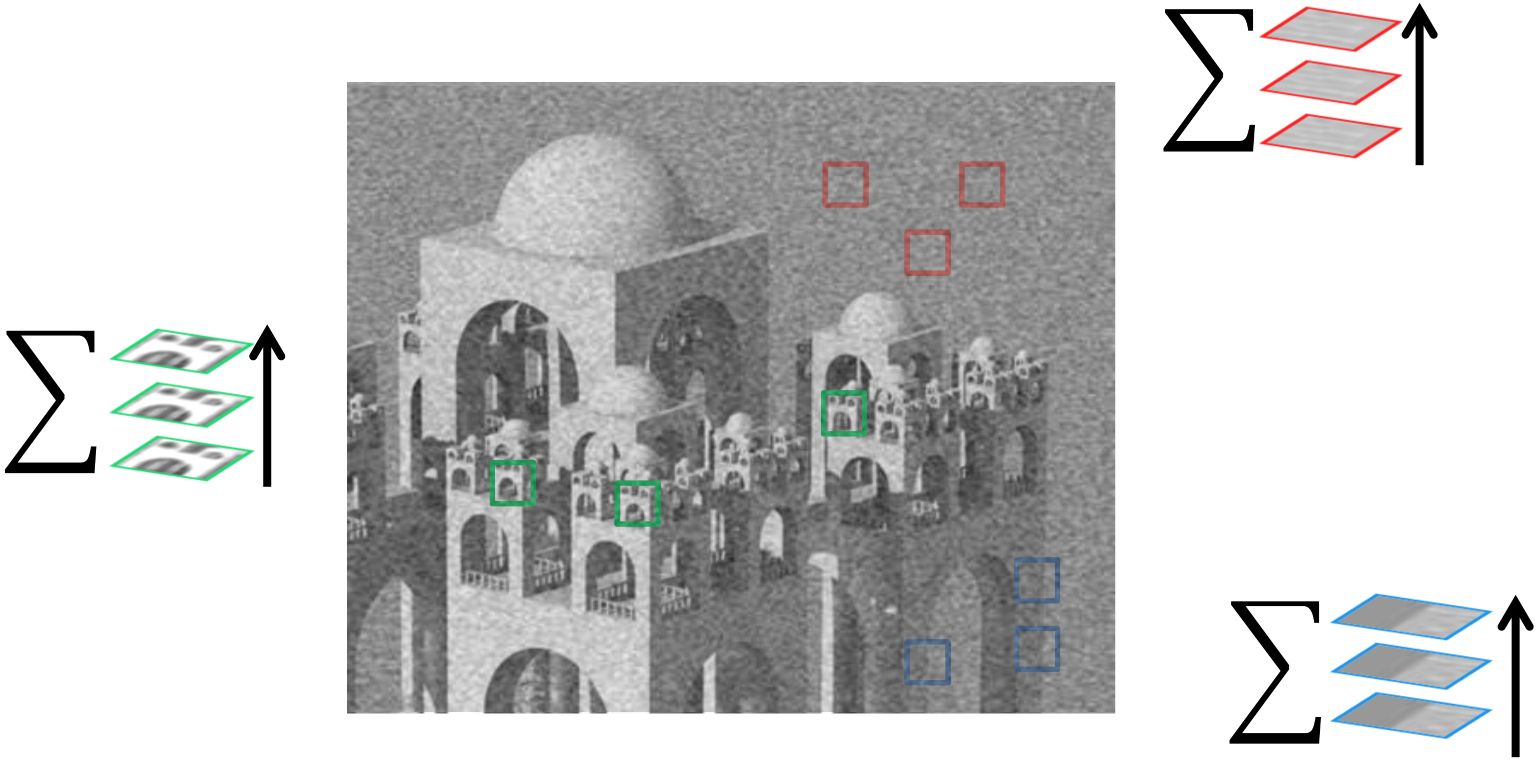


Redundancy in natural images



Glasner *et al*. (2009)

Single image “time-like” denoising



Unfortunately, patches are not exactly the same

⇒ simple averaging just won’t work

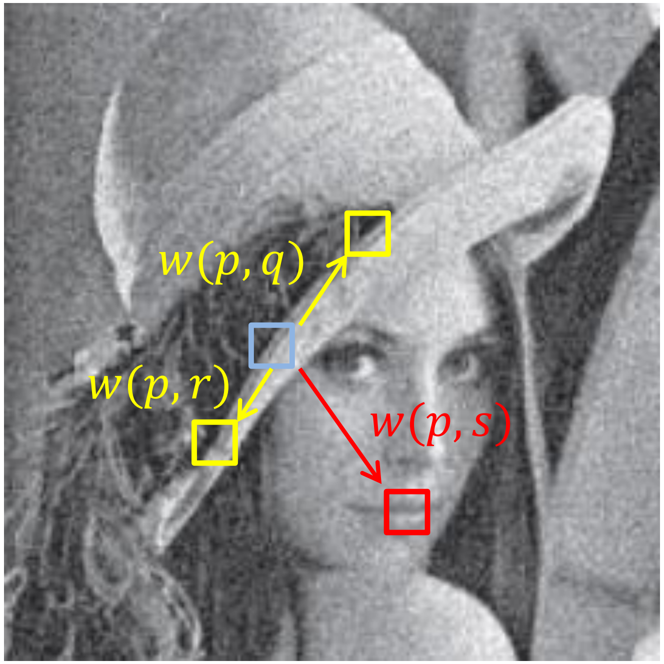
Non Local Means (NLM)

Baudes *et al*. (2005)

Use a weighted average based on similarity

$$\hat{x}(i) = \frac{1}{C_i} \sum_j y(j) \, e^{-\frac{GSSD\left(y(N_i) - y(N_j)\right)}{2\sigma^2}}$$

$w(i,j)$



From Bilateral Filter to NLM

$$\hat{x}(i)_{BL} = \frac{1}{C_i}\sum_j y(j)e^{-\frac{|y(i)-y(j)|^2}{2\sigma^2}} e^{-\frac{\|i-j\|^2}{2\rho^2}}$$

intensity weight spatial weight

$\rho \to \infty$

$$\hat{x}(i)_{NLM_{1x1}} = \frac{1}{C_i}\sum_j y(j)\, e^{-\frac{|y(i)-y(j)|^2}{2\sigma^2}}$$

From Bilateral Filter to NLM

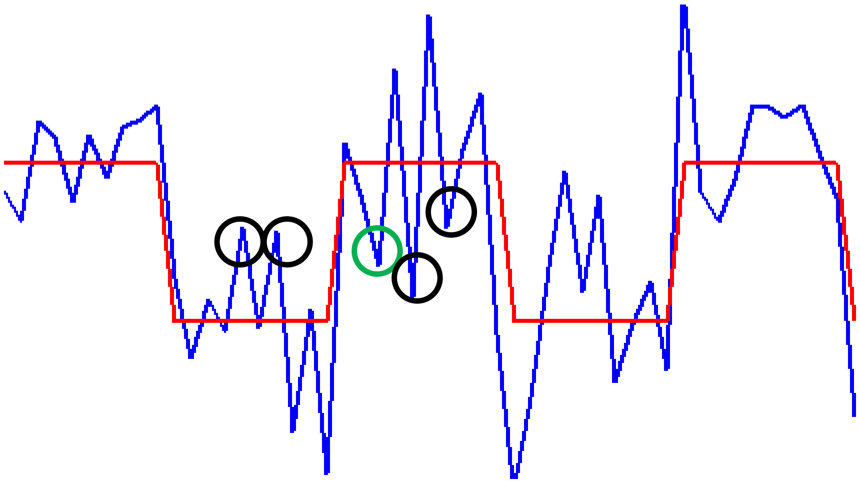
$$\hat{x}(i)_{NLM_{1x1}} = \frac{1}{C_i}\sum_j y(j)\, e^{-\frac{|y(i)-y(j)|^2}{2\sigma^2}}$$

Patch similarity

$$\hat{x}(i)_{NLM} = \frac{1}{C_i}\sum_j y(j)\, e^{-\frac{GSSD\left(y(N_i)-y\left(N_j\right)\right)}{2\sigma^2}}$$

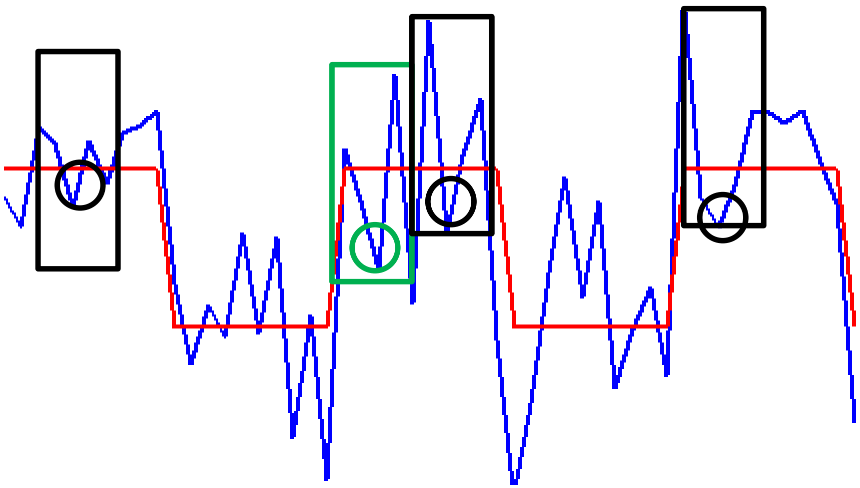
Why NLM is Better?

Bilateral Filtering



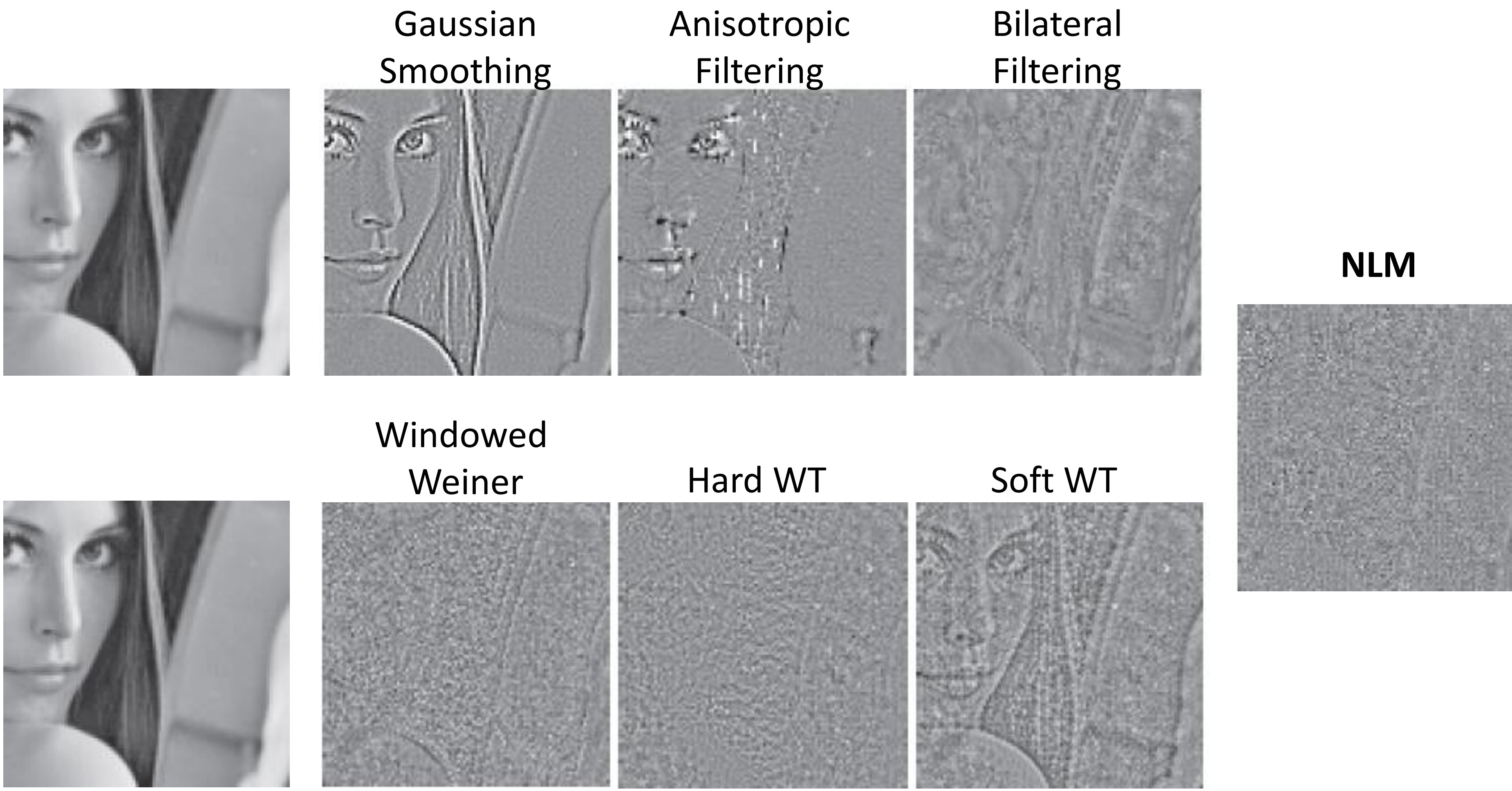
Mixing ⇒ bias

Non Local Means



No Mixing ⇒ Less bias

Performance Evaluation
Method Noise



Buades *et al*. (2005)

What’s Next?

- The idea of grouping sounds good

$\Rightarrow$ reduces mixing

- Denoise = “extract the common (the signal)”

- NLM: common = weighted average
- Can a sparser representation do better?

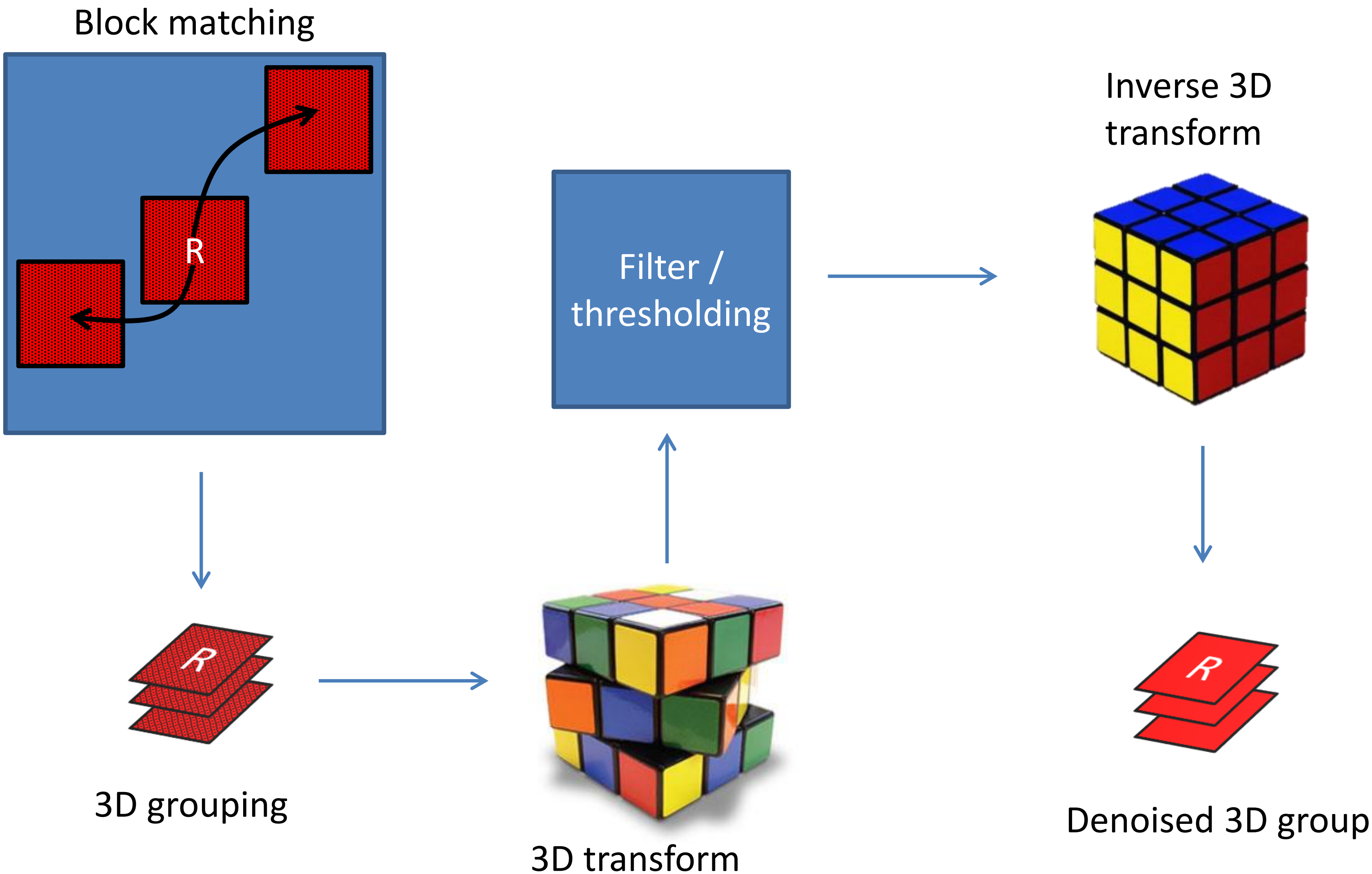
BM3D

Block Matching 3D collaborative filtering

(Dabov *et al*. 2007)

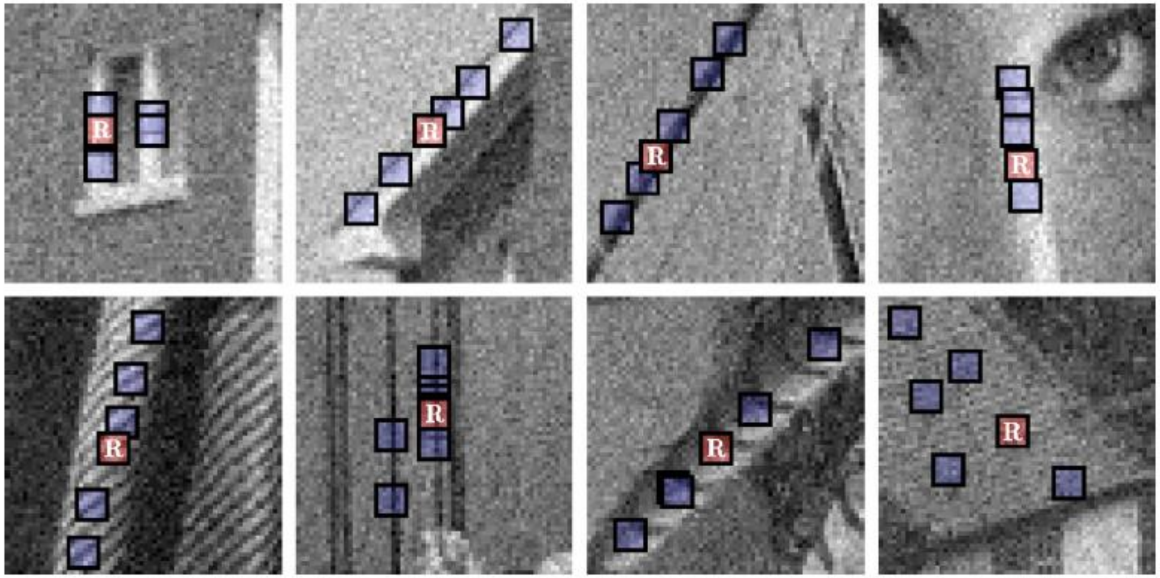
- Group patches with similar local structure (BM)
- Jointly denoise each group (3D)
- Smart Fusion of multiple estimates

A Single BM3D Estimate



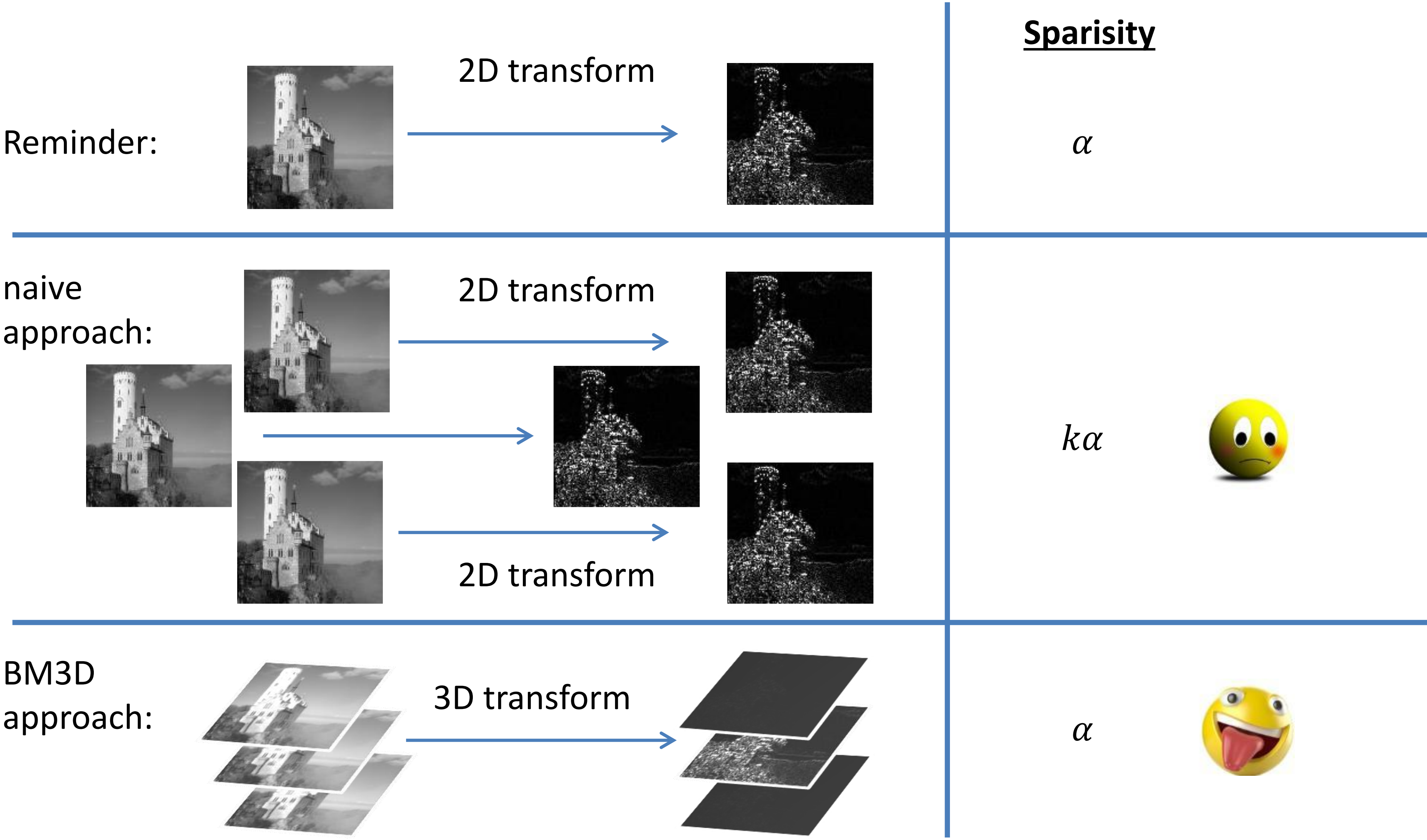
Grouping by Block Matching

- For every noisy reference block:
  - Calculate SSD between noisy blocks
  - If SSD<thr ⇒ add to group



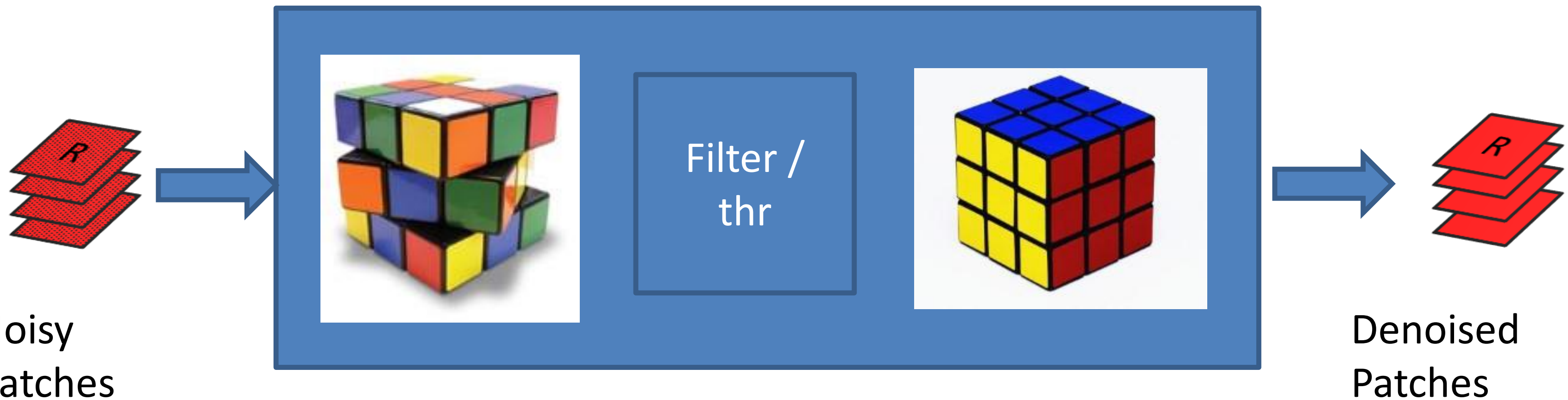
Dabov *et al*. (2007)

3D Transform



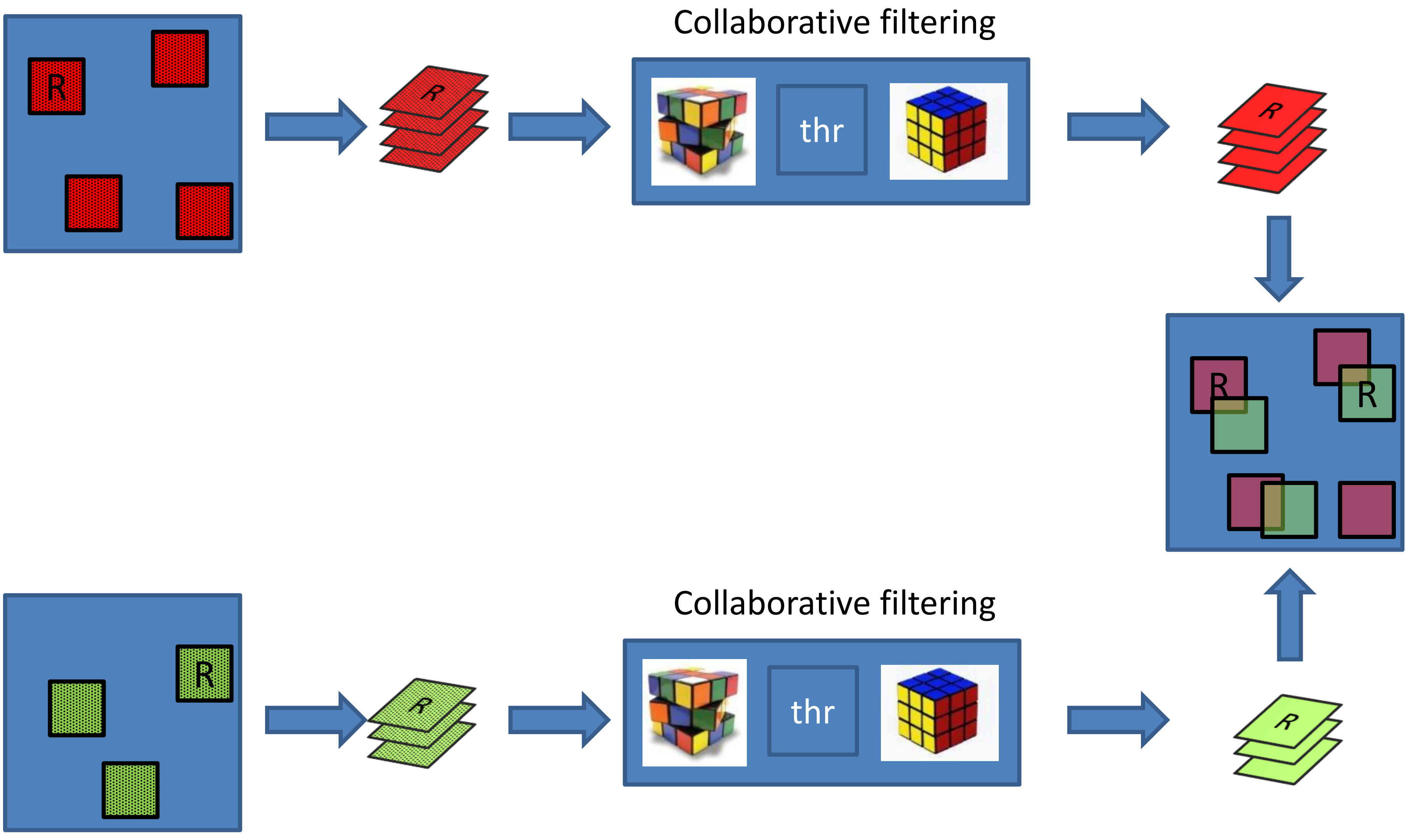
Collaborative Filtering

- Use hard thresholding or Wiener filter
- Each patch in the group gets a denoised estimate

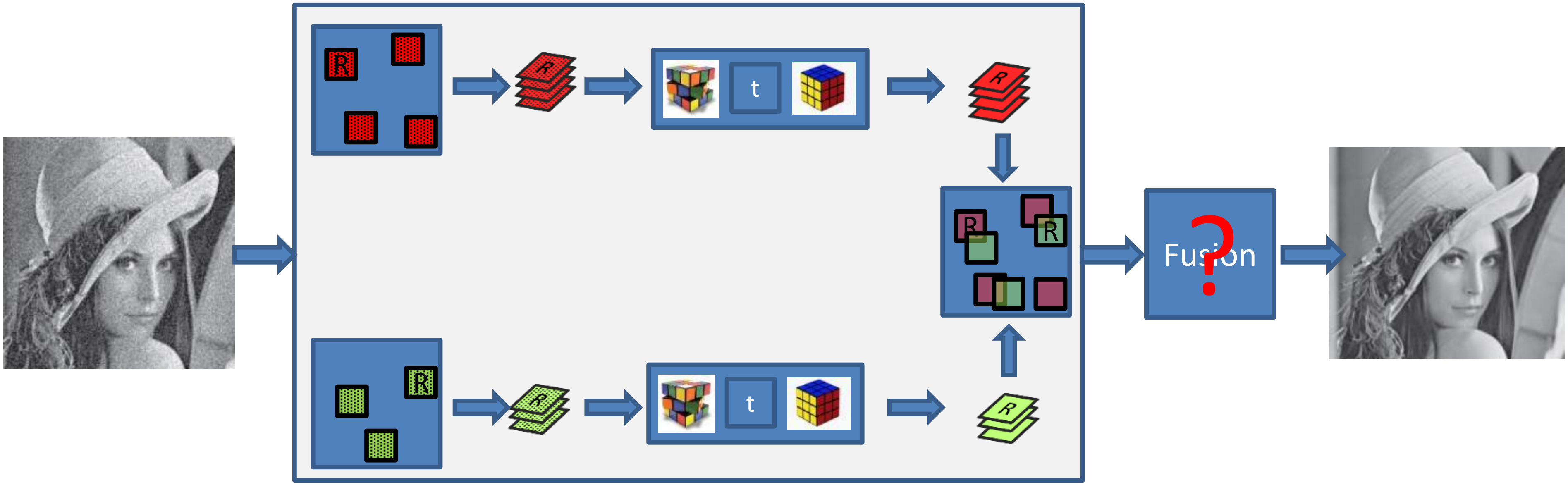


- Unlike NLM – where only central pixel in reference patch got an estimate

Multiple BM3D Estimates



Basic BM3D Denoiser



Fusion

- Each pixel gets multiple estimates from different groups

- Naive approach
  Average all estimates of each pixel
  …. not all estimates are as good

- Suggestion
  Give higher weight to more reliable estimates

BM3D - Fusion

- Give each estimate a weight according to denoising quality of its group
- Quality = Sparsity induced by the denoising

Hard thresholding

$$w \propto \frac{1}{\#Non\ Zero\ Coefficients}$$

Weiner filtering

$$w \propto \frac{1}{||Filter||^2}$$

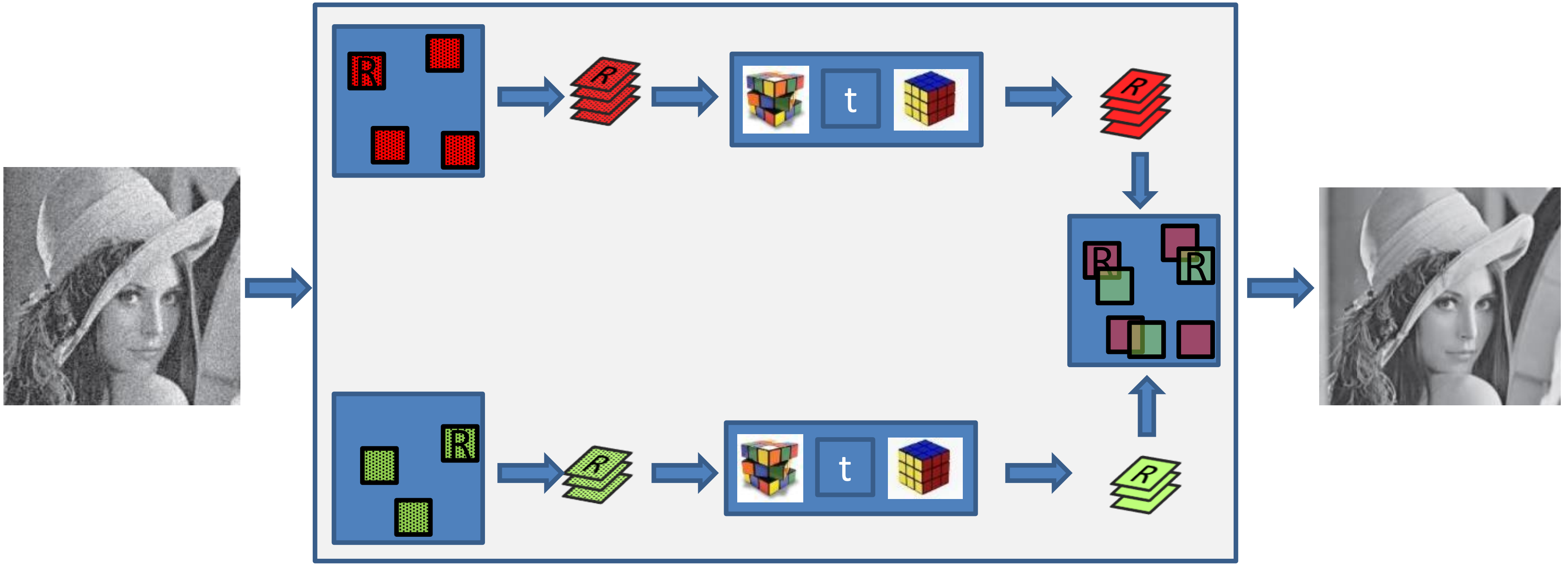
BM3D in Practice

- Noise may result in poor matching
  $\Rightarrow$ Degrades de-noising performance

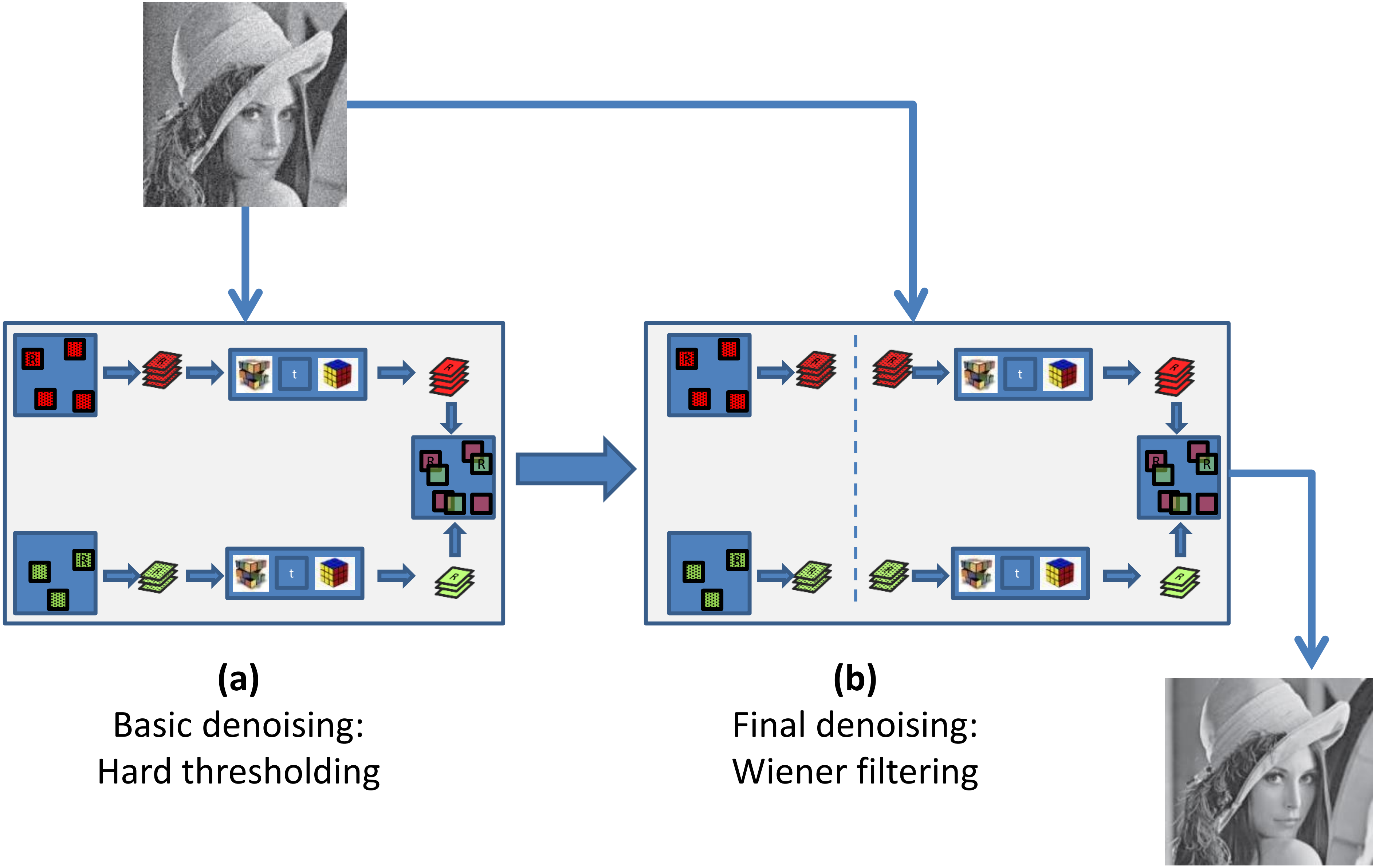
- Improvements:

1. Match using a smoothed version of the image
2. Perform BM3D in 2 phases:
   a. Basic BM3D estimate $\Rightarrow$ improved 3D groups
   b. Final BM3D

Basic BM3D Denoiser



Two phase BM3D Denoising



(a)
Basic denoising:
Hard thresholding

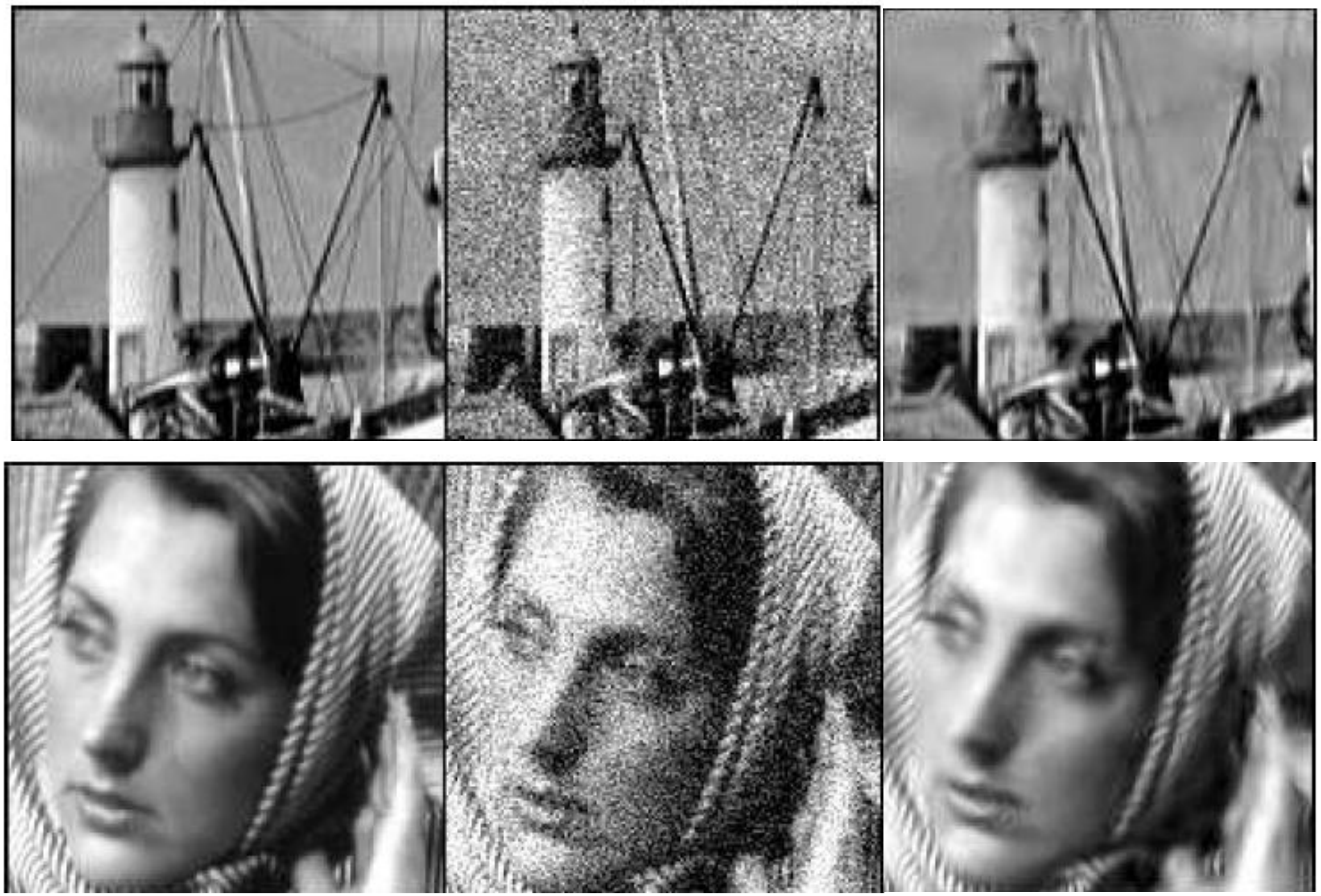
(b)
Final denoising:
Wiener filtering

Results and Comparison

- Comparison:
  - Different levels of noise
  - Different sets of images

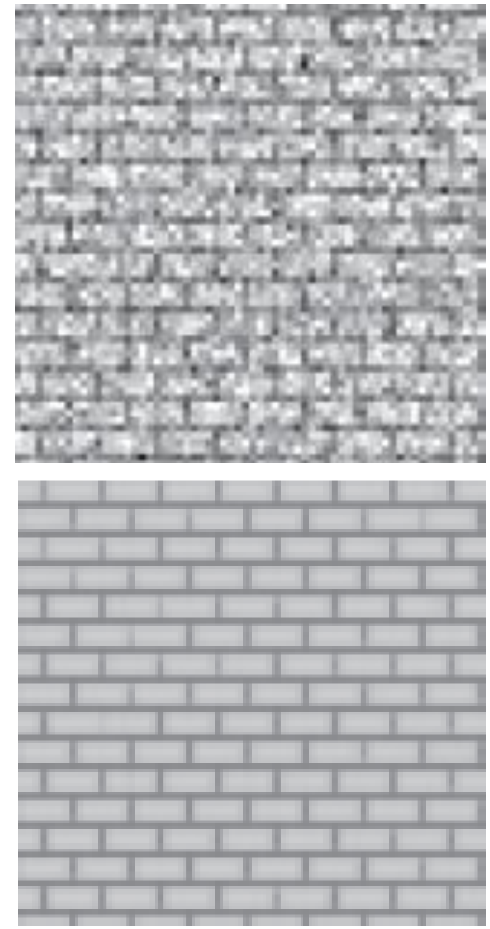
- Evaluation methods:
  - MSE/PSNR
  - Visual comparison to noisy and/or original images

GSM

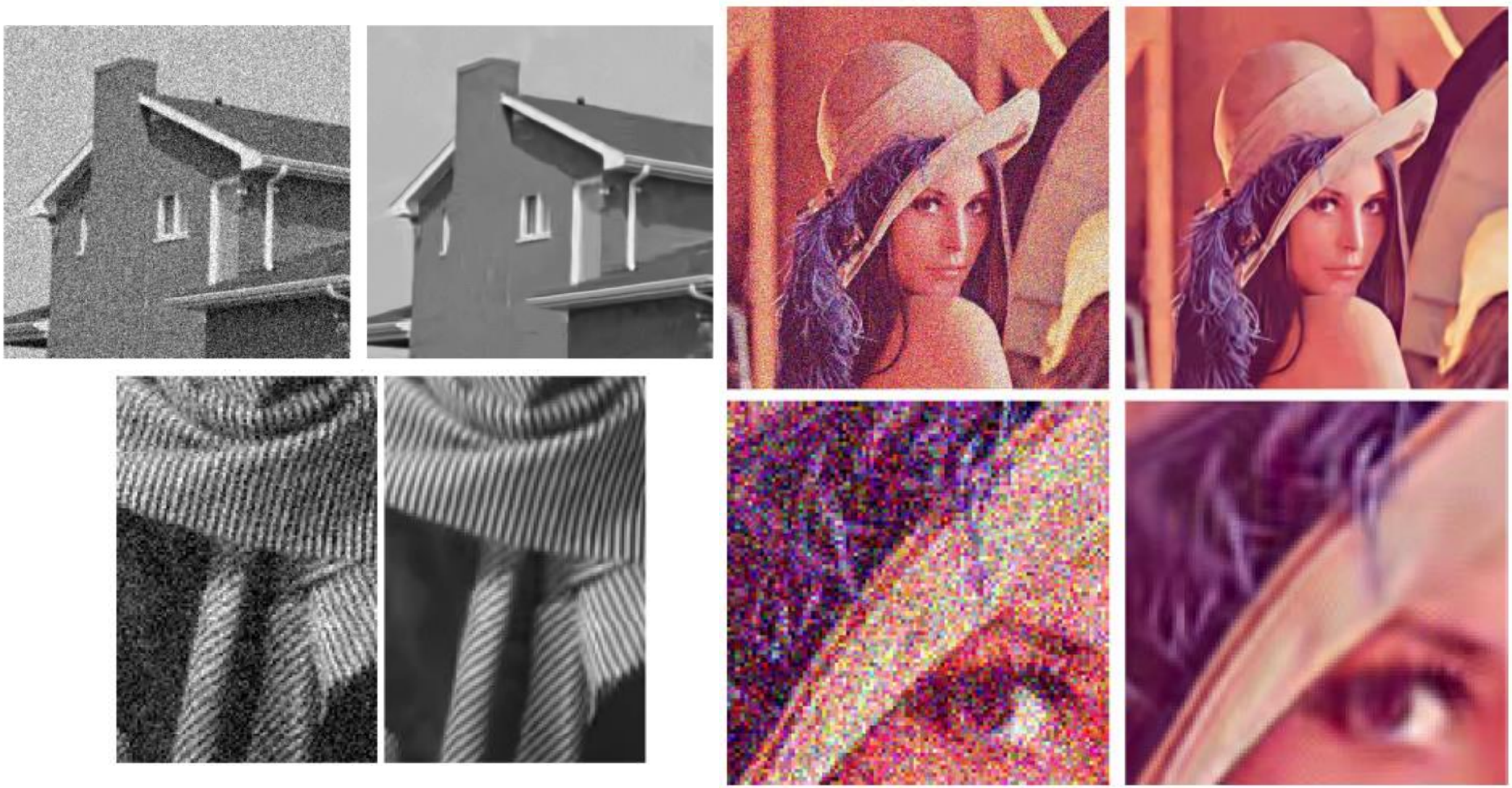


original noisy denoised

NLM



BM3D



Comparison

Image	σ	GF	AF	TV	YNF	EWF	TIHWT	NL-means	
Boat	8	53	38	39	39	33	28	23	
Lena	20	120	114	110	129	105	81	68	29.8dB
Barbara	25	220	216	186	176	111	135	72	29.5dB
Baboon	35	507	418	365	381	396	365	292	
Wall	35	580	660	721	598	325	712	59	

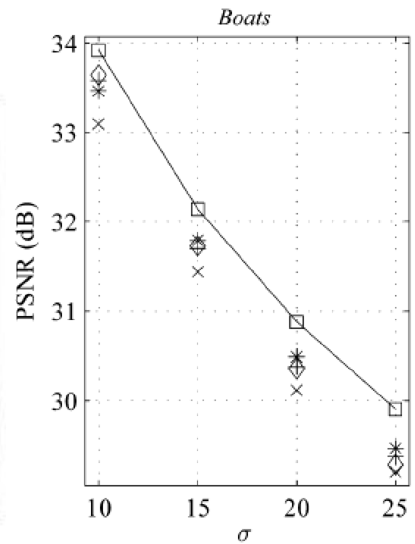
σ / PSNR	*Lena*	*Barb*	*Boats*	*Fgrpt*	*House*	*Peprs*
1 / 48.13	48.46	48.37	48.44	48.46	48.85	48.38
2 / 42.11	43.23	43.29	42.99	43.05	44.07	43.00
5 / 34.15	38.49	37.79	36.97	36.68	38.65	37.31
10 / 28.13	35.61	34.03	33.58	32.45	35.35	33.77
15 / 24.61	33.90	31.86	31.70	30.14	33.64	31.74
20 / 22.11	32.66	30.32	30.38	28.60	32.39	30.31
25 / 20.17	31.69	29.13	29.37	27.45	31.40	29.21
50 / 14.15	28.61	25.48	26.38	24.16	28.26	25.90
75 / 10.63	26.84	23.65	24.79	22.40	26.41	24.00
100 / 8.13	25.64	22.61	23.75	21.22	25.11	22.66

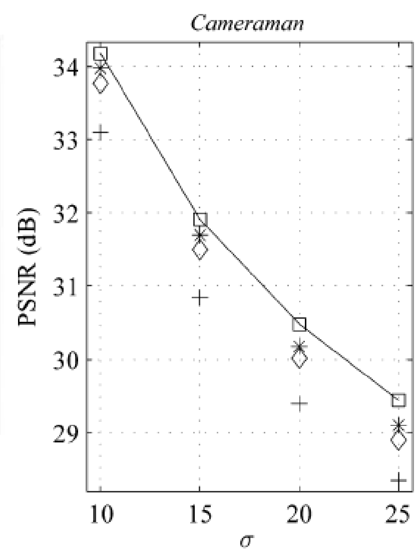
σ / PSNR	*C.man* 256^2	*House* 256^2	*Peppers* 256^2	*Montage* 256^2	*Lena* 512^2	*Barbara* 512^2	*Boats* 512^2	*F.print* 512^2	*Man* 512^2	*Couple* 512^2	*Hill* 512^2	*Lake* 512^2
2 / 42.11	43.96	44.63	43.48	46.47	43.59	43.66	43.18	42.90	43.61	43.17	43.04	43.02
5 / 34.16	38.29	39.83	38.12	41.14	38.72	38.31	37.28	36.51	37.82	37.52	37.14	36.58
10 / 28.14	34.18	36.71	34.68	37.35	35.93	34.98	33.92	32.46	33.98	34.04	33.62	32.85
15 / 24.61	31.91	34.94	32.70	35.15	34.27	33.11	32.14	30.28	31.93	32.11	31.86	31.08
20 / 22.11	30.48	33.77	31.29	33.61	33.05	31.78	30.88	28.81	30.59	30.76	30.72	29.87
25 / 20.18	29.45	32.86	30.16	32.37	32.08	30.72	29.91	27.70	29.62	29.72	29.85	28.94
30 / 18.59	28.64	32.09	29.28	31.37	31.26	29.81	29.12	26.83	28.86	28.87	29.16	28.18
35 / 17.25	27.93	31.38	28.52	30.46	30.56	28.98	28.43	26.09	28.22	28.15	28.56	27.50
50 / 14.16	25.84	29.37	26.41	27.35	28.86	27.17	26.64	24.36	26.59	26.38	27.08	25.78
75 / 10.63	24.05	27.20	24.48	25.04	27.02	25.10	24.96	22.68	25.10	24.63	25.58	24.11
100 / 8.14	22.81	25.50	22.91	23.38	25.57	23.49	23.74	21.33	23.97	23.37	24.45	22.91

Comparison

GSM

BM3D





BLS-GSM +

Exemplar based X

BM3D □

Comments

- Average improvement from naive Gaussian filtering to NLM – 4-5 dB
- Average improvement of 1 dB over 4 years (from BLS-GSM(2003) to BM3D(2007))
- Saturation in PSNR over the last 4 years (BM3D still considered state of the art)

Machine Learning for Noise Reduction

To support noise reduction or obtain original image from its noisy image (observed image), we can have **two learning-based approaches**:

(a) learning the denoising function F:

$x\_{original} = F(y\_{observed})$; $F = argmin_F \|F(Y) - X\|_2^2$

(b) learning the noise image (residual image) n:

$x\_{original} = y\_{observed} - n$.

$$\hat{x} = \arg\min_x \frac{1}{2\sigma^2} \|y - x\|^2 + \lambda \Phi(x), \qquad (1)$$

Machine Learning for Joint Noise Reduction and De-Mosaicking

To solve the joint demosaicking-denoising problem, one of the most frequently used approaches in the literature relies on the following linear observation model

$$\mathrm{y} = \mathrm{M}\mathrm{x} + \mathrm{n}, \qquad (1)$$

$$\mathrm{x}^\star = \arg\max_{\mathrm{x}} \log(p(\mathrm{x}|\mathrm{y})) = \arg\max_{\mathrm{x}} \log(p(\mathrm{y}|\mathrm{x})) + \log(p(\mathrm{x})), \qquad (2)$$

$$\mathbf{x}^\star = \arg\min_{\mathbf{x}} \frac{1}{2\sigma^2} \|\mathbf{y} - \mathbf{M}\mathbf{x}\|_2^2 + \phi(\mathbf{x}) \qquad (3)$$