Mini-Project #1 Slidedoc

Zhengqi Wang
MEng in ECE @ Duke University

Presentation Overview

Project Goals Description

Project Goals

The primary goal of our Mini-Project #1 in compressed sensing image recovery is to explore and implement the principles of compressed sensing to recover images that have undergone significant information loss. Specifically, we aim to apply the LASSO (Least Absolute Shrinkage and Selection Operator) image reconstruction algorithm to estimate missing pixels in corrupted grayscale images. This project involves simulating corruption in two distinct images - ”Fishing Boat” and ”Nature” - to evaluate our algorithm’s performance across different scenarios of spectral content and textures.

Figure: Original pictures of ’Fishing Boat’ and ’Nature’.

Importance of the Problem

Compressed sensing addresses critical issues in data acquisition, reducing the need for extensive data without compromising the integrity of image recovery. It's pivotal in fields where collecting full data sets is impractical, offering advancements in medical imaging, astronomy, and beyond, by enabling accurate reconstructions from sparse observations.

Compressed Sensing and Underdetermined Systems

Compressed sensing is inherently linked to solving underdetermined linear systems, where the number of measurements is less than the dimensionality of the signal. This connection is crucial for image recovery, leveraging sparsity to reconstruct images from fewer samples than traditionally required, challenging conventional bounds on sampling theory. We aspire to achieve a detailed understanding and hands-on experience with the mathematical formulation of L1 regularized regression (LASSO), its application to solving underdetermined linear systems, and the practical considerations in tuning the algorithm for image recovery.

Broader Applications

The methodology extends beyond image recovery to diverse areas such as secure communication, biomedical signal processing, and environmental monitoring. By efficiently processing incomplete data, this approach enhances capabilities in real-time diagnostics, surveillance, and remote sensing, demonstrating its broad impact.

Subject Images Analysis

The chosen images, "Fishing Boat" and "Nature," exemplify the complexity and variability of real-world scenarios. Their selection tests the robustness of compressed sensing techniques across different textures and patterns, validating the algorithm's versatility and effectiveness in practical applications.

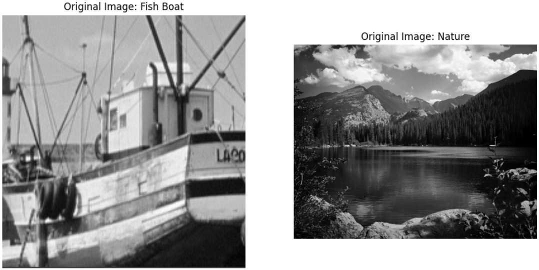

Figure: Two Original Picture: 'Fishing Boat' (left) and 'Nature' (right).

Expected Goals

- We aspire to achieve a detailed understanding and hands-on experience with the mathematical formulation of L1 regularized regression (LASSO), its application to solving underdetermined linear systems, and the practical considerations in tuning the algorithm for image recovery.
- The mathematical foundation of LASSO, combined with DCT and median filtering, provides a robust framework for image recovery.
- This approach leverages sparsity and efficient representation, essential for compressed sensing applications.

Why choose LASSO for Image Recovery?

- LASSO (L1-norm regularization) is chosen for its sparsity-promoting properties, essential in compressed sensing.
- It helps in recovering images from underdetermined systems by encouraging solutions with fewer non-zero coefficients.

Mathematical Formulation of LASSO

LASSO Description

LASSO is a regression analysis method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the statistical model it produces. The LASSO technique achieves this by constraining the sum of the absolute values of the regression coefficients (the L1 norm of the coefficients) to be less than a fixed value.

The optimization problem for LASSO can be written as:

$$\min_{\kappa} \|A\kappa - D\|_2^2 \quad \text{subject to} \quad \|\kappa\|_1 \leq \lambda,$$

where:

- A is the design matrix containing the predictors.
- κ is the vector of coefficients.
- D is the observed output.
- $\|\cdot\|_2$ denotes the L2 norm, representing the Euclidean distance or mean squared error.
- $\|\cdot\|_1$ denotes the L1 norm, representing the sum of the absolute values of the coefficients.
- λ is the regularization parameter controlling the strength of the penalty.

This is equivalent to a regularized version of the least squares, where the penalty is the L1 norm of the coefficients:

$$\min_{\kappa}\left(\|A\kappa - D\|_2^2 + \lambda\,\|\kappa\|_1\right),$$

which encourages sparsity in the coefficient vector κ by shrinking some coefficients to zero, effectively performing feature selection.

DCT as Basis and LASSO Application

Why DCT?

- The Discrete Cosine Transform (DCT) is well-suited as it represents images efficiently, capturing energy in fewer coefficients.
- Applying LASSO with DCT helps in isolating significant image features, facilitating recovery.
- The ”constant” basis vector accounts for the average color intensity, treated separately to maintain image brightness.

Mathematical formula for DCT as follows:

$$\alpha_u \cdot \beta_v \cdot \cos\left(\frac{\pi(2x-1)(u-1)}{2P}\right) \cdot \cos\left(\frac{\pi(2y-1)(v-1)}{2Q}\right),$$

where, x, u $\in \{1, 2, \ldots, P\}$, (number of pixels in horizontal direction),
y, v $\in \{1, 2, \ldots, Q\}$, (number of pixels in vertical direction)
The visualization of basis chips are attached as following iamge.

Examples of single basis chip for $K = 8$ and $K = 16$.

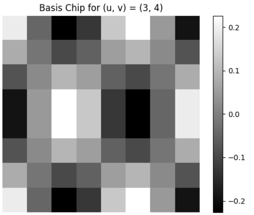

Figure: K=8 examples of basis chip

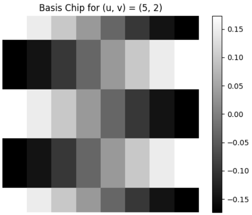

Figure: K=8 examples of basis chip

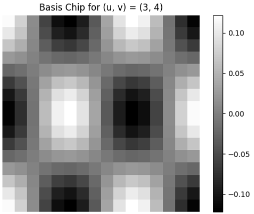

Figure: K=16 examples of basis chip

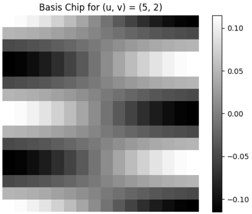

Figure: K=16 examples of basis chip

The following images are the two basis matrix used in this project. Left one under $K = 8$, right one under $K = 16$.

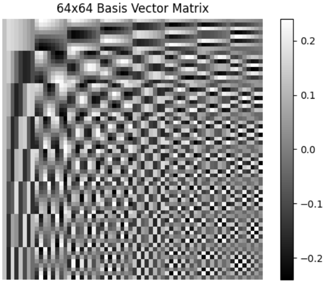

Figure: K=8 basis matrix result

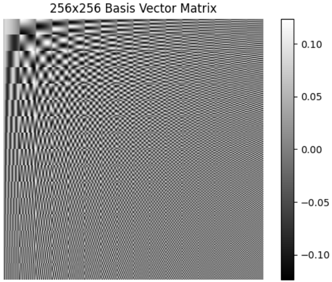

Figure: K=8 basis matrix result

Procedures of LASSO for Image Recovery

- **Basis Vector Matrix:** Foundation for creating a test block with a known sparse representation.
- A **Sandbox LASSO Problem** is constructed to understand and debug the recovery process.
- **Cross Validation:** Without a constant term, a block is formed by multiplying basis vectors with set non-zero weights.
- **Validation Parameter Selection:** The optimal validation parameter is selected by minimum mean square error.
- **Running LASSO:** The algorithm seeks to recover the sparsest solution, zeroing out the weight for the constant basis vector.
- **Comparing and Validation:** Knowledge of the true images aids in verifying the accuracy of LASSO's estimations.

This structured approach allows for a controlled environment to verify the effectiveness of LASSO in image recovery, ensuring that the model estimated is close to the known true model.

Improving Image Quality with Filtering Post-LASSO

Rationale for Applying Filtering

- Post-processing step to enhance image quality after sparse recovery via LASSO.
- Filtering helps in removing noise and artifacts introduced during image acquisition or reconstruction.

Median Filtering

- Median filtering is a non-linear process creating a new pixel value by finding the median of neighboring pixel values.
- Chosen for its effectiveness in mitigating impulsive noise, which is common in image recovery scenarios.
- Preserves edges while removing noise, which is crucial for maintaining the structural integrity of the image.

Median Filtering vs. Frequency-Selective Filtering

Why Median over Frequency-Selective:

- Median filtering operates in the spatial domain, directly targeting the pixel values, which makes it more effective for certain types of noise like salt-and-pepper.
- Frequency-selective filters (low-pass or high-pass) may not be as effective in preserving edges or may inadvertently remove important details along with the noise.

Implementation of Median Filtering:

- A typical filter size is 3×3, which is sufficient to remove noise without overly smoothing the image.
- Implementations are available in various software packages:
  - Python: `medfilt2` function from SciPy's multidimensional image processing module (`scipy.ndimage`).

Outcome:

- Image quality can be improved as validated by the reduction in error metrics after applying median filtering to the recovered image. But there is a great relationship between the degree of improvement and the sensed value.

Simulation Process

Simulation Overview

- Simulations performed to evaluate LASSO image recovery under various degrees of corruption.
- Applied to two images: ”Fishing Boat” and ”Nature”.
- Degrees of corruption quantified by S - the number of sensed pixels.

Sample Image Corruption Process

Process Description

To synthesize corrupted images for testing the reconstruction algorithm:

1. Divide the image into blocks of size $K \times K$.
2. Retain S sensed pixels within each block, marking the remaining as missing.
3. Represent missing pixels with invalid values like "NaN" or "-Inf" for clarity.
4. Maintain the same partitioning of sensed and missing pixels across all cross-validation iterations for consistency.

Visualization of the corruption process where $K = 8$ and $K = 16$, resulting in $K^2 - S$ missing pixels per block. The images on the following page represent sensed pixels in gray and missing pixels in orange. Images illustrating the original block, sample corrupted block and sample reconstructions blocks are displayed on the following pages.

The ”Nature”, ”Fish Boat” images after sensing ($K^2 - S$) pixels by random.

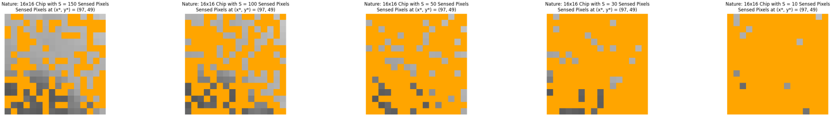

Figure: Nature images with K=16 and varying S values.

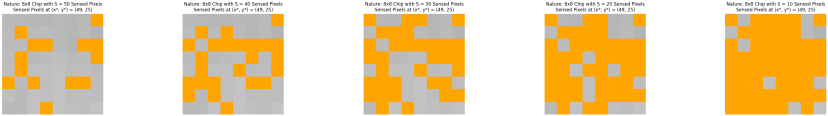

Figure: Fish Boat images with K=8 and varying S values.

Both images after reconstructing ($K^2 - S$) pixels from LASSO model. The reconstruction only apply towards unsensed pixels, and its color has been lightly changed.

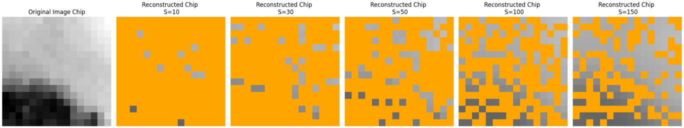

Figure: Nature image block reconstructions for various S values.

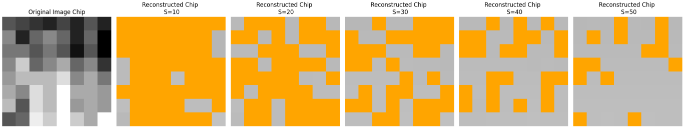

Figure: Fish Boat image block reconstructions for various S values.

Fullsize Image Reconstruction

Reconstruction for Different Values of S

The reconstruction process for each block of the image involves the following steps:

1. The original image is divided into blocks of size $K \times K$.
2. For each block, sensed pixels are retained randomly, and the rest are considered missing.
3. Cross-validation is performed to find the optimal regularization parameter λ for LASSO, choosing 3 pixels as testing each time and performing 20 iterations.
4. LASSO regression is then applied using the optimal λ to reconstruct the missing pixels.
5. The reconstruction performance is evaluated using the Mean Squared Error (MSE) between the original and reconstructed blocks.

Images illustrating the corruptions and reconstructions for the whole blocks using the best λ for each value of S are displayed on the following page.

Full size sensed images and reconstructed images for different sensed values. As the sensedvalue continues to decrease, the effect of our reconstruction has also brought slight improvement.

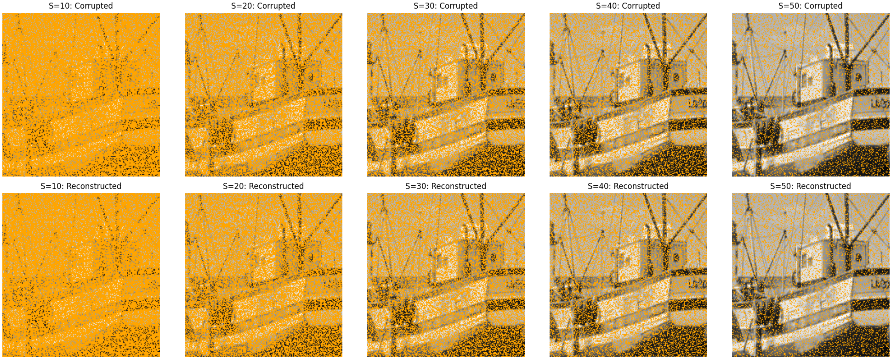

Figure: Reconstructed fish boat images (K=8) sample corrupted image and reconstructed image.

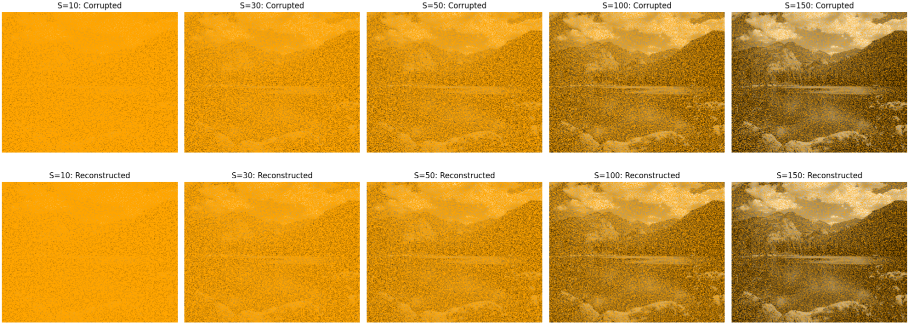

Figure: Reconstructed nature images (K=16) sample corrupted image and reconstructed image.

Post-Processing: Filtering Method

Median Filter Description

Median filtering replaces each pixel in an image by the median of its neighborhood.

- Sort all pixel values in an $m \times n$ block, centered at (x, y), to find the median.
- Replace the pixel value (f(x,y)) by the median.
- Good for mitigating impulsive noise.

Apply a median filter (MF) to improve the quality of recovered images:

- Set filter size as 3×3.
- Packages used in Python: `scipy.ndimage.median_filter`.

The images of comparing the error of the recovered image with median filtering and without median filtering.

The addition of median filter can reduce the noise in some reconstruction color blocks. However, as the sensed value increases, the median filter will cause the overall image to become blurrier.

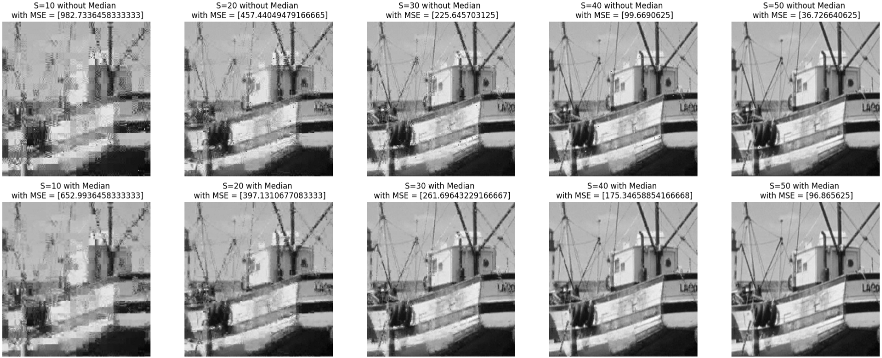

Figure: Reconstructed fish boat images (K=8) without and with median filtering, including MSE in the plot title.

The addition of median filter can reduce the noise in some reconstruction blocks. For this nature images, which has more complex information than fish boat, median filter can always have a good impact on reconstruction result.

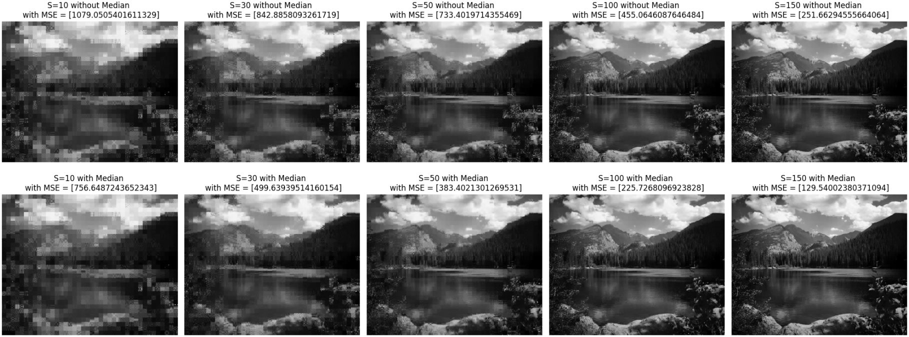

Figure: Reconstructed nature images (K=16) without and with median filtering, including MSE in the plot title.

Analysis of Mean Squared Error (MSE)

- As the number of sensed pixels increases, the reconstructed image quality visibly improves.
- The MSE decreases, indicating closer approximation to the original image.

Image	Sensed Pixels (S)	MSE
5*Fish Boat	10	0.296875
	20	1.3125
	30	2.203125
	40	4.546875
	50	4.453125
5*Nature	10	0.3203125
	30	2.671875
	50	6.99609375
	100	15.34765625
	150	24.37109375

Table: MSE values for different sensed pixel counts and images.

Analysis of Lambda

- By applying cross-validation with $\frac{S}{6}$ sensed pixels reserved for testing and iterating 20 times of cross-validation folds, we can get the corresponding MSE for different MSE. Thus we pick the λ with the minimum MSE as optimal λ.
- By using 8×8 block for the pixels in the original image, and took log_{10} of the lambda value used in each block before drawing. The whiter the color in the grayscale image, the greater the lambda value and the darker it is. Indicates that the smaller the value of lambda is. By using 22 lambda in the range of 1e-10 to 1e+11, we can draw and display the used lambda values as following slide.
- Whether it is the lambda image of nature or fish boat, some features of the original image can be reflected (such as the lake in the center of the nature image, the dark color block in the lower right corner of fish boat). Moreover, as the sensed value increases, the lambda The value also gradually becomes smaller, and tends to several fixed values (in this experiment, the values of lambda are usually 0.1 and 0.001, with a small number of other values). And as the sensed value increases, number of light color blocks also gradually decreases. The original features of the image also become more clearly visible.

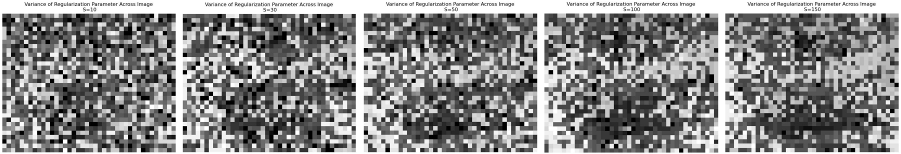

Figure: Visualization of log10 (regularization parameter) for Nature image.

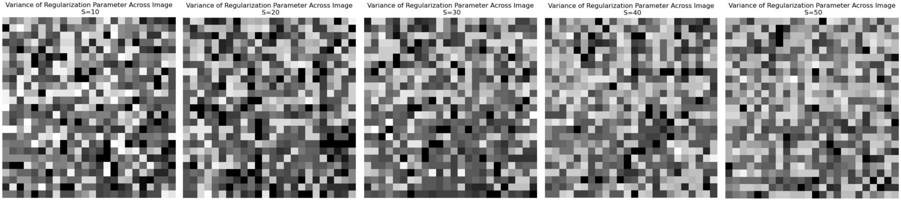

Figure: Visualization of log10 (regularization parameter) for Fish Boat image.

Some Uncertainties about Regularization Parameter

- In our grayscale image, there are very few types of colors displayed. This means that the lambda values taken during reconstruction are very concentrated, and concentrated at 0.1. This means that most of the blocks are using 0.1 as the lambda value. After I checked the logic of the algorithm, my lambda calculation process is completely consistent with that in the tutorial slides.
- After discussions with my peer students, I think the difference between my images and other students may be in my choice of cross-mapping method. I did not hand-write the logic of cross-mapping, but directly used `sklearn.model_selection` package's `train_test_split` equation. Even though I have ensured that a different random method will be used every time when performing cross-mapping, the result I get is that lambda only appears on a small number of values, and the distribution is inconsistent though the whole image.

MSE vs. Regularization Parameter

- The four images presented on following slide are plots of Mean Squared Error (MSE) vs. S for both subject images.
- Analysis reveals how reconstruction quality varies with the degree of corruption and the impact of median filtering.Through the comparison between MSE and lambda, we can find that under different sensed value conditions, all MSEs have global minimum values near 0.1 and 0.001. This also successfully explains why our previous lambda In the grayscale image, most lambda values are concentrated between 0.1 and 0.001.
- Interpretation of results indicates scenarios where LASSO recovery excels or faces limitations when lambda values are extreme small or extreme large.

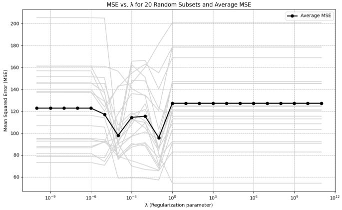

Figure: Fish Boat's MSE vs. Regularization Parameter(Lambda)

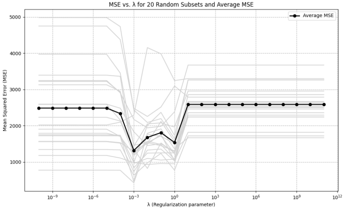

Figure: Nature's MSE vs. Regularization Parameter(Lambda)

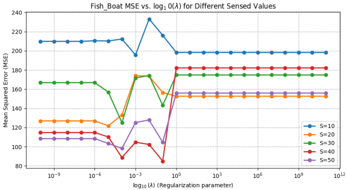

Figure: Fish Boat's MSE vs. Lambda under Different Sensed Values

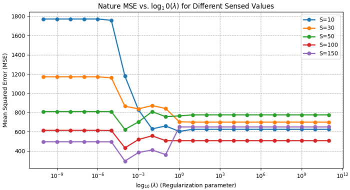

Figure: Nature's MSE vs. Lambda under Different Sensed Values

MSE vs. Regularization Parameter Before and After Median Filtering

The following images present corresponding MSE before and after median filtering before and after median filtering. Median filtering expose different outcome on two images.

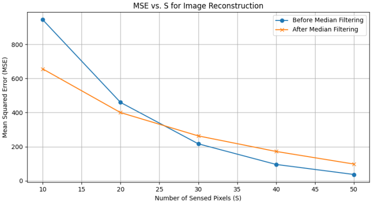

Figure: Plots of MSE vs. measure of corruption for image-level reconstruction for **fish boat image**, with and without median filtering.

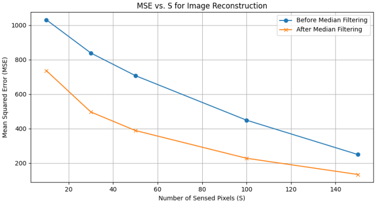

Figure: Plots of MSE vs. measure of corruption for image-level reconstruction for **nature image**, with and without median filtering.

Analysis towards MSE vs. Regularization Parameter

- In fish boat and nature images, median filtering shows different characteristics.
- In the case of fish boat, for the case of small sized value ($S < 30$), median filtering optimizes the reconstruction result and reduces the MSE successfully. It shows that the median filtering method successfully reduces the noise in the original reconstructed image. However, median filtering does not give positive effect in the case of large sensed value ($S > 30$). The reason for this is that the reconstruction is already very accurate, and the introduction of median filtering will only blur the features of the reconstructed image, resulting in a negative effect.
- In the case of nature images, median filtering has always led to better optimization results. The reason for this is that nature images are more complex and contain more information than fish boats. Simple reconstruction generates a lot of noise, which creates a need for median filtering.
- The main reason for this phenomenon is the difference in the nature of the pictures of fish boat and nature. Nature contains a lot of content and details, which also creates more demand for medial filtering.

Beirf Intro towards Field Test Image

In this section, we will delve into the application of the LASSO image reconstruction algorithm on a field test image. This particular image, presented in grayscale, has approximately 65% of its pixels missing. These absent pixels are notably indicated by NaN values in the associated data file.

Field Test Parameters

- The reconstruction process will involve segmenting the image into 8×8 blocks due to the chosen block size ($K = 8$). The sensed value within each block tends to be different. Additionally, to enhance the quality of the restored image and mitigate noise, a median filter with a 3×3 size will be employed post-reconstruction.
- I used the NaN values in the original image as missing pixels, and all the remaining non-NaN values as sensed pixels. At the same time, I performed 20 cross validations on the sensed pixels, and each time I selected $\lfloor \frac{S}{6} \rfloor$ pixels as training pixels. After the cross validation, I selected the best validation parameters for each block. The validation parameters are used for reconstruction, and median filtering is performed after reconstruction.

The ensuing slides will visually demonstrate the original, corrupted, and reconstructed images, exhibiting the effectiveness of the chosen methodology.

- Compared with the method of fixing the number of sensed pixels in each block, the method I can use all non-NaN values as sensed pixels has a better reconstruction effect. This has been well demonstrated in the results of my reconstruction without median filtering.
- The results of reconstruction without median filtering clearly reflect the characteristics of the image, but with a small amount of noise. After median filtering, these noises are also well repaired. But this is at the expense of image clarity.

Figure: Visualizations of the original (corrupted image) and the reconstructed images. Left: Corrupted Image Visualization. Center: Reconstructed without Median Filtering. Right: Reconstructed with Median Filtering.

- After cross-validation, we selected the smallest one from the MSE corresponding to the 23 lambdas as the lambda value of the reconstruction. Each of the block has size of 8×8
- These lambda values are also very concentrated, around 0.1, and 10. The performance on the image is that the color patches have fewer types of colors.
- The overall color of the grayscale image is dark (the overall lambda values are small).
- Meanwhile the regularization parameter visualization image also retains some features of the image, such as the piano keys in the lower half.

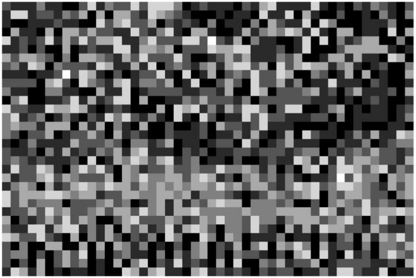

Figure: Visualization of $\log_{10}$(regularization parameter) as a function of image block.

Summary for Image Reconstruction

- Better image reconstruction results can be achieved by using cross-validation to get the best lambda and performing LASSO reconstructing grayscale images. The reconstruction results are better for images that contain a single message (simple image structure), are small, and have a uniform distribution of lost pixels.
- The reconstruction effect of LASSO depends very much on the selection of the regularization parameter. If the regularization selection is too small or too large, the reconstructed image will have blocks of blur and a lot of noise.
- Design decision of choosing a block size should be fitted with the image size. For subsequent median filtering, K must preferably be sited at 6 or higher. Otherwise, median filtering should not be used and other methods should be tried..
- Choosing 3*3 medial filtering is a conservative attempt, so that too many pixel features will not be lost, and the purpose of noise reduction can be achieved. When processing larger blocks, the range of medial filtering can be appropriately expanded to maintain better noise reduction effect.

Assessing LASSO for Image Reconstruction

The design choice for the basis and regularization parameter plays a important role in the quality and effectiveness of the reconstructed image, determining how well the algorithm can recover the original image from a set of sampled pixels.

- **Basis Choice:** The choice of basis functions, such as the Discrete Cosine Transform (DCT), affects the system by determining how the image information is compressed and represented. Different bases can capture varying levels of detail and affect the sparsity of the solution, which is crucial for reconstruction.
- **Regularization Choice:** Regularization controls the trade-off between fitting the data and maintaining a smoother model. In LASSO, the regularization parameter affects the number of non-zero coefficients in the model, influencing the sparsity of the reconstructed image. The right level of regularization can help suppress noise while retaining essential image features. However, a wrong choice can lead to block blurring and noise in the reconstructed image. Therefore, it is reasonable to choose the regularization parameter that corresponds to the minimum value of MSE. The result of the reconstruction with the most suitable lambda will be the closest to the original image.

Lessons Learned and Future Directions

- Effective cross-validation strategy was key for selecting the optimal regularization parameter, which ensured a balance between fitting the known data and generalizing well to unknown pixels.
- Sensed value selection should based on the recovering image's quality and block size. We are committed to using fewer sensed pixels to come and go for better recovery results.
- Efficiency in code execution was improved by streamlining the lambda selection process, which allowed for a one-time computation of all lambda values and their corresponding reconstruction outcomes.
- Future improvements will focus on enhancing algorithm efficiency, possibly through more advanced optimization techniques or parallel processing to reduce computational time.
- A significant achievement was the ability to implement and adapt complex mathematical concepts into a functional image reconstruction algorithm, showcasing the practical application of theoretical knowledge.

References

- For **LASSO** regression:
  - Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 267-288.
- For Median Filtering:
  - **scipy:** Virtanen, P., Gommers, R., Oliphant, T. E., et al. (2020). SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods*.
- Python packages:
  - **scikit-learn:** Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.
  - **matplotlib:** Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3), 90-95.
  - **NumPy:** Harris, C. R., Millman, K. J., van der Walt, S. J., et al. (2020). Array programming with NumPy. *Nature*, 585(7825), 357-362.
  - **pandas:** McKinney, W. et al. (2010). Data Structures for Statistical Computing in Python. *Proceedings of the 9th Python in Science Conference*, 51-56.
  - **PIL:** Clark, F. (2015). Pillow (PIL Fork) Documentation.
  - **skimage:** van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., et al. (2014). scikit-image: image processing in Python. *PeerJ*.

Peer Collaborations

- This project is full of debates. The typical one is I had an argument with Zhongye Liu about the order of magnitude of MSE. My previous MSEs were very small in order of magnitude, less than a hundred. Liu's was very large, around a thousand. Later, after a long examination, we realized that the reason for the order of magnitude difference is the process of calculating MSE. I have directly performed the difference process on the pixel values. However, when the difference is negative, the pixel values overflow. For example, if A is 178 pixels and A' is 180 pixels, when we subtract A' from A, we want to get -2, but we actually get 253. This overflow of pixel values causes a deviation between the calculated MSE and the true value. After I translate the pixel values into int32, the problem was successfully solved.
- Likewise, I was guided by a friend on the algorithm. During cross-validation, Jingjing Feng helped me review the cross-validation design and pointed out that I hadn't chosen a different test set each time. With her help, I quickly changed the algorithm and got the correct results.

The End

Questions? Comments?