# Image Understanding HW1

43688514 Jia-Wei, Chiang

## A: Matlab Warmup

**1. Describe the result of each of the following Matlab commands.**

a. >>x=randperm(5)

Set x to a vector which contains 5 random permutation elements from 1 to 5

Example: x may be set to [1 3 5 2 4]

b. >> a = [1 2 3;4 5 6;7 8 9];
>> b = a(2,:);

Set a to a 3 times 3 vector which the first row is 1, 2, 3 and the second is 4, 5, 6, and the last row is 7, 8, 9.Set b to an vector which contains the second row of matrix a. The answer is [4 5 6].

c. >> f = [1501:2000];
>> g = find(f > 1850);
>> h = f(g);

Set f to a vector with elements from 1501 to 2000 in ascending order and the stepsize is 1. Set g to a vector that contains the indices of elements in f which are greater than 1850.
Set h to a vector that contains the elements in f which are indexed by g.

Result f: [1501 1502 1503 … 2000], g: [351 352 353 … 500], h: [1851 1852 1853 … 2000]

d. >> x = 22.*ones(1,10);
>> y = sum(x);

Set x to be a 1*10 matrix (row vector) of ones and all elements multiply 22.
Set y to be the sum of all elements in x.

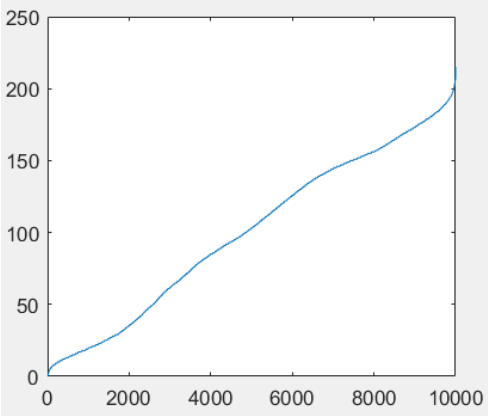
Result x: [22 22 22 22 22 22 22 22 22 22], y: 220

e. >> a = [1:100];
>> b = a([end:-1:1]);

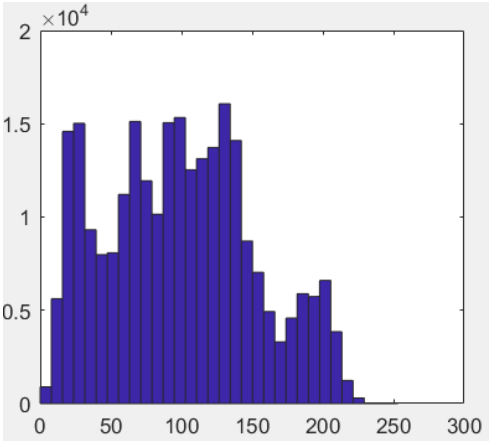
Set a to a vector with elements from 1 to 100 in ascending order and the stepsize is 1.Set b to a vector with elements from the last element of a to the first element of a in descending order and interval is 1.

Result a: [1 2 3 … 100], b: [100 99 98 … 1]

**2.(a) Reshape the intensities stored in *A* into a single 10,000-dimensional vector *x*, sort the values in *x* and plot the values**




**2.(b) Display a figure showing a histogram of *A*'s intensities with 32 bins using the *hist* function**




**2.(c) Create and display a new binary image the same size as *A*, which is white wherever the intensity in A is greater than a threshold *t*, and black everywhere else.**

**2.(d) Display the bottom right quadrant of *A***

**2.(e) Generate a new image (matrix), which is the same as *A*, but with *A*'s mean intensity value subtracted from each pixel. Set any negative values to 0**

**2.(f) Display the mirror-flipped version of image A**

**2.(g) Use the *min* and *find* functions to set *x* to the single minimum value that occurs in *A*, and set *r* to the row it occurs in and *c* to the column it occurs in**

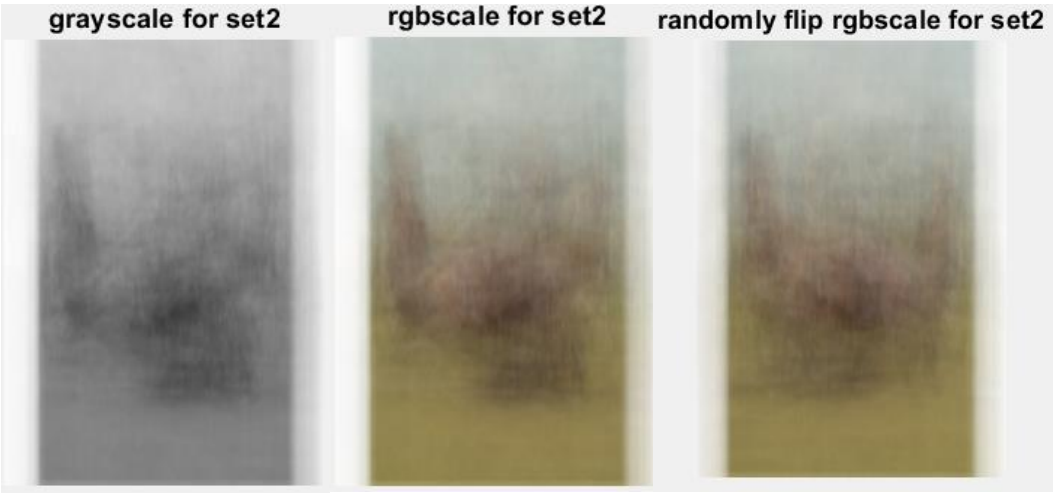
Minimum value x = 0

Find 9 pixels have same value: (81,1), (40,4), (53,20), (47,23), (37,26), (38, 26), (42, 27), (34, 33), (33, 35).

**2.(h) Let *v* be the vector: v = [1 8 8 2 1 3 9 8]. Use the *unique* function to compute the total number of unique values that occur in *v***

The total number of unique values that occur in v is 5.

## B: Computing average images

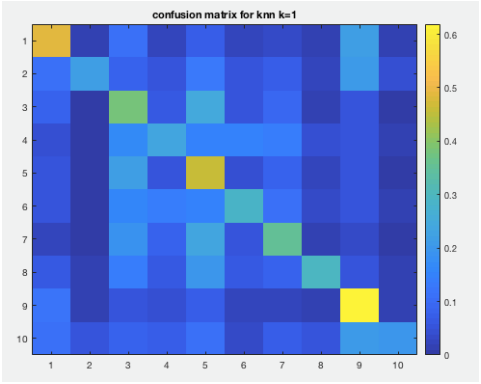




The average images show rough contour of pictures. The random mirror-flip average image looks blurrier than regular RGB scale, because half of them is flipped, which have less features.

# C: Image classification

## 1. display the first airplane in the test set

## 3. Compute a class confusion matrix




average classification rate: 0.3539

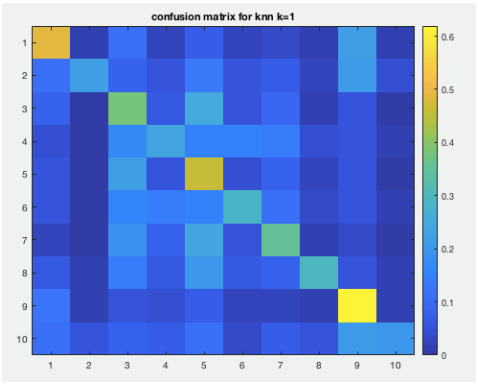
## 4. Construct a KNN-classifier that computes the K closest training images, and returns the most common label from this set

k=1, average missclassification rate: 0.6461, most common label: 4

k=3, average missclassification rate: 0.6697, most common label: 2

k=5, average missclassification rate: 0.6602, most common label: 4

k=1 has the best average misclassification.



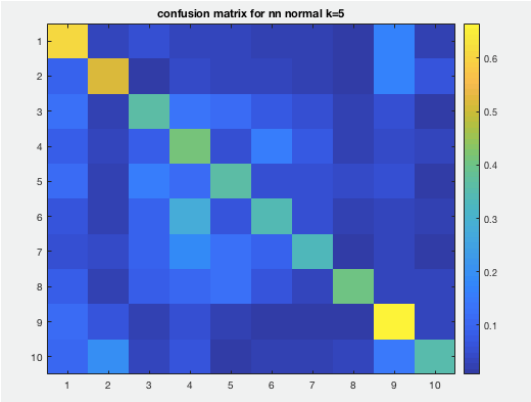
**5. KNN-classifier with normalized correlation**

k=1, average missclassification rate: 0.58, most common label: 8

k=3, average missclassification rate: 0.5837, most common label: 0

k=5, average missclassification rate: 0.5642, most common label: 0

With normalized correlation, k=5 has the best average misclassification.




**6. What is your best-performing system? Provide some explanation as to why you think this combination performed best**

My best-performing system is using normalized correlation with KNN-classifier k=5. Correlation Distance subtracts the mean value from all the pixels, it balances the lighting condition in data set.

The images showing above are horses but misclassifying as deer because their similar contour.