Homework #7

Table of Contents

Ian Blackstone, Helena-Nikolai Fujishin
Math 365, Fall 2016

Problems

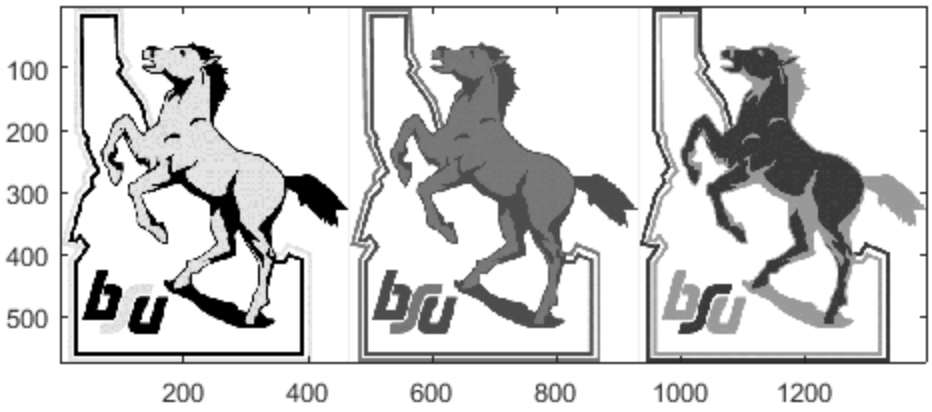
```
function hmwk1()
hmwk_problem(@prob1,'prob1');
hmwk_problem(@prob2,'prob2');
end
function hmwk_problem(prob,msg)
try
 prob()
 fprintf('% s : Success!\n',msg);
catch me
 fprintf('% s : Something went wrong.\n',msg);
 fprintf('% s\n',me.message);
end
fprintf('\n');
end
```

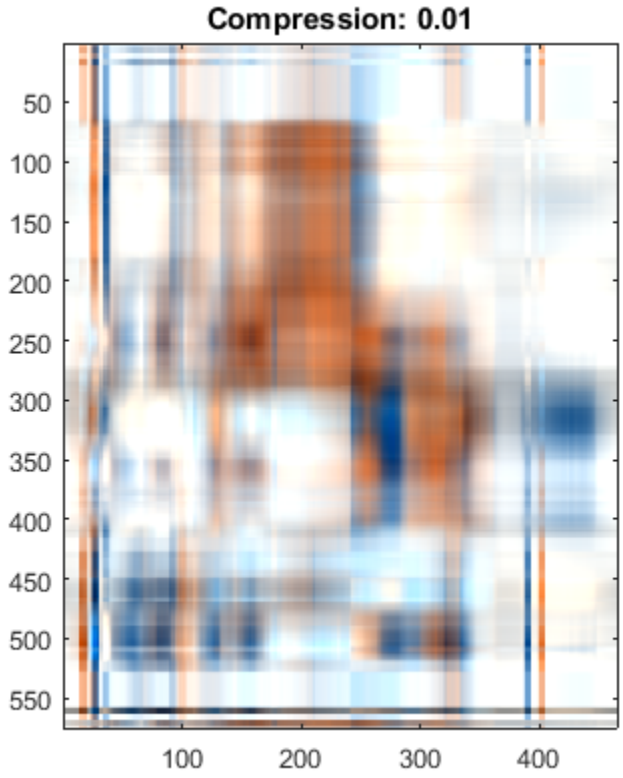
Problem #1 : Image Compression

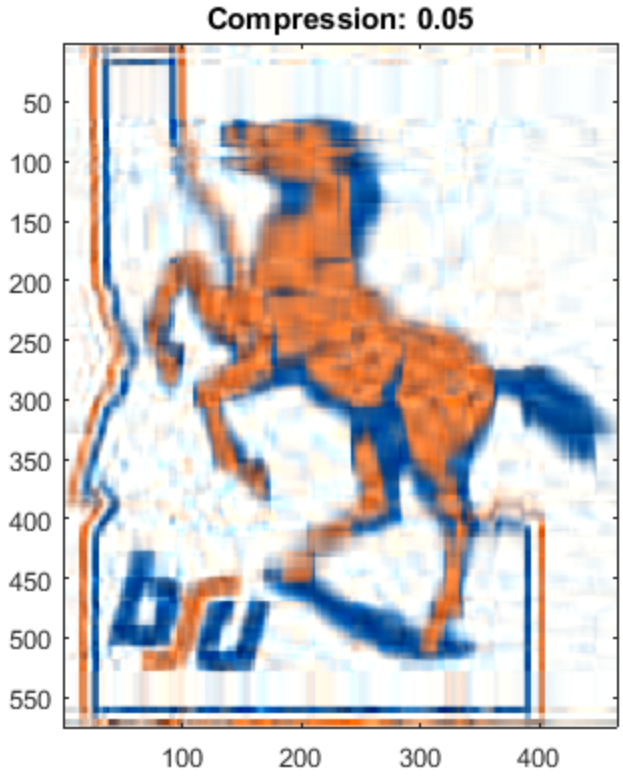
```
function prob1()
% Part a

% Load the image
A = imread('bsubronco.png');

% Find the size of the image
[m,n,p] = size(A);

% Make the data a 2D array
B = reshape(A,[m,3*n]);

% Rescale the data between 0 and 1.
B = double(B)/255;

% Display the image with the color data side by side
imagesc(B), axis image, colormap(gray);

% Part b
```

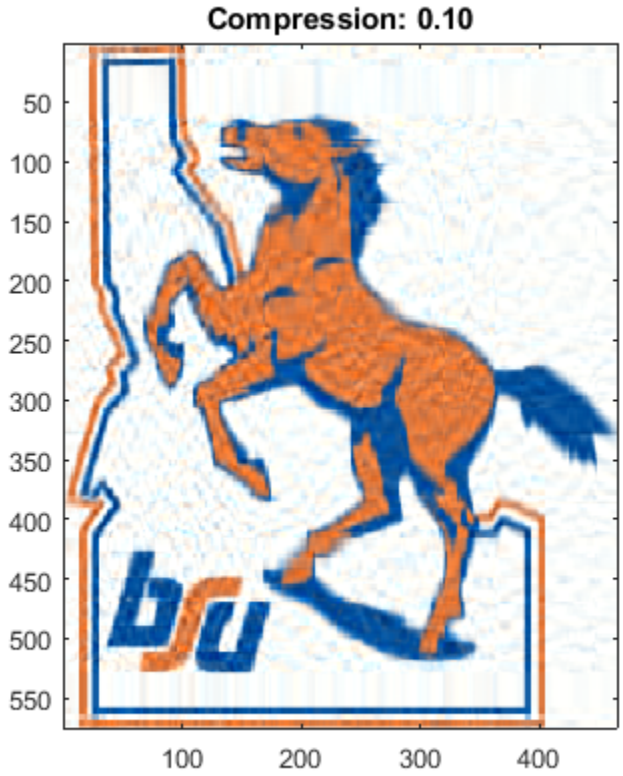
```
% Perform the singular value decomposition
[U,S,V] = svd(B);

% Declare our compression ratios
CR = [0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.75];

% Find the value of k for each compression ratio
k = floor(CR.*(3*m*n)/(m+3*n));

% Generate a compressed image at each ratio.
for p = 1:8
  Ak = U(:,1:k(p))*S(1:k(p),1:k(p))*V(:,1:k(p))';
  Ak = min(1,max(0,Ak));
  Ak = reshape(Ak,[m,n,3]);
  figure()
  imagesc(Ak), axis image, colormap(jet),title(sprintf('Compression:
 %.2f',CR(p))), drawnow
end

% Part c

E = diag(S);

% Display the singular values
figure()
semilogy(1:length(E),E','-',k(1),E(k(1)),'rx',k(2),E(k(2)),'gx',k(3),E(k(3)),'bx',
legend('Singular values','0.01 compression','0.05
 compression','0.1 compression','0.2 compression','0.3
 compression','0.4 compression','0.5 compression','0.75
 compression','location','southwest')
title('Singular values and compression ratios')
xlabel('Number of values')
ylabel('Singular value')

end
```

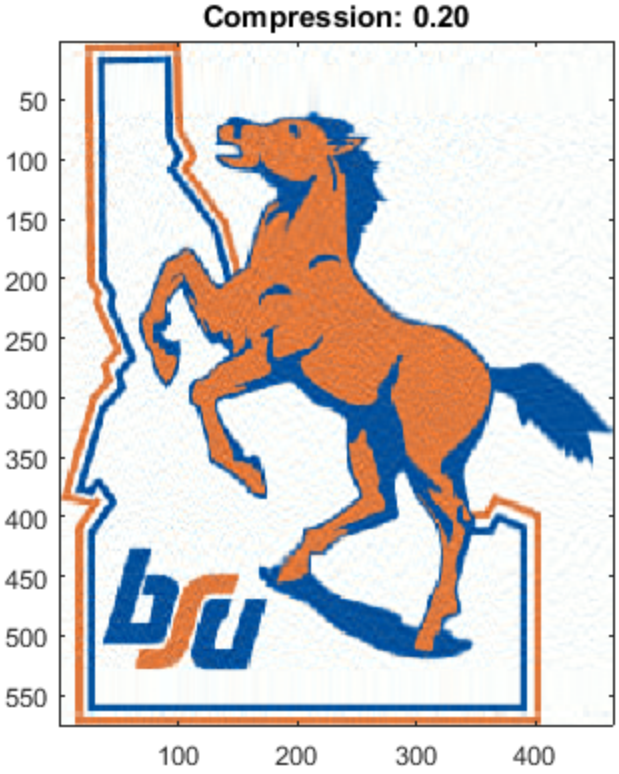
prob1 : Success!

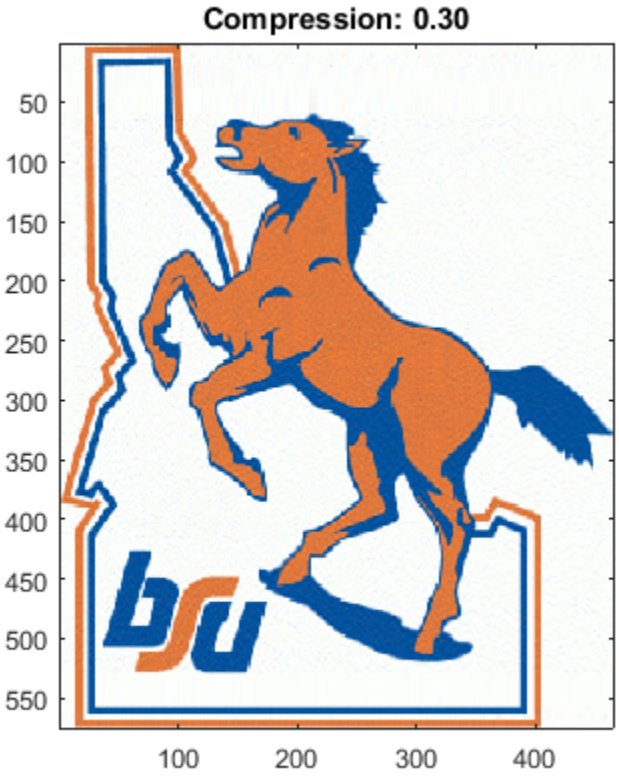

100
200
300
400
500
200
400
600
800
1000
1200

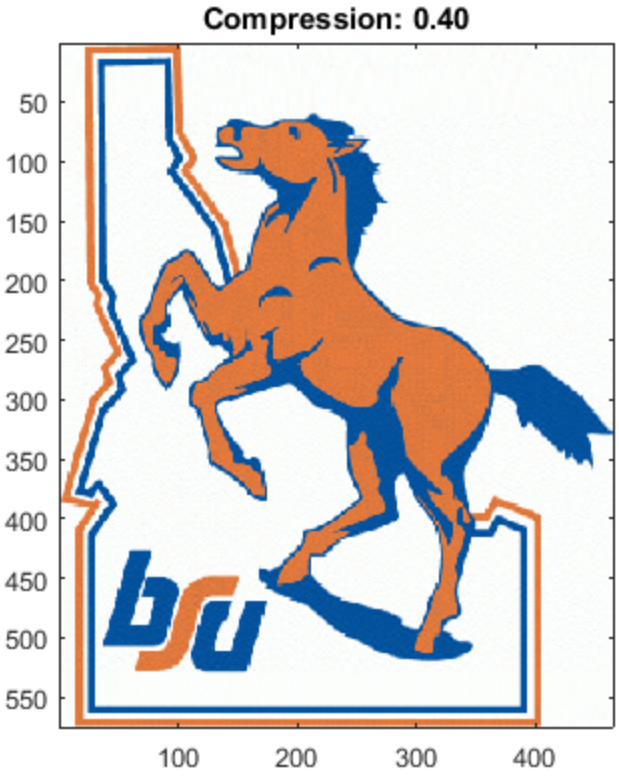

Compression: 0.01
50
100
150
200
250
300
350
400
450
500
550
100
200
300
400

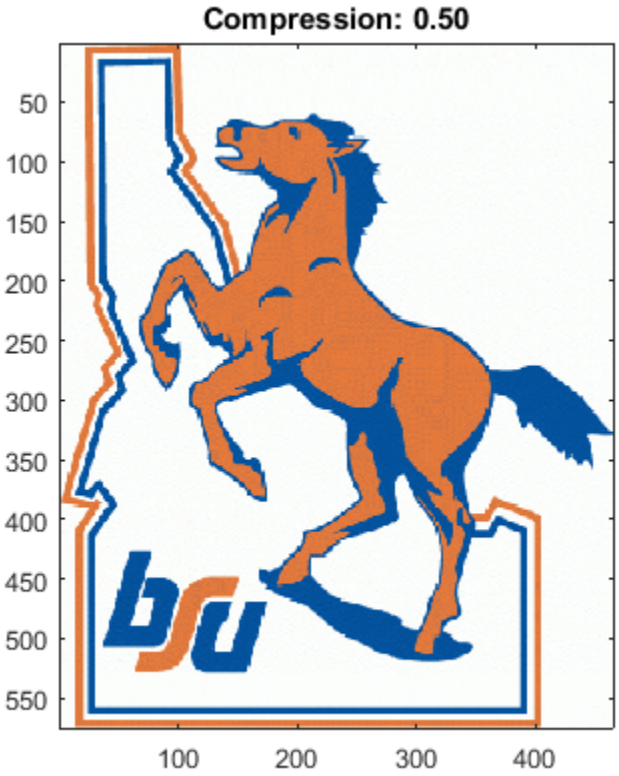

Compression: 0.05
50
100
150
200
250
300
350
400
450
500
550
100
200
300
400

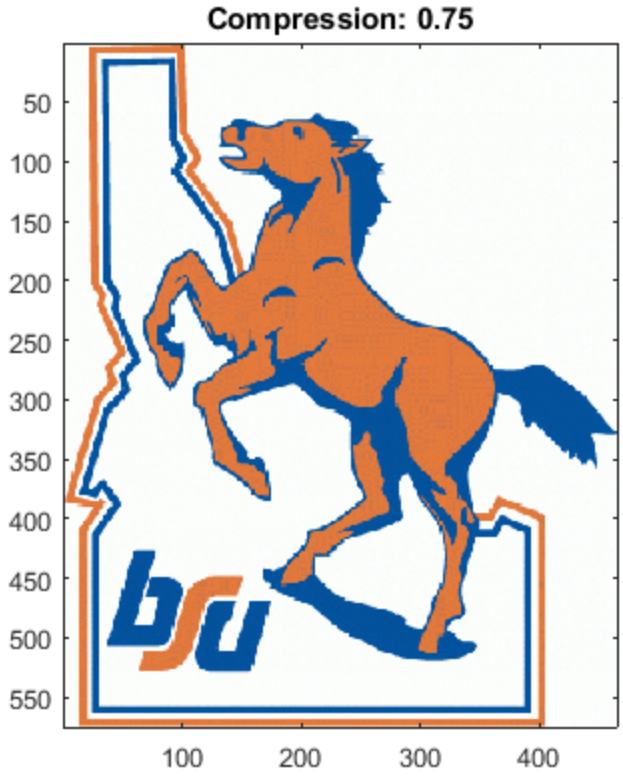

Compression: 0.10
50
100
150
200
250
300
350
400
450
500
550
100
200
300
400

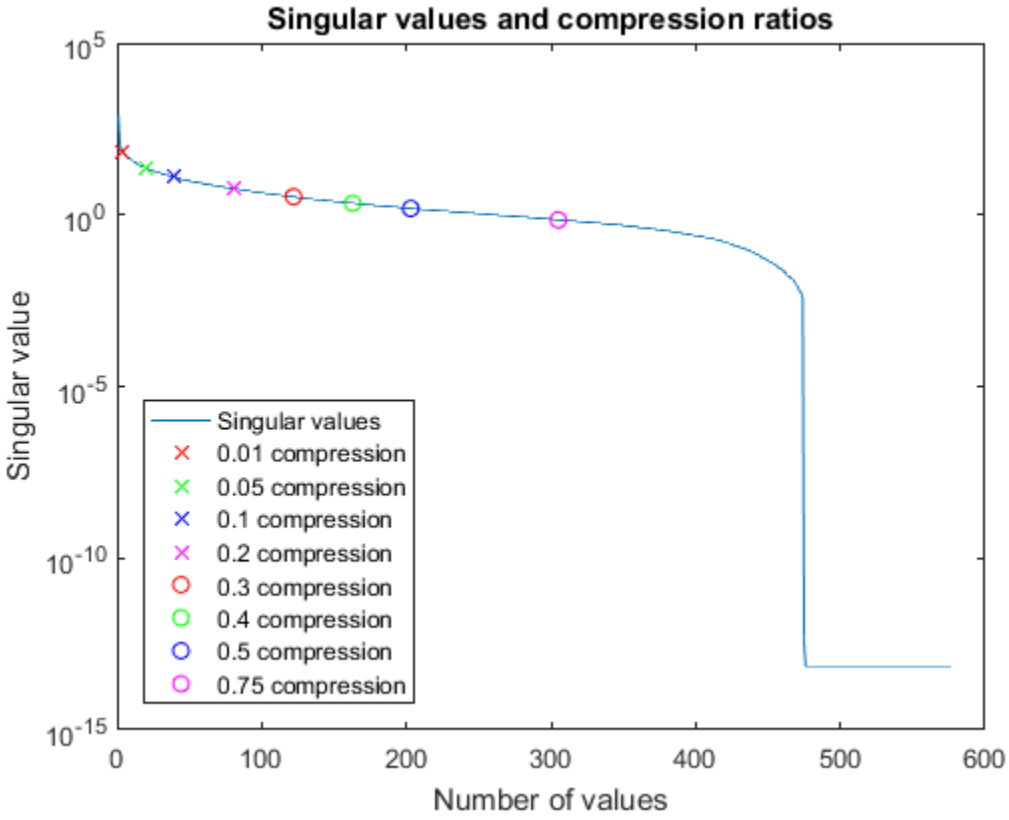

Compression: 0.20
50
100
150
200
250
300
350
400
450
500
550
100
200
300
400


Compression: 0.30
50
100
150
200
250
300
350
400
450
500
550
100
200
300
400


Compression: 0.40
50
100
150
200
250
300
350
400
450
500
550
100
200
300
400


Compression: 0.50
50
100
150
200
250
300
350
400
450
500
550
100
200
300
400


Compression: 0.75
50
100
150
200
250
300
350
400
450
500
550
100
200
300
400

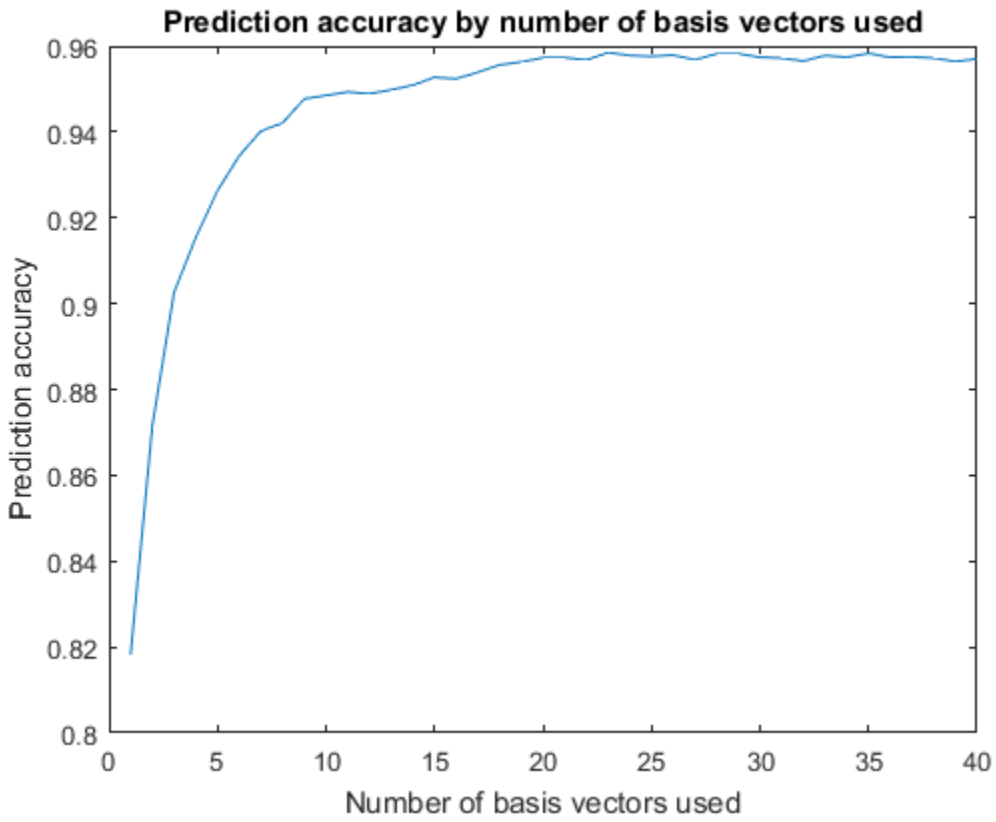

Singular values and compression ratios
10^5
10^0
10^{-5}
10^{-10}
10^{-15}
Singular value
0
100
200
300
400
500
600
Number of values
Singular values
0.01 compression
0.05 compression
0.1 compression
0.2 compression
0.3 compression
0.4 compression
0.5 compression
0.75 compression

Problem #2 : Image Classification

```
function prob2()

% Load in the training images
load mnist_training_data;

% Load in the test images
load mnist_test_data;

% Determine a "good" basis for the images using the SVD
[U0,S0] = mnist_svd(images0);
[U1,S1] = mnist_svd(images1);
[U2,S2] = mnist_svd(images2);
[U3,S3] = mnist_svd(images3);
[U4,S4] = mnist_svd(images4);
[U5,S5] = mnist_svd(images5);
[U6,S6] = mnist_svd(images6);
[U7,S7] = mnist_svd(images7);
[U8,S8] = mnist_svd(images8);
[U9,S9] = mnist_svd(images9);

% Determine what number the test images are

% An array to count the number of correct predicitons.
correct = zeros(1,40);

% Attempt to predict each image using an increasing number of basis
 vectors.
for k = 1:40
 for test = 1:length(testImages)
  B = double(testImages{test});
  [m,n] = size(B);
  % Flatten B into a vector b
  b = B(:);

  % Compute the residual between the "fit" of test image and the
 numbers in
  % the database.
  residual = zeros(10,1);
  residual(1) = norm(B(:)-U0(:,1:k)*U0(:,1:k)'*b);
  residual(2) = norm(B(:)-U1(:,1:k)*U1(:,1:k)'*b);
  residual(3) = norm(B(:)-U2(:,1:k)*U2(:,1:k)'*b);
  residual(4) = norm(B(:)-U3(:,1:k)*U3(:,1:k)'*b);
  residual(5) = norm(B(:)-U4(:,1:k)*U4(:,1:k)'*b);
  residual(6) = norm(B(:)-U5(:,1:k)*U5(:,1:k)'*b);
  residual(7) = norm(B(:)-U6(:,1:k)*U6(:,1:k)'*b);
  residual(8) = norm(B(:)-U7(:,1:k)*U7(:,1:k)'*b);
  residual(9) = norm(B(:)-U8(:,1:k)*U8(:,1:k)'*b);
  residual(10) = norm(B(:)-U9(:,1:k)*U9(:,1:k)'*b);

  [p , prediction] = min(residual);
  if prediction -1 == labels(test)
```

```
   correct(k) = correct(k) + 1;
  end
 end
end

figure()
plot(1:40,correct/length(testImages))
title('Prediction accuracy by number of basis vectors used')
ylabel('Prediction accuracy')
xlabel('Number of basis vectors used')

end
```

```
prob2 : Success!
```



Published with MATLAB® R2016a