# Tutorial 9: BM3D, L1-L2 Optimization

Sanketh Vedula
sanketh@cs.technion.ac.il

# Agenda

- L1-L2 Optimisation
  - Problem setting
  - Iterative Soft Thresholding Algorithm (ISTA)
  - Implementation: toy problem
- BM3D
  - The method

# Inverse problems

$$\mathbf{y} = \mathbf{D}\mathbf{x} + \eta$$

| If D is: | | Then recovering x is: |
|---|---|---|
| Focal blur | – | super-resolution |
| Motion-blur | – | deblurring |
| Missing values | – | inpainting |
| Identity | – | denoising |
| Projections | – | Tomography |

**Today: we want x to be sparse!**

# Sparse coding

$\mathrm{y} \in \mathbb{R}^n$ signal | observations

$\mathrm{D} \in \mathbb{R}^{n \times m} \ (n < m)$ dictionary | measurement matrix

$\mathrm{x} \in \mathbb{R}^m$ representation | measurements

$$(P_0^\epsilon): \qquad \min ||\mathbf{x}||_0, \text{ such that } \quad ||\mathbf{y} - \mathbf{D}\mathbf{x}||_2 < \epsilon$$

$$(P_0^\lambda): \qquad \arg\min_{\mathbf{x}} \ \frac{1}{2}||\mathbf{y} - \mathbf{D}\mathbf{x}||_2^2 + \lambda||\mathbf{x}||_0$$

Applications: many applications in image and signal processing;
CT, MRI, single-pixel camera, compressed sensing

# Sparse coding

$$(P_0^\lambda): \qquad \arg\min_{\mathbf{x}} \frac{1}{2}||\mathbf{y} - \mathbf{D}\mathbf{x}||_2^2 + \lambda||\mathbf{x}||_0$$

**NP-Hard problem!**
**Exponentially increasing search space with the dimension.**

**Orthogonal Matching Pursuit:**

init: support = { }, residual (r) = y
Until reaching the desired level of sparsity:

- Choose an unselected atom/column from D that minimises || r - Dx ||
- Add the chosen atom (d_i) into the support => support = support + {d_i}
- Update the residual (r) => r = || y - D x{d_i} ||
- Repeat until reaching desired level of sparsity or when r is close to 0

**Greedy scheme!**

**But it is guaranteed to recover the support perfectly if x is sufficiently sparse.**

More on this: Elad, 2009

# (Relaxed) Sparse coding

Instead of using greedy schemes, we can rather relax our problem in the following way and use our favourite convex optimization solvers to recover x

$$(P_1^\lambda): \qquad \arg\min_{\mathbf{x}} \ \frac{1}{2}||\mathbf{y} - \mathbf{D}\mathbf{x}||_2^2 + \lambda||\mathbf{x}||_1$$

- Pursuit algorithms: Basis Pursuit, ISTA (proximal gradient)
- Acceleration schemes: FISTA, SESOP

$$\mathbf{D} = \Phi \qquad \mathbf{x} = \mathbf{c}$$

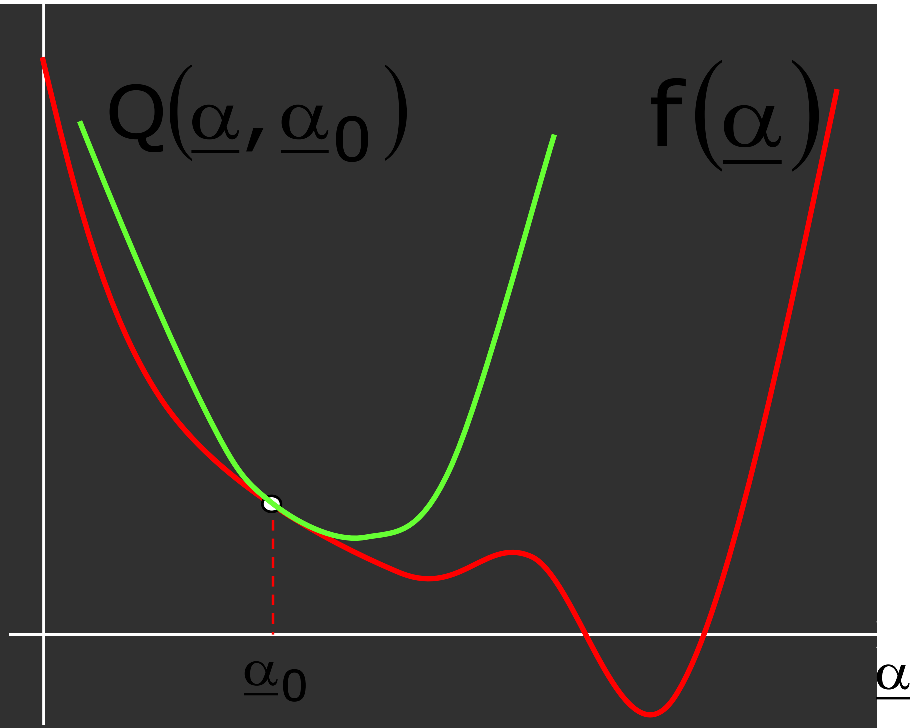
# ISTA

$$\arg\min_{\mathbf{c}} \frac{1}{2}\|y - \Phi\mathbf{c}\|_2^2 + \lambda\|\mathbf{c}\|_1$$

$$= \arg\min_{\mathbf{c}} \frac{1}{2}\langle y - \Phi\mathbf{c}, y - \Phi\mathbf{c}\rangle + \lambda\|\mathbf{c}\|_1$$

$$= \arg\min_{\mathbf{c}} \frac{1}{2}\langle \Phi\mathbf{c}, \Phi\mathbf{c}\rangle - \langle \Phi\mathbf{c}, y\rangle + \frac{1}{2}\langle y, y\rangle + \lambda\|\mathbf{c}\|_1$$

$$= \arg\min_{\mathbf{c}} \frac{1}{2}\langle \Phi^*\Phi\mathbf{c}, \mathbf{c}\rangle - \langle \Phi^* y, \mathbf{c}\rangle + \lambda\|\mathbf{c}\|_1$$




$$\arg\min_{\mathbf{c}} \underbrace{\frac{1}{2}\langle \Phi^*\Phi\mathbf{c}, \mathbf{c}\rangle - \langle \Phi^* y, \mathbf{c}\rangle}_{\text{convex \& smooth}} + \underbrace{\lambda\|\mathbf{c}\|_1}_{\text{convex \& non- smooth}} = \arg\min_{\mathbf{c}} g(\mathbf{c}) + h(\mathbf{c})$$

Smooth + non-smooth optimisation problem

Fix $\mathbf{c}$ and approximate $g(\mathbf{c})$ around it

Second-order with fixed curvature $1/\eta$

$$g(\mathbf{u} - \mathbf{c}) \approx g(\mathbf{c}) + \langle \nabla g(\mathbf{c}), \mathbf{u} - \mathbf{c}\rangle + \frac{1}{2\eta}\|\mathbf{u} - \mathbf{c}\|_2^2$$

# ISTA

$$\arg\min_{\mathbf{c}} g(\mathbf{c}) + h(\mathbf{c})$$

$$h(\mathbf{c}) = \lambda \|\mathbf{c}\|_1$$

$$\approx \arg\min_{\mathbf{u}} g(\mathbf{u} - \mathbf{c}) + h(\mathbf{u})$$

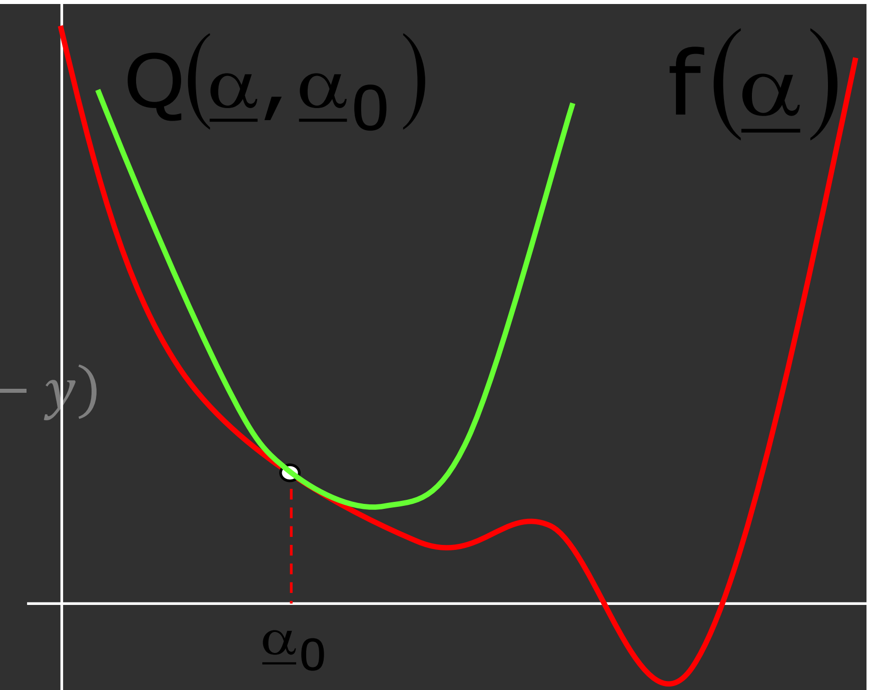
$$= \arg\min_{\mathbf{u}} \frac{1}{2\eta} \|\mathbf{u} - \mathbf{c} + \eta \nabla g(\mathbf{c})\|_2^2 + h(\mathbf{u})$$

$$\nabla g(\mathbf{c}) = \Phi^*(\Phi \mathbf{c} - y)$$

$$= \arg\min_{\mathbf{u}} \frac{1}{2} \|\mathbf{u} - \mathbf{z}\|_2^2 + \eta\lambda \|\mathbf{u}\|_1$$

$$\mathbf{z} = \mathbf{c} - \eta \nabla g(\mathbf{c})$$

$$= \mathbf{c} - \eta \Phi^*(\Phi \mathbf{c} - y)$$




$$= \arg\min_{\mathbf{u}} \sum_{\mathbf{n} \in \mathbb{Z}^d} \frac{1}{2} (u_{\mathbf{n}} - z_{\mathbf{n}})^2 + \eta\lambda |u_{\mathbf{n}}|$$

$$= \left\{ \arg\min_{u_{\mathbf{n}}} \frac{1}{2} (u_{\mathbf{n}} - z_{\mathbf{n}})^2 + \eta\lambda |u_{\mathbf{n}}| \right\}_{\mathbf{n} \in \mathbb{Z}^d}$$

Reduced to one-dimensional minimization problems

$$\min_{u} \frac{1}{2} (u - z)^2 + \mu |u|$$

First term smooth with derivative $g'(u) = u - z$

Second term non-smooth with sub-differential set

$$\partial h(u) = \mu \begin{cases} \operatorname{sign} u & : u \neq 0 \\ [-1,1] & : u = 0 \end{cases}$$

**Optimality condition:** $0 \in g'(u^*) + \partial h(u^*)$

# ISTA

$$\min_{u} \frac{1}{2}(u-z)^2 + \mu|u|$$

Sub-differential set

$$g'(u) = u - z \qquad \partial h(u) = \mu \begin{cases} \operatorname{sign} u & : u \neq 0 \\ [-1,1] & : u = 0 \end{cases}$$

For the optimality to hold, both should satisfy

**Optimality condition:**

$$0 \in g'(u^*) + \partial h(u^*) = \begin{cases} \{u^* - z + \mu \operatorname{sign} u^*\} & : u^* \neq 0 \\ [u^* - z - \mu, u^* - z + \mu] & : u^* = 0 \end{cases}$$

$$\arg\min_{u} \frac{1}{2}(u-z)^2 + \mu|u|$$

$$= \begin{cases} z - \mu & : z > +\mu \\ 0 & : -\mu \leq z \leq +\mu \\ z + \mu & : z < -\mu \end{cases}$$

Then u has the following solution

$$= \mathcal{S}_\mu(z)$$

# ISTA

$$\arg\min_{\mathbf{u}} g(\mathbf{u}-\mathbf{c}) + h(\mathbf{u})$$

$$= \left\{\arg\min_{u_{\mathbf{n}}} \frac{1}{2}(u_{\mathbf{n}} - z_{\mathbf{n}})^2 + \eta\lambda|u_{\mathbf{n}}| \right\}_{\mathbf{n}\in\mathbb{Z}^d}$$

$$= \left\{\mathcal{S}_{\eta\lambda}(z_{\mathbf{n}})\right\}_{\mathbf{n}\in\mathbb{Z}^d}$$

$$= \mathcal{S}_{\eta\lambda}(\mathbf{z})$$

element-wise application

$$= \mathcal{S}_{\eta\lambda}\left(\mathbf{c} - \eta\Phi^*(\Phi\mathbf{c} - y)\right)$$

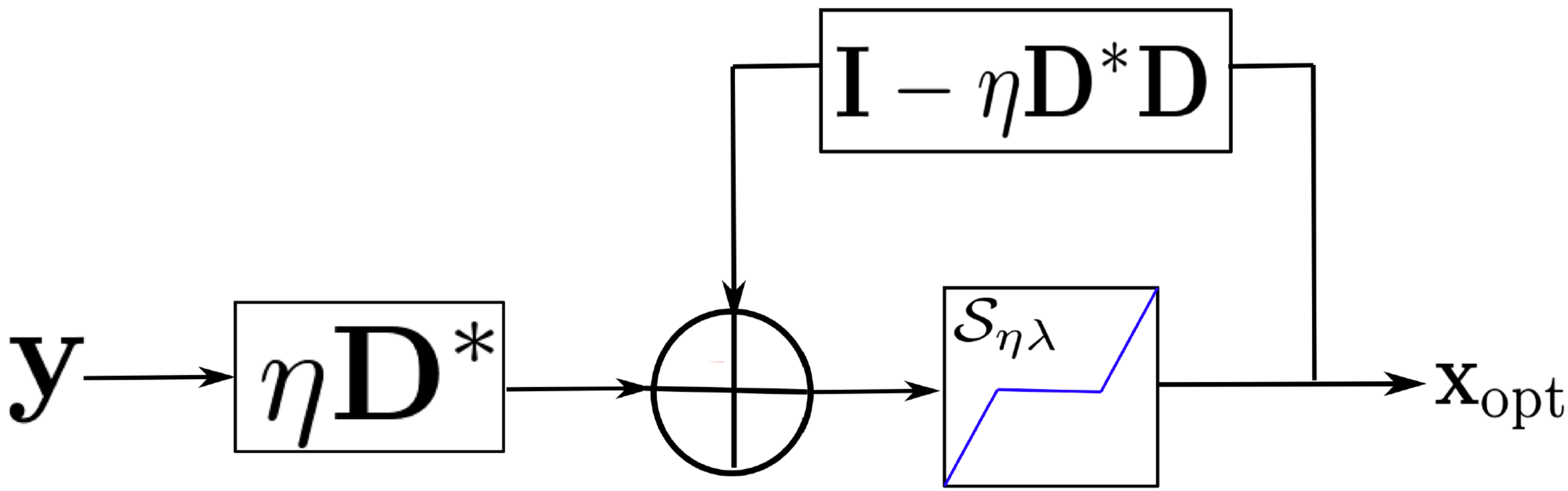
grad. step on $g$

$$\mathbf{z} = \mathbf{c} - \eta\Phi^*(\Phi\mathbf{c} - y)$$

- Start with initial $\mathbf{c}^0$
- For $k = 0,1,\dots,$ until convergence
  - Iterate $\mathbf{c}^{k+1} = \mathcal{S}_{\eta\lambda}\left(\mathbf{c} - \eta\Phi^*\left(\Phi\mathbf{c}^k - y\right)\right)$

# ISTA

PyNotebook

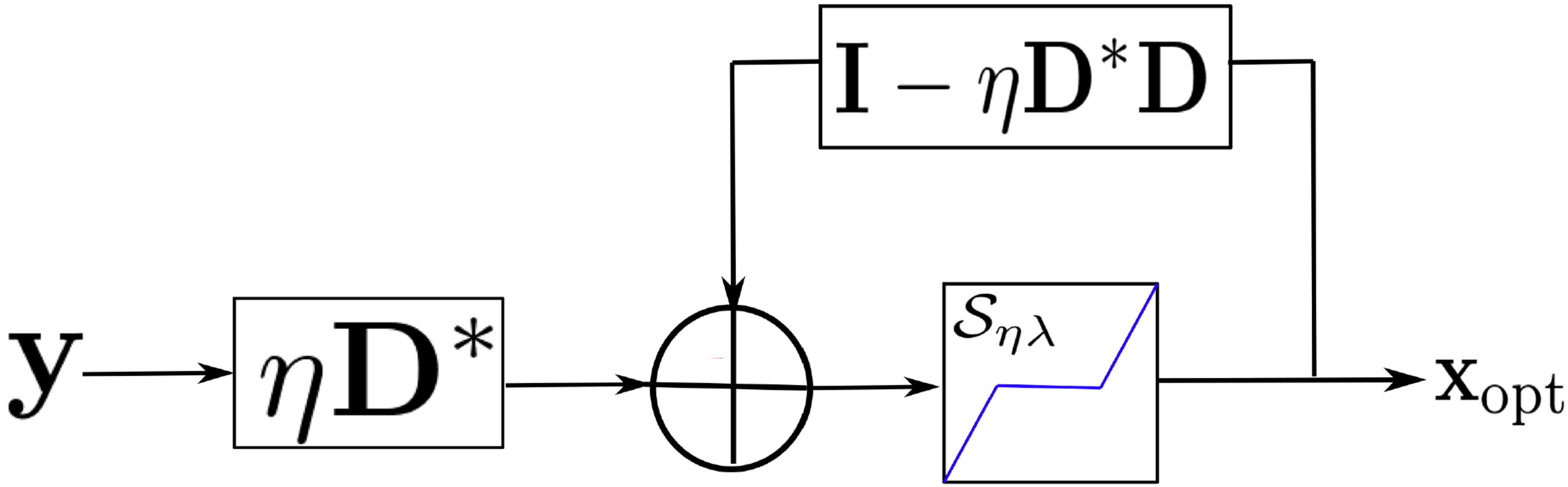



Update scheme:

$$\mathbf{x}^{k+1} = \mathcal{S}_{\eta\lambda}(\mathbf{x}^k - \eta\mathbf{D}^*(\mathbf{D}\mathbf{x}^k - \mathbf{y}))$$

# FISTA

PyNotebook



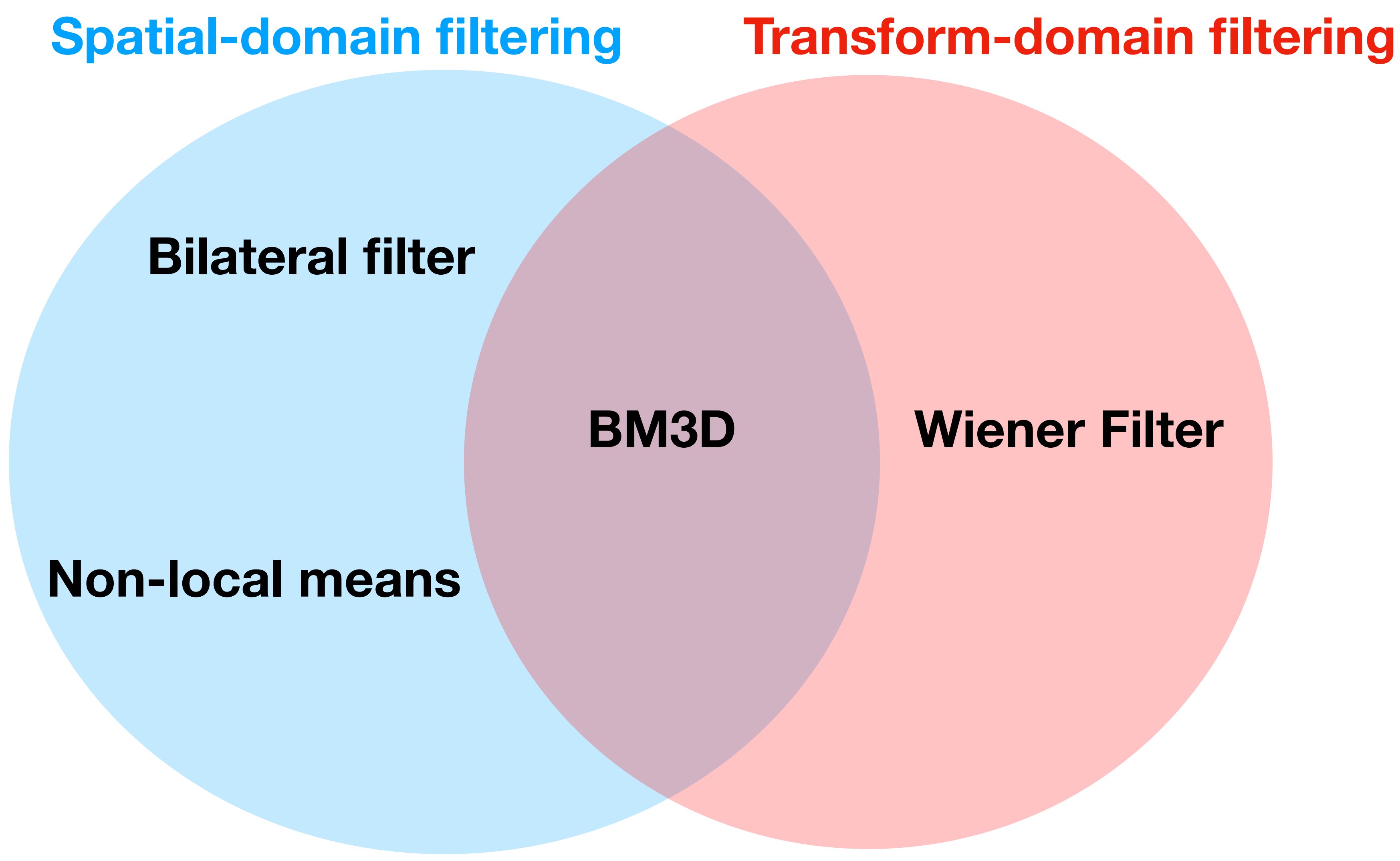

Update scheme:

$$\mathbf{x}^{k+1} = \mathcal{S}_{\eta\lambda}(\mathbf{x}^k - \eta\mathbf{D}^*(\mathbf{D}\mathbf{x}^k - \mathbf{y}))$$

$$t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2} \qquad \mathbf{x}^{k+1} = \mathbf{x}^k + \frac{t_k - 1}{t_{k+1}}(\mathbf{x}^k - \mathbf{x}^{k-1})$$

Beck and Teboulle, 2009

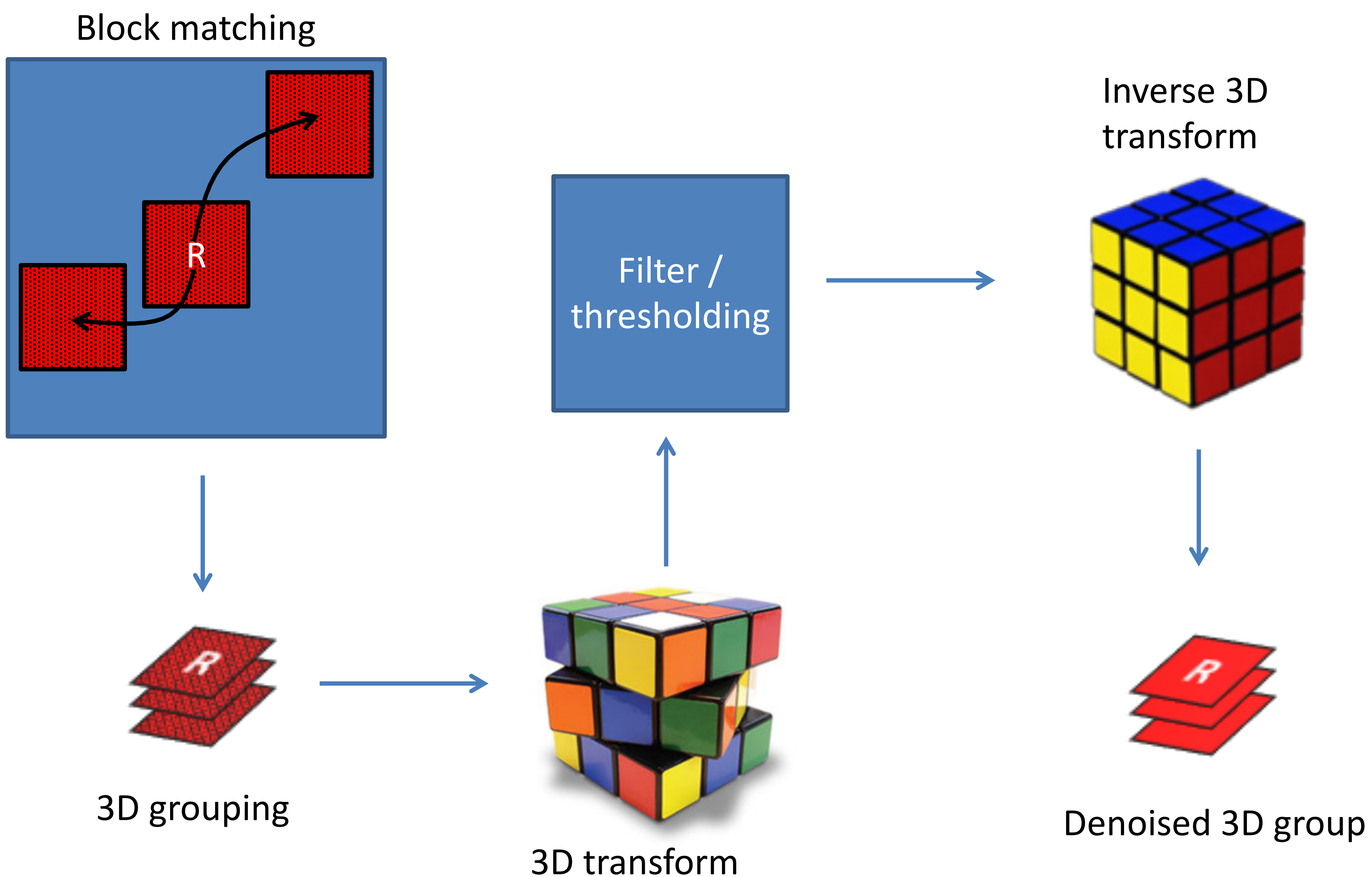
# Image denoising



# BM3D

- BM3D = Block Matching 3D Collaborative Filtering
- Ideas:
    - Group patches with similar local structure (BM)
    - Jointly denoise each group (3D)
    - Smart fusion of the estimates

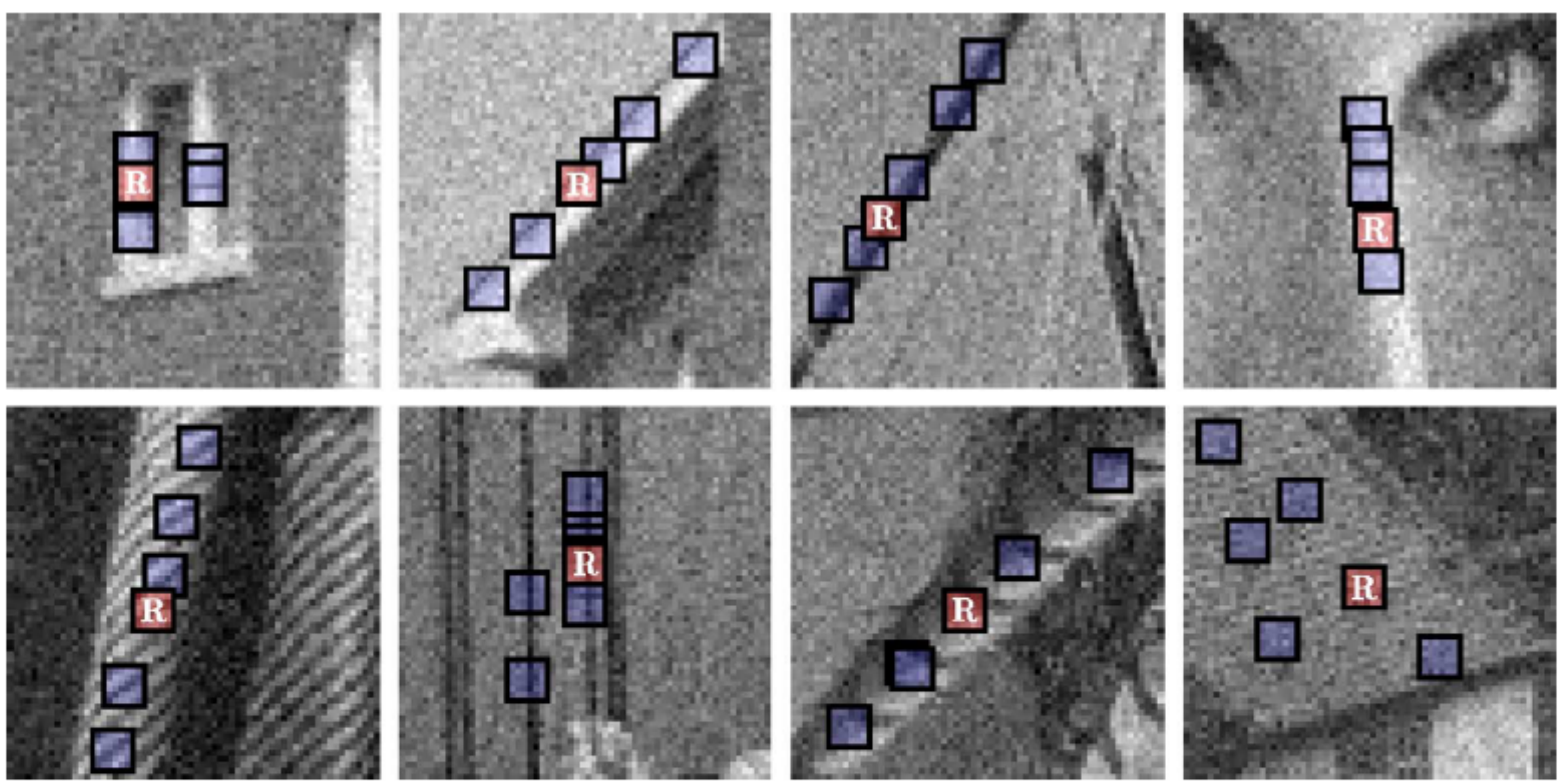
# BM3D - 1st round




Dabov et al., 2007

# Grouping by block matching

- For every reference block:
  - Calculate SSD (sum of squared difference) between noisy blocks
  - If SSD < Threshold => add it to the group



Dabov et al., 2007

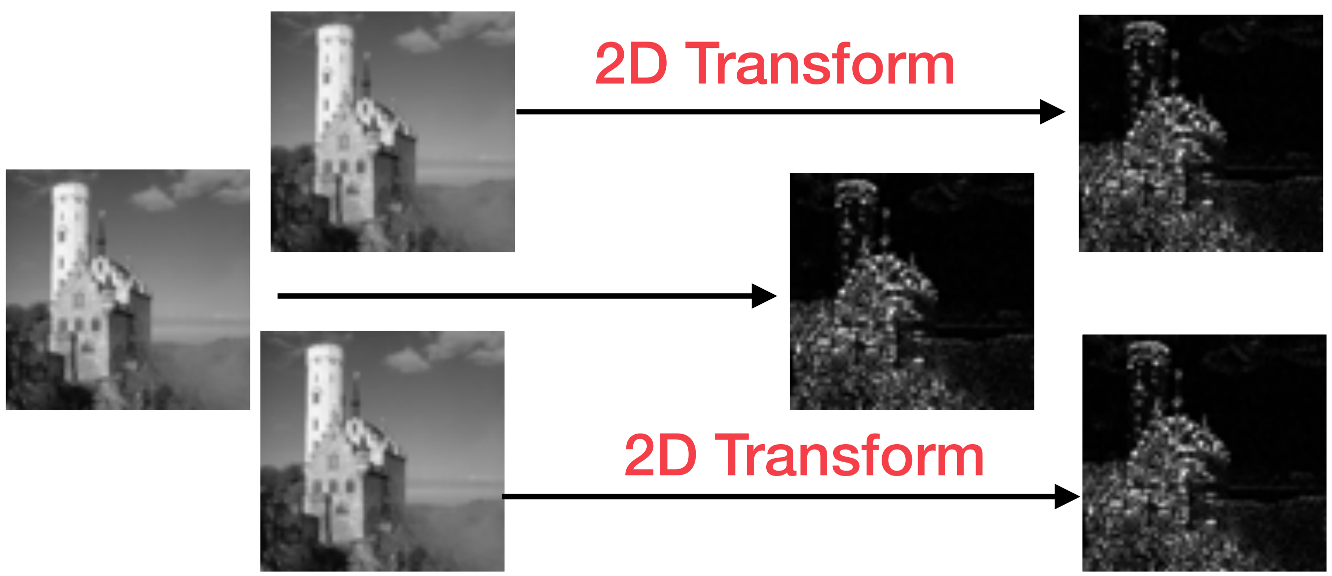
# 3D Transform

Sparsity induced

Reminder:


$$\alpha$$

Naive:



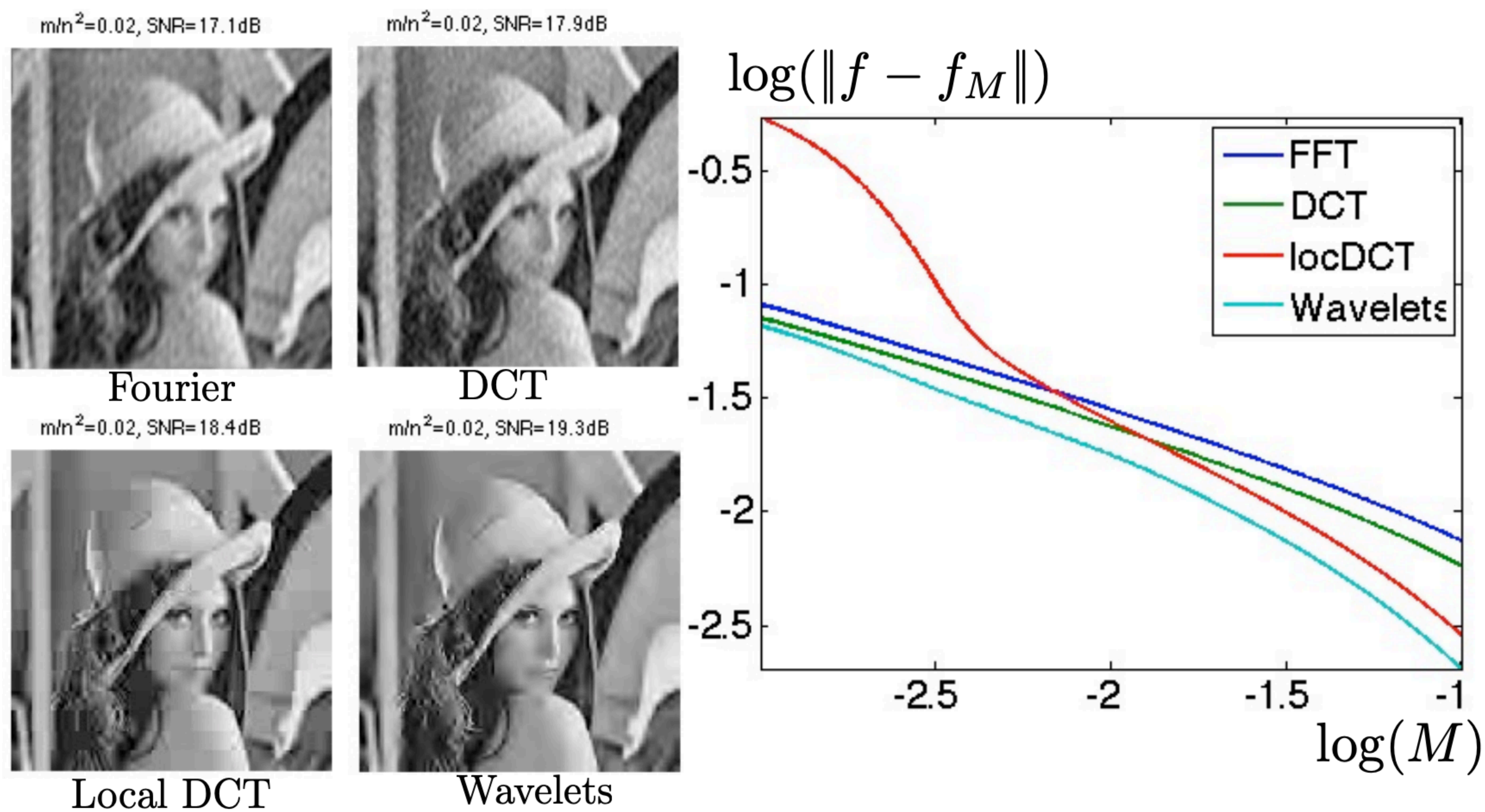

$$k\alpha$$

BM3D:


$$\alpha$$

Dabov et al., 2007

# Which transform domain ?

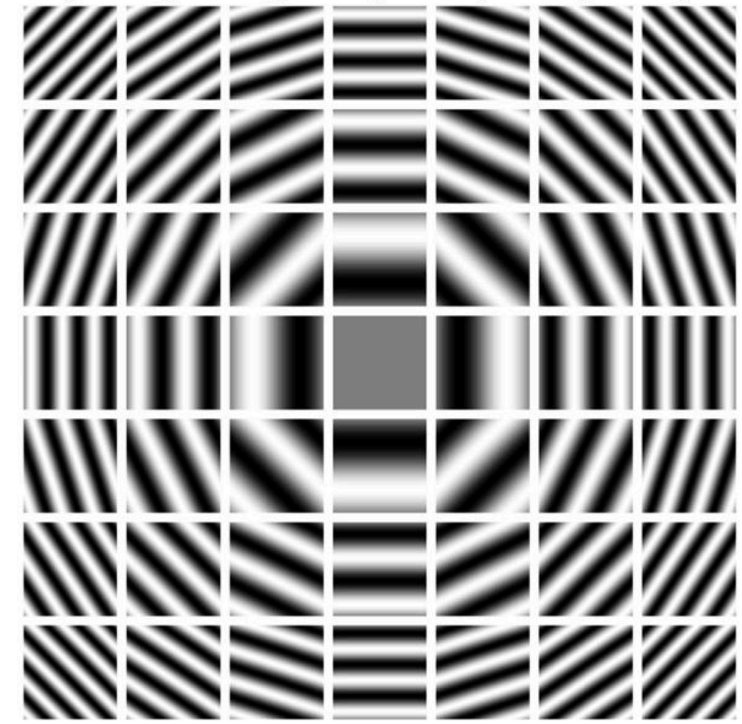



$$\text{Best basis} \iff \text{Fastest error decay } \|f - f_M\|^2$$

(c) Numerical tours, Gabriel Peyre

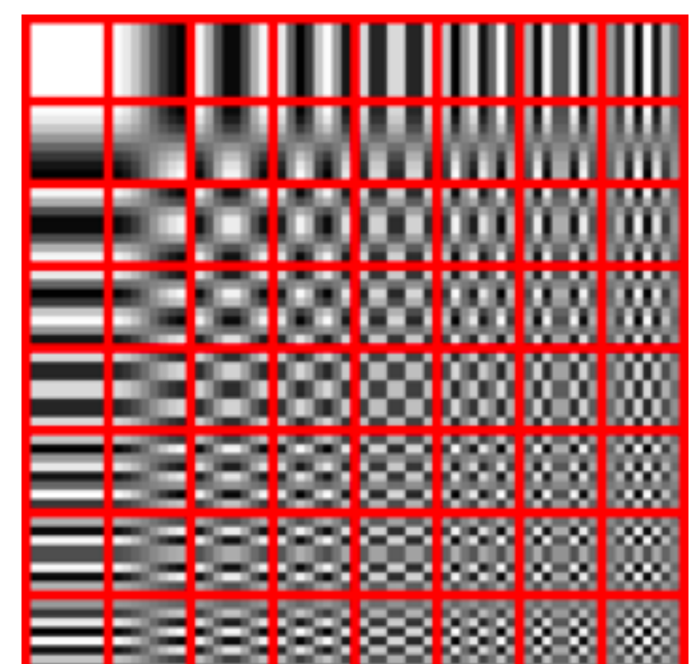
# DFT vs DCT

$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-\frac{2\pi i}{N}kn}$$
$$= \sum_{n=0}^{N-1} x_n \cdot [\cos(2\pi kn/N) - i \cdot \sin(2\pi kn/N)],$$



$$X_k = \sum_{n=0}^{N-1} x_n \cos\left[\frac{\pi}{N}\left(n + \frac{1}{2}\right)k\right] \qquad k = 0, \ldots, N-1.$$

Used in JPEG

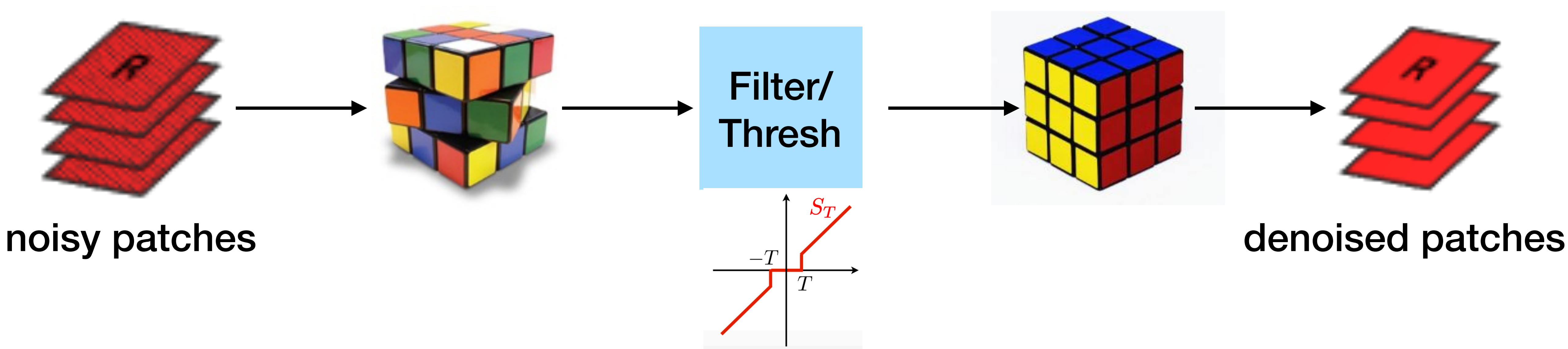


Calculating Wavelet basis is expensive.

DCT basis works better for natural images

# Collaborative filtering

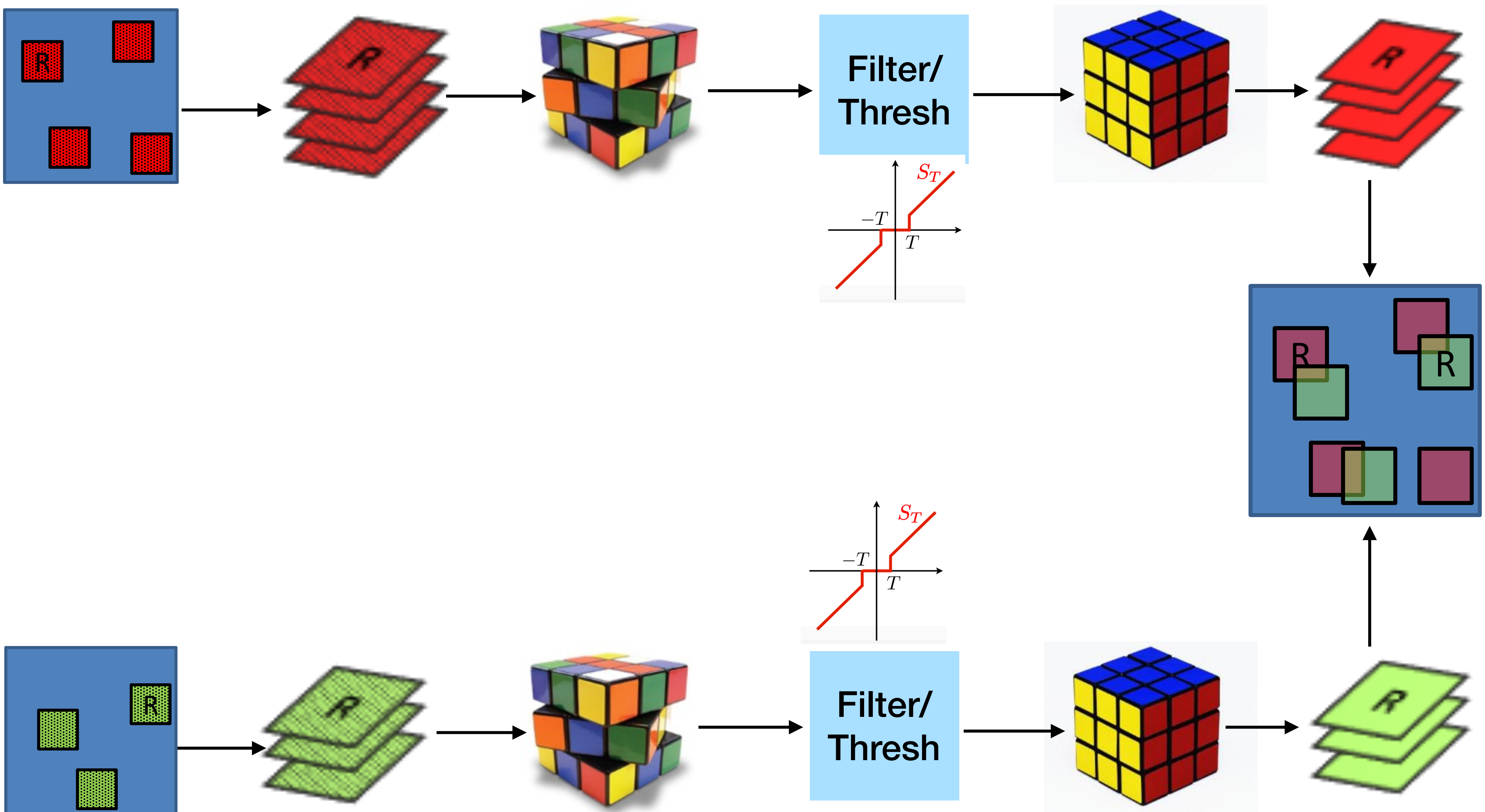
- if step-1: Use hard thresholding in the transform domain
    - else if step-2: Use Weiner filter
- Each patch in the group gets a denoised estimate




**Unlike in NLM — where only the centre pixel was getting the estimates**

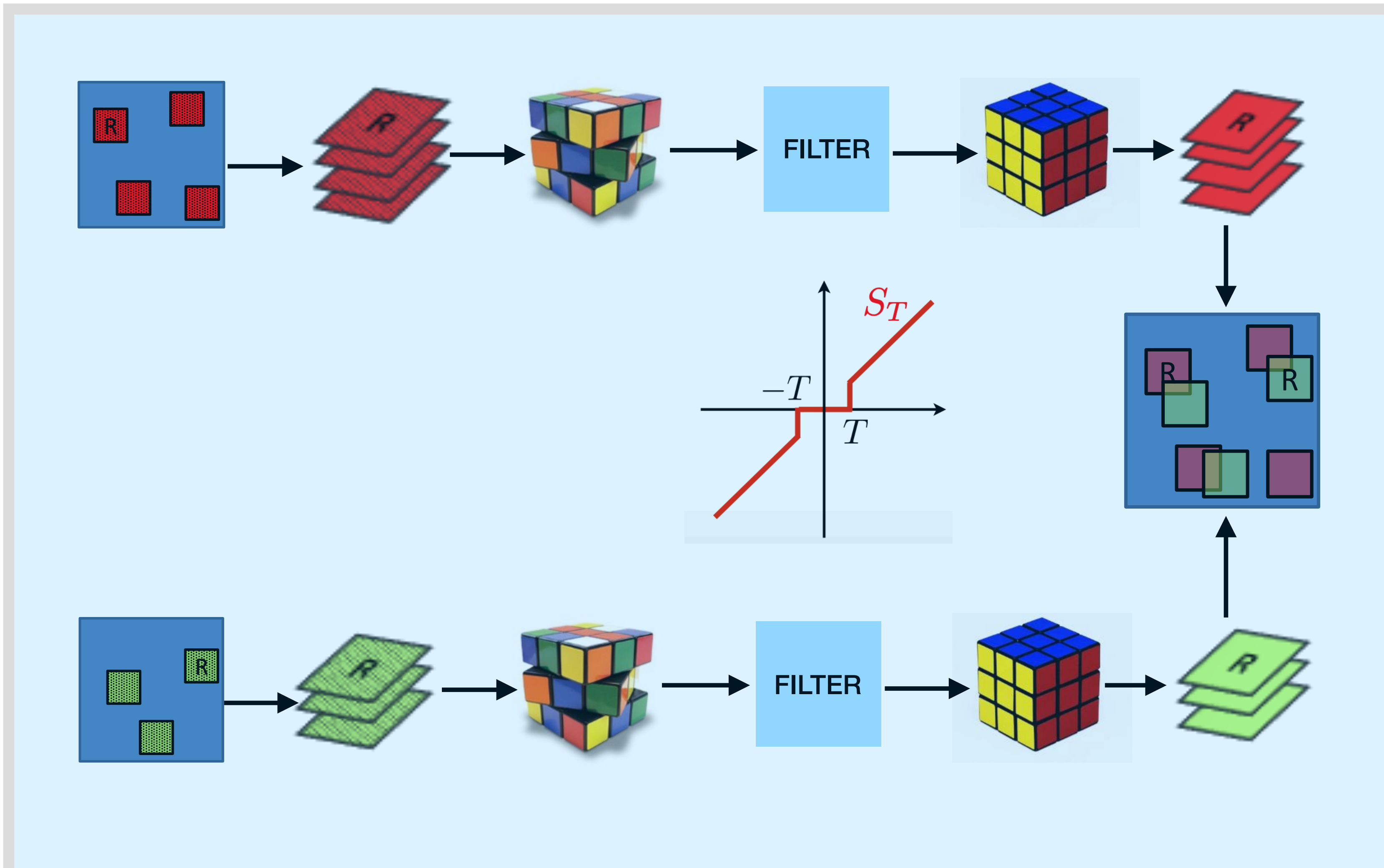
Dabov et al., 2007

# Multiple BM3D estimates



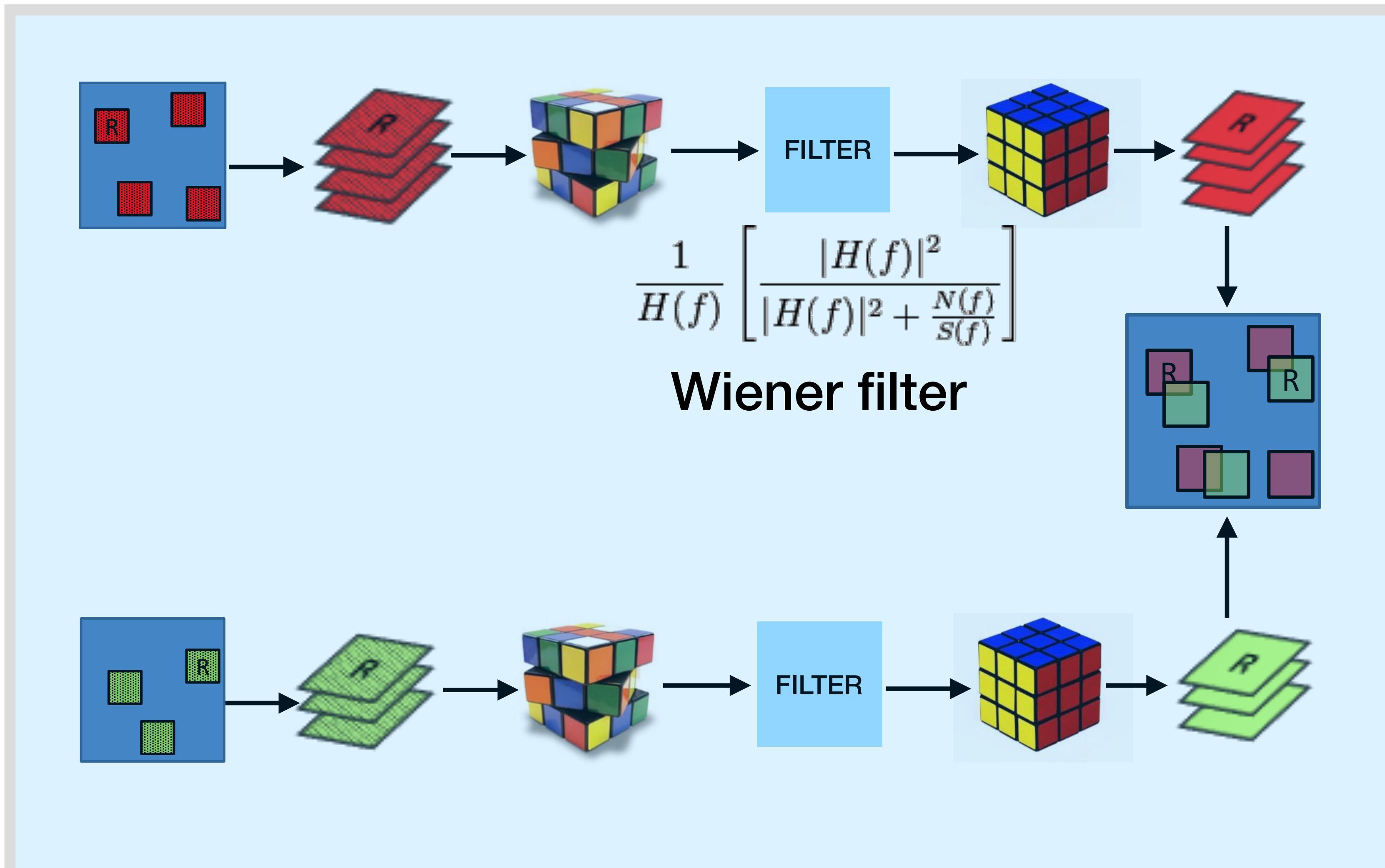

Dabov et al., 2007

# Stage-1 BM3D denoiser




Dabov et al., 2007

# Stage-2 BM3D denoiser




# BM3D - Fusion

Each pixel gets multiple estimates from different groups.

Naive approach: average all estimates of each pixel … not all patches are as good

Suggestion: Give higher weights to more reliable estimates

Reliable?

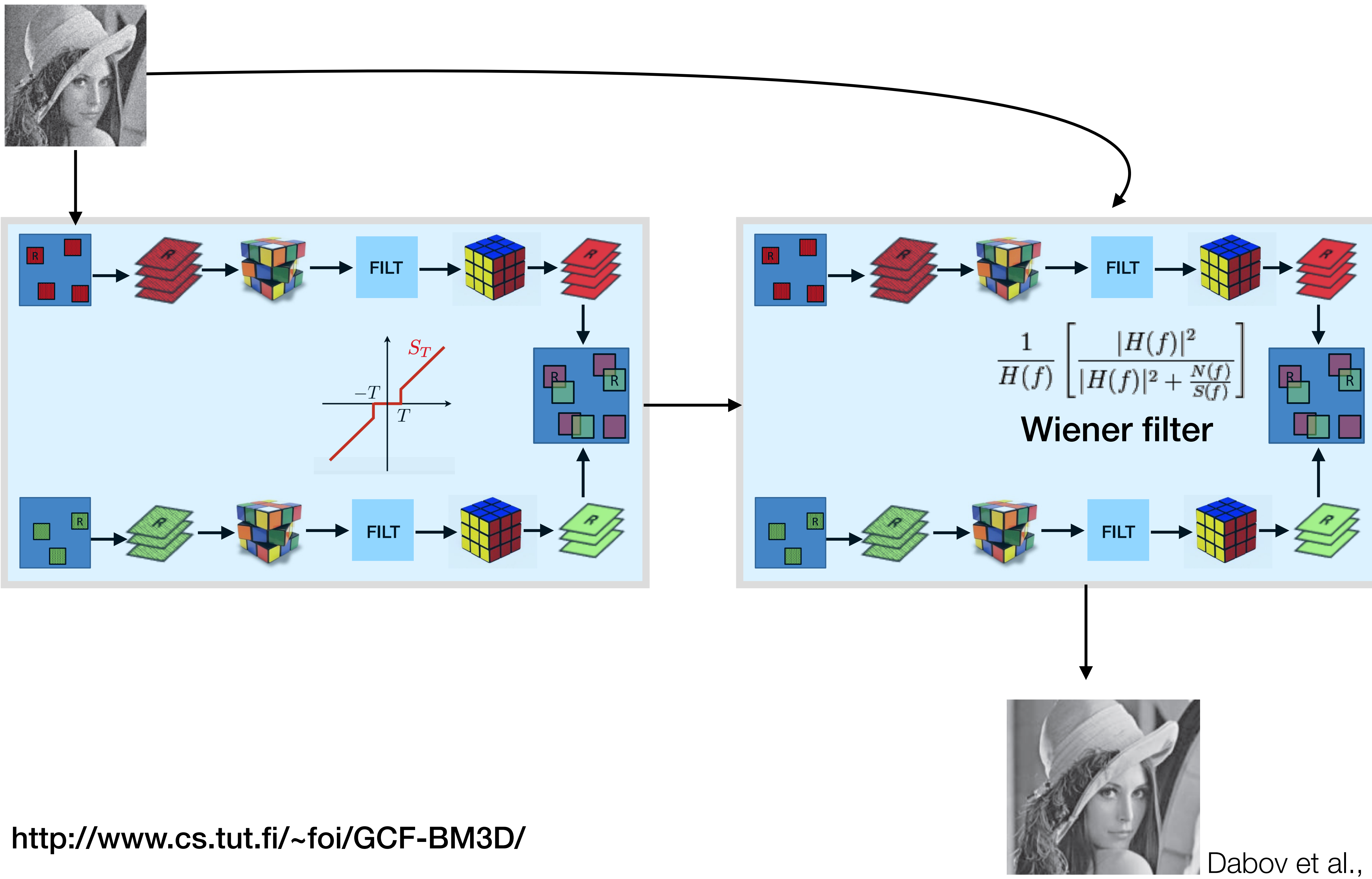
**Hard thresholding:**

**W proportional to** $\frac{1}{\text{no. of non-zero coefficients}}$

**Wiener Filter:**

**W proportional to** $\frac{1}{||\text{filter}||^2}$

# Two-stage BM3D denoising




http://www.cs.tut.fi/~foi/GCF-BM3D/

Dabov et al., 2007

# BM3D - summarised

Stage1:

- Gather similar patches and pile them into a “block” (set high enough threshold while matching to account for noise)
- Transform the “block” using a 3D transform (DCT)
- Apply hard thresholding
- Weigh the importance of each patch and fuse the patches to reconstruct the image

Stage 2:

- Start with the image from stage 1
- Gather similar patches and pile them into a “block”
- Transform the “block” using a 3D transform (DCT)
- Apply Wiener filter
- Weigh the importance of each patch and fuse the patches to reconstruct the image

**Runs in ~8 seconds for 256x256 image**

**state-of-the-art until 2014-15**

**Extended to videos — BM4D (match video blocks).**

**http://www.cs.tut.fi/~foi/GCF-BM3D/**

# Thanks!!