Implementation of the weighted nuclear norm minimization for image denoising

Andrei Kanavalau

Abstract—In this project the weighted nuclear norm minimization (WNNM) algorithm for image denoising is implemented in Python. Implementation choices are explained and most important components identified. Parameter tuning is performed. The performance of WNNM based denoising is compared to that achieved using Gaussian, Bilateral, and non-local means (NLM) filters. WNNM outperforms the other techniques including NLM approach based on both visual assessment and metric such as peak signal to noise ratio (PSNR) and structural similarity index (SSIM). A number of ways in which both computational time and denoising performance of the algorithm could be improved are identified. The code developed can be found at https://github.com/kanavalau/EE367_Project_WNNM_denoising

Index Terms—EE376, Computational Imaging, Weighted Nuclea Norm Minimization

1 Introduction

Image denoising is an import step of the image processing pipeline, which has a significant impact on any downstream uses ranging from simple consumer needs, people want to take visually pleasing photos in various conditions with practical devices, to object identification and localization in robotics, and information extraction in scientific imaging. As a result noise reduction remains an actively investigated problem in the field of computational imaging.

The task of image denoising is to find the best estimate of the true image, y, based on a noisy reading x. In this case the image formation model can be written simply $x = y + \eta$ as where η is the random noise. The sources of noise in imaging include but are not limited to amplification, read, and shot noise. With the exception of shot noise the noise is independent of the signal and can be approximated to follow a zero mean Gaussian distribution. Shot noise follows a Poisson distribution. In this project we concentrate on reducing Gaussian noise as it is more commonly observed.

The aim of this project is to implement a state of the art technique based on weighted nuclear norm minimization. Specifically the algorithm detailed in [1] is implemented. Significant components of the approach are not explicitly detailed in the paper and are experimented with. Performance of the final algorithm is compared to that achieved using Gaussian, Bilateral, and non-local means (NLM) filters both visually and on the basis of two numerical metrics: peak signal to noise ratio (PSNR) and structural similarity index (SSIM). Further suggestions on how the method could be improved are also discussed.

2 Related Work

All noise removal techniques estimate the true pixel intensities by taking into account information from across the image. The simplest methods such as Gaussian or median filtering rely on just a small area around each pixel and assume that the intensities of the surrounding pixels are representative of the central pixel. This would be expected to work well on uniform patches in images but is clearly going to lead to blurring of edges and other sharp features. The simplest improvement comes in the form of bilateral filtering, which introduces another simple idea that if the intensity of a nearby pixel is significantly different from the central pixel its contribution should be weighted down.

Bilateral filtering, although better at preserving edges still only relies on a small area around each pixel. A big improvement is achieved by using non-local methods where similarity of the area around the pixel to other parts of the image can be used for noise reduction. One of the simplest methods using this approach is the non-local means (NLM) [2]. Current state of the art approaches such as block matching and 3D filtering [3] and weighted nuclear norm minimization (WNNM) [1] are non-local methods.

As in many other areas of computational imaging high performance in image denoising has been achieved using convolutional neural networks (CNNs) in recent years [4]. Specifically referred to as denoising convolutional neural networks (DnCNNs) they are usually trained on the sets of noisy and clean images to predict the noise. In applications the output of the network corresponding to the noisy image is subtracted from the said image to obtain an estimate of the clean image.

3 Methods

In this section the theory of WNNM is first presented followed by an overview of implementation details. Other denoising methods used in the project are then formulated and comparison metrics used are presented.

3.1 Weighted nuclear norm minimization

3.1.1 Theory

Using x to denote the true clean image and η additive Gaussian white noise with zero mean and variance σ^2 the image formation model can be written as

$$y = x + \eta. \tag{1}$$

WNNM approach starts by going around the image in a sliding window fashion identifying central patches y_j and finding similar patches in a search window around the central patch. The patches are then stacked in a matrix Y_j with each column corresponding to a patch. The relationship between the noisy patch matrix and the clean one is given by

$$Y_j = X_j + N_j. \tag{2}$$

The idea of weighted nuclear norm minimization is that X_j should be approximately low rank with just a small number of singular values capturing the true low dimensional structure of the image and the rest being much smaller or zero. Can expect to obtain a good estimate of the true image by solving the following optimization problem

$$\text{minimize}_{X_j} \frac{1}{\sigma^2}||Y_j - X_j||_F^2 + ||X_j||_{w,*}. \tag{3}$$

The first term of the objective $||\cdot||_F$ is the Frobenius norm given by the sum of squares of all the entries of the matrix. For a matrix A with A_{ij} representing the entry in row i column j it is given by

$$||A||_F = \sqrt{\sum_i \sum_j A_{ij}^2}. \tag{4}$$

This term ensures similarity between the noisy patches and the denoised estimate. The second term is the weighted nuclear norm, which is the sum of singular values weighted by a vector w. Using $\sigma_i(A)$ and w_i to represent the ith singular value of A and the ith entry of w respectively the weighted nuclear norm can be written as

$$||A||_{w,*} = \sum_i w_i \sigma_i(A). \tag{5}$$

This term drives the estimate of the clean patches to be low rank. The regularization parameter between the two terms is chosen to be the variance of the noise in the image. The optimization problem given in equation (3) is solved by first performing singular value decomposition (SVD) on Y_j such that

$$Y_j = U\Sigma V'. \tag{6}$$

The estimate of the clean patch $\hat{X}_j$ is obtained by soft-thresholding the singular values according to

$$\mathcal{S}_w(\Sigma) = \max\{\Sigma_{ii} - w_i, 0\} \tag{7}$$

and setting the result as the singular values of $\hat{X}_j$ giving

$$\hat{X}_j = U\mathcal{S}_w(\Sigma)V'. \tag{8}$$

The weights are chosen such that larger singular values corresponding to structure in the patches are scaled down less than the small singular values likely corresponding to the noise. The ith weight is given by

$$w_i = \frac{c\sqrt{n}}{\sigma_i(X_j) + \epsilon} \tag{9}$$

where c is a parameter to be specified, n is the number of similar patches (i.e. number of columns in Y_j) and ϵ a small number to avoid division by 0. The only issue remaining is that $\sigma_i(X_j)$ cannot be computed directly. The solution is to initially estimate them using

$$\hat{\sigma}_i(X_j) = \sqrt{\max\{\sigma_i^2(Y_j) - n\sigma^2, 0\}} \tag{10}$$

and then compute equations (8) and (9) iteratively with $\hat{\sigma}_i(X_j) = \sigma_i(\hat{X}_j)$.

By repeating the above procedure for every patch an estimate of the clean image $\hat{x}$ can be obtained. A number of iterations can be performed to further improve the final output. The overall procedure is summarized in algorithm 1.

Algorithm 1 Image Denoising by WNNM [1]

Input: Noisy image y
  Initialize $\hat{x}^{(0)} = y$, $y^{(0)} = y$
  for $k = 1 : K$ **do**
    Interative regularization $y^{(k)} = \hat{x}^{(k-1)} + \delta(y - \hat{x}^{(k-1)})$
    for each patch y_j in $y^{(k)}$ **do**
      Find similar patch group Y_j
      Estimate weight vector w
      Singular value decomposition $Y_j = U\Sigma V'$
      Get the estimation: $\hat{X}_j = U\mathcal{S}_w(\Sigma)V'$
    end for
    Aggregate $\hat{X}_j$ to form the clean image $\hat{x}^{(k)}$
  end for
Output: Clean image $\hat{x}^{(K)}$

3.1.2 Implementation

WNNM for denoising is implemented in Python. The above section presents the denoising algorithm as specified in [1], which leaves a number of steps under-specified. This section details the specific choices made in this project. First is the procedure for identifying similar patches. Based on [3] mean squared error between patches was chosen to be the distance metric between different patches as

$$d_{\text{WNNM}}(y_i, y_j) = \frac{||y_i - y_j||_2^2}{m^2} \tag{11}$$

where m is the width of each patch. Two methods for choosing which patches should be included in Y_j were investigated. For the first one the distance metric was thresholded such that y_i is included in Y_j if $d_{\text{WNNM}}(y_i, y_j) \leq \tau$. For the second one a specified number n of patches with smallest distances to the central patch were included. It was found that th e latter approach achieves more consistent denoising performance.

Another decision regarded the number of iterations performed when estimating $\hat{\sigma}_i(X_j)$. It was found that good results are obtained with just one iteration. Slight improvement is achieved by increasing the number of iterations and 3 was chosen as a result.

No specific details on how X_j is used to form $\hat{x}$ are provided. One option would be to just use the first column corresponding to the central patch. At the same time all column could be used to reconstruct the patches they correspond to. In both cases the number of times each pixel has been contributed to from different patches needs to be kept track of and averaged at the end. In this project we only had time to implement the first approach.

Two methods of performing singular value decomposition were also investigated. In the first one SVD was applied directly to Y_j while in second one the columns of Y_j were first centered by subtracting the means. The means were then added back when reconstructing $\hat{X}_j$. Slight improvement in performance was achieved using the latter approach.

Finally it was observed that after a number of outer iterations the intensities in $\hat{x}^{(K)}$ would noticeably exceed the maximum intensity. The first approach was to normalize the output by the maximum value observed. The entries of each $\hat{X}_j$ were later clipped to be between minimum and maximum intensity values achieving significant improvements in denoising performance.

3.2 Other denoising methods used

3.2.1 Gaussian

Gaussian filter estimates intensity of a pixel in the clean image by a taking weighted average of the noisy pixels in a patch around the pixel. Using $I_n(z)$ to denote pixel intensity in the noisy image at location z and z_j to denote the location of the central pixel in noisy patch y_j, the estimate of the intensity of the denoised central pixel $\hat{I}_d(z_j)$ is given by

$$\hat{I}_d(z_j) = \frac{\sum_{z \text{ in } y_j} I_n(z) W_{\mathrm{G}}(z, z_j)}{\sum_{z \text{ in } y_j} W_{\mathrm{G}}(z, z_j)} \tag{12}$$

where the Gaussian weight is given by

$$W_{\mathrm{G}}(z, z_j) = \exp\left(-\frac{||z - z_j||_2^2}{2\sigma_{\mathrm{G}}^2}\right) \tag{13}$$

and σ_{G} and the patch size are tunable parameters.

3.2.2 Bilateral

Bilateral filtering modifies the Gaussian filtering procedure by introducing a weight corresponding to how similar the pixel intensities are. The equation is

$$\hat{I}_d(z_j) = \frac{\sum_{z \text{ in } y_j} I_n(z) W_{\mathrm{B}}(z, z_j)}{\sum_{z \text{ in } y_j} W_{\mathrm{B}}(z, z_j)} \tag{14}$$

where the Bilateral weight is given by

$$W_{\mathrm{B}}(z, z_j) = \exp\left(-\frac{||z - z_j||_2^2}{2\sigma_{\mathrm{B},1}^2}\right) \exp\left(-\frac{(I_n(z) - I_n(z_j))^2}{2\sigma_{\mathrm{B},2}^2}\right) \tag{15}$$

with $\sigma_{\mathrm{B},1}$, $\sigma_{\mathrm{B},2}$, and the patch size as tunable variables.

3.2.3 Non-local means

Non-local means extends the approach from comparing single pixels to comparing patches and weighing contribution from central pixels in other patches based on their similarity. Using SW to refer to the search window

$$\hat{I}_d(z_j) = \frac{\sum_{y_i \text{ in } SW} I_n(z_i) W_{\mathrm{NLM}}(y_i, y_j)}{\sum_{y_i \text{ in } SW} W_{\mathrm{NLM}}(y_i, y_j)} \tag{16}$$

with the NLM weight between two patches given by

$$W_{\mathrm{NLM}}(y_i, y_j) = \exp\left(-\frac{||y_i - y_j||_{2,\sigma_{\mathrm{NLM},2}}^2}{2\sigma_{\mathrm{NLM},1}^2}\right). \tag{17}$$

The term in the numerator of the exponent is the two norm weighted by a Gaussian kernel with standard deviation $\sigma_{NLM,2}$ thus giving higher weight to pixels closer to the center of patches. The tunable parameters in the NLM algorithm are the patch size, the search window size, and the two standard deviation parameters $\sigma_{NLM,1}$, $\sigma_{NLM,2}$.

3.3 Metrics

In this project two metrics are used for comparing the performance of different denoising techniques: peak signal to noise ratio (PSNR) and structural similarity index (SSIM). With x and y representing the vectorized clean and noisy images respectively the metrics are given by

$$PSNR(x, y) = 20 \log_{10}\left(\frac{\max\{x\}}{||x - y||_2^2}\right), \tag{18}$$

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}, \tag{19}$$

where μ_x and μ_y are the means of x and y respectively. Similarly σ_x^2 and σ_y^2 are the variances of x and y while σ_{xy} is the covariance. The two constants are set to $c_1 = (0.01L)^2$ and $c_2 = (0.03L)^2$ where L is the dynamic range.

4 Results

4.1 Parameter tuning

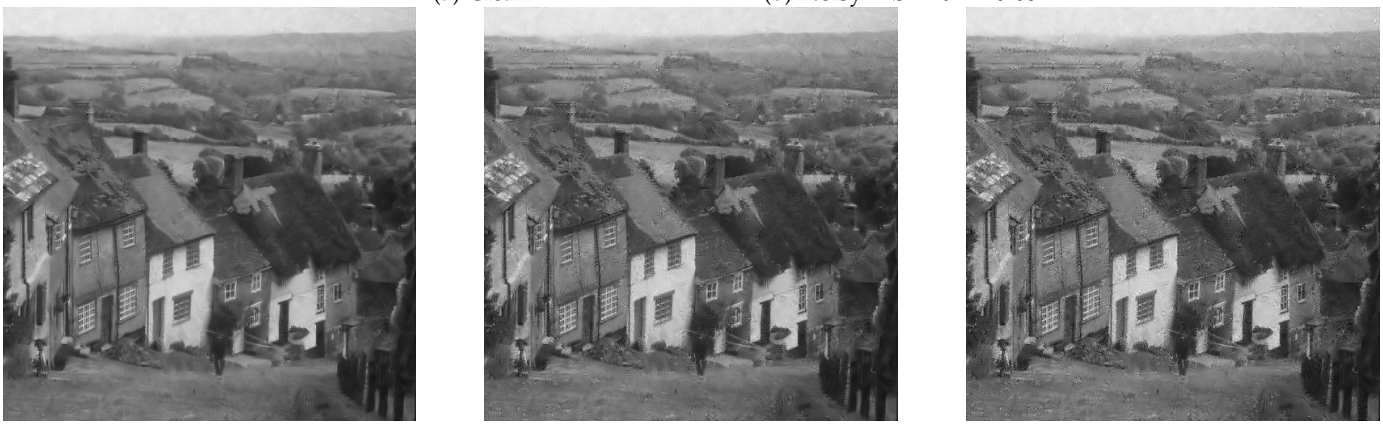
The results are obtained by applying denoising to images with zero mean Gaussian noise with $\sigma = 0.1$. For WNNM a number of parameters are available for tuning to improve performance namely the weights constant c, patch and search window sizes, number of similar patches n, regularization constant δ, and the number of outer iterations K. Patch and search window sizes are limited by computational time considerations and were chosen to be 6×6 and 18×18 respectively. The remaining 4 parameters were tuned in pairs. First $K = 3$ and $\delta = 0.1$ were fixed while c and n were varied. The results are shown in figure 1. Based on the results $c = 0.005$ and $n = 50$ were chosen. With c and n fixed K and δ were varied with the results shown in figure 2. It can be seen that $K = 3$ and $\delta = 0.1$ achieve optimal results. Note that different random noise is introduced in the two tuning runs.

For Gaussian, Bilateral, and NLM filter parameters that achieve the best performance on a set of 15 grayscale images are chosen namely $\sigma_{\mathrm{G}} = \sigma_{\mathrm{B},1} = \sigma_{\mathrm{NLM},1} = 0.9$, $\sigma_{\mathrm{B},2} = 0.4$, and $\sigma_{\mathrm{NLM},2} = 0.1$. For NLM the same patch and search window size as for WNNM are used and the remaining two parameters are specified based on the results .

4.2 Comparison of denoising techniques

In this section we compare the denoising performance of the four methods. Figures 3 and 4 show sample images denoised using different techniques with PSNR and SSIM values in the captions. As expected NLM and WNNM approaches achieve superior visual quality when compared to Gaussian and Bilateral filters. When compared to NLM, WNNM seems to be better at discerning edges and gradients while somewhat struggling with uniform patches.

(a) Clean (b) Noisy. $PSNR = 20.09$



(c) WNNM denoised with $c = 0.01$ and $n = 20$. $PSNR = 29.07$

(d) WNNM denoised with $c = 0.005$ and $n = 20$. $PSNR = 27.55$

(e) WNNM denoised with $c = 0.001$ and $n = 20$. $PSNR = 26.22$

(f) WNNM denoised with $c = 0.01$ and $n = 50$. $PSNR = 29.36$

(g) WNNM denoised with $c = 0.005$ and $n = 50$. $PSNR = \mathbf{29.41}$

(h) WNNM denoised with $c = 0.001$ and $n = 50$. $PSNR = 29.39$

(i) WNNM denoised with $c = 0.01$ and $n = 100$. $PSNR = 29.10$

(j) WNNM denoised with $c = 0.005$ and $n = 50$. $PSNR = 29.17$

(k) WNNM denoised with $c = 0.001$ and $n = 50$. $PSNR = 29.21$

Fig. 1: WNNM results for different values of parameters c and n with $K = 3$ and $\delta = 0.1$

(a) Clean

(b) Noisy. $PSNR = 20.10$

(c) WNNM denoised with $K = 1$. $PSNR = 24.29$

(d) WNNM denoised with $K = 3$ and $\delta = 0.05$. $PSNR = 29.32$

(e) WNNM denoised with $K = 3$ and $\delta = 0.1$. $PSNR = \mathbf{29.37}$

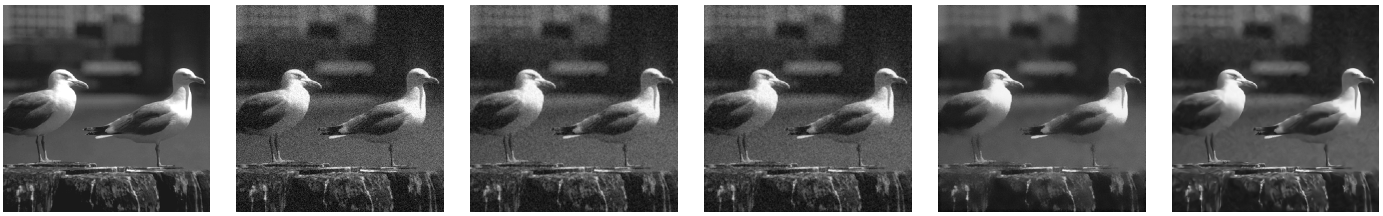
(f) WNNM denoised with $K = 3$ and $\delta = 0.25$. $PSNR = 29.37$

(g) WNNM denoised with $K = 5$ and $\delta = 0.05$. $PSNR = 28.53$

(h) WNNM denoised with $K = 5$ and $\delta = 0.1$. $PSNR = 28.74$

(i) WNNM denoised with $K = 5$ and $\delta = 0.25$. $PSNR = 29.24$

Fig. 2: WNNM results for different values of parameters K and δ with $c = 0.005$ and $n = 50$



(a) Clean | (b) Noisy $PSNR = 20.31$ $SSIM = 0.39$ | (c) Gaussian filter $PSNR = 28.67$ $SSIM = 0.83$ | (d) Bilateral filter $PSNR = 28.38$ $SSIM = 0.81$ | (e) NLM $PSNR = 29.49$ $SSIM = 0.88$ | (f) WNNM $PSNR = 31.70$ $SSIM = 0.93$

Fig. 3: Clean, noisy (zero mean Gaussian noise with $\sigma = 0.1$), and denoised images produced using the different methods.

(a) Clean | (b) Noisy $PSNR = 20.64$ $SSIM = 0.51$ | (c) Gaussian filter $PSNR = 26.37$ $SSIM = 0.78$ | (d) Bilateral filter $PSNR = 26.62$ $SSIM = 0.78$ | (e) NLM $PSNR = 26.45$ $SSIM = 0.78$ | (f) WNNM $PSNR = 27.43$ $SSIM = 0.81$

Fig. 4: Clean, noisy (zero mean Gaussian noise with $\sigma = 0.1$), and denoised images produced using the different methods.

TABLE 1: Average PSNR and SSIM values achieved when denoising is applied to a set of 15 images with zero mean Gaussian noise $\sigma = 0.1$.

	PSNR	SSIM
Noisy	20.22	0.49
Gaussian	26.36	0.80
Bilateral	26.61	0.80
NLM	26.97	0.82
WNNM	28.40	0.86

On the basis of the two metrics used WNNM comfortably outperforms the other approaches.

Table 1 shows the average PSNR and SSIM values obtained when different denoising techniques are applied to a set of 15 grayscale test images. The average performance obtained across different techniques is as expected with WNNM coming out on top, followed by NLM, Bilateral, and Gaussian filters.

4.3 Discussion of possible improvements

One limitation of the approach that has not been investigated in this work is the importance of knowing the variance of the noise as perfect specification has been provided to the algorithm so far. This is something that would need to be addressed when the algorithm is applied to real noisy images.

A number of aspects of the implementation could be improved. The algorithm could be vectorized in order to reduce the required computational time, which currently stands at around 2 minutes per outer loop iteration. Furthermore better performance could potentially be achieved by modifying the way similar patches are identified. Currently the distance metric is computed on the noisy patches while what is actually of interest is the distance between clean patches. An intermediate denoising algorithm could be added such as simple low pass filtering of the patches before the distance is computed to improve the similar patch identification process. Finally more information could be extracted from the estimated matrix of clean patches $\hat{X}_j$ by for example incorporating the obtained clean estimates of all the similar patches into their original locations weighted by the distance to the central patch.

5 Conclusions and Further Work

In this work the WNNM algorithm was implemented for image denoising. Details of the implementation were highlighted and discussed. Algorithm parameters were tuned and the method was compared to other techniques investigated in the class namely Gaussian, Bilateral, and non-local means filters. Sample images for visual assessment of denoising performance were presented. PSNR and SSIM metrics were computed across a set of 15 denoised greyscale images. The results demonstrated significant improvement achieved using WNNM over other approaches tested. A number of possible further modifications to the algorithm were also discussed.

References

[1] S. Gu, L. Zhang, W. Zuo, and X. Feng, "Weighted nuclear norm minimization with application to image denoising," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[2] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2, 2005, pp. 60–65 vol. 2.

[3] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering," *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2080–2095, 2007.

[4] V. Jain and S. Seung, "Natural image denoising with convolutional networks," in *Advances in Neural Information Processing Systems*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds., vol. 21. Curran Associates, Inc., 2008.