Computer Vision: Fall 2022 — Lecture 5

Dr. Karthik Mohan

Univ. of Washington, Seattle

October 14, 2022

Check-In

Today

1. Computational Complexity of Algorithms
2. tSNE for Image Visualization
3. Embeddings
4. Total Variation (TV) methods for image smoothing

References

1. tSNE paper
2. Total Variation

Notion of Complexity for Algorithms

Interview Favorite

Almost any interview that involves coding (MLE, Data Science, SWE) - You will get this question from the interviewer. What's the overall complexity - time and space of an algorithm?

Notion of Complexity for Algorithms

Interview Favorite

Almost any interview that involves coding (MLE, Data Science, SWE) - You will get this question from the interviewer. What's the overall complexity - time and space of an algorithm?

1. Computational Complexity

In terms of the data dimensions, what's the order of time an algorithm takes to completion? Example - If you have to sum up N integers - What's the computational complexity?

Notion of Complexity for Algorithms

Interview Favorite

Almost any interview that involves coding (MLE, Data Science, SWE) - You will get this question from the interviewer. What's the overall complexity - time and space of an algorithm?

1. Computational Complexity

In terms of the data dimensions, what's the order of time an algorithm takes to completion? Example - If you have to sum up N integers - What's the computational complexity?

2. Space Complexity

What extra storage space do you need to compute your result or run your algorithm? Example - If you have to sum up N integers stored in a list - What's the space complexity?

Notion of Complexity for Algorithms

Which is better?

$O(1), O(N), O(N^2)$?

Notion of Complexity for Algorithms

Which is better?

$O(1), O(N), O(N^2)$?

Which is faster?

To sum up the diagonal entries of a matrix or to multiply all the elements in the same matrix?

Notion of Complexity for Algorithms

Which is better?

$O(1), O(N), O(N^2)$?

Which is faster?

To sum up the diagonal entries of a matrix or to multiply all the elements in the same matrix?

Which is faster?

If you time the answer to the previous question for $N = 2$ in Python - You may not notice any difference in the time taken. But make $N = 10k$ and suddenly you see that the $O()$ difference starts to show up. $O()$ means you are on the order of the stated complexity, but constants might be different.

Notion of Complexity for Algorithms

Dot Product/Inner Product of tSNE embeddings Complexity

Let's say you want to take the dot product of the embeddings of two images, I_1 and I_2. The images are in dimension $m \times n$ pixels. Let's say the embeddings are from tSNE and have a dimension of $N = 500$. What's the computational complexity of the dot product of the embeddings?

Notion of Complexity for Algorithms

Dot Product/Inner Product of tSNE embeddings Complexity

Let's say you want to take the dot product of the embeddings of two images, I_1 and I_2. The images are in dimension $m \times n$ pixels. Let's say the embeddings are from tSNE and have a dimension of $N = 500$. What's the computational complexity of the dot product of the embeddings?

Same Complexity!

Summing up N integers has the same computational complexity as a dot product of two tSNE embeddings of dimension N! Would the run time be exactly the same as well?

ICE #1

Computational Complexity of Matrix-Matrix Multiplication

Let's say you computed the SVD of X and got factors, U, Σ, V. You now store a reduced form as $\tilde{U} \in^{m \times k}, \tilde{\Sigma} \in^{k}, \tilde{V} \in^{k \times n}$. For the purpose of a projection operation, you need to compute $Z = \tilde{U}\tilde{V}$. What's the computational complexity of obtaining Z?

Hint: What's the complexity of multiplying $\tilde{U}$ with just the first column of $\tilde{V}$? Now multiply that with the number of columns in $\tilde{V}$ to get the answer!

1. $O(mnk)$
2. $O(mnk^2)$
3. $O(mn^2k)$
4. $O(m^2nk^2)$

ICE #2

Computational Complexity of a Convolution!

Let C be a convolution matrix of size $k \times k$. Let's say you have an input matrix, $I \in^{m \times n}$. Now you convolve I with C i.e. $Y = I * C$. What is the computational complexity of computing Y? Your answer should be in terms of m, n, k. The amazing thing about this question is that it didn't matter what C looks like - It could be a blur kernel, a sharpen kernel or a smoothing kernel and the answer is the same!

1. $O(mnk)$
2. $O(m^2n^2k)$
3. $O(mnk^2)$
4. $O(mn^2k)$

Computational Complexity of kMeans

Faster Algorithm?

What does it mean to say there is a faster algorithm?

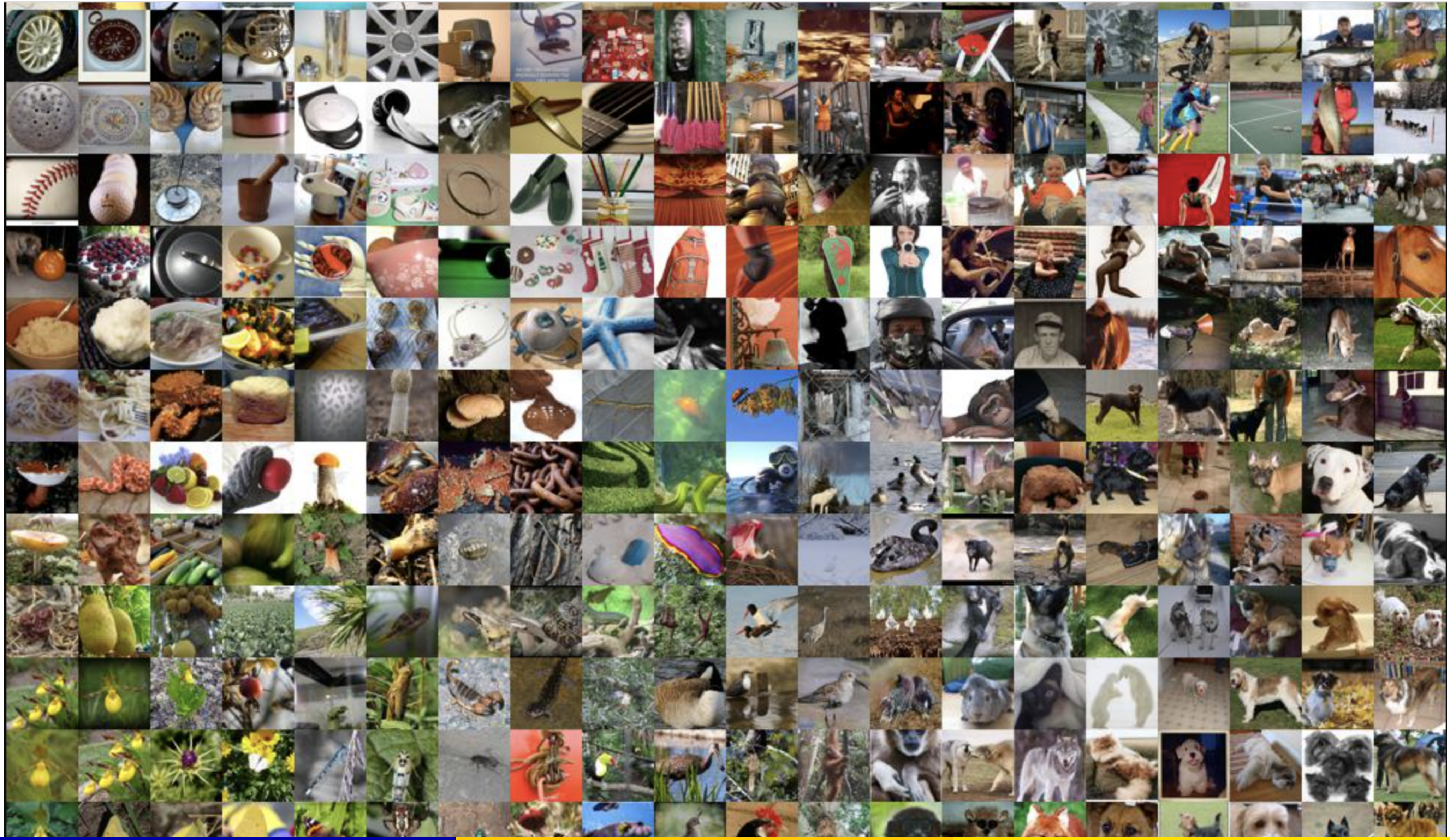
A_2 is a faster algorithm than A_1 to solve a problem if $O(A_2) < O(A_1)$.
Example: Which is faster: Selection Sort or Merge Sort?

Clustering for Data Visualization

Images

Let's say we had 1000 images and wanted to "cluster" them onto a super-grid of images so that similar images are closely placed on the super-grid and dis-similar are placed further away. k-means clustering will only get us half-way there!

Data Visualization: Stochastic Neighborhood Embeddings (SNE)!



SNE

High-level Idea

Find an embedding of images in 2 dimensions that put similar images close to each other and dis-similar images further away from each other.

SNE

High-level Idea

Find an embedding of images in 2 dimensions that put similar images close to each other and dis-similar images further away from each other.

Soft clustering

We don't have a K here. But if you look at any neighborhood of the super grid of images - They will look similar! We can call this soft-clustering.

SNE

Similarity measure through Probabilities

Let $x_1, x_2, \ldots$ represent features of the data in their original dimensions (e.g. images).

$$p_{j|i} = \frac{e^{-\|x_i - x_j\|_2^2 / 2\sigma_i^2}}{\sum_{k \neq i} e^{-\|x_i - x_k\|_2^2 / 2\sigma_i^2}}$$

SNE

Similarity measure through Probabilities

Let $x_1, x_2, \ldots$ represent features of the data in their original dimensions (e.g. images).

$$p_{j|i} = \frac{e^{-\|x_i - x_j\|_2^2 / 2\sigma_i^2}}{\sum_{k \neq i} e^{-\|x_i - x_k\|_2^2 / 2\sigma_i^2}}$$

Low-dimensional embedding Probabilities

Let $y_1, y_2, \ldots$ represent features of the data in lower (embedded) dimensions (e.g. 2 dimensions).

$$q_{j|i} = \frac{e^{-\|y_i - y_j\|_2^2 / 2\sigma_i^2}}{\sum_{k \neq i} e^{-\|y_i - y_k\|_2^2 / 2\sigma_i^2}}$$

Use the q probabilities for chaining

Image Chain

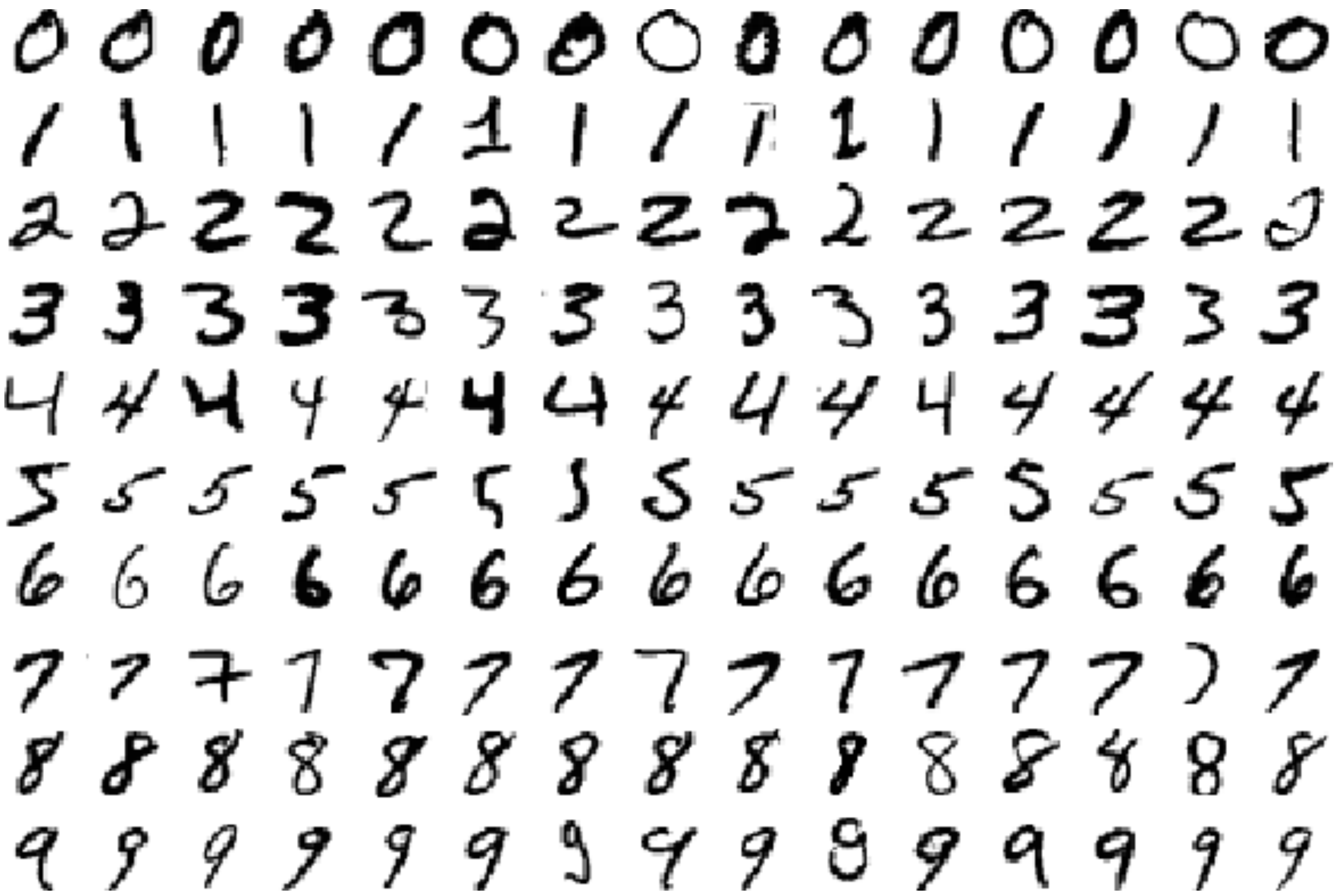
ICE #5 (3 mins break out)

Let's say you want to create a video that has 1000 images (e.g. the one we looked at earlier) in a sequence so that the images in the video transforms smoothly from one to the next. How would you go about doing this if you learned a tSNE representation for the images?

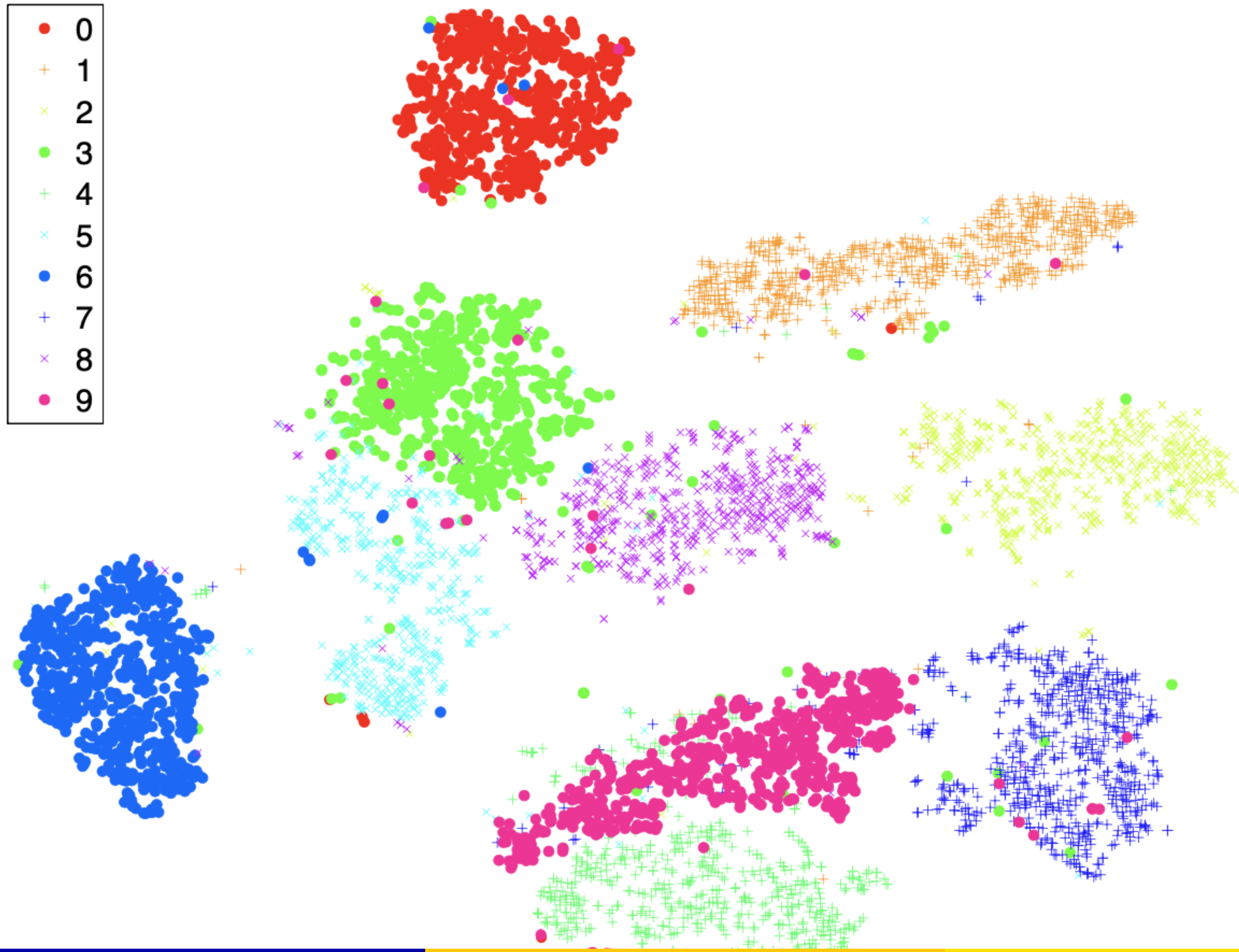
How do we create this grid?

tSNE Reference Paper

MNIST digits data set



MNIST tSNE embeddings



Next Topic: Total Variation (TV) for Image Smoothing

Image Smoothing Motivation

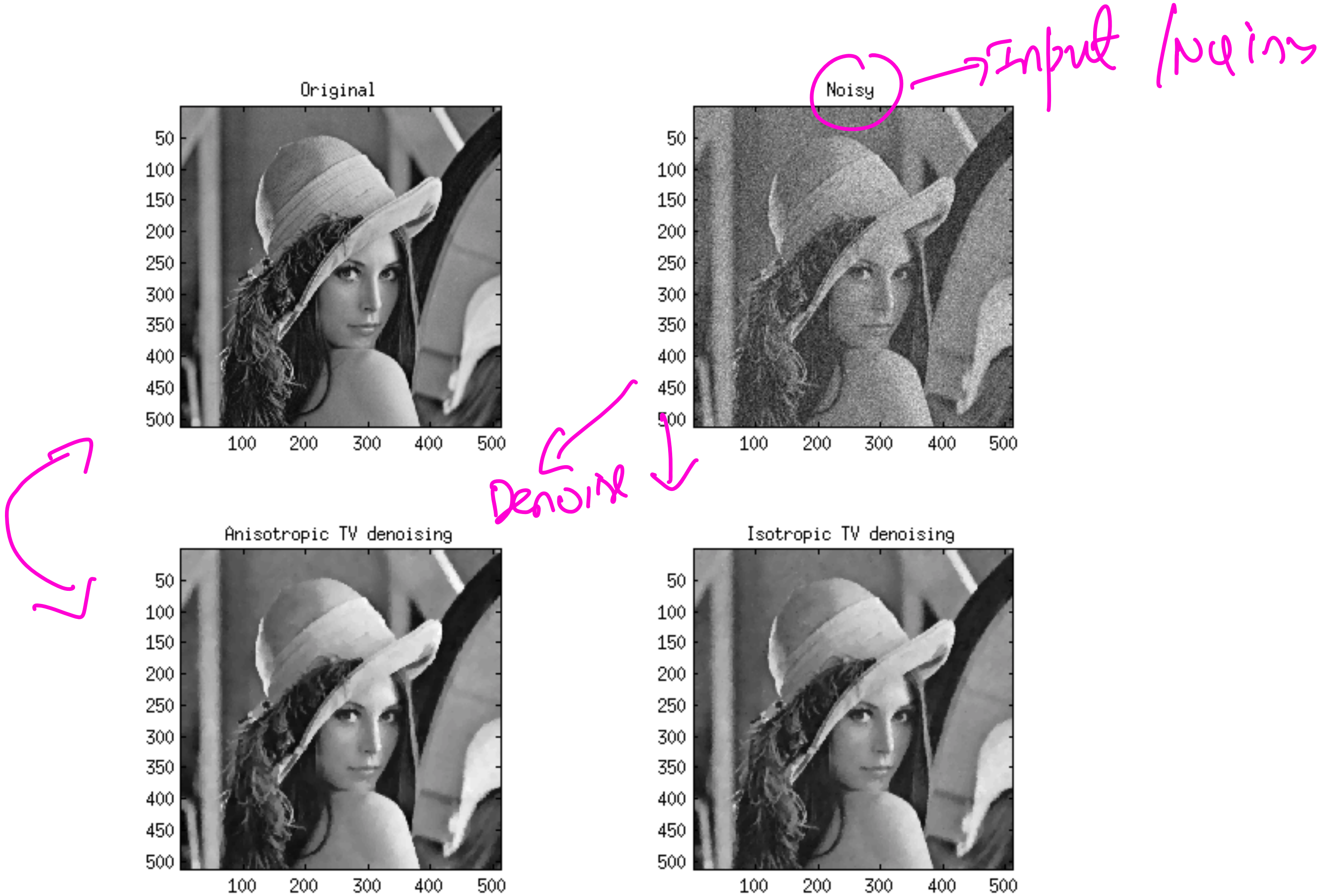


Image Smoothing Motivation

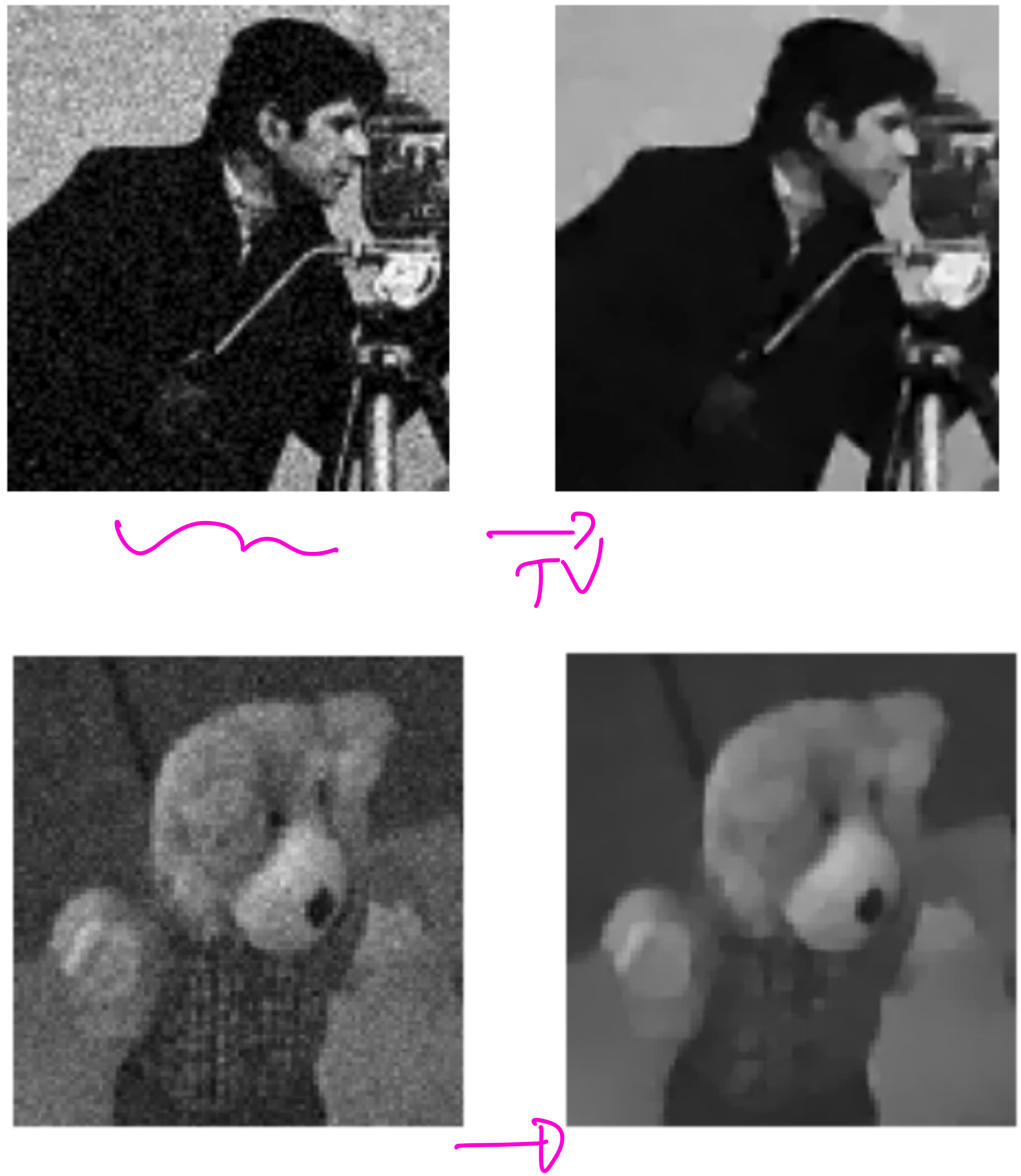
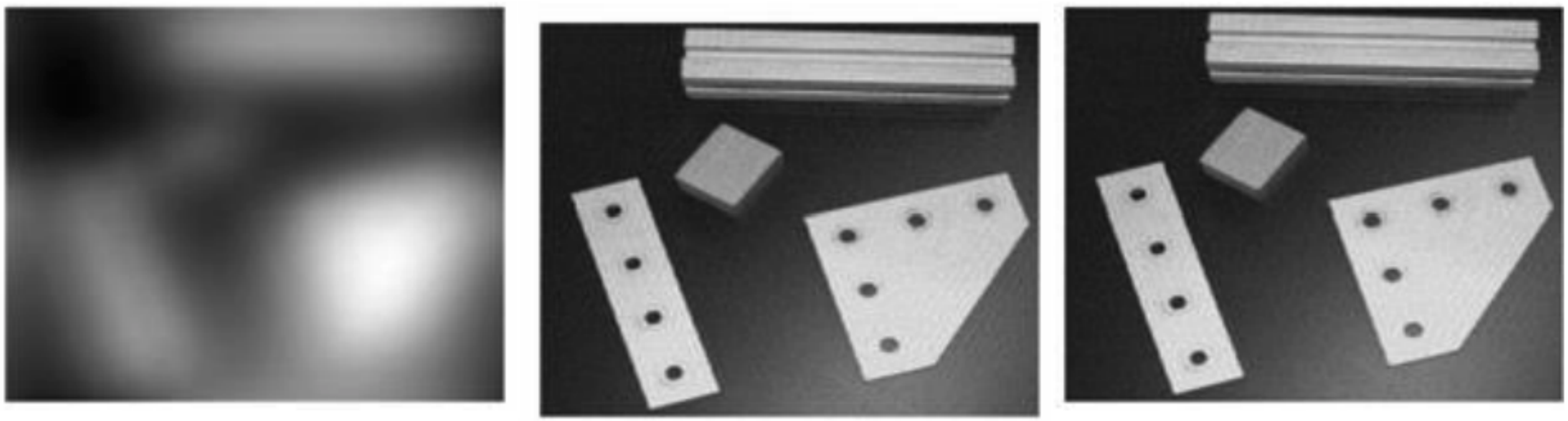


Image Smoothing Motivation



Blurred and Noisy image | Total Variation reconstruction | LASSO regularization reconstruction

Background for TV

Typical Regularizers

Total Variation (TV)

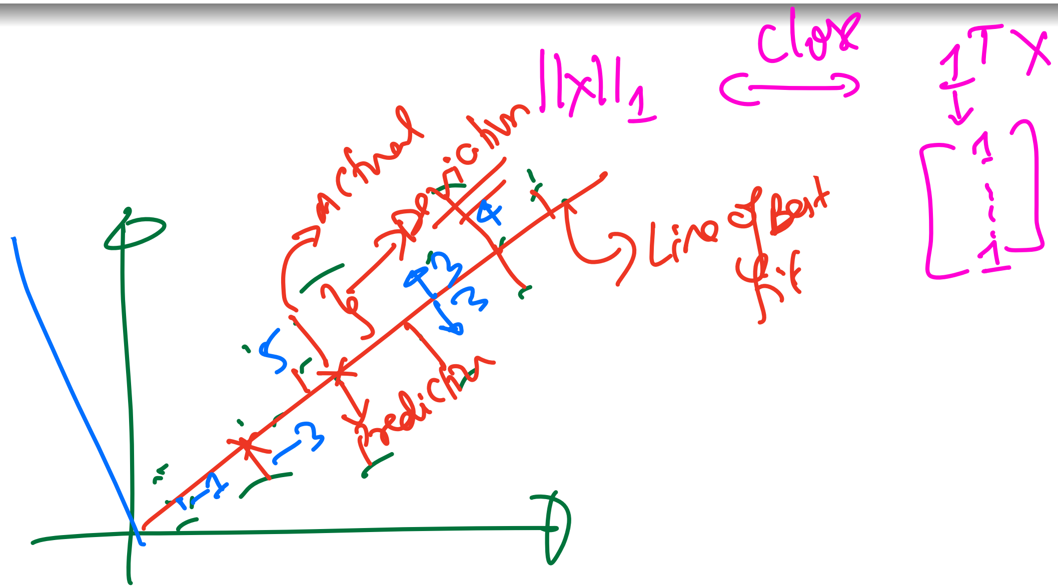
Is based on the concept of Regularization and using ℓ_1 or ℓ_2 norms. We will look into this background next.

Norms

ℓ_1 norm

The ℓ_1 norm of a vector is the sum of the absolute values of the elements in the vector!

$$\|x\|_1 = \sum_i |x_i|$$



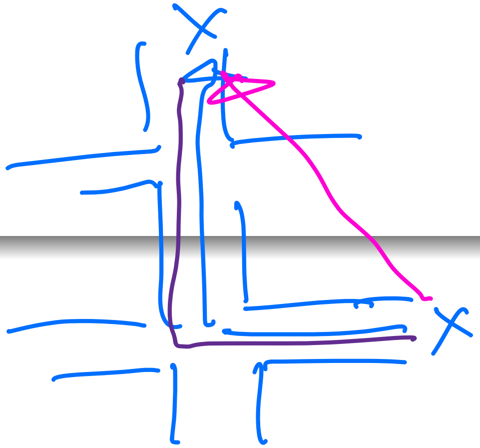
Norms

ℓ_1 norm

The ℓ_1 norm of a vector is the sum of the absolute values of the elements in the vector!

$$\|x\|_1 = \sum_i |x_i|$$

Manhattan Distance



ℓ_2 norm

$$\|x\|_2 = \sqrt{\sum_i |x_i|^2}$$

Euclidean Distance

Norms

ℓ_1 norm

The ℓ_1 norm of a vector is the sum of the absolute values of the elements in the vector!

$$\|x\|_1 = \sum_i |x_i|$$

ℓ_2 norm

$$\|x\|_2 = \sqrt{\sum_i |x_i|^2}$$

Notice!

$$\|x\|_2 \leq \|x\|_1$$

ICE #3

ℓ_1 and ℓ_2 norm of a matrix

Consider a simple and normalized image matrix,

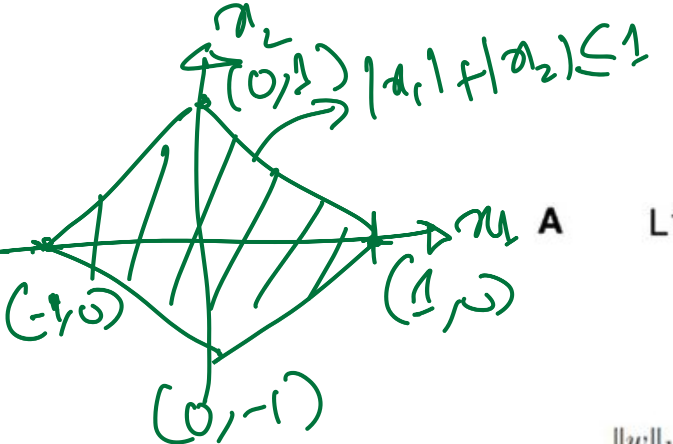
$$X = \begin{bmatrix} 1 & -2 \\ -1 & 4 \end{bmatrix}$$

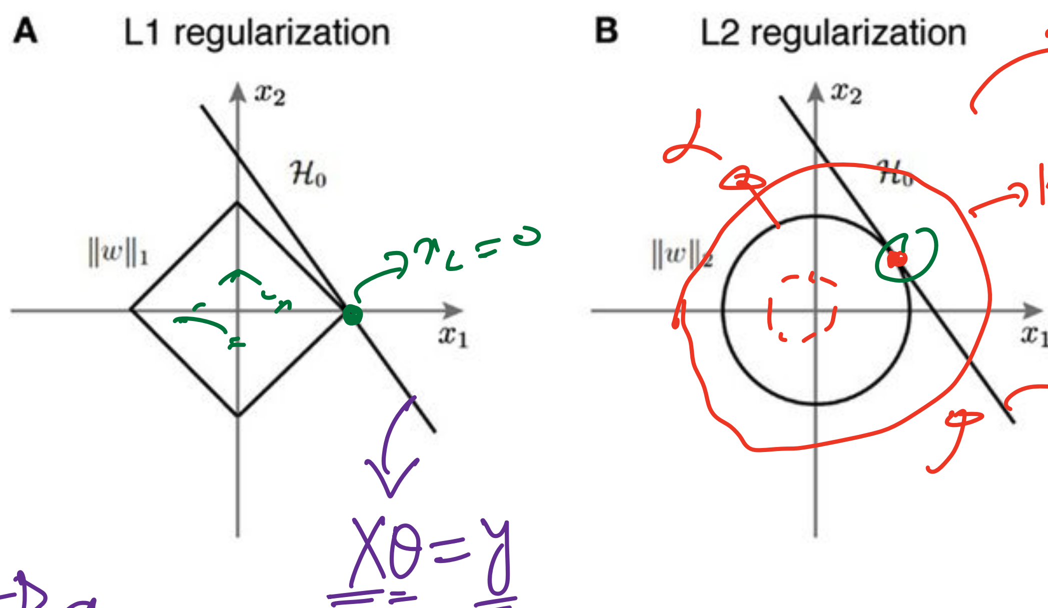
Treat the image X as a column vector. What would be the ℓ_1 and ℓ_2 norm of that column vector? Pick the closest option

1. 8 and 4
2. 2 and 5
3. 2 and 4
4. 8 and 5

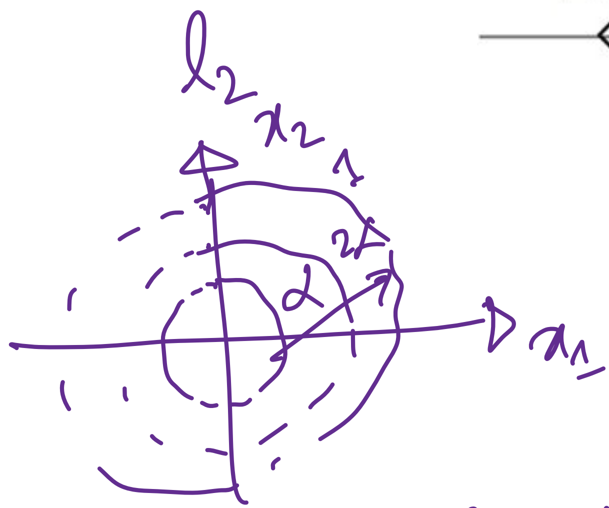
Norms and Norm Balls

Linear Systems





$\min \|\theta\|_2$ s.t. $X\theta = y$

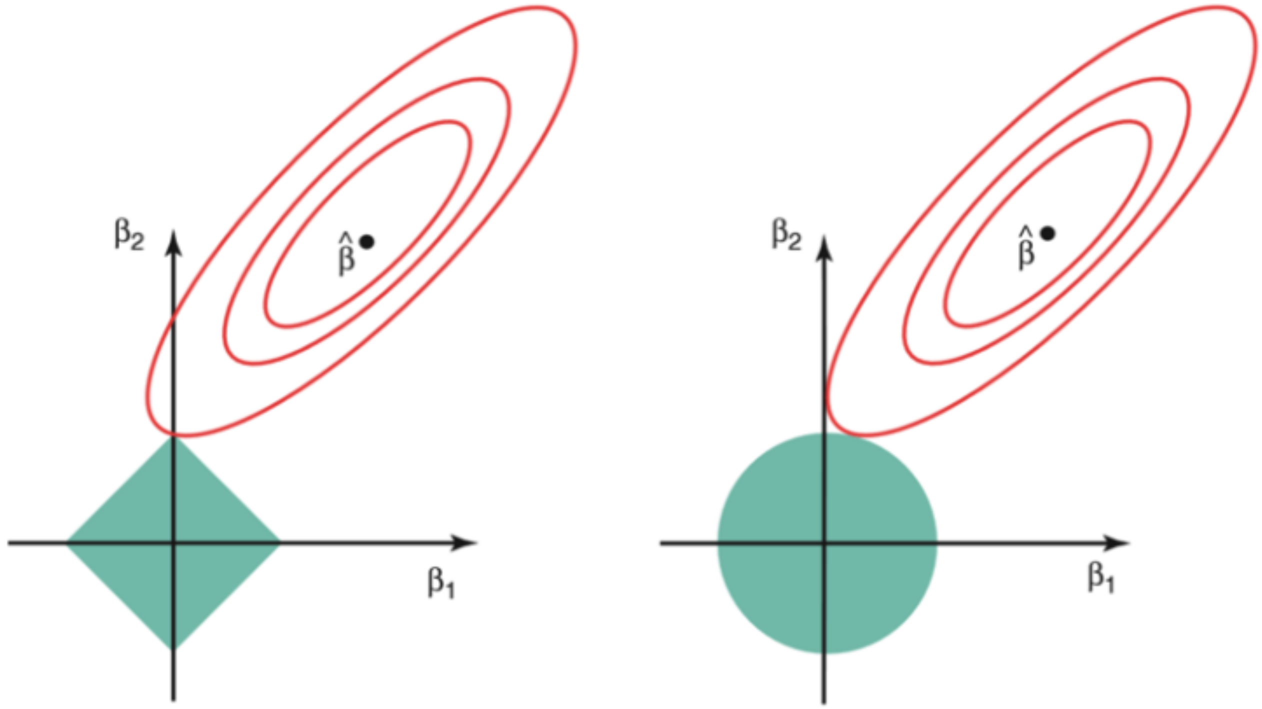


$x_1^2 + x_2^2 \leq \alpha$

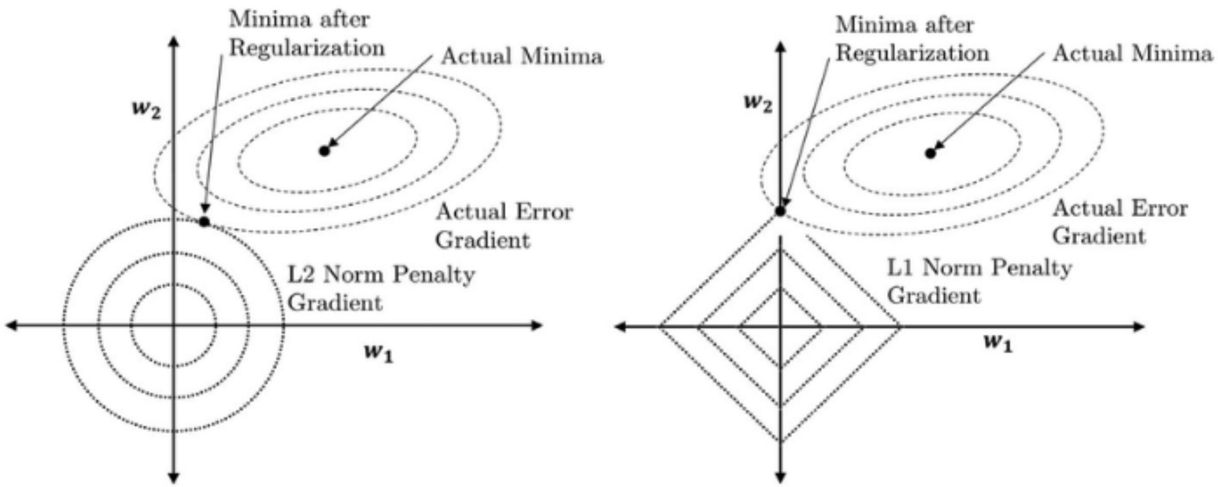
$\|x\|_1 = |x_1| + |x_2|$

$\|x\|_1 \leq \alpha \Rightarrow |x_1| + |x_2| \leq \alpha$

Norms and Norm Balls



Norms and Norm Balls



ℓ_1 norm and sparsity

Sparsity as a regularizer

ℓ_1 norm for the reasons described in the previous slide is known to produce sparse solutions (i.e. a vector with a bunch of zeros).

ℓ_1 norm and sparsity

Sparsity as a regularizer

ℓ_1 norm for the reasons described in the previous slide is known to produce sparse solutions (i.e. a vector with a bunch of zeros).

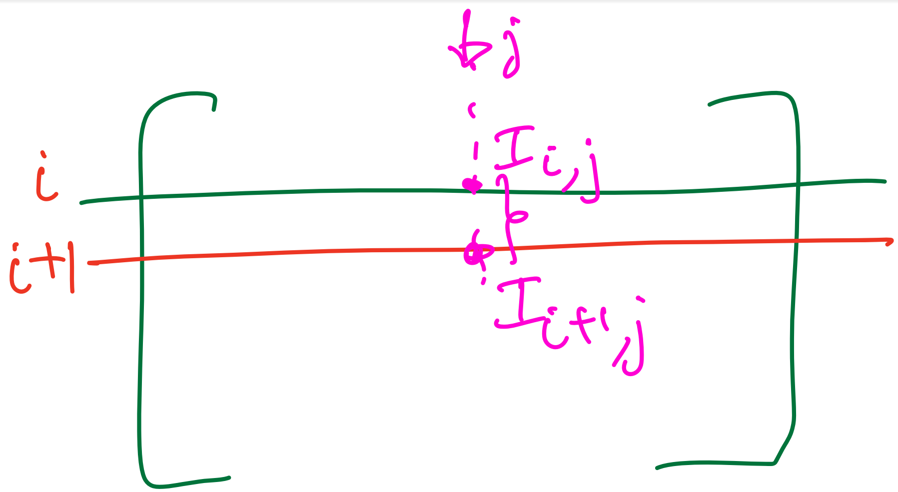
Sparsity for image denoising

This "sparsifying" property is what can be used to help denoise image. And that brings to TV!

Applying Sparsity to Image Smoothing

Horizontal Image Gradient Matrix, D_x

$$[D_x]_{i,j}(I) = I_{i+1,j} - I_{i,j} \rightarrow 0$$



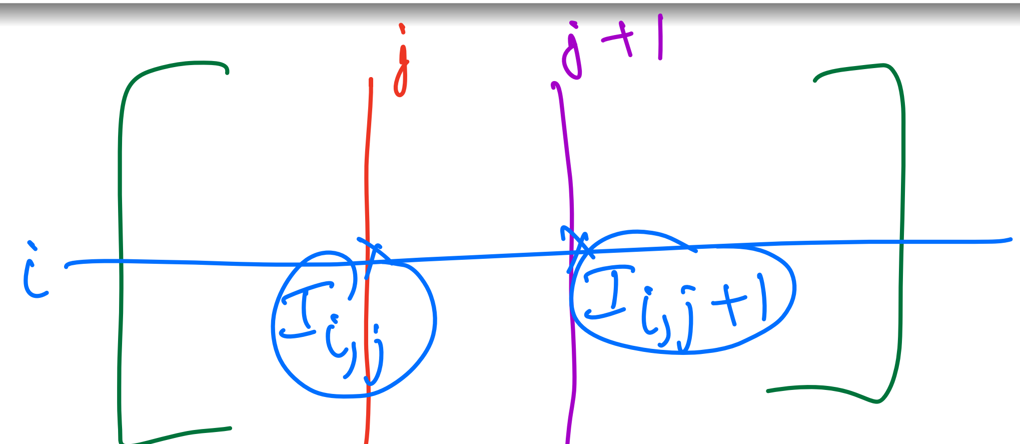
Applying Sparsity to Image Smoothing

Horizontal Image Gradient Matrix, D_x

$$[D_x]_{i,j}(I) = I_{i+1,j} - I_{i,j}$$

Vertical Image Gradient Matrix, D_y

$$[D_y]_{i,j}(I) = I_{i,j+1} - I_{i,j}$$



Total Variation (TV) definitions

TV

Let A be a measurment matrix that measured an image and gave an output f. Given f and A, can you reconstruct the image u in such a way that it is denoised as well? To do this, we solve the following optimization problem!

$$\min_{u} \frac{1}{2}\|Au - f\|_2^2 + TV(u)$$

parameter

Hyper-parameter

fn. that's a regularizer

l_1 norm

I → Noisy Image

$\hat{I}$ → De-noised Image

$$\min_{\hat{I}} \|\hat{I} - I\|_2^2 + \lambda TV(\hat{I})$$

stay close

make it smooth

Total Variation (TV) definitions

Anisotropic TV

$$\min_u \frac{1}{2}\|Au - f\|_2^2 + \lambda TV_{aiso}(u)$$

where,

$$TV_{aiso}(u) = \|D_x(I)\|_1 + \|D_y(I)\|_1$$

Applying l_1 norm to gradients

Keep Gradients in x & y direction sparse

Total Variation (TV) definitions

Anisotropic TV

$$\min_u \frac{1}{2}\|Au - f\|_2^2 + TV_{aiso}(u)$$

where,

$$TV_{aiso}(u) = \|D_x(I)\|_1 + \|D_y(I)\|_1$$

Isotropic TV

$$\min_u \frac{1}{2}\|Au - f\|_2^2 + TV_{iso}(u)$$

$$where,$$

$$\text{TV}_{aiso}(u) = \|\sqrt{|D_x(I)|^2 + |D_y(I)|^2}\|_1$$

Image Smoothing with TV

Using TV

Given the noisy image, using anisotropic and isotropic TV gives the results as below!

← Gradients not sparse

← Gradients are sparse

MRI Reconstruction from noisy input