Wavelet for Graphs and its Deployment to Image Processing*

Michael Elad

The Computer Science Department
The Technion – Israel Institute of technology
Haifa 32000, Israel

The research leading to these results has been received funding from the European union's Seventh Framework Programme (FP/2007-2013) ERC grant Agreement ERC-SPARSE- 320649

*Joint work with

Idan Ram

Israel Cohen

The Electrical Engineering department
Technion – Israel Institute of Technology

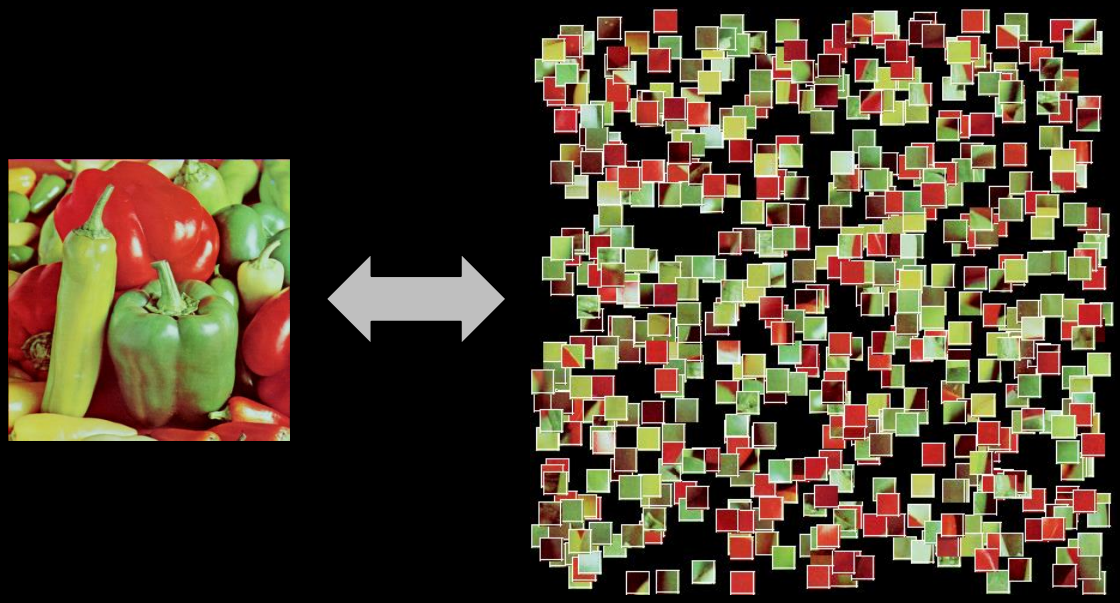
SPARS 2013

Signal Processing with Adaptive Sparse Structured Representations.
July 8-11, 2013. EPFL, Lausanne.

Technion
Israel Institute of Technology

Patch-Based Processing of Images

In the past decade we see more and more researchers suggesting to process a signal or an image with a paradigm of the form:



Break the given image into overlapping (small) patches

↓

Operate on the patches separately or by exploiting inter-relation between them

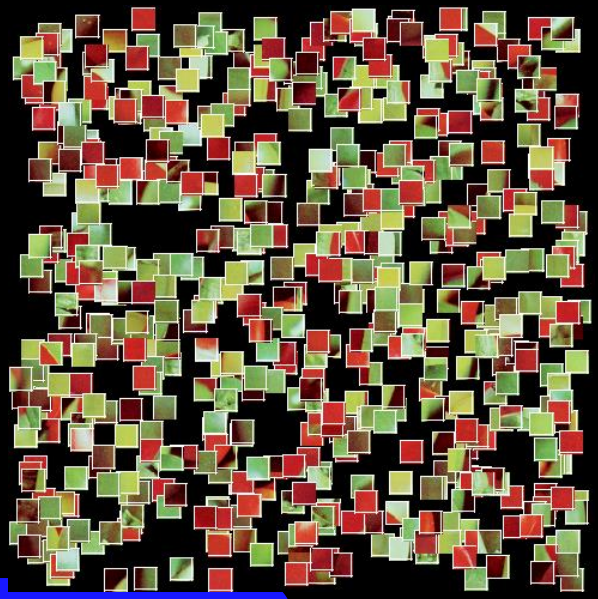
↓

Put back the resulting patches into a result canvas

Patch-Based Processing of Images

In the past decade we see more and more researchers suggesting to process a signal or an image with a paradigm of the form:

Surprisingly, these methods are very effective, actually leading to today's state-of-the-art in many applications



Common theme: The image patches are believed to exhibit a highly-structured geometrical form in the embedding space they reside in

Patches … Patches … Patches …

Who are the researchers promoting this line of work?
Many leading scientists from various places

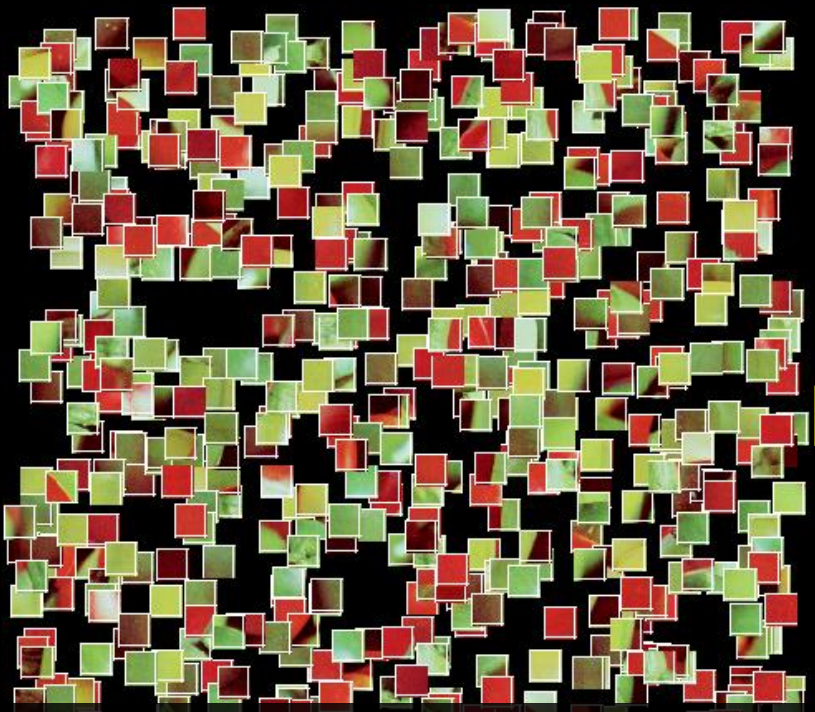
Various Ideas:

Non-local-means
Kernel regression
Sparse representations
Locally-learned dictionaries
BM3D
Structured sparsity
Structural clustering
Subspace clustering
Gaussian-mixture-models
Non-local sparse rep.
Self-similarity
Manifold learning
…

This Talk is About …

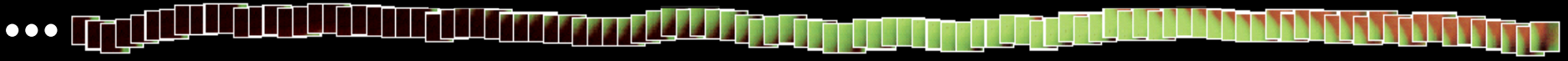
A different way to treat an image using its overlapped patches

Process the Patches

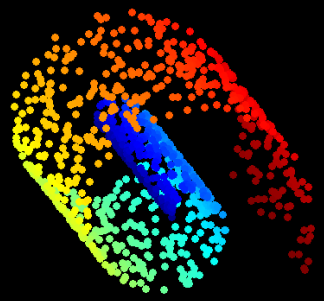


Order to form the shortest possible path

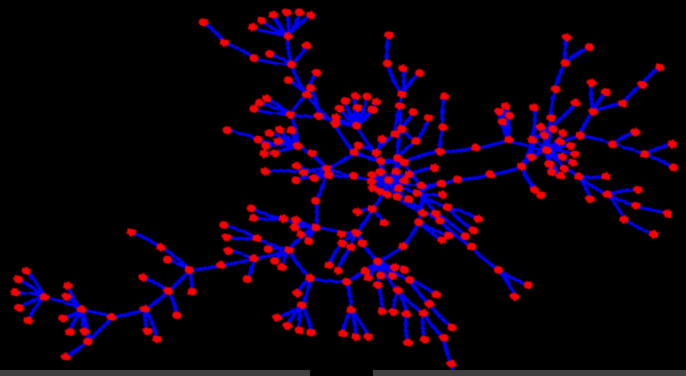
This ordering induces a very interesting permutation on the image pixels



Surprisingly, This Talk is Also About …



Processing of Non-Conventionally Structured Signals



Many signal-processing tools (filters, alg., transforms, …) are tailored for uniformly sampled signals

Now we encounter different types of signals: e.g., point-clouds and graphs. Can we extend classical tools to these signals?

Our goal: Generalize the wavelet transform to handle this broad family of signals

In the process, we will find ourselves returning to "regular" signals, handling them differently

In fact, this is how this work started in the first place

Part I – GTBWT

Generalized Tree-Based Wavelet Transform – The Basics

This part is taken from the following two papers:

- I. Ram, M. Elad, and I. Cohen, "Generalized Tree-Based Wavelet Transform", IEEE Trans. Signal Processing, vol. 59, no. 9, pp. 4199–4209, 2011.
- I. Ram, M. Elad, and I. Cohen, "Redundant Wavelets on Graphs and High Dimensional Data Clouds", IEEE Signal Processing Letters, Vol. 19, No. 5, pp. 291–294 , May 2012.

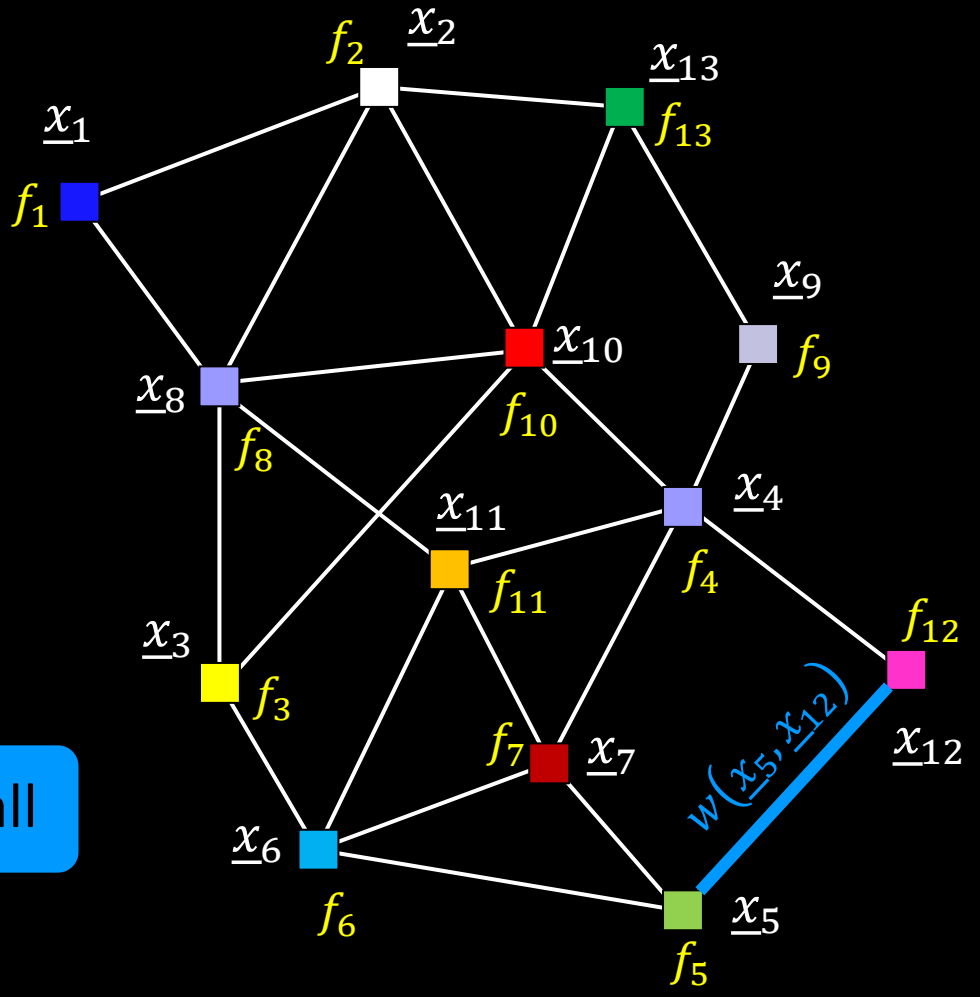
Problem Formulation

- We are given a graph:
  - The $i-th$ node is characterized by a d-dimen. feature vector $\underline{x}_i$
  - The $i-th$ node has a value f_i
  - The edge between the $i-th$ and $j-th$ nodes carries the distance $w(\underline{x}_i, \underline{x}_j)$ for an arbitrary distance measure $w(\cdot,\cdot)$.

- Assumption: a "short edge" implies close-by values, i.e.

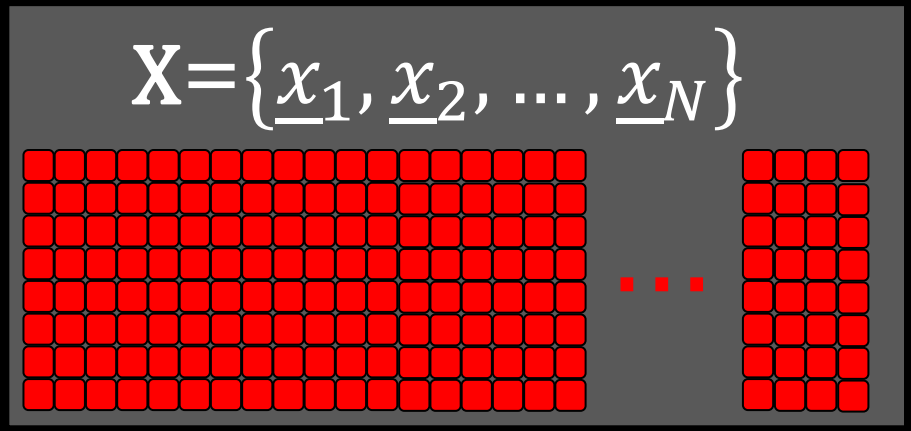
$$w(\underline{x}_i, \underline{x}_j) \text{ small} \rightarrow |f_i - f_j| \text{ small}$$

for almost every pair (i, j).

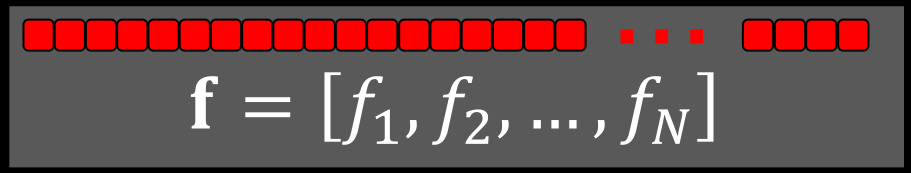


Different Ways to Look at This Data

- We start with a set of d-dimensional vectors $\mathbf{X} = \{\underline{x}_1, \underline{x}_2, \ldots, \underline{x}_N\} \in \mathbb{R}^d$
  These could be:
  - Feature points for a graph's nodes,
  - Set of coordinates for a point-cloud.



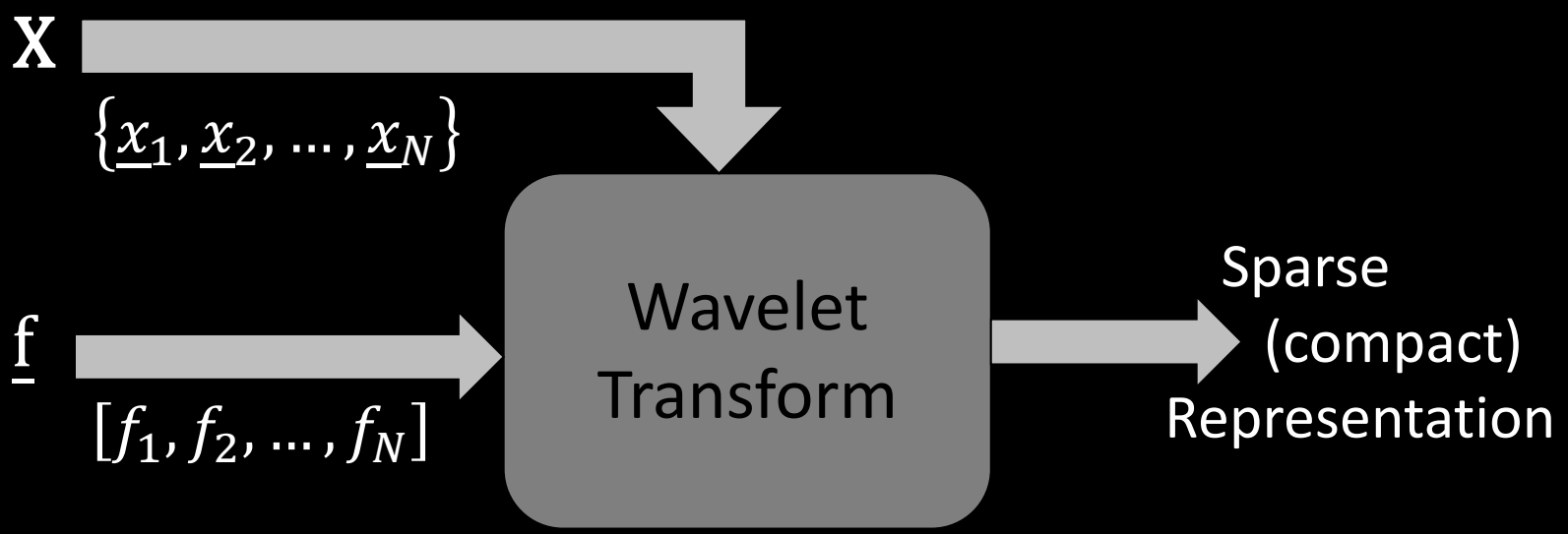
- A scalar function is defined on these coordinates, $f: \mathbf{X} \to \mathbb{R}$, giving $\mathbf{f} = [f_1, f_2, \ldots, f_N]$.



- We may regard this dataset as a set of m samples taken from a high dimensional function $f: \mathbb{R}^d \to \mathbb{R}$.

- The assumption that small $w(\underline{x}_i, \underline{x}_j)$ implies small $|f_i - f_j|$ for almost every pair (i, j) implies that the function behind the scene, f, is "regular".

Our Goal



Why Wavelet?

- Wavelet for regular piece-wise smooth signals is a highly effective "sparsifying transform". However, the signal (vector) $\mathbf{f}$ is not necessarily smooth in general.
- We would like to imitate this for our data structure.

Wavelet for Graphs – A Wonderful Idea

I wish we would have thought of it first …


"Diffusion Wavelets"

R. R. Coifman, and M. Maggioni, 2006.


"Multiscale Methods for Data on Graphs and Irregular …. Situations"

M. Jansen, G. P. Nason, and B. W. Silverman, 2008.


"Wavelets on Graph via Spectal Graph Theory"

D. K. Hammond, and P. Vandergheynst, and R. Gribonval, 2010.


"Multiscale Wavelets on Trees, Graphs and High … Supervised Learning"

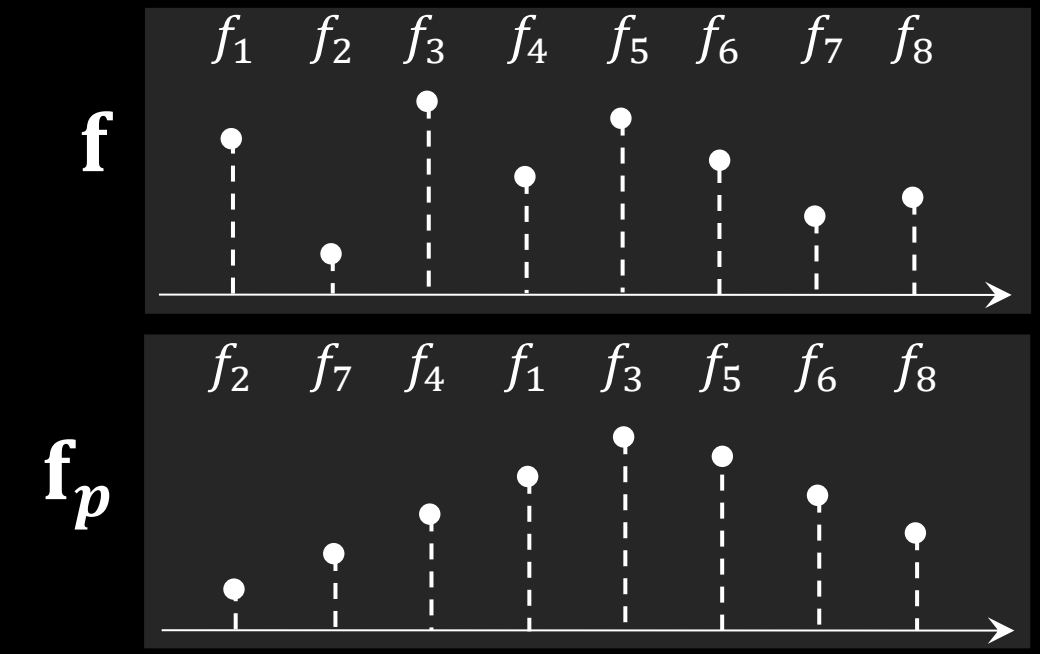
M . Gavish, and B. Nadler, and R. R. Coifman, 2010.


"Wavelet Shrinkage on Paths for Denoising of Scattered Data"

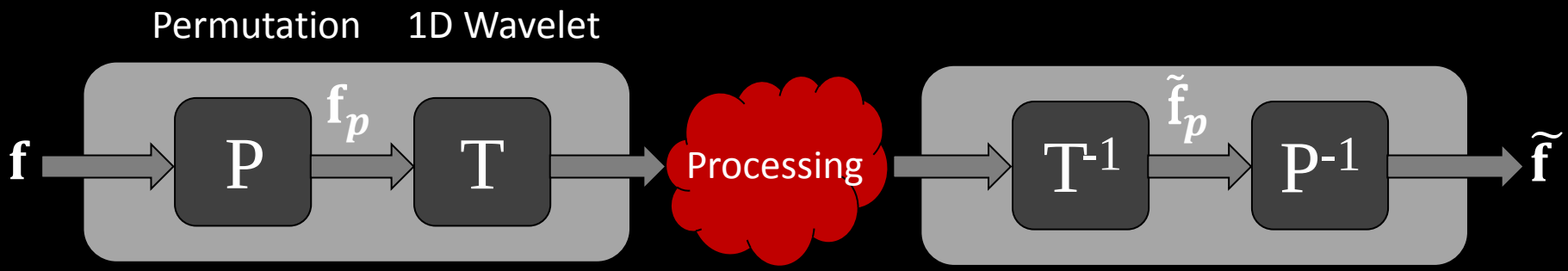
D. Heinen and G. Plonka, 2012.

…

The Main Idea (1) - Permutation

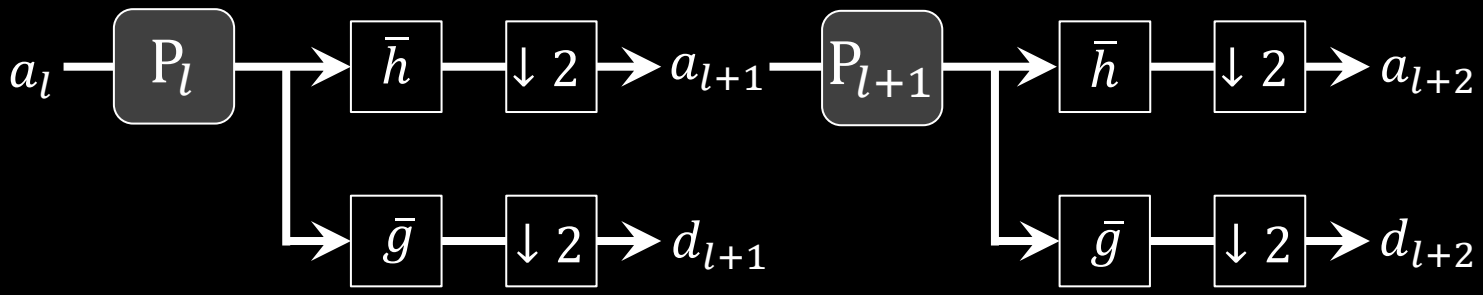


Permutation using
$\mathbf{X} = \{x_1, x_2, \ldots, x_N\}$



The Main Idea (2) - Permutation

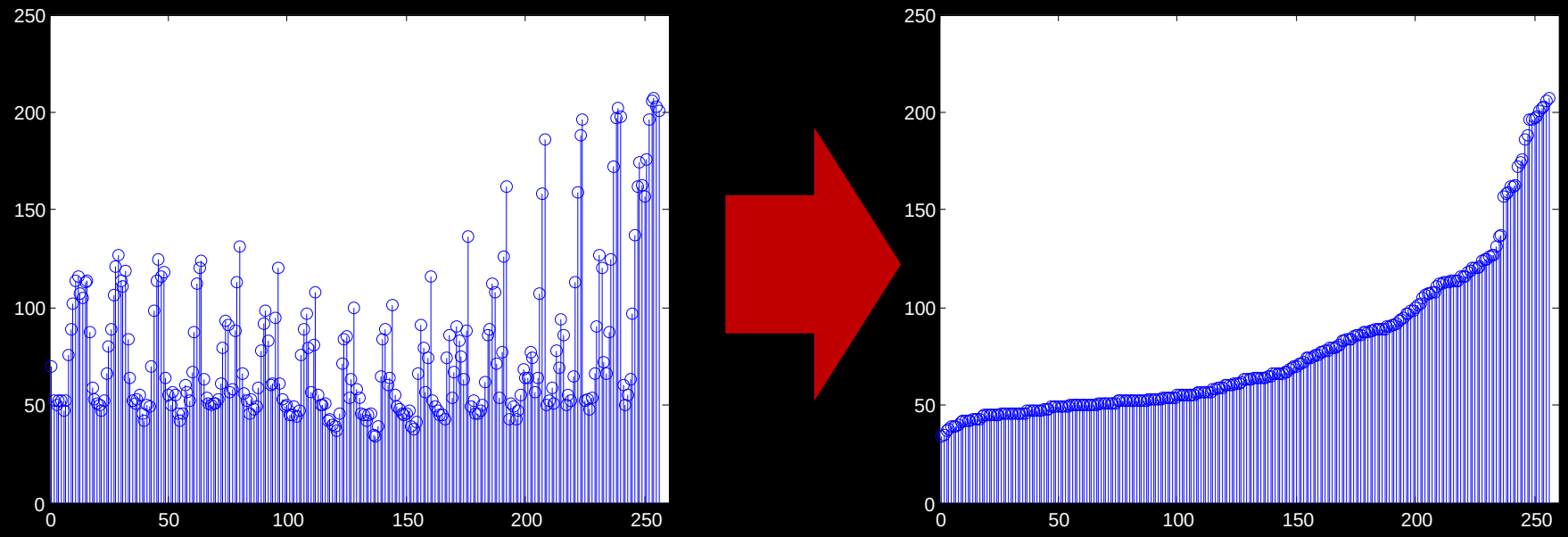
- In fact, we propose to perform a different permutation in each resolution level of the multi-scale pyramid:



- Naturally, these permutations will be applied reversely in the inverse transform.
- Thus, the difference between this and the plain 1D wavelet transform applied on $\mathbf{f}$ are the additional permutations, thus preserving the transform's linearity and unitarity, while also adapting to the input signal.

Building the Permutations (1)

- Lets start with $\mathbf{P}_0$ – the permutation applied on the incoming signal.
- Recall: the wavelet transform is most effective for piecewise regular signals.
  $\rightarrow$ thus, $\mathbf{P}_0$ should be chosen such that $\mathbf{P}_0\mathbf{f}$ is most "regular".
- So, … for example, we can simply permute by sorting the signal $\mathbf{f}$ …



Building the Permutations (2)

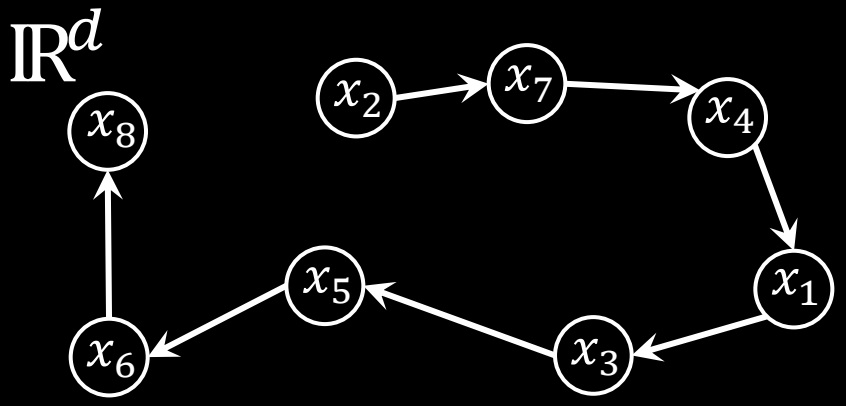
- **However**: we will be dealing with corrupted signals $\mathbf{f}$ (noisy, missing values, …) and thus such a sort operation is impossible.
- To our help comes the feature vectors in $\mathbf{X}$, which reflect on the order of the signal values, f_k. Recall:

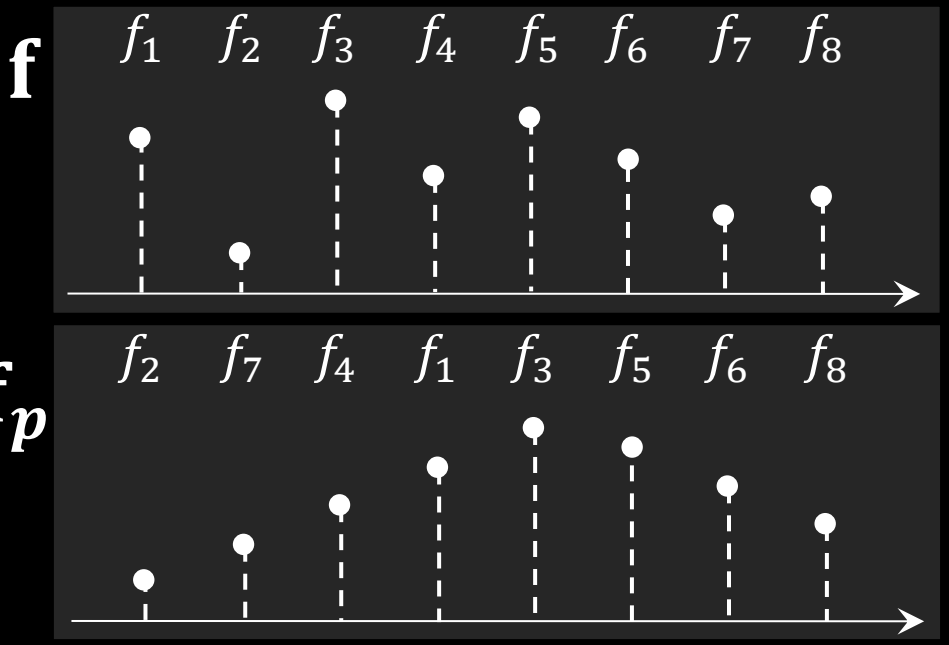
Small $w(x_i, x_j)$ implies small $|f(x_i) - f(x_j)|$ for almost every pair (i, j)

- Thus, instead of solving for the optimal permutation that "simplifies" $\mathbf{f}$, we order the features in $\mathbf{X}$ to the shortest path that visits in each point once, in what will be an instance of the Traveling-Salesman-Problem (TSP):

$$\min_{\mathrm{P}} \sum_{i=2}^{N} |f^p(i) - f^p(i-1)| \quad \Rightarrow \quad \min_{\mathrm{P}} \sum_{i=2}^{N} w(x_i^p, x_{i-1}^p)$$

Building the Permutations (3)



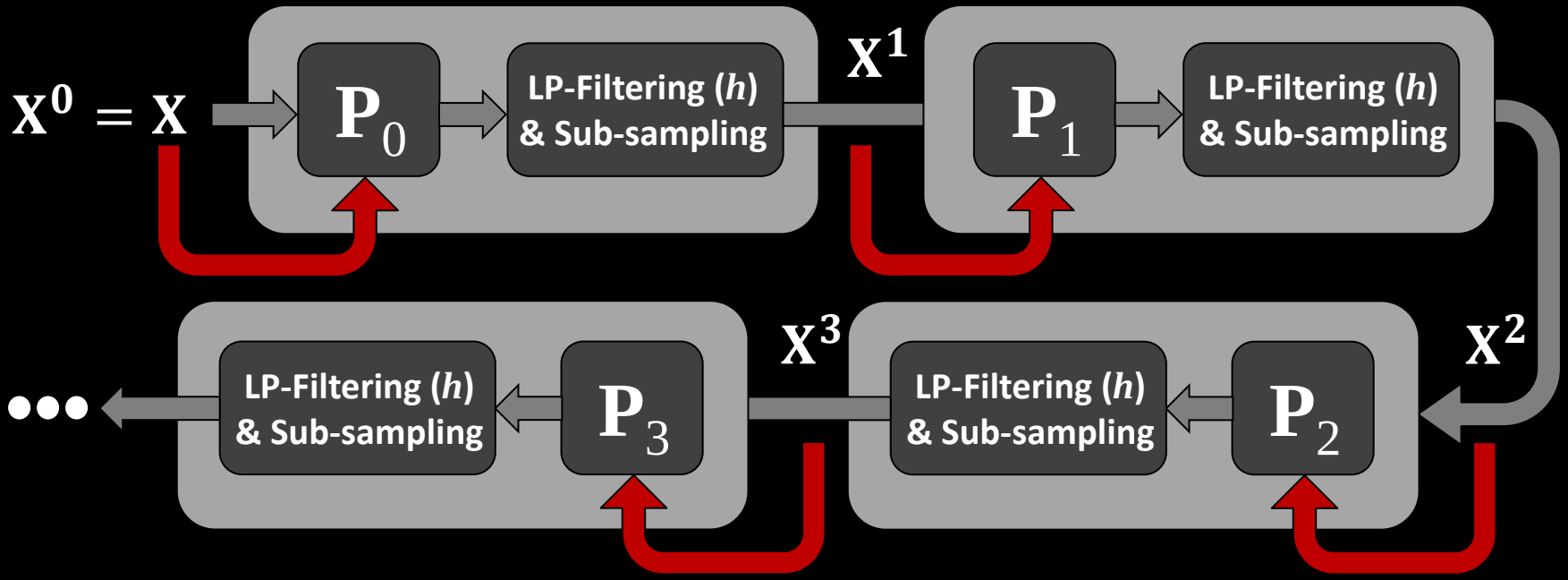


We handle the TSP task by a greedy (and crude) approximation:

- Initialize with an arbitrary index j;
- Initialize the set of chosen indices to $\Omega(1)=\{j\}$;
- Repeat $k=1:1:N\text{-}1$ times:
  - Find x_i – the nearest neighbor to $x_{\Omega(k)}$ such that $i \notin \Omega$;
  - Set $\Omega(k+1)=\{i\}$;
- Result: the set Ω holds the proposed ordering.

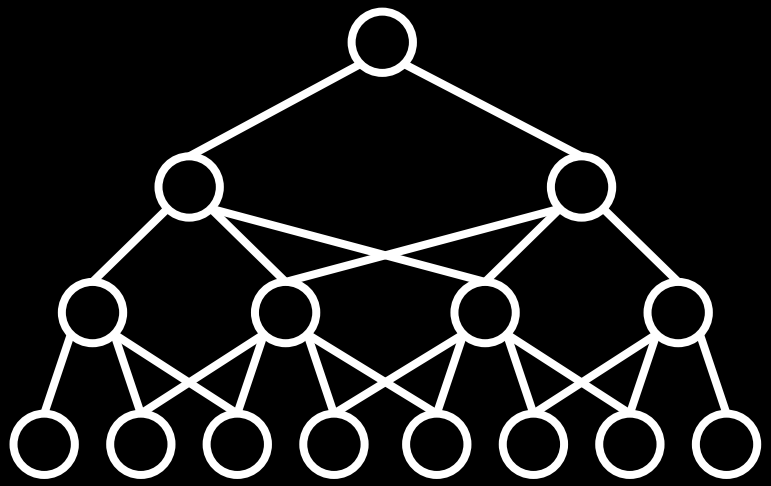
Building the Permutations (4)

- So far we concentrated on P_0 at the finest level of the multi-scale pyramid.
- In order to construct P_1, P_2, … , P_{L-1}, the permutations at the other pyramid's levels, we use the same method, applied on propagated (reordered, filtered and sub-sampled) feature-vectors through the same wavelet pyramid:

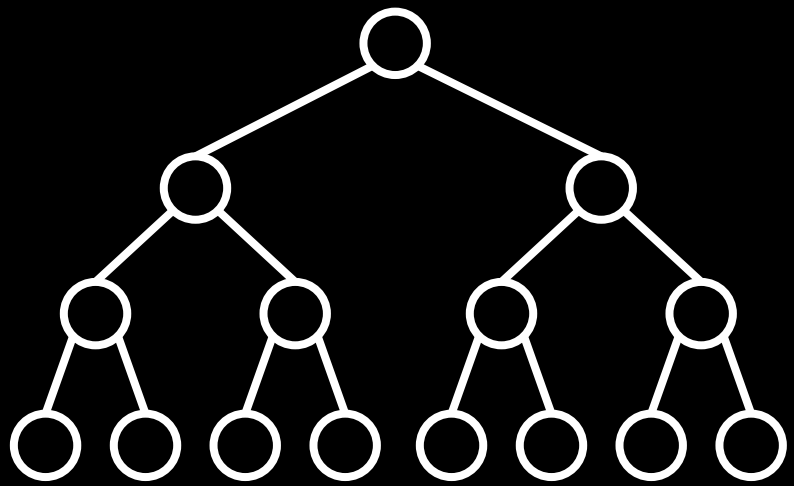


Why “Generalized Tree …”?

“Generalized” tree



Tree (Haar wavelet)



- Our proposed transform: Generalized Tree-Based Wavelet Transform (GTBWT).
- We also developed a Redundant version of this transform based on the stationary wavelet transform [Shensa, 1992] [Beylkin, 1992] – also related to the “A-Trous Wavelet” (will not be presented here).

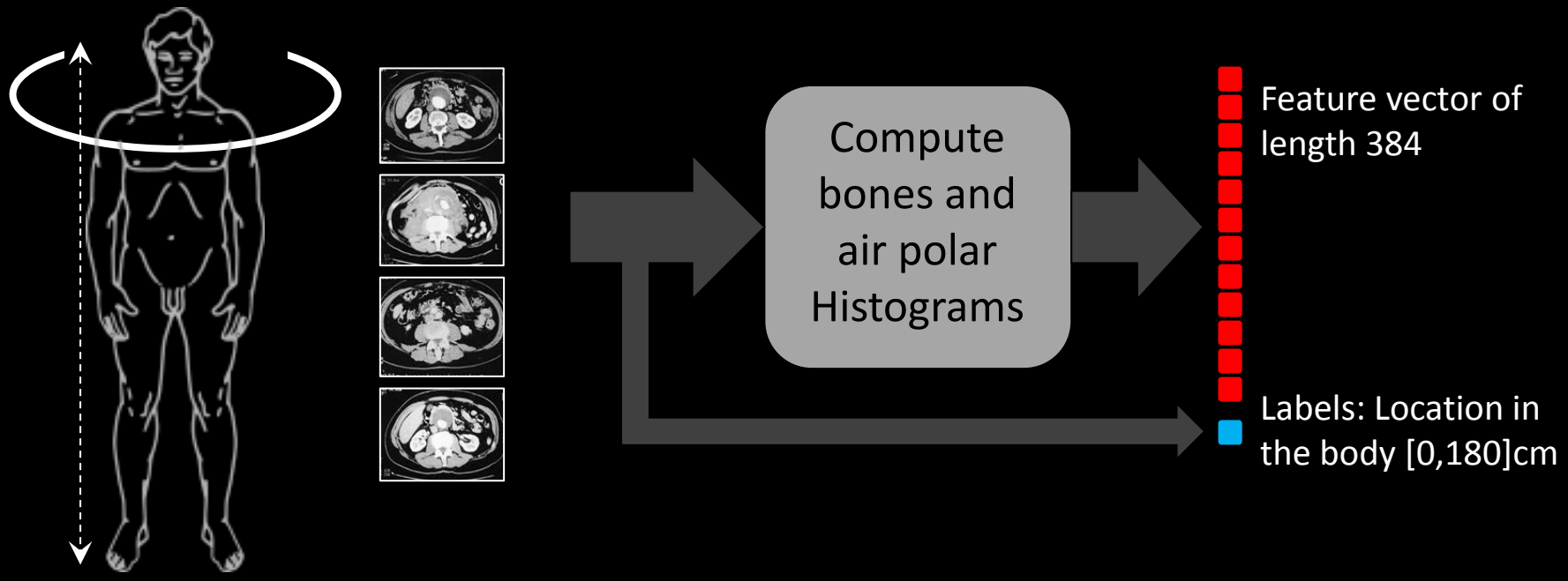
Treating Graph/Cloud-of-points

- Just to complete the picture, we should demonstrate the (R)GTBWT capabilities on graphs/cloud of points.
- We took several classical machine learning train + test data for several regression problems, and tested the proposed transform in
  - Cleaning (denoising) the data from additive noise;
  - Filling in missing values (semi-supervised learning); and
  - Detecting anomalies (outliers) in the data.
- The results are encouraging. We shall present herein one such experiment briefly.

SKIP?

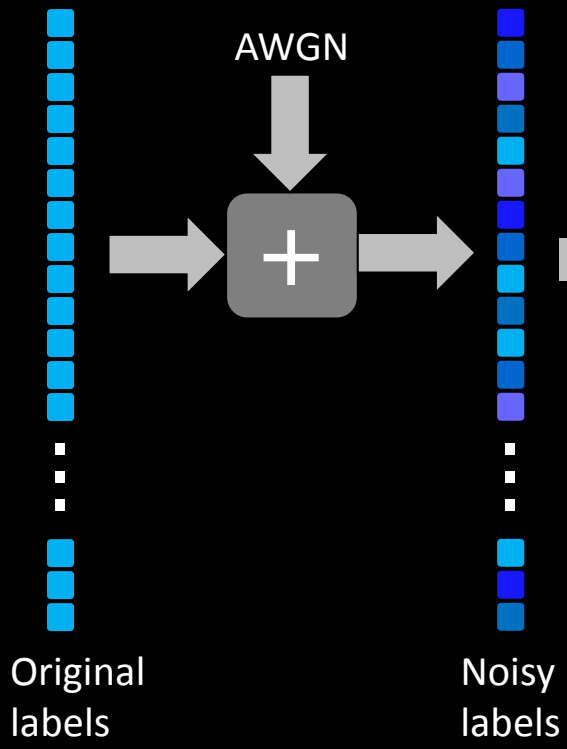
Treating Graphs: The Data

Data Set: Relative Location of CT axial axis slices



More details: Overall 53500 such pairs of feature and value, extracted from 74 different patients (43 male and 31 female).

Treating Graphs: Denoising



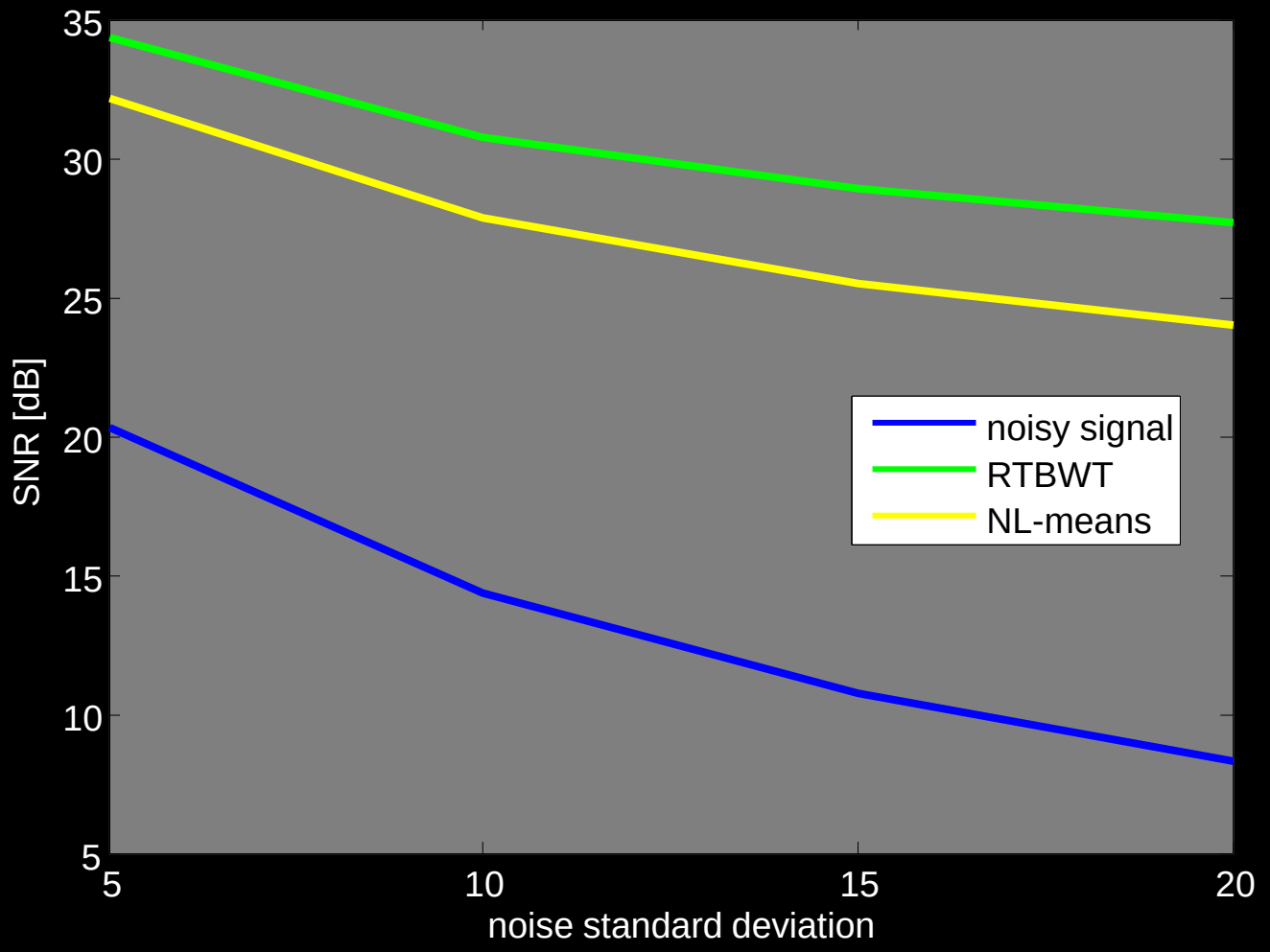
Denoising by NLM-like algorithm

Find for each point its K-NN in feature-space, and compute a weighted average of their labels

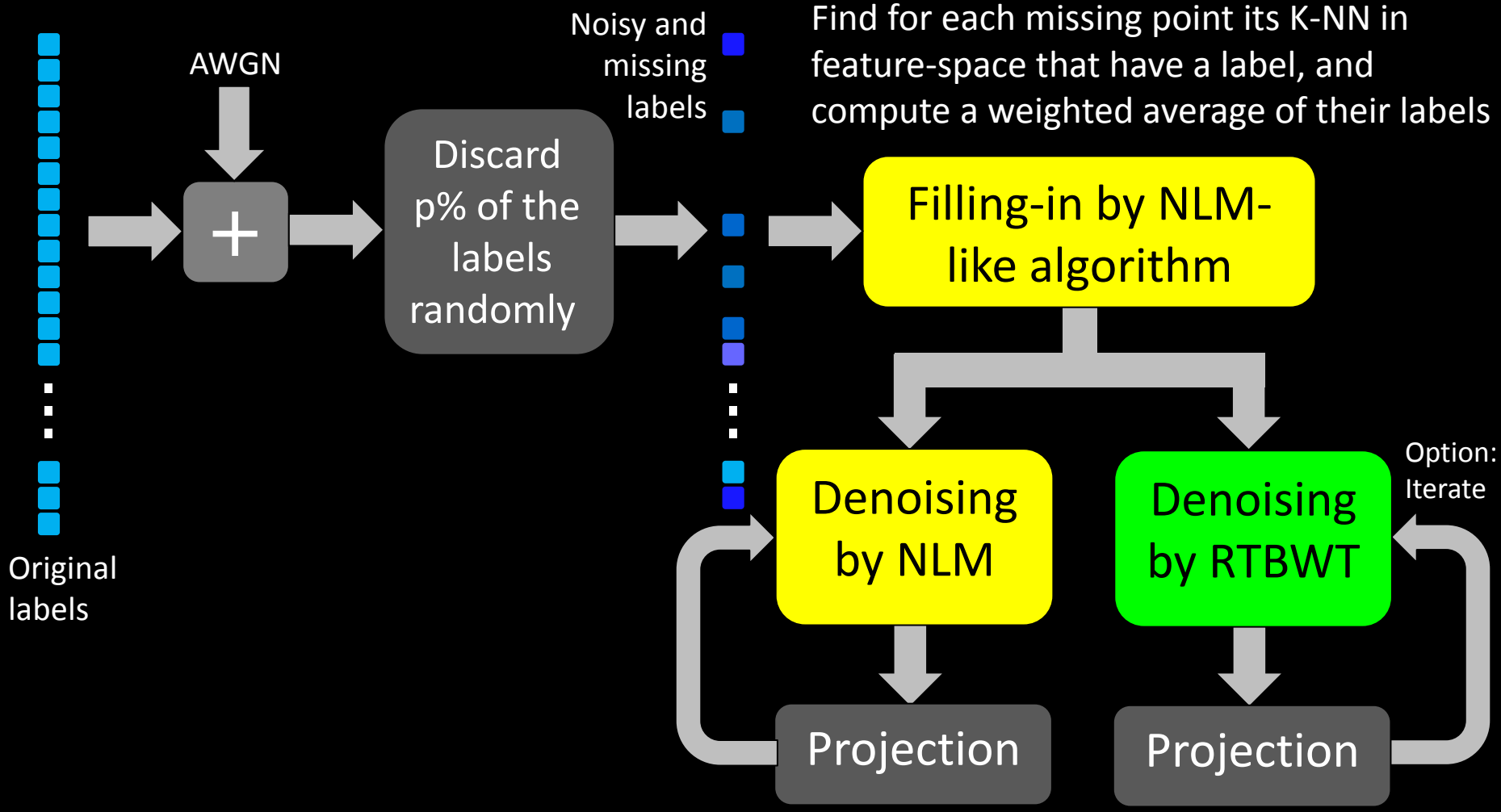
Denoising by THR with RTBWT

Apply the RTBWT transform to the point-cloud labels, threshold the values and transform back

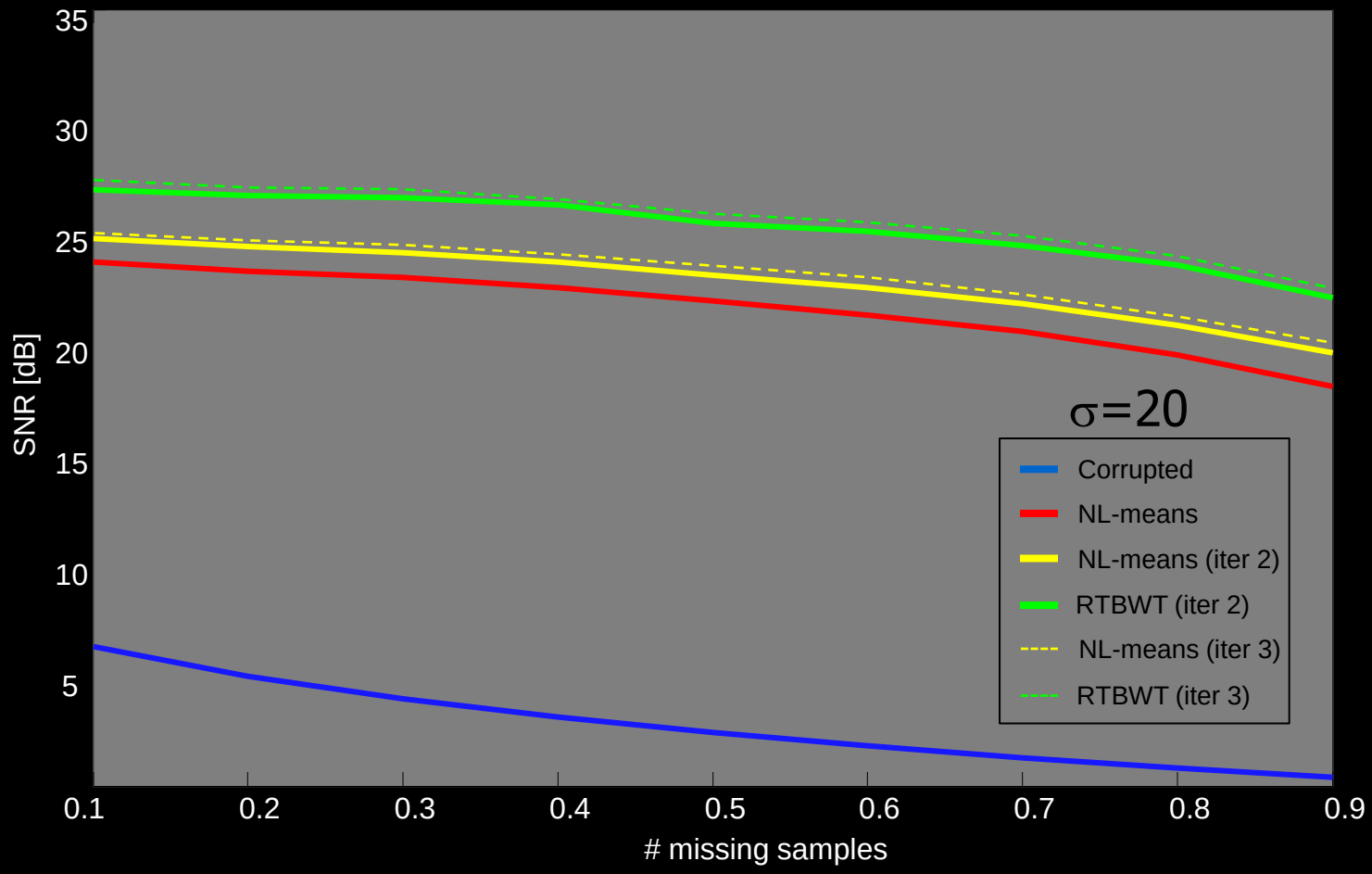
Treating Graphs: Denoising



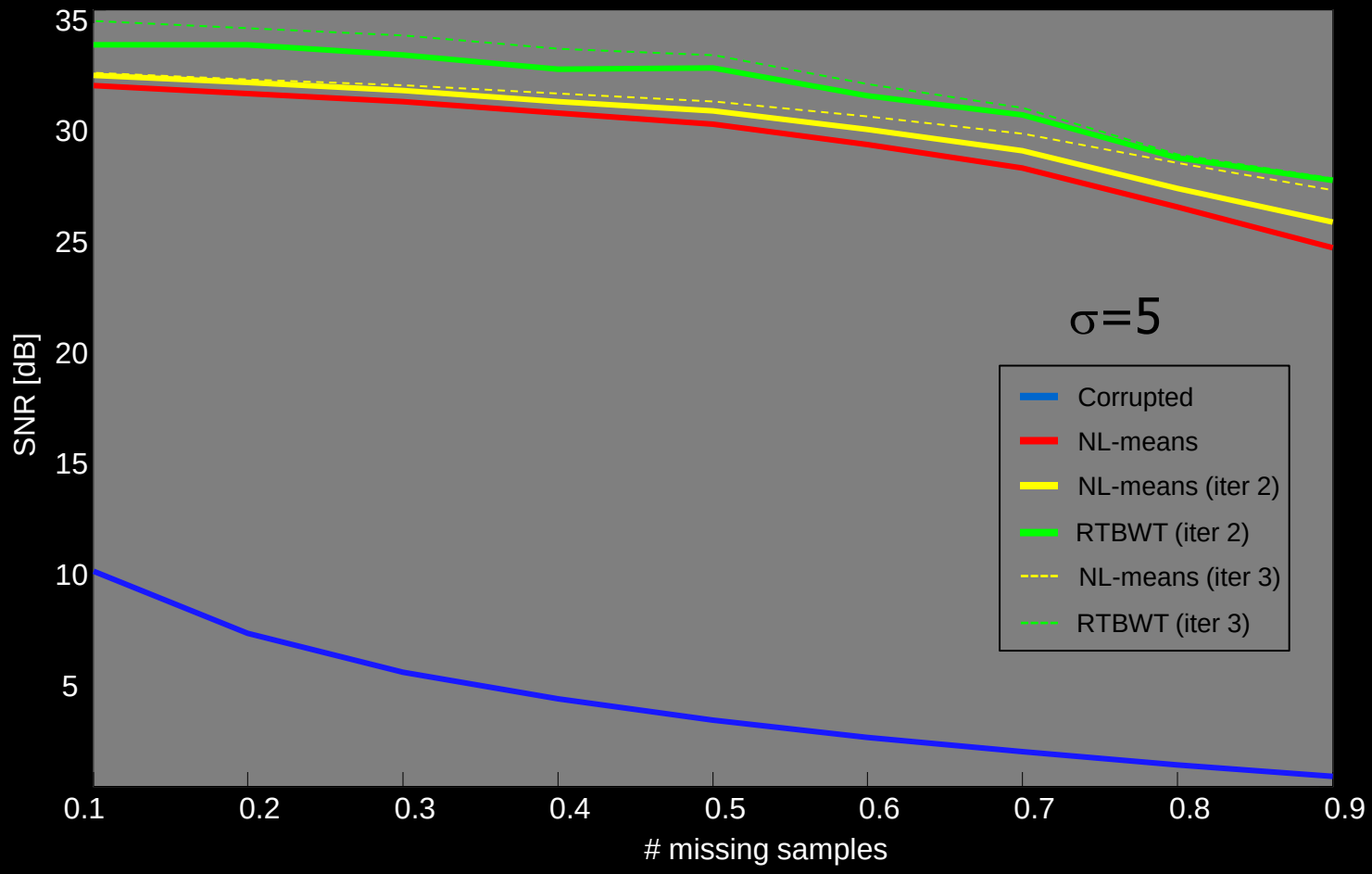
Treating Graphs: Semi-Supervised Learning



Treating Graphs: Semi-Supervised Learning



Treating Graphs: Semi-Supervised Learning



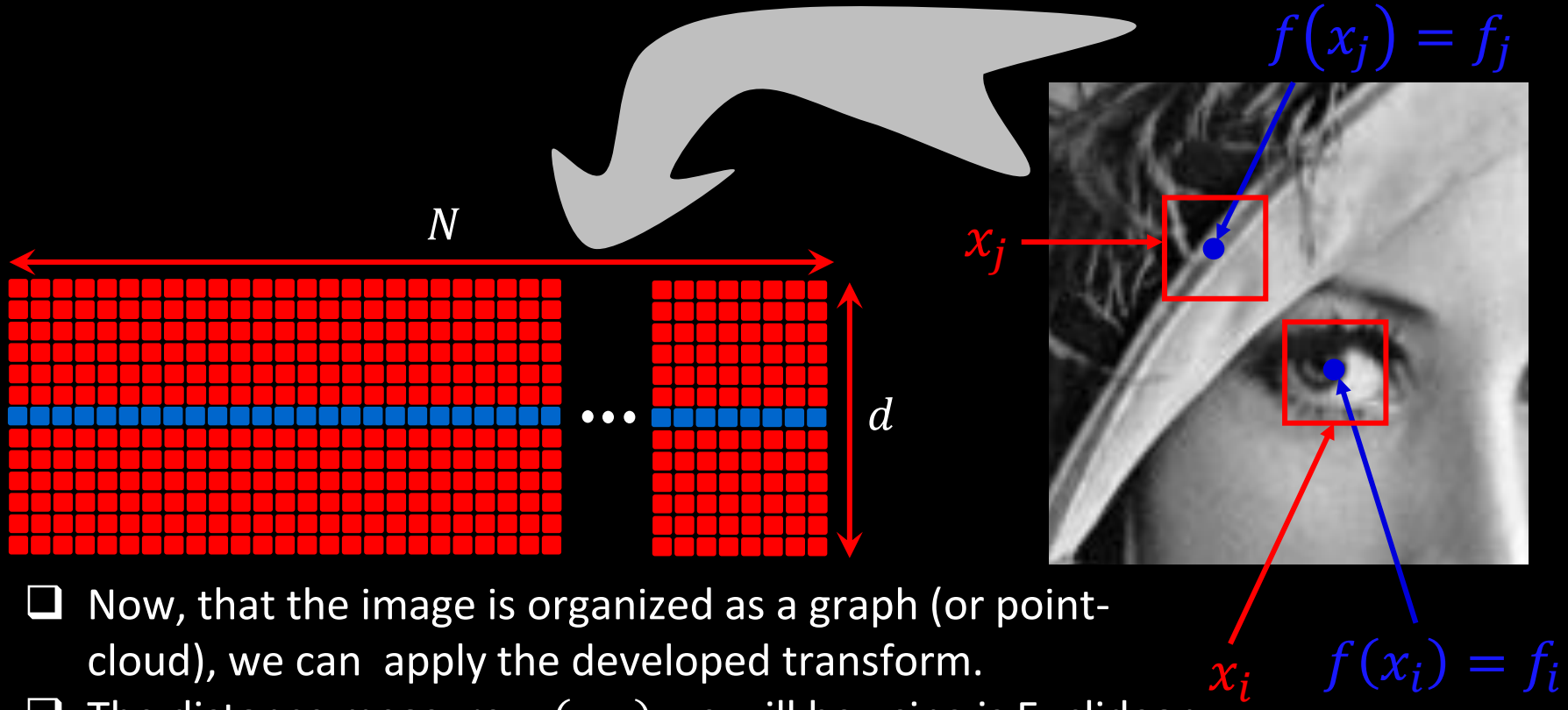
Part II – Handling Images

Using GTBWT by Handling Image Patches

This part is taken from the same papers mentioned before …

- I. Ram, M. Elad, and I. Cohen, "Generalized Tree-Based Wavelet Transform", IEEE Trans. Signal Processing, vol. 59, no. 9, pp. 4199–4209, 2011.
- I. Ram, M. Elad, and I. Cohen, "Redundant Wavelets on Graphs and High Dimensional Data Clouds", IEEE Signal Processing Letters, Vol. 19, No. 5, pp. 291–294 , May 2012.

Could an Image Become a Graph?

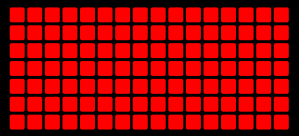


- Now, that the image is organized as a graph (or point-cloud), we can apply the developed transform.
- The distance measure $w(\bullet, \bullet)$ we will be using is Euclidean.
- It seems that after this "conversion", we forget all about spatial proximities.

Our Transform

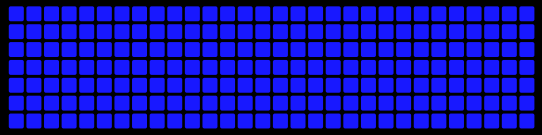
Lexicographic ordering of the N pixels

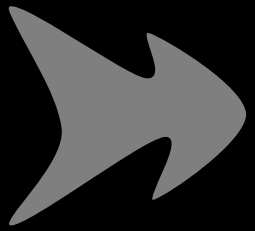
$\mathbf{X}$: Array of overlapped patches of size dN



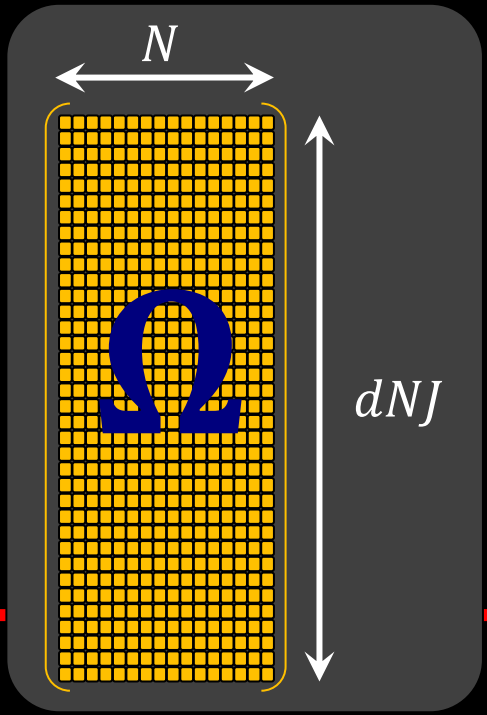
Applying a J redundant wavelet of some sort including permutations

We obtain an array of dNJ transform coefficients





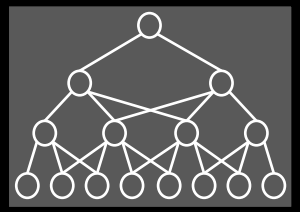
- All these operations could be described as one linear operation: multiplication of $\underline{f}$ by a huge matrix $\mathbf{\Omega}$.
- This transform is adaptive to the specific image.



Lets Test It: M-Term Approximation

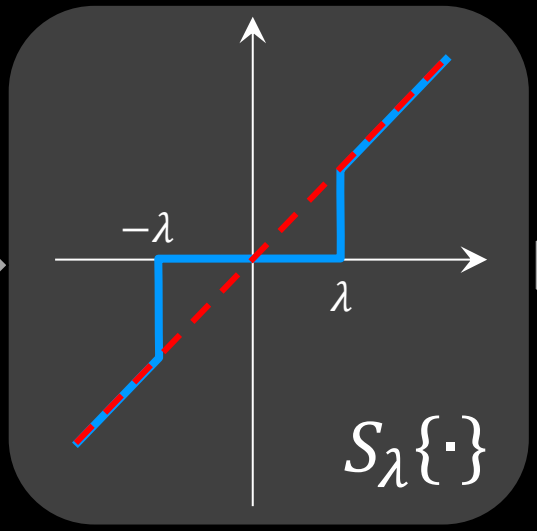
Original Image

Multiply by $\mathbf{\Omega}$:
Forward GTBWT



$\mathbf{f}$

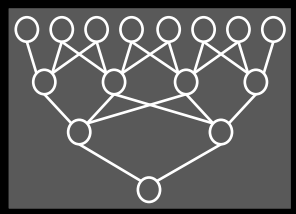
$\mathbf{\Omega f}$



$S_\lambda\{\mathbf{\Omega f}\}$

M non-zeros

Multiply by $\mathbf{D}$:
Inverse GTBWT



Output image

$\hat{\mathbf{f}}$

Show

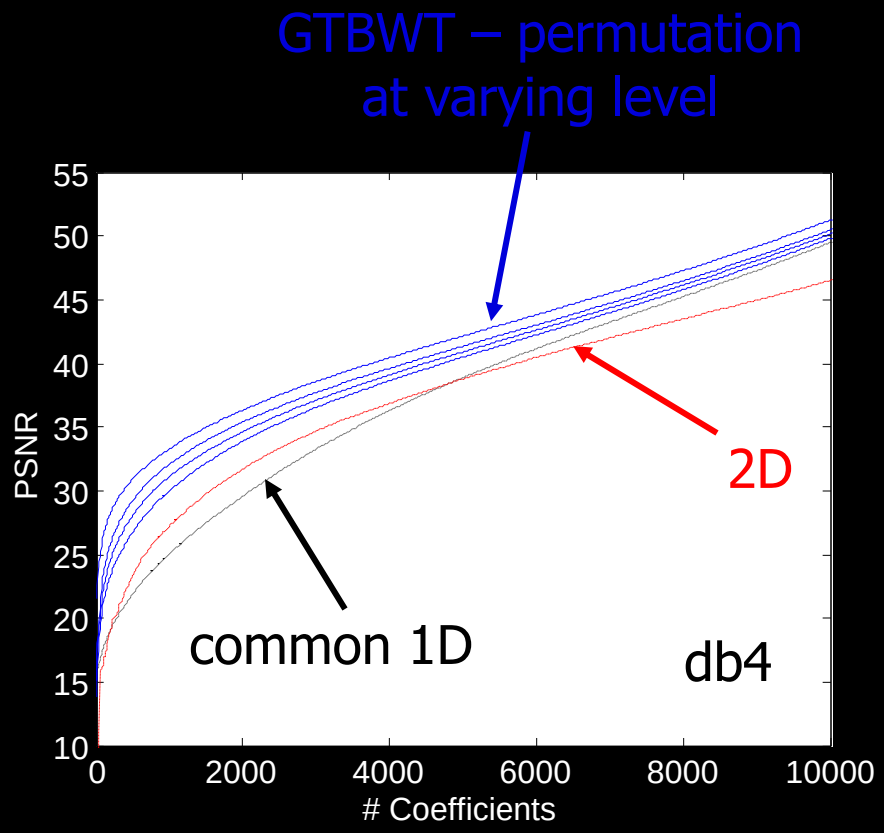
$$\left\|\mathbf{f}-\hat{\mathbf{f}}\right\|^2 = \left\|\mathbf{f}-\mathbf{D}S_\lambda\{\mathbf{\Omega f}\}\right\|^2$$

as a function of M

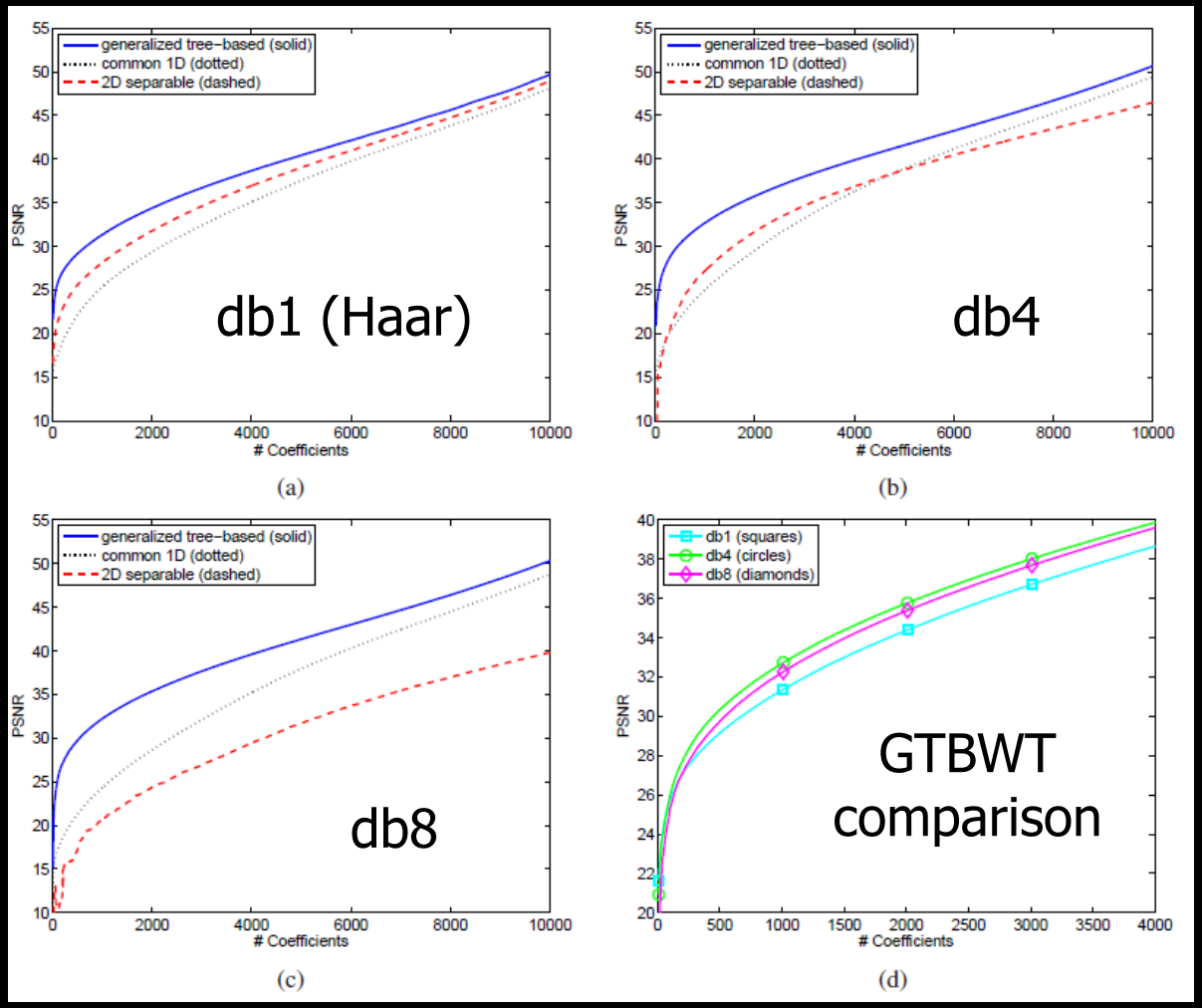
Lets Test It: M-Term Approximation

For a 128×128 center portion of the image Lenna, we compare the image representation efficiency of the

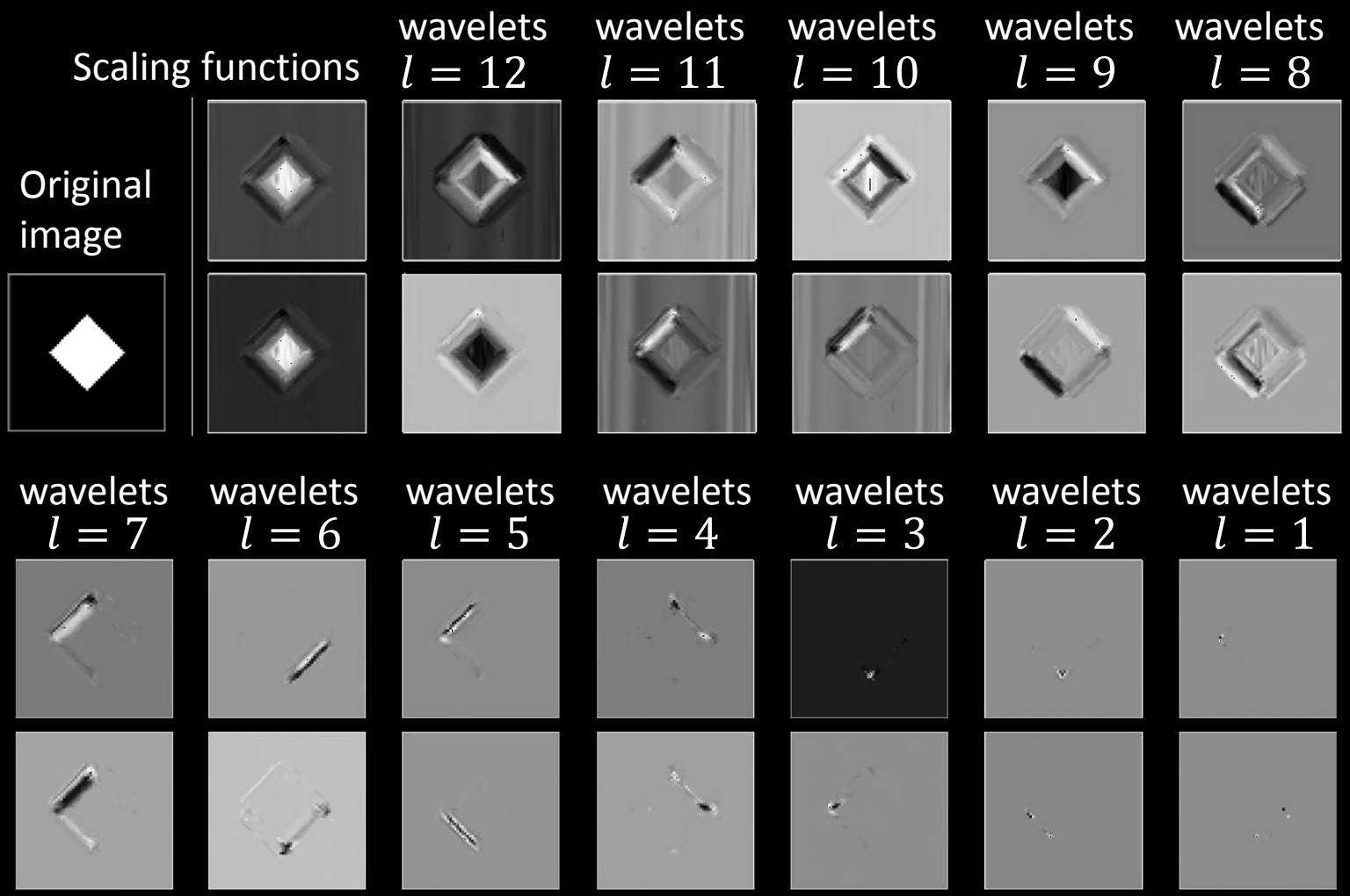
- GTBWT
- A common 1D wavelet transform
- 2D wavelet transform



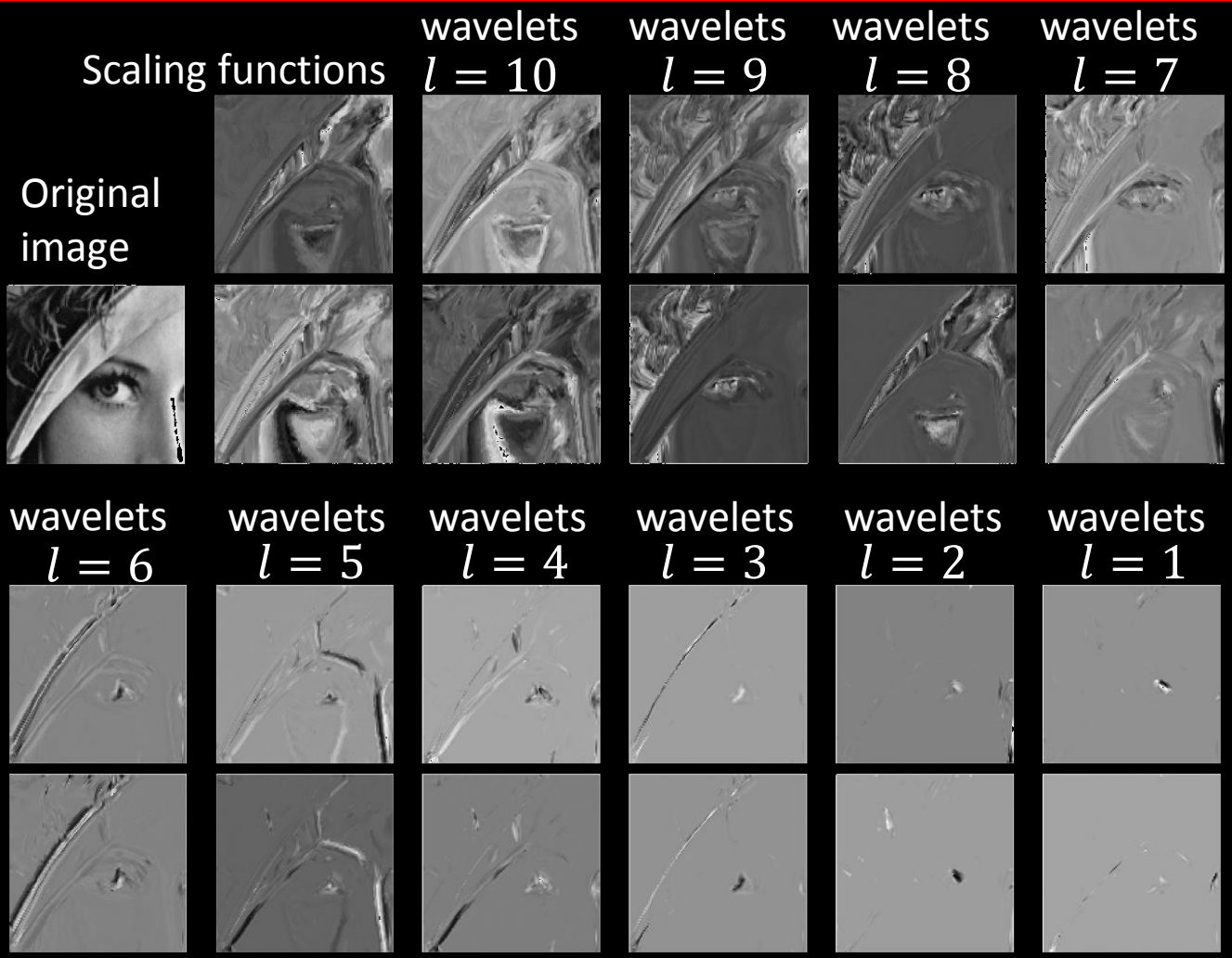
Comparison Between Different Wavelets



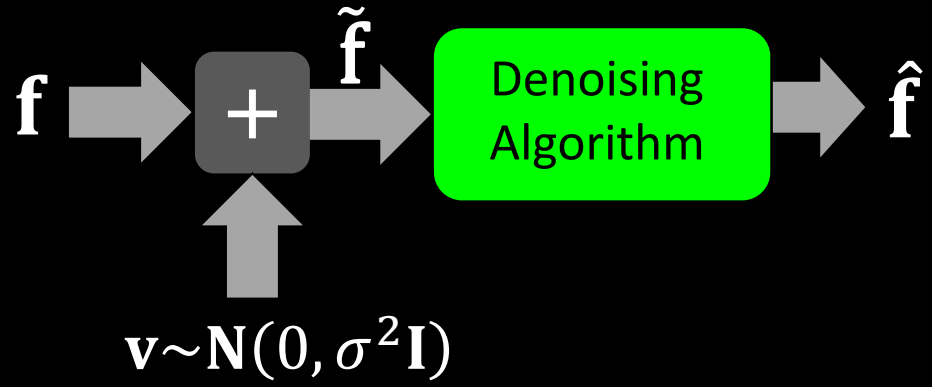
The Representation's Atoms – Synthetic Image



The Representation's Atoms – Lenna



Lets Test It: Image Denoising



Approximation by the THR algorithm:

$$\hat{\mathbf{f}} = \mathbf{D}S_\lambda\{\boldsymbol{\Omega}\tilde{\mathbf{f}}\}$$

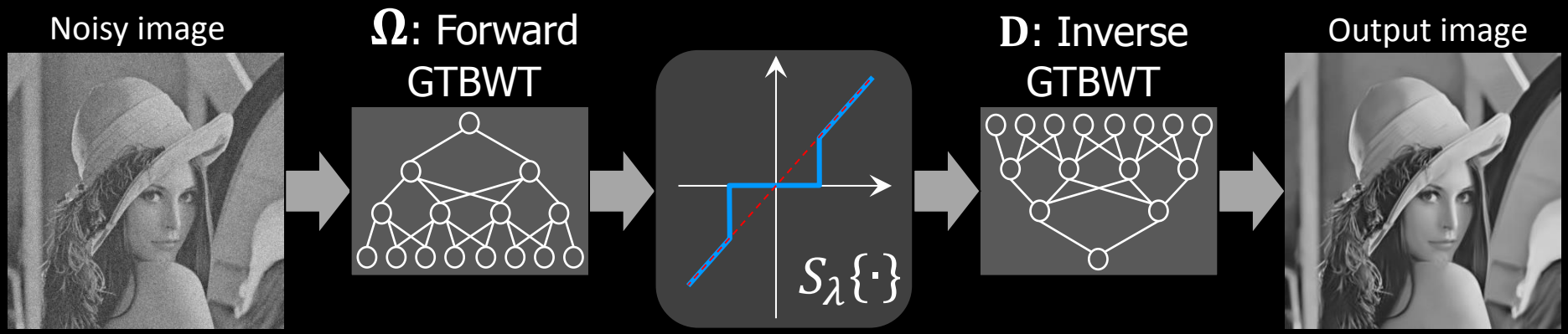


Image Denoising – Improvements

Cycle-spinning: Apply the above scheme several (10) times, with a different GTBWT (different random ordering), and average.

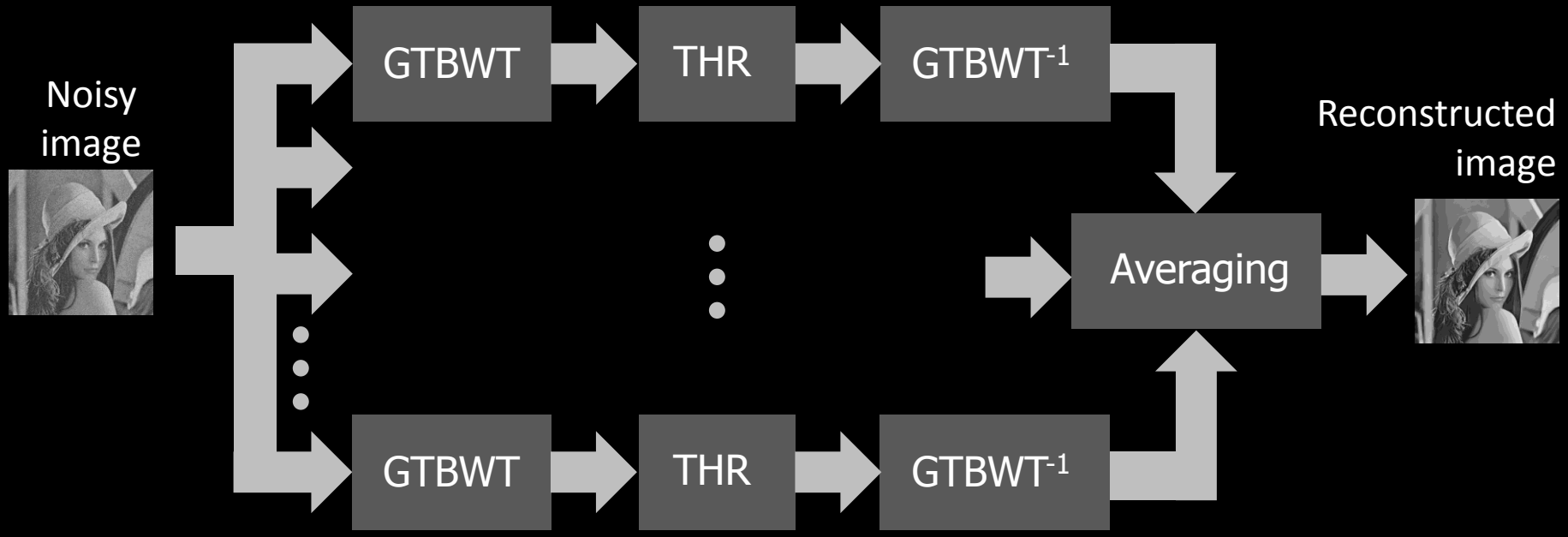


Image Denoising – Improvements

Sub-image averaging: A by-product of GTBWT is the propagation of the whole patches. Thus, we get n transform vectors, each for a shifted version of the image and those can be averaged.

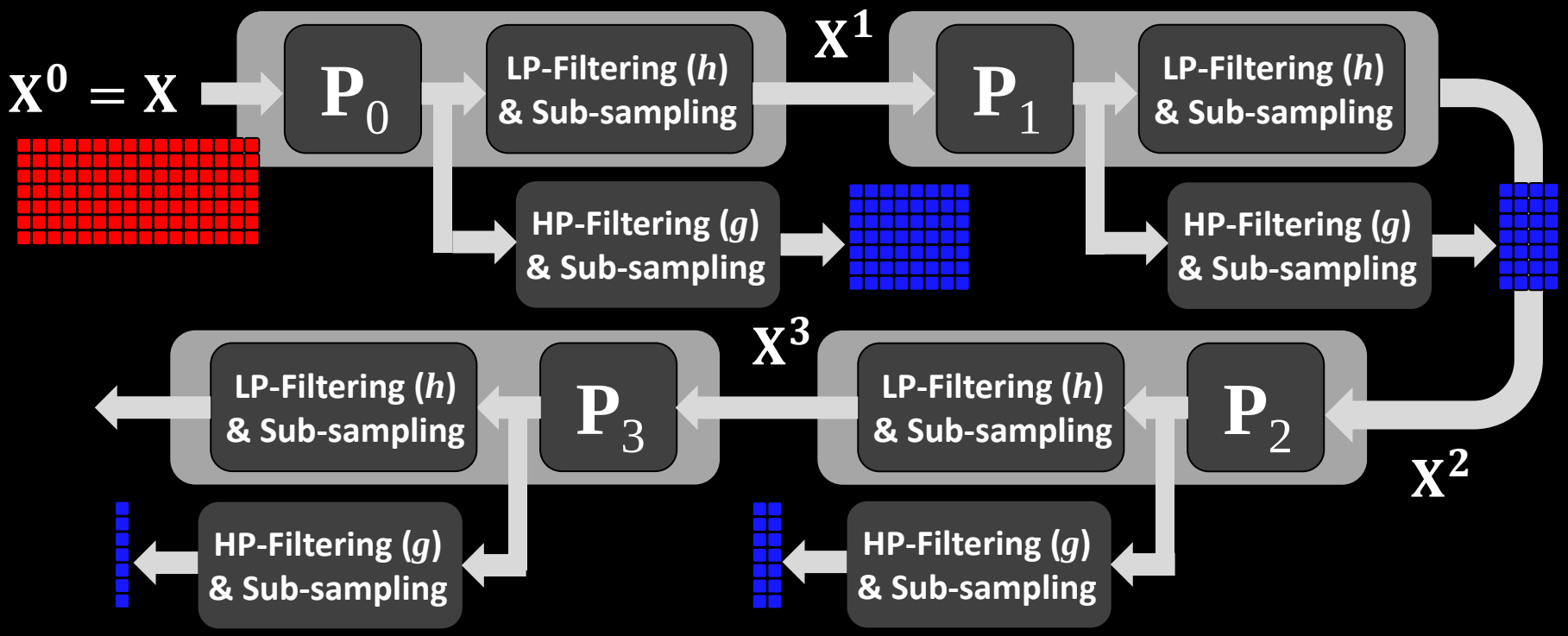
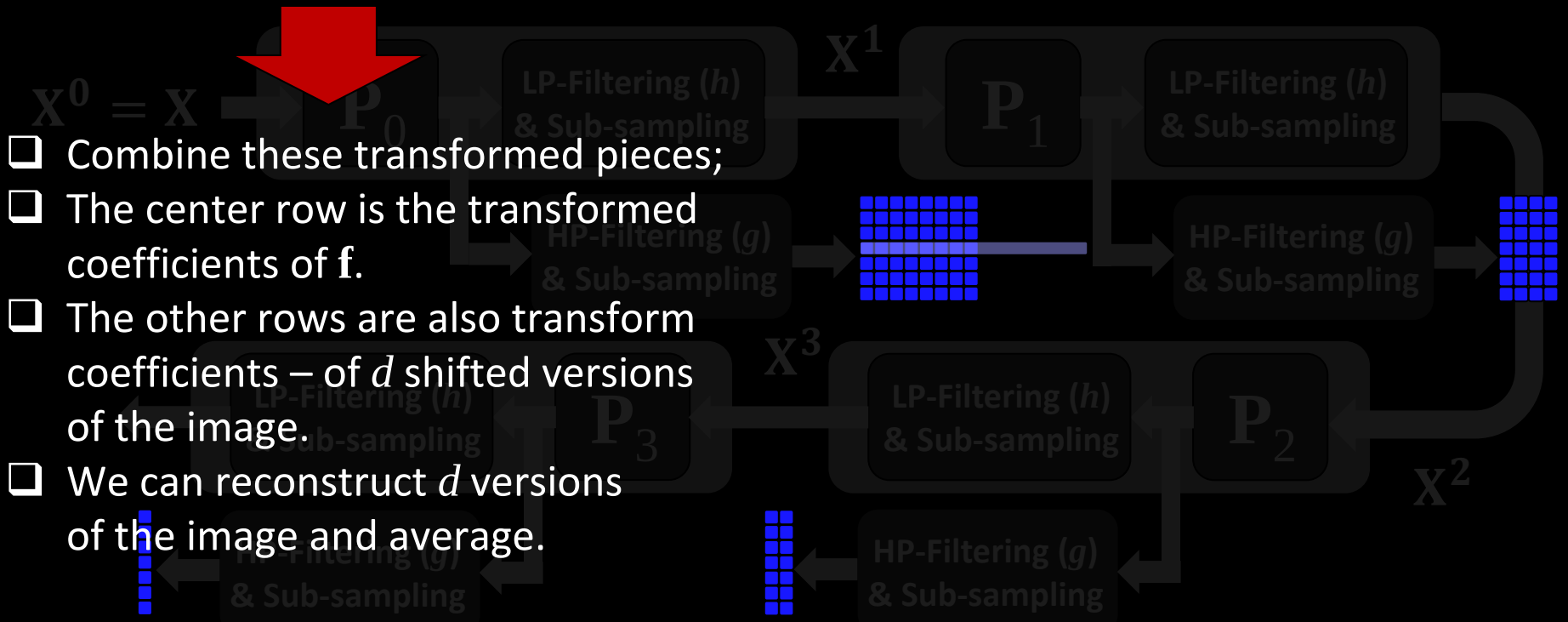


Image Denoising – Improvements

Sub-image averaging: A by-product of GTBWT is the propagation of the whole patches. Thus, we get n transform vectors, each for a shifted version of the image and those can be averaged.



- Combine these transformed pieces;
- The center row is the transformed coefficients of $\mathbf{f}$.
- The other rows are also transform coefficients – of d shifted versions of the image.
- We can reconstruct d versions of the image and average.

Image Denoising – Improvements

Restricting the NN: It appears that when searching the nearest-neighbor for the ordering, restriction to near-by area is helpful, both computationally (obviously) and in terms of the output quality.

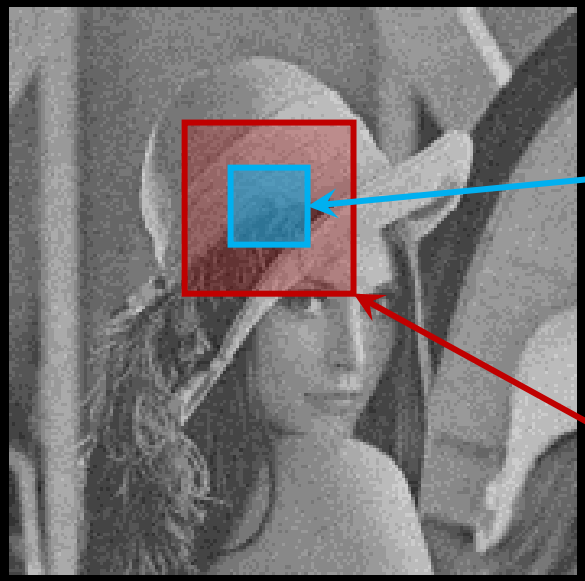
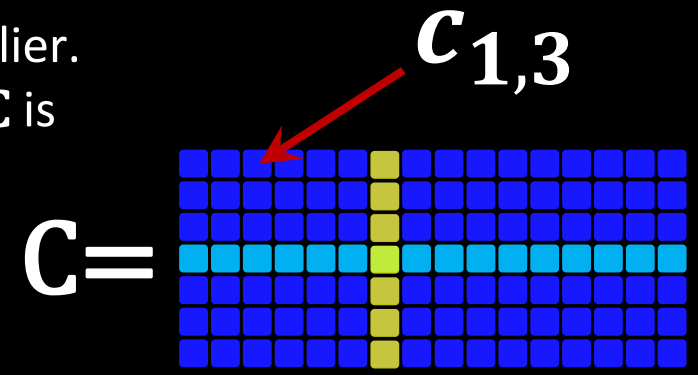


Image Denoising – Improvements

Improved thresholding: Instead of thresholding the wavelet coefficients based on their value, threshold them based on the norm of the (transformed) vector they belong to:

- Recall the transformed vectors as described earlier.
- Classical thresholding: every coefficient within **C** is passed through the function:

$$c_{i,j} = \begin{cases} c_{i,j} & \left|c_{i,j}\right| \geq T \\ 0 & \left|c_{i,j}\right| < T \end{cases}$$



- The proposed alternative would be to force "joint-sparsity" on the above array of coefficients, forcing all rows to share the same support:

$$c_{i,j} = \begin{cases} c_{i,j} & \left\|c_{*,j}\right\|_2 \geq T \\ 0 & \left\|c_{*,j}\right\|_2 < T \end{cases}$$

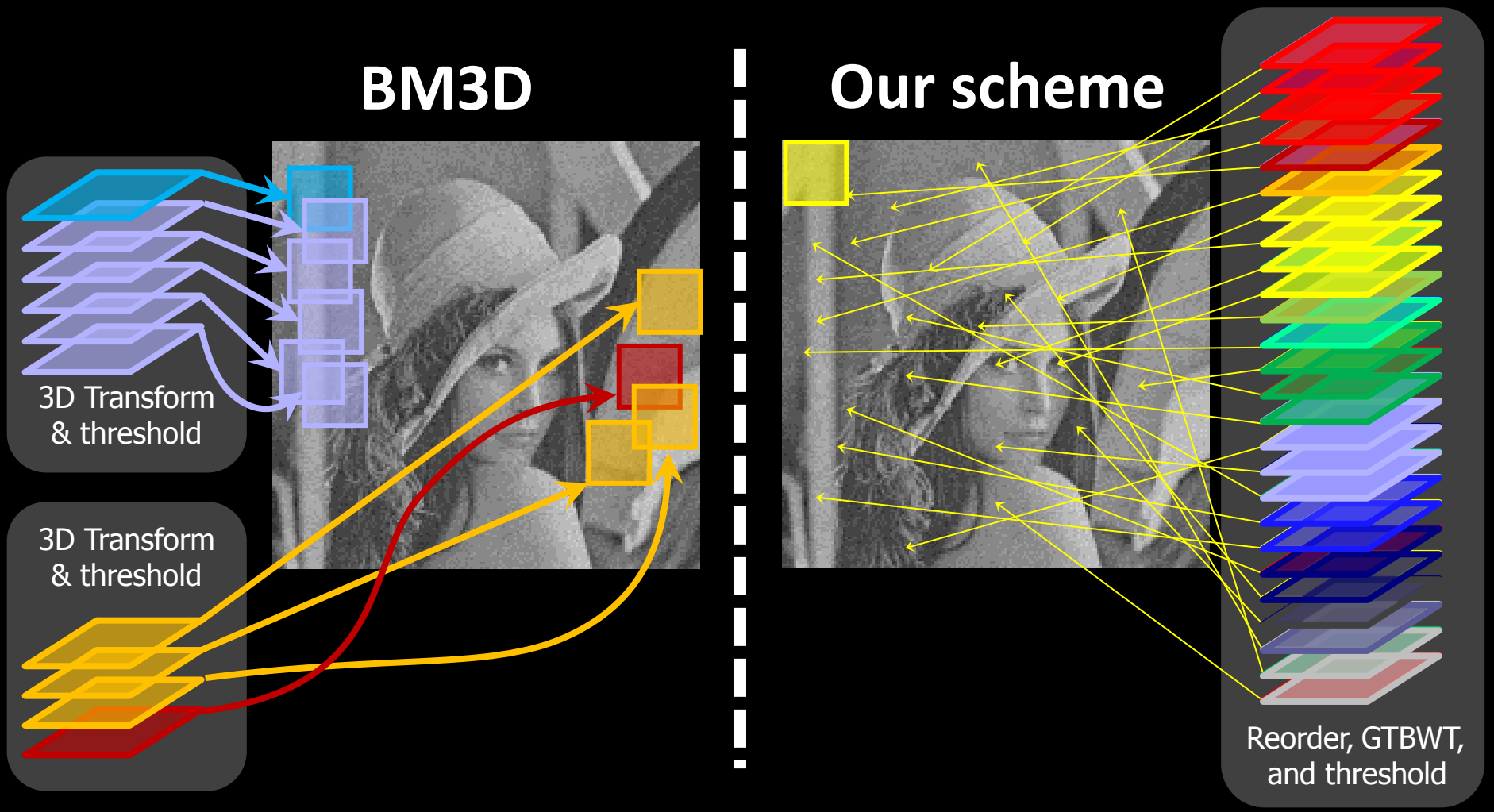
Image Denoising – Results

- We apply the proposed scheme with the Symmlet 8 wavelet to noisy versions of the images Lena and Barbara
- For comparison reasons, we also apply to the two images the K-SVD and BM3D algorithms.

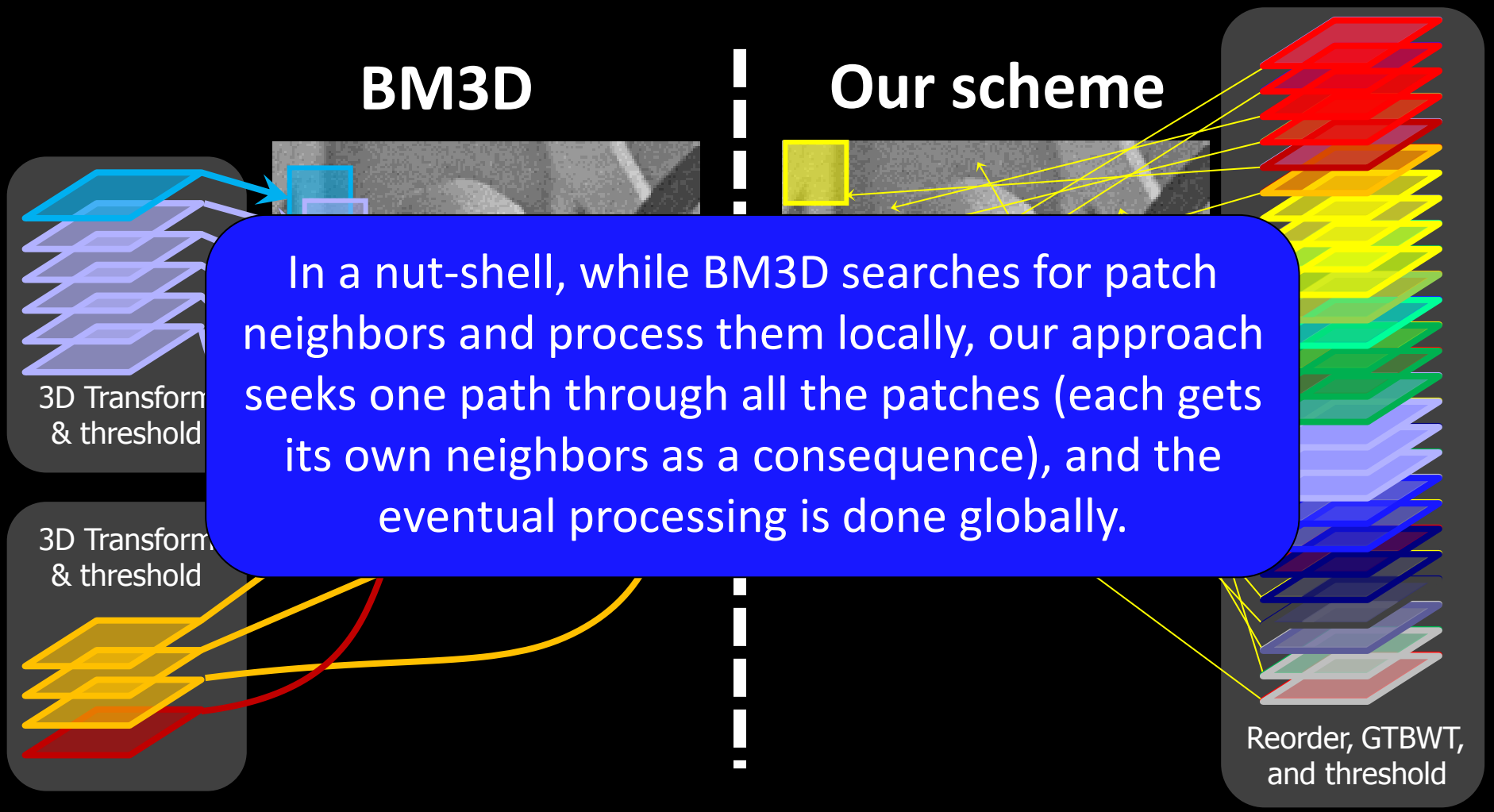
σ/PSNR	Image	K-SVD	BM3D	GTBWT
10/28.14	Lena	35.51	35.93	35.87
	Barbara	34.44	34.98	34.94
25/20.18	Lena	31.36	32.08	32.16
	Barbara	29.57	30.72	30.75

- The PSNR results are quite good and competitive.
- What about run time?

Relation to BM3D?



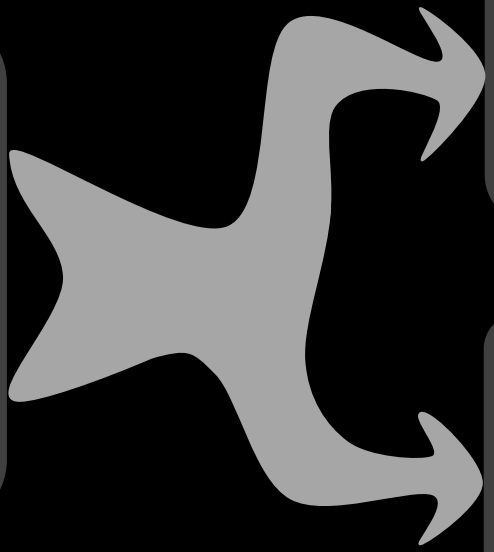
Relation to BM3D?



In a nut-shell, while BM3D searches for patch neighbors and process them locally, our approach seeks one path through all the patches (each gets its own neighbors as a consequence), and the eventual processing is done globally.

What Next?

We have a highly effective sparsifying transform for images. It is "linear" and image adaptive



A: Refer to this transform as an abstract sparsification operator and use it in general image processing tasks

B: Streep this idea to its bones: keep the patch-reordering, and propose a new way to process images

Part III – Frame

Interpreting the GTBWT as a Frame and using it as a Regularizer

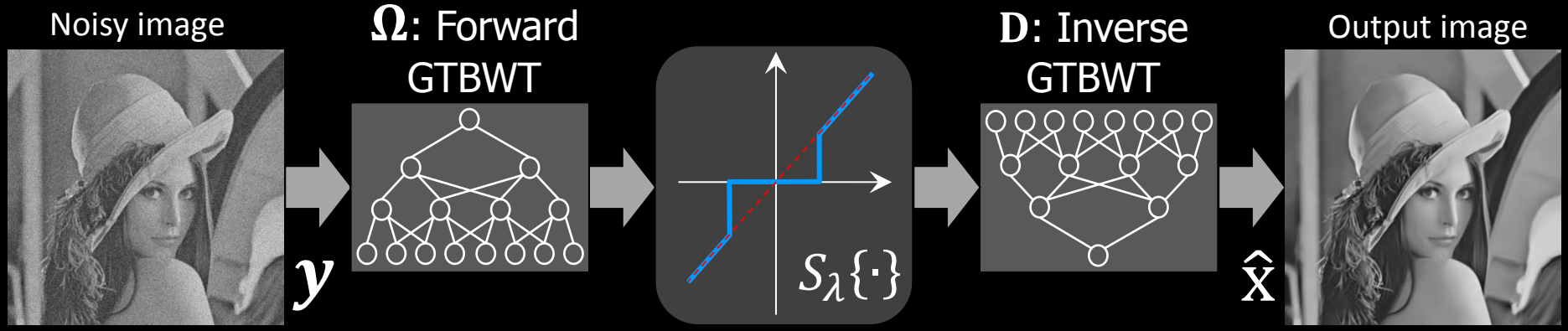
This part is documented in the following draft:

- I. Ram, M. Elad, and I. Cohen, "The RTBWT Frame – Theory and Use for Images", working draft to be submitted soon.

We rely heavily on

- Danielyan, Katkovnik, and Eigiazarian, "BM3D frames and Variational Image Deblurring", IEEE Trans. on Image Processing, Vol. 21, No. 4, pp. 1715-1728, April 2012.

Recall Our Core Scheme



Or, put differently, $\hat{x} = \mathbf{D} \cdot T\{\boldsymbol{\Omega} y\}$: We refer to GTBWT as a redundant frame, and use a "heuristic" shrinkage method with it, which aims to approximate the solution of

Synthesis: $\hat{x} = \mathbf{D} \cdot \underset{\alpha}{\text{Argmin}} \|\mathbf{D}\alpha - y\|_2^2 + \lambda \|\alpha\|_p^p$

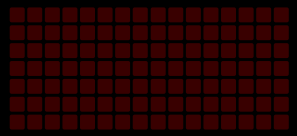
or

Analysis: $\hat{x} = \underset{f}{\text{Argmin}} \|x - y\|_2^2 + \lambda \|\boldsymbol{\Omega} x\|_p^p$

Recall: Our Transform (Frame)

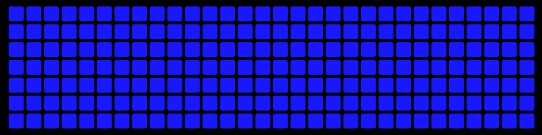
Lexicographic ordering of the N pixels

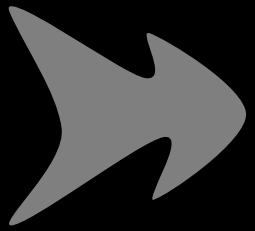
$\mathbf{X}$: Array of overlapped patches of size dN



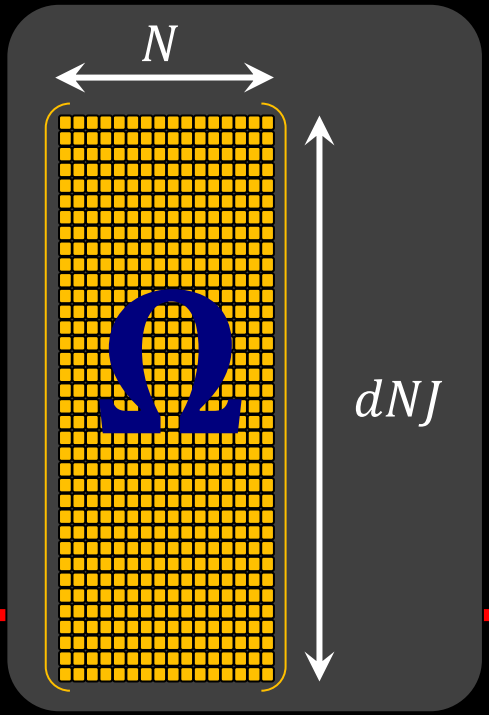
Applying a J redundant wavelet of some sort including permutations

We obtain an array of dNJ transform coefficients

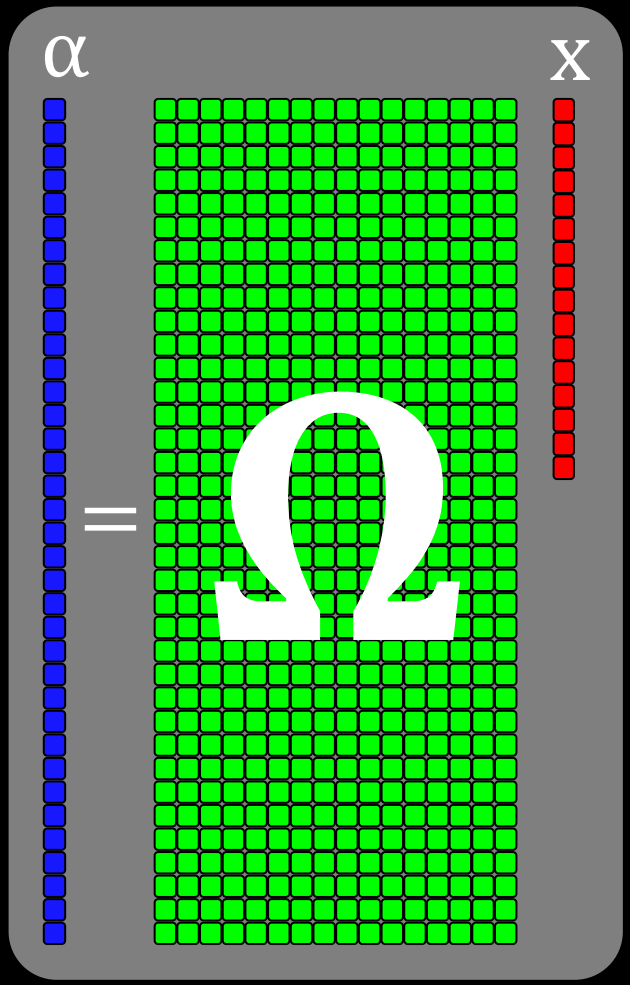


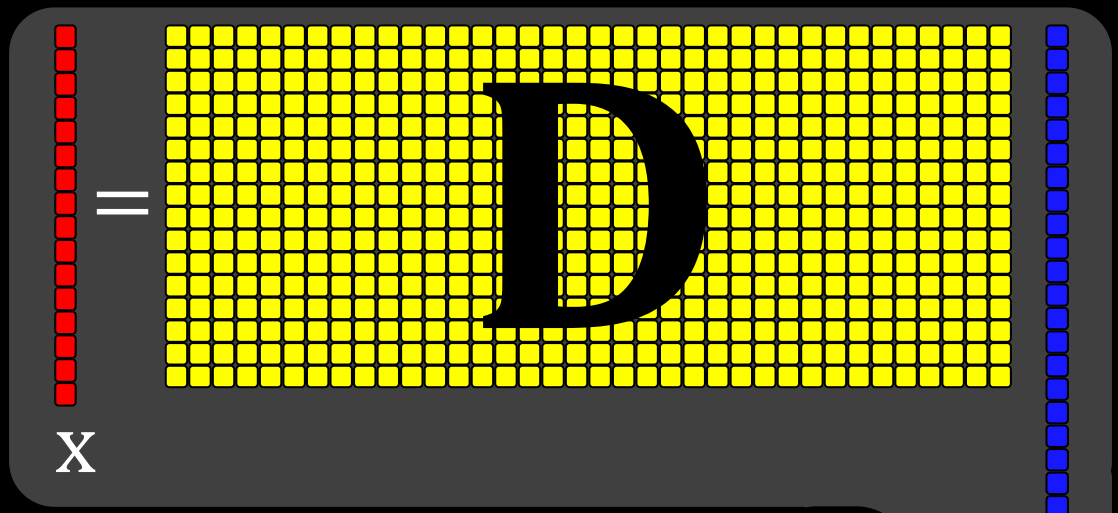


- All these operations could be described as one linear operation: multiplication of $\underline{f}$ by a huge matrix $\mathbf{\Omega}$
- This transform is adaptive to the specific image



Our Notations





$$\mathbf{I} = \mathbf{D}\Omega$$

(Not a Moore-Penrose pair)

α

What Can We Do With This Frame?

We could solve various inverse problems of the form:

$$y = \mathbf{A}x + v$$

where: x is the original image
v is an AWGN, and
$\mathbf{A}$ is a degradation operator of any sort

We could consider the synthesis, the analysis, or their combination:

$$\{\hat{x}, \hat{\alpha}\} = \underset{\alpha, x}{\text{Argmin}} \; \|y - \mathbf{A}x\|_2^2 + \frac{1}{\beta}\|\mathbf{D}\alpha - x\|_2^2 + \lambda\|\alpha\|_p^p + \frac{1}{\mu}\|\mathbf{\Omega}x - \alpha\|_2^2$$

$\beta = 0$, $\mu = \infty$ $\rightarrow$ Synthesis

$\beta = \infty$, $\mu = 0$ $\rightarrow$ Analysis

Generalized Nash Equilibrium*

Instead of minimizing the joint analysis/synthesis problem:

$$\{\hat{\mathrm{x}},\hat{\alpha}\} = \underset{\alpha,\mathrm{x}}{\mathrm{Argmin}} \|\mathrm{y}-\mathbf{A}\mathrm{x}\|_2^2 + \frac{1}{\beta}\|\mathbf{D}\alpha-\mathrm{x}\|_2^2 + +\lambda\|\alpha\|_p^p + \frac{1}{\mu}\|\mathbf{\Omega}\mathrm{x}-\alpha\|_2^2$$

break it down into two separate and easy to handle parts:

and solve iteratively

$$\mathrm{x}_{k+1} = \underset{\mathrm{x}}{\mathrm{Argmin}} \ \|\mathrm{y}-\mathbf{A}\mathrm{x}\|_2^2 + \frac{1}{\beta}\|\mathbf{D}\alpha_k-\mathrm{x}\|_2^2$$

$$\alpha_{k+1} = \underset{\alpha}{\mathrm{Argmin}} \ \lambda\|\alpha\|_p^p + \frac{1}{\mu}\|\mathbf{\Omega}\mathrm{x}_{k+1}-\alpha\|_2^2$$

* Danielyan, Katkovnik, and Eigiazarian, "BM3D frames and Variational Image Deblurring", IEEE Trans. on Image Processing, Vol. 21, No. 4, pp. 1715-1728, April 2012.

Deblurring Results

Original

Blurred

Restored

Deblurring Results

Image	Input PSNR	BM3D-DEB ISNR	IDD-BM3D ISNR init. with BM3D-DEB	Ours ISNR Init. with BM3D-DEB	Ours ISNR 3 iterations with simple initialization
Lena	27.25	7.95	7.97	8.08	8.20
Barbara	23.34	7.80	7.64	8.25	6.21
House	25.61	9.32	9.95	9.80	10.06
Cameraman	22.23	8.19	8.85	9.19	8.52

$$\text{Blur PSF} = \frac{1}{1 + i^2 + j^2} \quad -7 \le i, j \le 7$$

$\sigma^2=2$

Part IV – Patch (Re)-Ordering

Lets Simplify Things, Shall We?

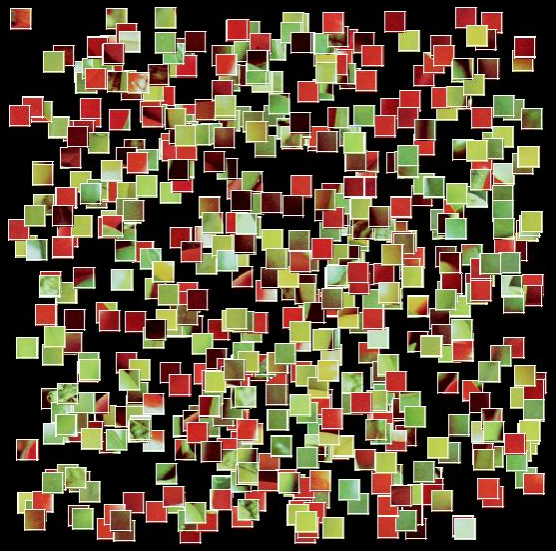
This part is based on the paper:

- I. Ram, M. Elad, and I. Cohen, "Image Processing using Smooth Ordering of its Patches", IEEE Transactions on Image Processing, Vol. 22, No. 7, pp. 2764–2774 , July 2013.

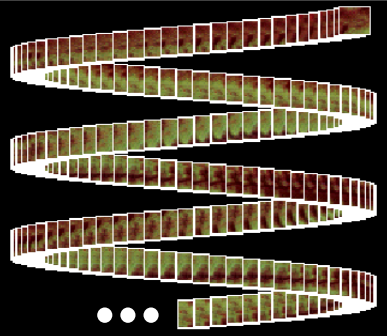
Returning to the Basics

Suppose we start with a clean image

We extract all (with overlaps) patches of size B×B (e.g. B=20)



Then we order these patches to form the shortest path, as before

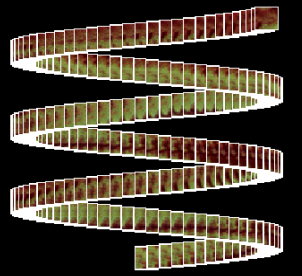


This reordering induces a permutation on the image pixels

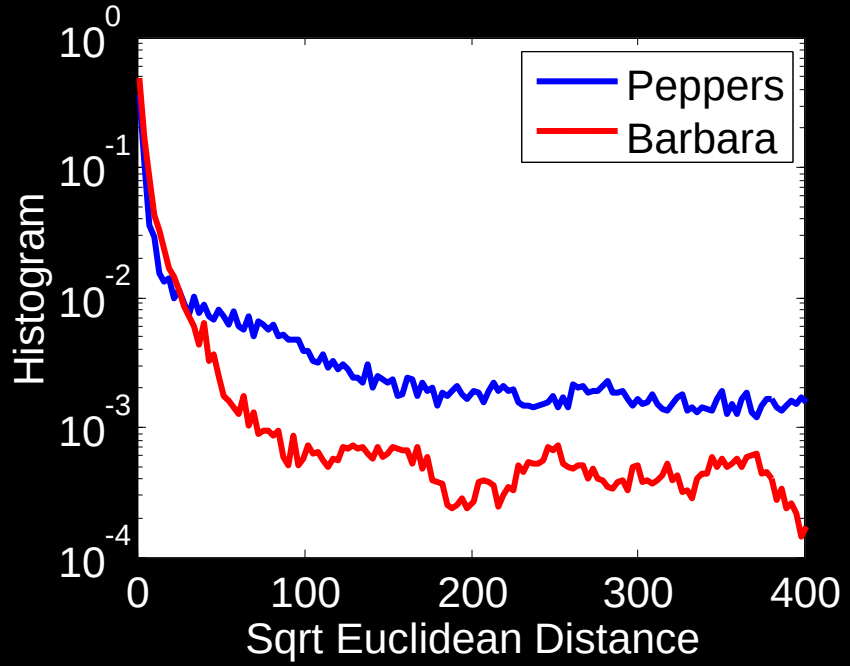
What should we expect from this permutation?

Spatial Neighbor ≠ Euclidean Neighbor

What should we expect?

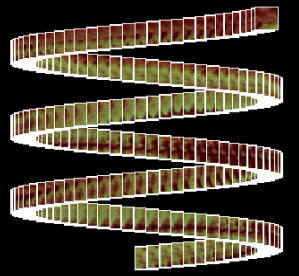


Spatial neighbors are not necessarily expected to remain neighbors in the new ordering

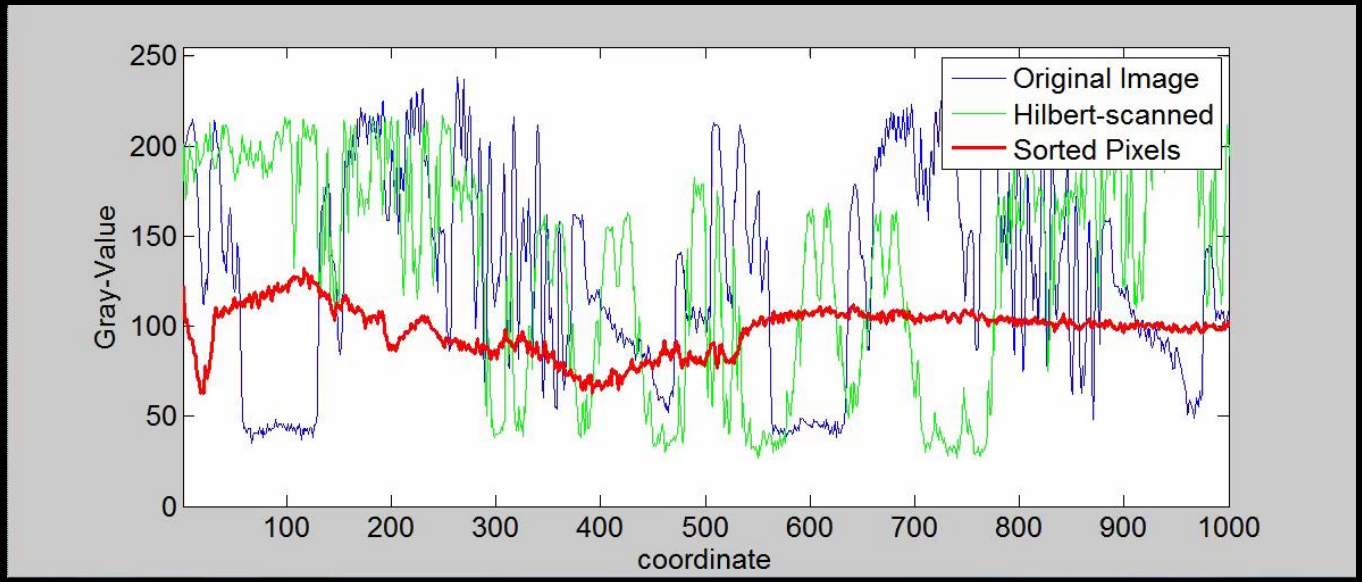


The Reordered Signal is More Regular

What should we expect?



- The new path is expected to lead to very smooth* (or at least, piece-wise smooth) 1D signal.
- The ordering is expected to be robust to noise and degradations → the underlying signal should still be smooth.



* Measure of smoothness:

$$\frac{1}{L}\sum_{k=2}^{L}|\mathrm{f}[k]-\mathrm{f}[k-1]|$$

1. Raster scan: 9.57
2. Hilbert curve: 11.77
3. Sorted (ours): 5.63

Processing the Permuted Pixels

Assumptions:

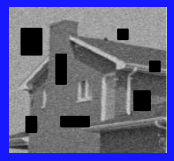
- After a shortest-path reordering of the patches form a clean image, we expect a highly regular signal.
- Reordering a **corrupted** image is likely to lead to a good quality sort as well, due to the robustness brought by the patch-matching.

An Idea: Given a corrupted image of the form:

$$y = \mathbf{M}x + v$$

where: x is the original image
v is an AWGN, and
$\mathbf{M}$ is a point-wise degradation operator,

Apply this process:



Corrupted Image $\tilde{f}$

Re-order the pixels to a 1D signal

Process the 1D signal with a simple filter

Re-order the pixels back to their location

Use the Reordering for Denoising

Noisy with σ=25 (20.18dB)

Reconstruction: 32.65dB *

* This result is obtained with (i) cycle-spinning, (ii) sub-image averaging, (iii) two iterations, (iv) learning the filter , and (v) switched smoothing.

Extract all (with overlaps) patches of size 10×10

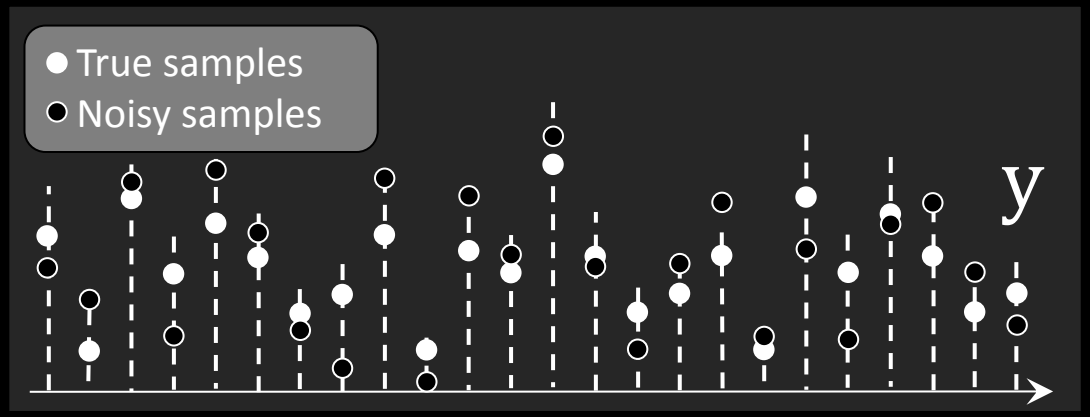
Order these patches as before (TSP)

Take the center-row – it represents a permutation of the image pixels to a regular function

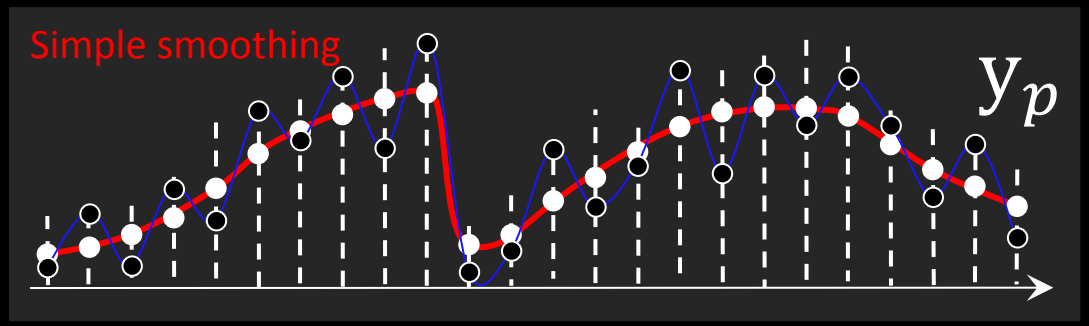
Smooth/filter the values along this row in a simple way

Intuition: Why Should This Work?

Noisy with σ=25 (20.18dB)



Ordering based on the noisy pixels



Reconstruction: 32.65dB

The “Simple Smoothing” We Do

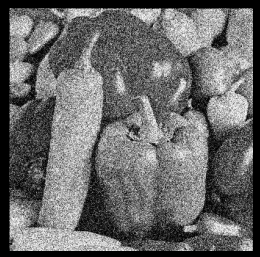
Simple smoothing works fine

but

We can do better by a training phase

Training image

Original image



Noisy image

Apply the permutation on the pixels

Compute the TSP permutation

Apply the permutation on the pixels

Apply a 1D filter h

+

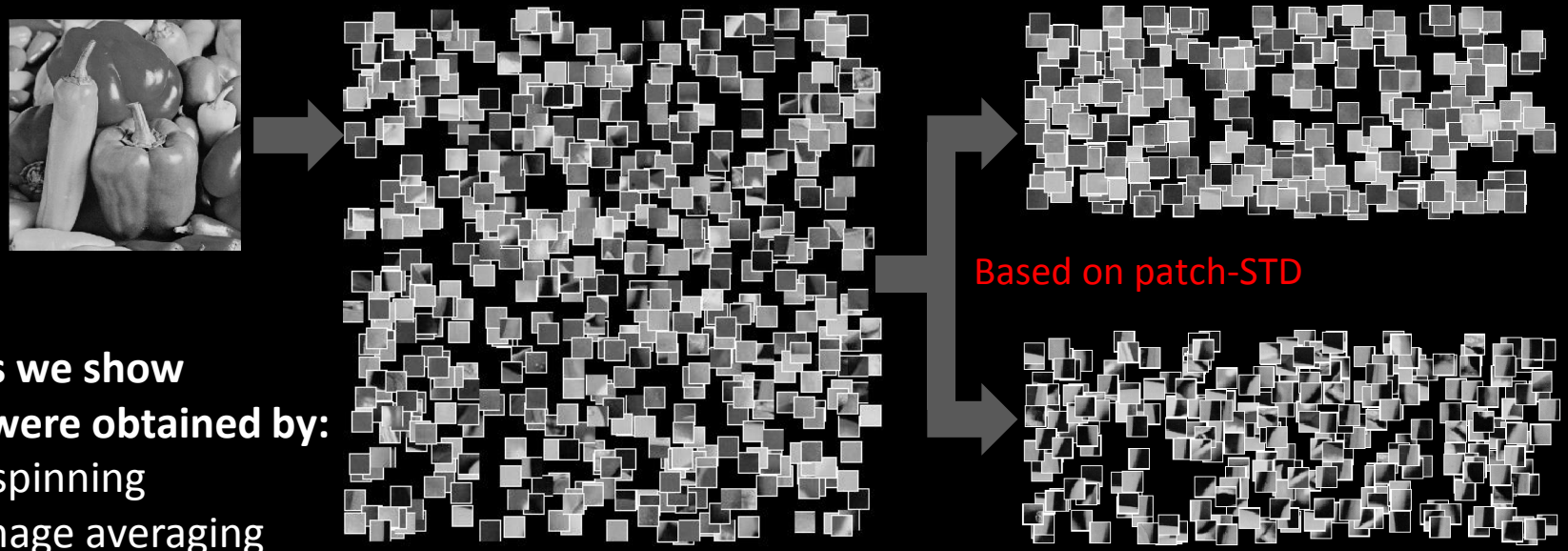
-

optimize h to minimize the reconstruction MSE

Naturally, this is done off-line and on other images

Filtering – A Further Improvement

Cluster the patches to smooth and textured sets, and train a filter per each separately



The results we show hereafter were obtained by:

(i) Cycle-spinning
(ii) Sub-image averaging
(iii) Two iterations
(iv) Learning the filter , and
(v) Switched smoothing.

Denoising Results Using Patch-Reordering

Image		σ/PSNR [dB]		
		10 / 28.14	25 / 20.18	50 / 14.16
Lena	K-SVD	35.49	31.36	27.82
	1st iteration	35.33	31.58	28.54
	2nd iteration	35.41	31.81	29.00
Barbara	K-SVD	34.41	29.53	25.40
	1st iteration	34.48	30.46	27.17
	2nd iteration	34.46	30.54	27.45
House	K-SVD	36.00	32.12	28.15
	1st iteration	35.58	32.48	29.37
	2nd iteration	35.94	32.65	29.93

Bottom line: (1) This idea works very well;
(2) It is especially competitive for high noise levels; and
(3) A second iteration almost always pays off.

What About Inpainting?

0.8 of the pixels are missing

Extract all (with overlaps) patches of size 9×9

Order these patches as before distance uses EXISTING pixels only

Take the center-row – it represents a permutation of the image pixels to a regular function

Fill the missing values in a simple (cubic interpolation) way

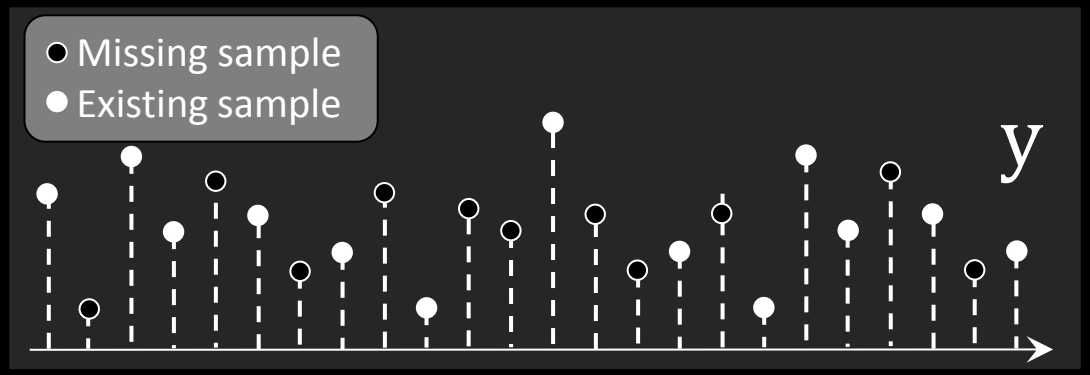
Reconstruction: 29.71dB*

* This result is obtained with (i) cycle-spinning, (ii) sub-image averaging, and (iii) two iterations.

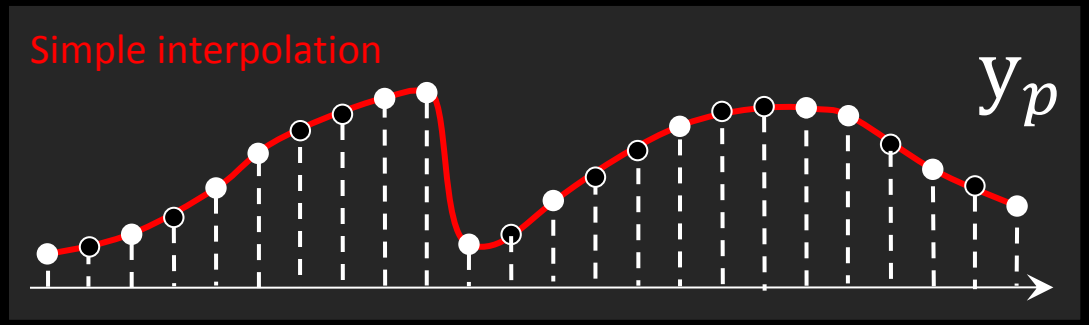
The Rationale

0.8 of the pixels are missing

Reconstruction: 27.15dB



Ordering



Inpainting Results – Examples

Given data 80% missing pixels	Bi-Cubic interpolation	DCT and OMP recovery	1st iteration of the proposed alg.	3rd iteration of the proposed alg.

Inpainting Results

Reconstruction results from 80% missing pixels using various methods:

Image	Method	PSNR [dB]
Lena	Bi-Cubic	30.25
	DCT + OMP	29.97
	Proposed (1st iter.)	30.25
	Proposed (2nd iter.)	31.80
	Proposed (3rd iter.)	31.96
Barbara	Bi-Cubic	22.88
	DCT + OMP	27.15
	Proposed (1st iter.)	27.56
	Proposed (2nd iter.)	29.34
	Proposed (3rd iter.)	29.71
House	Bi-Cubic	29.21
	DCT + OMP	29.69
	Proposed (1st iter.)	29.03
	Proposed (2nd iter.)	32.10
	Proposed (3rd iter.)	32.71

Bottom line:

(1) This idea works very well;

(2) It is operating much better than the classic sparse-rep. approach; and

(3) Using more iterations always pays off, and substantially so.

Part IV – Time to Finish

Conclusions and a Bit More

Conclusions

We propose a new wavelet transform for scalar functions defined on graphs or high dimensional data clouds

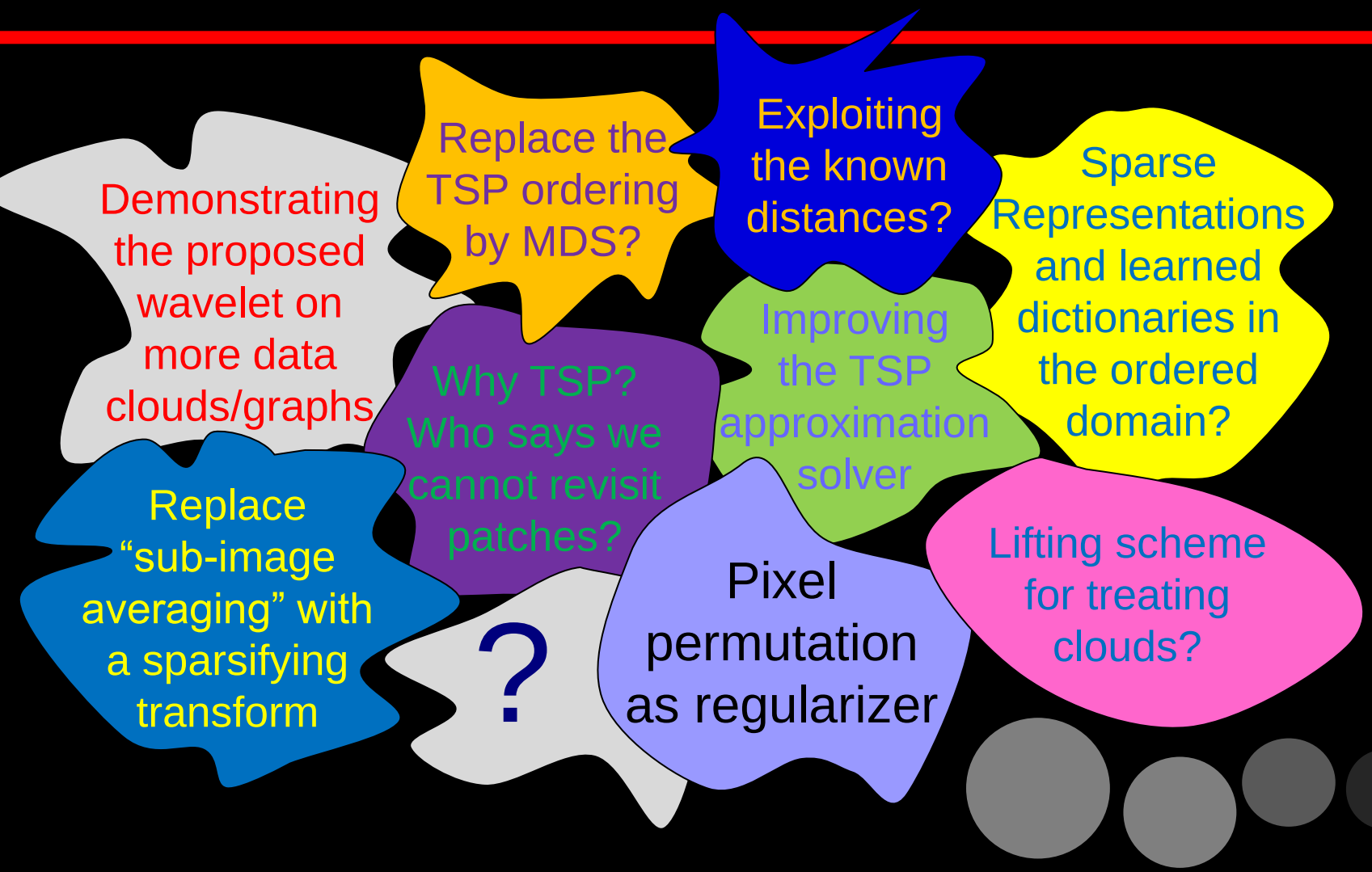
The proposed transform extends the classical orthonormal and redundant wavelet transforms

We demonstrate the ability of these transforms to efficiently represent and denoise images

We also show that the obtained transform can be used as a regularizer in classical image processing Inverse-Problems

Finally, we show that using the ordering of the patches only, quite effective denoising and inpainting can be obtained

What Next ?


Demonstrating the proposed wavelet on more data clouds/graphs
Replace the TSP ordering by MDS?
Exploiting the known distances?
Sparse Representations and learned dictionaries in the ordered domain?
Improving the TSP approximation solver
Why TSP? Who says we cannot revisit patches?
Replace "sub-image averaging" with a sparsifying transform
?
Pixel permutation as regularizer
Lifting scheme for treating clouds?

Thank you for your time

thanks to the organizers
of this event
Volkan Cevher and Matthias Seeger

Questions?

Post-Docs Are Needed