To help give you a sense for what will be on the take-home midterm, we have provided you with several sample questions. Please note that the exam will be slightly longer than this.

Problem 1 *Machine Learning Fundamentals* **(16 points)**

(a) **(4 points)** List two methods to prevent underfitting and overfitting respectively.

(b) **(4 points)** How does increasing or decreasing a linear model's regularization parameter (the λ term in L2 regularization) affect the learned model's complexity?

(c) **(4 points)** Explain the roles of training, validation, and testing datasets in the machine learning workflow. Why is it important to separate these datasets?

(d) **(4 points)** Explain the purpose of cross-entropy loss in classification tasks. Why is it preferred over MSE (mean-squared-error) loss in classification?

Problem 2 *Image Filtering* **(16 points)**

Given the original 5×5 grayscale image matrix:

$$\text{Input Matrix} = \begin{bmatrix} 10 & 20 & 20 & 20 & 10 \\ 20 & 30 & 40 & 30 & 20 \\ 20 & 40 & 200 & 40 & 20 \\ 20 & 30 & 40 & 30 & 20 \\ 10 & 20 & 20 & 20 & 10 \end{bmatrix}$$

A noise spike is present at the center of the matrix (200). Your goal is to reduce this noise while preserving other details.

(a) **(4 points) Median Filtering** (3×3 filter window with zero padding) Apply a 3×3 median filter to the entire image using zero-padding to keep the output matrix the same size (5×5) as the input. Please provide the filtered output matrix.

(b) **(4 points) Gaussian Filtering** (3×3 filter window without padding) Apply a Gaussian filter with a 3×3 kernel using the following values (not normalized), without any padding, so that the output matrix size is reduced to 3×3:

$$\text{Gaussian Kernel} = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

After filtering, normalize by dividing each pixel value by 16. Please provide the resulting 3×3 output matrix.

(c) **(2 points) Is the above Gaussian filter kernel separable?** If so, pleased write the two $1D$ rectangular filter kernels that can be used to reconstruct the 3×3 Gaussian filter. If not, please give the reason.

(d) **(4 points) Bilateral Filtering** (3×3 filter window without padding) Apply a bilateral filter with a 3×3 window centered on each pixel. Use the Gaussian kernel from Question (b) for spatial weighting. For intensity weighting, we will use a simplified scheme that makes computation easier: we will use an *intensity threshold* of 30, which means that pixels with at most an intensity difference of 30 from the center pixel would contribute. For example, if the center pixel is 50, only neighboring pixels within the intensity range of $[20, 80]$ will contribute to the filtered results. More precisely, the equation for this simplified bilateral filter is as follows:

$$I_{\text{filtered}}(x) = \frac{1}{W(x)} \sum_{x' \in \mathcal{N}(x)} I(x') \cdot G_{\sigma_s}(\|x - x'\|) \cdot \delta(I(x), I(x'))$$

where:

- $I_{\text{filtered}}(x)$ is the intensity of the filtered image at pixel x;
- $\mathcal{N}(x)$ is the spatial neighborhood of pixel x;
- $I(x')$ is the intensity of the neighboring pixel x';
- G_{σ_s} is the spatial Gaussian kernel with spatial standard deviation σ_s. For simplification, we just use the Gaussian kernel from Question (b) as an approximation for G_{σ_s};
- $W(x)$ is the normalization factor:

$$W(x) = \sum_{x' \in \mathcal{N}(x)} G_{\sigma_s}(\|x - x'\|) \cdot \delta(I(x), I(x'))$$

- $\delta(I(x), I(x'))$ is the indicator function enforcing the intensity difference threshold:

$$\delta(I(x), I(x')) = \begin{cases} 1 & \text{if } |I(x) - I(x')| \leq 30 \\ 0 & \text{otherwise} \end{cases}$$

This equation combines both spatial and intensity constraints for bilateral filtering, using the given Gaussian kernel and intensity threshold. Please provide the resulting 3×3 output matrix after applying this bilateral filtering using a 3×3 filter window without padding.

(e) **(2 points)** Compare the output matrices from the three filters. Discuss in a few sentences: which filter best reduced the noise spike while preserving details and explain why.

Problem 3 *Fourier Transform* **(16 points)**

(a) **(2 points)** If we rotate an image clockwise by 90°, what effect does this have on its Fourier transform?

a) The Fourier transform rotates 90° clockwise.

b) The Fourier transform rotates 90° counter-clockwise.

c) The Fourier transform rotates 180°.

d) No general statement can be made about the Fourier transform.

(b) **(2 points)** If we scale an image by a factor of 2, doubling its size, what effect does this have on its Fourier transform?

a) The Fourier transform scales by a factor of 2.

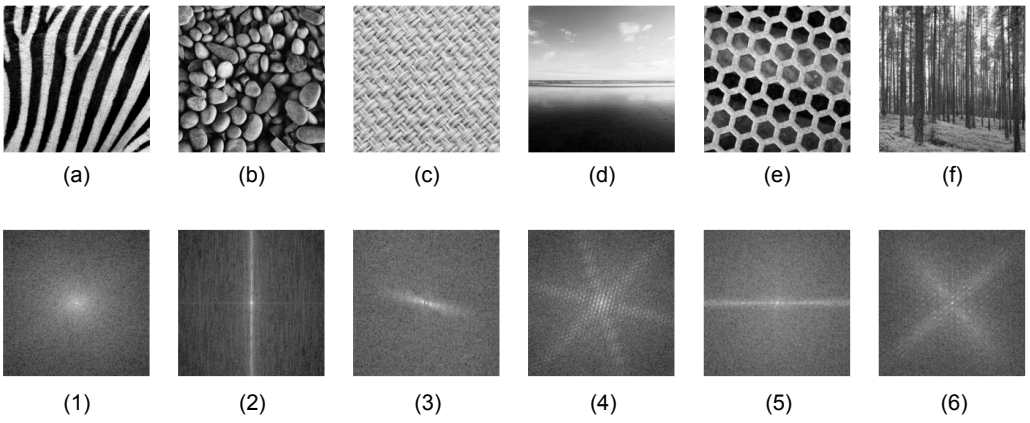
b) The Fourier transform scales by a factor of $\frac{1}{2}$.

c) The Fourier transform is unaffected.

d) No general statement can be made about the Fourier transform.

(c) **(12 points)** Please match the following images to their respective Fourier transforms.

a) _____ b) _____ c) _____ d) _____ e) _____ f) _____



Problem 4 *Optical Flow* **(16 points)**

You are given `Image 1` and two optical flows: a forward flow and a backward flow. The forward flow specifies where a pixel `Image1[i, j]` goes in `Image2`. It contains a pair of values `(x, y)` that specifies the displacement of the pixel. Similarly, backward flow specifies where a pixel in `Image2[i, j]` goes in `Image1`.

Complete the functions `forward_warp_image()` and `backward_warp_image()` which uses forward flow and backward flow to warp `Image 1` using forward and backward/inverse warping respectively. The warped image should look similar to `Image 2` (shown below).

In the code that we have given you, the optical flow for the pair of images (`Image1`, `Image2`) is represented as a matrix of shape `(H, W, 2)` where `flow[i, j]` represents the displacement `(x, y)` such that moving the image pixel `Image1[i, j]` by `(x, y)` (resulting in `(i+y, j+x)`) will give the `Image2` pixel.

Colab link: https://colab.research.google.com/drive/1h7VTyrmUymcgIvy-StXym1BxM_d3uMe-?usp=sharing

Image 1

Image 2