Lab in Data analysis

- Matrix basics
- Dim reduction
- Intrinsic dim
- Assignments

Lab: Matrix basic

a) Column centralization

```
[n, p] = size(A);
Am = mean(A)                  % mean of every column
Ac = A - repmat(Am, n, 1)     % Ac has zero mean for each cloumn
```

b) Row centralization of A via transposed matrix A’

c) Column standardization z = (x - mean)/std

```
As = std(A)                   % calculate the std for each column
Az = Ac./repmat(As, n, 1)     % Normalized the columns
                              % Check the result by looking at std(Az)
```

Lab: Matrix basic

d) Transformation of matrices

% If one want to interchange two columns of A, say i:th and j:th columns

% then one can either multiply A by transformation matrix T, where j:th

% column of T is $\mathbf{e}_i$=(0, .., 1, .. 0) and i:th column of T is $\mathbf{e}_j$ or manually

```
B = A*T
```

e) Randomly reorder columns or rows

```
[n, p] = size(A);
C = A(randperm(n), :);    % rearrange the rows randomly
D = A(:, randperm(p));    % rearrange the columns randomly
```

Lab: Matrix basic

f) Some nonlinear transforms, logarithm, exponential, or squares

S = A.^2 % A^2 is the usual multiplication and A need to be square

E = exp(A)

L = Log(P) % P is a positive matrix

g) Generated matrices

covm = cov (A) % Covariance matrix

corm = corr (A) % Correlation matrix

Y = pdist(A, 'method’) % dissimilarity vector,

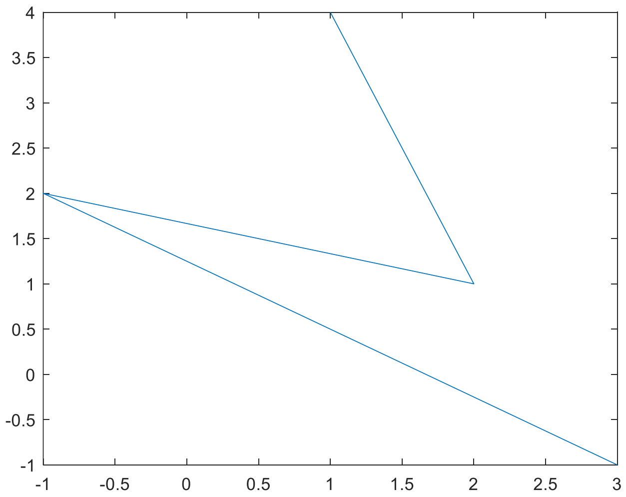
% method: euclidean, minkowski, seuclidean, mahalanobis

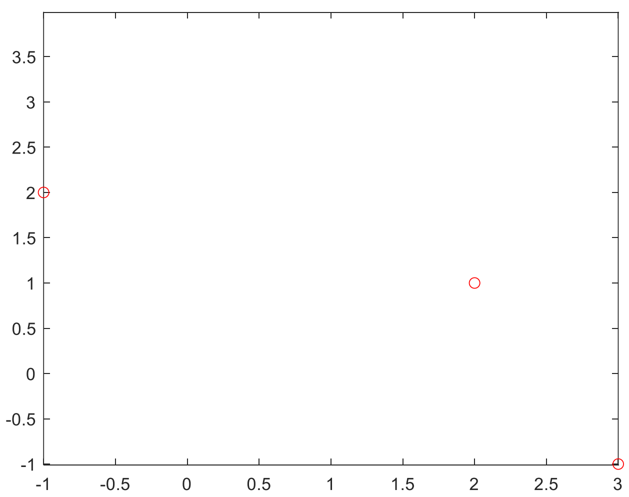
Ds = squareform(Y) % Square dissimilarity matrix

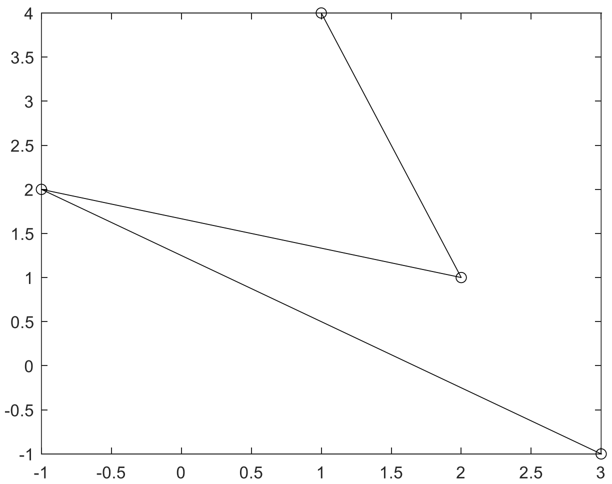
Lab: Plot

a) 2-dim Plot

```
plot ( A(:, i), A(:, j) );                % As a curve
plot ( A(:, i), A(:, j), ‘ro’ );          % Plot for observations
plot ( A(:, i), A(:, j), ‘o-’ );          % Curve & observations
```







Lab: Plot

a) 2-dim Plot

```
plot ( A(:, i), A(:, j) );                                    % As a curve

plot ( A(:, i), A(:, j), ‘*’ );                               % Plot for observations

plot ( A(I1, i), A(I1, j), ‘o’, A(I2, i), A(I2, j), ‘+’, );   % Clusters
```

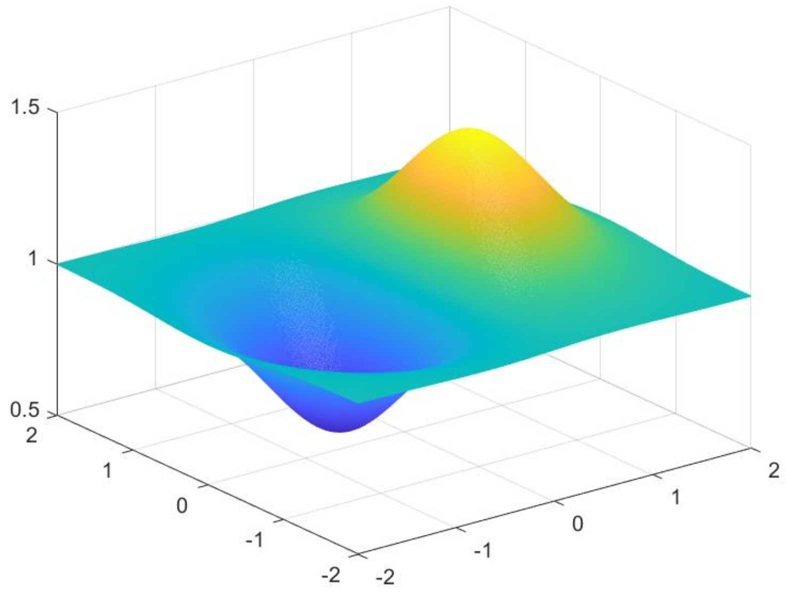
Lab: Plot

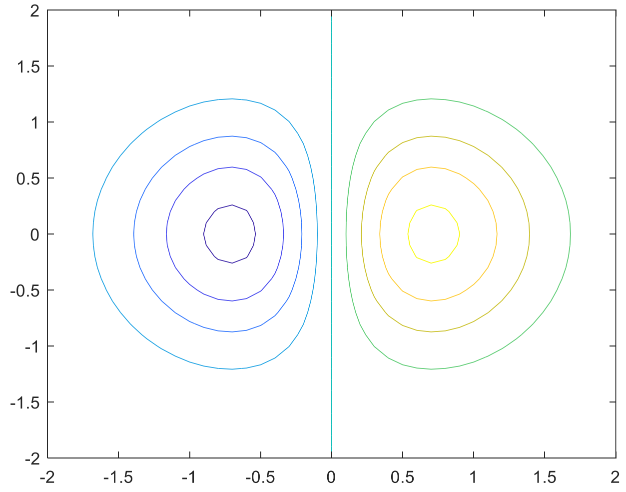
b) 3-dim Plot

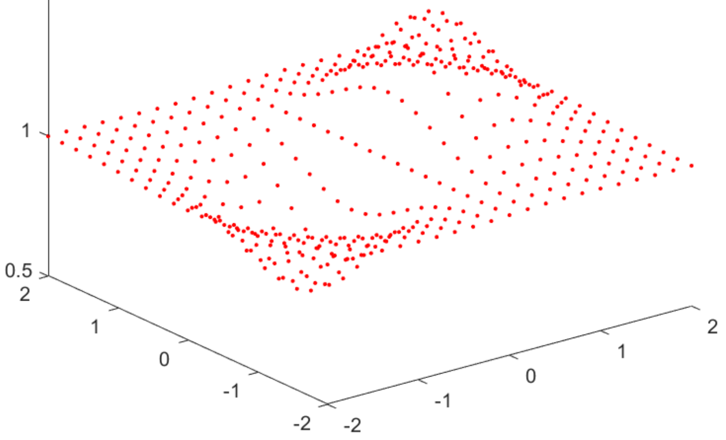
```
plot3 ( A(:, i), A(:, j), A(:,k) );              % As a curve
plot3 ( A(:, i), A(:, j), A(:,k), '*' );         % Plot of observations
plot3 ( A(:, i), A(:, j), A(:,k), '*-');         % Curve & observations

mesh(X, Y, Z)                                    % plot graph of 2-dim function
contour(X, Y, Z)                                 % plot level set of 2-dim function
```

Lab: Plot

b) 3-dim Plot







Lab: Plot

c) Scatterplot

```
scatter ( A(:,i), A(:, j) )                                          % 2-dim plot

scatter( A(I1, i), A(I1, j), 'o', A(I2, i), A(I2, j), '+', );        % Clusters

scatter3( A(:,i), A(:, j), A(:,k) )                                  % 3-dim plot
```

d) plotmatrix

```
plotmatrix(data)
```

Lab: Randomly generating dataset

a) U = rand (n, p) % randomly generated n × p matrix with element in [0, 1]

b) N = randn (n, p) % randomly normal distribution

c) Un = unifrnd (a, b, n, p) % uniformly random numbers from [a, b]

d) Multivariat normal random numbers

M = mvnrnd (mu, sig, n); % mu is p-dim vector, sig is p × p positive matrix,

% n is the number of points

Lab: Dim reduction

a) Eigenvalues, eigenvectors

```
[eigvec, eigval] = eig(A)        % for square matrix
eigval = diag(eigval)            % eigenvalues in a column vector
eigval = flipud(eigval)          % largest to smallest
eigvec = eigvec(:,3:-1:1);       % permutation of eigenvectors
```

b) Projection

```
X                   % dataset
A                   % Covariance or correlation matrix
k                   % lower dim 2, or 3
P = eigvec(:,1:k);  % projection matrix, eigen vectors v1 ... vk
Xk = X*P;           % Projected dataset, dim k
```

Lab: Dim reduction

c) Singular values decomposition

```
[U, D, V] = svd(X)              % X = U*D*V'
k =                             % lower dim 2 or 3
Uk = U(:,1:k )                  % first k columns
Dk = D(1:k, 1:k ) ;             % k × k diag matrix
Vk = V(:,1:k );                 % first k columns
Xk = Uk*Dk*Vk' ;                % k-dim approximation
```

Lab: Dim reduction

d) Factor analysis

```
X                                   % dataset
k =                                 % number of latent variables

[Lam, Psi, T, Stat] = factoran(X, k, 'rotate', 'method’) ;

                                    % method = varimax, promax, ...
                                    % T: Rotation matrix
                                    % Stat: statistics
```

Lab: Intrinsic Dimension

a) Nearest neighbor approach

```
X;                              % dataset
[n, p] = size(X);
Y      = pdist(X) ;             % interpoint distance vector
idhat = idpettis(Y, n);         % intrinsic dim
```

b) Correlation, maximum likelihood, packing numbers

```
X                               % dataset
                                % method = CorrDim, MLE, PackingNumbers
                                % MLE is default
ID = intrinsic_dim(X, 'method’)     % CorrDim, MLE, PackingNumbers
```

Lab: Assignments

1) PCA : Generate n =150, p = 6 normally distributed random variables that have high variance in some dimension and low variance in another dimension. For example, you might use thenfollowing MATLAB code to do this:
x1 = randn(150,2)*100;   x2 = randn(150,4); X = [x1,x2]; Try PCA using both correlation and covariance matrices.
Is the one with the covariance matrix very informative? Which one would bebetter to use in this case?

2) LDA: Consider Fisher's iris data, Repeat the example in the book and show the data in 1–D,   along with the kernel density estimate.
Discuss how good the mappings are for each case.
a) Taking two classes at a time, i.e., [setosa, versicolor]; [versicolor, virginica]; [virginica, setosa].
b) looking at the data in another way and take the two classes as [Sepals, Petals].

3) Factor analysis:
a) Repeat example 2.5 (dataset: *stockreturns*) with rotation varimax and compare results
b) Collect your own data by looking at the last 20 days at the Stockholm's Nasdaq index for these companies: ABB, SAAB, Volvo, Atlas Copco; NCC, JM, Skanska; SEB, Nordea bank, Handelsbanken. Then carry out a factor analysis for your data.

4) Dimensional analysis: The nearest neighbor approach: Similarly as in example 2.8 study the following curve: $x_1 = \frac{t \cos t}{1+t^2},\ x_2 = \frac{t \sin t}{1+t^2},\ x_3 = t,\ -2\pi \le t \le 2\pi$
a) Use idpettis.m to estimate the intrinsic dim
b) Add noise of various sizes to your cuvre and thereafter study the intrinsic dim.
c) Is there any threshold number of noise size for intrinsic dim estimate ?

Lab: Assignments

5) Singular value decomposition: Leukemia dataset

a) Apply the SVD to dataset Leukemia, choose a proper lower dim k via elbow in the plot of singular vaules, then plot the dimension reduced data in both 2-dim and 3-dim (in case your k is at least three).

b) Even try the analysis by PCA (covaraince) and compare the resuts from these two different methods.

6) Dimension analysis:

a) Repeat the following example 2.10 on next page by using CorrDim and compare the results.

b) Modify the code in genLDdata.m so that:

The sphere will be replaced by an ellipsoid, e.g. $\frac{x_1^2}{4^2} + \frac{x_2^2}{5^2} + \frac{x_3^2}{6^2} = 1$

The vertical segment will be replaced by the curve,

$$x_1 = \frac{t\cos(t^2)}{1+t^2},\ x_2 = \frac{t\sin(t^2)}{1+t^2},\ x_3 = t\ ,\ \ -2\pi \le t \le 2\pi$$

The horizontal segment will be replaced by a segment connecting the ellipsoid and the cub.

Then study the intrinsic dim by PackningNumbers and also the local dim.

Lab: Example 2.10

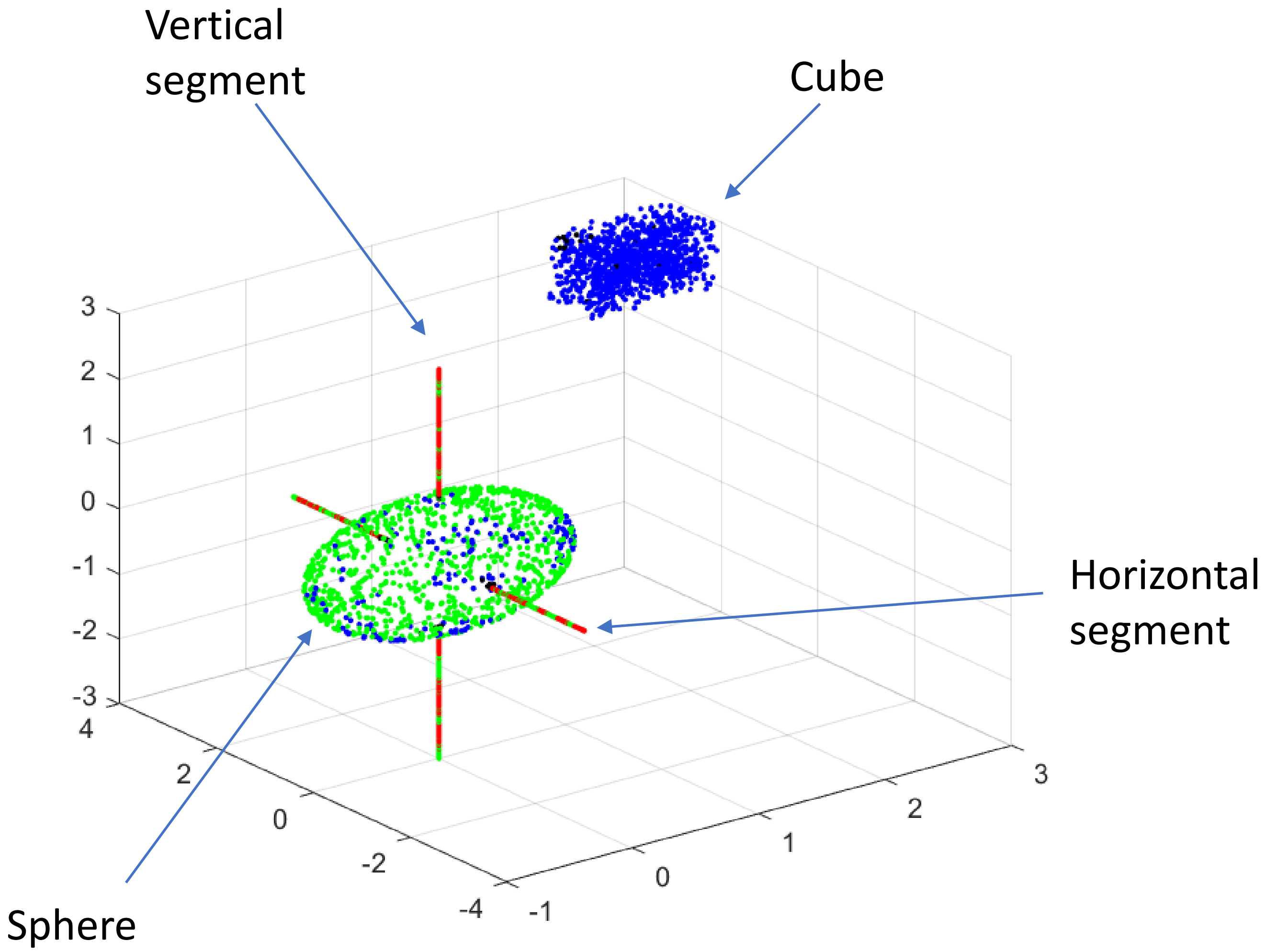
```
A=genLDdata;
 % estimate the global intrisic dimensionality
% default is MLE
ID = intrinsic_dim(A) % 1.5229
% get the pairwise interpoint distance
%  use the default Euclidean distance
 Ad = squareform(pdist(A));
 % get the dimension of data
[nr, nc] = size(A);
Ldim = zeros(nr, 1);
Ldim2 = Ldim;
[Ads, J] =  sort(Ad, 2);
% set the neighborhood size
K = 100;
for m =1:nr;
Ldim(m, 1) = ...
    intrinsic_dim(A(J(m,1:k), :));
end
```

Lab: Example 2.10

```
% local dimension
  Ldim(Ldim>3) = 4;
  Ldim = ceil(Ldim);
 % Tabulate them and find count and procentage for dim 1, 2, 3, 4
 tabulate(Ldim)

% Scatterplot with color map
 ind1= find(Ldim == 1);
 ind2= find(Ldim == 2);
ind3= find(Ldim == 3);
ind4= find(Ldim == 4);
scatter3(A(ind1, 1), A(ind1, 2),A(ind1, 3), 'r.’)
 hold on
scatter3(A(ind2, 1), A(ind2, 2),A(ind2, 3), 'g.')
scatter3(A(ind3, 1), A(ind3, 2),A(ind3, 3), 'b.')
scatter3(A(ind4, 1), A(ind4, 2),A(ind4, 3), 'k.’)
hold off
```

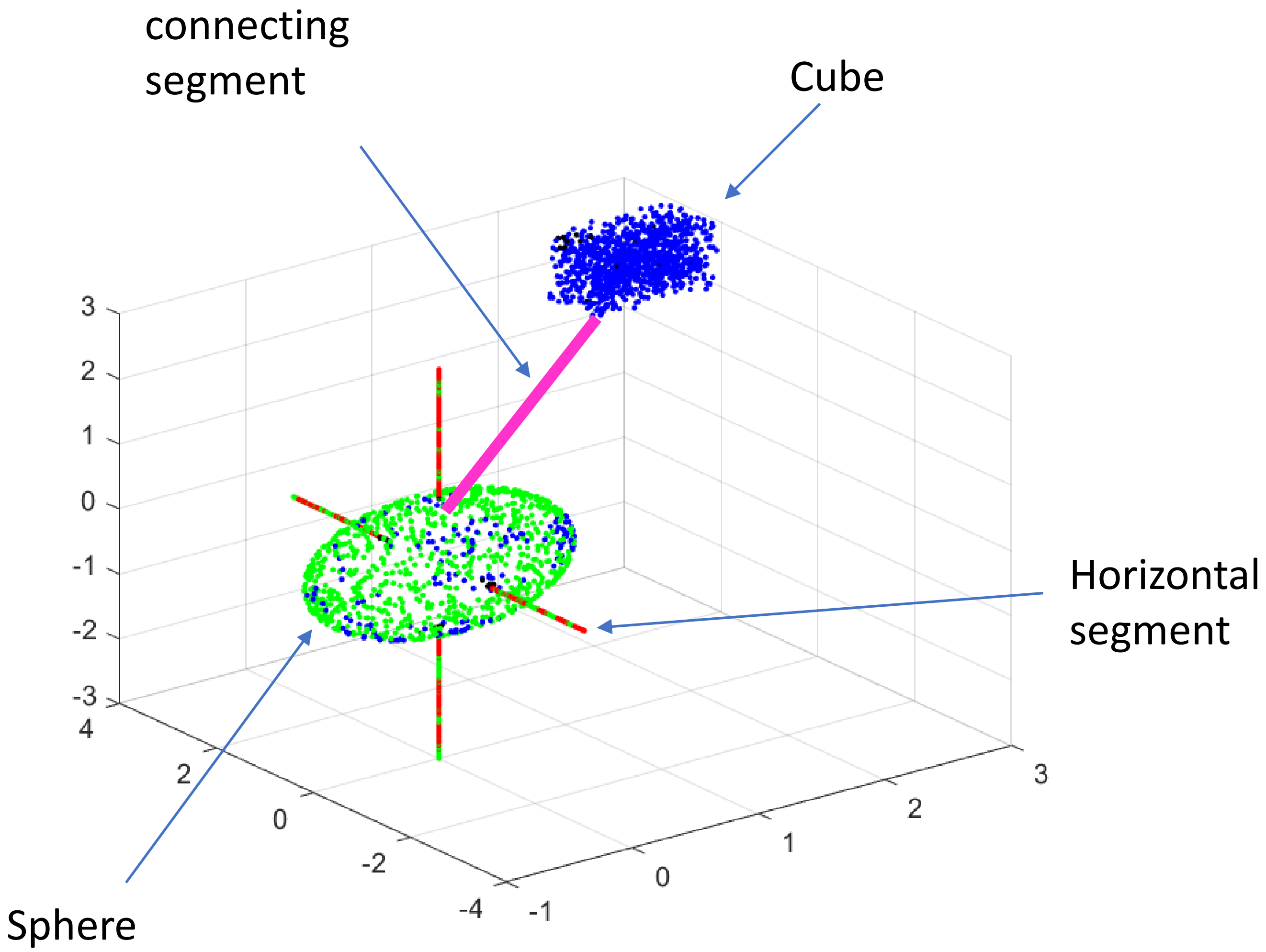
Lab: Example 2.10

```
% local dimension
  Ldim(Ldim>3)=4;
  Ldim=ceil(Ldim);
 % Tabulate them
 tabulate(Ldim)

% Scatterplot with color map
 ind1= find(Ldim == 1);
ind2= find(Ldim == 2);
ind3= find(Ldim == 3);
ind4= find(Ldim == 4);
scatter3(A(ind1, 1), A(ind1, 2),A(ind1, 3), 'r.')
 hold on
scatter3(A(ind2, 1), A(ind2, 2),A(ind2, 3), 'g.')
scatter3(A(ind3, 1), A(ind3, 2),A(ind3, 3), 'b.')
scatter3(A(ind4, 1), A(ind4, 2),A(ind4, 3), 'k.')
hold off
```



Lab: Example 2.10

```
% local dimension
 Ldim(Ldim>3)=4;
 Ldim=ceil(Ldim);
% Tabulate them
tabulate(Ldim)

% Scatterplot with color map
ind1= find(Ldim == 1);
ind2= find(Ldim == 2);
ind3= find(Ldim == 3);
ind4= find(Ldim == 4);
scatter3(A(ind1, 1), A(ind1, 2),A(ind1, 3), 'r.')
hold on
scatter3(A(ind2, 1), A(ind2, 2),A(ind2, 3), 'g.')
scatter3(A(ind3, 1), A(ind3, 2),A(ind3, 3), 'b.')
scatter3(A(ind4, 1), A(ind4, 2),A(ind4, 3), 'k.')
hold off
```



genLDdata.m

```
function   A = genLDdata
% This function will return 1,000 points
% in 3 dimensions.
%   X = genLDdata
%   It consists of data along two lines,
%   data on the surface of a sphere, and
%   data in a cube. This data set can be
%   used to explore the estimation of
%   local dimension.
```

```
% sample from the surface of a sphere
X1 = randn(1000,1);
X2 = randn(1000,1);
X3 = randn(1000,1);
lambda = sqrt(X1.^2 + X2.^2 + X3.^2);
X1 = X1./lambda;
X2 = X2./lambda;
X3 = X3./lambda;
X = [X1, X2, X3];
plot3(X(:,1), X(:,2), X(:,3) ,'r.')
```

genLDdata.m

```
% sample from a cube
X1 = rand(1000,1) + 2;
X2 = rand(1000,1) + 2;
X3 = rand(1000,1) + 2;

XX = [X1, X2, X3];
hold on
plot3(XX(:,1), XX(:,2), XX(:,3), 'g.')
```

```
% sample from lines attached to a sphere
X1 = zeros(1000,1);
X2 = zeros(1000,1);
X3 = 2*rand(1000,1) + 1;
L1 = [X1, X2, X3];

X1 = zeros(1000,1);
X2 = zeros(1000,1);
X3 = -2*rand(1000,1) + -1;
L2 = [X1, X2, X3];

X1 = zeros(1000,1);
X2 = 2*rand(1000,1)+1;
X3 = zeros(1000,1);
L3 = [X1, X2, X3];
```

genLDdata.m

```
X1 = zeros(1000,1);
X2 = -2*rand(1000,1)-1;
X3 = zeros(1000,1);
L4 = [X1, X2, X3];

A = zeros(6000,3);
A(1:1000,:) = X;
A(1001:2000,:) = XX;
A(2001:3000,:) = L1;
A(3001:4000,:) = L2;
A(4001:5000,:) = L3;
A(5001:6000,:) = L4;
```

```
plot3(A(:,1), A(:,2), A(:,3), 'b.')
grid on
box on

hold off
```

EDA toolbox

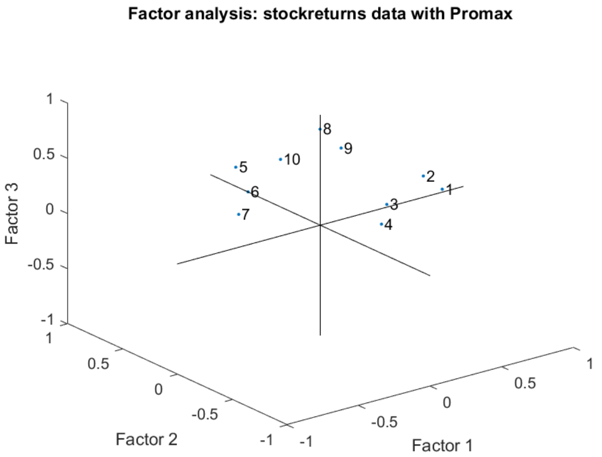
https://www.routledge.com/Exploratory-Data-Analysis-with-MATLAB-Third-Edition/Martinez-Martinez-Solka/p/book/9781498776066

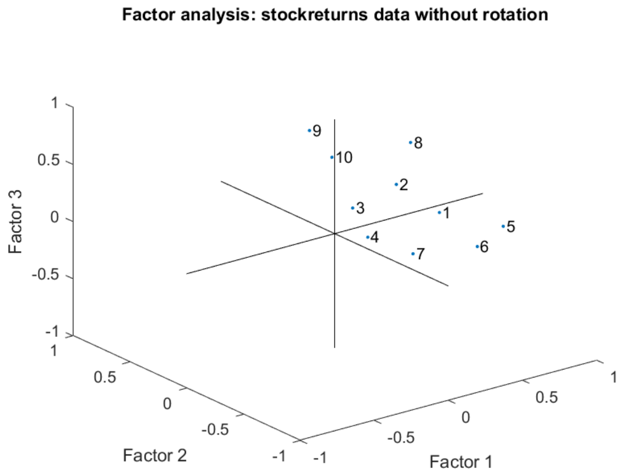
Lab report template

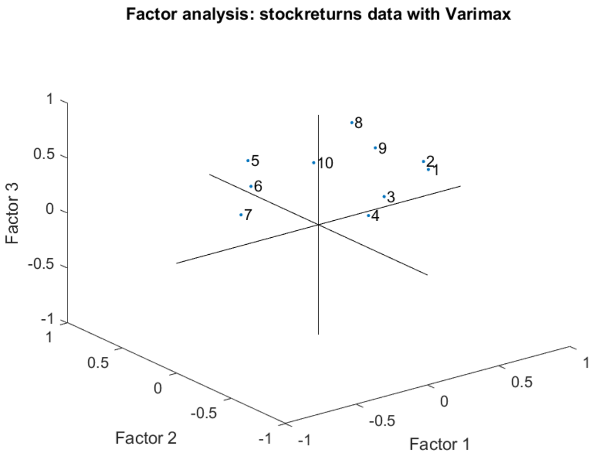
MA661E Lab report

Anders Andersson, Sven Svensson

Exercise 1. Factor analysis







Varimax rotation is an orthogonal rotation of the factor axes to maximize the variance of the squared loadings of a factor (column) on all the variables (rows) in a factor matrix, which has the effect of differentiating the original variables by extracted factor. Each factor will tend to have either large or small loadings of any particular variable. A varimax solution yields results which make it as easy as possible to identify each variable with a single factor. This is the most common rotation option.

Promax rotation is an alternative non-orthogonal rotation method which is computationally faster than the direct oblimin method and therefore is sometimes used for very large datasets.

Comparison: For this simple dataset we see that both methods work very well, and have separated the clusters much better than the method without rotation

Exercise 2. ……….

Exercise 3. ……….

Lab report template

Appendix: Codes

Exercise 1. Factor analysis

Repeat the example 2.5 (modification in plot: in 3-dim instead of 2-dim)

Code

```
load stockreturns
lab={'1', '2', '3', '4', '5','6', '7', '8', '9', '10'};
t=-1:0.1:1;
[Lam, Psi] = factoran(stocks,3,'rotate','none');
plot3(Lam(:, 1), Lam(:, 2), Lam(:, 3), '.')
text(Lam(:, 1)+0.02, Lam(:, 2), Lam(:, 3), lab)
hold on
plot3(t, 0*t,0*t, 'k', 0*t, t, 0*t, 'k', 0*t, 0*t, t, 'k')
title('Factor analysis: stockreturns data without Rotation')
Figure
[LProt, PProt]=factoran(stocks,3,'rotate','promax');
plot3(LProt(:, 1), LProt(:, 2), LProt(:, 3), '.')
text(LProt(:, 1)+0.02, LProt(:, 2), LProt(:, 3), lab)
hold on
plot3(t, 0*t,0*t, 'k', 0*t, t, 0*t, 'k', 0*t, 0*t, t, 'k')
title('Factor analysis: stockreturns data with Promax')
```

Exercises 2. PCA

xxx

xxx

xxx

Lab report hand-in

Hand-in of lab report latest at 11:59 pm, Monday, 2024-02-26 via Canvas