Image Clustering with Noise

Christopher Qian and Hyungjoo Seo

University of Illinois at Urbana-Champaign

Why Cluster Images?

- Save space
  - Instead of storing $N \cdot M \cdot 3$ values, store $N \cdot M$ clusters and $K \cdot 3$ cluster values
- Image segmentation
  - Break image into smaller, more interpretable components
- Artistic reasons

K = 3 K = 7 K = 15

Clustering with Soft Labels

- After K-means, for each $\boldsymbol{x_i}$, we compute responsibilities:

$$r_{ik} = \frac{\pi_k \exp(-\frac{1}{\sigma^2}||x_i - \mu_k||_2^2)}{\sum_{j=1}^{K} \pi_j \exp(-\frac{1}{\sigma^2}\left||x_i - \mu_j|\right|_2^2)}$$

- $\pi_k = \frac{1}{n}\sum_{i=1}^{n} I(z_i = k)$
- We then replace x_i with $\sum_{j=1}^{K} r_{ij}\mu_j$.
- σ^2 is a tuning parameter. (We do not just use estimated σ^2 from GMM)

K = 2

K = 3

K = 7

Adding Noise

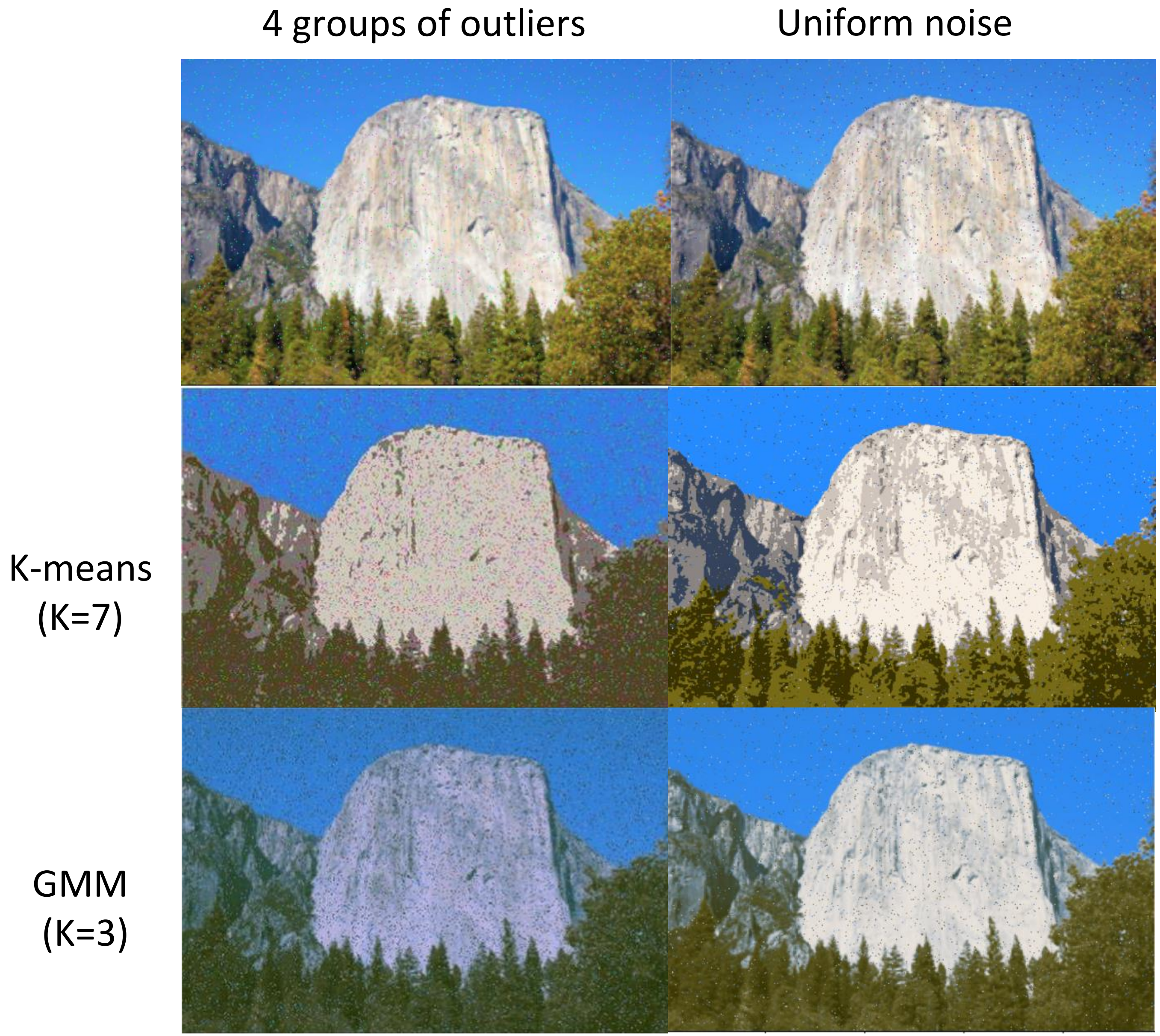
- We would like to be able to cluster noisy images well (accomplish one of our three goals in the beginning)
- We would want a similar output to if there was no noise at all
  - Ideally, we also want to remove the noise!
- Our approaches will incorporate spatial information of the pixels

Uniform

Pink

Multiple Colors

- Without even considering removing the noise, the clustering algorithms are affected in different ways
- K-means can be adversely affected by outlier noise, but uniform noise doesn't change much
- We mainly focus on uniform noise



Approach 1:
Pre-process to remove noise

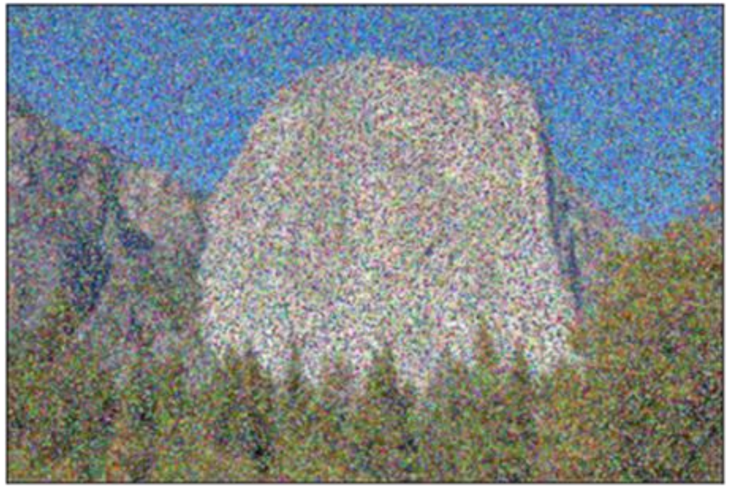
- N out of P pixels are corrupted:
  => Noise-to-signal ratio =: N/P (%)
- White uniform noise shown

Original

Noisy ratio: 3.0%

Noisy ratio: 30.0%

Noisy ratio: 75.0%



Approach 1:
Pre-process to remove noise (3%)

Noisy ratio: 3.0%

Front-end Filtering

Mean Filter

Median Filter

Gauss Filter

2D Conv Filt

Bilateral Filt

Approach 1:
Pre-process to remove noise (30%)

Noisy ratio: 30.0%

Front-end
Filtering

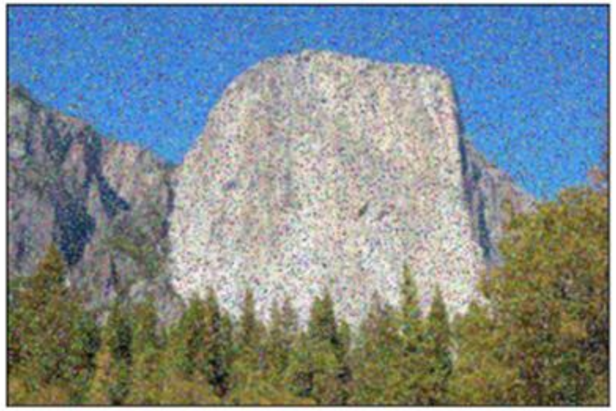
Mean Filter

Median Filter

Gauss Filter

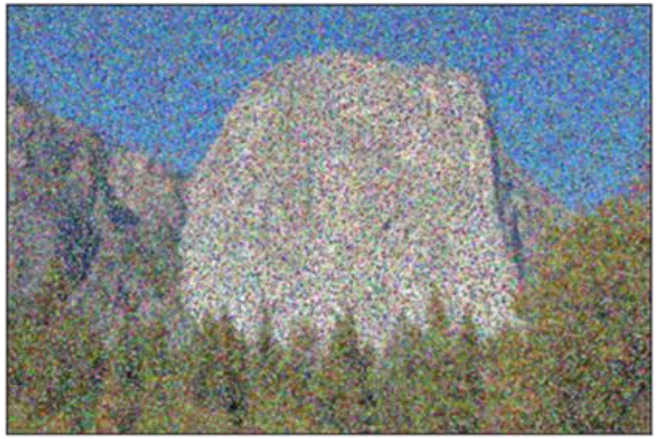
2D Conv Filt

Bilateral Filt



Approach 1:
Pre-process to remove noise (75%)

Noisy ratio: 75.0%



Front-end
Filtering

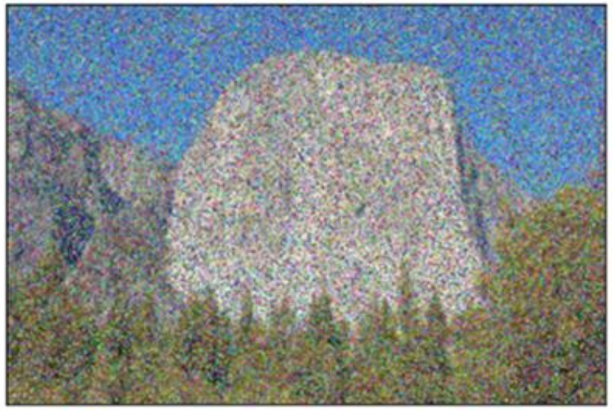
Mean Filter

Median Filter

Gauss Filter

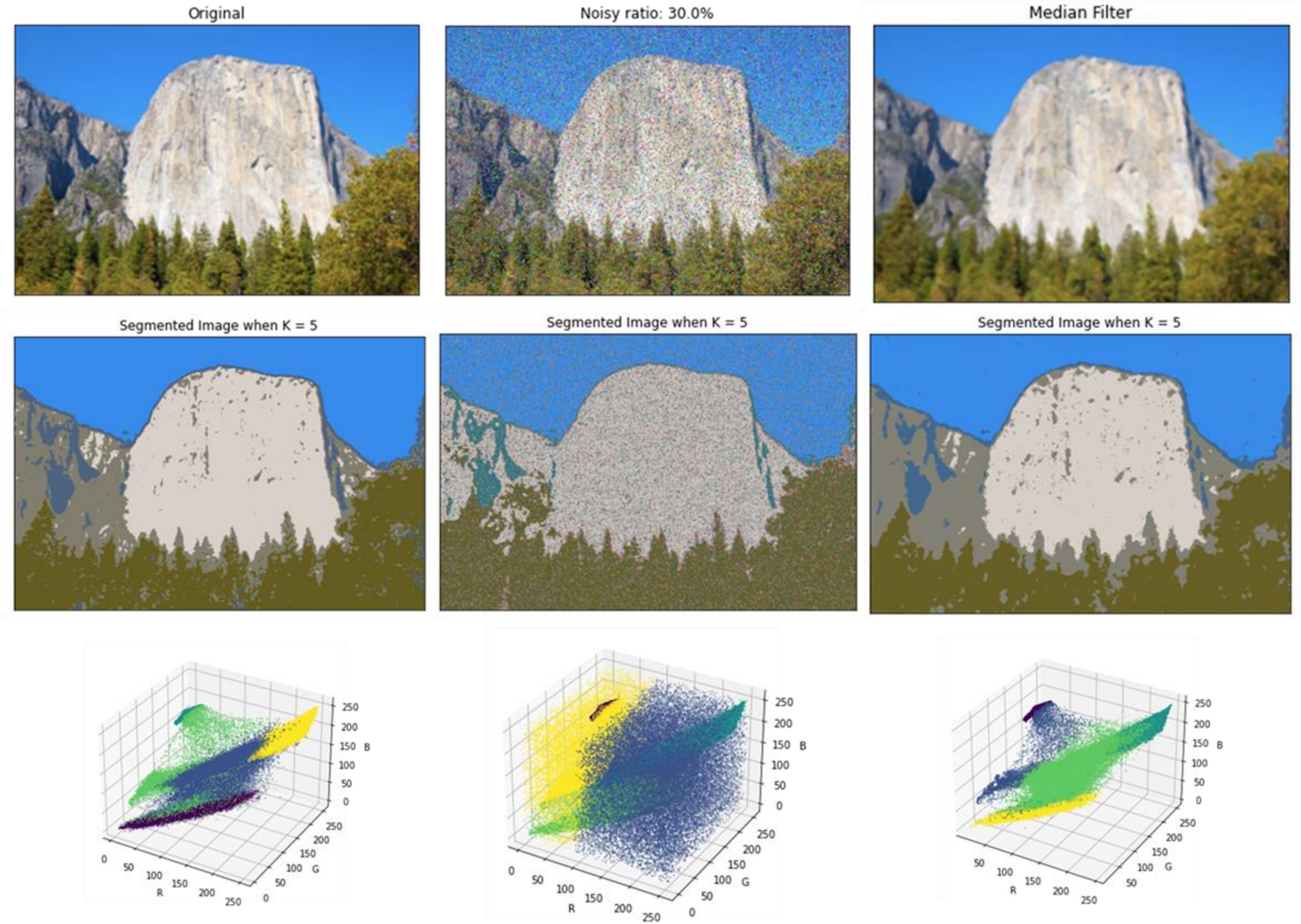
2D Conv Filt

Bilateral Filt



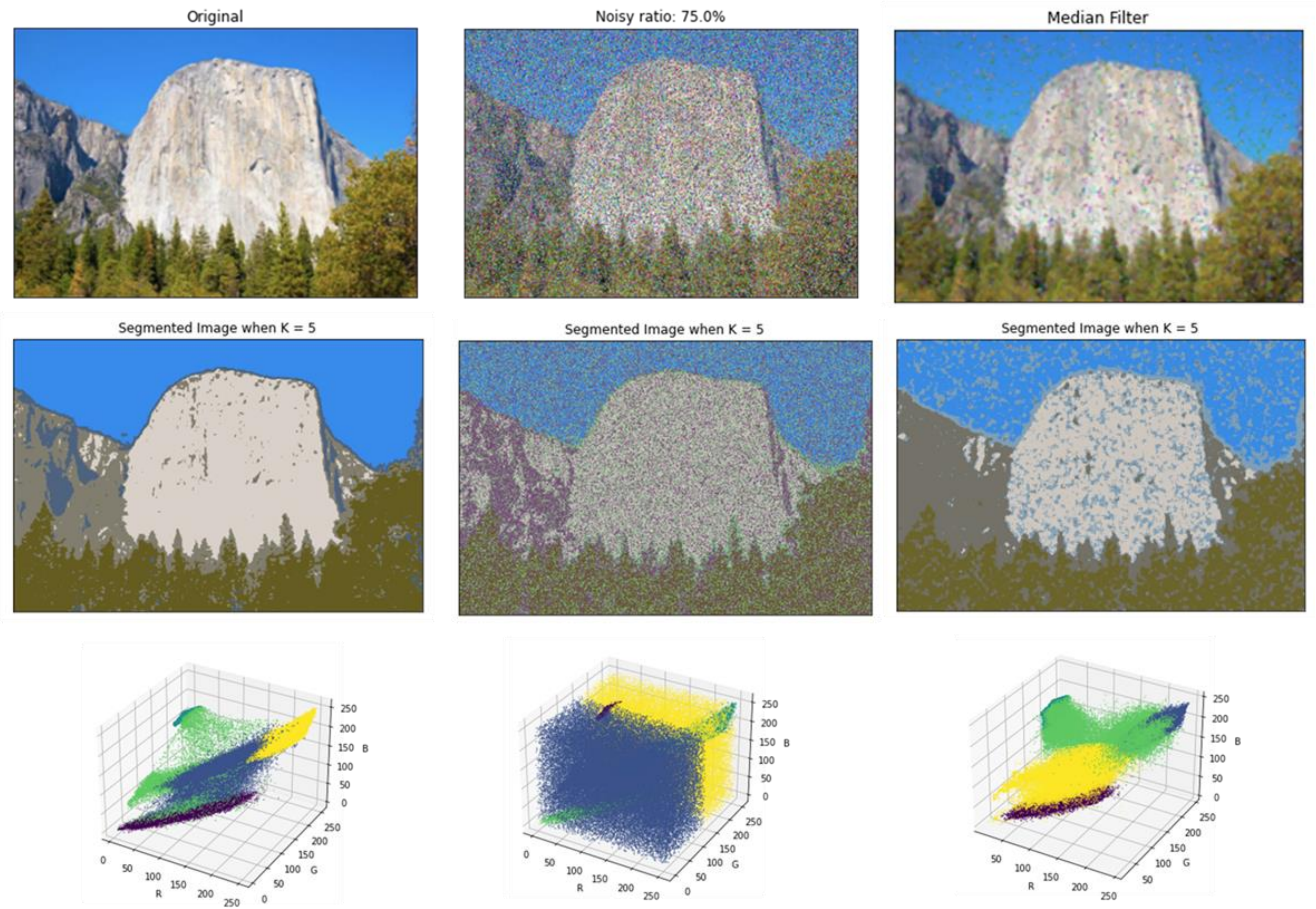
Approach 1: Pre-processing

- 30% ratio case
- Perform GMM clustering (K = 5)
- Almost recovered original result !



Approach 1: Pre-processing

- 75% ratio case
- Perform GMM clustering (K = 5)
- **Further processing is needed.**



Approach 2: Postprocess to remove noise

- Any idea used to remove noise in preprocessing can be applied here
- After performing clustering, we have estimated cluster means. These can be used in postprocessing
- Intuitively, pixels should be likely to have similar cluster assignments to neighboring pixels. We assume the following distribution on the latent labels:

$$\log p(z|x) \propto \log p(x|z) + \log p(z)$$
$$= -\sum_{i=1}^{n} \left\|x_i - \mu_{z_i}\right\|_2^2 - \sum_{(s,t)} w_{st} z_s z_t$$

- Assume 2 clusters, $z_i \in \{-1,1\}$.
- $w_{st} = 0$ if pixels are not neighbors.
- Then, for each z_i, integrate out the other variables to update the responsibility

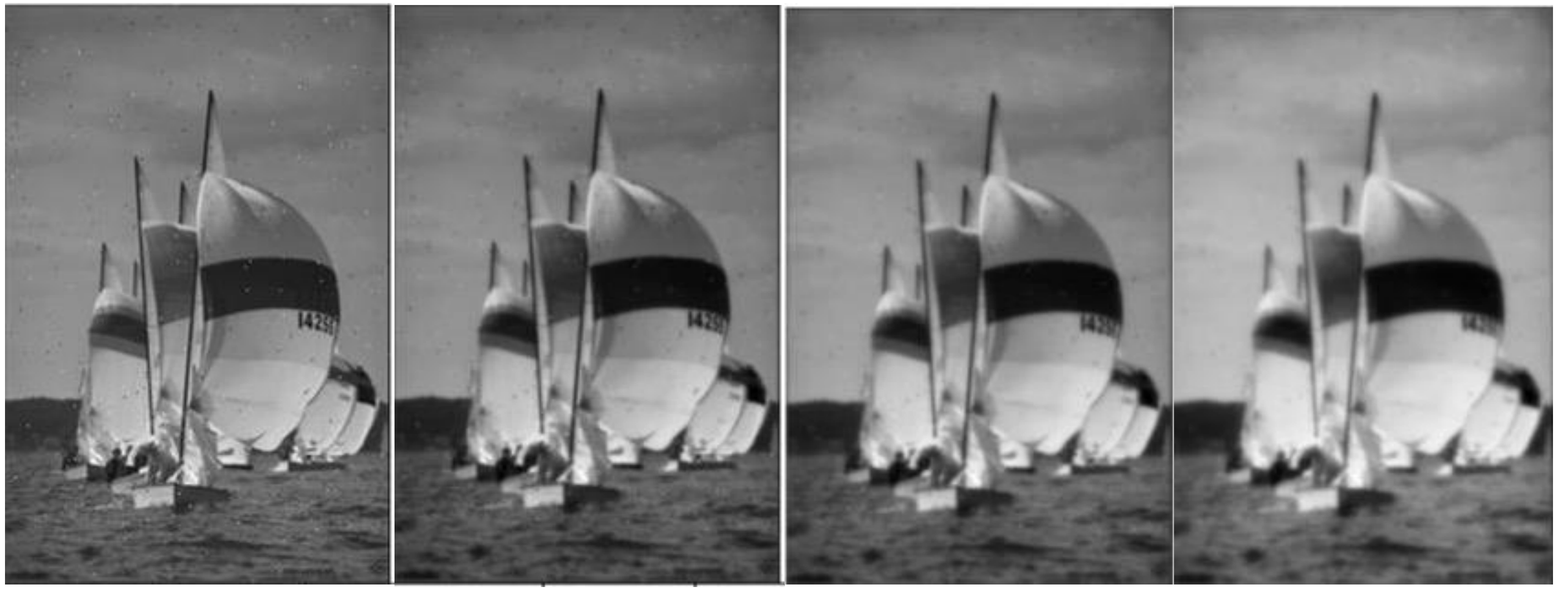
- Even evaluating the probability is intractable, so we use a mean-field approximation
- This leads to the following set of updates:

Repeat Until Convergence

For $i = 1{:}n$

For $k = 1{:}K$

Set $r_{ik} \propto \exp(\sum_{j \in nbr(i)} w_{ij} r_{jk} + \left|\left|x_i - \mu_j\right|\right|_2^2)$

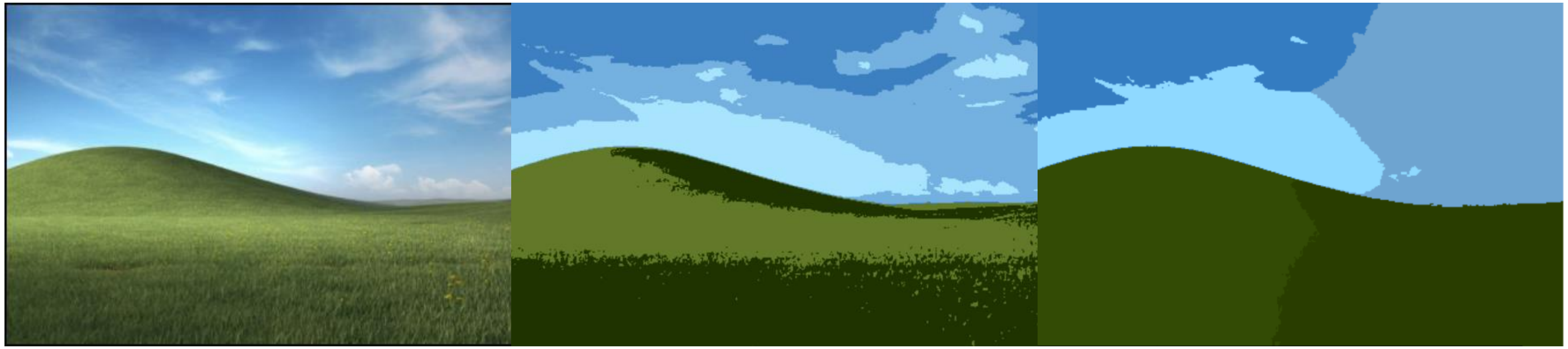


- Results aren't the greatest for El Capitan example
- If w's are set to be very large, then produces interesting results.
- When w's are large, the probability assignments are almost hard assignments, we can save space by discarding the probabilities and keeping the highest probability assignment

K = 3 K = 7 K = 15

Approach 3: Joint Clustering and Denoising

- One way to incorporate spatial information is to cluster in RGB-XY space (we append the coordinates to each data point).
- Tuning parameter in how much to weigh XY coordinates
- The cluster means will be 5-dimensional. When we replace each data point with its cluster mean, we only replace the RGB part, not the position part.
- This is a naïve approach, but produces interesting results

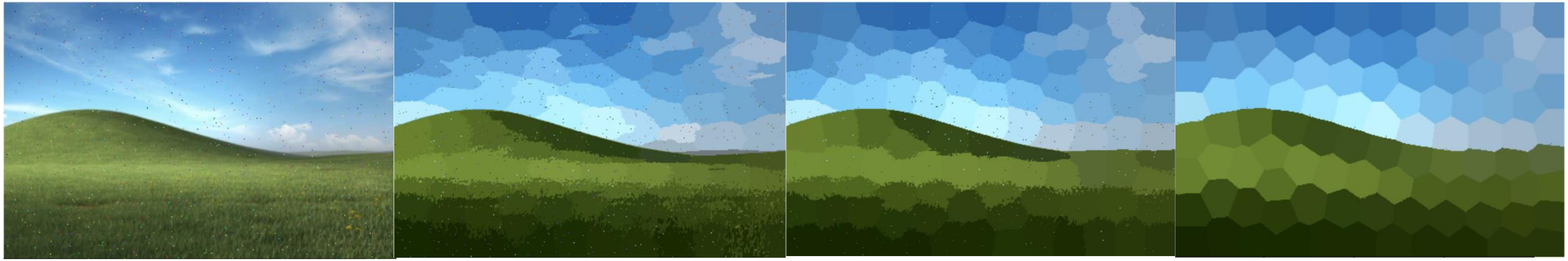


K-means

K-means + XY

K = 5

- Increasing the effect of position results in pronounced honeycomb tiling effect
- Does not work well with small K
- Increasing K too much causes it to tile noise as well



$\gamma = 240$ $\gamma = 120$ $\gamma = 20$

Graph-based Approach (Intro)

Graph Laplacian — Adjacency matrix

$$\mathbf{L} = \mathbf{D} - \mathbf{W}$$

$$D = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

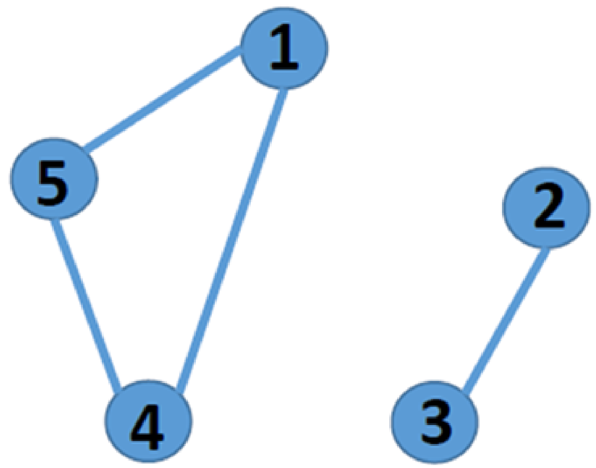
$$W = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

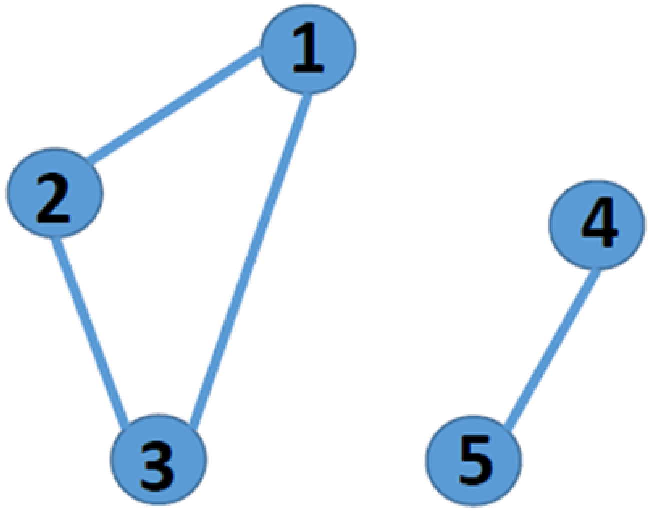
$$L = \begin{pmatrix} 1 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 1 \end{pmatrix}$$

Graph-based Approach (Intro-2)

Graph Laplacian   Adjacency matrix

L = D - W





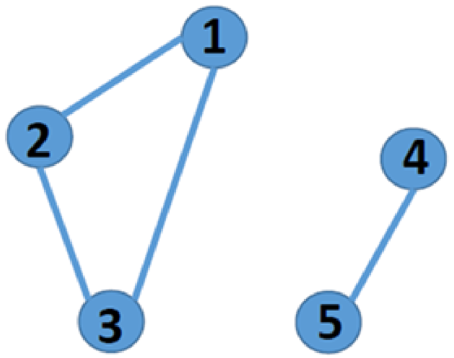
$$L = \begin{pmatrix} 2 & 0 & 0 & -1 & -1 \\ 0 & 1 & -1 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ -1 & 0 & 0 & 2 & -1 \\ -1 & 0 & 0 & -1 & 2 \end{pmatrix}$$

$$L = \begin{pmatrix} 2 & -1 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ -1 & -1 & 2 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & -1 & 1 \end{pmatrix}$$

Graph-based Approach (Intro-3)

Graph Laplacian Adjacency matrix

L = D - W



$$L = \begin{pmatrix} 2 & -1 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ -1 & -1 & 2 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & -1 & 1 \end{pmatrix}$$

$$\mathbf{L}\begin{pmatrix} \mathbf{1} \\ \mathbf{1} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix} = \mathbf{0} \quad \textbf{and} \quad \mathbf{L}\begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{1} \end{pmatrix} = \mathbf{0}$$

- **For k-clusters, k eigenvectors have low eigenvalues.**

$$\mathbf{U=} \begin{pmatrix} \mathbf{1} \\ \mathbf{1} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{1} \end{pmatrix}$$

Graph-based Approach (Proposed)

- Incorporate both spatial(x,y) and color(r,g,b) info in W

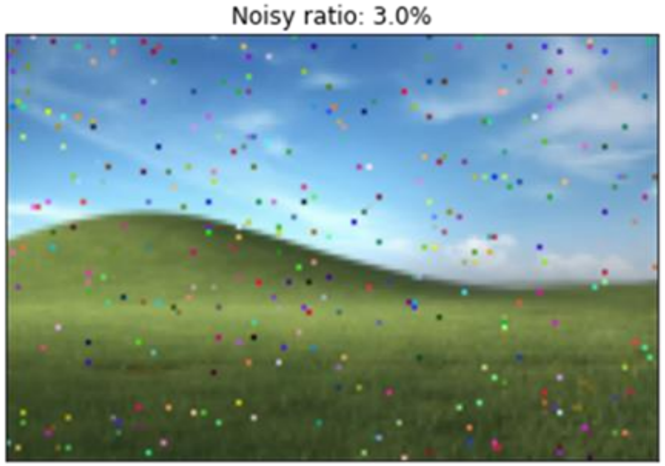
$$W_{ij} = \exp\left(-\gamma_s\left|(x,y)_i - (x,y)_j\right|^2 - \gamma_c\left|(r,g,b)_i - (r,g,b)_j\right|^2\right)$$

- Hyperparameter: γ_s, γ_c **controls each contribution**
- Nonlinear Radial basis function kernel (Gram matrix)

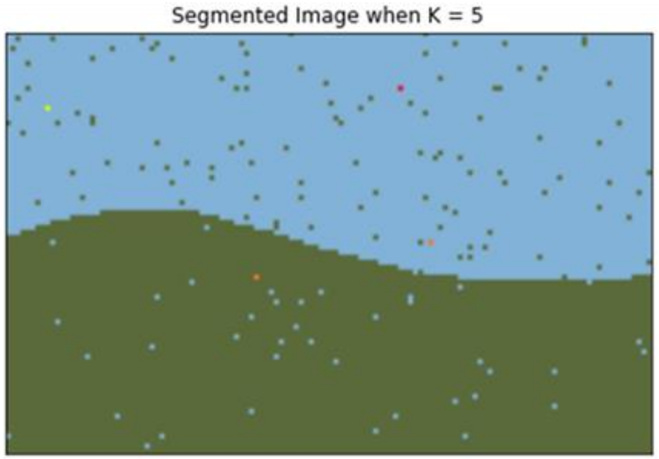
$\Rightarrow$ a mapping: (x, y, r, g, b) -> higher-dimensional space

- Signal: Both Spatial continuity and color continuity
- Noise: Either or neither
  For better separation of signal and noise

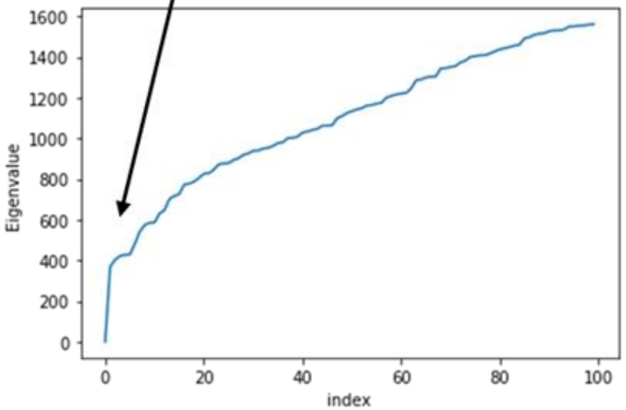
Graph-based Approach (Results)



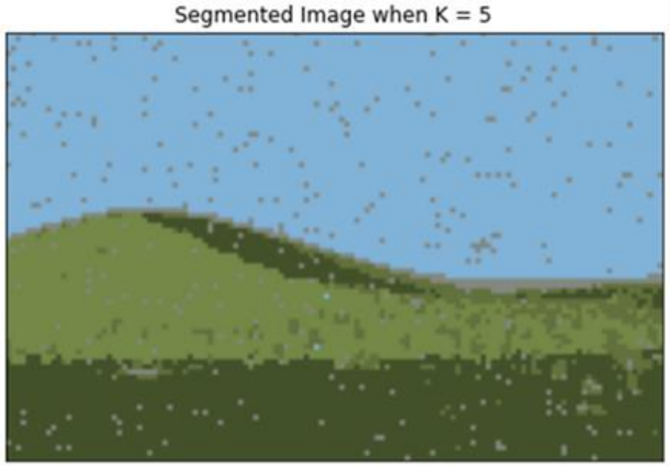
Proposed Spectral Clustering (K=5)

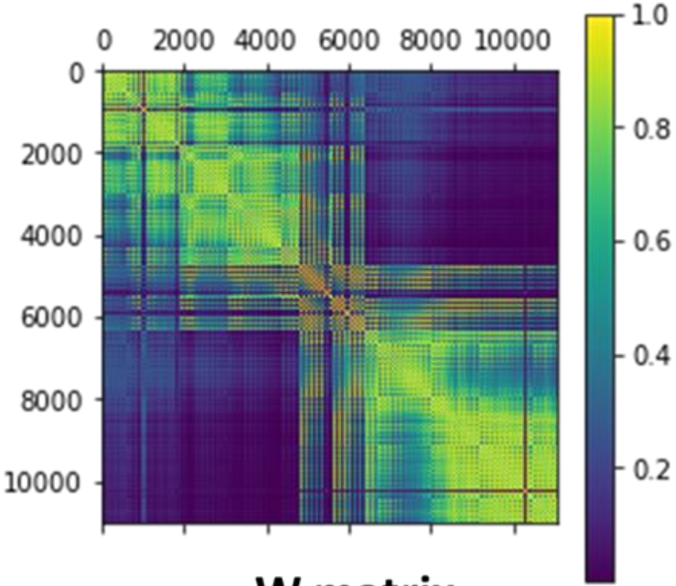


Flat @ k=1,2

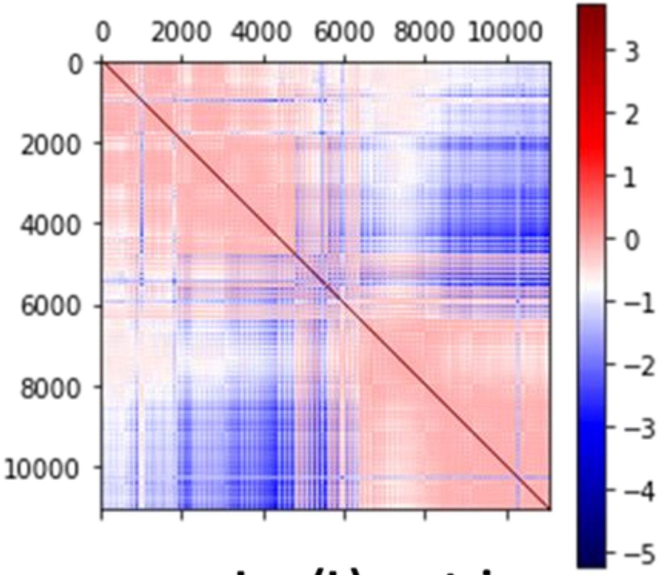


GMM(K=5)





W matrix



Log(L) matrix

What’s next?

- Pre- & post- processing
- For post-processing, try medium operation in iteration step
- Impact from other types of noise (Gaussian, colored, …)
- Kernel K-means vs. Spectral Clustering ?
- Graph-based pre-processing?