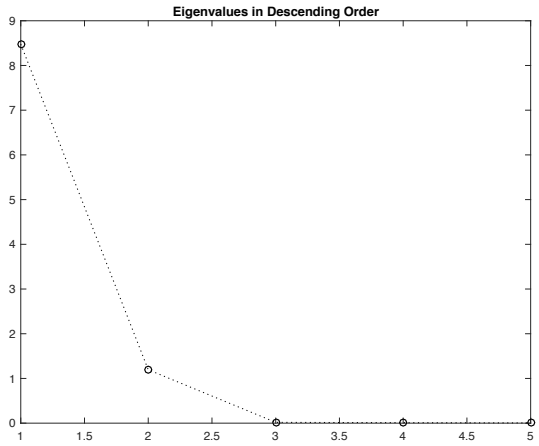
# Lab #3 Experiments & Analysis

1. Below is a dot plot of eigenvalues of the 195 * 195 covariance matrix and the corresponding PCA scores.




Since only the first two eigenvalues are nonzero, we only display the corresponding PCA scores of the three principal components, which are also coordinates of pixels, as images. The corresponding eigenvalues are: 8.476, 1.195, 0.018.

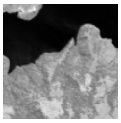


image 1

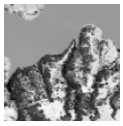


image 2

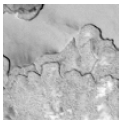


image 3

The contrasts in the first two images, which are constructed by the first two princial component, are the more salient, while the third image constructed by the third principal component is the least salient, mostly gray. The second image shows more differences (variations) in terms of light and dark, and I can see more details such as stripes and clearer edges. Also, there are clear boundries or edges between different “colors” in images 1 and 2 that are not observable in image 3.

Compared with the image made with RGB band (image 4, below), image 2 demonstrates most of the color compositions image 4 has according to my eyes, as the two images exhibit around the same level of details, although it seems that the image with RGB bands shows some more color variations in certain areas. Image 3 exhibits some color variation details of image 4 that are in neither image 2 nor image 1.

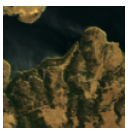


image 4

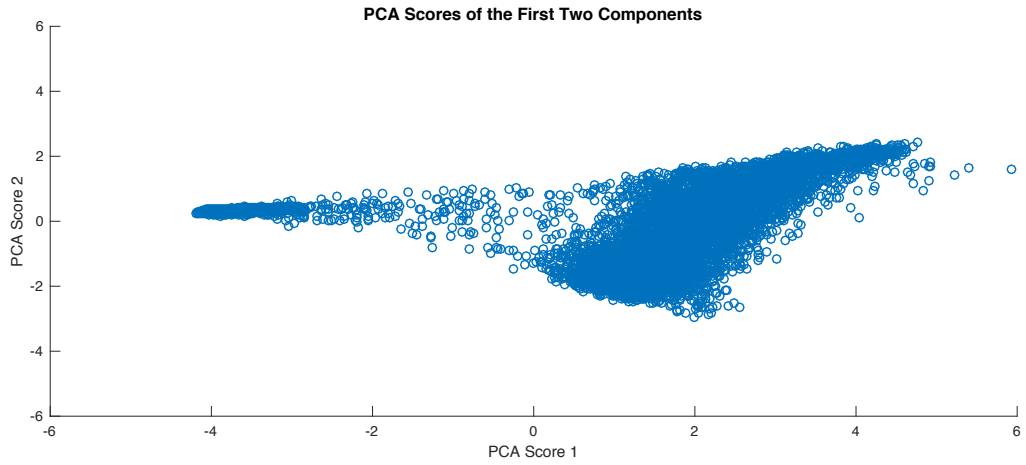
However, the images constructed by the last few eigenvectors cannot exhibit any details of the original image. Since the corresponding eigenvalues are approximately 0, the images constructed by those eigenvctors are noise and cannot capture the features of the original image.

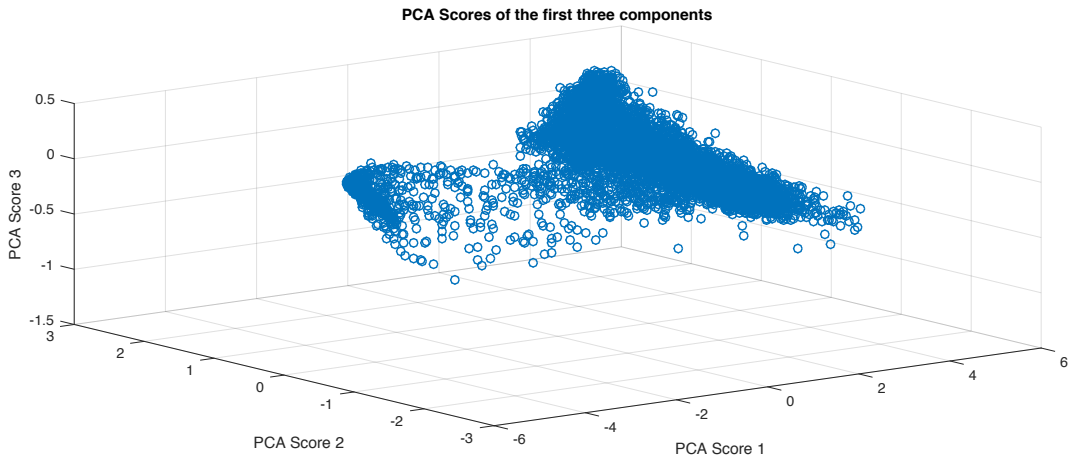


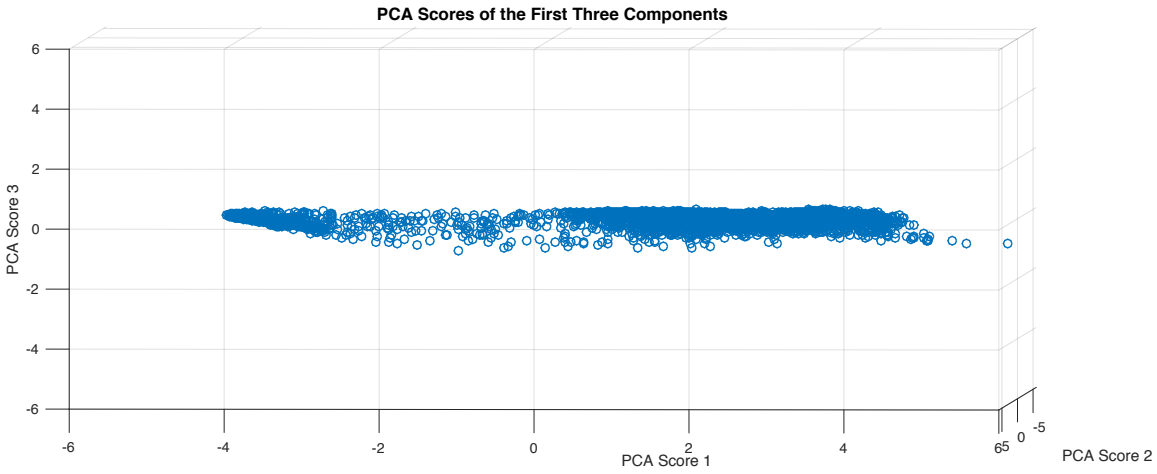
image 5 image 6 image 7

2. The subspace actually occupied by the data is 2-dimensional or 3-dimensional subspace.
    - Reason for 2-dimensional subspace: the eigenvalues plot shows that there are 2 non zero eigenvalues, which means that only the principal components corresponding to the two non zero eigenvalues can extract features of the original image. So it is reasonable to deduce that the subspace occupied by the data is 2-dimensional.
    - Reason for 3-dimensional subspace: each of the images 1-3 exhibits some distinct features of the image of RGB band, which I cannot find in the other two images. I also constructed an image using the eigenvector corresponding to the $4^{th}$ eigenvalue, but it is not very different from image 3.

3. Below are 2-D and 3-D scatter plots of PCA scores of the first 2 and 3 components:

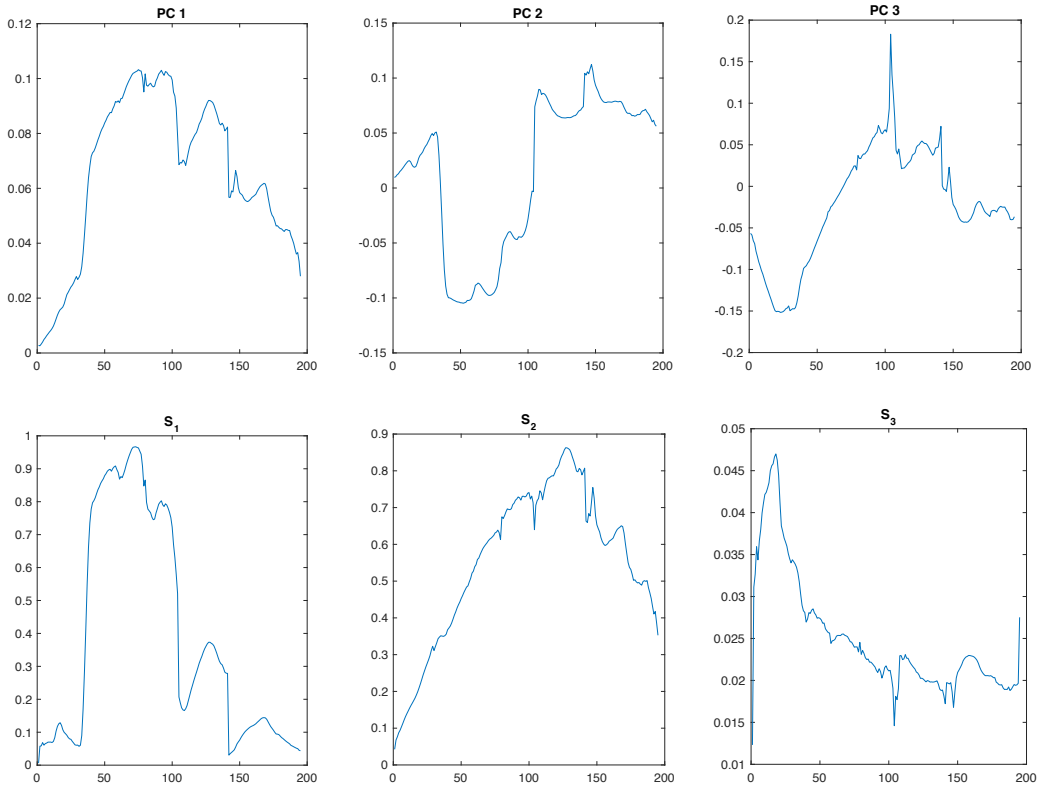





The 3-D scatter plots, especially the third one, shows that the shape of the data is still flat, which means that adding in the third component does not add much information on the z axis. Also, both 2-D and 3-D scatter plots exhibit two clusters of data points. Thus, the subspace occupied by the data is 2-dimensional.

4. Plots of the first three eigenvectors, and columns in matrix S (S1,S2,S3):




The plot of PC2 is similar to the flipped plot of S2, the second column of matrix S; the plot of PC3 is similar to the flipped plot of S3, the third column of matrix S. There are some similarities between the three plots of three components and columns in S. However, there are not enough evidence indicating that there are certain correlations between PCs and columns of S, so the plots of principal vectors are not physically interpretable.

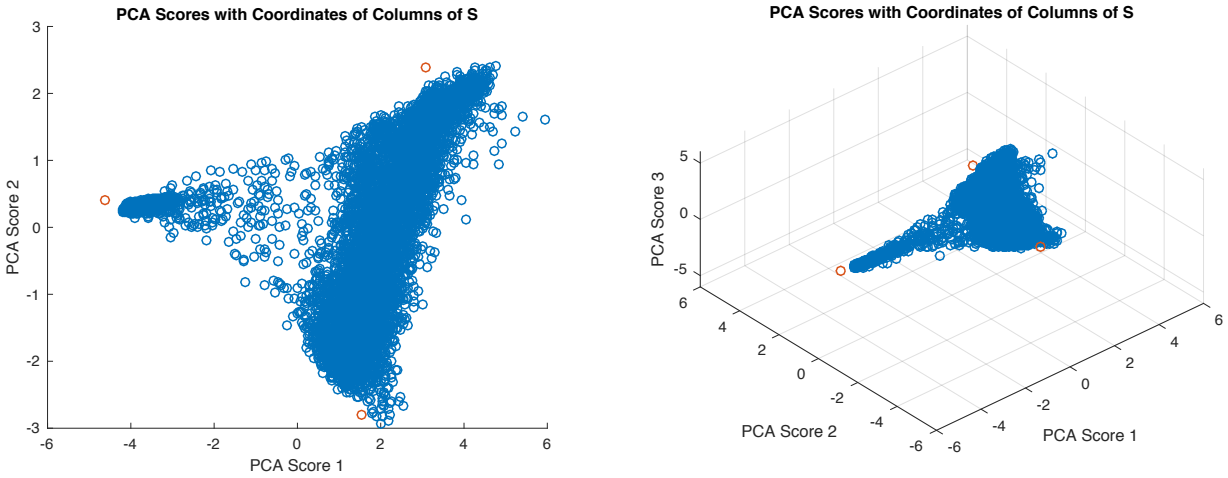
Each column of matrix S represents a material, and thus the plots of S1,S2,S3 are physically interpretable. But principal components are tools that extract salient features of the original data set, so they are more of a means to analyze the original data rather than being physically interpretable.

5. When the linear model X = SA + N holds, the actual subspace in which the data may theoretically lie is 3-dimensional. There are three column vectors in matrix S, each of which represents a material. Since all column vecotrs in S are linearly independent, the basis we use in our regression consists of three vectors, and thus spans a three dimensional subspace.
6. After performing an affine combination of the endmembers, we lose a dimension. After setting a 2 dimensional subspace, the third dimension is then fixed. Hence the data theoretically lie in a 2-dimensional subspace.
7. After performing convex combinations, we are getting a even smaller 2-dimensional subspace since there are more restrictions.
8. According to questions 5,6,7, if the restrictions in 6 or 7 are kept, the theoretical intrinsic dimensionality of the data is 2-dimensional; otherwise it is 3-dimensional. According to the theory part, we will perform minimization according to Eq. (5), i.e.

$$\arg\min_{\mathbf{A}} \frac{1}{2}||\mathbf{X} - \mathbf{SA}||_F^2$$
$$\text{s.t. } \mathbf{A}^\top \mathbb{1}_P = \mathbb{1}_N$$

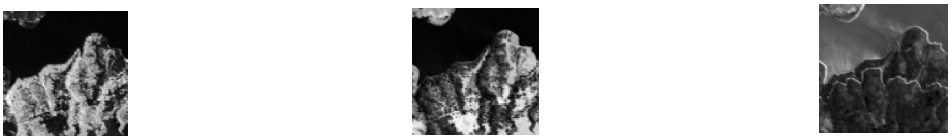
   the sum-to-one constraint is kept. Thus, we conclude that the theoretical intrinsic dimensionality of the data is 2-dimensional.

9. The scatter plots of PCA scores with coordinates of each column of S in the basis defined by PCA added:




   In both plots, we can see that coordinates of each column of S in the basis defined by PCA, which are denoted by the three red dots, locate in the three corners of all the data points. The location of the red dots makes sense because each of them represents a pure material; if some data points (blue) consist of only one material, they will locate in one of the corners of the triangle shaped scatter plot. The blue points not in the corner are linear combinations of the basis defined by PCA, which also means that the pixels consist of certain combination of the three materials. Thus, the PCA basis may be appropriate.

10. Images constructed by the coefficients of abundances without the sum-to-one constraint are the following:



Material 1 Material 2 Material 3

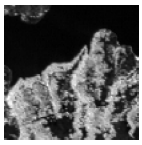
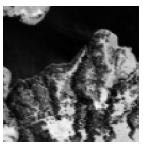
Through experiments, I found that if the value of certain pixel entry is greater than 1, the constructed image using this pixel is white. If the value of certain pixel entry is less or equal to 0, the corresponding image is black. Otherwise the image is gray. Therefore I have the following observations:

- The upper part of all of the three images is either black or gray (black in both material 1 and 2, gray in material 3). It doesn't make much sense to me – being black in both images of material 1 and 2 indicates that this part is doesn't contain any material 1 or 2, which means that it has to consist of 100% material 3 (it has to consist of a certain material), but it does not consist of 100% material 3 (since it is not white in material 3 image). This is a contradiction, which means that the proportions of the three materials in certain pixels do not add up to 1. This idea can be reflected by the (part of) actual matrix A we solved for unconstraied optimization problem:

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0.2811 | 0.2760 | 0.2435 | 0.4839 | 0.3496 |
| 2 | 0.5954 | 0.5874 | 0.6234 | 0.3425 | 0.4775 |
| 3 | 1.5209 | 1.4123 | 1.2886 | 0.7874 | 0.8702 |

- Some pixels (e.g. the edges) in images corresponding to material 1 and 2 are black, and they are white in images corresponding to material 3. This part makes sense because it indicates that the edge consist of 100% (or at least majority) of material 3. Also, some pixels in images Material 1 and 3 are black while being white in Material 2.

11. Images constructed by the coefficients of abundances with the sum-to-one constraint are the following:

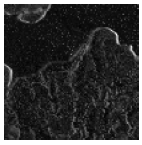
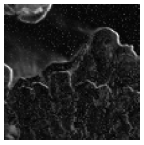




Material 1 Material 2 Material 3

Comparing with images in question 10, the color scaling here is more logical. The black area in both Material 1 and 2 becomes white in Material 3, indicating that these pixels consist of mostly material 3. The reason behind this improvement is the sum-to-one constraint, which enforces the sum of the propo-portions of each material to 1.

12. From question 8, K = 2, so we have X_proj 10201*2 matrix, S_proj 2*3 matrix. After performing the same calculations as part 11, the resulting A is a 3*10201 matrix. Using tic toc function, we know that it takes 0.003388 seconds solving for A using the projected X and projected S, which is much shorter than the 0.011342 seconds when performing the same computation using the original X and S.

13. RMSE's for unconstrained and constrained optimization display as:

The image to the left is RMSE without constrain while the one to the right is RMSE with constrain. Since RMSE measures the error/noise, we want it as small as possible, which means that the less clear the RMSE image or the less details the RMSE image gives, the better the approach is. Thus, since the image to the left has less details, the unconstrained approach gives better values.

Also, the mean RMSE for unconstrained approach is 0.0115, which is smaller than that for constrained approach is 0.0147. This also yields the conclusion that the unconstrained approach is better in terms of smaller errors or noise.

The reason why unconstrained approach is better is that (A) arg min of RMSE is the same as that of equation (3), i.e. the unconstrained approach. Adding in additional constraints will only lead to a higher RMSE. Also, the highest values of RMSE represents the most significant noise.

Matlab code:

1.

```
load('data_lab3.mat')
for i=1:195
    im_vec(:,i) = reshape(im(:,:,i),101.^2,1);
end

im_cov = cov(im_vec);
im_mean = mean(im_vec);
im_eig = eigs(im_cov,5);
%plot eigenvalues
figure
plot(im_eig,':ok')
title('Eigenvalues in Descending Order')

%eigenvectors, eigenvalues
[PCVec,PCVal]=eigs(im_cov,3);
%pca
PCScore1 = PCVec(:,1).'*((im_vec(1,:)-im_mean)).';
for i = 2:10201
    PCScore1 = [PCScore1;PCVec(:,1).'*((im_vec(i,:)-im_mean)).'];
end

PCScore2 = PCVec(:,2).'*((im_vec(1,:)-im_mean)).';
for i = 2:10201
    PCScore2 = [PCScore2;PCVec(:,2).'*((im_vec(i,:)-im_mean)).'];
end

PCScore3 = PCVec(:,3).'*((im_vec(1,:)-im_mean)).';
for i = 2:10201
    PCScore3 = [PCScore3;PCVec(:,3).'*((im_vec(i,:)-im_mean)).'];
end
%display the first 3 eigenvectors
figure
imshow(mat2gray(reshape(PCScore1,101,101)),[]);
figure
imshow(mat2gray(reshape(PCScore2,101,101)),[]);
figure
imshow(mat2gray(reshape(PCScore3,101,101)),[]);

%display with RGB bands
figure
imshow(mat2gray(im(:,:,[30,20,7])),[])

%pca scores for the last few components
```

```
[PCVec_all,PCVal_all]=eigs(im_cov,194);
%for i = 192:194
%   figure
%   imshow(mat2gray(reshape(PCVec_all(:,i),1,195)),[])
%end

figure
hold on
for i = 192:194
    plot(PCVec_all(:,i))
end
hold off

%last pca score
PCScore_l1 = PCVec_all(:,194).'*((im_vec(1,:)-im_mean)).';
for i = 2:10201
    PCScore_l1 = [PCScore_l1;PCVec_all(:,194).'*((im_vec(i,:)-im_mean)).'];
end

PCScore_l2 = PCVec_all(:,193).'*((im_vec(1,:)-im_mean)).';
for i = 2:10201
    PCScore_l2 = [PCScore_l2;PCVec_all(:,193).'*((im_vec(i,:)-im_mean)).'];
end

PCScore_l3 = PCVec_all(:,192).'*((im_vec(1,:)-im_mean)).';
for i = 2:10201
    PCScore_l3 = [PCScore_l3;PCVec_all(:,192).'*((im_vec(i,:)-im_mean)).'];
end

figure
imshow(mat2gray(reshape(PCScore_l1,101,101)),[]);
figure
imshow(mat2gray(reshape(PCScore_l2,101,101)),[]);
figure
imshow(mat2gray(reshape(PCScore_l3,101,101)),[]);
```

3.

```
%Scatter plots of PCA scores
figure
scatter(PCScore1,PCScore2)
title('PCA Scores of the first two components')
figure
scatter3(PCScore1.',PCScore2.',PCScore3.')
title('PCA Scores of the first three components');
```

4.

```
%Plot eigenvectors
for i = 1:3
    figure
    plot(PCVec(:,i))
end
%Plot columns of S
for i = 1:3
    figure
    plot(S(:,i))
end
```

9.

```
% 9. projection of S
S_mean = mean(S.');
S_mean = S_mean.';
```

```
for i = 1:3
    coord1_S(i) =(PCVec(:,1).'*(S(:,i)-S_mean));
end

for i = 1:3
    coord2_S(i) = (PCVec(:,2).'*(S(:,i)-S_mean));
end

for i = 1:3
    coord3_S(i) = (PCVec(:,3).'*(S(:,i)-S_mean));
end

figure
hold on
scatter(PCScore1.',PCScore2.')
scatter(coord1_S.',coord2_S.')
title('PCA Scores with Coordinates of Columns of S')
hold off

figure
scatter3(PCScore1.',PCScore2.',PCScore3.')
hold on
scatter3(coord1_S.',coord2_S.',coord3_S.')
title('PCA Scores with Coordinates of Columns of S')
hold off
```

10.

```
%10. Calculate A
X = im_vec.';
A = (inv(S.'* S))*S.'*X;
%plot abundance
for i = 1:3
    figure
    imshow(mat2gray(reshape(A(i,:),101,101)),[])
end
```

11.

```
%11.
B_C = [S.'*S, -1*ones(3,1)];
D_E = [ones(1,3),0];
B_C_D_E = [B_C;D_E];

F = S.'*X;
G = [ones(1,10201)];
F_G = [F;G];

A_lambdatrans = inv(B_C_D_E)*F_G;
sol_A = [A_lambdatrans(1,:);A_lambdatrans(2,:);A_lambdatrans(3,:)];

%plot abundance
for i = 1:3
    figure
    imshow(mat2gray(reshape(sol_A(i,:),101,101)),[])
end
```

12.

```
% Projection X
for i = 1:2
    X_proj(:,i) =X.'*PCVec(:,i);
```

```
end

%Projection of S
for i = 1:2
    S_proj(i,:) = S.'*PCVec(:,i);
end

%calculat new A
tic
Bp_Cp = [S_proj.'*S_proj, -1*ones(3,1)];
Dp_Ep = [ones(1,3),0];
Bp_Cp_Dp_Ep = [Bp_Cp;Dp_Ep];

Fp = S_proj.'*X_proj.';
Gp = [ones(1,10201)];
Fp_Gp = [Fp;Gp];

proj_A_lambdatrans = inv(Bp_Cp_Dp_Ep)*Fp_Gp;
proj_A = [proj_A_lambdatrans(1,:);proj_A_lambdatrans(2,:);proj_A_lambdatrans(3,:)];
toc
```

13.

```
%unconstrained
for i = 1:10201
    RMSE_unc(i)= (1/sqrt(195))*norm(X(:,i)-S*A(:,i));
end

%constrained
for i = 1:10201
    RMSE_con(i)= (1/sqrt(195))*norm(X(:,i)-S*sol_A(:,i));
end

figure
imshow(mat2gray(reshape(RMSE_unc,101,101)),[]);
figure
imshow(mat2gray(reshape(RMSE_con,101,101)),[]);

%compute mean
RMSE_unc_mean = mean(RMSE_unc);
RMSE_con_mean = mean(RMSE_con);
```