Eric Yang

304263623

Homework 2

1a.

```
function [U,S,V] = MySVDFunction(A)
    [Q L] = Jacobi(A' * A);
    V = Q;
    S = sqrt(L);
    U = A * V * pinv(S);
endfunction
```

```
>> C = gallery('chow',8)
C =

   1   1   0   0   0   0   0   0
   1   1   1   0   0   0   0   0
   1   1   1   1   0   0   0   0
   1   1   1   1   1   0   0   0
   1   1   1   1   1   1   0   0
   1   1   1   1   1   1   1   0
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1

>> [U,S,V] = MySVDFunction(C)
U =

  -1.4910e-001  -3.8989e-001   5.0503e-001  -5.0729e-001   4.3414e-001  -3.1327e-001   1.6354e-001  -3.6465e-017
  -2.1954e-001  -4.8243e-001   3.6807e-001   1.3444e-002  -3.9170e-001   5.3795e-001  -3.7495e-001   5.1493e-017
  -2.8392e-001  -4.4828e-001  -5.2736e-002   5.1366e-001  -2.7658e-001  -3.4152e-001   5.0861e-001  -1.7897e-017
  -3.4047e-001  -2.9638e-001  -4.3288e-001   2.2977e-001   5.0295e-001  -1.2225e-001  -5.3681e-001  -9.6193e-018
  -3.8763e-001  -6.6653e-002  -4.7919e-001  -4.0486e-001   7.4273e-002   4.9032e-001   4.5370e-001  -1.6295e-017
  -4.2409e-001   1.8058e-001  -1.5596e-001  -4.2147e-001  -5.3282e-001  -4.7457e-001  -2.7651e-001  -5.9626e-017
  -4.4885e-001   3.8039e-001   2.8755e-001   2.0529e-001   1.4005e-001   8.7328e-002   4.1989e-002  -7.0711e-001
  -4.4885e-001   3.8039e-001   2.8755e-001   2.0529e-001   1.4005e-001   8.7328e-002   4.1989e-002   7.0711e-001

S =

Diagonal Matrix

  6.0208e+000            0            0            0            0            0            0            0
            0  1.9513e+000            0            0            0            0            0            0
            0            0  1.1387e+000            0            0            0            0            0
            0            0            0  8.0938e-001            0            0            0            0
            0            0            0            0  6.4520e-001            0            0            0
            0            0            0            0            0  5.5752e-001            0            0
            0            0            0            0            0            0  5.1349e-001            0
            0            0            0            0            0            0            0  6.5009e-020

V =

  -4.4885e-001  -3.8039e-001   2.8755e-001  -2.0529e-001   1.4005e-001  -8.7328e-002   4.1989e-002   7.0711e-001
  -4.4885e-001  -3.8039e-001   2.8755e-001  -2.0529e-001   1.4005e-001  -8.7328e-002   4.1989e-002  -7.0711e-001
  -4.2409e-001  -1.8058e-001  -1.5596e-001   4.2147e-001  -5.3282e-001   4.7457e-001  -2.7651e-001   5.9882e-017
  -3.8763e-001   6.6653e-002  -4.7919e-001   4.0486e-001   7.4273e-002  -4.9032e-001   4.5370e-001   1.3425e-016
  -3.4047e-001   2.9638e-001  -4.3288e-001  -2.2977e-001   5.0295e-001   1.2225e-001  -5.3681e-001  -5.3017e-017
  -2.8392e-001   4.4828e-001  -5.2736e-002  -5.1366e-001  -2.7658e-001   3.4152e-001   5.0861e-001   6.6570e-017
  -2.1954e-001   4.8243e-001   3.6807e-001  -1.3444e-002  -3.9170e-001  -5.3795e-001  -3.7495e-001  -2.2335e-017
  -1.4910e-001   3.8989e-001   5.0503e-001   5.0729e-001   4.3414e-001   3.1327e-001   1.6354e-001  -5.6379e-017
```

```
>> F = hadamard(8)
F =

   1   1   1   1   1   1   1   1
   1  -1   1  -1   1  -1   1  -1
   1   1  -1  -1   1   1  -1  -1
   1  -1  -1   1   1  -1  -1   1
   1   1   1   1  -1  -1  -1  -1
   1  -1   1  -1  -1   1  -1   1
   1   1  -1  -1  -1  -1   1   1
   1  -1  -1   1  -1   1   1  -1

>> [U,S,V] = MySVDFunction(F)
U =

  -0.0529648   0.2431232  -0.3535534   0.3535534   0.7731799   0.1258042   0.2695819   0.0421619
  -0.0314501  -0.4585753  -0.3535534  -0.3535534  -0.1309705   0.1739280   0.7009130   0.0060231
   0.1551046  -0.3222509  -0.3535534   0.3535534  -0.1583457   0.5913303  -0.3774147  -0.3238984
   0.2602123   0.7026258  -0.3535534  -0.3535534  -0.2175094   0.0936082   0.0539164  -0.3600372
  -0.3778508   0.2630985  -0.3535534   0.3535534  -0.4845516   0.0242451   0.0539164   0.5474696
   0.4598505  -0.1326260  -0.3535534  -0.3535534   0.1835417  -0.0649599  -0.3774147   0.5836083
   0.2757110  -0.1839708  -0.3535534   0.3535534  -0.1302826  -0.7413795   0.0539164  -0.2657330
  -0.6886127  -0.1114245  -0.3535534  -0.3535534   0.1649382  -0.2025763  -0.3774147  -0.2295942

S =

Diagonal Matrix

   2.8284        0        0        0        0        0        0        0
        0   2.8284        0        0        0        0        0        0
        0        0   2.8284        0        0        0        0        0
        0        0        0   2.8284        0        0        0        0
        0        0        0        0   2.8284        0        0        0
        0        0        0        0        0   2.8284        0        0
        0        0        0        0        0        0   2.8284        0
        0        0        0        0        0        0        0   2.8284

V =

  -0.00000   0.00000  -1.00000   0.00000   0.00000  -0.00000  -0.00000   0.00000
  -0.00000   0.00000  -0.00000   1.00000   0.00000  -0.00000  -0.00000   0.00000
  -0.00171  -0.06009  -0.00000  -0.00000   0.24126   0.18315   0.45750   0.83386
  -0.60756   0.77600  -0.00000  -0.00000   0.16692   0.02905  -0.00000   0.00000
   0.23398   0.11662  -0.00000  -0.00000   0.18834   0.69627   0.45750  -0.44954
  -0.08954  -0.22852  -0.00000  -0.00000   0.68117   0.31791  -0.60999   0.05111
  -0.35166  -0.36122  -0.00000  -0.00000   0.47862  -0.45553   0.45750  -0.31618
   0.66667   0.44488  -0.00000  -0.00000   0.43058  -0.41502  -0.00000  -0.00000
```

```
>> H = hilb(8)
H =

   1.000000   0.500000   0.333333   0.250000   0.200000   0.166667   0.142857   0.125000
   0.500000   0.333333   0.250000   0.200000   0.166667   0.142857   0.125000   0.111111
   0.333333   0.250000   0.200000   0.166667   0.142857   0.125000   0.111111   0.100000
   0.250000   0.200000   0.166667   0.142857   0.125000   0.111111   0.100000   0.090909
   0.200000   0.166667   0.142857   0.125000   0.111111   0.100000   0.090909   0.083333
   0.166667   0.142857   0.125000   0.111111   0.100000   0.090909   0.083333   0.076923
   0.142857   0.125000   0.111111   0.100000   0.090909   0.083333   0.076923   0.071429
   0.125000   0.111111   0.100000   0.090909   0.083333   0.076923   0.071429   0.066667

>> [U,S,V] = MySVDFunction(H)
U =

  -7.2027e-001  6.2948e-001  2.7755e-001  -8.6517e-002  2.0656e-002  -3.8106e-003  5.2387e-004  4.7154e-005
  -4.3253e-001  -1.2567e-001  -6.4493e-001  5.5014e-001  -2.6613e-001  8.5791e-002  -1.8809e-002  -2.5346e-003
  -3.1884e-001  -2.8642e-001  -3.3515e-001  -3.3625e-001  6.2407e-001  -4.1879e-001  1.5800e-001  3.3173e-002
  -2.5524e-001  -3.2757e-001  -5.1307e-002  -4.5925e-001  -3.1954e-002  5.7971e-001  -4.9454e-001  -1.7987e-001
  -2.1386e-001  -3.3209e-001  1.4402e-001  -2.7565e-001  -4.1938e-001  1.2321e-001  5.6650e-001  4.8504e-001
  -1.8452e-001  -3.2354e-001  2.7286e-001  -1.1272e-002  -3.6553e-001  -4.1851e-001  7.4693e-002  -6.8728e-001
  -1.6251e-001  -3.1027e-001  3.5704e-001  2.4896e-001  -1.8007e-002  -3.5236e-001  -5.6884e-001  4.8969e-001
  -1.4534e-001  -2.9562e-001  4.1158e-001  4.7728e-001  4.7797e-001  4.0826e-001  2.8290e-001  -1.3831e-001

S =

Diagonal Matrix

  1.6959e+000            0            0            0            0            0            0            0
            0  2.9813e-001            0            0            0            0            0            0
            0            0  2.6213e-002            0            0            0            0            0
            0            0            0  1.4677e-003            0            0            0            0
            0            0            0            0  5.4369e-005            0            0            0
            0            0            0            0            0  1.2943e-006            0            0
            0            0            0            0            0            0  1.7989e-008            0
            0            0            0            0            0            0            0  1.1115e-010

V =

  -7.2027e-001  6.2948e-001  2.7755e-001  -8.6517e-002  2.0656e-002  -3.8106e-003  5.2387e-004  4.7154e-005
  -4.3253e-001  -1.2567e-001  -6.4493e-001  5.5014e-001  -2.6613e-001  8.5791e-002  -1.8809e-002  -2.5346e-003
  -3.1884e-001  -2.8642e-001  -3.3515e-001  -3.3625e-001  6.2407e-001  -4.1879e-001  1.5800e-001  3.3173e-002
  -2.5524e-001  -3.2757e-001  -5.1307e-002  -4.5925e-001  -3.1954e-002  5.7971e-001  -4.9454e-001  -1.7987e-001
  -2.1386e-001  -3.3209e-001  1.4402e-001  -2.7565e-001  -4.1938e-001  1.2321e-001  5.6650e-001  4.8504e-001
  -1.8452e-001  -3.2354e-001  2.7286e-001  -1.1272e-002  -3.6553e-001  -4.1851e-001  7.4693e-002  -6.8728e-001
  -1.6251e-001  -3.1027e-001  3.5704e-001  2.4896e-001  -1.8007e-002  -3.5236e-001  -5.6884e-001  4.8969e-001
  -1.4534e-001  -2.9562e-001  4.1158e-001  4.7728e-001  4.7797e-001  4.0826e-001  2.8290e-001  -1.3831e-001
```

```
>> P = pascal(8)
P =

     1     1     1     1     1     1     1     1
     1     2     3     4     5     6     7     8
     1     3     6    10    15    21    28    36
     1     4    10    20    35    56    84   120
     1     5    15    35    70   126   210   330
     1     6    21    56   126   252   462   792
     1     7    28    84   210   462   924  1716
     1     8    36   120   330   792  1716  3432

>> [U,S,V] = MySVDFunction(P)
U =

  -3.6516e-004  -1.0119e-002  -1.1148e-001   5.2794e-001  -7.3790e-001   3.8525e-001   1.2343e-001  -2.4614e-002
  -2.6245e-003  -4.6930e-002  -2.8791e-001   5.6215e-001   4.7819e-002  -6.0972e-001  -4.4905e-001   1.5229e-001
  -1.0958e-002  -1.3125e-001  -4.6860e-001   3.0411e-001   4.0848e-001   1.4752e-002   5.7955e-001  -4.0942e-001
  -3.4536e-002  -2.7466e-001  -5.2535e-001  -1.0826e-001   1.9278e-001   4.4310e-001  -1.4125e-001   6.1822e-001
  -9.0915e-002  -4.5540e-001  -3.2980e-001  -3.5838e-001  -2.4971e-001  -4.6656e-004  -4.0669e-001  -5.6514e-001
  -2.1067e-001  -5.8010e-001   1.2031e-001  -1.4901e-001  -2.7679e-001  -4.3622e-001   4.6691e-001   3.1229e-001
  -4.4345e-001  -4.2854e-001   4.8859e-001   3.6978e-001   3.2129e-001   2.9839e-001  -2.0662e-001  -9.6476e-002
  -8.6567e-001   4.2129e-001  -2.1711e-001  -1.1698e-001  -8.3693e-002  -6.2824e-002   3.4536e-002   1.2842e-002

S =

Diagonal Matrix

  4.5437e+003            0            0            0            0            0            0            0
            0  1.4880e+002            0            0            0            0            0            0
            0            0  1.1943e+001            0            0            0            0            0
            0            0            0  1.9535e+000            0            0            0            0
            0            0            0            0  5.1189e-001            0            0            0
            0            0            0            0            0  8.3730e-002            0            0
            0            0            0            0            0            0  6.7202e-003            0
            0            0            0            0            0            0            0  2.2009e-004

V =

  -3.6516e-004  -1.0119e-002  -1.1148e-001   5.2794e-001  -7.3790e-001   3.8525e-001   1.2343e-001  -2.4614e-002
  -2.6245e-003  -4.6930e-002  -2.8791e-001   5.6215e-001   4.7819e-002  -6.0972e-001  -4.4905e-001   1.5229e-001
  -1.0958e-002  -1.3125e-001  -4.6860e-001   3.0411e-001   4.0848e-001   1.4752e-002   5.7955e-001  -4.0942e-001
  -3.4536e-002  -2.7466e-001  -5.2535e-001  -1.0826e-001   1.9278e-001   4.4310e-001  -1.4125e-001   6.1822e-001
  -9.0915e-002  -4.5540e-001  -3.2980e-001  -3.5838e-001  -2.4971e-001  -4.6656e-004  -4.0669e-001  -5.6514e-001
  -2.1067e-001  -5.8010e-001   1.2031e-001  -1.4901e-001  -2.7679e-001  -4.3622e-001   4.6691e-001   3.1229e-001
  -4.4345e-001  -4.2854e-001   4.8859e-001   3.6978e-001   3.2129e-001   2.9839e-001  -2.0662e-001  -9.6476e-002
  -8.6567e-001   4.2129e-001  -2.1711e-001  -1.1698e-001  -8.3693e-002  -6.2824e-002   3.4536e-002   1.2842e-002
```

1b.

Chow = 9.2615 x 10^19

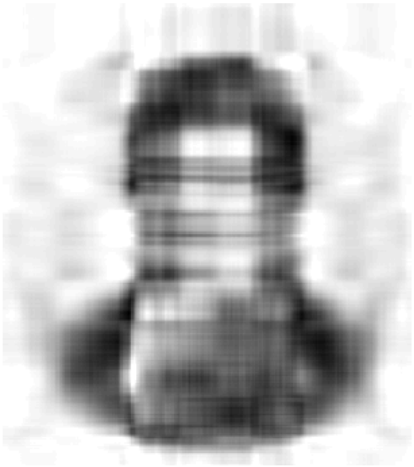
Hadamard = 1

Hilbert = 1.5258 x 10^10

Pascal = 2.0645 x 10^7

1c.

K = 5



K=10

K = 20

K = 50

2a.

```
>> a = quaternion(3,2,5,4);
>> b = quaternion(4,5,3,1);
>> myQMult(a,b)
ans = -17 + 16i + 47j + 0k
```

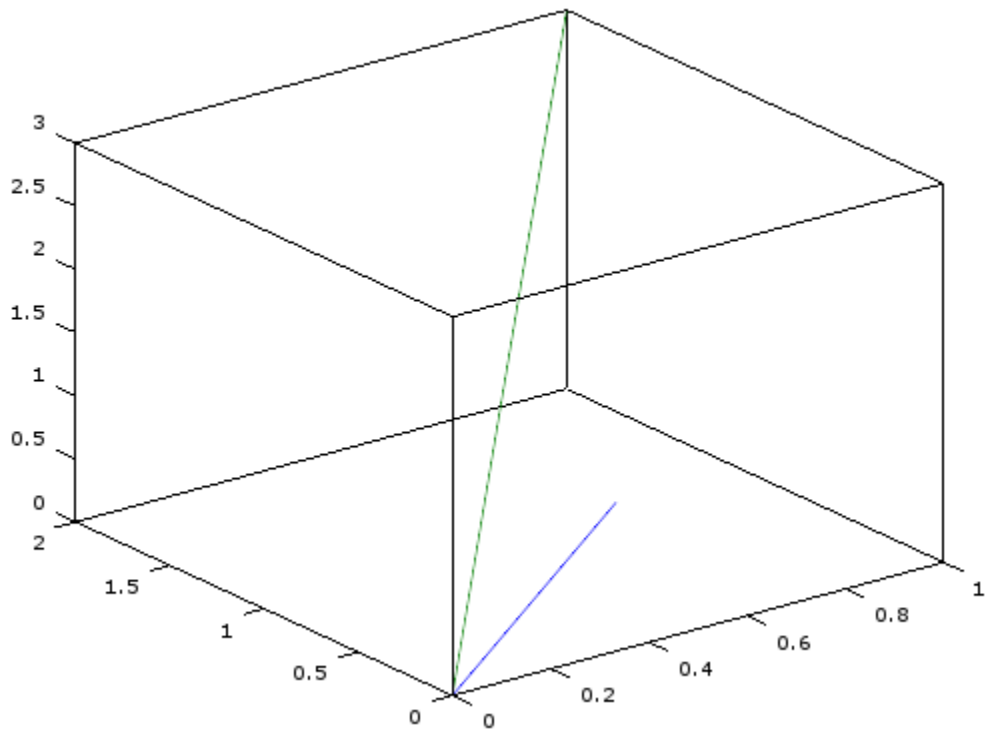
2b.

```
>> q = quaternion(8,-2,3,-1);
>> q_inv = myQInv(q)
q_inv = 0.1026 + 0.02564i - 0.03846j + 0.01282k
>> q*q_inv
ans = 1 + 0i + 0j + 0k
```

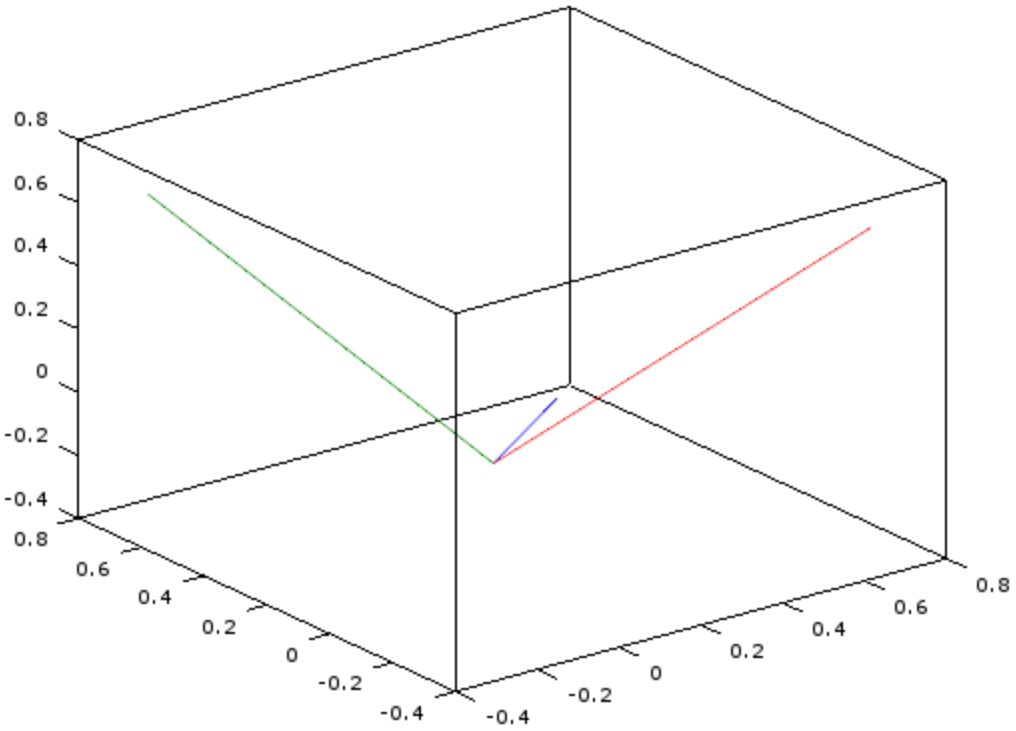
2c.

```
>> r = myQRot(axis, angle)
r = 0.9659 + 0i + 0j + 0.2588k
>> v = quaternion(1,0,0)
v = 0 + 1i + 0j + 0k
>> vr = r*v*conj(r)
vr = 0 + 0.866i + 0.5j + 0k
```

2d.



2e.



Plots are the same for negated p

3a.

```
function A = Build()
  A = [];
  for i=[1:13]
    I = imread (["./faces/basis/f" num2str(i) ".jpg"]);
    IVec = reshape(I, 148*100, 1);
    IVec = double(IVec);
    IVec = IVec / norm(double(IVec));
    A=[A IVec];
  endfor
endfunction
```

```
>> A = Build()
A =

 Columns 1 through 5:

   0.011119   0.010958   0.010867   0.011154   0.011115
   0.011119   0.010958   0.010867   0.011154   0.011115
   0.011119   0.010958   0.010867   0.011154   0.011115
   0.011119   0.010958   0.010867   0.011154   0.011115
   0.011119   0.010958   0.010867   0.011154   0.011115
   0.011119   0.010958   0.010867   0.011154   0.011115
   0.011119   0.010958   0.010867   0.011154   0.011115
   0.011119   0.010958   0.010867   0.011154   0.011115
   0.011119   0.010958   0.010867   0.011154   0.011115
   0.011119   0.010958   0.010867   0.011154   0.011115
   0.011119   0.010958   0.010867   0.011154   0.011115
   0.011119   0.010958   0.010867   0.011154   0.011115
   0.011119   0.010958   0.010867   0.011154   0.011115
   0.011119   0.010958   0.010867   0.011154   0.011115
   0.011119   0.010958   0.010867   0.011154   0.011115
   0.011119   0.010958   0.010867   0.011154   0.011115
   0.011119   0.010958   0.010867   0.011154   0.011115
   0.011119   0.010958   0.010867   0.011154   0.011115
   0.011119   0.010958   0.010867   0.011154   0.011115
   0.011119   0.010958   0.010867   0.011154   0.011115
   0.011119   0.010958   0.010867   0.011154   0.011115
   0.011119   0.010958   0.010867   0.011154   0.011115
```

3b.

```
IVec =

   0.010659
   0.011096
   0.011140
   0.011052
   0.011140
   0.011140
   0.011096
   0.011140
   0.011140
   0.011140
   0.011140
   0.011140
   0.011140
   0.011140
   0.011140
   0.011140
   0.011140
   0.011140
   0.011140
   0.011140
   0.011140
   0.011140
   0.011140
```

3c.

```
>> x = pinv(A)*b
x =

   0.0188724
   0.0698072
  -0.0303447
   0.3625146
   0.1527518
   0.1384549
  -0.0722029
  -0.0411344
   0.1665731
  -0.1843220
   0.1346673
   0.0065293
   0.2612789
   0.0188724
```

```
>> v = A*x;
>> v=v*norm(double(b));
>> x = pinv(A)*b;
>> v = A*x;
>> v=v*norm(double(b));
>> v = uint8(v);
>> v = reshape(v,[148 100]);
>> imshow(v);
```

```
>> (v-b)'*(v-b)
ans =   0.018380
```

3d.

```
>> x = pinv(A)*b
x =

   0.0188724
   0.0698072
  -0.0303447
   0.3625146
   0.1527518
   0.1384549
  -0.0722029
  -0.0411344
   0.1665731
  -0.1843220
   0.1346673
   0.0065293
   0.2612789
   0.0188724
```

Basis images f1, f2, f3

3e.

Correlation between f1 and f3 is highest at 0.78896

```
Columns 1 through 6:

   1.00000   0.77949   0.78896   0.73634   0.64674   0.72517
   0.77949   1.00000   0.85108   0.75760   0.73039   0.75224
   0.78896   0.85108   1.00000   0.81834   0.75684   0.76698
   0.73634   0.75760   0.81834   1.00000   0.82415   0.85383
   0.64674   0.73039   0.75684   0.82415   1.00000   0.79217
   0.72517   0.75224   0.76698   0.85383   0.79217   1.00000
   0.75183   0.82252   0.88610   0.75510   0.75128   0.72583
   0.68633   0.76354   0.77228   0.66198   0.69190   0.68576
   0.73361   0.81203   0.89138   0.80377   0.81568   0.74767
   0.68953   0.77943   0.82430   0.75168   0.82210   0.73902
   0.74744   0.81891   0.84917   0.76101   0.75765   0.71614
   0.68379   0.71930   0.79688   0.76182   0.76503   0.71900
   0.70912   0.76944   0.81195   0.76983   0.78012   0.72456
```

3f.

```
>> comp1 = U(:, 1)
comp1 =

  -0.25646
  -0.27506
  -0.28757
  -0.27213
  -0.26907
  -0.26363
  -0.28798
  -0.26554
  -0.29370
  -0.28098
  -0.28802
  -0.27700
  -0.28562
```

```
>> comp2 = U(:, 2)
comp2 =

   0.230097
   0.071657
   0.017735
   0.467931
   0.268780
   0.536558
  -0.203901
  -0.325722
  -0.154472
  -0.088130
  -0.257618
  -0.213632
  -0.266842
```

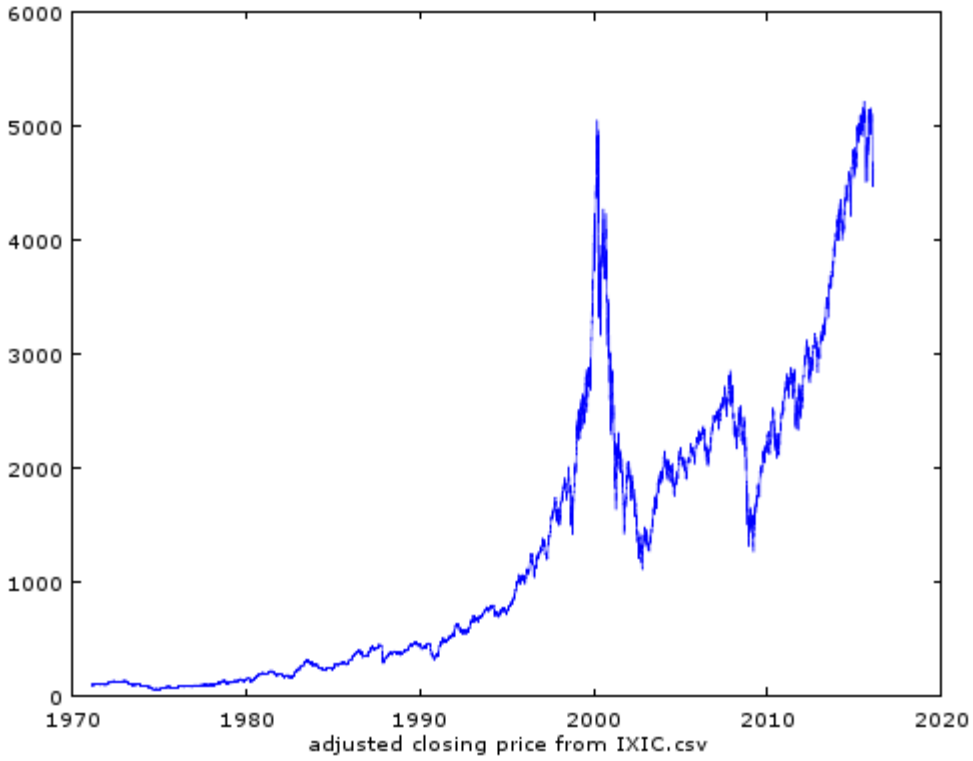
3g.

Images 7 and 11 have (x,y) positions closest

4a.

```
time =

   1971.1
   1971.1
   1971.1
   1971.1
   1971.1
   1971.1
   1971.1
   1971.1
   1971.1
   1971.1
   1971.1
   1971.1
   1971.1
   1971.2
   1971.2
   1971.2
   1971.2
   1971.2
   1971.2
   1971.2
   1971.2
```

```
ixic =

    100.000
    100.840
    100.760
    100.690
    101.450
    102.050
    102.190
    101.740
    101.420
    100.700
     99.680
     99.720
    100.640
    101.230
    101.340
    101.780
    101.840
    102.070
    102.780
```



4b.

Coefficients of polyfit for 1971-02-05 – 2000-03-09:

```
p =

 Columns 1 through 4:

  4.4897e-002  -3.5616e+002   1.0595e+006  -1.4008e+009

 Column 5:

  6.9454e+011
```

Coefficients of polyfit for 2009-03-10 – 2015-11-04:

```
>> polyfit(time2,ixic2,4)
warning: matrix singular to machine precision, rcond = 3.47181
e-026
warning: called from
    polyfit at line 119 column 5
ans =

 Columns 1 through 4:

  -1.3514e+001   1.0880e+005  -3.2845e+008   4.4070e+011

 Column 5:

  -2.2174e+014
```

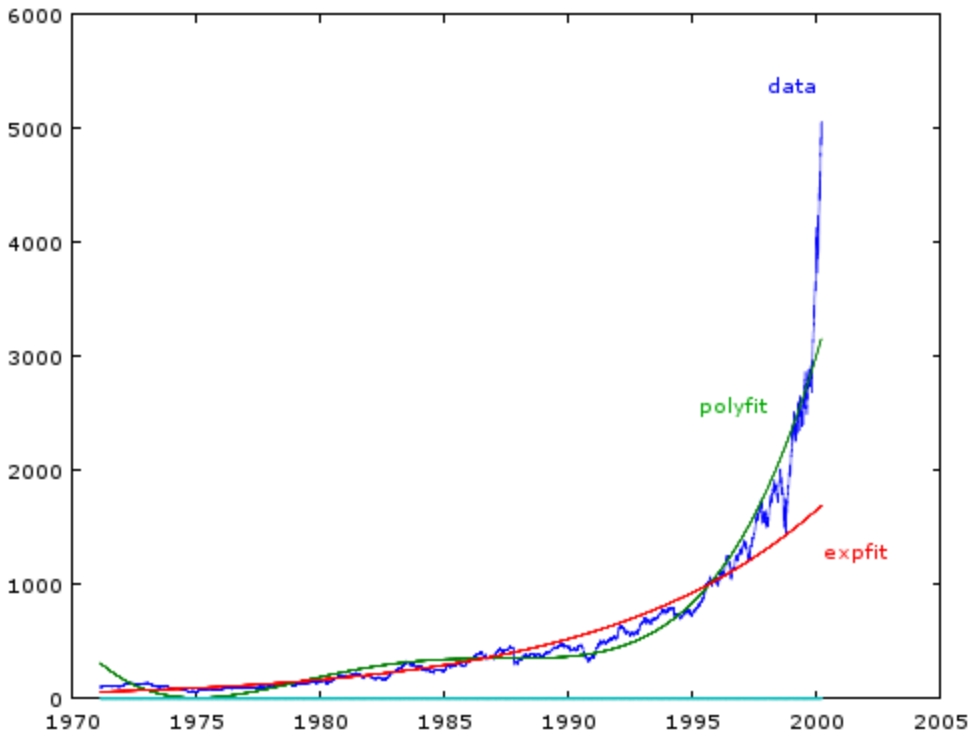
Coefficients of exp fit for 1971-02-05 – 2000-03-09:

```
>> x1 = pinv(time11)*b1
x1 =

     0.11506
  -222.71210
```

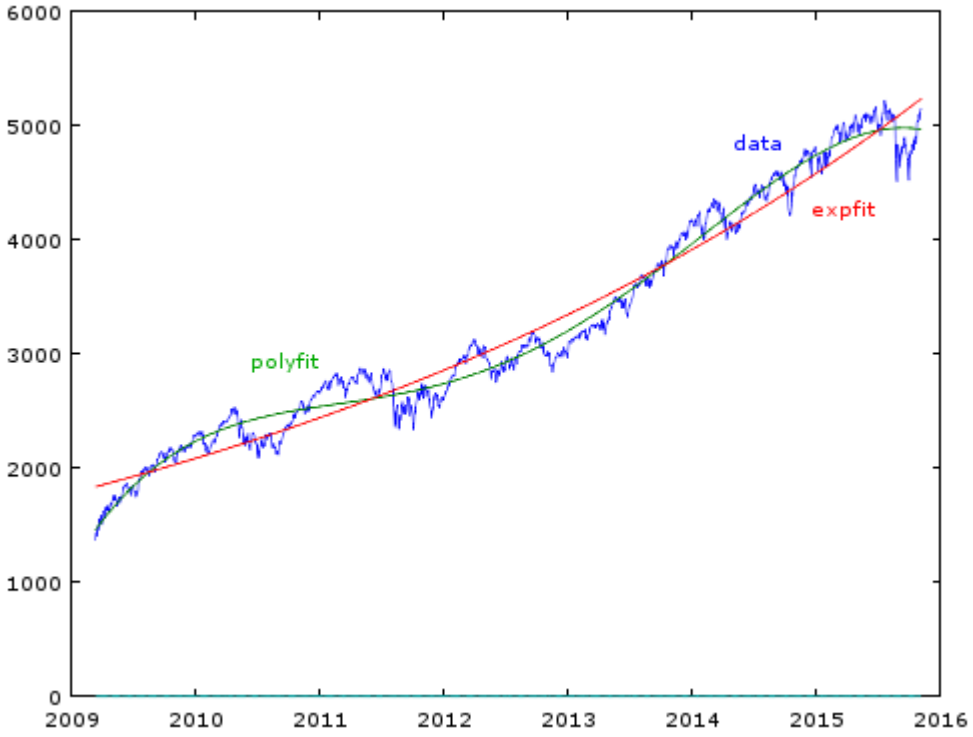
Coefficients of exp fit for 2009-03-10 – 2015-11-04:

```
>> x2 = pinv(time22)*b2
x2 =

     0.15741
  -308.74903
```

Graph of data set 1971-02-05 – 2000-03-09 with expfit and polyfit:



Graph of data set 2009-03-10 – 2015-11-04 with expfit and polyfit:



Squared error of polyfit for 1971-02-05 – 2000-03-09: 1.6906e+008

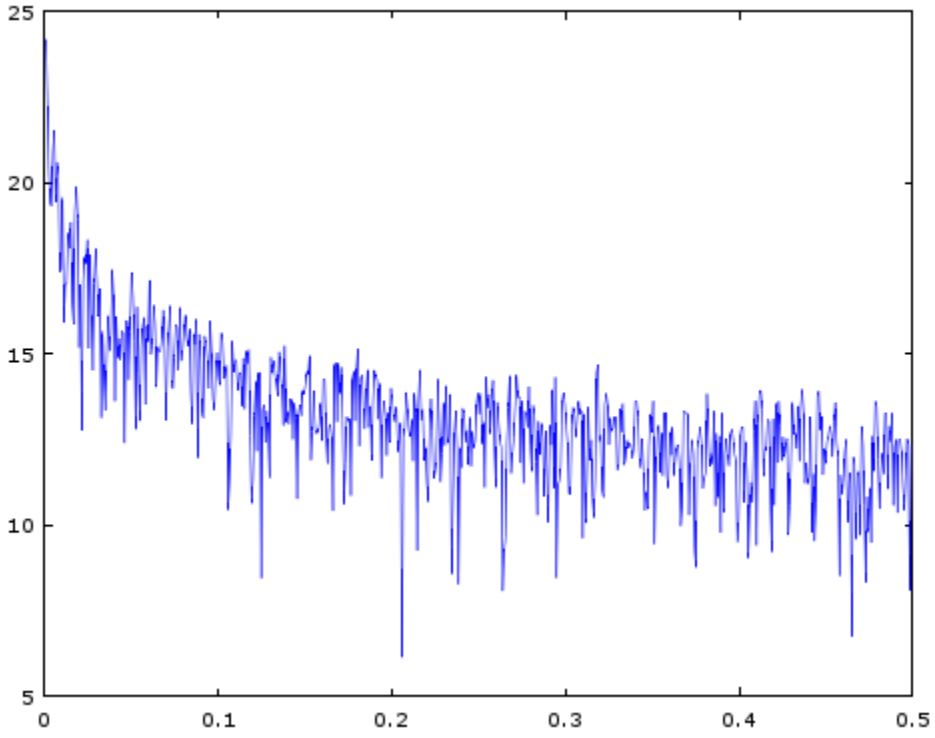
Squared error of expfit for 1971-02-05 – 2000-03-09: 6.0131e+009

Squared error of polyfit for 2009-03-10 – 2015-11-04: 2.9578e+007

Squared error of expfit for 2009-03-10 – 2015-11-04: 1.9377e+010

Polyfit generally has less squared error

5.



The frequency is about 0.018283 per day or 0.219396 per month