Gene clustering
Gene sorting

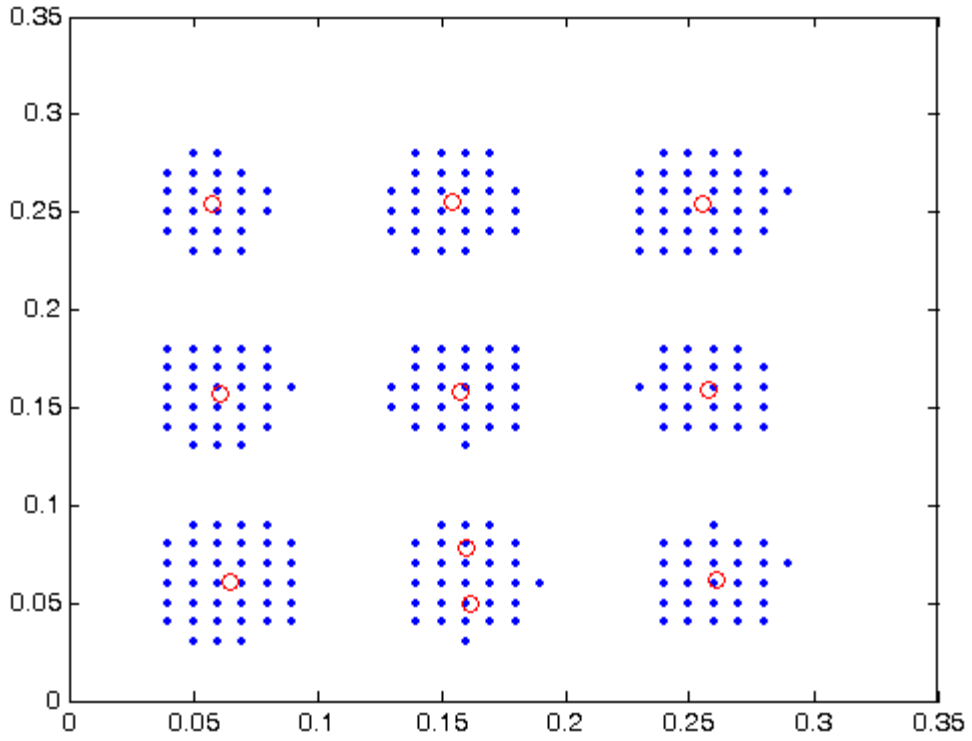
Outline

- Clustering
  - K-means
  - Annealed K-means
- Kohonen’s Self-organizing algorithm
- Yeast gene analysis
- Cancer gene analysis

- High dimensional data clustering
  - K-means
  - Annealed K-Means
  - Euclidean distance measure
  - Mahalanobis distance measure

Data clustering

- Find cluster centers



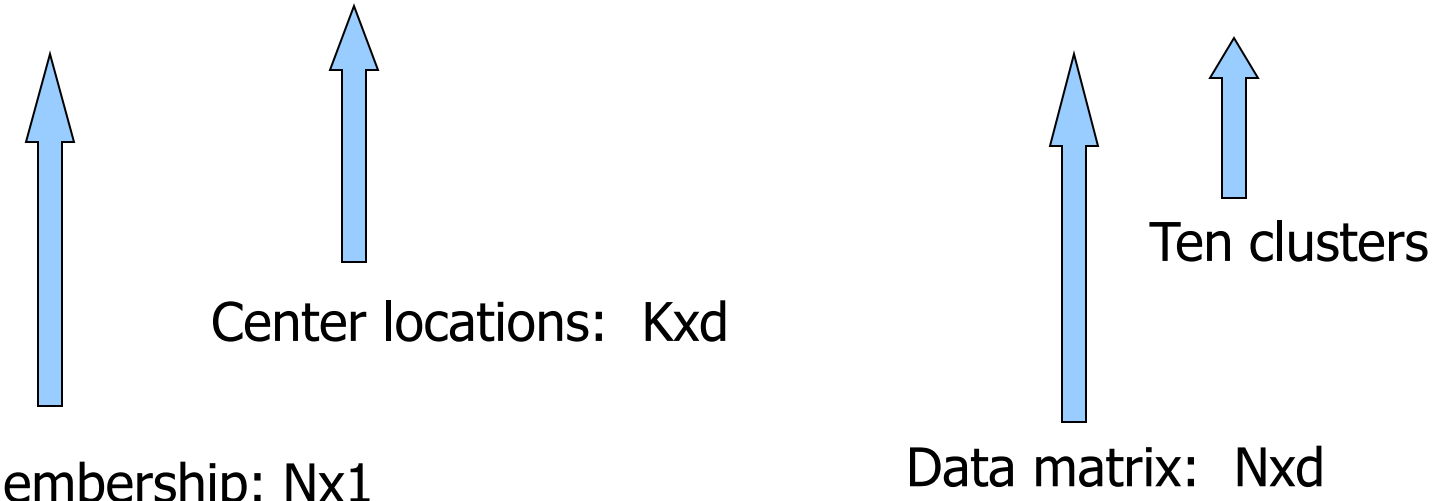
Matlab stats toolbox

- Add directory, toolbox\stats, to matlab path
- Load data_9.mat
- Kmeans clustering

```
[cidx, ctrs] = kmeans(XX,10);
  plot(XX(:,1),XX(:,2),'.');
  hold on;
  plot(ctrs(:,1),ctrs(:,2),'ro');
```

Matlab stats toolbox

```
[cidx, ctrs] = kmeans(XX,10);
```



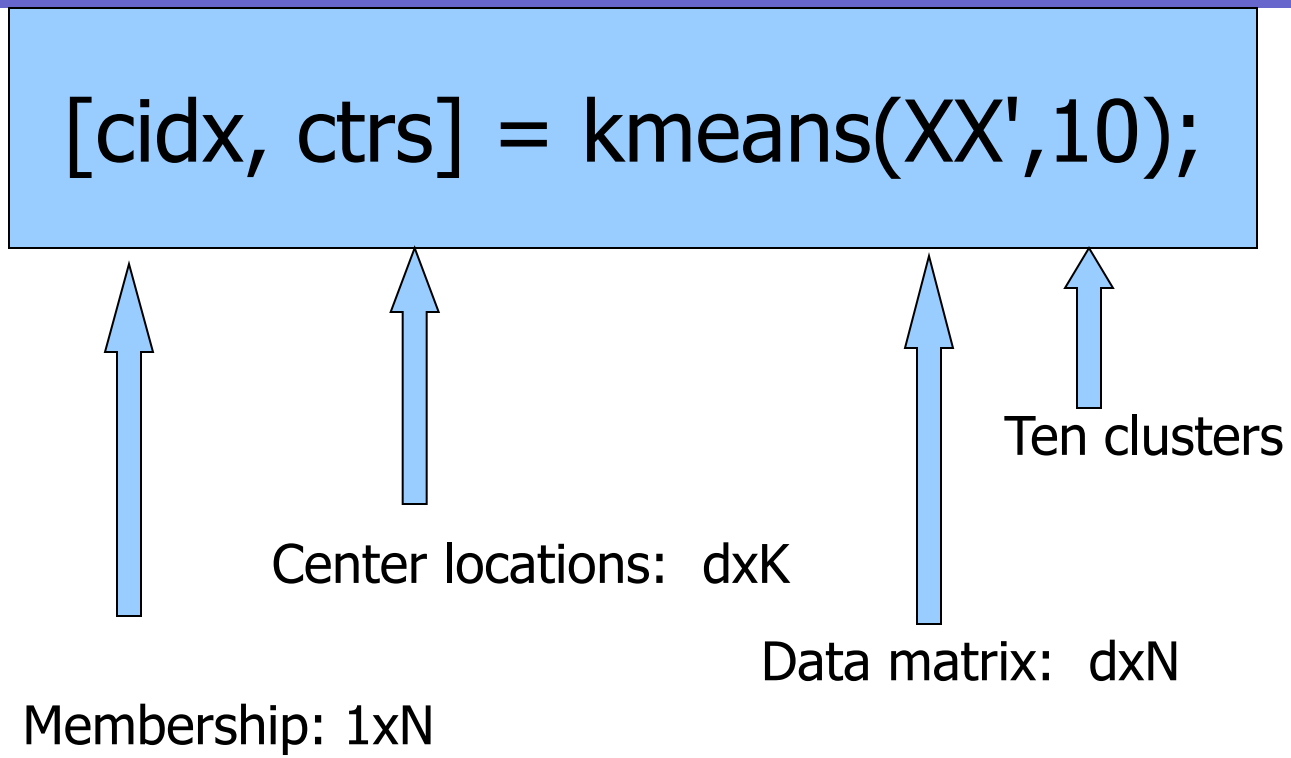
Download

- Internet

  My_Kmeans Parallel codes

- Example
  - Add directory, cluster, to matlab path
  - Load data_9.mat
  - Kmeans clustering

```
[cidx, ctrs] = kmeans(XX',10);
plot(XX(:,1),XX(:,2),'.');
hold on;
plot(ctrs(1,:),ctrs(2,:),'ro');
```


[cidx, ctrs] = kmeans(XX',10);
Ten clusters
Center locations: dxK
Data matrix: dxN
Membership: 1xN

K-means

- An iterative approach
  - Non-overlapping partition
  - Means of partitioned groups
- Non-overlapping partition : Each data x is partitioned to its nearest center

Stepwise procedure

1. Input unsupervised high dimensional data and the number K of clusters
2. Randomly initialize K centers near the center of given data
3. Partition all data to K clusters using current means
4. Update the center position of each group
5. If halting condition holds, exit
6. Go to step 3

Exercise

- Draw a while-loop flow chart to illustrate how the K-means algorithm partitions high-dimensional data to K clusters

Exercise

- Apply the K-means method to clustering analysis of data in data_25.mat
- Calculate the mean distance between each data point and its correspondent center

Annealed K-means

- Use overlapping memberships
- Overcome the local minimum problem
- Relaxation under an annealing process

Overlapping memberships

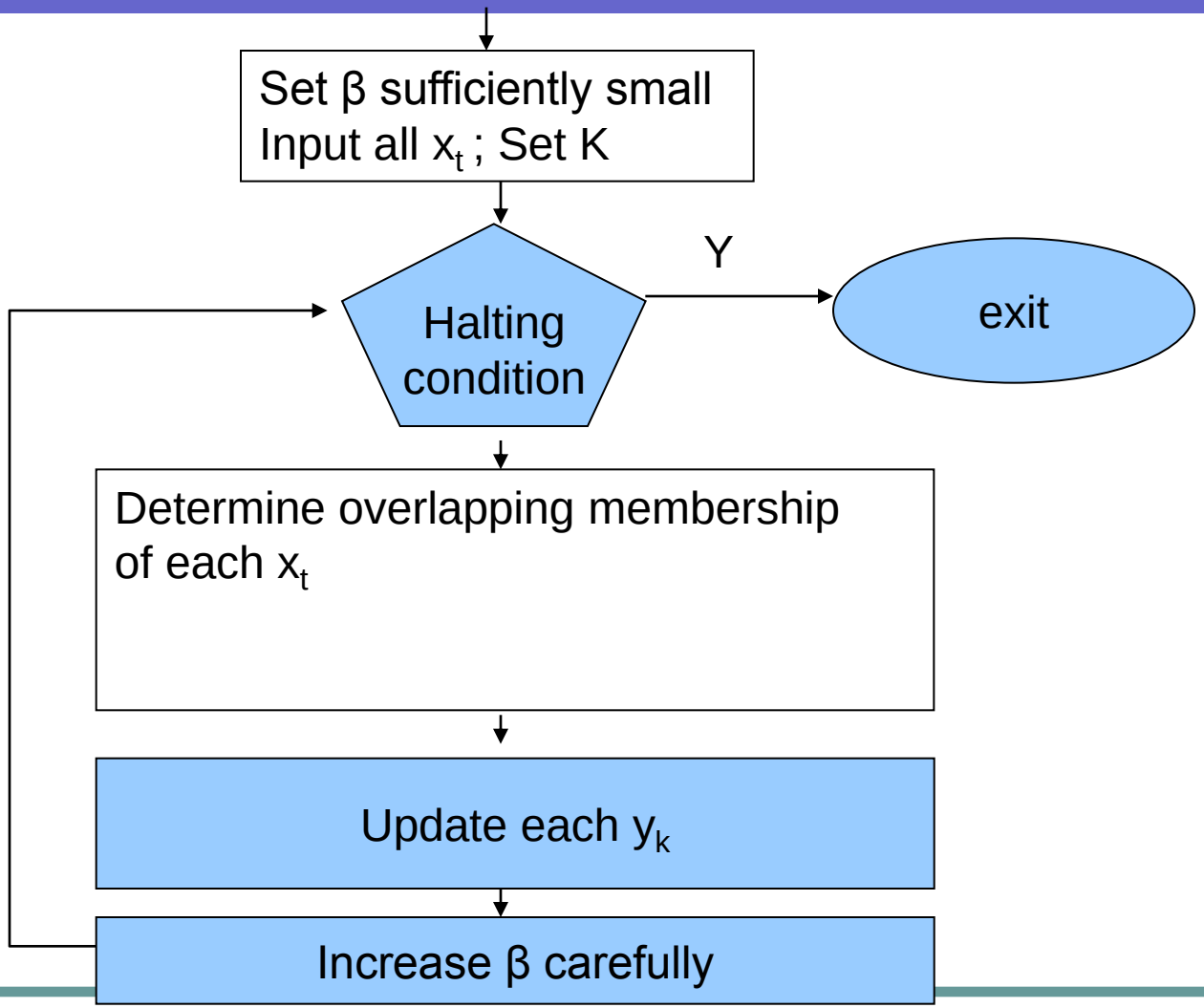
- Each x has an overlapping membership to each cluster
- The membership to the ith cluster is proportional to $\exp(-\beta d_i)$, where d_i denote the distance between x and the ith center

Annealed K-means

1. Input unsupervised high dimensional data and the number K of clusters
2. Set β to sufficiently small positive number; randomly initialize K centers near the center of given data
3. Determine overlapping memberships of all data to K clusters
4. Update the center position of each group
5. Increase β carefully
6. If halting condition holds, exit
7. Go to step 3

Exercise

- Draw a while-loop flow chart to illustrate how the annealed K-means algorithm partition high-dimensional data to K clusters
- Implement the flow chart using Matlab codes


Set β sufficiently small
Input all x_t ; Set K
Halting condition
Y
exit
Determine overlapping membership of each x_t
Update each y_k
Increase β carefully

- Minimize the weight distance

$$\mathrm{E}_i = \sum_t v_i[t] \left\| \mathbf{x}[t] - \mathbf{y}_i \right\|^2$$

$$\frac{dE_i}{dy_i} = 0$$

$$\sum_t v_i[t](\mathbf{x}[t] - \mathbf{y}_i) = 0$$

$$\mathbf{y}_\mathrm{i} = \frac{\sum_t v_i[t]\mathbf{x}[t]}{\sum_t v_i[t]}$$

$$v_k \propto \exp(-\beta d_k)$$

$$v_k = C \exp(-\beta d_k)$$

$$\sum_k v_k = 1$$

$$\sum_k C \exp(-\beta d_k) = 1$$

$$C = \frac{1}{\sum_k \exp(-\beta d_k)}$$

$$v_k = \phi(\mathbf{d}) = \frac{\exp(-\beta d_k)}{\sum_j \exp(-\beta d_j)}$$

Overlapping membership

- Sufficiently small β
  - v_k equals 1/K
- Sufficiently large β
  - $v_k = 1$ if $d_k = \min_i d_i$
    $= 0$ otherwise
  - All v_k are determined by the winner-take-all principle

- Why annealing?
- Sufficiently large β leads to the K-means algorithm
- Can annealing improve the performance?
- How to measure the performance for clustering?

$$\mathrm{E}_i = \sum_t v_i[t] \left\| \mathbf{x}[t] - \mathbf{y}_i \right\|^2$$

$$\mathrm{E} = \sum_i \mathrm{E}_i$$

$$= \sum_i \sum_t v_i[t] \left\| \mathbf{x}[t] - \mathbf{y}_i \right\|^2$$

$$= \sum_t \sum_i v_i[t] \left\| \mathbf{x}[t] - \mathbf{y}_i \right\|^2$$

$$\text{if } \delta_i[t] = v_i[t] \in \{0,1\}$$

E sums up the distance of each $\mathbf{x}[t]$ *to* the nearest center

Criterion of K-means

- Minimal distance to the nearest center

$$\mathrm{E}(\mathrm{y}) = \sum_{t} \sum_{i} \delta_i[t] \left\| \mathbf{x}[t] - \mathbf{y}_i \right\|^2$$

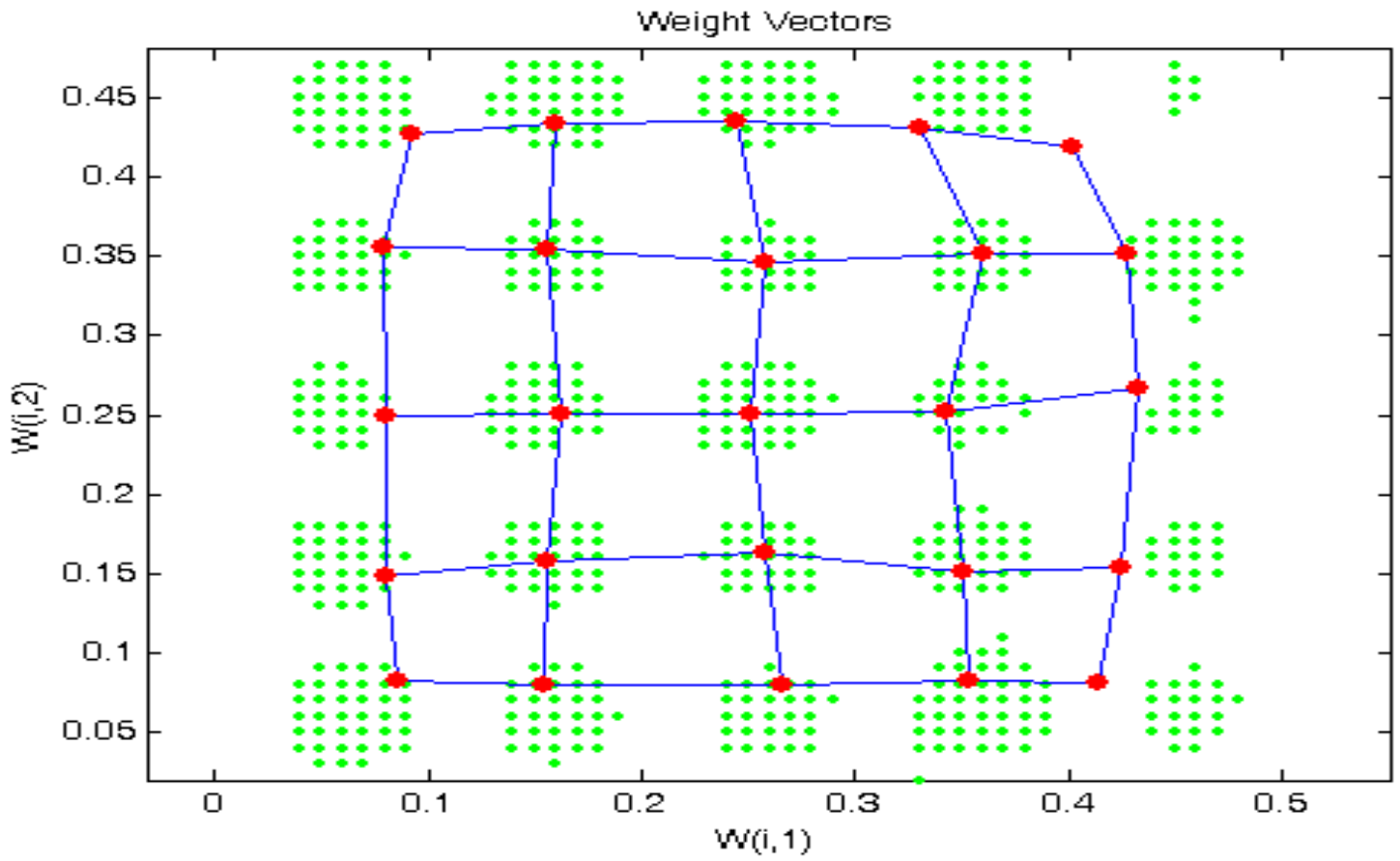
Cross distance

- Refer to Math Programming

Self-organization

- Clustering
- Define neighboring relations of clusters on a 2D lattice
- Sorting high-dimensional data on a lattice

Clustering by SOM



SOM

- Download Matlab Neural Networks toolbox
- Add directory, nnet, with sub-directories to matlab path
- Apply the SOM method to clustering analysis of data in data_25.mat

Load data

```
load data_25
plot(XX(:,1),XX(:,2),'.g')
hold on
```

SOM

Initialization

```
net = newsom([0 0.2; 0 0.1],[5 5],'gridtop');
net.trainParam.epochs=100;
```

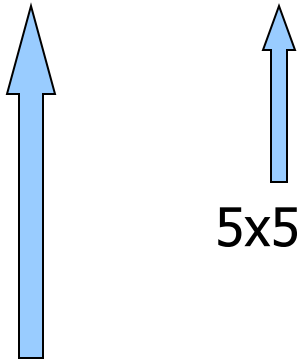
Training

```
net = train(net,XX');
```

Display

```
plotsom(net.iw{1,1},net.layers{1}.distances)
```

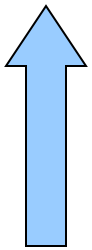
```
net = newsom([0 0.2; 0 0.1],[5 5],'gridtop');
```



5x5 lattice

2x2 matrix for two-dimensional inputs
dx2 matrix for d-dimensional inputs

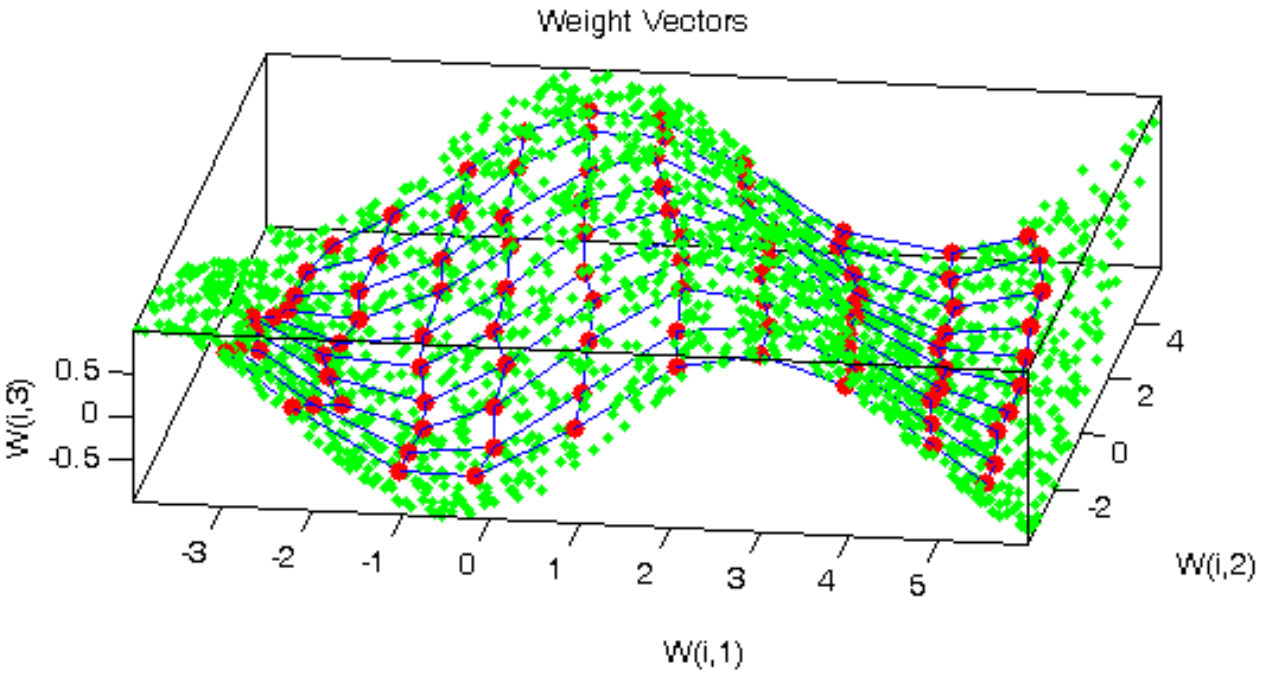
```
net = train(net,XX');
```



dxN data matrix

Surface construction

demo_fr2_som.m



Surface reconstruction

```
P = rand(2,2000)*4-2;
f = exp(-sum(P.^2));
P =[P;f];
net = newsom([0 0.2; 0 0.1; 0 0.1],[10 10],'gridtop');
plotsom(net.layers{1}.positions)
net.trainParam.epochs=100;
net = train(net,P);
plot3(P(1,:),P(2,:),P(3,:),'.g')
hold on
plotsom(net.iw{1,1},net.layers{1}.distances)
```

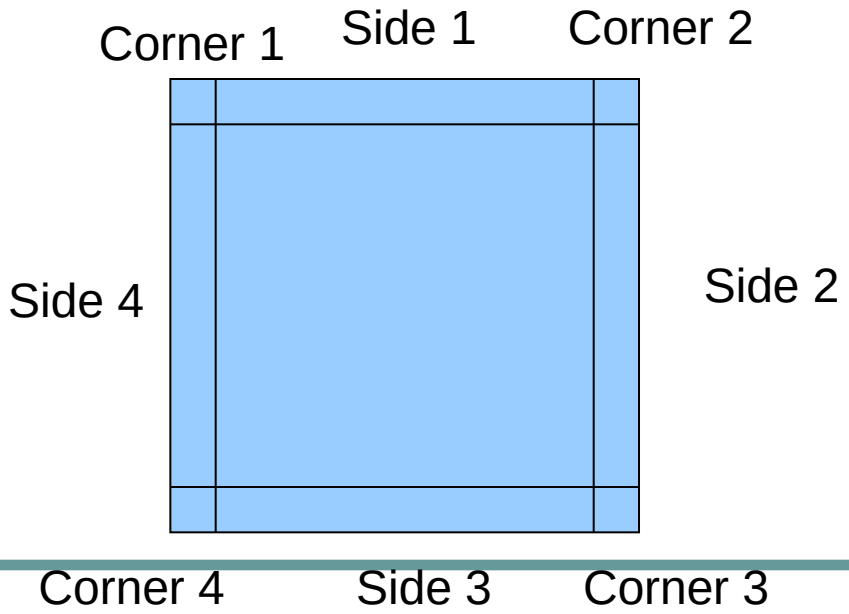
Self-organization

- Maximal fitting and minimal wiring principle
- Sort high dimensional data on a 2D lattice

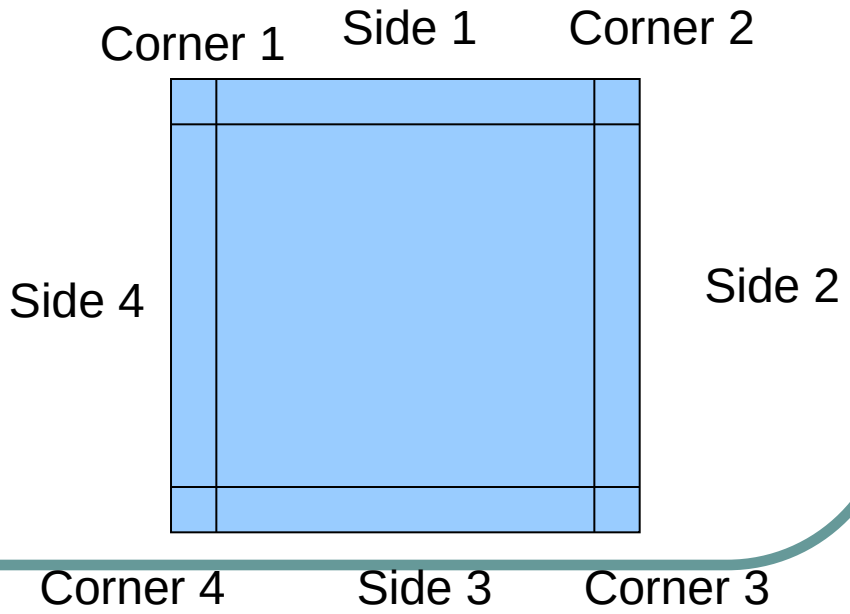
Neural optimization

- K-means
  - Minimal distance to the nearest center
- Kohonen’s self-organizing
  - Minimal wiring distance between neighboring centers

- Neighboring relations on a lattice
- An M-by-M lattice
  - A node is indexed by (i,j), i,j=1,…,M
  - C1: i=1,j=1
  - C2: i=1,j=M
  - C3: i=M,j=1
  - C4: i=M,j=M



- Neighboring relations on a lattice
- An M-by-M lattice
  - A node is indexed by (i,j), i,j=1,…,M
  - S1: i=1,j=2:M-1
  - S2: i=2:M-1,j=M
  - S3: i=M,j=2:M-1
  - S4: i=2:M-1,j=1



- If (i,j) not within S1-S4 and C1-C4

  NB(i,j)= {(i-1,j),(i+1,j),(i,j-1),(i, j+1)}
- NB(i,j) can be separately defined if (i,j) belongs S1- S4 and C1-C4

- Wiring Criterion

$$W = \sum_{k} \sum_{(i,j) \in NB(k)} \left\| y_{h(i,j)} - y_k \right\|^2$$

$$h(i, j) = (i - 1) * M + j$$

Kohonen's self-organizing algorithm

- Self-organizing map - Wikipedia, the free encyclopedia
- Neural networks (1996)
- Neural networks (2002)
- Neurocomputing (2011)

SOM matlab toolbox

- SOM Toolbox

- Neural Networks 15(3), 337-347 (2002)

- Neurocomputing 74(12-13), 2228-2240 (2011)

Minimal wiring and maximal fitting criteria

- Maximal fitting to a center
- Minimal distance to the nearest center

$$\mathrm{E(y)} = \sum_t \sum_i \delta_i[t] \left\| \mathbf{x}[t] - \mathbf{y}_i \right\|^2$$

- Wiring Criterion

$$\mathrm{W} = \sum_k \sum_{(i,j) \in NB(k)} \left\| y_{h(i,j)} - y_k \right\|^2$$

$$h(i,j) = (i-1) * M + j$$

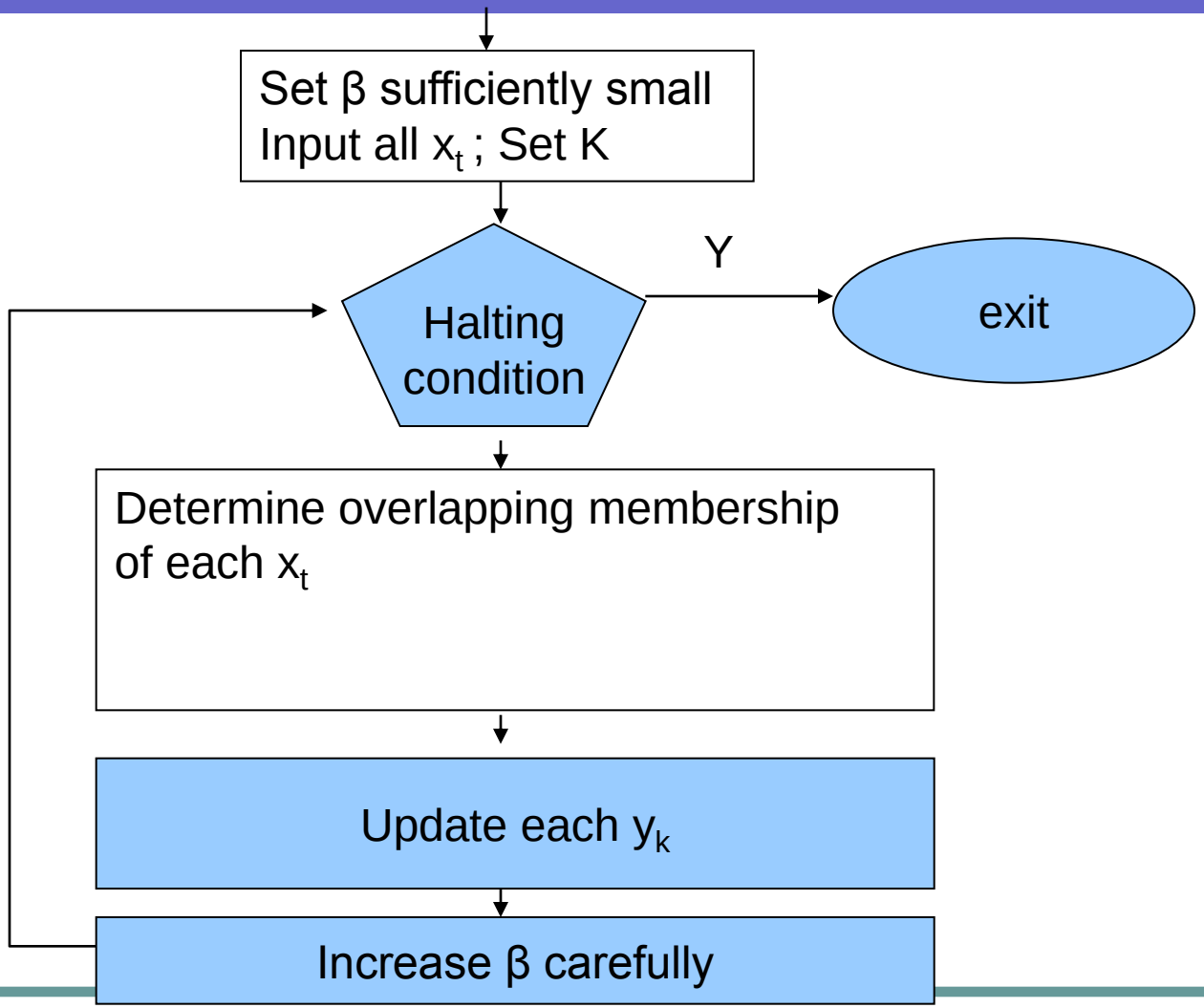
Minimal wiring and maximal fitting criteria

- $$F(y) = E + \lambda W$$

$$E(y) = \sum_t \sum_i \delta_i[t] \left\| \mathbf{x}[t] - \mathbf{y}_i \right\|^2$$

$$W = \sum_k \sum_{(i,j) \in NB(k)} \left\| y_{h(i,j)} - y_k \right\|^2$$

$$h(i, j) = (i - 1) * M + j$$


Set β sufficiently small
Input all x_t ; Set K
Halting condition
Y
exit
Determine overlapping membership of each x_t
Update each y_k
Increase β carefully

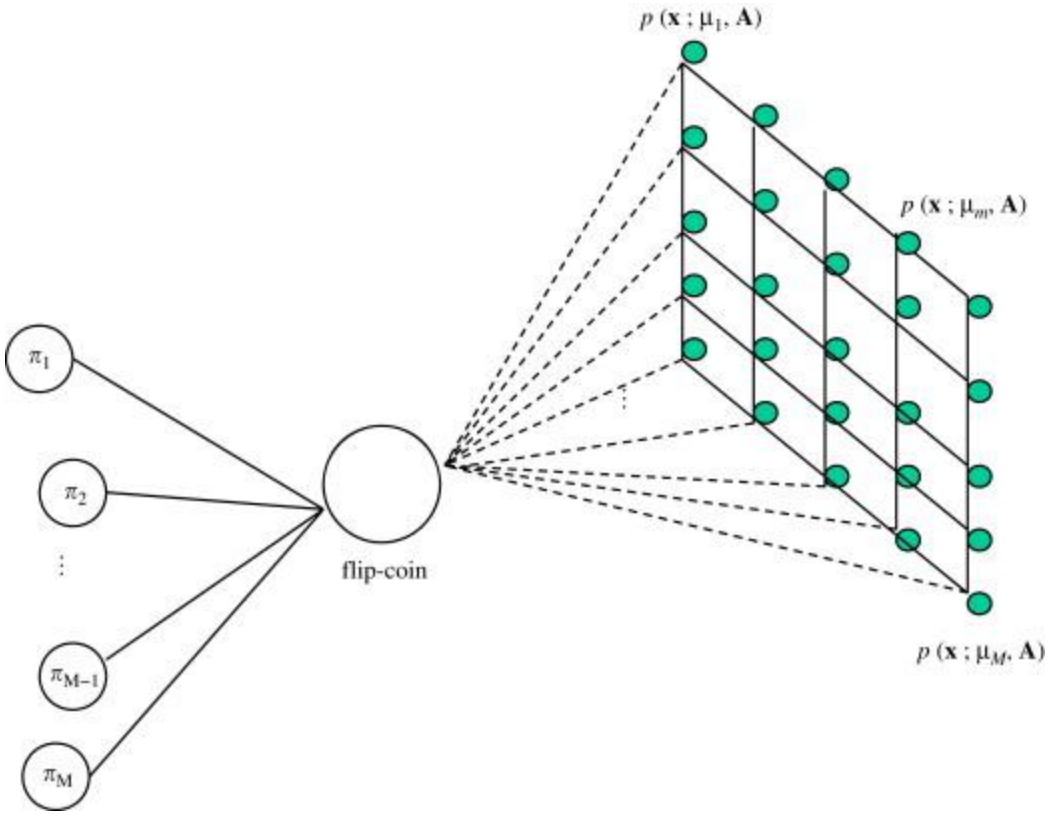
- Minimize the weight distance

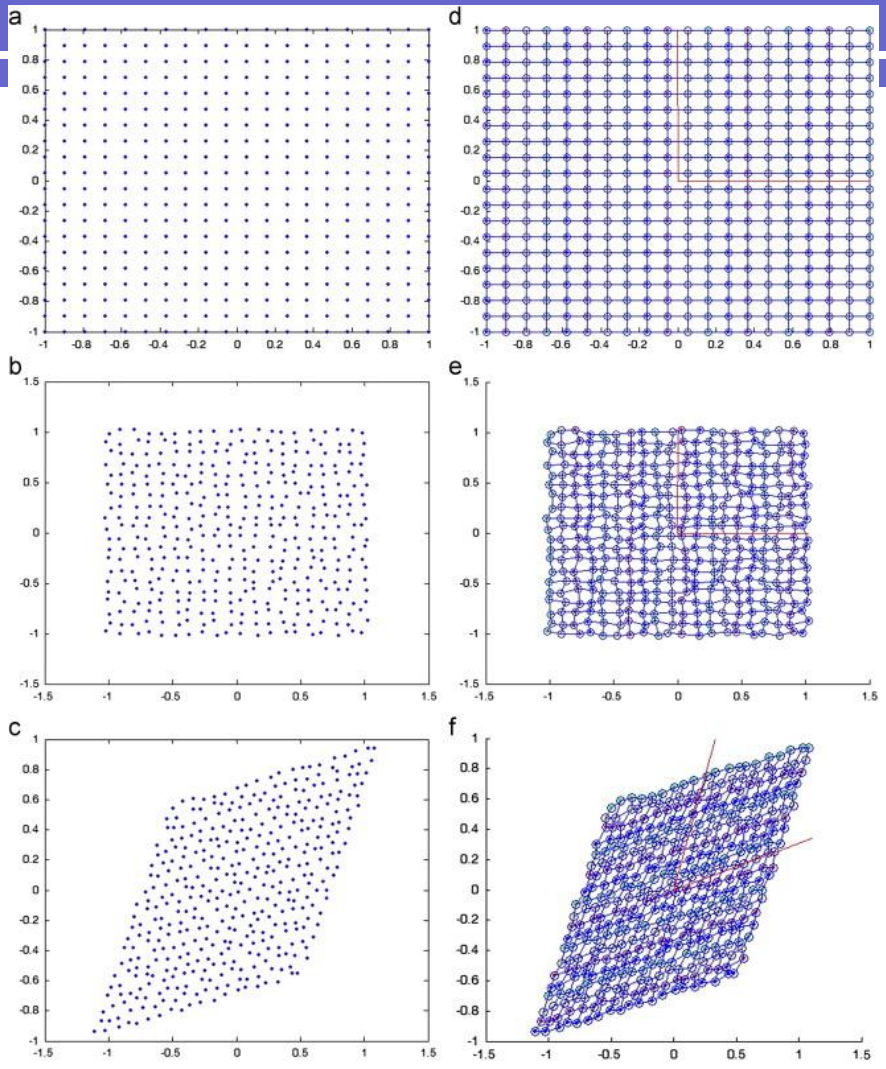
$$\mathrm{E}_k = \sum_t v_k[t]\left\|\mathbf{x}[t]-\mathbf{y}_i\right\|^2 , \mathrm{W}_\mathrm{k} = \sum_{h\in NB(k)}\left\|y_h - y_k\right\|^2$$

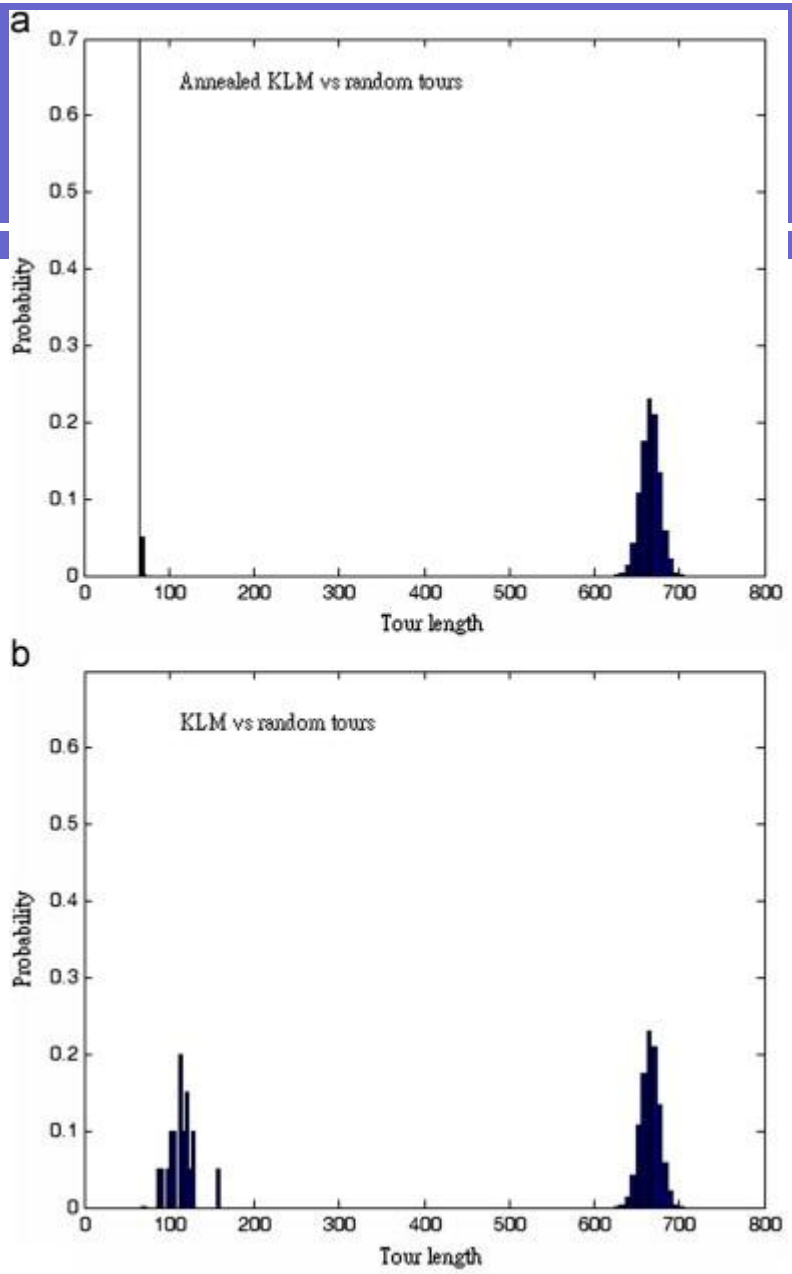
$$\frac{d(E_k + \lambda W_k)}{dy_i} = 0$$

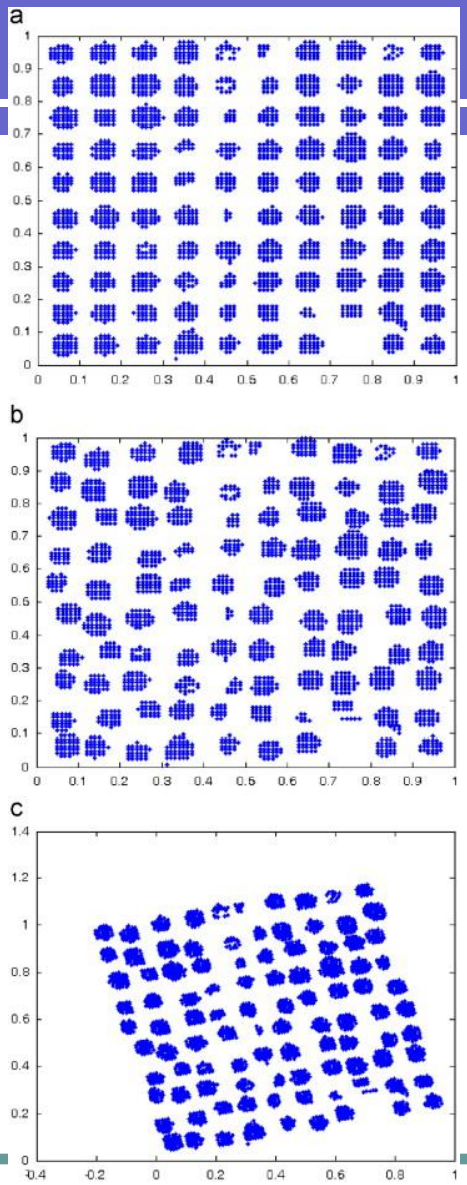
$$\sum_t v_k[t](\mathbf{x}[t]-\mathbf{y}_k) + \lambda \sum_{h\in NB(k)}(y_h - y_k) = 0$$

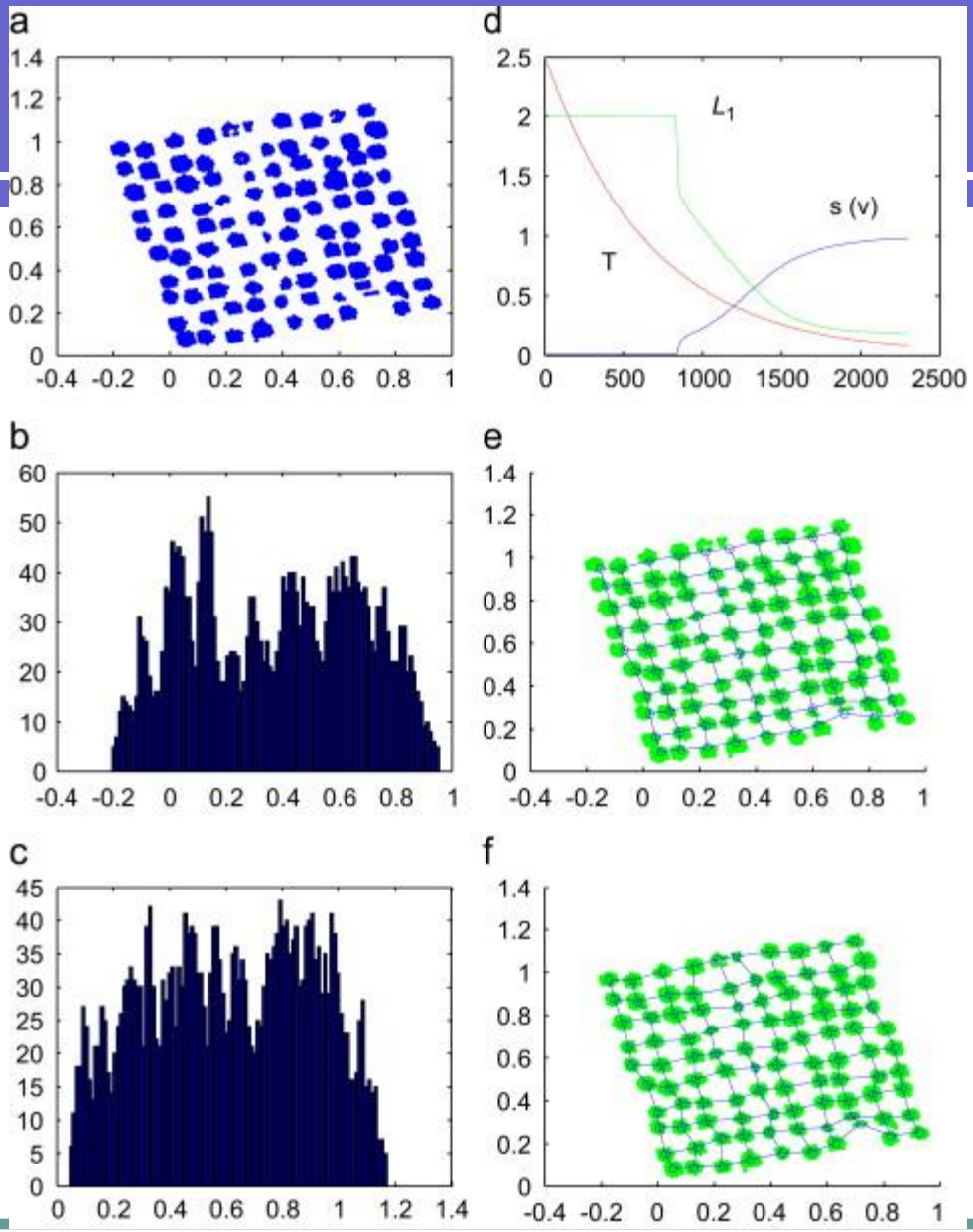
$$Solve \;\; y_k$$

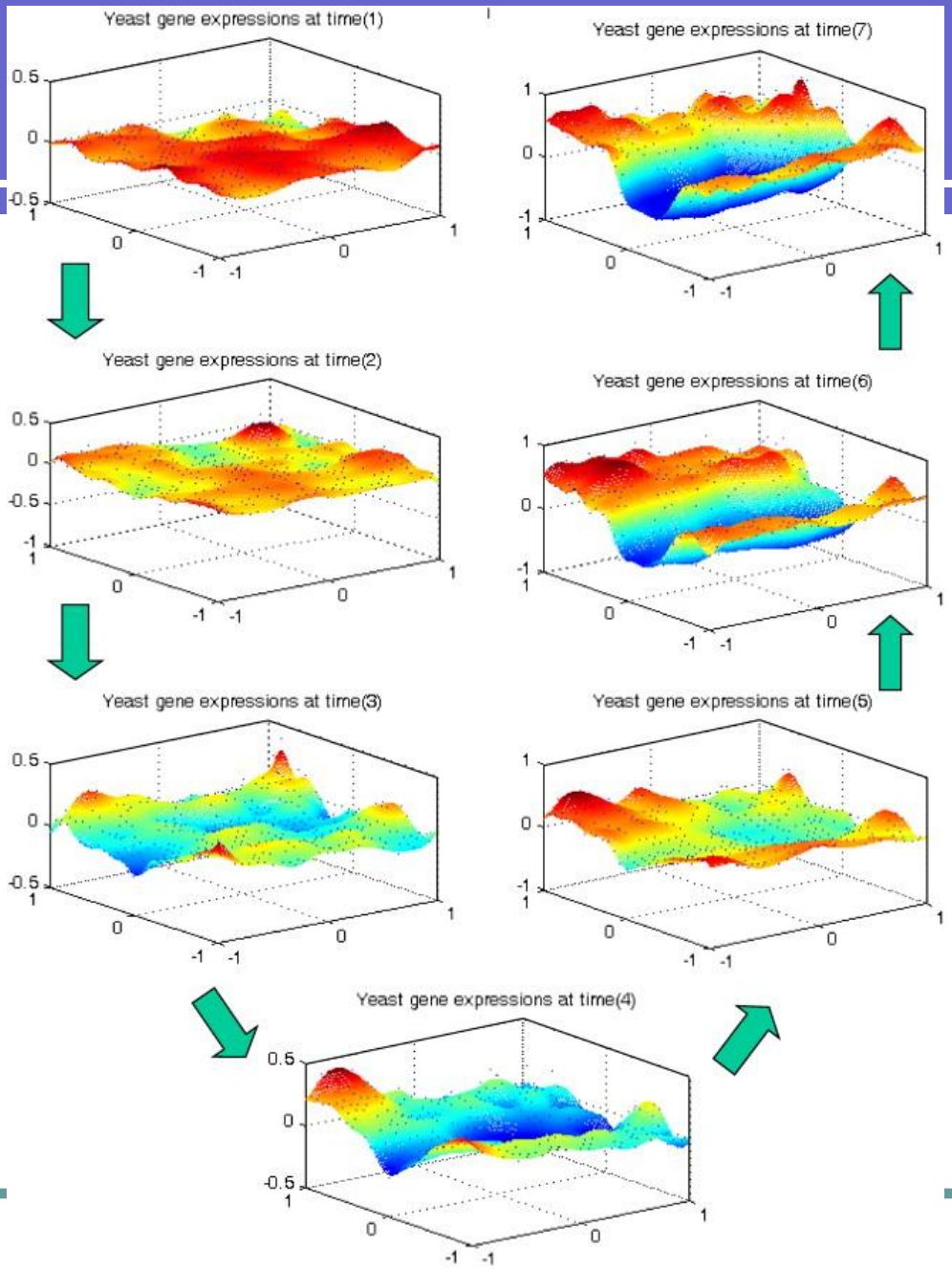

$p(\mathbf{x}; \mu_1, \mathbf{A})$
$p(\mathbf{x}; \mu_m, \mathbf{A})$
$p(\mathbf{x}; \mu_M, \mathbf{A})$
π_1
π_2
π_{M-1}
π_M
flip-coin

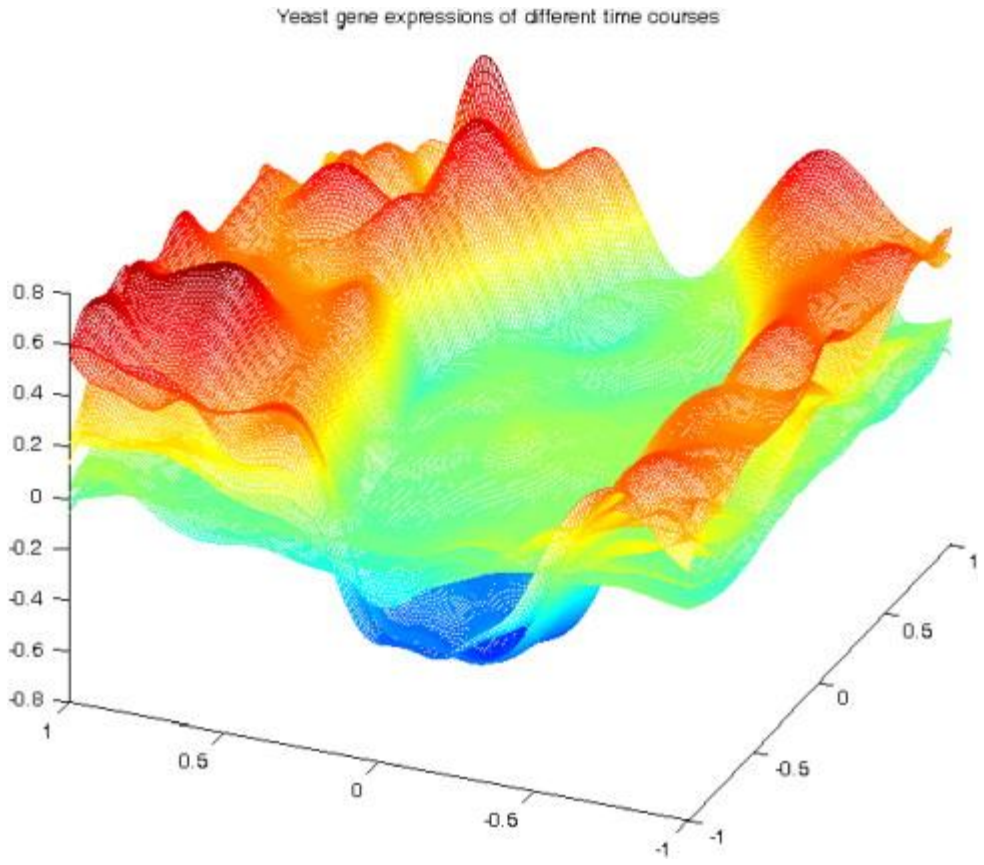

a
d
b
e
c
f


a
Annealed KLM vs random tours
Probability
Tour length
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0
100
200
300
400
500
600
700
800
b
KLM vs random tours
Probability
Tour length
0
0.1
0.2
0.3
0.4
0.5
0.6
0
100
200
300
400
500
600
700
800


a
b
c


a
d
b
e
c
f
L_1
s (v)
T

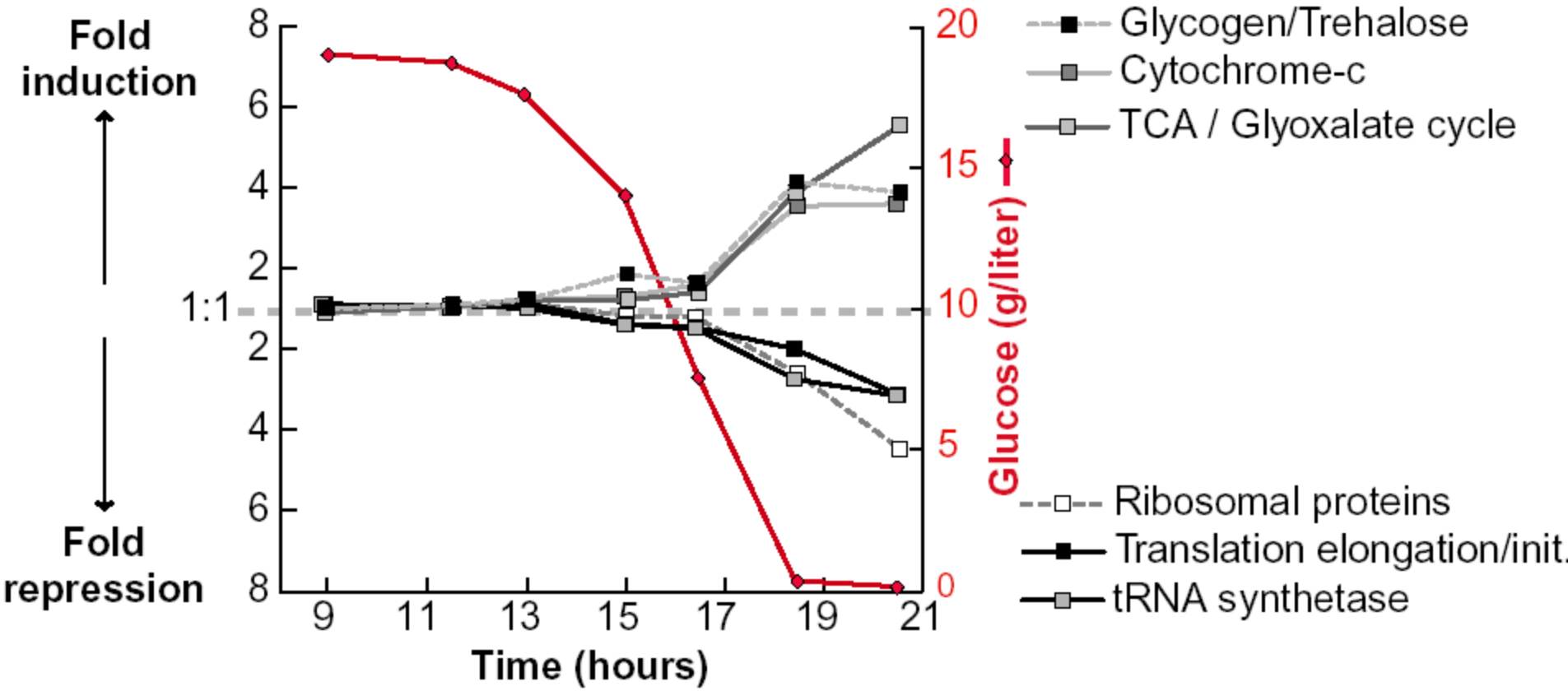

Yeast gene expressions at time(1)
Yeast gene expressions at time(7)
Yeast gene expressions at time(2)
Yeast gene expressions at time(6)
Yeast gene expressions at time(3)
Yeast gene expressions at time(5)
Yeast gene expressions at time(4)


Yeast gene expressions of different time courses
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
1
0.5
0
-0.5
-1
1
0.5
0
-0.5
-1

Microarray expressions of yeast genes

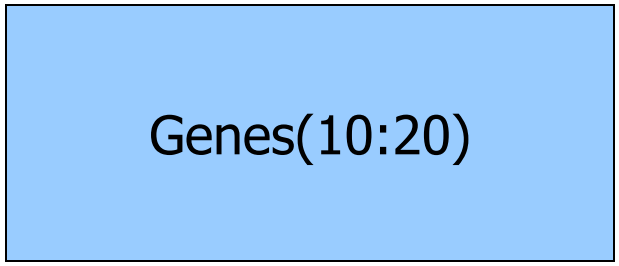
Pat_Brown_Lab_Home_Page

Exploring the Metabolic and Genetic Control of Gene Expression on a Genomic Scale



Load gene matrix

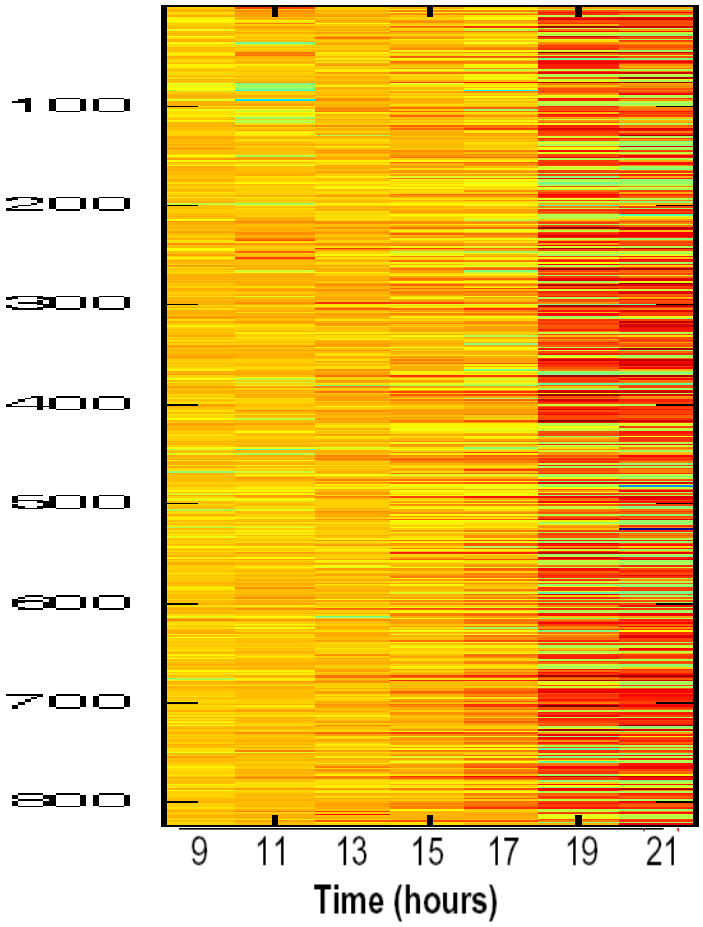
- Load yeastdata822



Display names of genes numbered between 10 and 20

Gene matrix: 822x7

Gene id



Problem

- Apply the k-means algorithm to process data_25
- Apply the annealed k-means algorithm to process data_25
- Numerical simulations for comparisons of the two algorithms
  - Quantitative evaluations
  - Statistics of mean and variance summarized over 10 executions

Problem

- Apply the k-means algorithm to process yeast_822
- Apply the annealed k-means algorithm to process yeast_822
- Numerical simulations for comparisons of the two algorithms
  - Quantitative evaluations
  - Statistics of mean and variance summarized over 10 executions

Gene clustering by kmeans

- Load data

```
load yeastdata822.mat;
times=[0 9.5 11.5 13.5 15.5 17.5 19.5];
plot(times,yeastvalues);
```

Gene clustering by kmeans

Kmeans analysis

```
[cidx, ctrs] = kmeans(yeastvalues, 16);
```

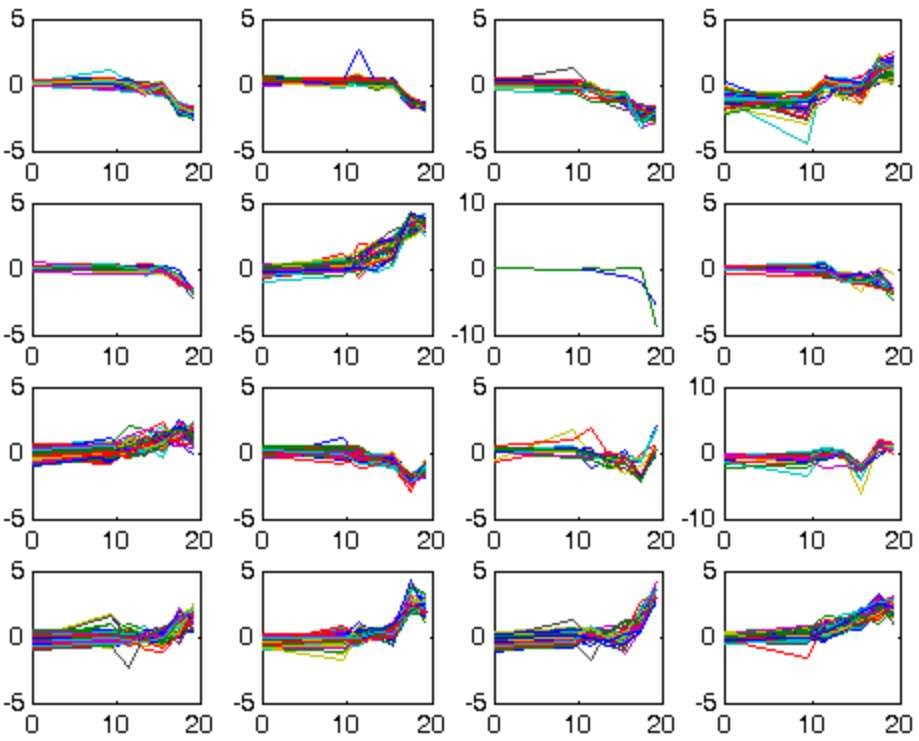
Display

```
for c = 1:16
    subplot(4,4,c);
    plot(times,yeastvalues((cidx == c),:)');
end
```

Genes belonging the second cluster

```
genes(cidx==2)
```

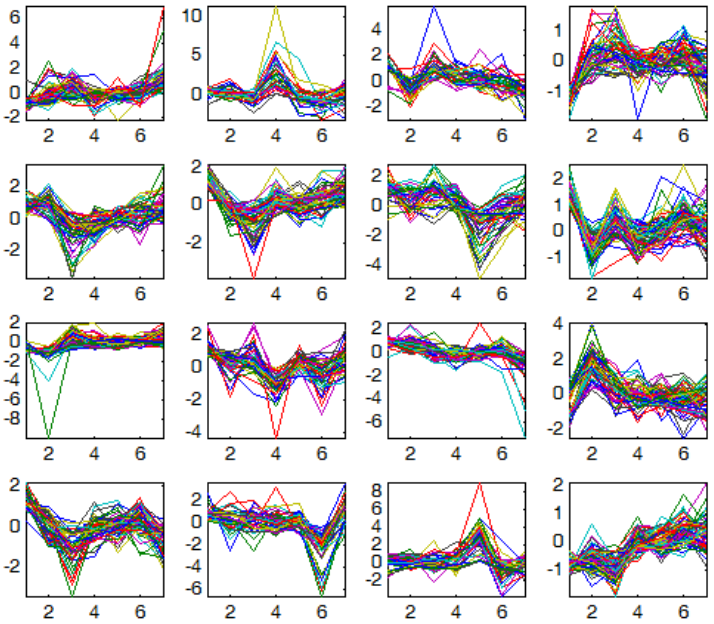
Expressions of 16 gene clusters



Statistical Dependency

```
load yeastdata822.mat;
X=yeastvalues;
XX=X';
W=jader(XX);
Y=W*XX;
X=Y';
```

yeast_ICA_kmeans.m



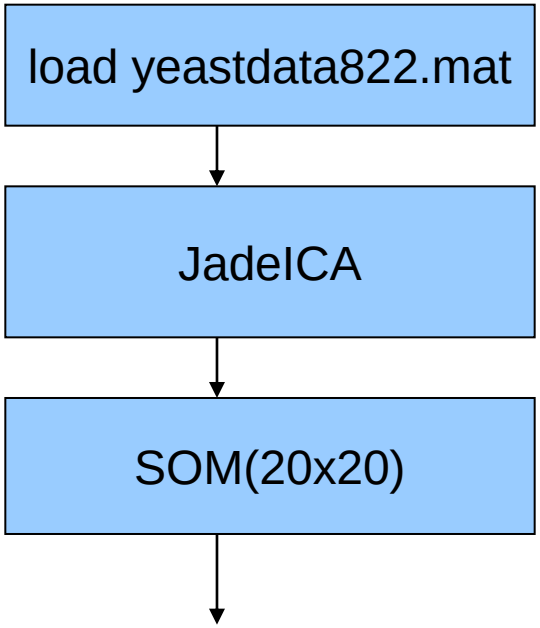
Problem : Yeast_ICA_Kmean

- Apply Jade_ICA to translate yeast data to independent components
- Is it reasonable to employ Euclidean distance to measure similarity of genes?
- What is the impact of ICA on gene clustering?
- Numerical simulations
  - ICA
  - Annealed k-means clustering
  - Describe clusters of genes

Problem

- Apply the Kohonen's self-organizing algorithm to sort yeast_822 on a lattice
- Refer to Neurocomputing 74(12-13), 2228-2240 (2011)
- Visualize yeast gene expressions at each time course

Gene_ICA_SOM



demo_gene_som.m

yeast822_som_20x20.mat

Cross Distances

```
Centers=net.iw{1,1};
Y=Centers;
X=P';

M=size(Y,1);N=size(X,1);
A=sum(X.^2,2)*ones(1,M);
C=ones(N,1)*sum(Y.^2,2)';
B=X*Y';
D=sqrt(A-2*B+C);
```

$$D_{ij} = (\mathbf{x}_i - \mathbf{y}_j)(\mathbf{x}_i^T - \mathbf{y}_j^T)$$

$$= \mathbf{x}_i\mathbf{x}_i^T - 2\mathbf{x}_i\mathbf{y}_j^T + \mathbf{y}_j\mathbf{y}_j^T$$

$$= A_{ij} + B_{ij} + C_{ij}$$

Cross_Distance Matrix

- D: 822x400
- Cross distances between 822 genes and 400 gene centers
- 400 clusters

Query

- How many genes in each cluster?

```
[xx ind]=min(D');
sum(ind==k); % how many genes in the kth cluster
```

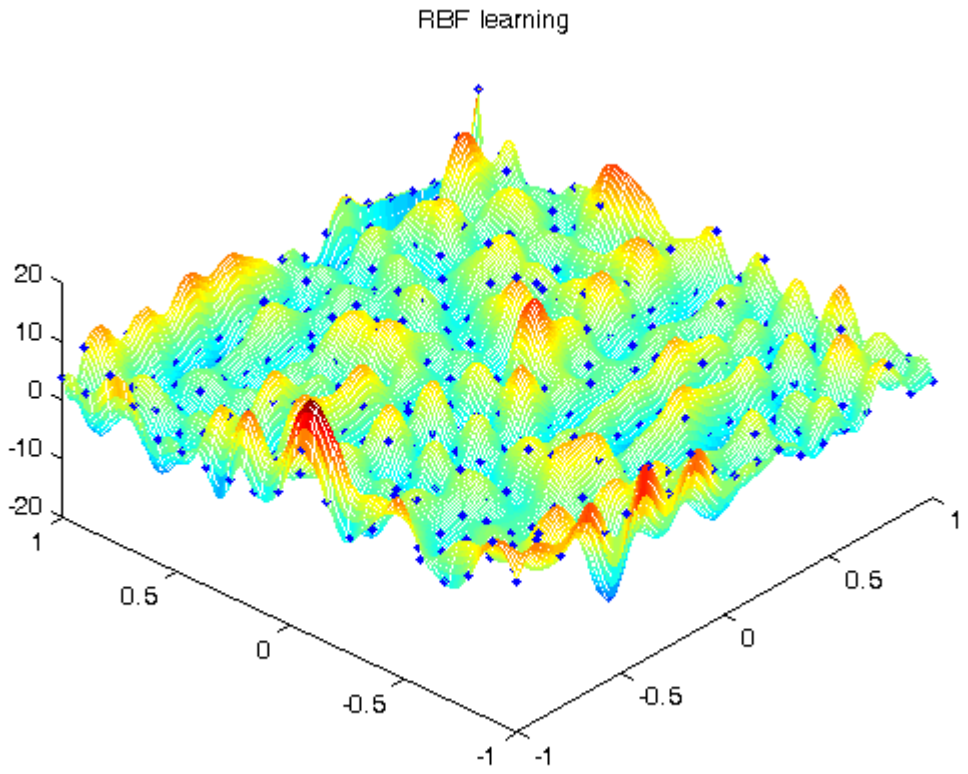
- Which cluster a gene belongs?

```
ind(i);
```

Visualization of gene density

```
[xx ind]=min(D'); K=20; M=K^2;
for k=1:M
    num(k)=sum(ind==k); % how many genes in the kth cluster
end
a =  linspace(-1,1,K);
x =  [];
for i=1:K
            xx = [ones(20,1)*a(i)  a'];
            x   = [x ; xx];
end
y = num; x=x' ;
```

Fitting (x,y)



Visualization of gene expressions

```
Centers=net.iw{1,1};
Y=Centers;
a =  linspace(-1,1,K);
x =  [];
for i=1:K
            xx = [ones(20,1)*a(i)  a'];
            x   = [x ; xx];
end
time = 1;
y = Y(:, time); x=x'
```

Problem

- Apply ICA to translate yeast_822 to independent components
- Apply SOM to process independent components of yeast_822
- Visualize yeast gene expressions at each time course

Problem

- Apply ICA to translate yeast_822 to independent components
- Apply annealed SOM to process independent components of yeast_822
- Visualize yeast gene expressions at each time course