Assignment 4 - Advanced Image Processing

Kartik Gokhale, Pulkit Agarwal

March 2022

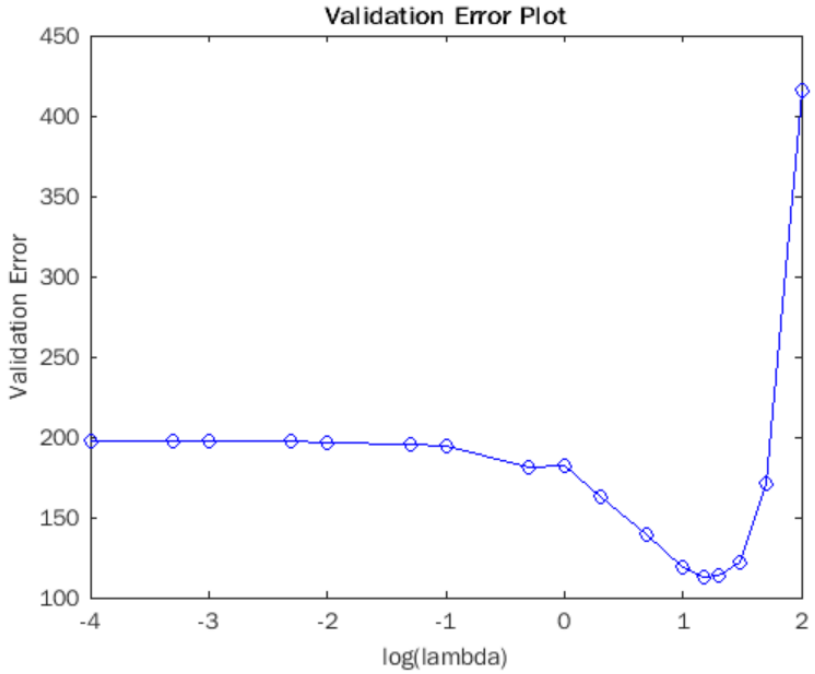
Contents

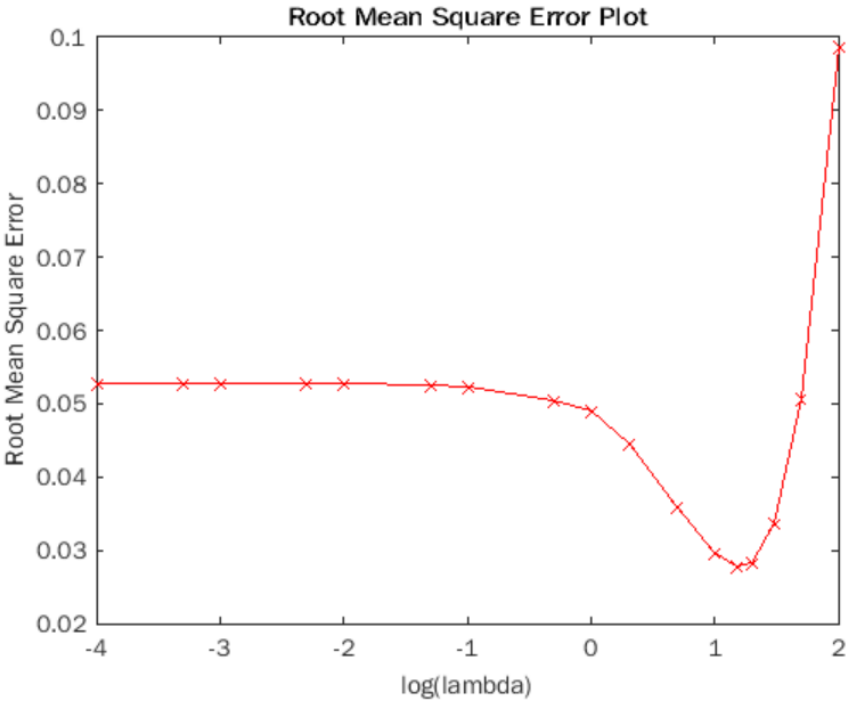
Note - To obtain the results reported in this document, run the MATLAB files specified alongside the section headers above.

1 Problem 1 - Q1.m

1.1 Part A

The two plots for VE and RMSE are obtained as follows:





Note that both the plots are initially nearly constant, i.e. the error is not depending on λ to a great extent when λ is very small. Eventually, as we reach close to $\lambda = 1$, the error starts decreasing, reaches a minima, and then increases with a large slope. Thus, except for the difference in scale, both the plots have almost the same shape of the curve. In fact, the optimal value of λ obtained from both the plots is 15.

1.2 Part B

Suppose the two sets, reconstruction set R and validation set V, are the same. Then the reported error is simply $\frac{1}{|R|}||y(R) - \Phi(R)x_g||^2$, where x_g is the minimizer to $||y(R) - \Phi(R)x||^2 + \lambda||x||_1$. Since $y = \Phi x + \eta$, so we have $\Phi(R)(x_g - x) = \eta(R)$, which is fixed. This means that the rows of Φ corresponding to set R will not be able to give any information about $||x_g - x||_2$. So the error term computed using cross validation with $R = V$ will not provide any useful data in terms of similarity to the actual error term.

1.3 Part C

Theorem 1 in the paper gives an estimate of the relation between VE and RMSE, which holds with a sufficiently high probability. In particular, it says that when the number of elements in the validation set V is large enough, RMSE is bounded by

$$\text{RMSE} \in [h(\lambda, +) \times VE - \sigma_n^2, h(\lambda, -) \times VE - \sigma_n^2]$$

with a probability that depends on λ. As $|V|$ increases, the difference between the upper bound and the lower bound becomes tighter. Thus, we can estimate RMSE with VE.

1.4 Part D

The theorem stated that setting $\lambda = 2\sigma\sqrt{\tau \frac{\log p}{N}}$ for some $\tau > 2$ will be a valid choice with a probability at least $\left(1 - 2e^{-\frac{1}{2}(\tau-2)\log p}\right)$. Note that, the theorem had derived this value as a lower bound for λ, unlike the λ found using cross validation, which is an exact value. Also, the λ above is set in a general environment, and does not depend on the particular use-case. When using cross validation, we employ some part of the dataset itself to get an optimal λ. This ensures that the found λ gives the best possible results for this particular compressed sensing setting.

2 Problem 2

2.1 Part A

Consider a dictionary D. Suppose we have a vector v represented as

$$v = \sum_i k_i D_i$$

where D_i represents the i^{th} column(element) of the dictionary. Now, we apply a known derivative filter to v, say $d(v)$. Thus, we obtain

$$d(v) = d(\sum_i k_i D_i)$$

However, we know derivatives are linear, that is $d(f_1 + f_2) = d(f_1) + d(f_2)$, where f_1, f_2 are vectors. Hence, we have

$$d(v) = \sum_i k_i d(D_i)$$

which is the linear combination of elements of a new dictionary, with elements as the same derivative filter applied to the elements of the original dictionary. Thus, we obtain the new dictionary by applying the same derivative filter to the elements of the original dictionary.

2.2 Part B

Consider a dictionary D. Suppose we have a vector v represented as

$$v = \sum_i k_i D_i$$

where D_i represents the i^{th} column(element) of the dictionary. Now, we consider the subset which was rotated by angle α. Let the rotation transformation be represented as $R_\alpha(v)$ which rotates a vector v by angle α. However, we know rotations are linear, that is $R_\alpha(f_1 + f_2) = R_\alpha(f_1) + R_\alpha(f_2)$, where f_1, f_2 are vectors. Hence, we have

$$R_\alpha(v) = \sum_i k_i R_\alpha(D_i)$$

which is the linear combination of elements of a new dictionary, say D_1, with elements as the same rotation transform of angle α applied to the elements of the original dictionary We know all elements rotated by α can be expressed by this dictionary. We do the same for the other subset. Let the rotation transformation be represented as $R_\beta(v)$ which rotates a vector v by angle β

$$R_\beta(v) = \sum_i k_i R_\beta(D_i)$$

which is the linear combination of elements of a new dictionary, say D_2, with elements as the same rotation transform of angle β applied to the elements of the original We know all elements rotated by β can be expressed by this dictionary.
And thus, all elements can be expressed by the joint dictionary $D_1 \cup D_2$. Hence, we obtain the new dictionary by applying both angle transforms to each dictionary element and then taking a union of the resulting dictionaries obtained.

2.3 Part C

Consider a dictionary D. Suppose we have a vector v represented as

$$v = \sum_i k_i D_i$$

where D_i represents the i^{th} column(element) of the dictionary. Now, we consider the provided quadratic transformation

$$I_{new}(x, y) = \alpha(I_{old}(x, y))^2 + \beta I_{old}(x, y) + \gamma$$

Applying the quadratic transformation to the vector v

$$I_{new}(v) = \alpha(\sum_i k_i D_i)^2 + \beta(\sum_i k_i D_i) + \gamma$$

Where we consider element wise operations in the above. Opening the square term, we get:

$$I_{new}(v) = \alpha(\sum k_i^2 D_i^2 + 2\sum_{i \neq j} k_i k_j D_i D_j) + \beta(\sum k_i D_i) + \gamma$$

There are 4 terms in the above equation. The first term of every transformed vector can be written as a linear combination of each dictionary column multiplied with itself element-wise(say D_1). The second term of every transformed vector can be written as a linear combination of each pair of distinct dictionary columns multiplied with each other element-wise(say D_2). The third term is a scale and thus, a linear transform represented by the original dictionary(D itself). The final term is a constant term and can be written as a linear combination of the columns of a unit matrix of same dimension as the vector(say D_3). Thus, our new dictionary is obtained on concatenation Thus,

$$\boldsymbol{D_{new} = [D_1|D_2|D_3|D]}$$

2.4 Part D

Consider a dictionary D. Suppose we have a vector v represented as

$$v = \sum_i k_i D_i$$

where D_i represents the i^{th} column(element) of the dictionary. Now, we apply a blur kernel to v, say $b(v)$. Thus, we obtain

$$b(v) = b(\sum_i k_i D_i)$$

However, we know blur kernels are linear, that is $b(f_1 + f_2) = b(f_1) + b(f_2)$, where f_1, f_2 are vectors. Hence, we have

$$b(v) = \sum_i k_i b(D_i)$$

which is the linear combination of elements of a new dictionary, with elements as the same blur kernel applied to the elements of the original dictionary. Thus, we obtain the new dictionary by applying the same blur kernel to the elements of the original dictionary.

2.5 Part E

Consider a dictionary D. Suppose we have a vector v represented as

$$v = \sum_i k_i D_i$$

where D_i represents the i^{th} column(element) of the dictionary. Now, we apply a linear combination of blur kernels to v, say $b(v)$. Thus, we obtain

$$b(v) = b(\sum_i k_i D_i)$$

However, we know linear combinations of blur kernels are linear, that is $b(f_1 + f_2) = b(f_1) + b(f_2)$, where f_1, f_2 are vectors. Hence, we have

$$b(v) = \sum_i k_i b(D_i)$$

which is the linear combination of elements of a new dictionary, with elements as the same linear combination of blur kernels applied to the elements of the original dictionary. However, we need to represent the linear combination of the blur kernels. For this, we construct a dictionary to represent the linear combination of the blur kernels of the set B. Consider the set of dictionaries D^i such that, D^i model linear combinations of blur kernels applied to the i^{th} column of the original dictionary.

$$D^i = [b^k(D_k)]$$

where b^k refers to the k^{th} blur kernel in set B.
Hence, we obtain the new dictionary as the dictionary $[D^i]$ where each D^i is another dictionary obtained on applying each blur kernel in set B to a column in D. Thus, our new dictionary is each blur kernel applied to each element of D.

2.6 Part F

Consider a dictionary D. Suppose we have a vector v represented as

$$v = \sum_i k_i D_i$$

where D_i represents the i^{th} column(element) of the dictionary. Let the Radon transformation be represented as $R_\theta(v)$. However, we know Radon transforms are linear, that is $R_\theta(f_1 + f_2) = R_\theta(f_1) + R_\theta(f_2)$, where f_1, f_2 are vectors. Hence, we have

$$R_\theta(v) = \sum_i k_i R_\theta(D_i)$$

which is the linear combination of elements of a new dictionary, say D_1, with elements as the same Radon transform applied to the elements of the original dictionary.

2.7 Part G

Consider a dictionary D. Suppose we have a vector v represented as

$$v = \sum_i k_i D_i$$

where D_i represents the i^{th} column(element) of the dictionary. Now, we consider the subset which was translated by displacement (x_1, y_1). Let the translation transformation be represented as $T_{(x_1,y_1)}(v)$ which translates a vector v by (x_1, y_1). However, we know translations are linear, that is $T_{(x_1,y_1)}(f_1 + f_2) = T_{(x_1,y_1)}(f_1) + T_{(x_1,y_1)}(f_2)$, where f_1, f_2 are vectors. Hence, we have

$$T_{(x_1,y_1)}(v) = \sum_i k_i T_{(x_1,y_1)}(D_i)$$

which is the linear combination of elements of a new dictionary, say D_1, with elements as the same translation transform of (x_1, y_1) applied to the elements of the original dictionary We know all elements translated by (x_1, y_1) can be expressed by this dictionary. We do the same for the other subset. Let the translation transformation be represented as $T_{(x_2,y_2)}(v)$ which rotates a translates v by (x_2, y_2). Similarly, as done above,

$$T_{(x_2,y_2)}(v) = \sum_i k_i T_{(x_2,y_2)}(D_i)$$

which is the linear combination of elements of a new dictionary, say D_2, with elements as the same translation by (x_2, y_2) applied to the elements of the original We know all elements translated by (x_2, y_2) can be expressed by this dictionary.
And thus, all elements can be expressed by the joint dictionary $D_1 \cup D_2$, taking into account the padding. Hence, we obtain the new dictionary by applying both translation transforms to each dictionary element and then taking a union of the resulting dictionaries obtained., equalising their size by adding zero padding.

3 Problem 3

3.1 Low Rank Approximation

The goal is to solve for the minimum of the objective function

$$J(A_r) = ||A - A_r||_F^2$$

where A is a known $m \times n$ matrix of rank greater than r, and A_r is a rank-r matrix, where $r < m, r < n$.
We invoke the Eckart–Young–Mirsky Theorem which states a useful property of the Singular Value Decomposition(SVD) of a matrix.
The theorem says that the solution to the above optimization problem is given by

$$A_r = \sum_{i=1}^{r} S_{ii} u_i v_i^T$$

where S_{ii} capture the r-largest singular values of the matrix A
An outline of the idea is that

$$||A - A_r||_F^2 = \sum_{i=r+1}^{n} \sigma_i^2$$

which implies that we must minimize the singular values of the residual matrix, thus, we use the top r-singular values and their corresponding vectors to construct the best low rank approximation of A.
The main purpose of **Image Inpainting** is to repair the damaged image or remove the unwanted objects in the image.**Image Inpainting** Algorithms are based on Low-Rank Approximation and Texture Direction. The first step of such algorithms is to decompose the image using low-rank approximation method.

3.2 Procrustes Problem

The goal is to solve for the minimum of the objective function

$$J(R) = ||A - RB||_F^2$$

This is the standard Procrustes problem. We note that

$$\begin{aligned} J(R) &= ||A - RB||_F^2 \\ &= trace((A - RB)^T (A - RB) \end{aligned}$$

Opening brackets an simplifying

$$= trace(A^T A + B^T B) - 2trace(A^T RB)$$

The first term is constant and the last term has a negative sign, thus we maximise the last term

$$= trace(A^T RB) = trace(RBA^T)$$

Consider SVD of $BA^T = USV^T$. We obtain the maximum value of the trace when we have

$$R = VU^T$$

A small caveat is as follows: If determinant of VU^T is negative. In this case,

$$R = V \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & -1 \end{bmatrix} U^T$$

This is used in **shape analysis** in images. Shapes are considered equivalent if there is a similarity transform connecting them. Rotation using an orthonormal rotation matrix is one such similartiy transform. Thus, we use this objective function for shape analysis and to analyse the mean and variation in the shape of objects in the real world such as that of the human hand in **medical imaging**

4 Problem 4

4.1 Details of the Paper

Title: **Structure-Preserving Color Normalization and Sparse Stain Separation for Histological Images**
Venue: IEEE Trans Med Imaging
Authors; Abhishek Vahadane, Tingying Peng, Amit Sethi, Shadi Albarqouni, Lichao Wang, Maximilian Baust, Katja Steiger, Anna Melissa Schlitter, Irene Esposito, Nassir Navab
Date: August, 2016
The paper can be accessed here.

4.2 Context for NMF

The Staining and scanning of tissue samples for microscopic examination is fraught with undesirable color variations. When comparing tissue samples, color normalization and stain separation of the tissue images can be helpful for both pathologists and software. Techniques that are used for natural images fail to utilize structural properties of stained tissue samples and produce undesirable color distortions. The stain concentration cannot be negative. Tissue samples are stained with only a few stains and most tissue regions are characterized by at most one effective stain. To model these physical phenomena that define the tissue structure, it is necessary to first decompose images into stain density maps that are sparse and non-negative. Non-negative matrix factorization (NMF) was used in one of the pioneering unsupervised frameworks for stain separation that derived image-specific stain color basis. The non-negative constraints on the stain density and color appearance matrix capture the important property of stains that can only absorb but not emit light, which makes both their color basis and density non-negative. It, therefore, solves the following problem:

$$\min_{W,H} \frac{1}{2}||V - WH||_F^2, W, H \geq 0$$

4.3 Solving the Problem

They first convert a given RGB image to optical density using Beer-Lambert law. Then, they add a sparseness constraint , i.e., we propose an improved NMF cost function for stain separation by including l_1 sparseness regularization on stain mixing coefficients H_j

$$\min_{W,H} \frac{1}{2}||V - WH||_F^2 + \lambda \sum_{j=1}^{r} ||H(j,:)||_1 W, H \geq 0$$

The joint non-convex optimization above is solved by alternating between W and H which optimizes one set of parameters while keeping the other fixed starting with an initialization of W by random elements from the training set (V). These two alternating steps are called sparse coding for H and dictionary learning for W, and resolved using NNF as mentioned above with the sparse constraint. The key step in the normalization scheme in method B is an accurate stain separation of both source and target images based on sparse regularized NMF

4.4 Dictionary Elements

Dictionary learning or estimating W is done by using parameter free block-coordinate descent with warm restart which does not require learning rate tuning and still guarantees convergence to a global optimum for convex optimization.
The dictionary represents the stain color appearance matrix whose columns represent color basis of each stain such that number of rows is the number of stains, Stain density maps can then be computed by multiplying the optical density with Moore-Penrose pseudoinverse of color appearance matrix W. This is the significance of the dictionary elements

5 Problem 5

NNSC lends itself naturally to Poisson denoising due to the non-negativity constraints – on measurements as well as data. We use the objective function as discussed in the lecture for Poisson denoising, with $W = -I_0 R$. Thus,

$$J(H) = \sum_{k=1}^{n}\sum_{l=1}^{N} -y_{kl} log(Wf_l)_k + (Wf_l)_k + \rho \sum_{l=1}^{n}\sum_{c=1}^{r} f_{cl}$$

We optimize the above for each f_l to obtain f.
To remove Gaussian noise in the signal, we could impose a MRF prior model on the data or penalise the difference between values at neighbouring data points, which is a standard technique for signal denoising. This would change the objective function to

$$J(H) = \sum_{k=1}^{n}\sum_{l=1}^{N} -y_{kl} log(Wf_l)_k + (Wf_l)_k + \rho \sum_{l=1}^{n}\sum_{c=1}^{r} f_{cl} + \sum_{|i-j|<1} |f_i - f_j|_1$$