3rd IEEE Global Conference on Signal & Information Processing
Orlando, Florida, USA December 14-16 2015

December 15, 2015

Globalized BM3D using Fast Eigenvalue Filtering

Tokyo University of Agriculture and Technology

Koki Suwabe, Masaki Onuki, Yuki Iizuka and Yuchi Tanaka

MSP Lab Multidimensional media Signal Processing
Graduate School of BASE, TUAT

Outline

* Image denoising
* Previous method
  * Improving method by eigenvalue filtering for denoising
  * Eigenvalue filtering using Chebyshev polynomial approximation
  * BM3D
* Proposed method
* Evaluation
* Conclusion

Image Denoising

Image denoising: estimating the true image from the observed image

Observation model

$$\mathbf{z} = \mathbf{y} + \mathbf{n}$$

noise $\mathbf{n} \sim \mathcal{N}(0, \sigma^2)$

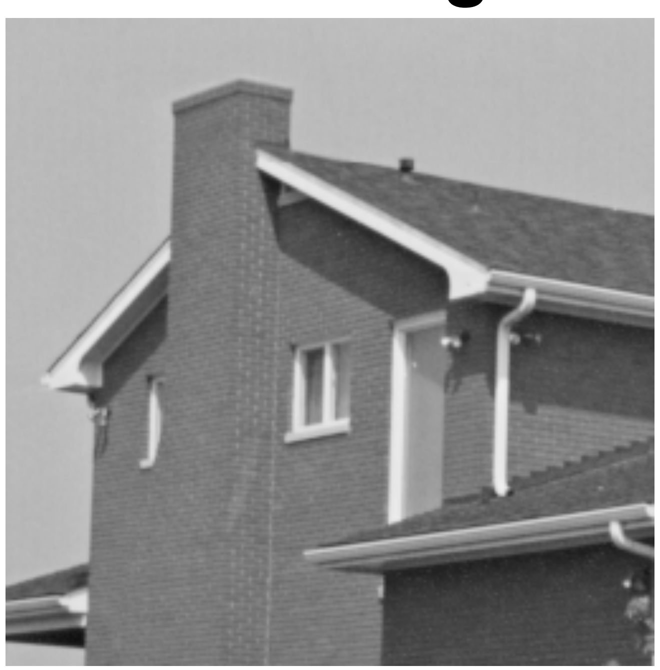
$\mathbf{z} \in \mathbb{R}^N$: Observed image

$\mathbf{y} \in \mathbb{R}^N$: True image

$\mathbf{n} \in \mathbb{R}^N$: Noise signal

N : The number of pixels

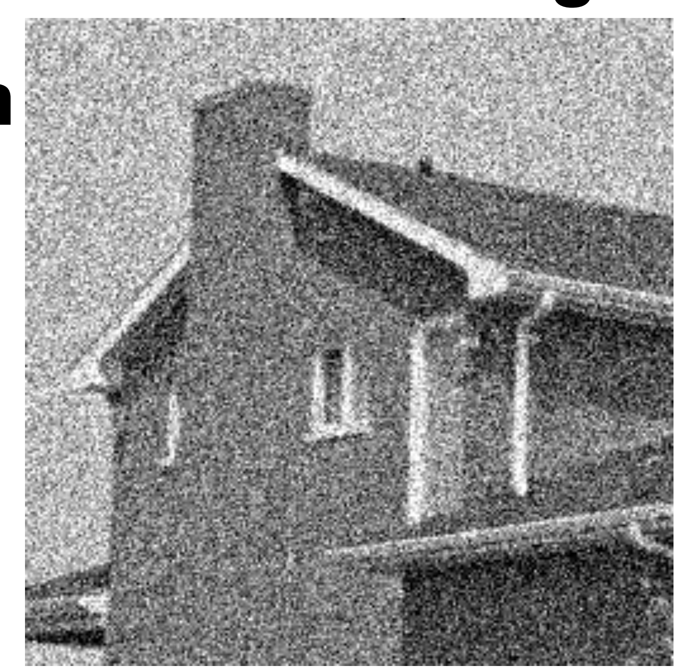
True image



Noise contamination

Denoising

Observed image



Filter Matrix and Its Decomposition

- **Denoising methods can be expressed as $\mathbf{W} \in \mathbb{R}^{N \times N}$**

  Ex.) Gaussian Filter, Bilateral Filter, Non-local means

  Restored image

$$\hat{\mathbf{y}} = \mathbf{W}\mathbf{z}$$

- **The filter matrix is decomposed as**

$$\mathbf{W} = \mathbf{V}\mathbf{S}\mathbf{V}^{-1}$$

Eigenvalue matrix

$$\mathbf{S} = \mathrm{diag}[\lambda_1 \cdots \lambda_N]$$

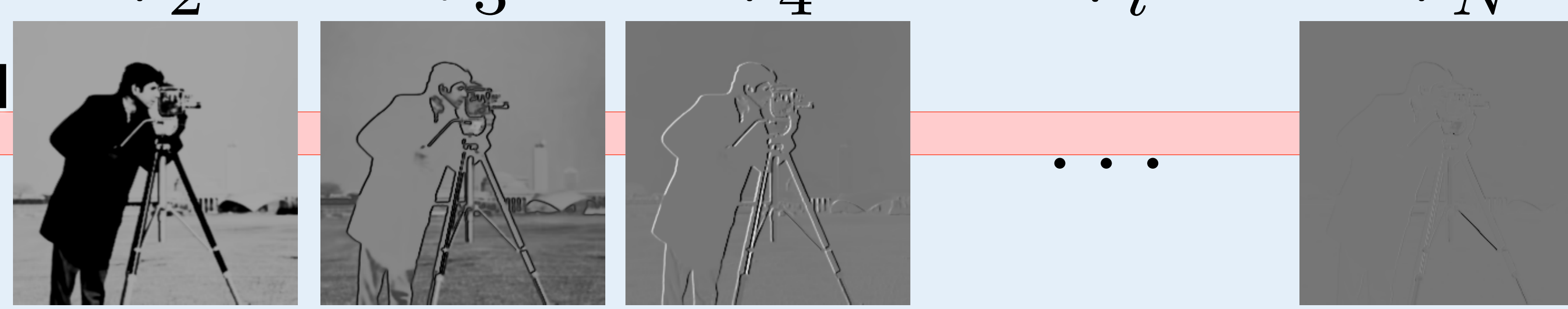
Eigenvector matrix

$$\mathbf{V} = [\mathbf{v}_1 \cdots \mathbf{v}_N]$$

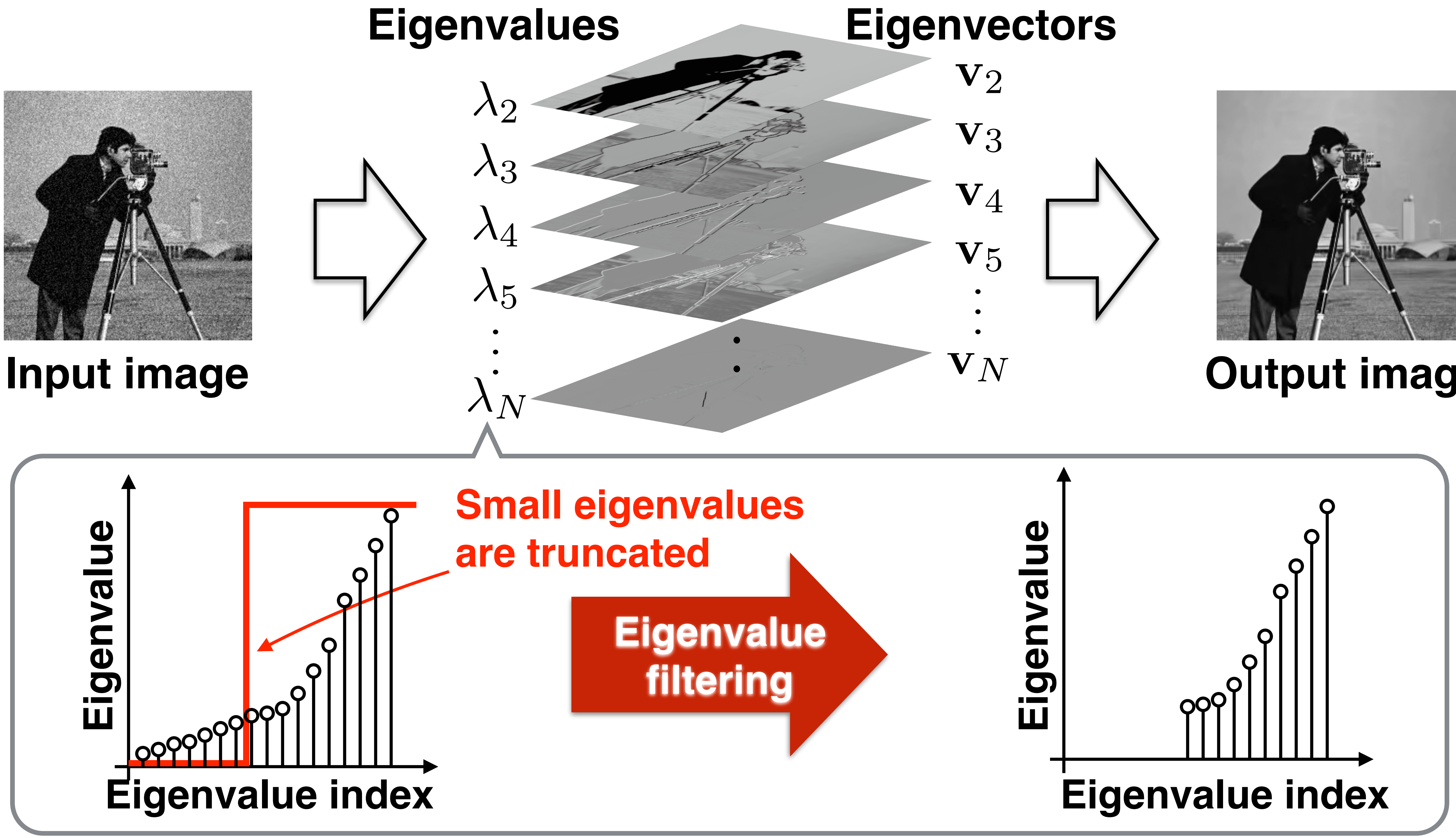
Eigenvalue — **Large** λ_2 λ_3 λ_4 $\cdots$ λ_i $\cdots$ λ_N **Small**

$\mathbf{v}_2$ $\mathbf{v}_3$ $\mathbf{v}_4$ $\cdots$ $\mathbf{v}_i$ $\cdots$ $\mathbf{v}_N$

Elements of eigenvector — **Oscillated slowly** $\cdots$ **Oscillated rapidly**



Eigenvalue Filtering for the Filter Matrix



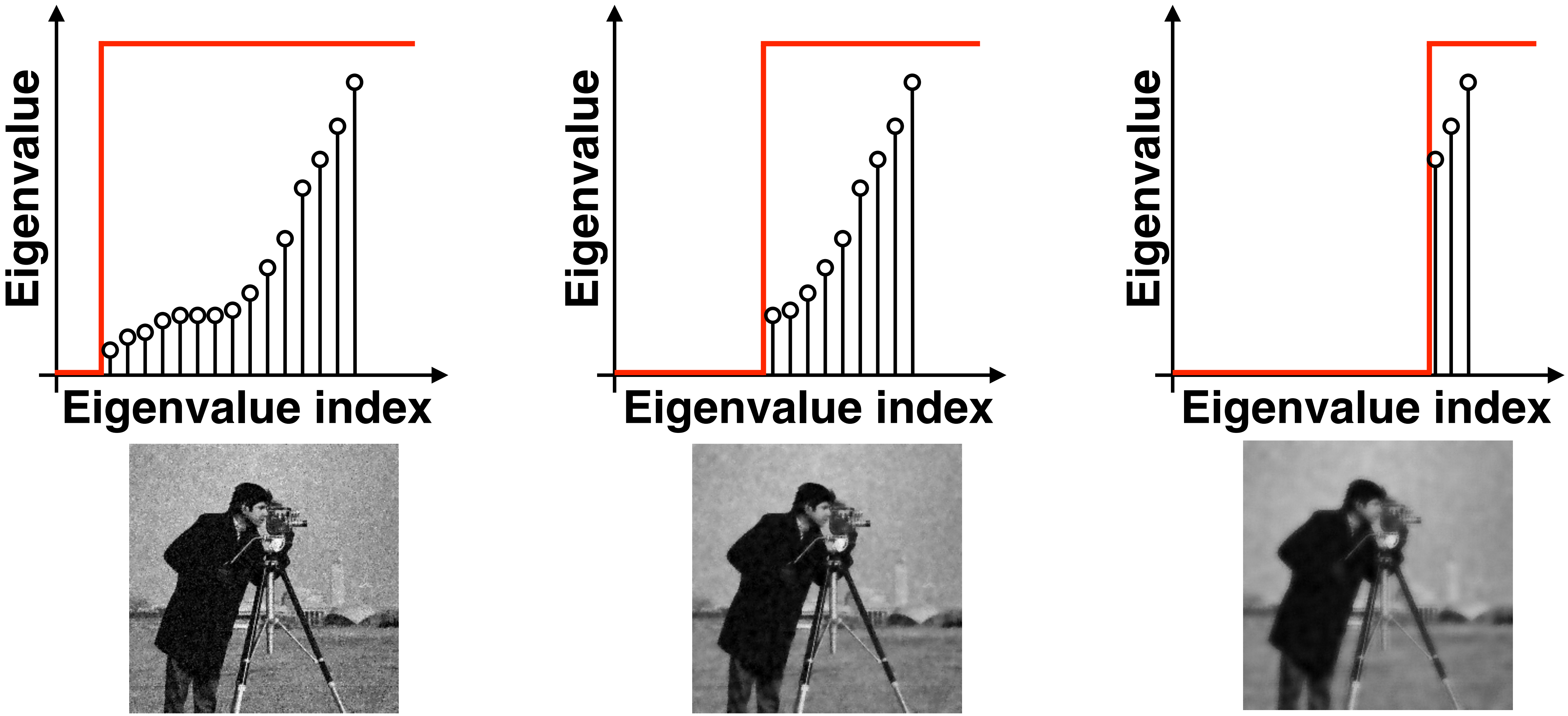
→ **The restored image becomes smoother**

Eigenvalue Filtering for the Filter Matrix

Eigenvalue filtering

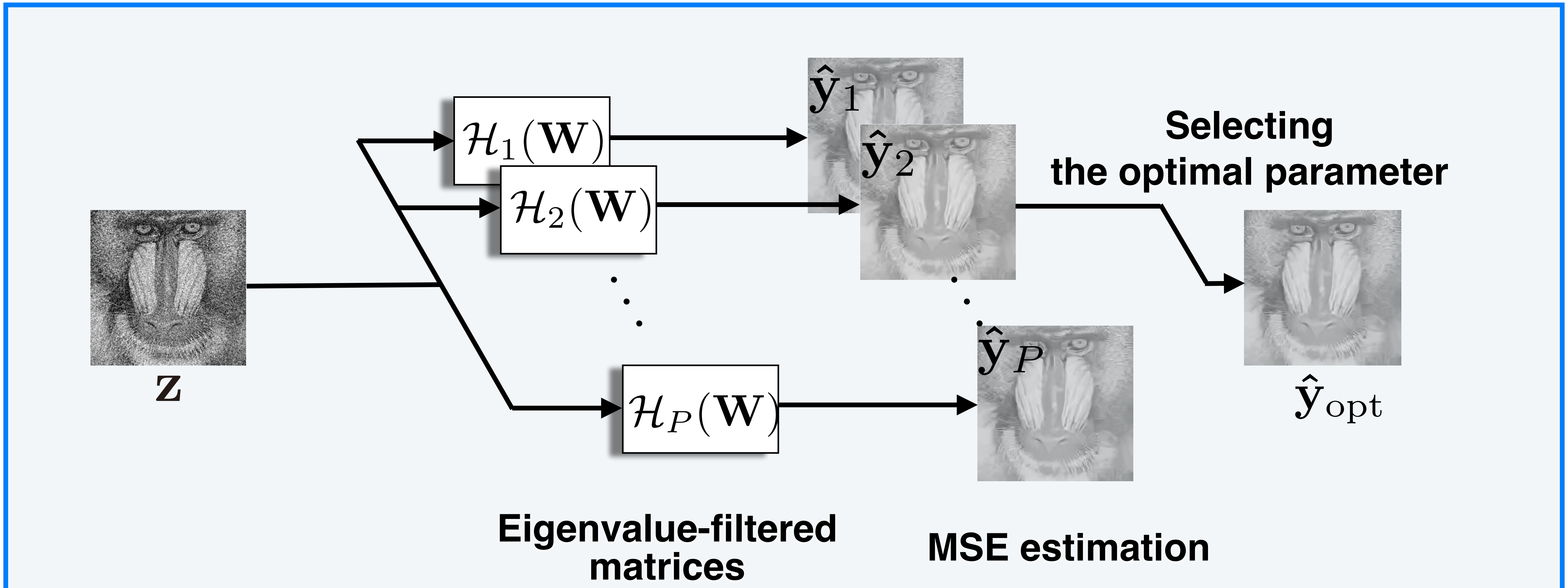
$$\mathcal{H}(\mathbf{W}) = \mathbf{V}\mathrm{diag}(h(\lambda_1), \cdots, h(\lambda_i), \cdots, h(\lambda_N))\mathbf{V}^{-1}$$

$h(\cdot)$: Arbitrary filter kernel



- **The smoothing strength is controlled according to the filter kernel**

Parameter Selection of Eigenvalue Filtering



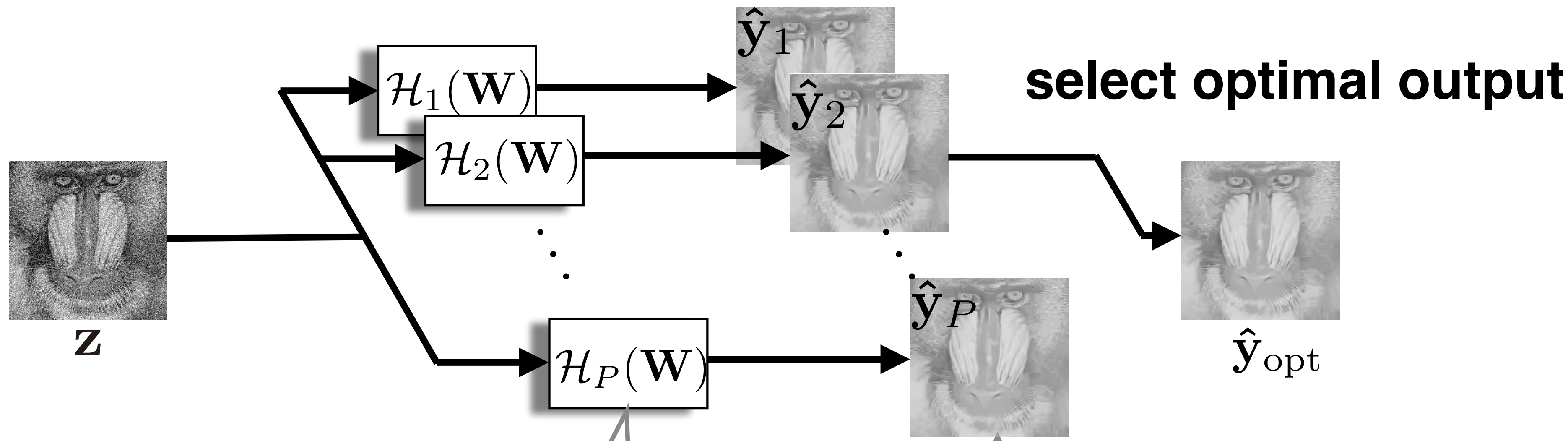
I. **Perform eigenvalue filtering using various filter kernels controlled by the parameter**

II. **Obtain restored images using each eigenvalue-filtered matrices**

III. **Estimate MSEs of each restored image**

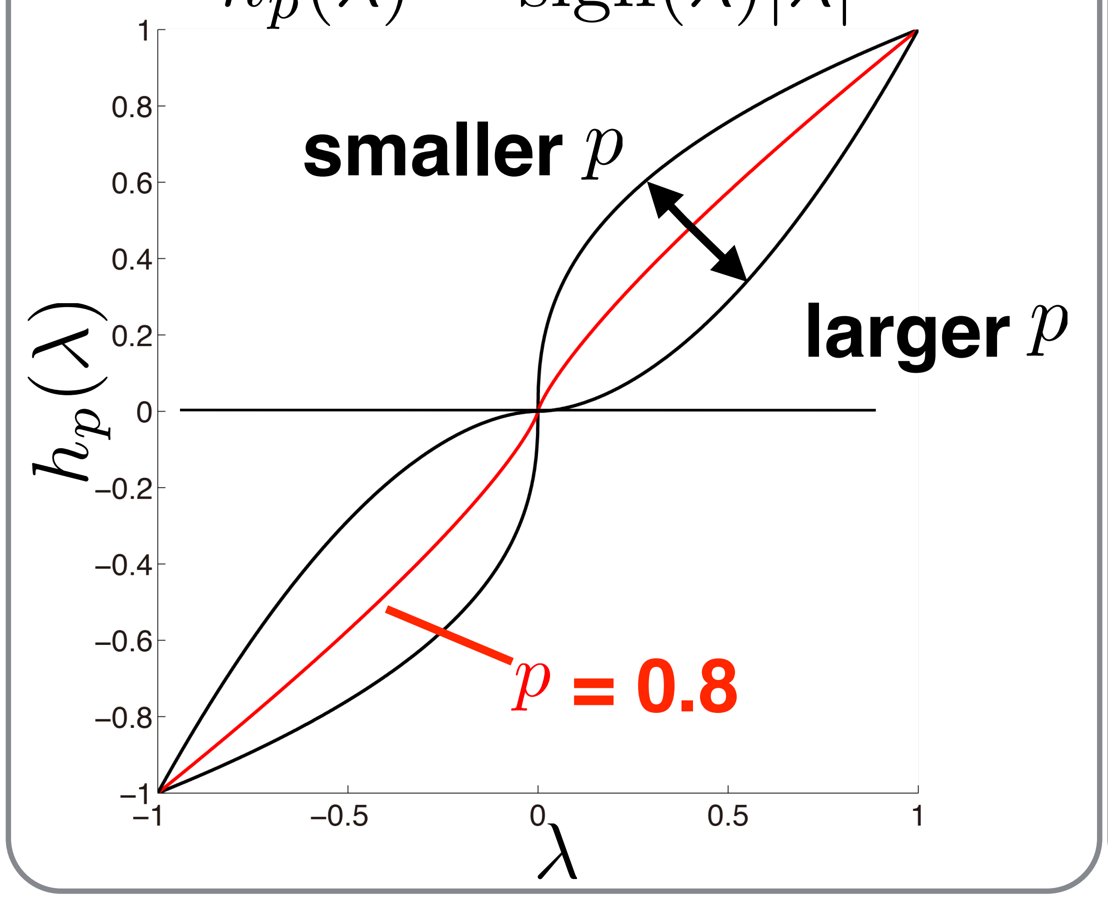
IV. **Select an optimal output (an image having minimum MSE)**

Improving Method by Eigenvalue Filtering

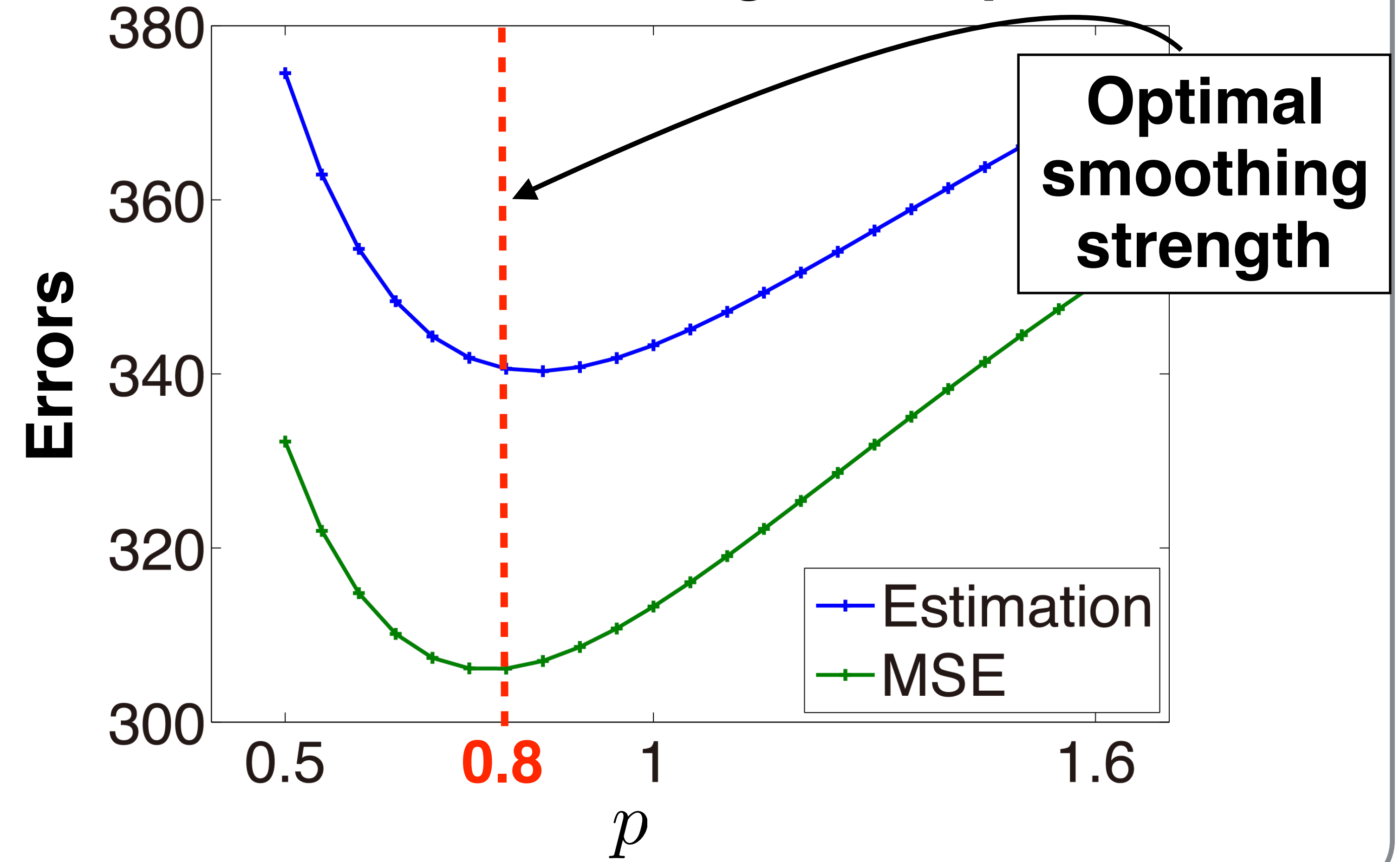


Filter kernel

$$h_p(\lambda) = \text{sign}(\lambda)|\lambda|^p$$



MSE transition according to the parameter p



Approximation of Filter Kernels by CPA

- **Eigendecomposition takes much computational cost**

➡ **Eigenvalue filtering by Chebyshev polynomial approximation(CPA)** [1]

CPA for scalar function

$$h(y) = \frac{1}{2}c_0 + \sum_{k=1}^{\infty} c_k T_k(y)$$

$h(\cdot)$: Arbitrary function

Chebyshev polynomial

$$T_k(y) = \cos(k \arccos(y))$$

Chebyshev coefficient

$$c_k = \frac{2}{\pi} \int_{-1}^{1} \frac{T_k(y)h(y)}{\sqrt{1-y^2}}\, dy = \frac{2}{\pi} \int_{0}^{\pi} \cos(k\theta) h(\cos\theta)\, d\theta$$

Chebyshev polynomials are obtained by recurrence relation

Recurrence relation	$T_k(y) = 2yT_{k-1}(y) - T_{k-2}(y)$
Initial conditions	$T_0(y) = 1,\ T_1(y) = y$

[1] M. Onuki, S. Ono, K. Shirai, and Y. Tanaka, "Non-local/local image filters using fast eigenvalue filtering," in *Proc. ICIP*, 2015.

Eigenvalue Filtering by CPA

- **Eigenvalue filtering can be realized without eigendecomposition**

$$\hat{\mathbf{y}} = \mathcal{H}(\mathbf{W})\mathbf{z} = \left(\frac{1}{2}c_0\mathbf{I} + \sum_{k=1}^{d} c_k \mathcal{T}_k(\mathbf{W}) \right) \mathbf{z}$$

CPA for a filter matrix

$$\mathcal{H}(\mathbf{W}) = \frac{1}{2}c_0\mathbf{I} + \sum_{k=1}^{\infty} c_k \mathcal{T}_k(\mathbf{W})$$

Chebyshev polynomial

$$\mathcal{T}_k(\mathbf{W}) = \mathbf{V}\mathrm{diag}(\cos k\theta_1, \ldots, \cos k\theta_i, \ldots, \cos k\theta_N)\mathbf{V}^{-1}$$

Chebyshev coefficient

$$c_k = \frac{2}{\pi}\int_0^{\pi} \cos(k\theta) h(\cos\theta)\, d\theta$$

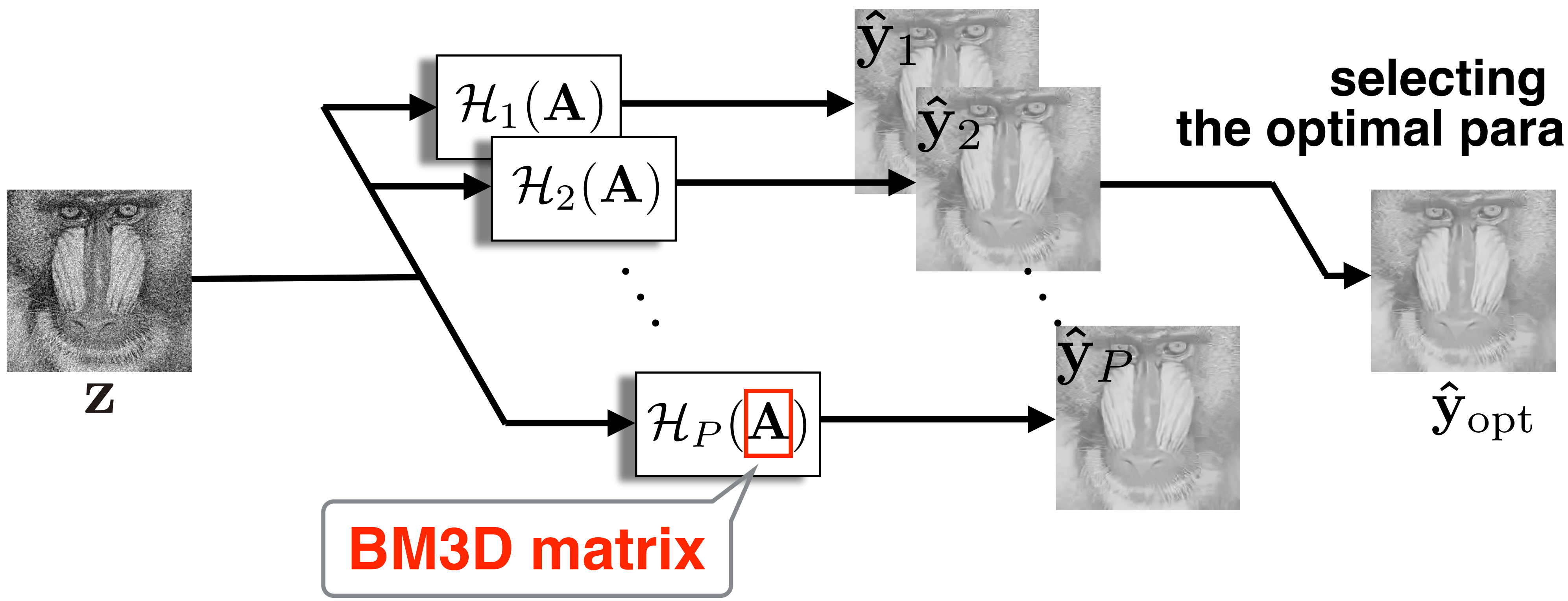
$h(\cdot)$: Arbitrary function

Recurrence relation	$\mathcal{T}_k(\mathbf{W}) = 2\mathbf{W}\mathcal{T}_{k-1}(\mathbf{W}) - \mathcal{T}_{k-2}(\mathbf{W})$
Initial conditions	$\mathcal{T}_0(\mathbf{W}) = \mathbf{I},\ \mathcal{T}_1(\mathbf{W}) = \mathbf{W}$

Purpose of Proposed Method

Purpose Applying eigenvalue filtering to **state-of-the-art methods**

I.e.) BM3D [2]



Next topic

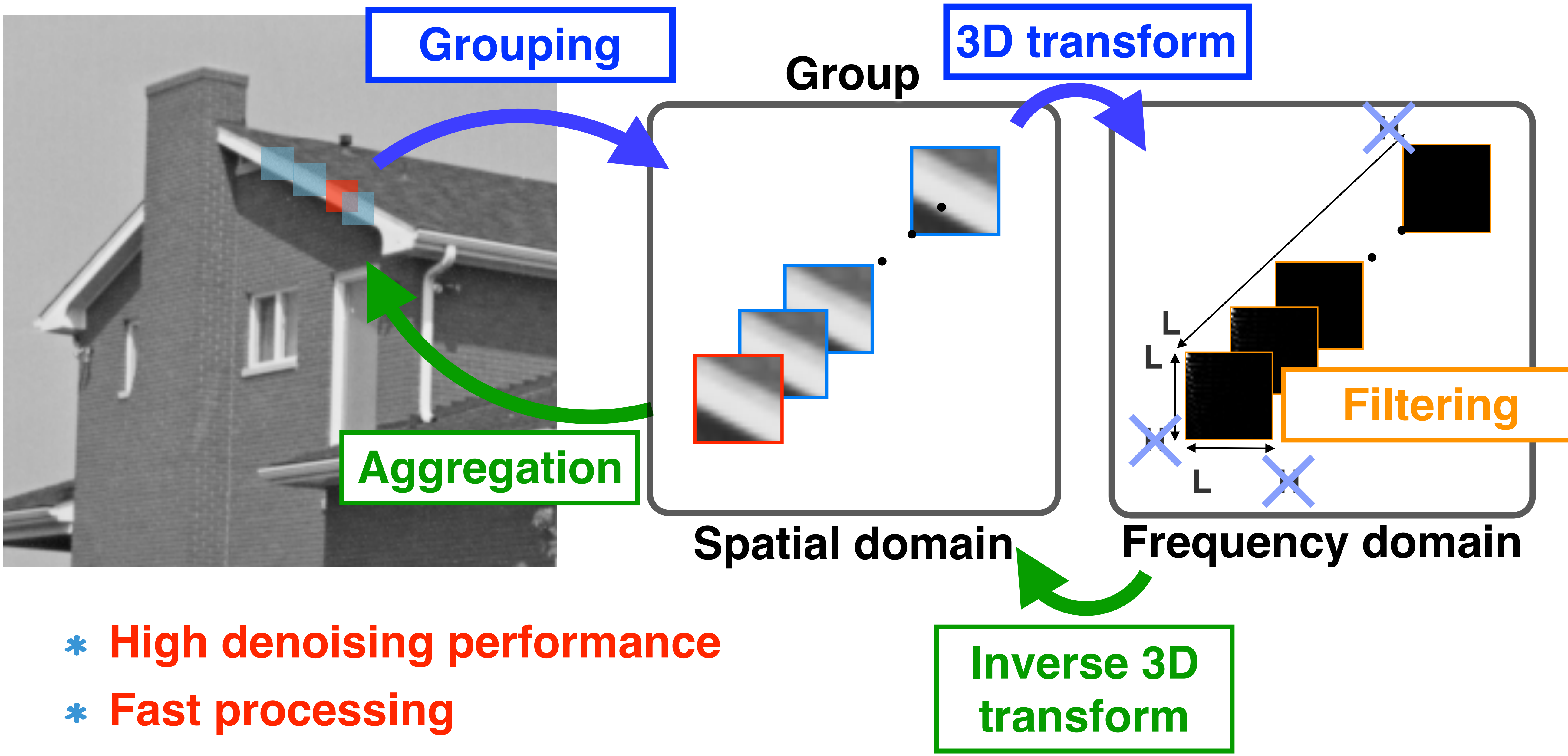
- BM3D algorithm and its matrix representation
- Problem of matrix construction
- Solution (Proposed method)

[2] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-D transform-domain collaborative filtering", *IEEE Trans. Image Process.*, vol. 16, no. 8, pp. 2080–2095, Aug. 2007.

BM3D Algorithm

Block Matching and 3D Filtering (BM3D):

Redundant filtering using similarity among blocks



- **High denoising performance**
- **Fast processing**

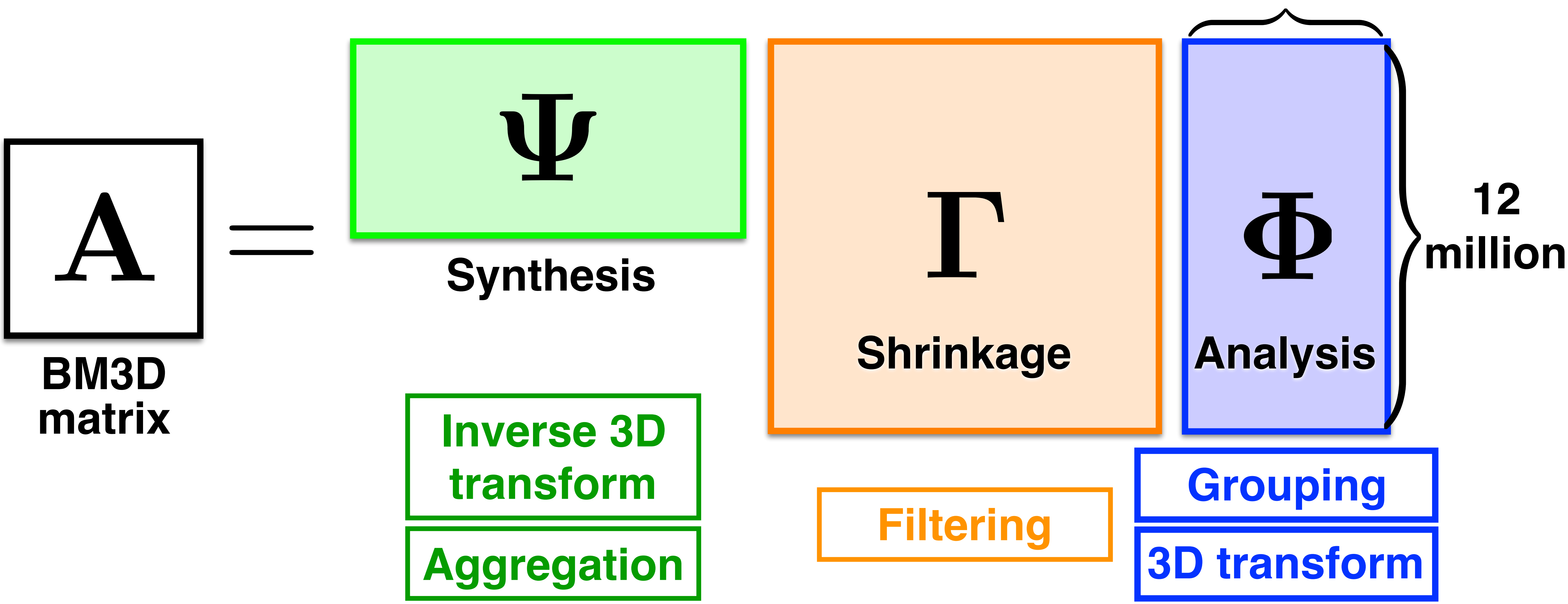
Matrix Construction and its problem

- **BM3D is expressed as a filter matrix**

$$\hat{\mathbf{y}} = \mathcal{F}_{\mathrm{BM3D}}(\mathbf{z}) = \mathbf{\Psi\Gamma\Phi z} = \mathbf{Az}$$

- **Construction of $\mathbf{\Phi}$ and $\mathbf{\Psi}$ needs much computational cost**

Ex.) Image with 1024×1024 pixels



→ **It is hard to construct the BM3D matrix**

Proposed Method

Restored image using eigenvalue filtering by CPA

$$\hat{\mathbf{y}}_p = \mathcal{H}_p(\mathbf{A})\mathbf{z} = \left(\frac{1}{2}c_0\mathbf{I} + \sum_{k=1}^{d} c_k \mathcal{T}_k(\mathbf{A})\right)\mathbf{z} = \frac{1}{2}c_0\mathbf{z} + \sum_{k=1}^{d} c_k \mathcal{T}_k(\mathbf{A})\mathbf{z}$$

Previous method

$$\mathcal{T}_k(\mathbf{A})\mathbf{z} = 2\mathbf{A}\mathcal{T}_{k-1}(\mathbf{A})\mathbf{z} - \mathcal{T}_{k-2}(\mathbf{A})\mathbf{z}$$

$$\mathcal{T}_0(\mathbf{A})\mathbf{z} = \mathbf{z} \ , \qquad \mathcal{T}_1(\mathbf{A})\mathbf{z} = \mathbf{A}\mathbf{z}$$

$$\mathcal{B}_k(\mathbf{z}) = \mathcal{T}_k(\mathbf{A})\mathbf{z}$$

Replacing

$$\mathcal{F}_{\mathrm{BM3D}}(\mathbf{z}) = \mathbf{A}\mathbf{z}$$

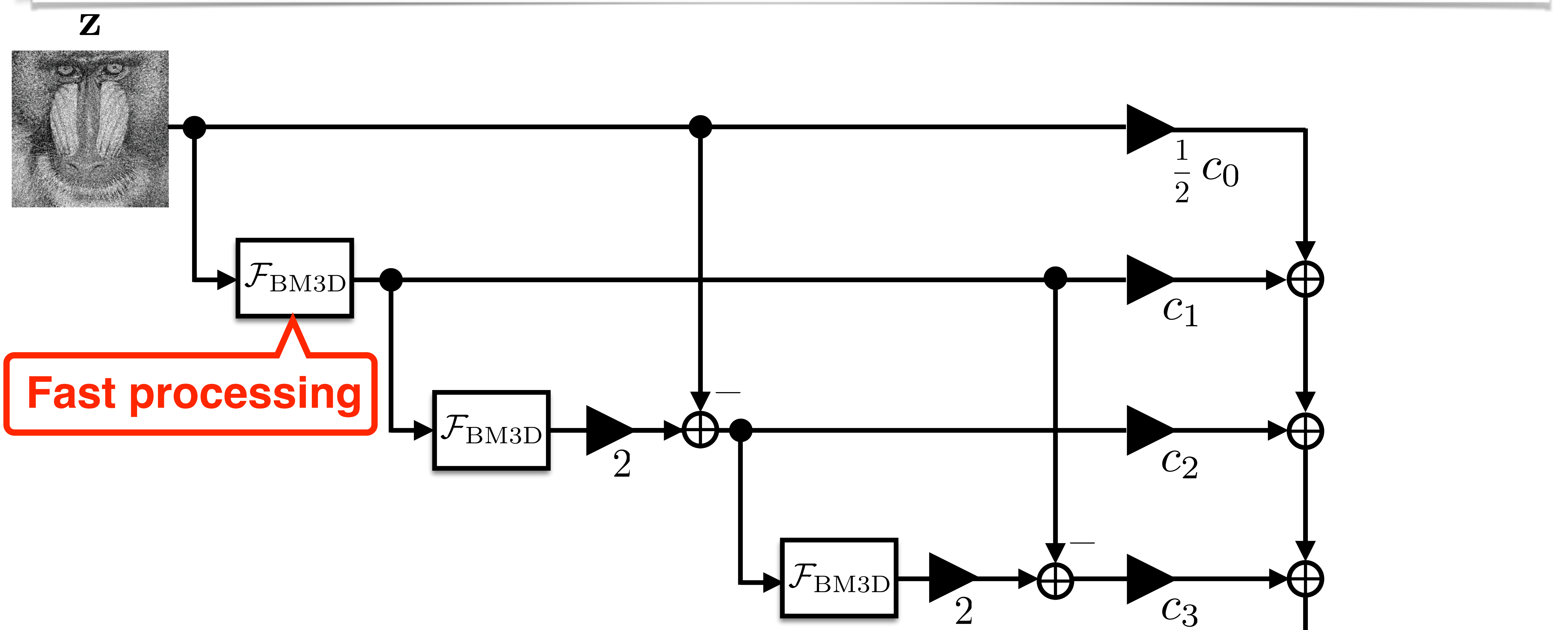
Proposed method

$$\mathcal{T}_k(\mathbf{A})\mathbf{z} \simeq \mathcal{B}_k(\mathbf{z}) = 2\mathcal{F}_{\mathrm{BM3D}}(\mathcal{B}_{k-1}(\mathbf{z})) - \mathcal{B}_{k-2}(\mathbf{z})$$

$$\mathcal{T}_0(\mathbf{A})\mathbf{z} = \mathcal{B}_0(\mathbf{z}) = \mathbf{z} \ , \quad \mathcal{T}_1(\mathbf{A})\mathbf{z} = \mathcal{B}_1(\mathbf{z}) = \mathcal{F}_{\mathrm{BM3D}}(\mathbf{z})$$

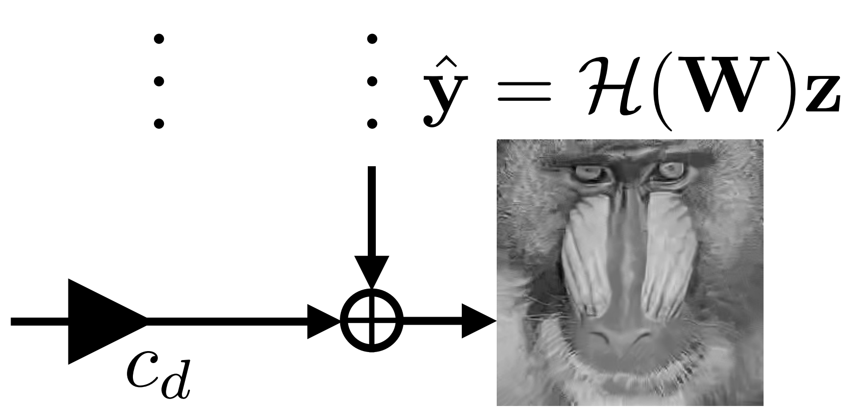
→ **Matrix construction is not required**

Fast Eigenvalue Filtering



Restored image using CPA

$$\hat{\mathbf{y}} = \mathcal{H}(\mathbf{W})\mathbf{z} = \frac{1}{2}c_0\mathbf{z} + \sum_{k=1}^{d} c_k \mathcal{T}_k(\mathbf{W})\mathbf{z}$$



Eigenvalue filtering is realized only by using BM3D operators and Chebyshev coefficients

Eigenvalue distribution on each step

Problem：Input-dependency of the BM3D

CPA: $\mathbf{A}$ **must be fixed regardless of the degree of polynomials**

$$\mathcal{T}_k(\mathbf{A})\mathbf{z} = 2\mathbf{A}\mathcal{T}_{k-1}(\mathbf{A})\mathbf{z} - \mathcal{T}_{k-2}(\mathbf{A})\mathbf{z}$$

Needs verification

BM3D: $\mathcal{F}_{\mathrm{BM3D}}$ **is adaptive to the input image**

$$\mathcal{T}_k(\mathbf{A})\mathbf{z} = 2\mathcal{F}_{\mathrm{BM3D}}(\mathcal{T}_{k-1}(\mathbf{A})\mathbf{z}) - \mathcal{T}_{k-2}(\mathbf{A})\mathbf{z}$$

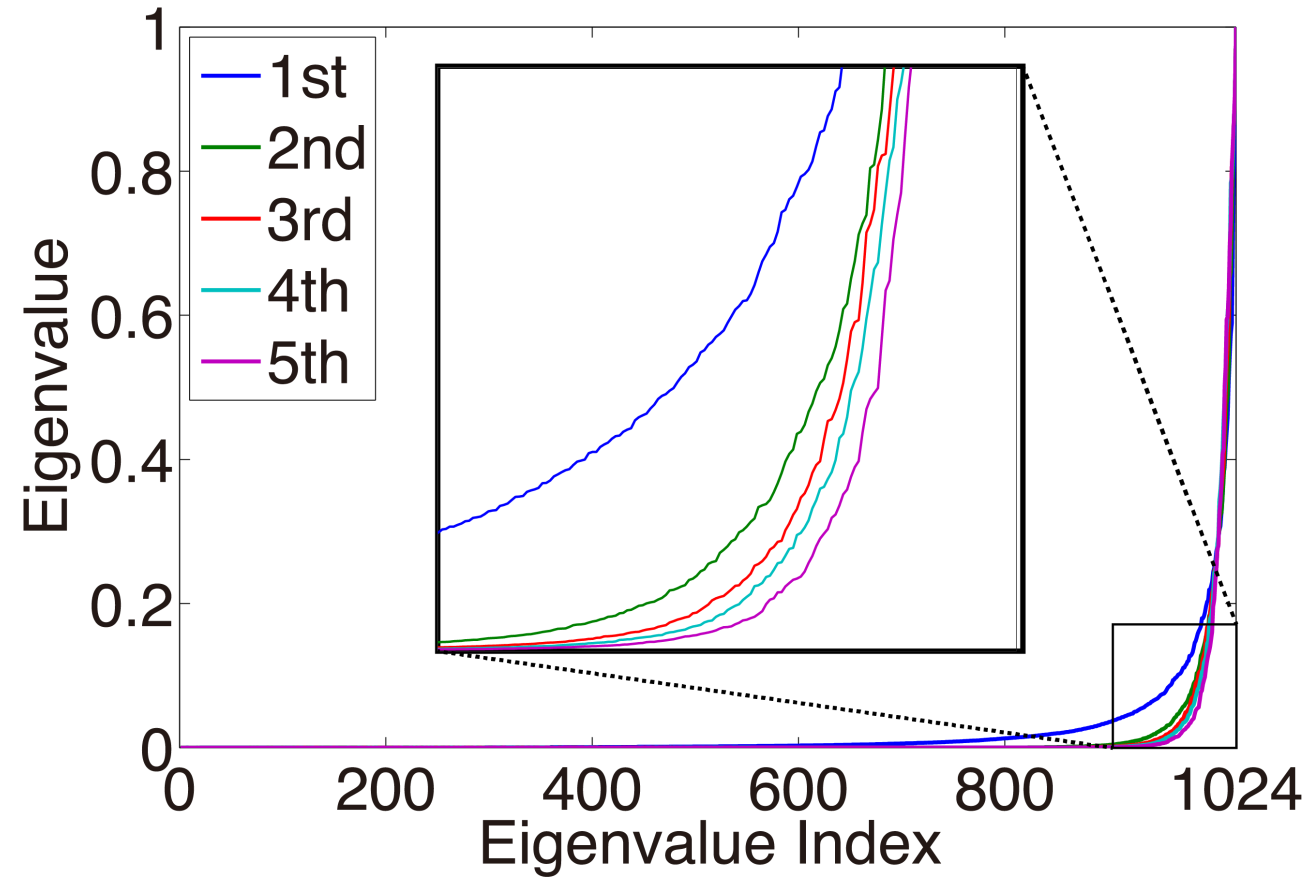
Due to Block matching and filter coefficients

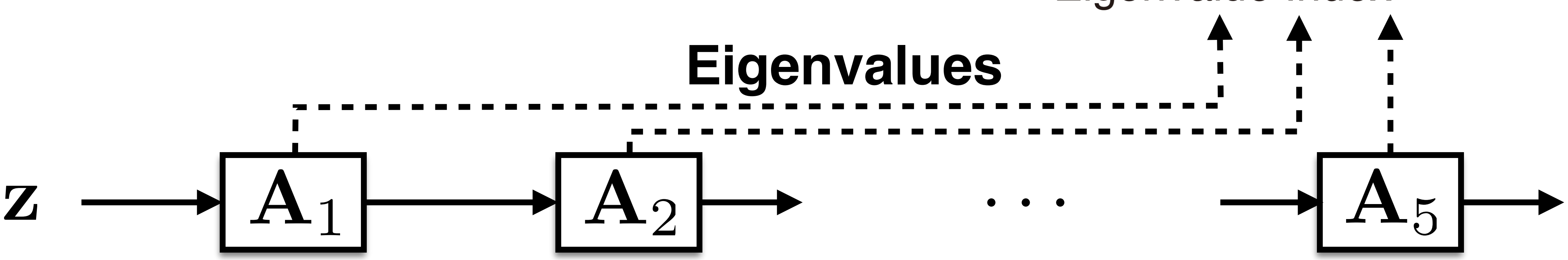
Verification Experiment

- **Verify eigenvalue distributions according to iteration numbers**

Test sub-image $\mathbf{z}$

Mandrill (32×32)

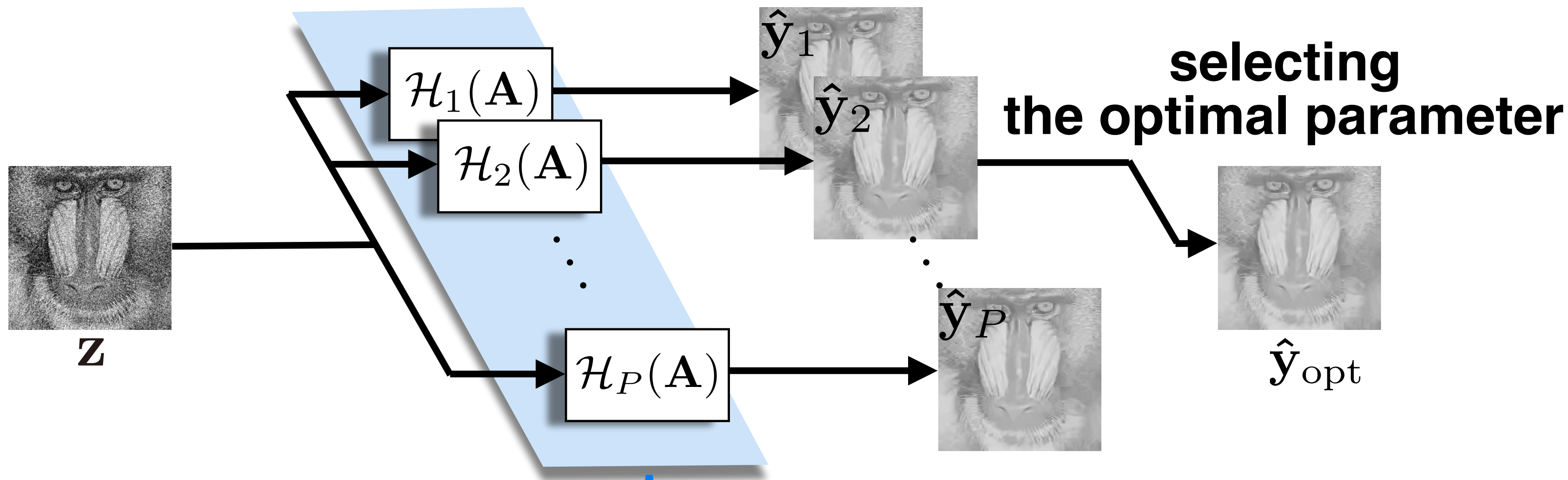




$\mathbf{A}_n$: n th matrix of BM3D given by $\mathcal{T}_{n-1}(\mathbf{A})\mathbf{z}$

- **Eigenvalue distributions could be assumed to be consistent regardless of the iteration number**

Summary of Proposed Method



Eigenvalue filtering by CPA

$$\mathcal{H}(\mathbf{A}) = \frac{1}{2}c_0\mathbf{I} + \sum_{k=1}^{d} c_k \mathcal{T}_k(\mathbf{A})$$

$$\mathcal{F}_{\mathrm{BM3D}}(\mathbf{z}) = \mathbf{A}\mathbf{z}$$

Previous method

$$\mathcal{T}_k(\mathbf{A}) = 2\mathbf{A}\mathcal{T}_{k-1}(\mathbf{A}) - \mathcal{T}_{k-2}(\mathbf{A})$$
$$\mathcal{T}_0(\mathbf{A}) = \mathbf{I}\ , \quad \mathcal{T}_1(\mathbf{A}) = \mathbf{A}$$

Proposed method

$$\mathcal{B}_k(\mathbf{z}) = 2\mathcal{F}_{\mathrm{BM3D}}(\mathcal{B}_{k-1}(\mathbf{z})) - \mathcal{B}_{k-2}(\mathbf{z})$$
$$\mathcal{B}_0(\mathbf{z}) = \mathbf{z}\ , \quad \mathcal{B}_1(\mathbf{z}) = \mathcal{F}_{\mathrm{BM3D}}(\mathbf{z})$$

Experiment

Denoising performance assessment

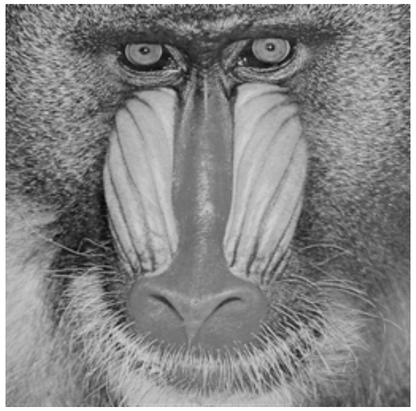
- **Comparison**: BM3D, Global Image Denoising(GLIDE) [3]
  - GLIDE : Improving method by eigenvalue filtering
- **Test images**: *Bridge*, *Mandrill*, *Goldhill*, *Building*
- **Noise strength**: $\sigma \in \{10, 20, 30, 40, 50\}$
- **Measure**: PSNR, SSIM

Conditions

- Intel Xeon E5-2690 2.9GHz CPU
- 62.9 GB RAM
- 12 core parallel computing

Bridge

Mandrill



Goldhill

Building

[3] H. Talebi and P. Milanfar, “Global image denoising,” *IEEE Trans. Image Process.*, vol. 23, no. 2, pp. 755–768, Feb. 2014.

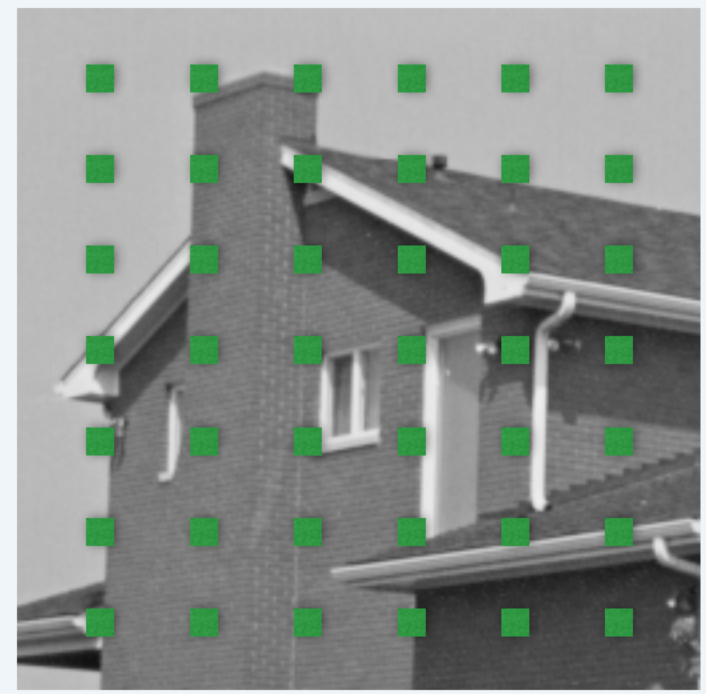
Experiment

Global Image Denoising (GLIDE)

estimate eigenvalue/eigenvector from a portion of a pre-filtered image

Pre-filtering

Sampling



Advantage **Fast processing**

Disadvantage **Eigenvalue filtering can not be performed exactly**

[3] H. Talebi and P. Milanfar, "Global image denoising," *IEEE Trans. Image Process.*, vol. 23, no. 2, pp. 755–768, Feb. 2014.

Experiment

Denoising performance assessment

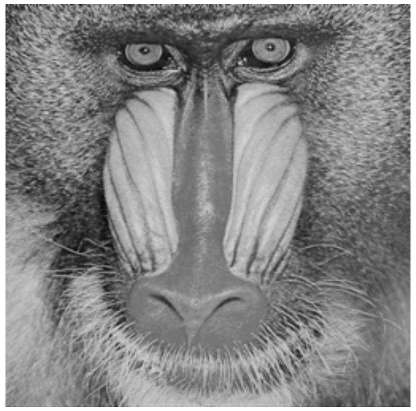
- **Comparison** — BM3D, Global Image Denoising(GLIDE) [3]
  - GLIDE : Improving method by eigenvalue filtering
- **Test images** — *Bridge*, *Mandrill*, *Goldhill*, *Building*
- **Noise strength** — $\sigma \in \{10, 20, 30, 40, 50\}$
- **Measure** — PSNR, SSIM

Conditions

- Intel Xeon E5-2690 2.9GHz CPU
- 62.9 GB RAM
- 12 core parallel computing

Bridge

Mandrill



Goldhill

Building

[3] H. Talebi and P. Milanfar, "Global image denoising," *IEEE Trans. Image Process.*, vol. 23, no. 2, pp. 755–768, Feb. 2014.

Performance Comparison

σ	Method	*Bridge*	*Mandrill*	*Goldhill*	*Building*
10	BM3D	29.84 / 0.911	30.56 / 0.905	31.80 / 0.880	**33.16** / **0.939**
	GLIDE	29.81 / **0.913**	30.54 / 0.904	31.72 / 0.881	32.91 / 0.938
	Proposed	**29.86** / **0.913**	**30.57** / **0.906**	**31.86** / **0.884**	**33.16** / **0.939**
20	BM3D	25.46 / 0.765	26.39 / 0.773	28.50 / 0.775	29.35 / 0.862
	GLIDE	25.62 / 0.784	26.55 / 0.788	28.57 / **0.785**	29.30 / 0.865
	Proposed	**24.66** / **0.789**	**26.56** / **0.791**	**28.59** / 0.784	**29.40** / **0.866**
30	BM3D	23.55 / 0.647	24.33 / 0.651	26.91 / 0.706	27.32 / 0.790
	GLIDE	23.68 / 0.678	24.57 / 0.686	26.71 / 0.711	27.26 / 0.792
	Proposed	**23.73** / **0.679**	**24.58** / **0.689**	**26.96** / **0.714**	**27.37** / **0.794**
40	BM3D	22.51 / 0.572	23.10 / 0.558	**25.84** / 0.654	25.89 / 0.722
	GLIDE	22.43 / 0.584	**23.23** / 0.573	25.70 / 0.640	25.87 / **0.729**
	Proposed	**22.55** / **0.586**	23.19 / **0.582**	25.83 / **0.655**	**25.90** / 0.724
50	BM3D	21.81 / 0.509	22.43 / 0.489	**25.04** / 0.610	24.93 / 0.663
	GLIDE	21.81 / **0.547**	**22.60** / 0.518	25.01 / **0.616**	24.85 / **0.680**
	Proposed	**21.93** / 0.540	22.59 / **0.525**	**25.04** / 0.615	**24.95** / 0.673

PSNR[dB] / SSIM

Original image

GLIDE

PSNR 22.43[dB]

SSIM 0.584

BM3D

PSNR 22.53[dB]

SSIM 0.571

Proposed

PSNR 22.71[dB]

SSIM 0.604

Bridge $\sigma = 40$

Visual Assessment

Original image

BM3D

BM3D	GLIDE	Proposed
22.53[dB] / 0.571	22.43[dB] / 0.584	22.71[dB] / 0.604

Visual Assessment

Original image

GLIDE

BM3D	GLIDE	Proposed
22.53[dB] / 0.571	22.43[dB] / 0.584	**22.71**[dB] / **0.604**

Visual Assessment

Original image

Proposed

BM3D	GLIDE	Proposed
22.53[dB] / 0.571	22.43[dB] / 0.584	22.71[dB] / 0.604

Execution Time

Image size	BM3D	GLIDE	Proposed
256x256	0.8	115.4	51.8
512x512	3.1	Out of Memory	225.1
1024x1024	18.1	Out of Memory	946.4

[sec]

* Faster than GLIDE
* Can be executed in commodity computers

Conditions

Intel Xeon E5-2690 2.9GHz CPU
62.9 GB RAM
12 core parallel computing

Conclusion

- **Purpose**

  Improvement of denoising performance for BM3D

- **Method**

  Eigenvalue filtering by CPA without matrix construction

- **Result**

  Better denoising performance visually and numerically

  Faster execution than GLIDE

- **Future work**

  Improvement of MSE estimation

Reference List

- **Eigenvalue filtering using CPA**

  M. Onuki, S. Ono, K. Shirai, and Y. Tanaka, "Non-local/local image filters using fast eigenvalue filtering," in *Proc. ICIP*, 2015.

- **BM3D**

  K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-D transform-domain collaborative filtering", *IEEE Trans. Image Process.*, vol. 16, no. 8, pp. 2080–2095, Aug. 2007.

- **Global image denoising**

  H. Talebi and P. Milanfar, "Global image denoising," *IEEE Trans. Image Process.*, vol. 23, no. 2, pp. 755–768, Feb. 2014.