| | | |
|---|---|---|
| NAME | : | SRAVANI KAMISETTY |
| SID | : | 304414410 |
| SUBJECT | : | MATHEMATICAL MODELLING & METHODS |
| HOMEWORK | : | #1 |

Mathematical Modeling & Methods - HW#1

Question 1:

(a) Standard basis vector of size 3 = 3*3 identity matrix

```
octave:1> eye (3)
ans =

Diagonal Matrix

   1   0   0
   0   1   0
   0   0   1
```

This needs to be transformed to matrix A which = [T(e1),T(e2),T(e3)] = [a1,a2,a3]

```
octave:2> a1 = [3;-2;1]
a1 =

   3
  -2
   1

octave:3> a2 = [6;0;7]
a2 =

   6
   0
   7
octave:5> a3 = [5;4;-1]
a3 =

   5
   4
  -1
```

then T(e1) = a1
hence T(e1) = T(1;0;0) = a1
or T(e1) = T(x,y,z) = (3x;-2x;x)

T(e2) = a2
or T(e2) = T(x,y,z) = (6y;0;7y)

T(e3) = a3
or T(e3) = T(x,y,z) = (5z;4z;-z)

or A(x,y,z) = [(3x, 6y, 5z);(-2x,0,4z);(x,7y,-z)] assuming (x,y,z are the values of corresponding e<i> vector)

(b)
Given a identity matrix of size 2

```
octave:6> eye (2)
ans =
```

Diagonal Matrix

```
   1   0
   0   1
```

[e1,e2] needs to be transformed into [4e2,-e1]

Hence A should be

```
octave:8> A = [0,-1;4,0]
A =

   0  -1
   4   0
```

Identity matrix * Tranformation matrix = Resultant matrix A
or transformation matrix  is also = A

transformation matrix =

```
   0  -1
   4   0
```

(c)

```
octave:11> M = [1, 0 ; 1 ,1]
M =

   1   0
   1   1
```

X axis is represented by the matrix C1 = [t; 0] where t is a variable definfing the value of x at that point in the x axis.

to transform X axis we need to : M * C1

```
   1   0  * t   = t
   1   1    0     t
```

The transformation is a matrix [t;t] which is represented by the line x=y. This is because for ex :

if t=4 the point which is defined by [t;t] = [4;4]
if t=6.5 the point which is defined by [t;t] = [6.5;6.5] i.e

y = mx + c is the eqn of any staright line.
where m = y2-y1/x2-x1 = 2.5/2.5 = 1; hence eqn => y = x +c

substitutitng (4,6) => 4 = 4 + c or c = 0

So the eqn of the line => y = x

Similarly the line y = 2 can be represented by the matrix c2 = [ t;2]

to transform X axis we need to : M * C1

```
   1   0  * t   = t
   1   1    2     t+2
```

The transformation is a matrix [t;t+2] which is represented by the line x+2=y. This is because for ex :

if t=4 the point which is defined by [t;t+2] = [4;6]
if t=6.5 the point which is defined by [t;t] = [6.5;8.5] i.e

y = mx + c is the eqn of any staright line.
where m = y2-y1/x2-x1 = 2/2 = 1; hence eqn => y = x +c

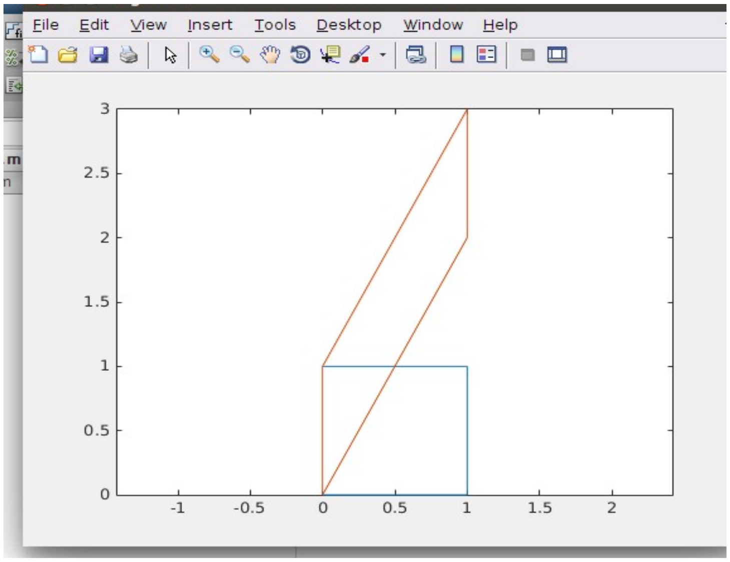
substitutitng (4,6) => 4 = 2 + c or c = 2

So the eqn of the line => y = x + 2

d.

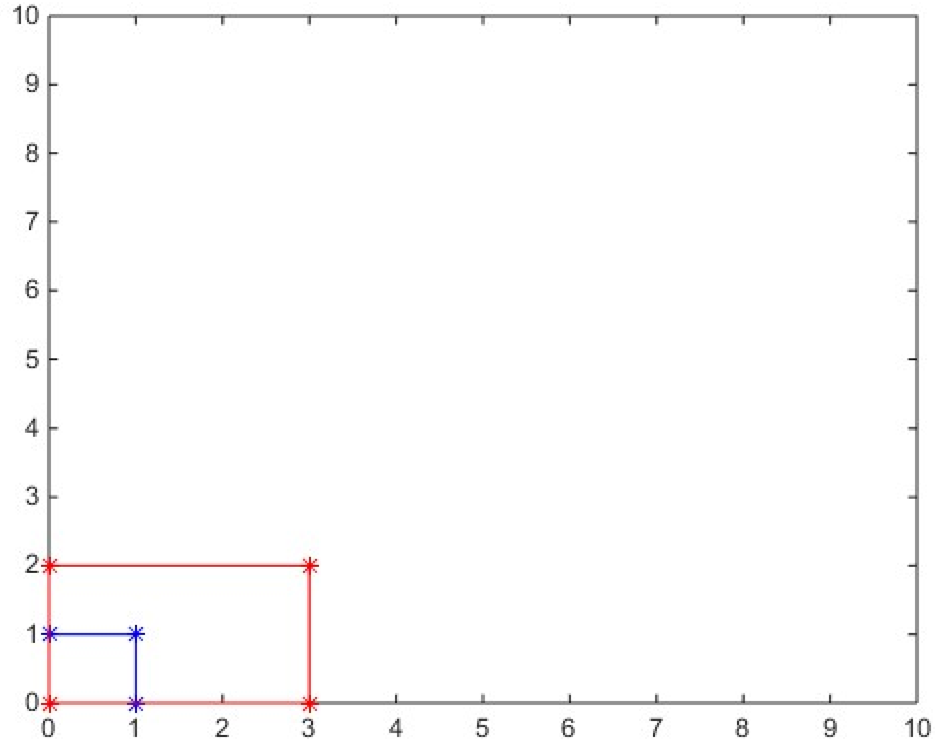
```
function [ ] = sheer( a,t )
  b = t * a;
  plot(a(1,:),a(2,:));
  hold on;
  plot(b(1,:),b(2,:));
  hold off;
  axis equal;
end
```

```
>> a = [0 1 1 0 0;0 0 1 1 0]
>> t = [1 0; 2 1]
>> sheer(a,t)
```

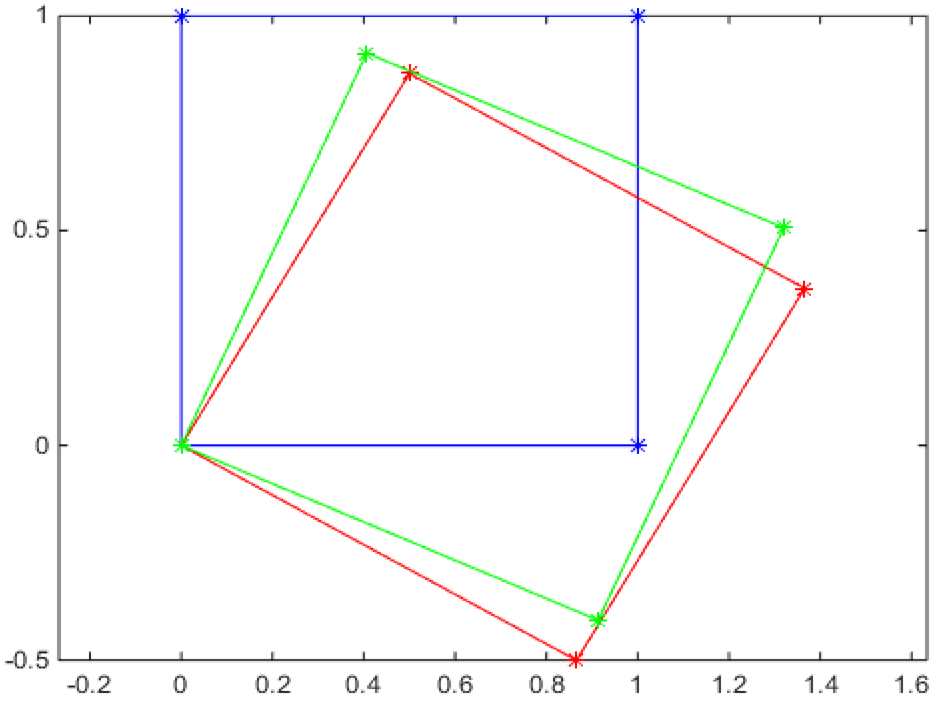


(e)

```
function stretch_plot(pts)
figure
stretch = [3 0; 0 2]
plot(pts(1,:),pts(2,:),'b*-');
axis([0 10 0 10])
stretch_sqr = stretch * pts;
hold on
plot(stretch_sqr(1,:), stretch_sqr(2,:), 'r*-');
end
```



```
f)function ref_plot(pts)
ref = [cos(pi/3) sin(pi/3); sin(pi/3) -cos(pi/3)];
plot(pts(1,:),pts(2,:),'b*-');
ref_sqr = ref * pts;
hold on
axis equal
plot(ref_sqr(1,:),ref_sqr(2,:)
ref = [cos(pi/5) sin(pi/5); sin(pi/5) -cos(pi/5)];
ref_sqr = ref * ref_sqr;
hold on
plot(ref_sqr(1,:),ref_sqr(2,:)
end
```



QUESTION 2:

1.

```
function I = id_mat(n)
X = 1 : n
Y = [1 : n]'
A = X' * X
A = A ./ A
Rows = X .* A
Cols = Y .* A
I = (Rows == Cols)
end

octave:2> id_mat(4)
X =

ans =

   1   0   0   0
   0   1   0   0
   0   0   1   0
   0   0   0   1
```

2.

```
abs_func = @(X) -1 .* X .* (X<0) + X .* (X>0)

octave:4>
octave:4> x = [-1 1]
x =

  -1   1

octave:5> abs_func(x)
ans =

   1   1
```

3.function LT = ltri(X)

```
[n,p] = size(X)
I = 1 : n
J = [1 : p]'
A = X' * X
A = A ./ A
Rows = I .* A
Cols = J .* A
L = (Rows <= Cols)
LT = X .* L
end

octave:13> x
x =

   1   2
   3   4

octave:14> ltri(x)

ans =

   1   0
   3   4
```

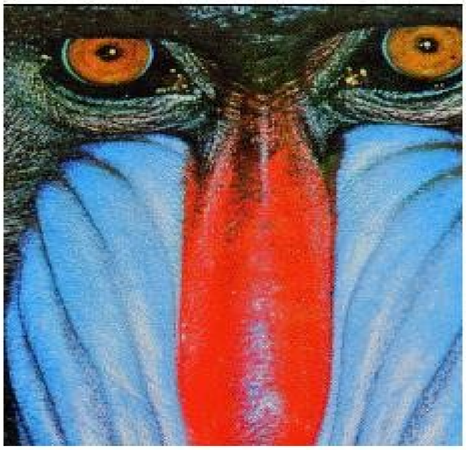
QUESTION 3:

a.

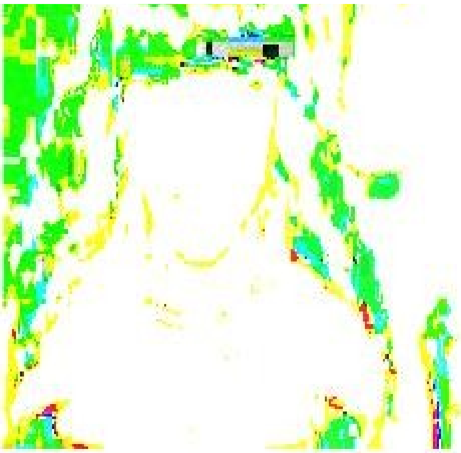
```
%interpolate function
function [ C ] = interpolate(A, B, t )
  C = (1-t).* A + t.* B
end

%code to show interpolation and extrapolation
[x2,map] =
imread('/home/kami/Documents/FALL/modelling/hw1/nikki_minaj.jpg');
load mandrill;
mandrill_cropped = Mandrill(1:252,1:228,1:3)
C = interpolate(double(x2),mandrill_cropped,0.25);
imshow(C)
C = interpolate(mandrill_cropped,double(x2),1.25);
imshow(C)

%code for the movie
function [] = mov( A,B,n)
    F(n) = struct('cdata',[],'colormap',[]);
    for j = 1:n
        C = interpolate(double(A),double(B),j-1/n);
        imshow(C);
        F(j) = getframe;
    end
    movie(F);
end
```

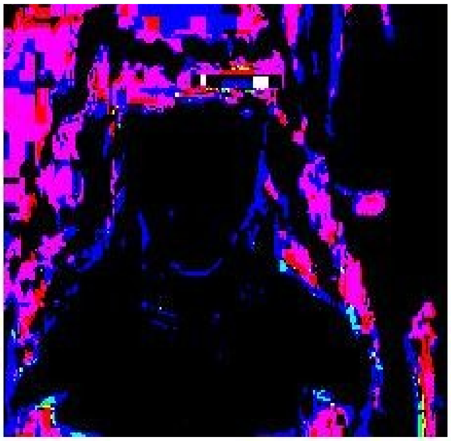
A :

B:



C : 0.25

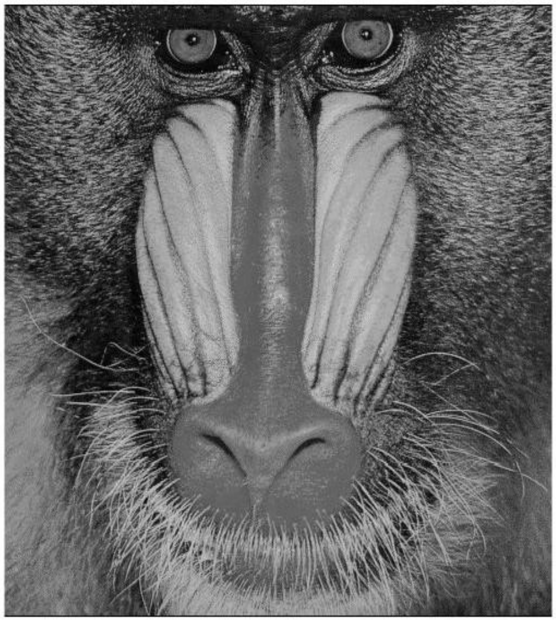




3b.

1.

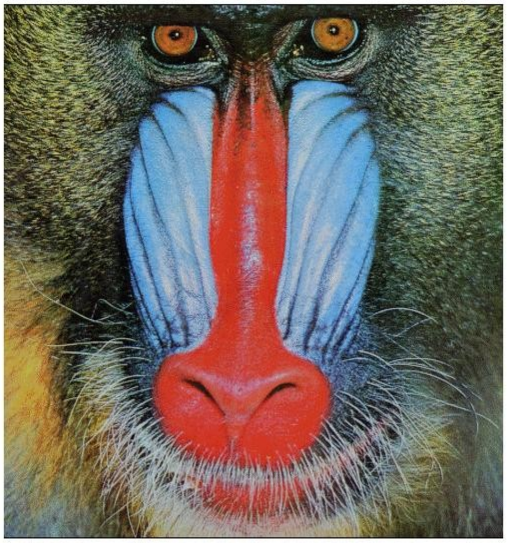
```
%grayscale
function [ G ] = grayscale( A )
%Converts an image to gray scale
% R = G = B
    [R,G,B] = image2rgb(A);
    G = cat(3, (R+G+B)/3, (R+G+B)/3, (R+G+B)/3);
end

C = grayscale(mandrill)
imshow(C)
```



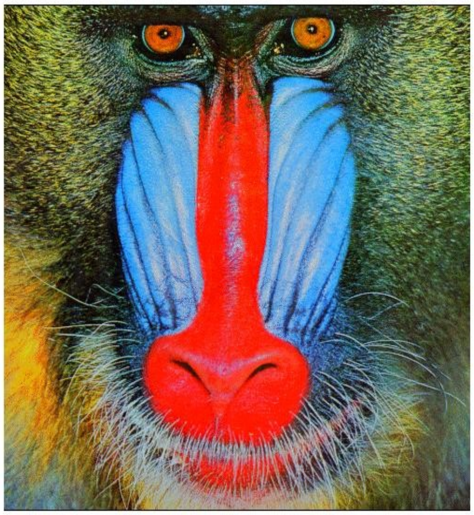
2.

```
%saturate
function [ S ] = saturate( A,t )
%changes image saturation
   grayScale = grayscale(A);
   S = interpolate(A,grayScale,t);
end
```

t = 0.25

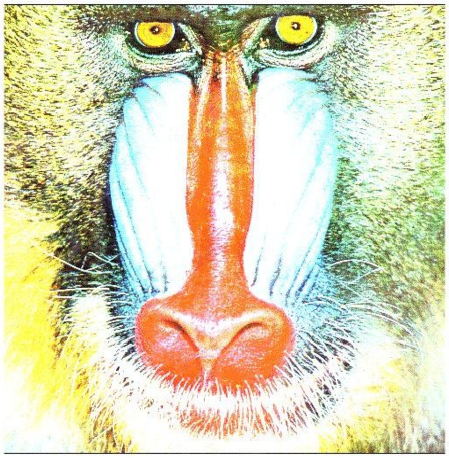


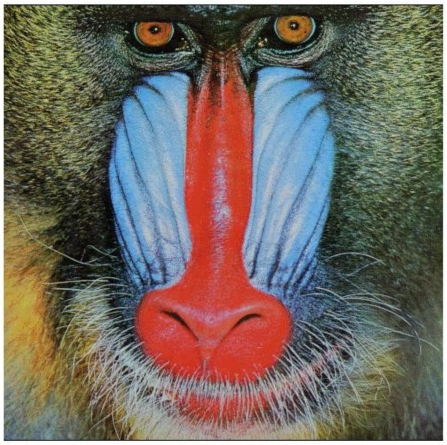
t = -0.25



3.

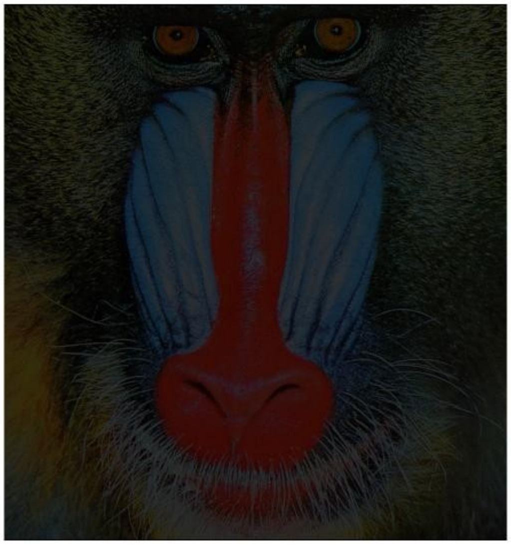
```
%brigten
function [ B ] = brighten( A,t )
%brightens the image
    B = t.* A;
end
C = brighten(mandrill,0.25)
imshow(C)
C = brighten(mandrill,-0.25)
imshow(C)
```





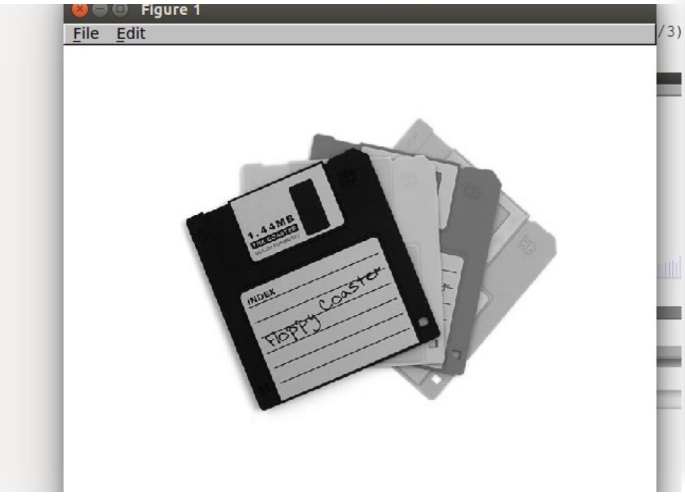
4.

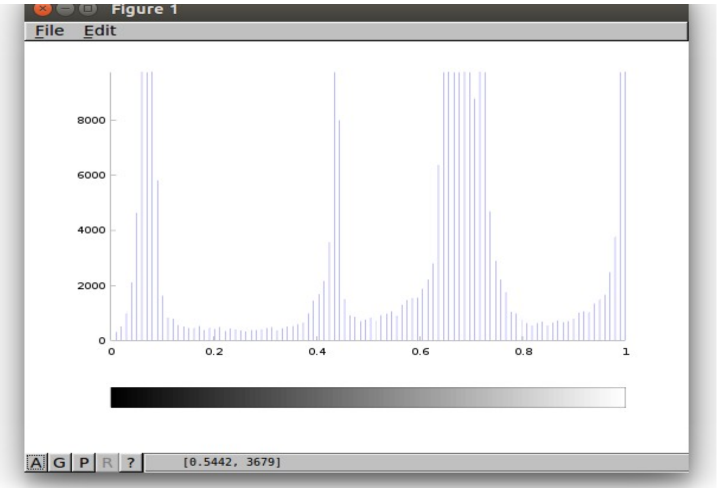
```
%darken
function [ D ] = darken(A,t )
% darkens the image
    hsvConverted = rgb2hsv(A);
    H = double(hsvConverted(:,:,1));
    S = double(hsvConverted(:,:,2));
    V = double(hsvConverted(:,:,3));
    V = t.* V;
    hsvDarkened = cat(3,H,S,V);
    D = hsv2rgb(hsvDarkened);
end

D = darken(mandrill,0.25)
imshow(D);
```



3c.

```
y = mat2gray(X);
imshow(y);
[R,G,B] = image2rgb(y);
G = cat(3, (R+G+B)/3, (R+G+B)/3, (R+G+B)/3);
imshow(G);
imhist(G,100);
```

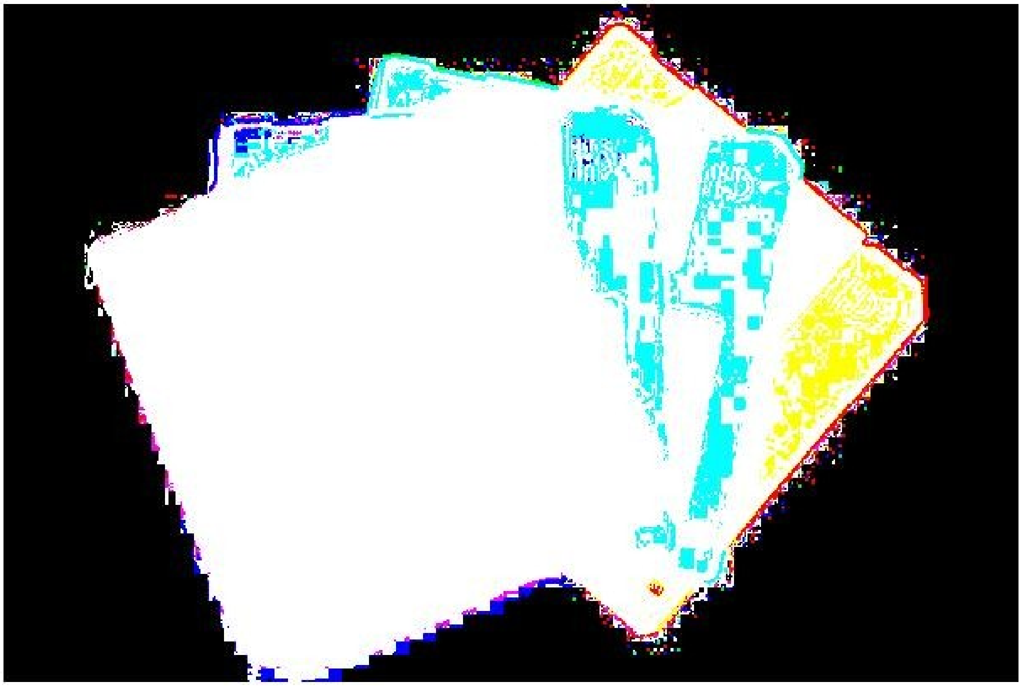




[0.5442, 3679]

3d.

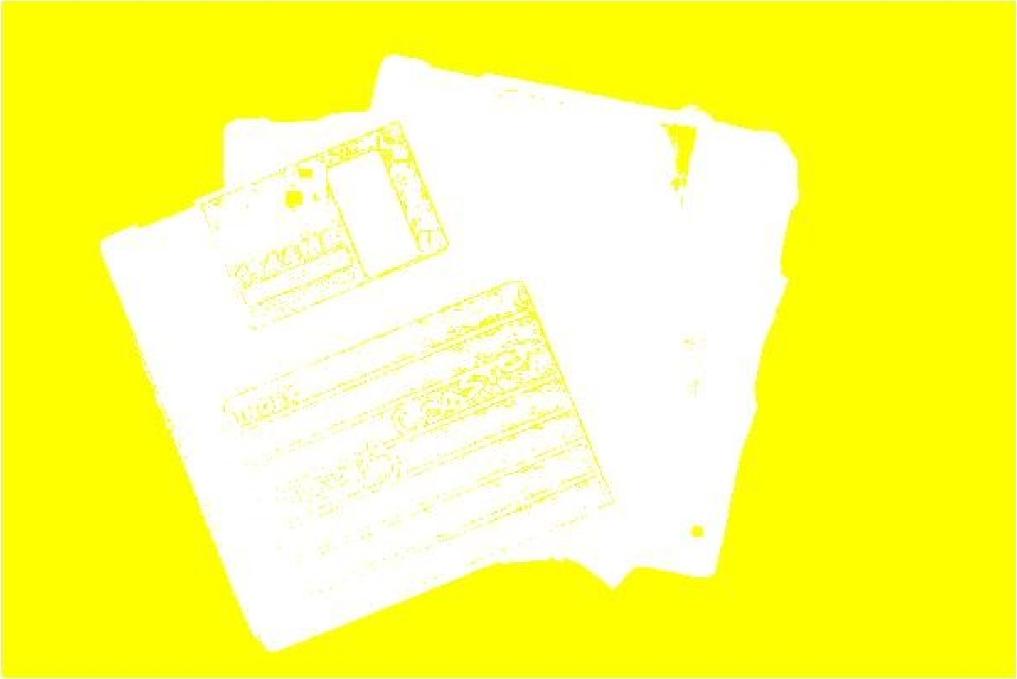
```
color_inversion
function [ Y ] = color_inversion( A )
% inverts the color in the image
    [R,G,B] = image2rgb(A);
    R1 = 255 - R;
    G1 = 255 - G;
    B1 = 255 - B;
    Y = rgb2image(R1,G1,B1);
end

Y = color_inversion(X)
imshow(Y)

% Using the floppy image instead of Mandrill because the Mandrill image has
r,g,b in the range 0-1. And to convert it to 1-255 range we need Image
processing toolkit which is not avaialble on my MATLAB.
```



3d2.

```
%Color transform
function [ Y ] = YCbCr( A )
%YCbCr
    T = [0.29900, 0.58700, 0.11400; -0.16874,-0.33126,0.50000;0.50000,-
0.41869,-0.08131];
    v = [0;128;128];
    [R,G,B] = image2rgb(A);
    [rows,cols] = size(R);
    for r = 1:rows
        for c = 1:cols
```

```
            color = [R(r,c);G(r,c);B(r,c)];
            color = T*color;
            color = color + v;
            R1(r,c) = round(color(1));
            G1(r,c) = round(color(2));
            B1(r,c) = round(color(3));
        end
    end
    Y = rgb2image(R1,G1,B1);
end
```



3d3.

```
>> T = [0.29900, 0.58700, 0.11400; -0.16874,-0.33126,0.50000;0.50000,-
0.41869,-0.08131];
>> min_value

min_value =

     0
     0
     0
>> max_value =

   255
   255
   255
>> T*max_value + v

ans =

   255
   128
   128
```

```
>> T*min_value + v

ans =

     0
   128
   128
```

Since for both max and min values of R,G,B components of color matrrix the resultant is positive, for values in between it will be positive.

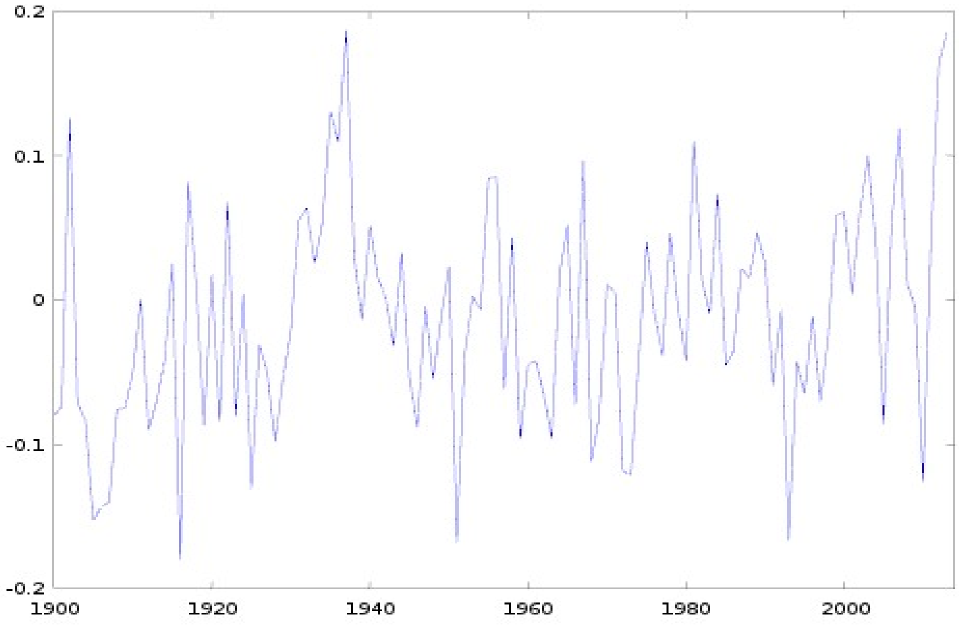
4. OVERHEATING

Problem 4:

a. Code to generate a plot of average temperature anamoly in July for the years 1900-2014 for United States

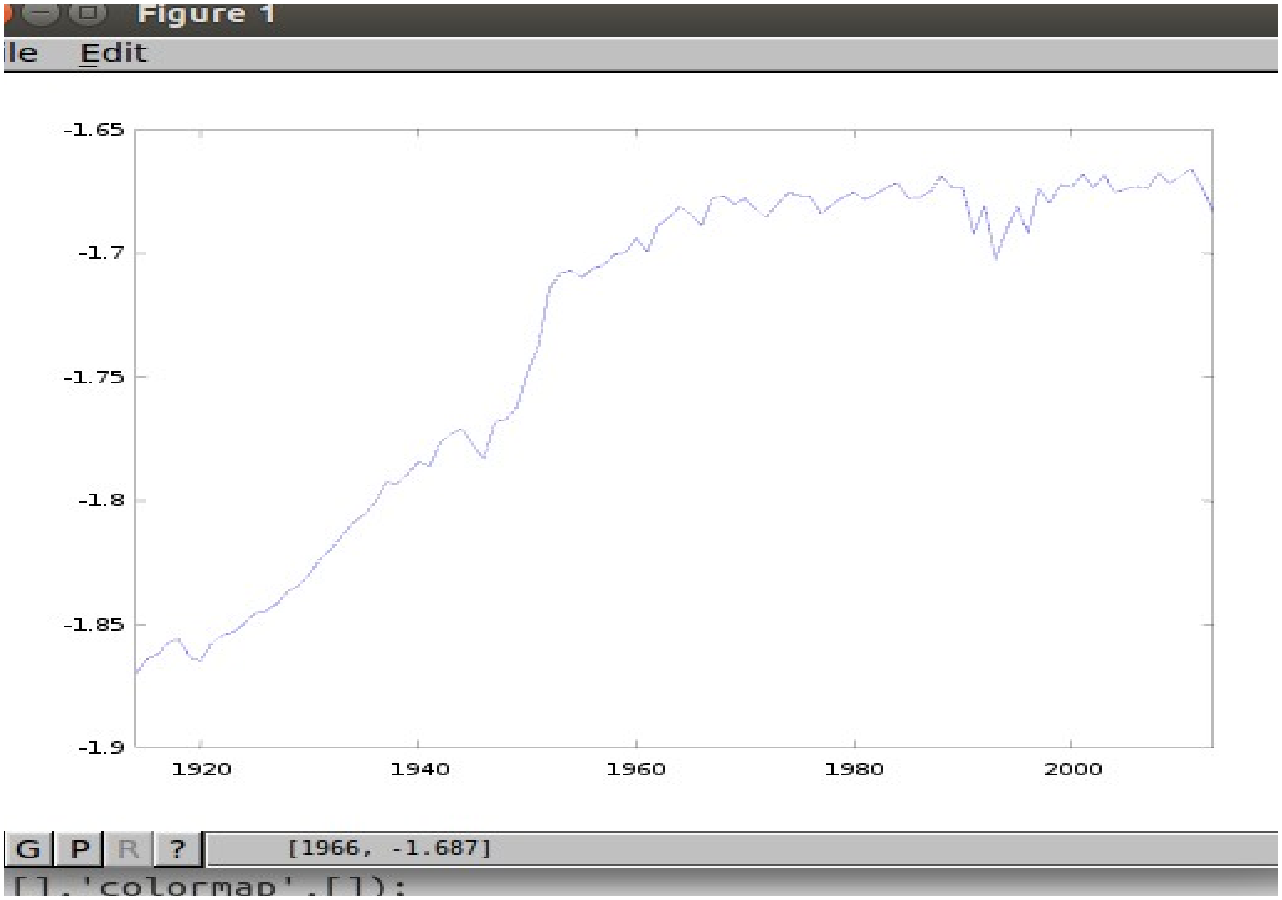
```
GHCN = csvread('ghcn.csv');
GHCN_in_centigrade  = (GHCN(:,3:74) - 2500) / 100;
temperature_anomaly = reshape( GHCN_in_centigrade, [36, 12, 135, 72] );
US_latitude  = 9:12
US_longitude = 15:20
my_years = 1900:2013
my_slice = temperature_anomaly( US_latitude, :, my_years - 1880,
US_longitude );

total_number_of_grid_squares = length(US_latitude) * length(US_longitude) *
12;
N = total_number_of_grid_squares;
monthly_anamoly = sum(sum(my_slice, 4),1);
for n = 1:114
  yearly_matrix(1:12) = monthly_anamoly(:,:,n);
  july_anomaly(n) = yearly_matrix(7)/N;
end
plot( my_years, july_anomaly )
xlim([1900,2014])
```

b.Code to



generate a plot of average temperature anamoly in July for the years 1900-2014 for entire world

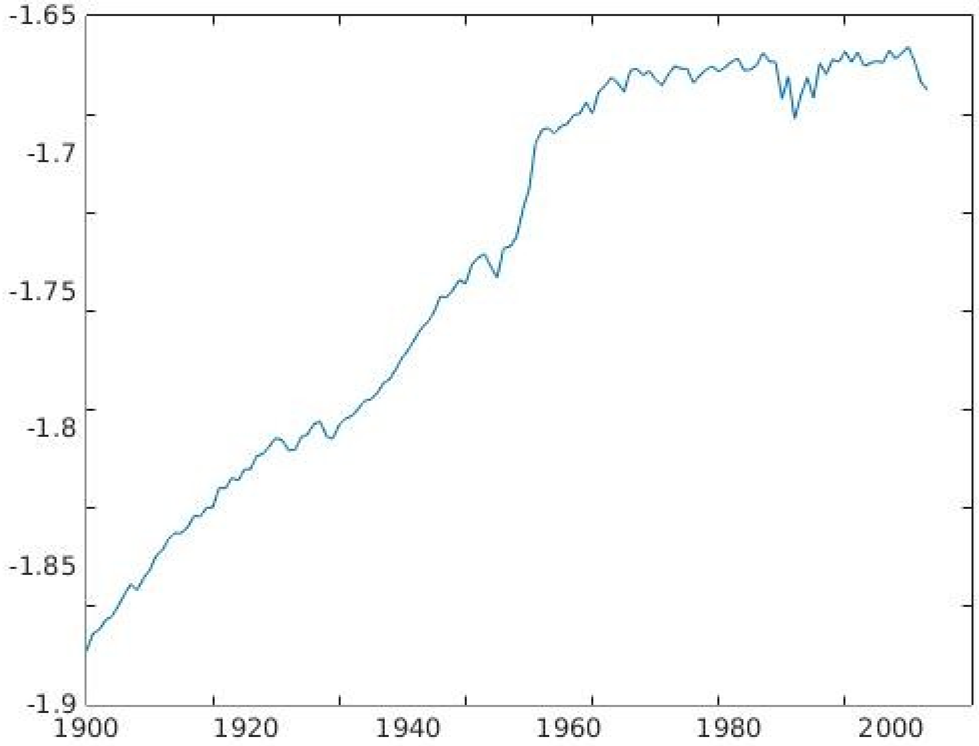
```
GHCN = csvread('ghcn.csv');
GHCN_in_centigrade  = (GHCN(:,3:74) - 2500) / 100;
temperature_anomaly = reshape( GHCN_in_centigrade, [36, 12, 135, 72] );
US_latitude  = 1:36
US_longitude = 1:72
my_years = 1900:2013
my_slice = temperature_anomaly( US_latitude, :, my_years - 1880,
US_longitude );

total_number_of_grid_squares = length(US_latitude) * length(US_longitude) *
12;
N = total_number_of_grid_squares;
monthly_anamoly = sum(sum(my_slice, 4),1);
for n = 1:114
  yearly_matrix(1:12) = monthly_anamoly(:,:,n);
  july_anomaly(n) = yearly_matrix(7);
end
plot( my_years, july_anomaly )
xlim([1914,2013])
```



[1966, -1.687]

[],'colormap',[]);

```
c. function [] = ghcn_hotplot()
GHCN =
csvread('/home/kami/Documents/FALL/modelling/hw1/Homework1/ghcn.csv');
    GHCN_in_centigrade  = (GHCN(:,3:74) - 2500) / 100;
    temperature_anomaly = reshape( GHCN_in_centigrade, [36, 12, 135, 72] );

    US_latitude  = 1:36;
    US_longitude = 1:72;
    my_years = 1880:2014;
    my_slice = temperature_anomaly( US_latitude, :, my_years - 1880,
US_longitude );

    colormap(hot)
    F(114) = struct('cdata',[],'colormap',[]);
    total_number_of_grid_squares = length(US_latitude) *
length(US_longitude) * 12;
    N = total_number_of_grid_squares;
    monthly_anamoly = sum(sum(my_slice, 4),1);

    for n = 1:110
      yearly_matrix(1:12) = monthly_anamoly(:,:,n);
      july_anomaly(n) = yearly_matrix(7);
      plot( my_years(1:n), july_anomaly );
      F(n) = getframe;
    end
    xlim([1880,2014]);
    %movie(F);
end
```


-1.65
-1.7
-1.75
-1.8
-1.85
-1.9
1900
1920
1940
1960
1980
2000