Exercise 4, Image compression with svd

HA 3.12.2018

```
close all;format compact;clear
img=imread('Seurasaari.jpg');
[M,N]=size(img)
```

```
M = 2736
N = 10944
```

imshow(img)

Convert to double

```
dimg=im2double(img);  % dimg is 3-dim array.
size(dimg)
```

```
ans = 1×3
        2736        3648           3
```

```
imshow(dimg)
```

```
Warning: Image is too big to fit on screen; displaying at 25%
```

```
title('Original image');
```

Take svd of all 3 layers.

Here semicolons are important, extremely!

```
tic
[u1,s1,v1]=svd(dimg(:,:,1));   % Layer 1, Red
[u2,s2,v2]=svd(dimg(:,:,2));   % Layer 2, Green
[u3,s3,v3]=svd(dimg(:,:,3));   % Layer 3, Blue
T=toc
```

```
 T = 67.0671
```

```
%{
T =
   52.6644
%}
```

Let's have a look at the singular values:

```
D1=diag(s1);D2=diag(s2);D3=diag(s3);
[D1(1:20) D2(1:20) D3(1:20)]
```

```
ans = 20×3
10^3 ×
    1.0052    1.1472    1.3175
    0.1336    0.1488    0.1986
    0.1112    0.0838    0.0847
    0.0835    0.0638    0.0798
    0.0596    0.0436    0.0509
    0.0516    0.0369    0.0408
    0.0440    0.0320    0.0378
    0.0385    0.0291    0.0327
    0.0374    0.0247    0.0298
    0.0343    0.0240    0.0286
     ⋮
```

```
[D1(1:20) D2(1:20) D3(1:20)]./[D1(1) D2(1) D3(1)]  %This works in new versions
```

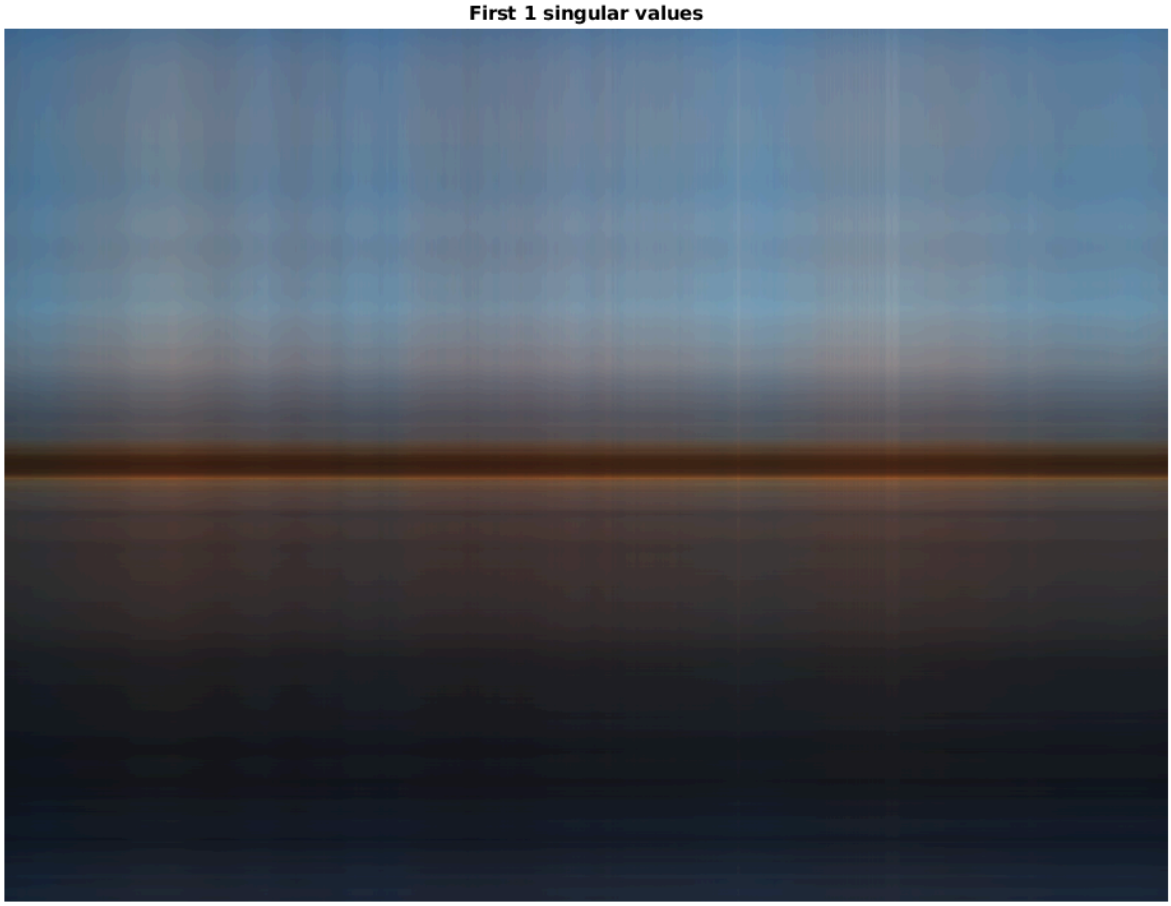
```
ans = 20×3
    1.0000    1.0000    1.0000
    0.1329    0.1297    0.1507
    0.1107    0.0731    0.0643
    0.0830    0.0556    0.0606
    0.0593    0.0380    0.0386
    0.0514    0.0321    0.0310
    0.0438    0.0279    0.0287
    0.0383    0.0253    0.0248
    0.0372    0.0216    0.0226
    0.0341    0.0209    0.0217
     ⋮
```

```
% The first few seem to dominate.
```

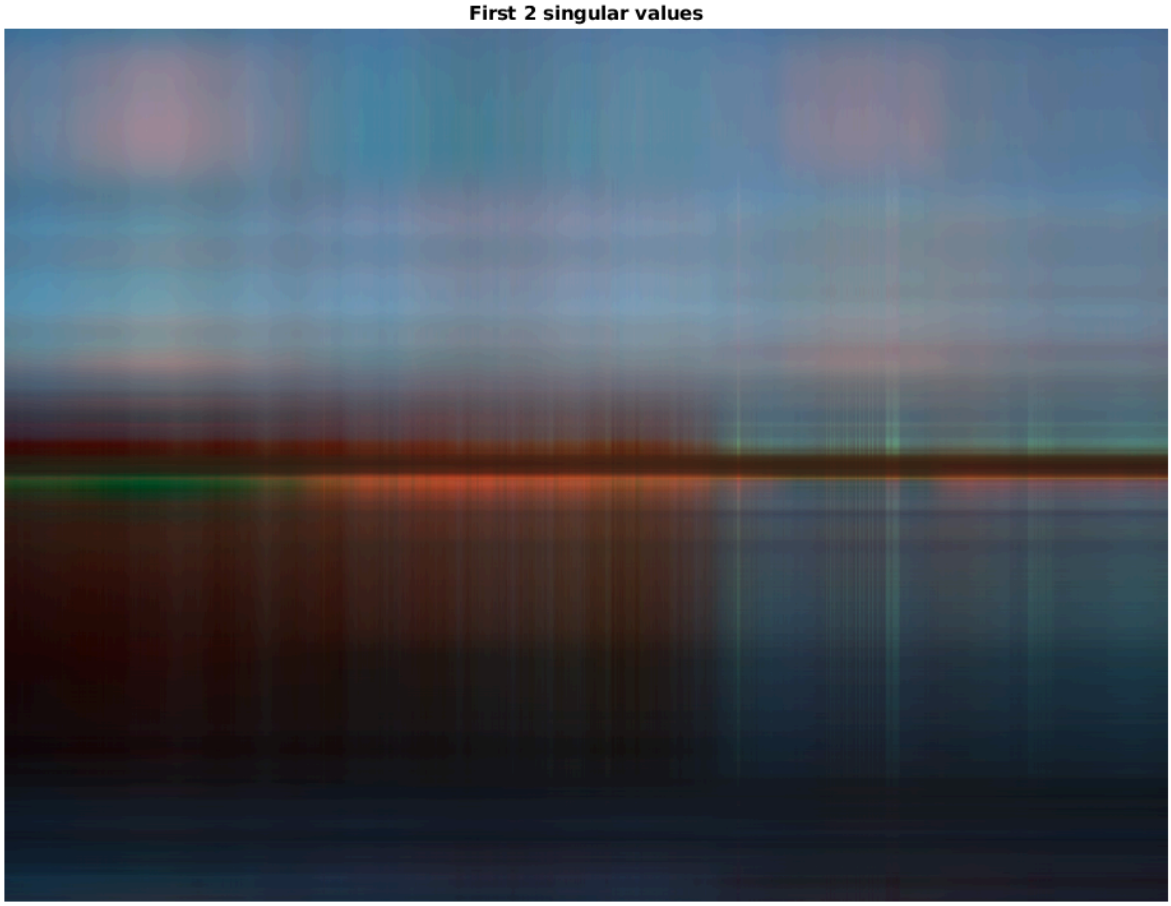
Show images in separate windows

```
close all
for k=[1:5 10 100];
   M1=u1(:,1:k)*s1(1:k,1:k)*v1(:,1:k)';
   M2=u2(:,1:k)*s2(1:k,1:k)*v2(:,1:k)';
   M3=u3(:,1:k)*s3(1:k,1:k)*v3(:,1:k)';
   kfig=k;
   if k==10
       kfig=6;
   elseif k==100
       kfig=7;
   end
   figure(kfig)
   imshow(cat(3,M1,M2,M3))
   title(['First ',num2str(k),' singular values'])
end
```

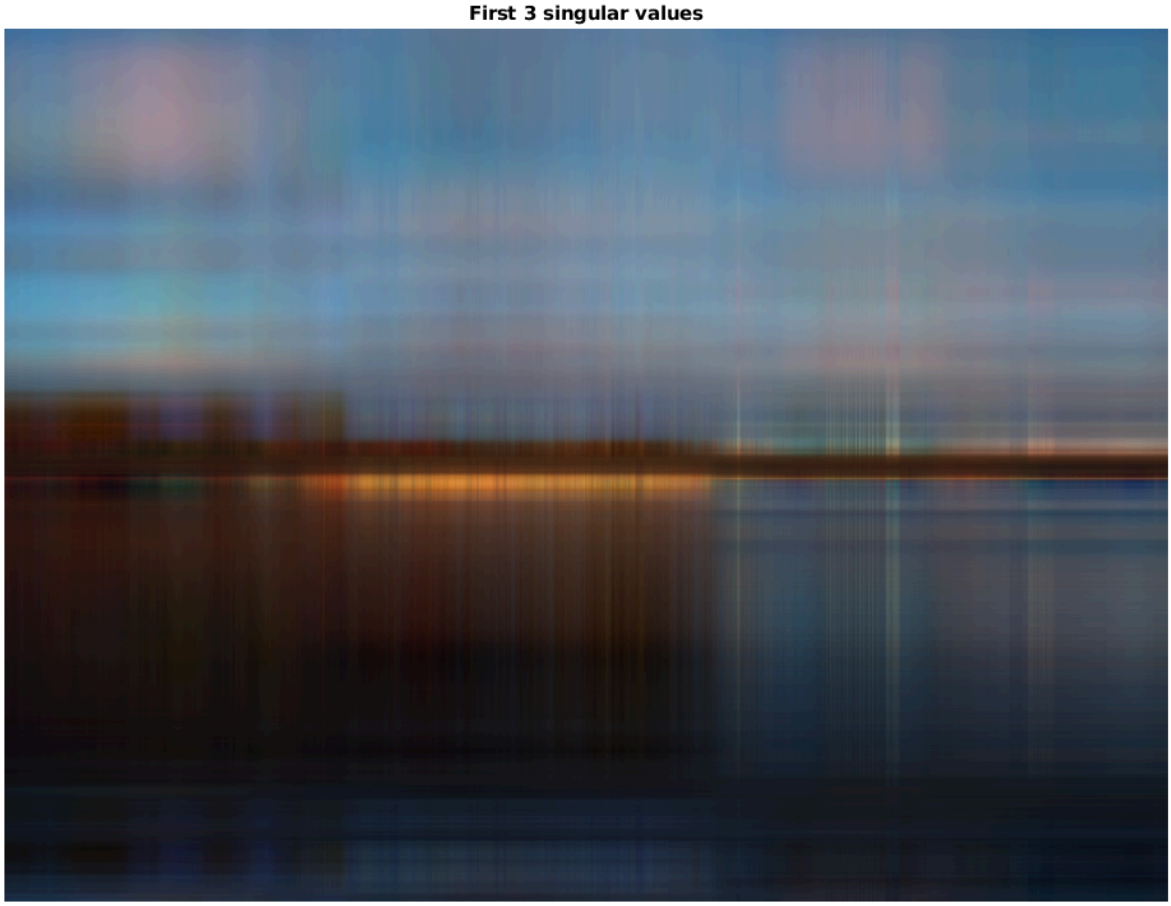
```
Warning: Image is too big to fit on screen; displaying at 25%
```



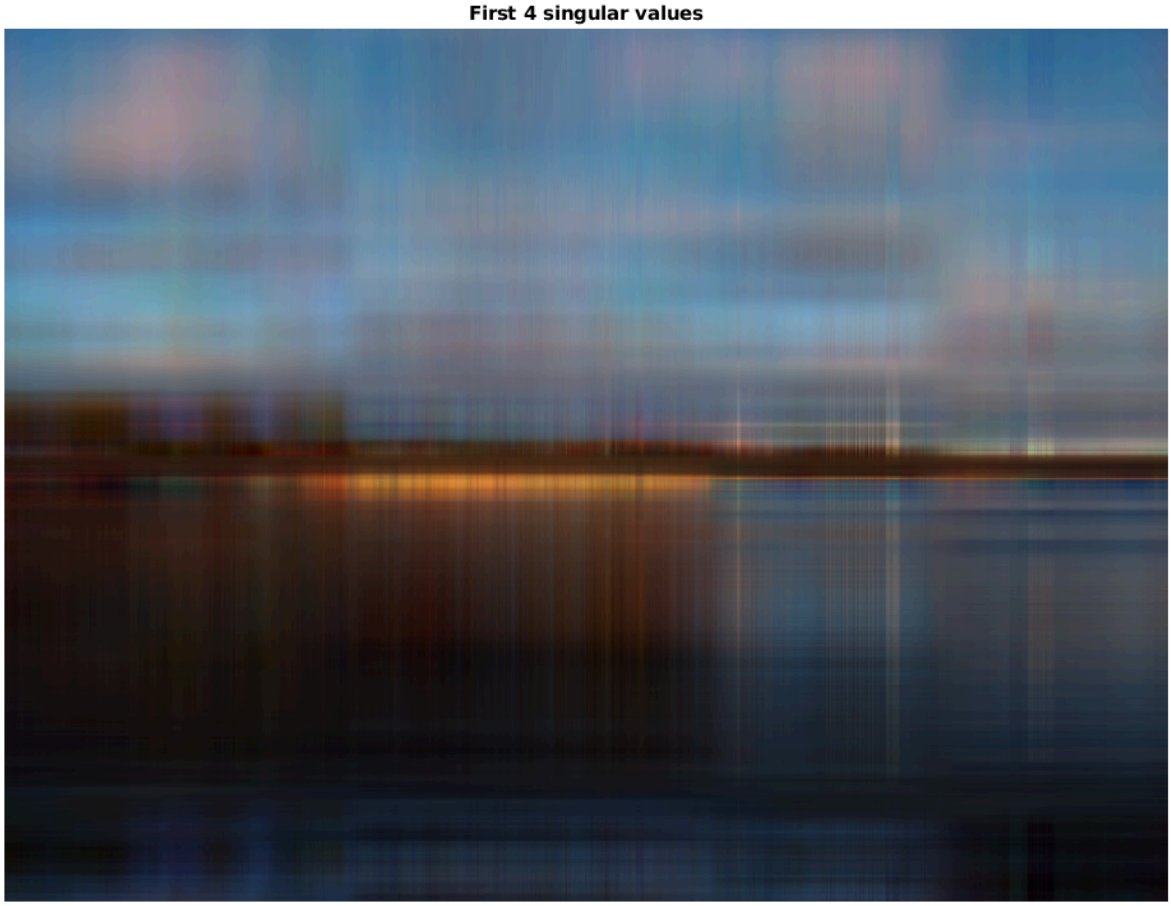

```
Warning: Image is too big to fit on screen; displaying at 25%
```



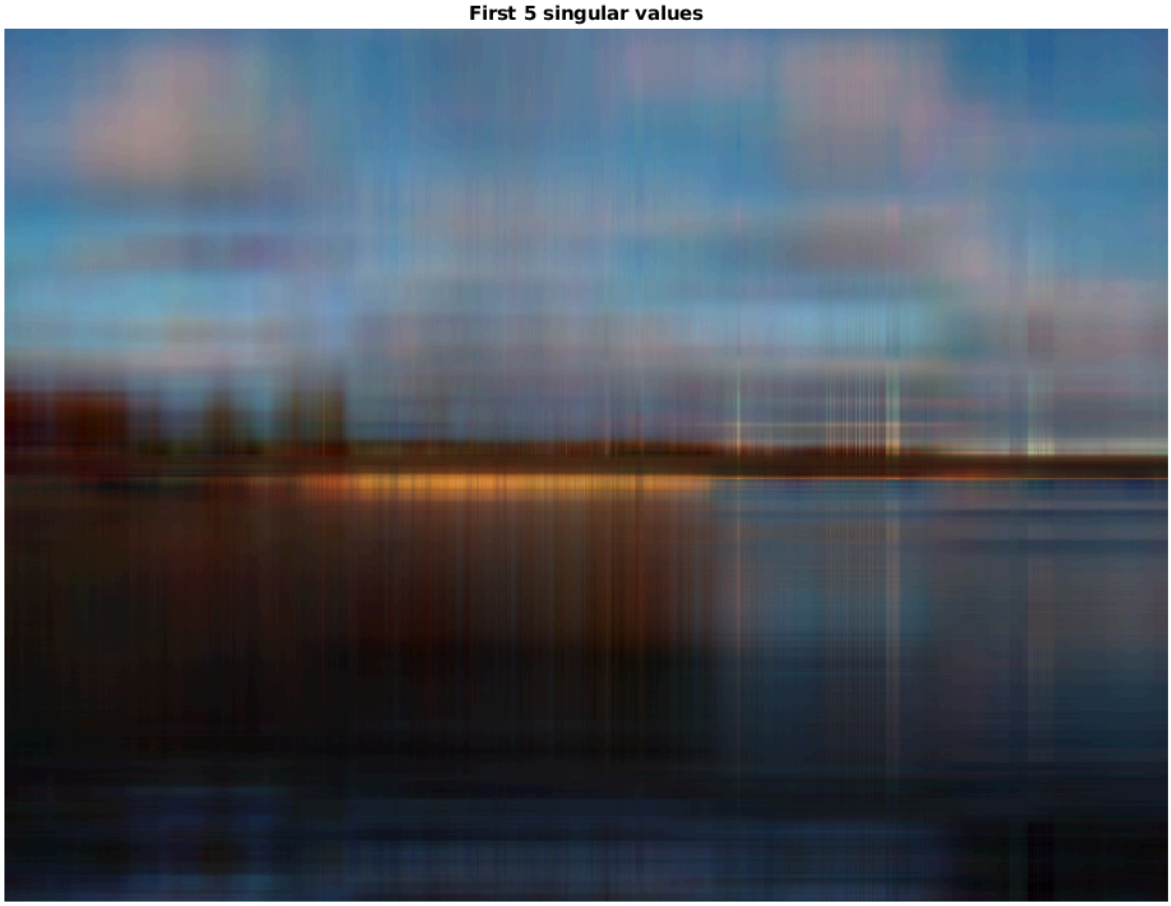

```
Warning: Image is too big to fit on screen; displaying at 25%
```



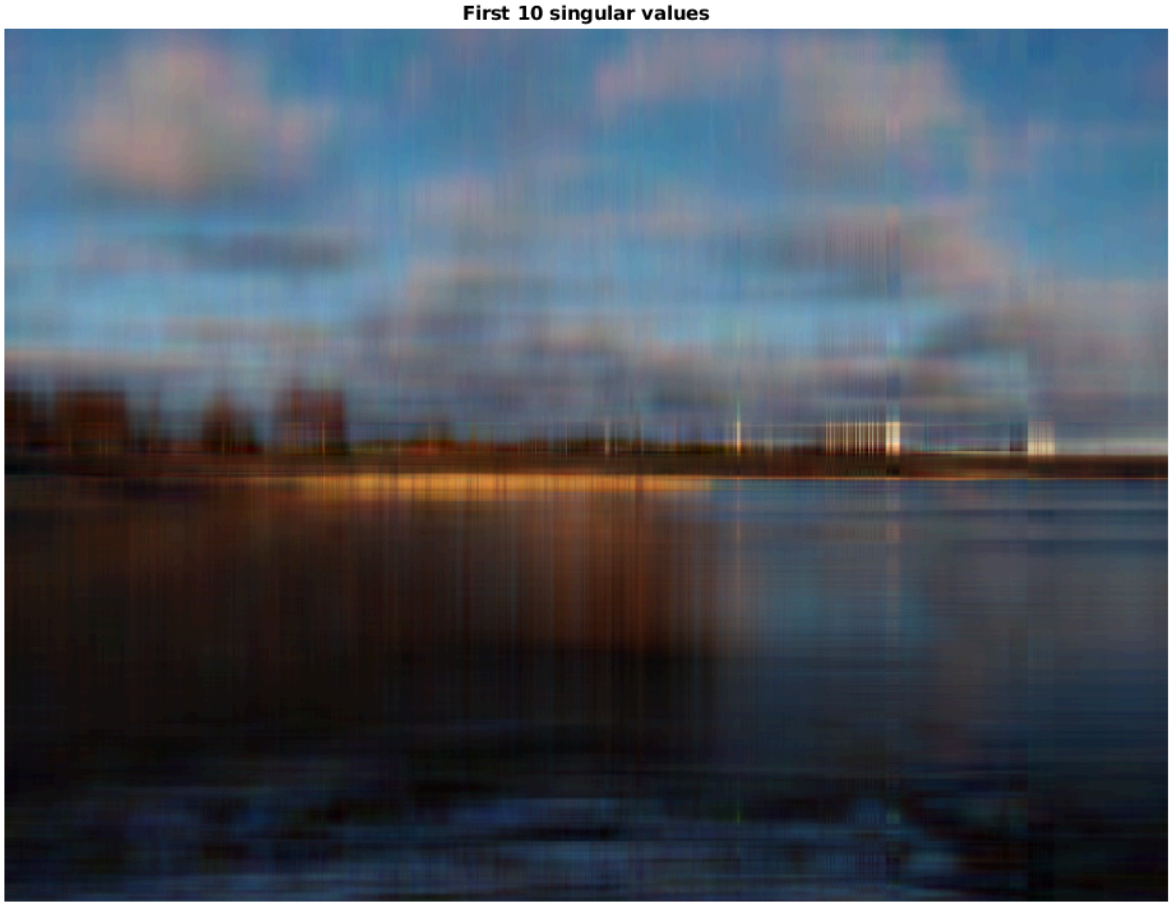

```
Warning: Image is too big to fit on screen; displaying at 25%
```




```
Warning: Image is too big to fit on screen; displaying at 25%
```




```
Warning: Image is too big to fit on screen; displaying at 25%
```

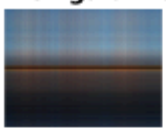



```
Warning: Image is too big to fit on screen; displaying at 25%
```

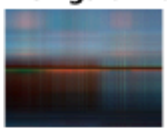
Use "matrix of windows" with subplot

```
close all
for k=1:8
   M1=u1(:,1:k)*s1(1:k,1:k)*v1(:,1:k)';
   M2=u2(:,1:k)*s2(1:k,1:k)*v2(:,1:k)';
   M3=u3(:,1:k)*s3(1:k,1:k)*v3(:,1:k)';
   subplot(4,2,k)
   imshow(cat(3,M1,M2,M3))
   title(['First ',num2str(k),' singular values'])
end
```

First 1 singular values



First 2 singular values

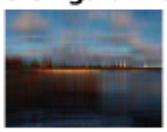


First 3 singular values

First 4 singular values

First 5 singular values

First 6 singular values



First 7 singular values

First 8 singular values

Is this useful?

Example: If you send a camera to Jupiter, you can do the svd's there and only need to send back to earth the n most significant singular values along with the corresponding columns of U and V. The value of n is of course "picture-dependent".