```
% gda14_05
% This code parallels "Method Summary 4, Factor Analysis".
clear all;

% This is a hypothetical scenario with synthetic data, only,
% in which the object is to determine the source(s) of a pollutant
% from observations of its concentration in dust collected
% on a 2D grid of sampling sites.  The pollutant is an element
% with four isotopes.  The presumption is that each source
% (two factories and a natural background source) has a distinct
% isotopic pattern and that a dust sample at any given location
% is a mixture of these patterns.  The pollutant emitted from the
% factories is concentrated near the factories; the natural
% pollutant is spread more uniformly across the area.

% This section creates the synthetic data.

% sample locations
Nx=20;
Ny=20;
x = [1:Nx]';
y = [1:Ny]';

% The factories will be numbered 1 and 2; the natural source 3

% factory locations, both as indices and (x,y) positions
ix1=3;  x1=x(ix1); iy1=3;  y1=y(iy1);
ix2=17; x2=x(ix2); iy2=17; y2=y(iy2);

% the factors are the fraction of the 4 isotopes for each source
% note that isotope 1 is present in a smaller
% fraction than the other three.  However, the
% measurement technique for determining it is also better.
f1 = [0.0104, 0.2084, 0.1762, 0.6050]';
f2 = [0.0165, 0.2748, 0.2365, 0.4722]';
f3 = [0.0140, 0.2410, 0.2210, 0.5240]';
w = [10, 1, 1, 1];

% We now build the sample matrix S(i,i). It gives
% the amount of isotope j observed at station i
% measured, say, in picograms per square meter per day
% of dust deposited on a surface exposed to the
% atmosphere

N = Nx*Ny; % number of samples
M = 4;     % number of isotopes
Strue = zeros( N, M ); % true (noise free) samples matrix
Sobs = zeros( N, M );  % observed (noisy) sample matrix
jofixiy =zeros(Nx,Ny); % sample number as function of (ix,iy)

j=1;
v=0.1;
sigmaS = 0.01;
for ix=[1:Nx]
for iy=[1:Ny]
    R1 = sqrt( (x(ix)-x1)^2 + (y(iy)-y1)^2 ); % distance to factory 1
    R2 = sqrt( (x(ix)-x2)^2 + (y(iy)-y2)^2 ); % distance to factory 2
    C1 = 800*(1/((R1/10)+1))+random('uniform',0,v,1); % loading 1 declines with R1
    C2 = 200*(1/((R2/10)+1))+random('uniform',0,v,1); % loading 2 declines with R2
```

```
    C3 = 10+random('uniform',0,v,1); % loading 3 is more-or-less constant
    Strue(j,:) = C1*f1' + C2*f2' + C3*f3';
    n = random('Normal', 0, sigmaS, 1, M );
    Sobs(j,:) = Strue(j,:) + n./w;
    jofixiy(ix,iy)=j;
    j=j+1;
end
end

% Step 1: State the problem in words
% A dust sample is assumed to contain four isotopes
% of a toxic chemical element.  The pollutant gets
% into the dust from factories (which are spatially
% localized) and natural sources (which are spatially
% distributed).  A factor is the pattern of isotopes
% associated with a particular source.  A sample is
% a mixture of the factors, where the loading represents
% the mass per unit area per unit time of the pollutant
% deposited from each souce.  In our scenario, the wind
% is physically mixing the dust particles, so that
% each sampe site receives some dust from each source.

% Step 2: Organize the data as a sample matrix S
% The matrix Sobs(i,j) is already in the correct form
% (isotope j at station i).

% Step 3: Establish weights that reflect the importance
% of the elements. The first isotope is known to have been
% measured with much better (say 10x better) accuracy than
% the others.
w = [10, 1, 1, 1];

% Step 4: Perform singular value decomposition and
% form the factor matrix F and loading matrix C
[U,SIGMA,V] = svd(Sobs*diag(w),0);
Fpre = V'*diag(1./w); % factors
Cpre = U*SIGMA;       % loadings

% Step 5: Determine the number P of important factors
% Plot the diagonal of ? as a function of row index i
% and choose P to include all rows with “large” ?_ii
% Since there are only M=4 singular values, we opt to
% print them out:
fprintf('Singular values\n');
```

```
Singular values
```

```
fprintf('%f %f %f %f\n', SIGMA(1,1), SIGMA(2,2), SIGMA(3,3), SIGMA(4,4) );
```

```
6665.622628 134.652708 0.317440 0.190751
```

```
fprintf('\n');
```

```
% We find diag(SIGMA) = [10189 183 3 0.2]
% SIGMA(4,4) is very much smaller than the rest
% so we will ignore it

% Step 6: Reduce the number of factors from M to P
```

```
P = 3;
FpreP = Fpre(1:P,:);
CpreP = Cpre(:,1:P);
SpreP = CpreP*FpreP;

% Step 7: Predict the data and examine the error
% We print out the error
E = Sobs - SpreP;
fprintf('RMS error of Sobs - SpreP\n');
```

```
RMS error of Sobs - SpreP
```

```
for i=[1:4]
    Ei = std(E(:,i));
    mi = std(Sobs(:,i));
    fprintf('  isotope %d is %f (mean is %f)\n', i, Ei, mi );
end
```

```
  isotope 1 is 0.000830 (mean is 0.922206)
  isotope 2 is 0.003824 (mean is 19.130556)
  isotope 3 is 0.002693 (mean is 16.121617)
  isotope 4 is 0.000675 (mean is 60.248322)
```

```
fprintf('\n');
```

```
% The rms errors of the isotopes are
% 0.000805, 0.003498, 0.002850, and 0.000652
% These seem acceptably small, given the overall
% size of the different isotopes in the sample matrix, which are
% 0.921996, 19.129602, 16.122053, and 60.246785

% Step 7: Interpret the factors and their loadings

% print out the factors, normalizing them so the isotopes in
% a factor sum to 1
for i=[1:P]
    norm=sum(FpreP(i,:));
    fprintf('factor %d: %f %f %f %f\n', i, FpreP(i,1)/norm, FpreP(i,2)/norm, FpreP(i,3)/norm,
end
```

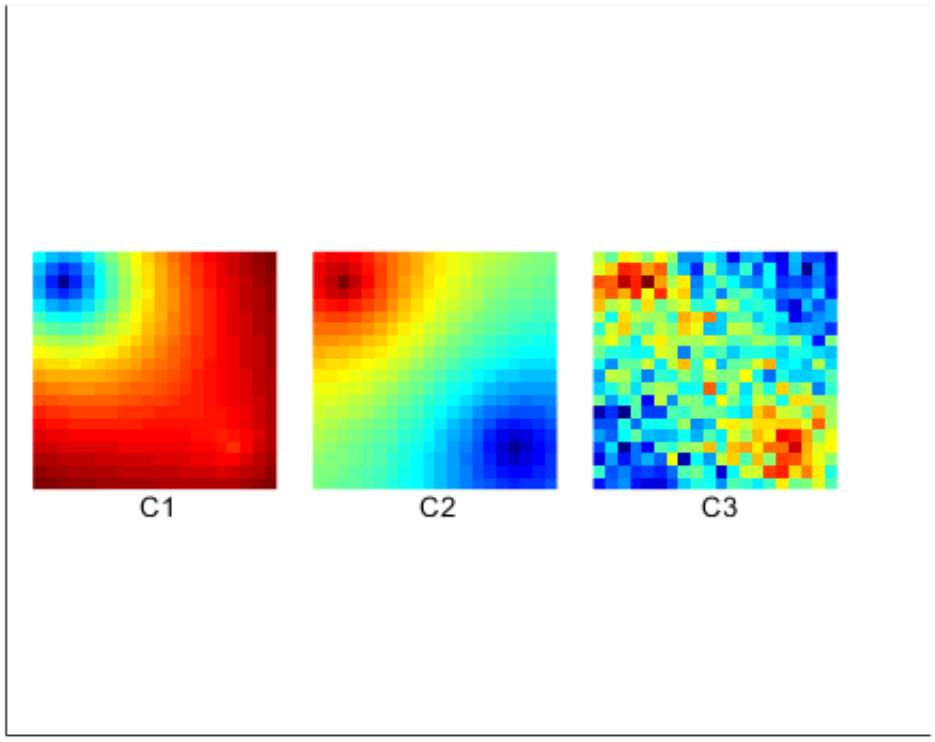
```
factor 1: 0.011629 0.221645 0.188524 0.578203
factor 2: 0.072039 0.877077 0.789346 -0.738461
factor 3: -0.039171 -5.182238 6.166795 0.054615
```

```
fprintf('\n');
```

```
% its not so clear to me that these factors are helpful,
% because the do not have a 1:1 correspondence to sources

% A map of the factor loading show a clear
% spatial pattern with two peaks, which we can assume
% to be the locations of two factories that are
% emitting the pollutant.
map1 = zeros(Nx,Ny);
map2 = zeros(Nx,Ny);
```

```
map3 = zeros(Nx,Ny);
j=1;
for ix=[1:Nx]
for iy=[1:Ny]
    map1(ix,iy)=Cpre(j,1);
    map2(ix,iy)=Cpre(j,2);
    map3(ix,iy)=Cpre(j,3);
    j=j+1;
end
end
gda_draw('     ', map1,'caption C1',' ',map2,'caption C2',' ',map3,'caption C3');
```




```
% Figure 1. A map of the factor loadings (colors) show a clear
% spatial pattern with two peaks, which we can assume to be the
% locations of two factories that are emitting the pollutant.

% measuring the locations of the two sources by eye gives indices
kx1 = 3;  ky1=3;
kx2 = 17; ky2=17;
% measuring a point far from either source gives index
kx3 = 1; ky3=20;

% An informative quantity is the sample closest to
% each of the point sources, and furthest away from both.
% We use SpreP and not Sobs, because we hope the former has
% rejected some of the noise present in the latter
s1 = SpreP( jofixiy(kx1,ky1), :);
s2 = SpreP( jofixiy(kx2,ky2), :);
s3 = SpreP( jofixiy(kx3,ky3), :);
s1 = s1/sum(s1);
s2 = s2/sum(s2);
```

```
s3 = s3/sum(s3);
% these give some sense of the composition of the sources
fprintf('Samples at three chosen points\n');
```

```
Samples at three chosen points
```

```
fprintf('s1 %f %f %f %f\n', s1(1), s1(2), s1(3), s1(4) );
```

```
s1 0.010908 0.213856 0.181321 0.593914
```

```
fprintf('s2 %f %f %f %f\n', s2(1), s2(2), s2(3), s2(4) );
```

```
s2 0.013023 0.236824 0.202329 0.547824
```

```
fprintf('s3 %f %f %f %f\n', s3(1), s3(2), s3(3), s3(4) );
```

```
s3 0.011712 0.222502 0.189422 0.576364
```

```
fprintf('\n');
```

```


% perhaps even a better estimate of sources 1 and 2
% is constructed by subtracting the background 3
% We use SpreP and not Sobs, because we hope the former has
% rejected some of the noise present in the latter
p1 = SpreP( jofixiy(kx1,ky1), :) - SpreP( jofixiy(kx3,ky3), :);
p2 = SpreP( jofixiy(kx2,ky2), :) - SpreP( jofixiy(kx3,ky3), :);
p1 = p1/sum(p1);
p2 = p2/sum(p2);
fprintf('Best estimate of the two point sources\n');
```

```
Best estimate of the two point sources
```

```
fprintf('p1 %f %f %f %f\n', p1(1), p1(2), p1(3), p1(4) );
```

```
p1 0.010290 0.207208 0.175092 0.607410
```

```
fprintf('p2 %f %f %f %f\n', p2(1), p2(2), p2(3), p2(4) );
```

```
p2 0.018167 0.293019 0.252972 0.435842
```

```
fprintf('\n');
```

```


```