Introduction

- ..\課程教學計劃表2018.pdf

Classification

- Handwriting-digit recognition
- Handwriting-character recognition
- A function from a set of handwriting images to labels

classification by construction of mathematical functions

- mathematical functions
- parametric mathematical functions
- function domain
- function co-domain

Handwriting 99 Multiplication

Handwriting Mult...

打開

…

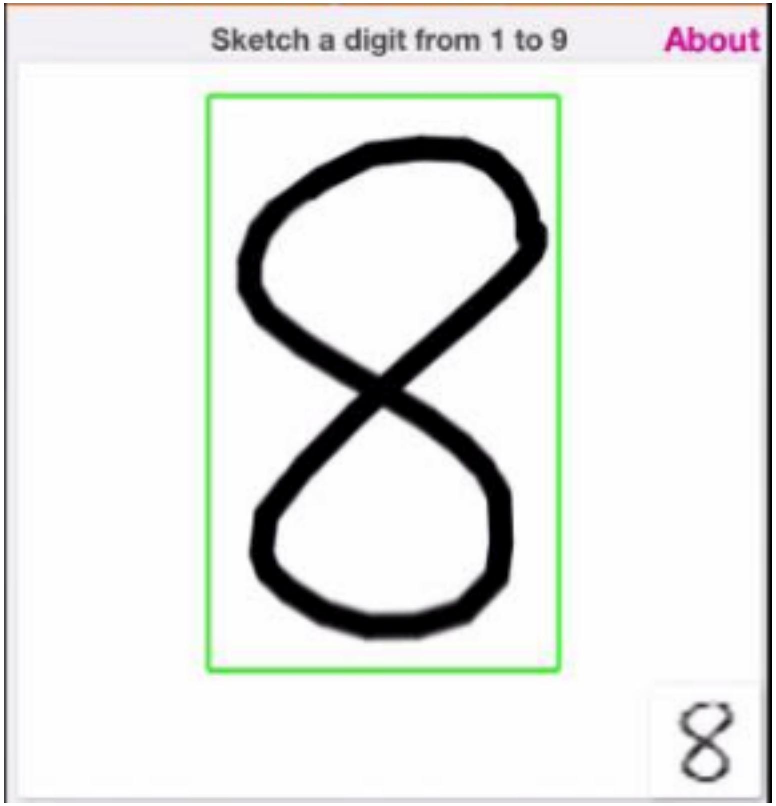
4.7 ★★★★★

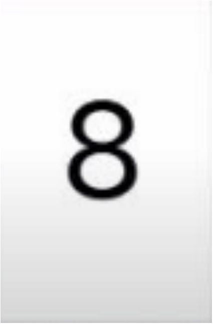
10份評分

4+

年齡

domain: a set of 280,000 hw-digit images

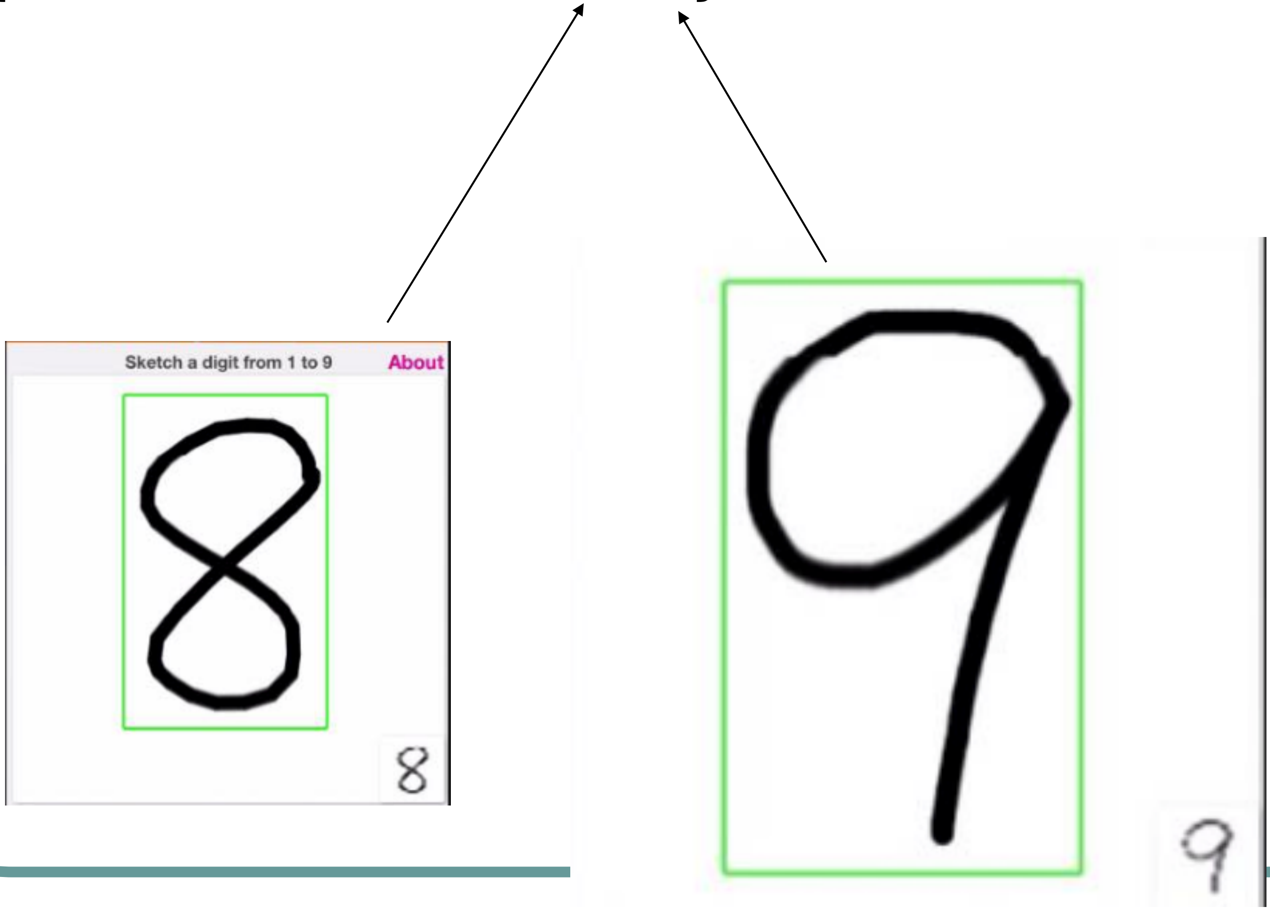




0000000100

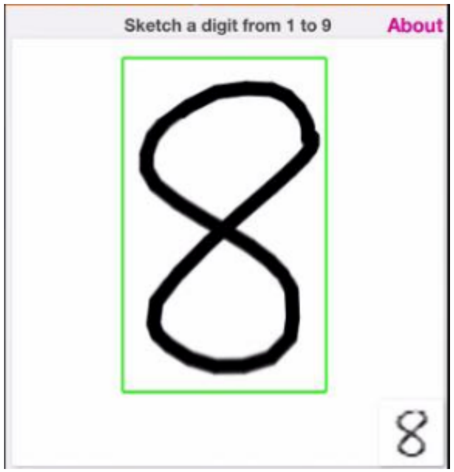
codomain

- {0,1,2,3,4,5,6,7,8,9}

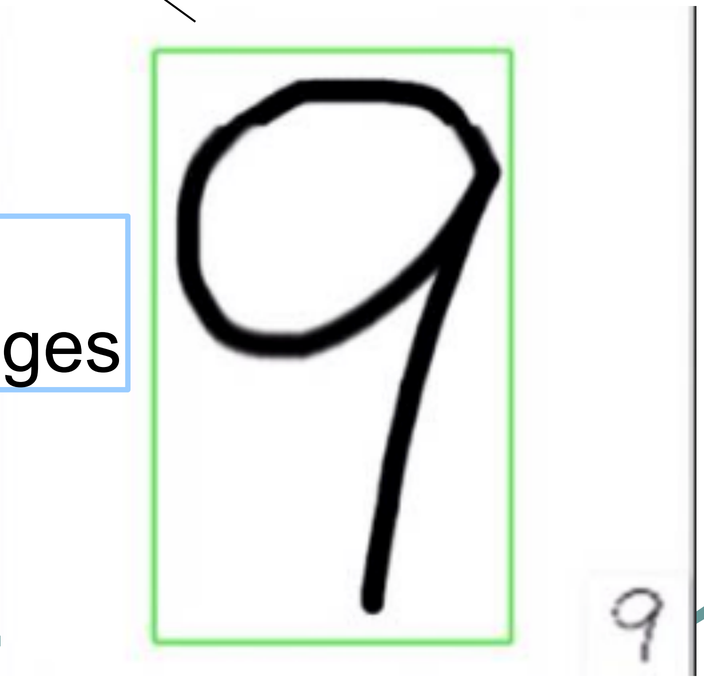


codomain

- {0,1,2,3,4,5,6,7,8,9}



280,000
hw-digit images



How to construct a function

- How to construct a function for classifying 280,000 handwriting digits ?
- Extension: how to construct a function for classifying more than 580,000 handwriting English-Letters ?
- What is the function?

Symbolic analysis

- Symbolic differentiation
- Symbolic integrations

diff

source codes

```
% input a string to specify a function
% find its derivative
ss=input('function:','s');
fx=inline(ss);
ss=['diff(' ss ')'];
ss1=eval([sprintf(ss)]);
fx1=inline(ss1)
```

Function Derivation

- Input a function
- Plot its derivative

source code

```
% input a string to specify a function
% plot its derivative
ss=input('function of x:','s');
fx=inline(ss);
x=sym('x');
ss=['diff(' ss ')'];
ss1=eval([sprintf(ss)]);
fx1=inline(ss1)
x=linspace(-1,1);
plot(x,fx(x),'b');hold on;
plot(x,fx1(x),'r');
```

2nd derivative

source codes

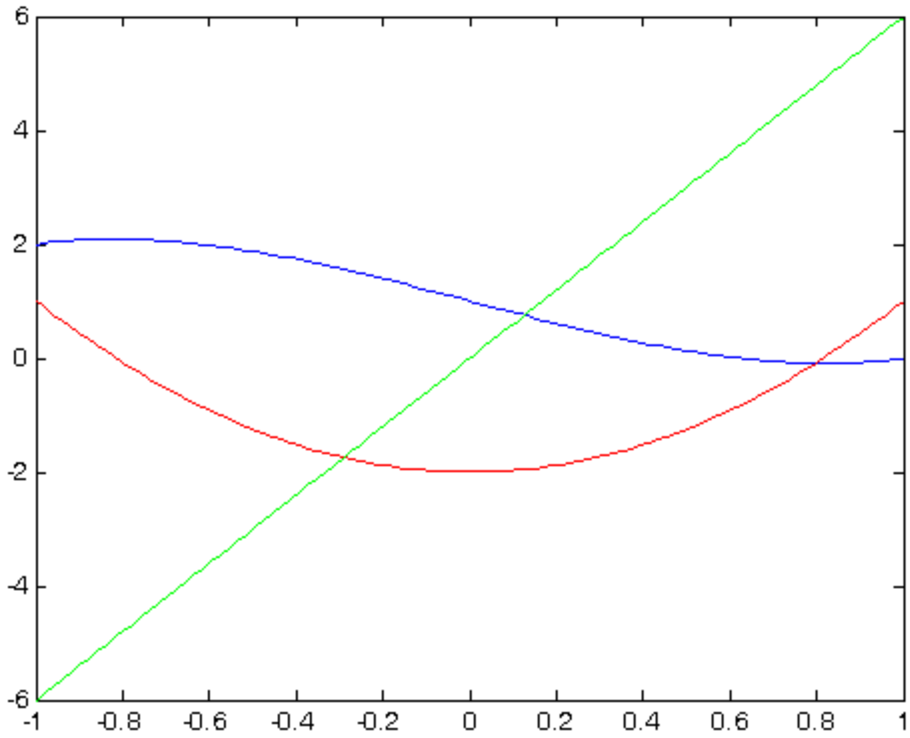
```
>> f_diff3
function of x:x.^3-2*x+1

fx1 =

     Inline function:
     fx1(x) = 3.*x.^2-2


fx2 =

     Inline function:
     fx2(x) = 6.*x
```



Symbolic integration

```
>> x=sym('x');
>> int(x.^2+x+1)

ans =

1/3*x^3+1/2*x^2+x
```

Symbolic integration

- int(1/(1+x^2)) returns atan(x)

  [Symbolic integration]

Numerical methods

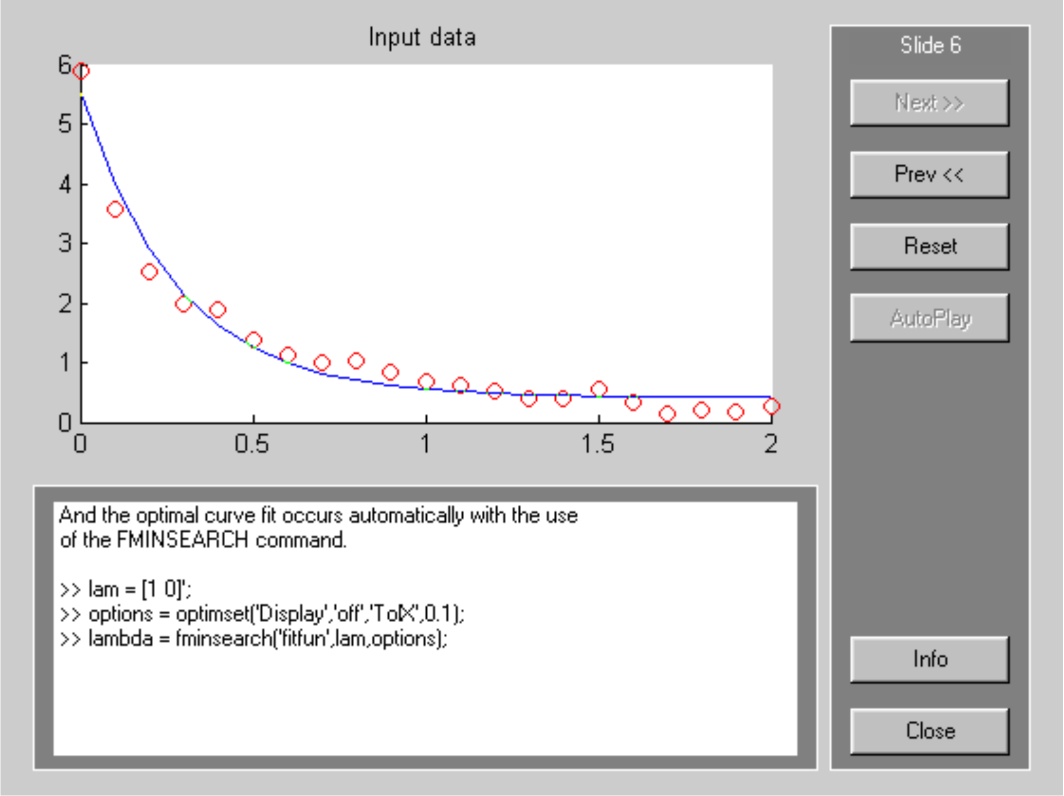
- Integration and differentiation
- Polynomial interpolation
- Linear system
- Nonlinear system
- Spline
- Initial value problem

Dimensionality and nonlinearity

	Linear	Nonlinear
Single variable	***Line fitting***	***Polynomial fitting spine***
Multiple variables	***Linear system***	***Function approximation***

Interpolation

Nonlinear function of Single variable



Linear system

Linear relation of multiple variables

$$\mathbf{Ax=b}$$

Given A and b, find x

Ex.

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

inv

```
A=[1 2 3;3 2 1;1 0 1];
inv(A)
```

Approximation by spline functions

Nonlinear function of Single variable

- Quadratic spline
- Cubic spline
- Natural cubic spline
- B-splines

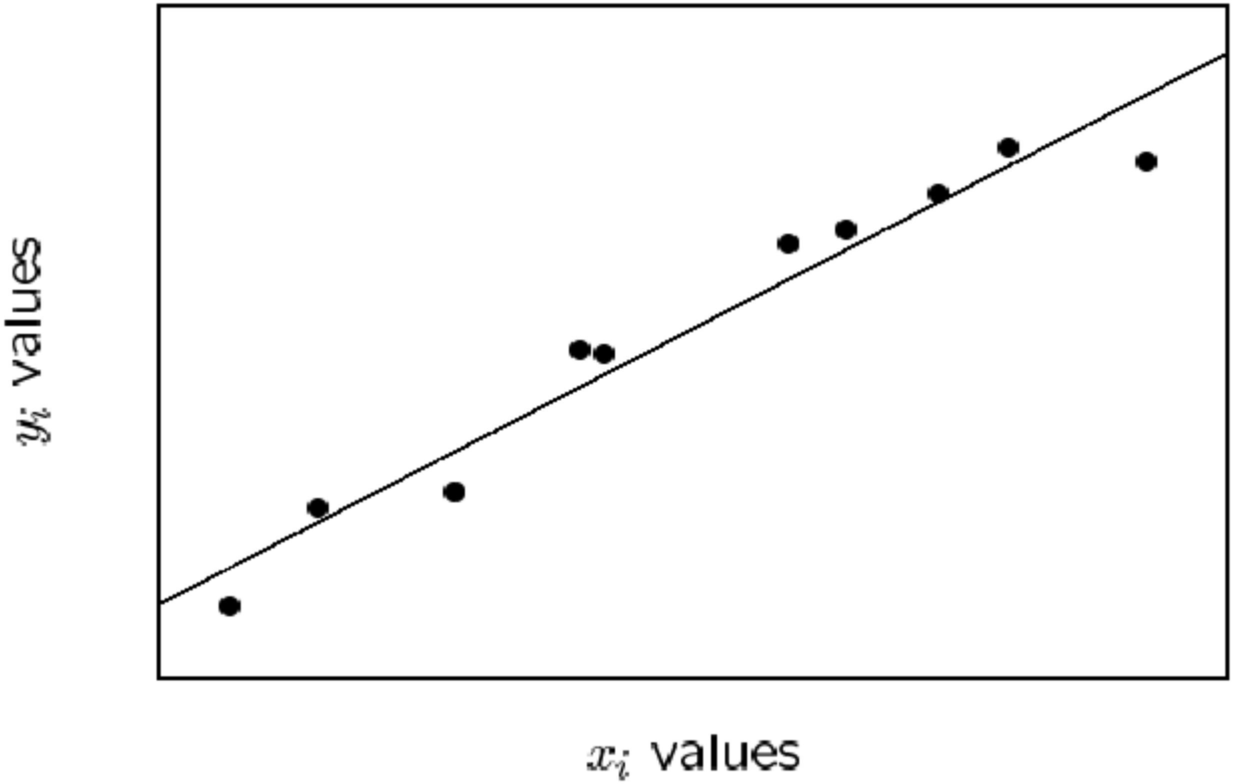
Pseudo inverse

- pinv
- Left-division

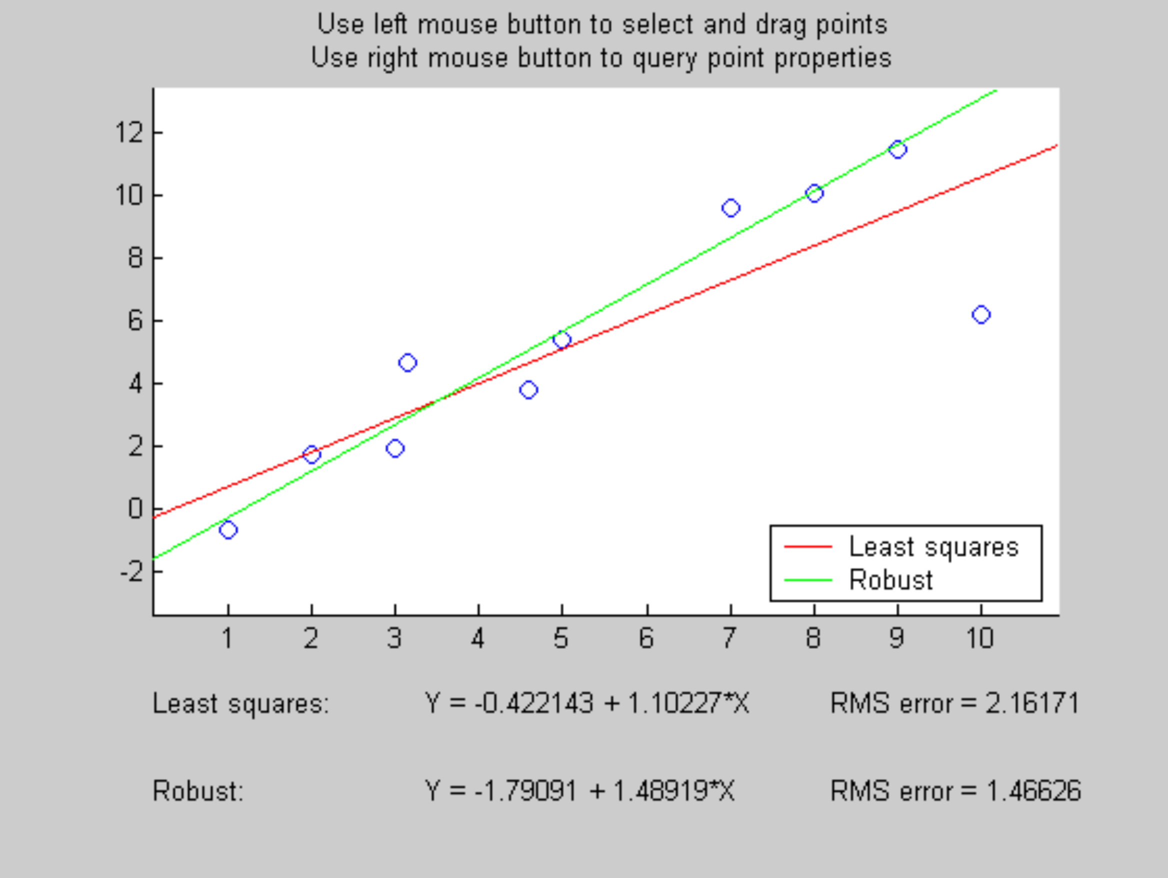
Least square methods

- Line fitting
- Hyper-plane fitting
- Nonlinear approximation

Line fitting



Smoothing and the least square method



Hyper-plane fitting

$$Given\ \{(\mathbf{a}_i, b_i) \mid \mathbf{a}_i \in R^2, b_i \in R\}_{i=1}^{N}$$

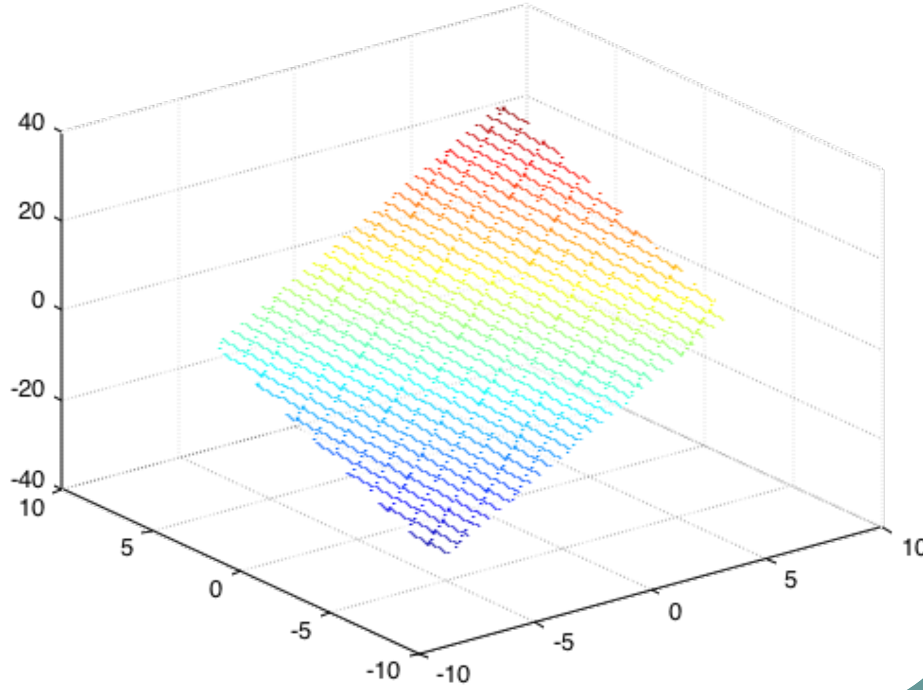
$$Assume\ \ \mathbf{a}^T\mathbf{x} = b$$

$$find\ \ \mathbf{x}\ \text{to minimize}\ \mathrm{E}(\mathbf{x})$$

Linear projection

$$y_1 = a_{11}x_1 + a_{12}x_2$$

$$y_1 = 2x_1 + 3x_2$$



Solving nonlinear systems

Iterative approaches

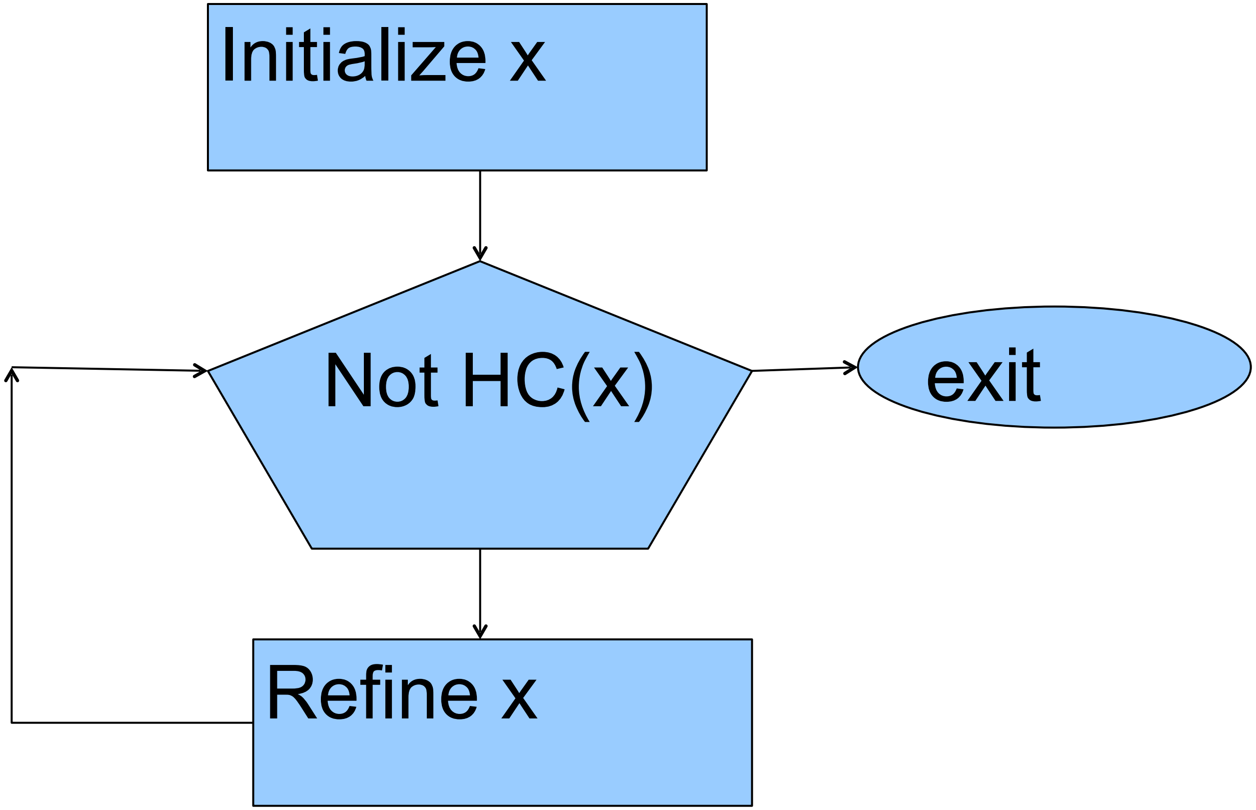
Newton's method

The steepest descent method

Iterative approach

- Iterative approach
- Random guess
- Refine current solution iteratively
- Convergence
- Computational complexity

Control flow



- While looping
- HC(x) : halting condition
- Refining rule : improve current feasible solution according to criteria defined for problem solving

Example: Nonlinear system

$$3x_1 - \cos(x_2 x_3) - \frac{1}{2} = 0$$

$$x_1^2 - 81(x_2 + 0.1)^2 + \sin(x_3) + 1.06 = 0$$

$$e^{-x_1 x_2} + 20x_3 + \frac{1}{3}(10\pi - 3) = 0$$

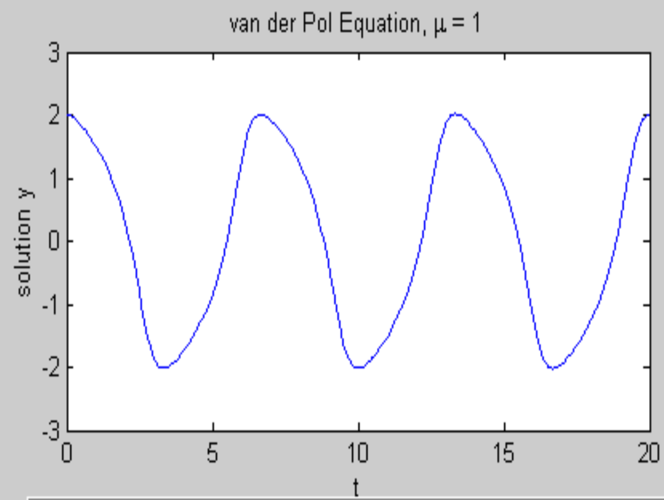
Initial Value Problem for ODEs

Initial Value Problem for ODEs

van der Pol equation with a parameter μ = Mu

$$y'' - \mu(1-y^2)y' + y = 0$$

Initial conditions: $y(0)=2,\ y'(0)=0$



The equation, written as a system of two first order ODEs, is evaluated in functions vdp1 for Mu = 1 and vdp1000 for Mu = 1000.

To examine vdp1, use the command

```
>> type vdp1;
```

The solution is plotted with

```
>> plot(t,y(:,1))
>> xlabel('t')
>> ylabel('solution y')
>> title('van der Pol Equation, \mu = 1')
```

Data Clustering

- K-means
- Annealed K-means

Parallel and distributed processes

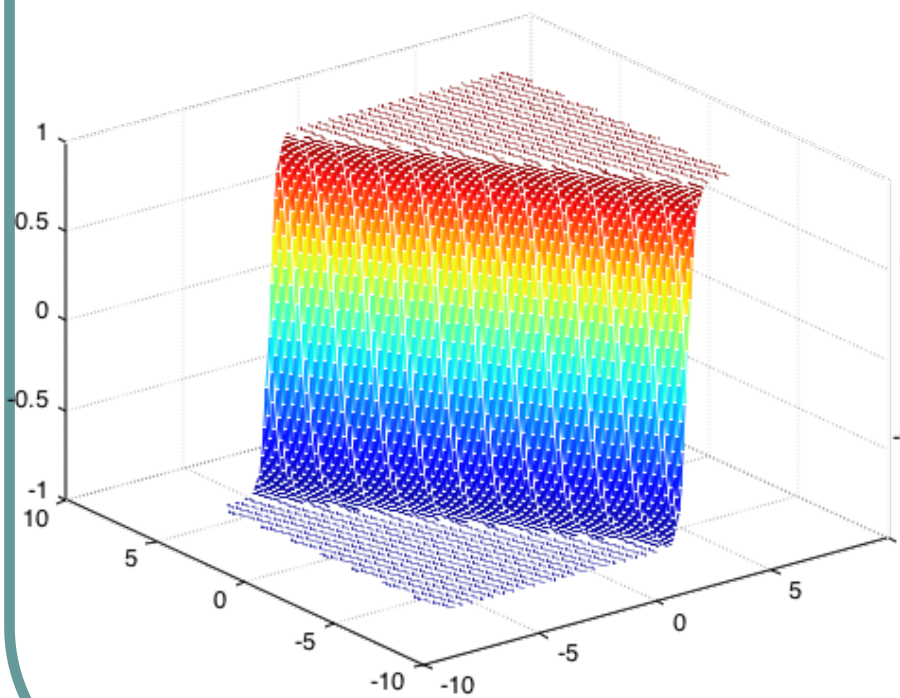
- Matlab - Parallel and distributed processes

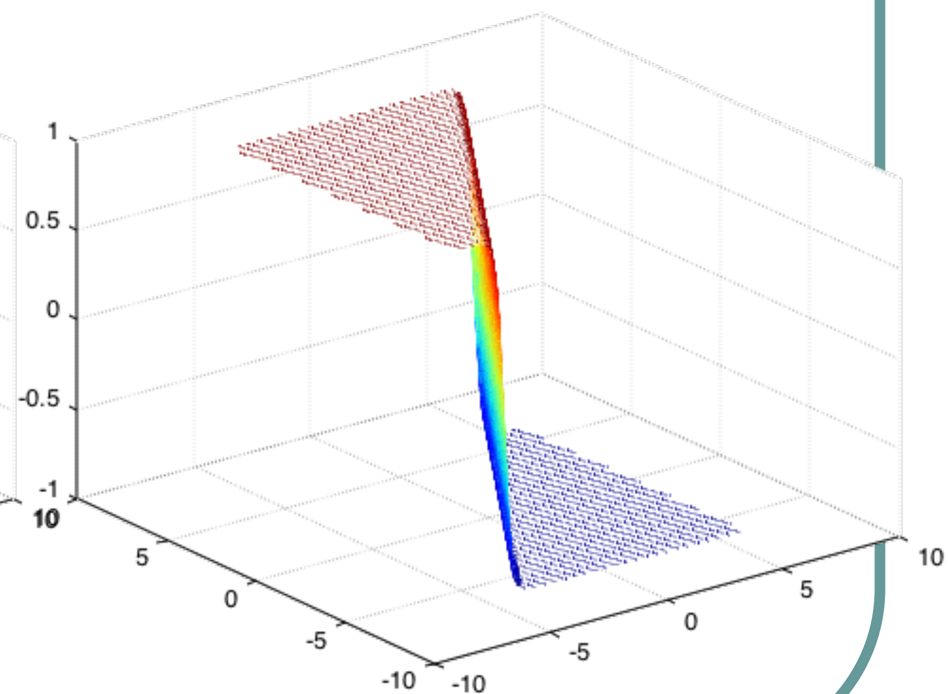
Function Approximation

- Multiple input variables
- Nonlinear mapping from domain to range

tanh

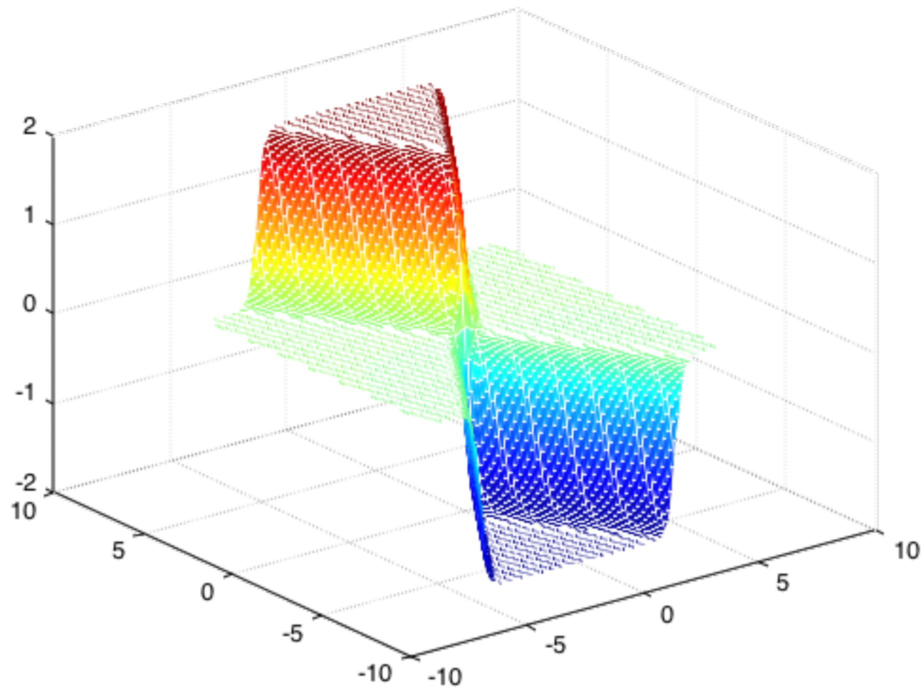
$$f(x1,x2)=y_1=2x_1+3x_2 \qquad f(x1,x2)=y_2=2x_1-3x_2$$





Two post-nonlinear projections

$$f(x1, x2) = \tanh(2x_1 + 3x_2) + \tanh(2x_1 - 3x_2)$$



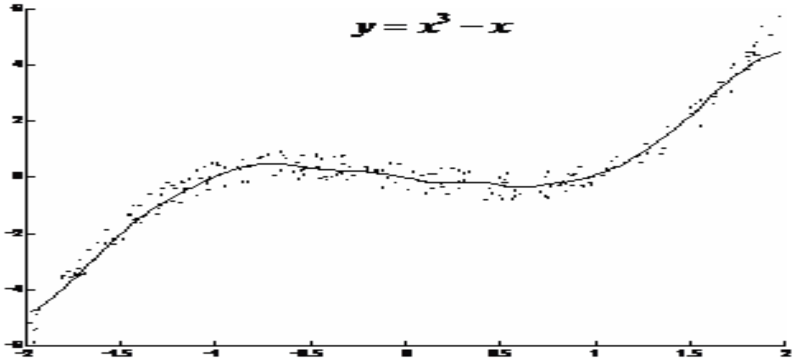
Intelligence computations

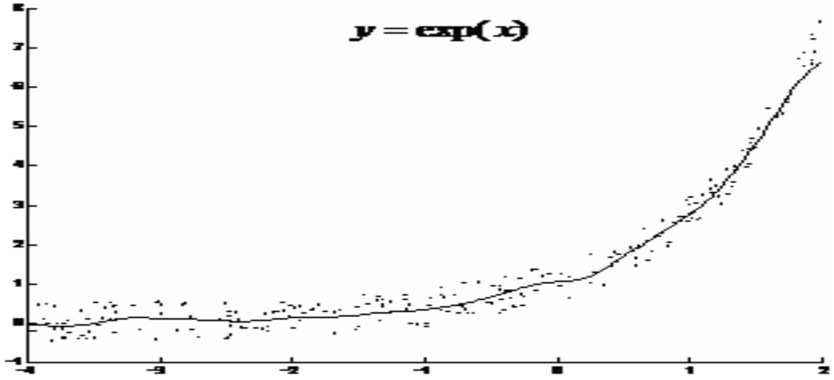
- Neural networks
- Machine Learning
- Data analysis
- Numerical computations

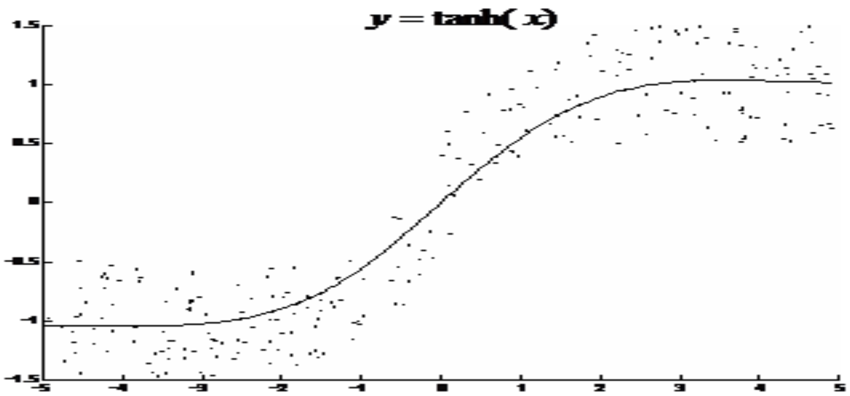
Function Approximation

- Function Approximation
  - R^d to R
  - Levenberg-Marquardt (LM) learning
  - MLP(Multilayer perceptrons) networks
  - MLPotts(multilayer Potts perceptrons) networks
  - Radial Basis Function
  - General Perceptrons
  - Modular NRBF neural networks

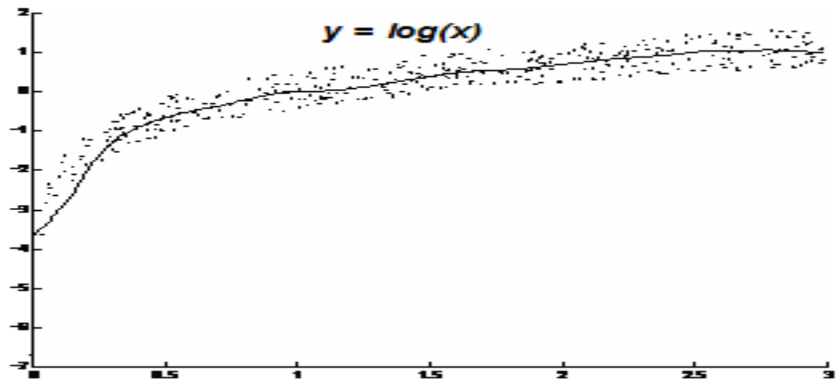
Function approximation

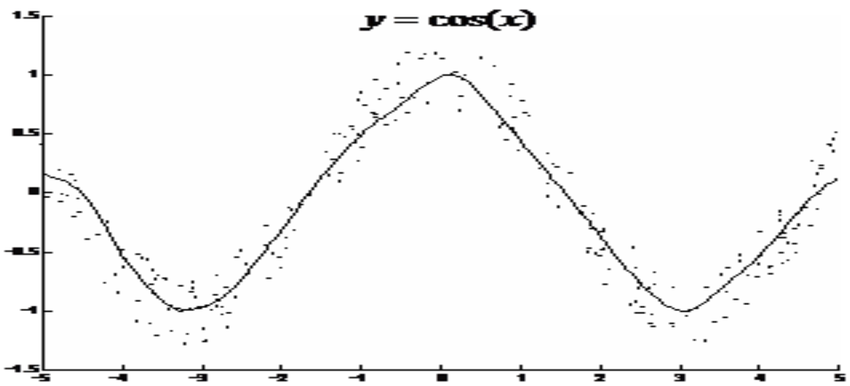






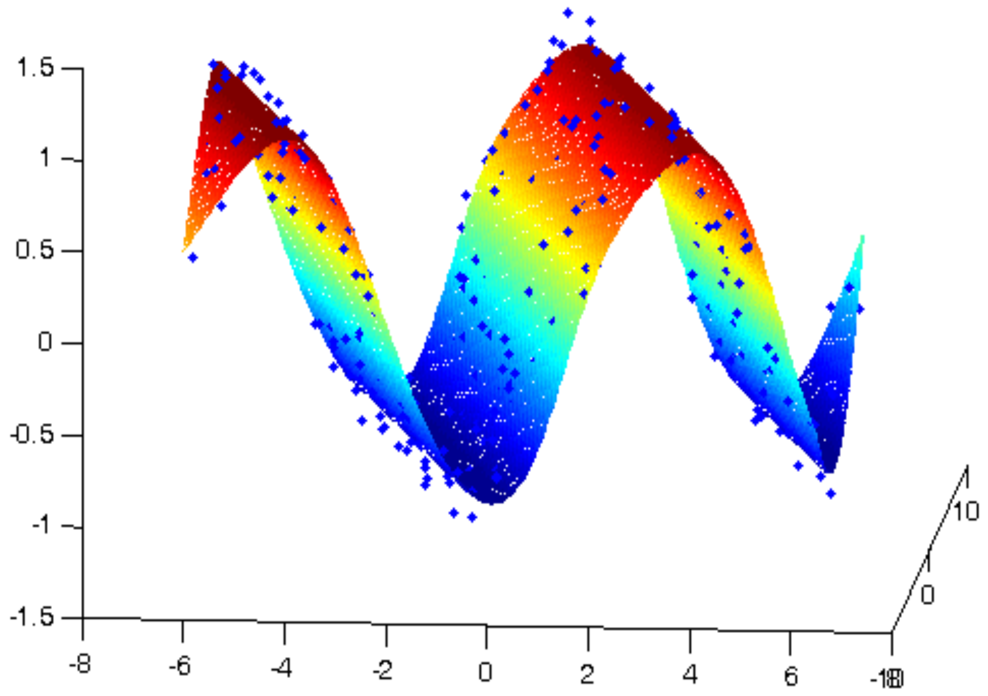
One-dimensional function approximation





LM learning for MLP

Two-dimensional Function Approximation

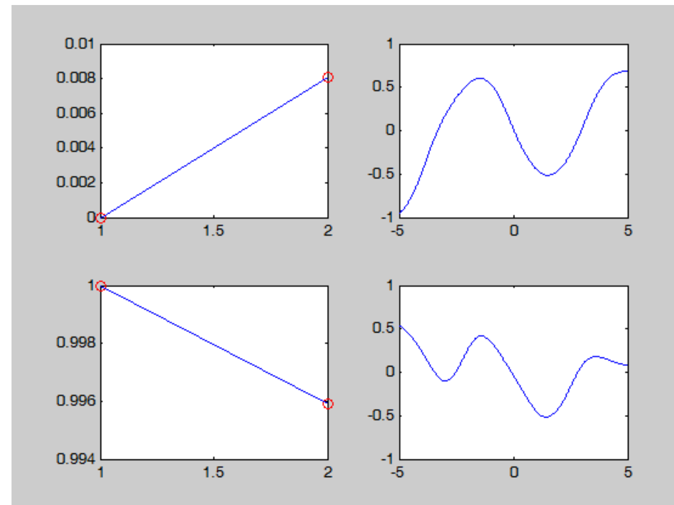


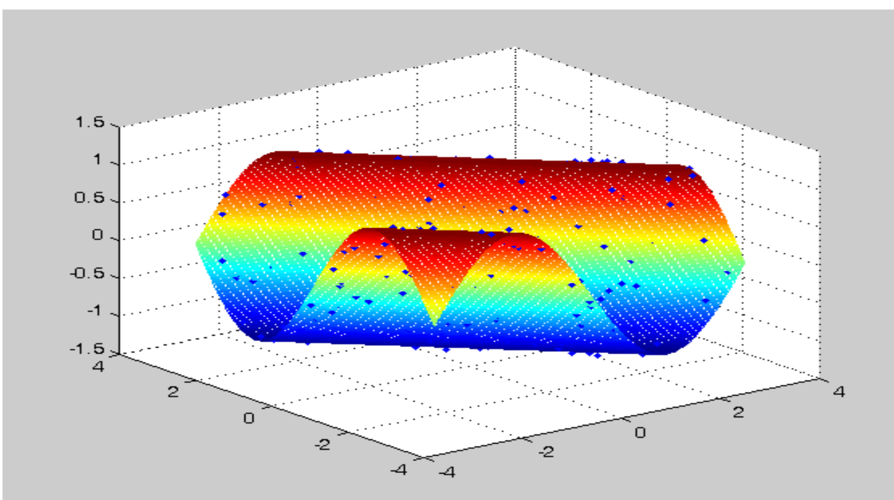
NNSYSID (2001)

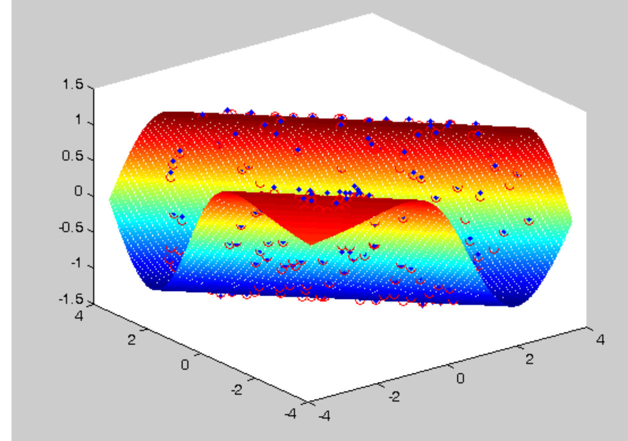
- The NNSYSID Toolbox
- Implementation of learning MLP networks by the LM method

Learning MLPotts networks

Two-dimensional function approximation by learning MLPotts networks (Wu 2008)

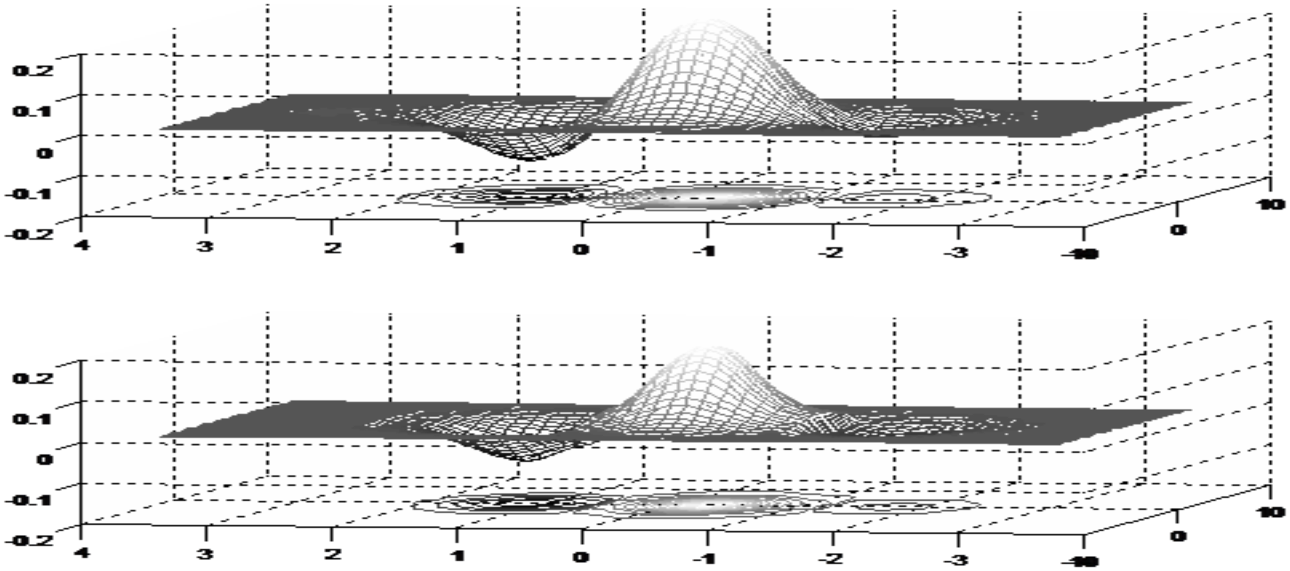






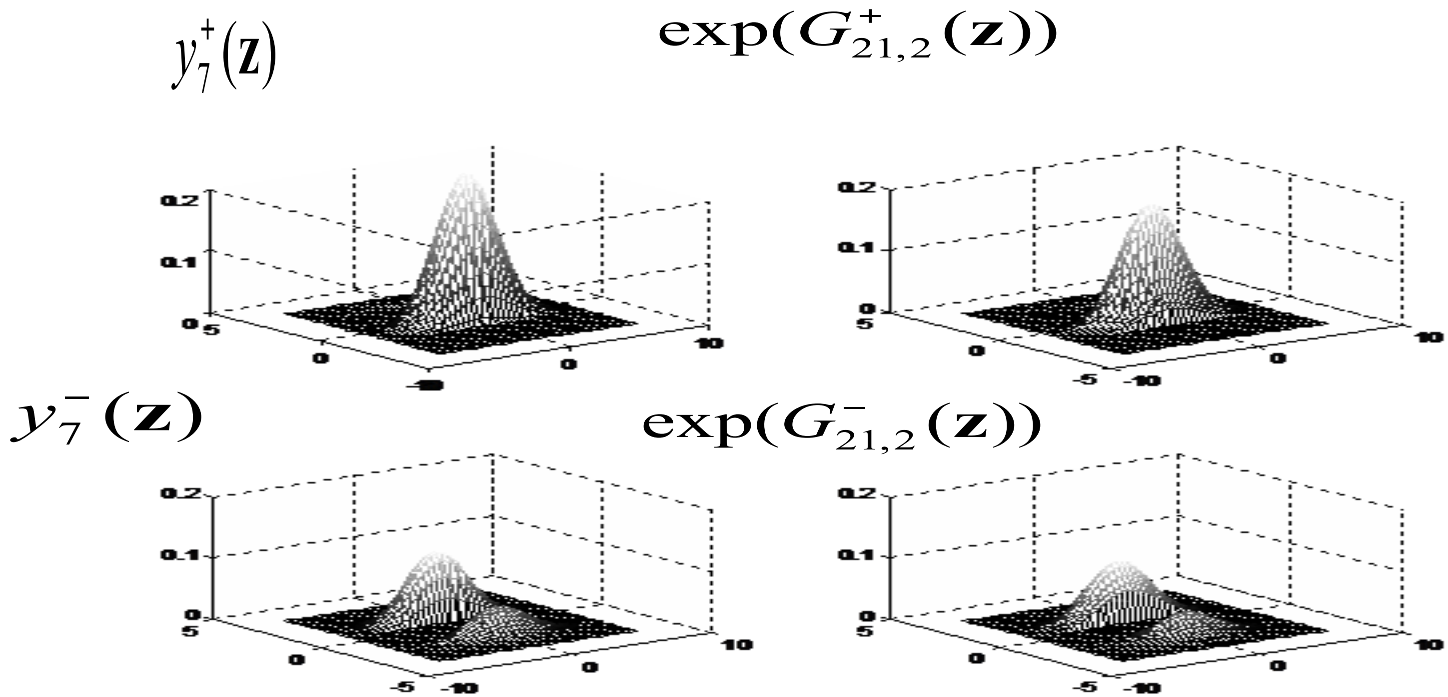
Approximating Gabor function

Learning generalized adalines (Wu et al 2006)



Approximating Gabor function

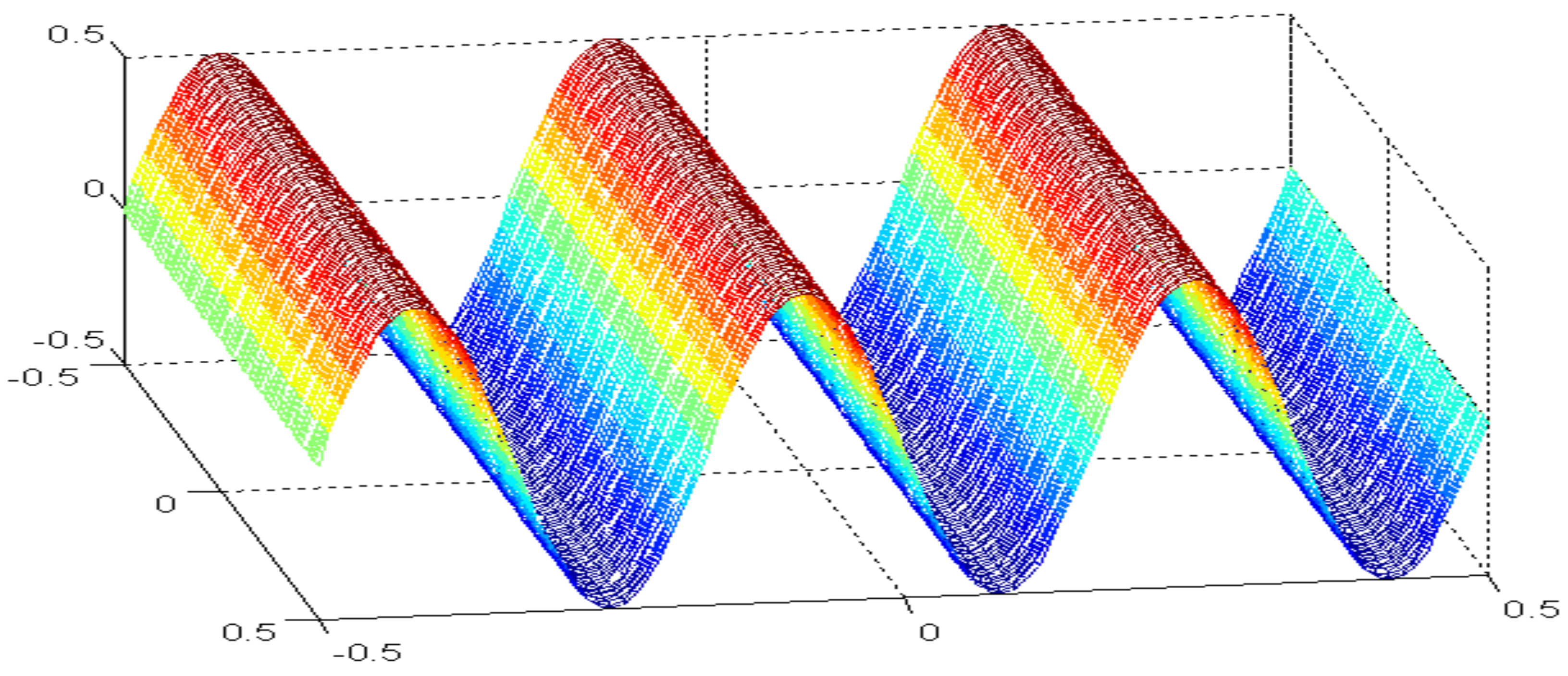
Learning gadaline networks

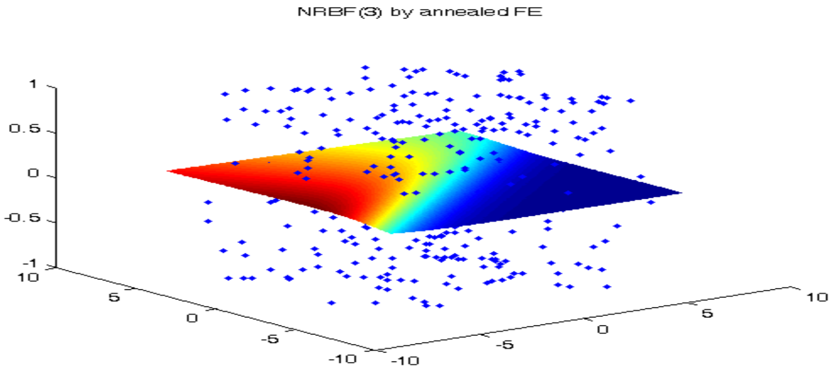


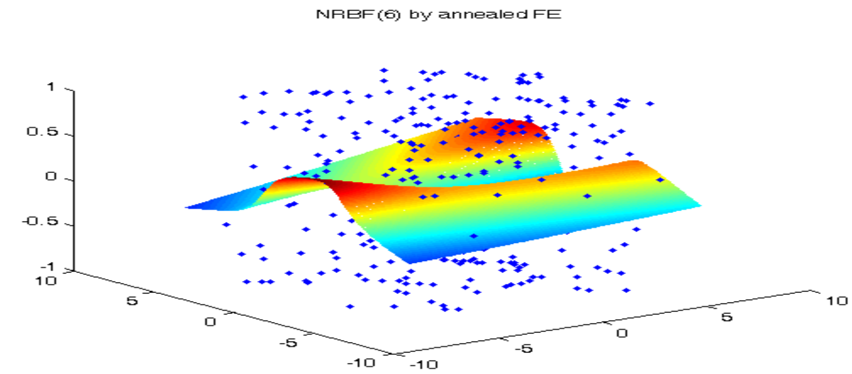
Sinusoidal function approximation

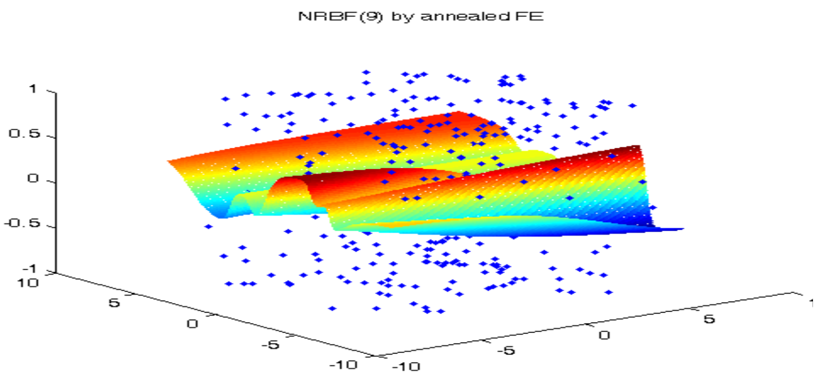
Learning gadaline networks (Wu et al 2006)

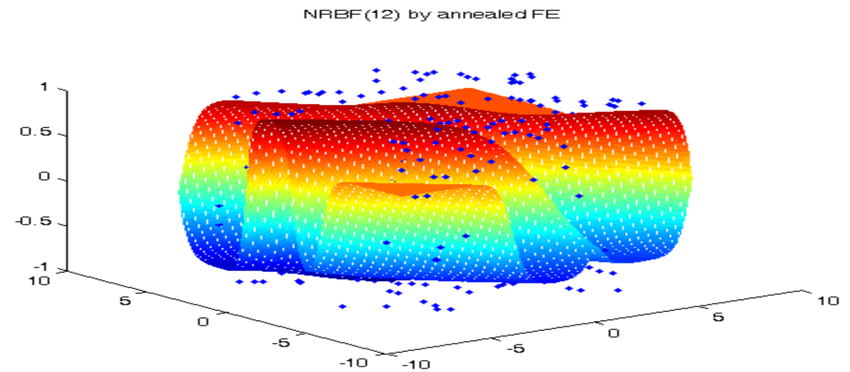
$y(\mathbf{z})$

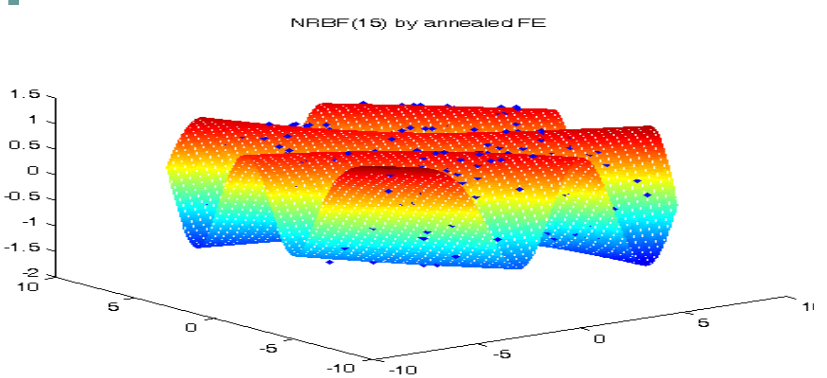


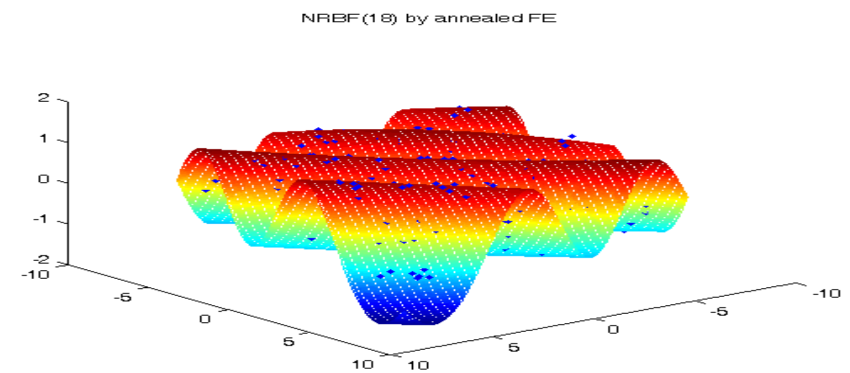

NRBF(3) by annealed FE

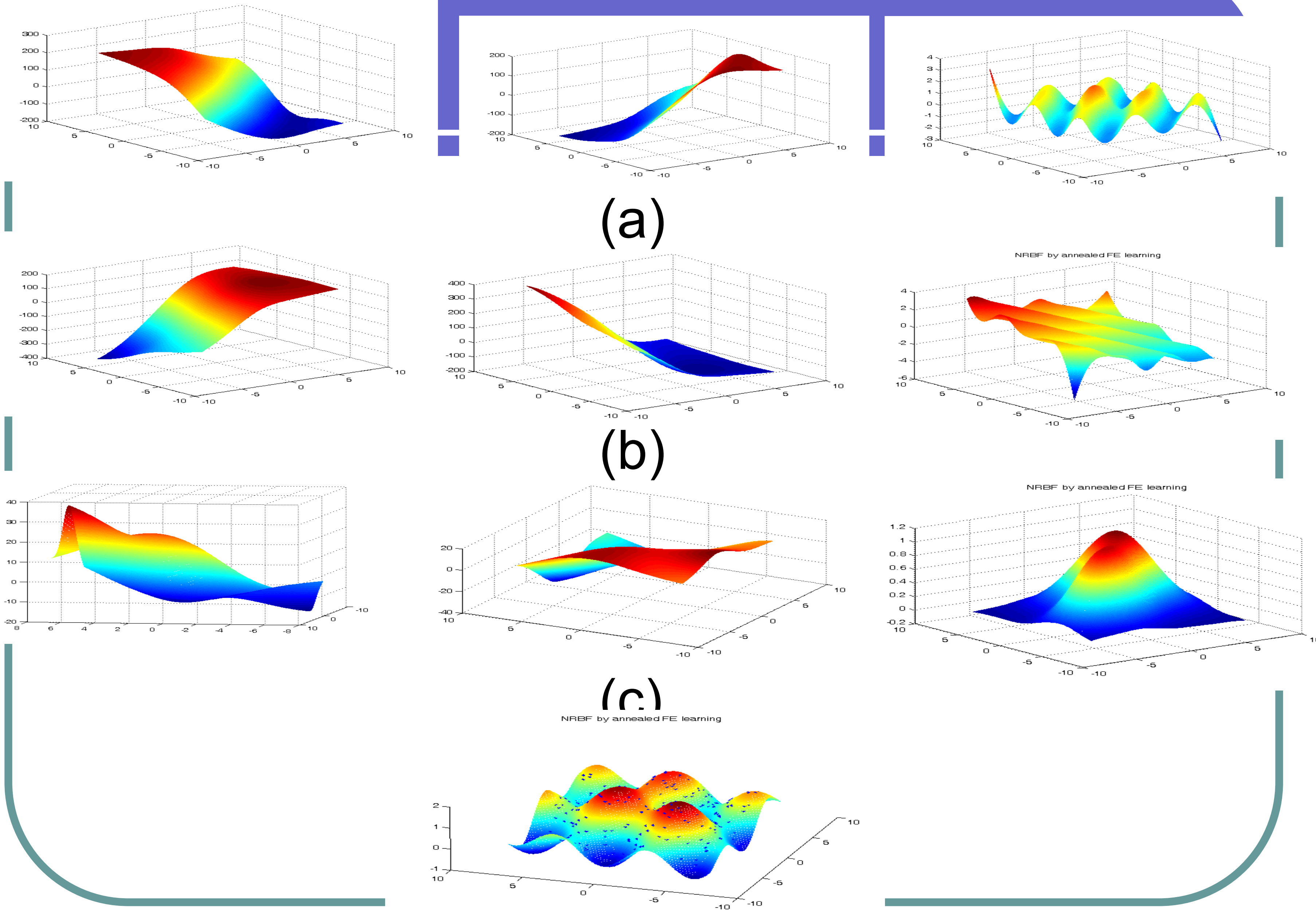

NRBF(6) by annealed FE


NRBF(9) by annealed FE


NRBF(12) by annealed FE

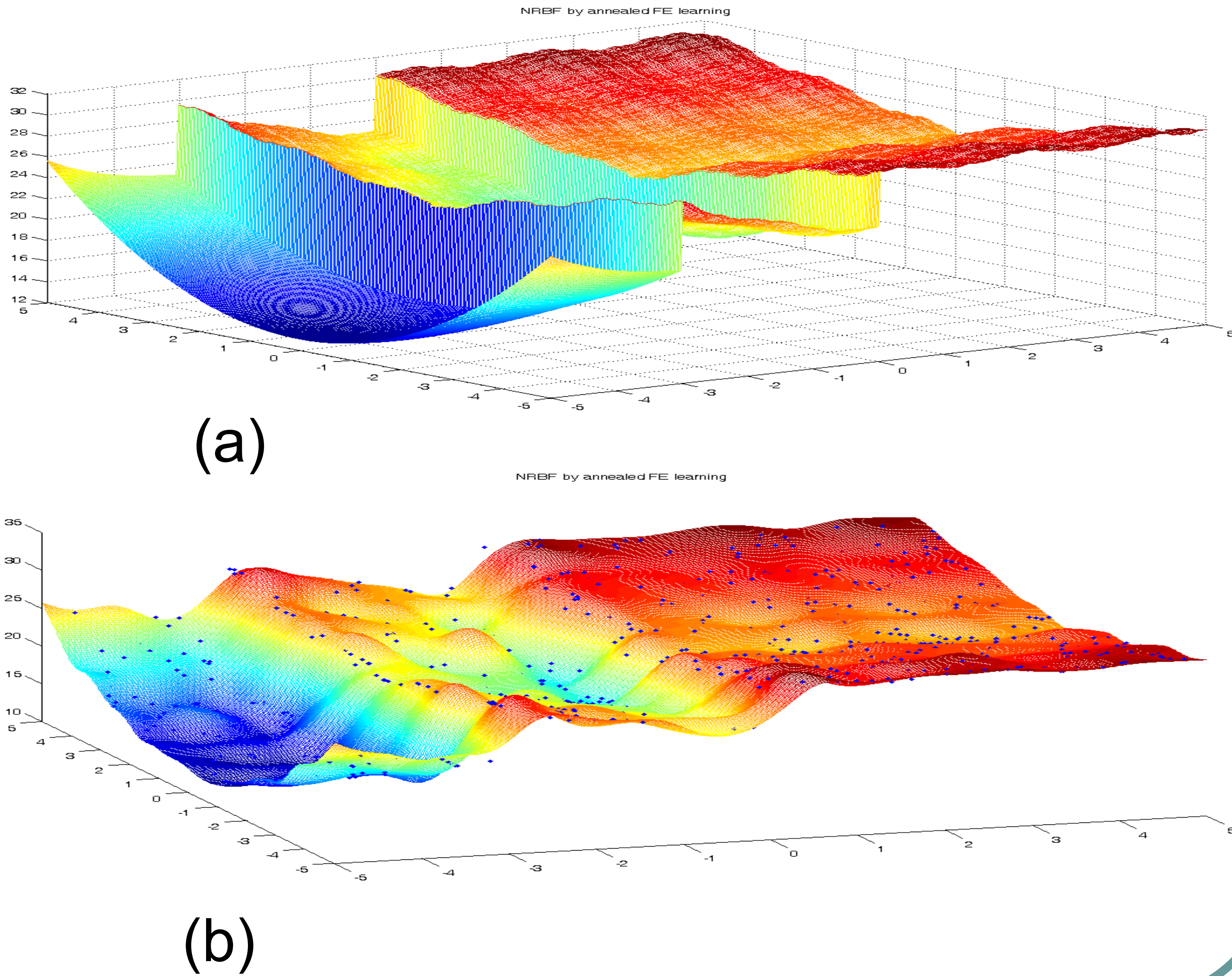

NRBF(15) by annealed FE


NRBF(18) by annealed FE



(a)
NRBF by annealed FE learning
(b)
NRBF by annealed FE learning
(c)
NRBF by annealed FE learning
(d)

Figure 6



NRBF by annealed FE learning
32
30
28
26
24
22
20
18
16
14
12
5
4
3
2
1
0
-1
-2
-3
-4
-5
(a)
NRBF by annealed FE learning
35
30
25
20
15
10
(b)

Figure 8

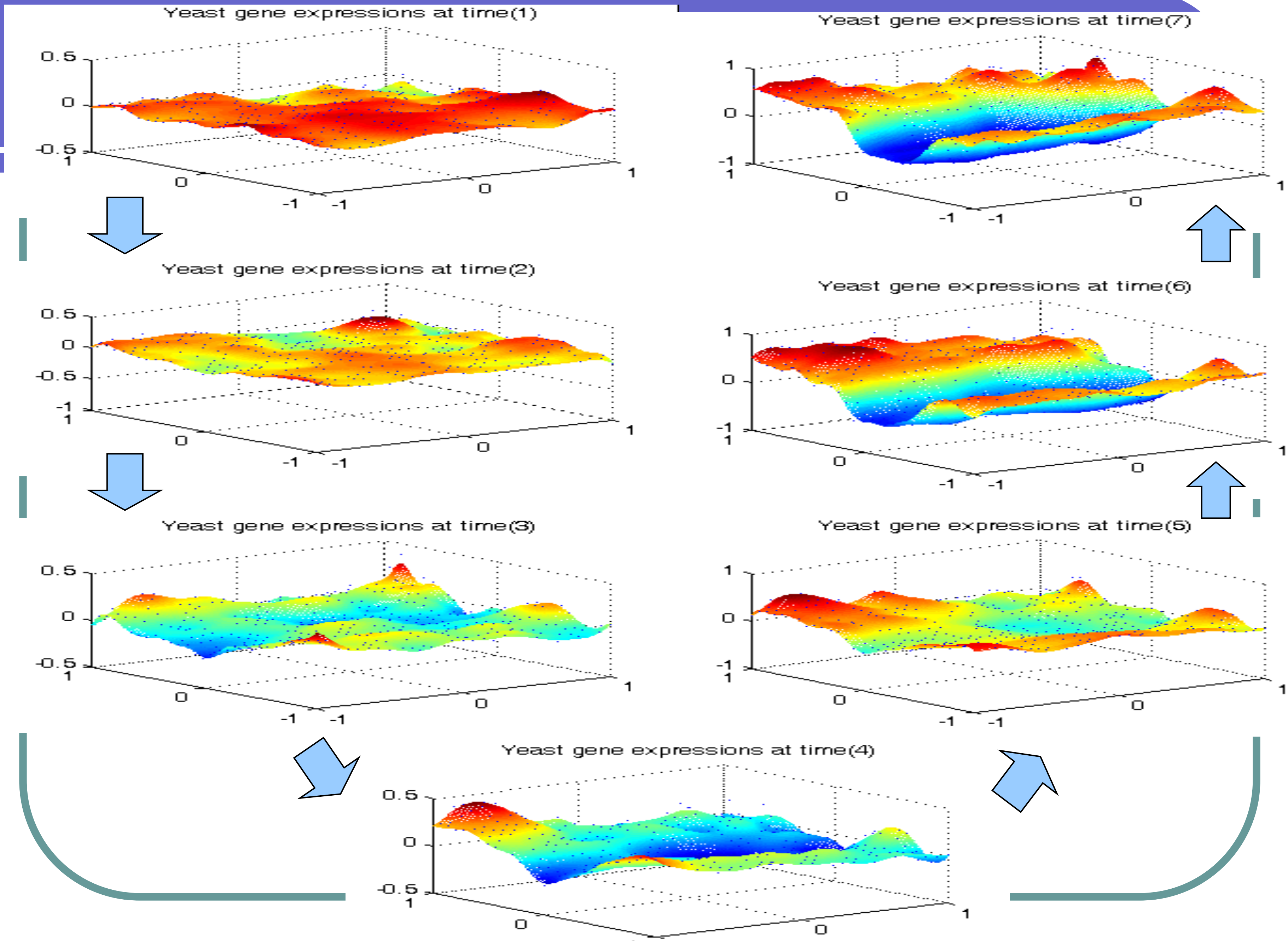

Yeast gene expressions at time(1)
Yeast gene expressions at time(2)
Yeast gene expressions at time(3)
Yeast gene expressions at time(4)
Yeast gene expressions at time(5)
Yeast gene expressions at time(6)
Yeast gene expressions at time(7)

Figure 9

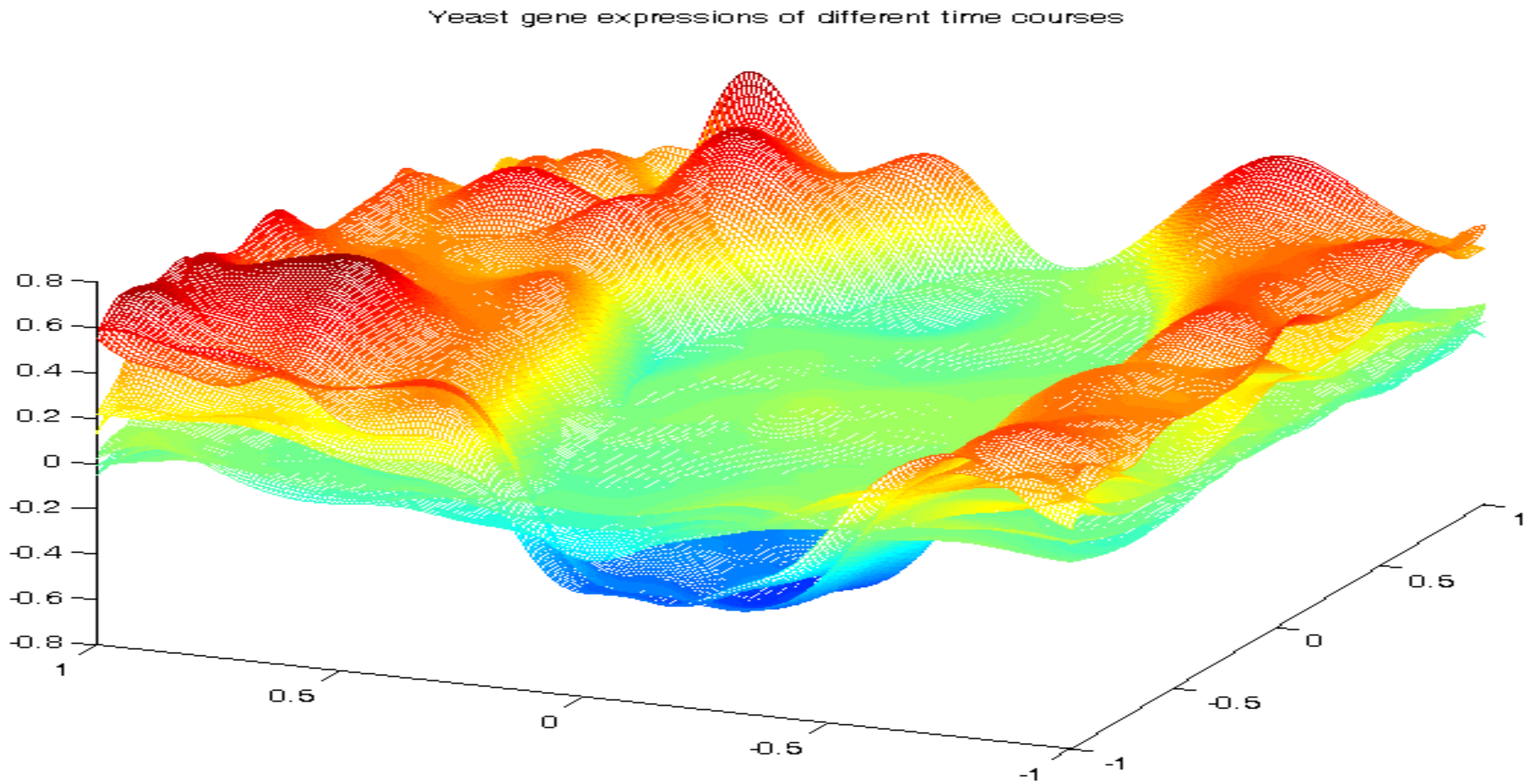

Yeast gene expressions of different time courses
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
1
0.5
0
-0.5
-1
1
0.5
0
-0.5
-1

Figure 10

Recursive function approximation

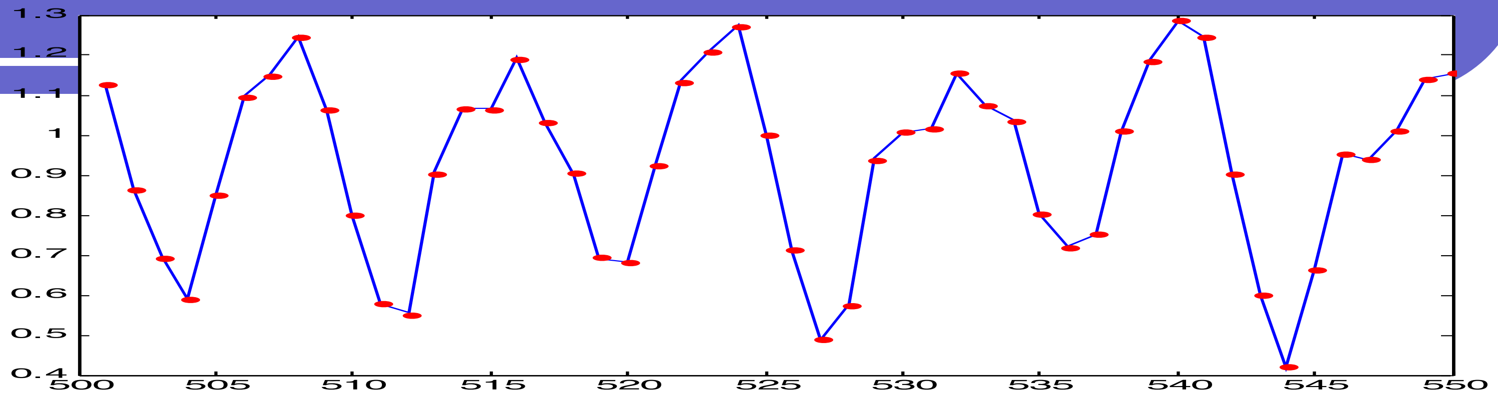
- Time-arrow data analysis
- Learning recurrent MLP networks
- Learning recurrent MLPotts networks

Recursion

- $$z_0, z_1, z_2, \cdots$$

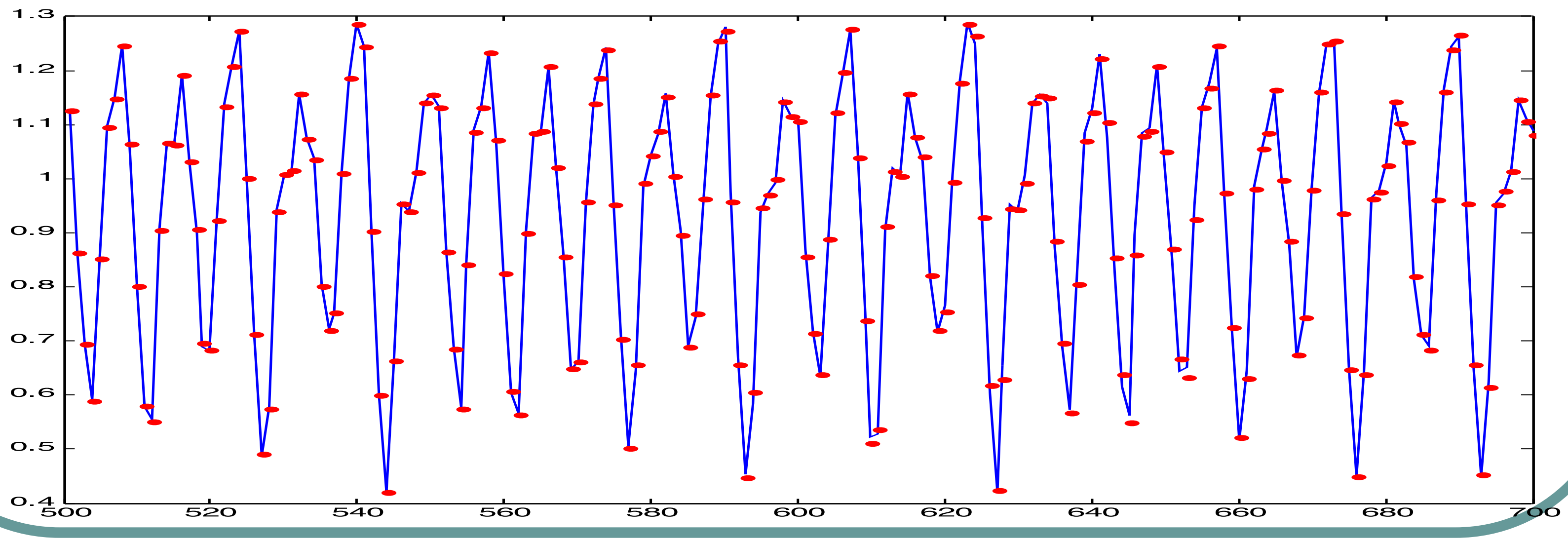
$$z_t = f\left(z_{t-\tau}, z_{t-\tau+1}, \cdots, z_{t-1}\right)$$

50-step-look-ahead long term predictions of Mackey-Glass 17 data



(a)

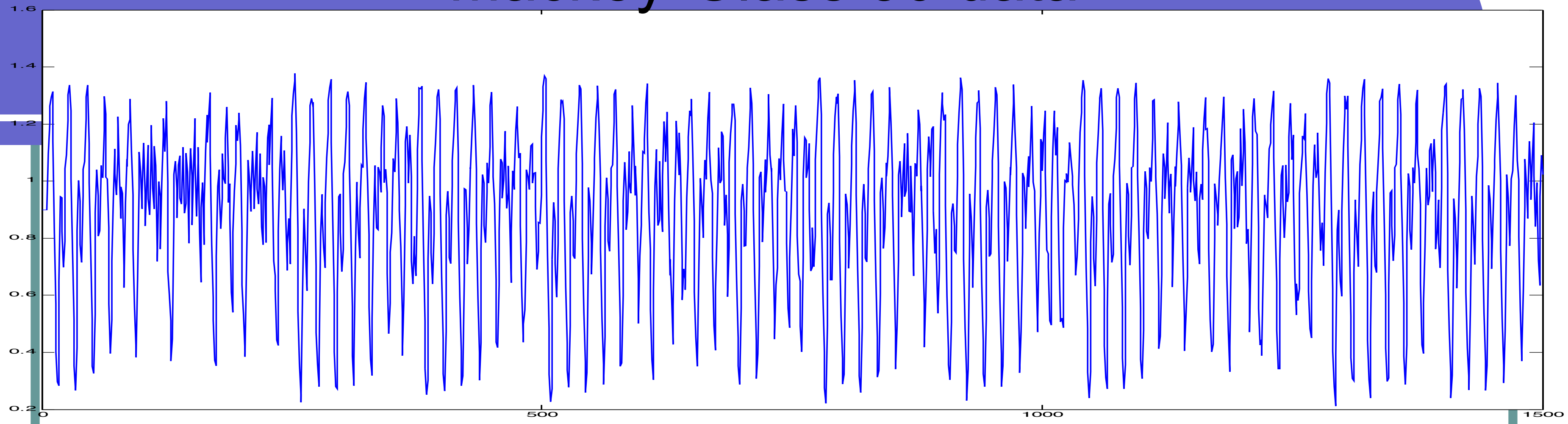
200-step-look-ahead long term predictions of Mackey-Glass 17 data



(b)

Figure 11

Mackey-Glass 30 data



50-step-look-ahead long term predictions of Mackey-Glass 30 data

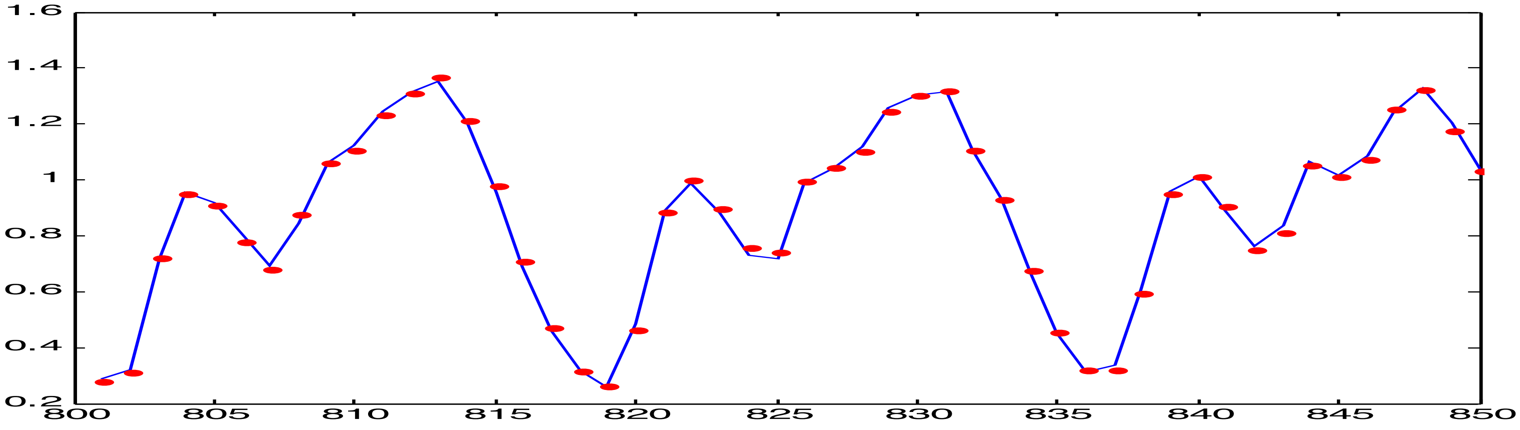
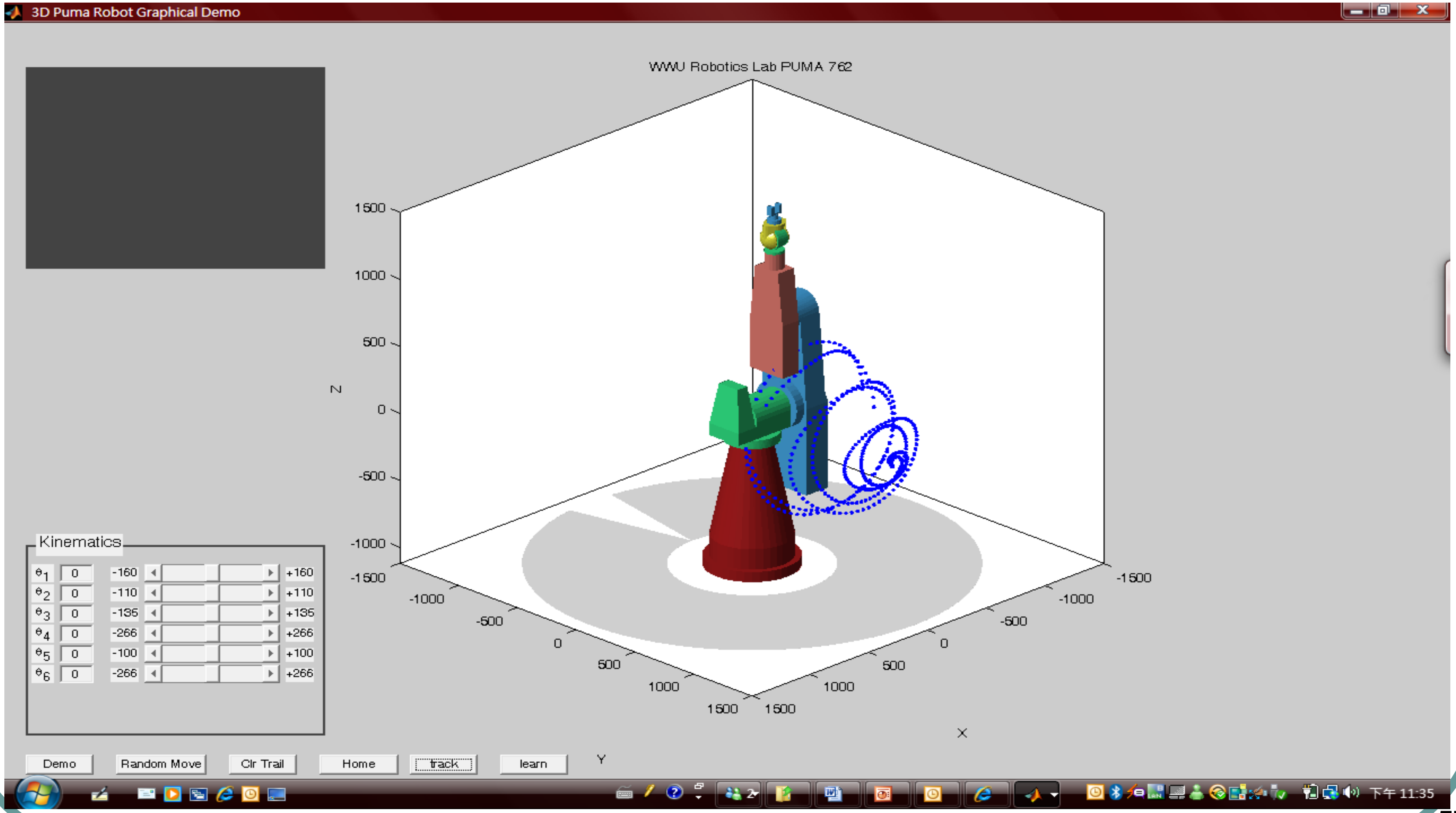
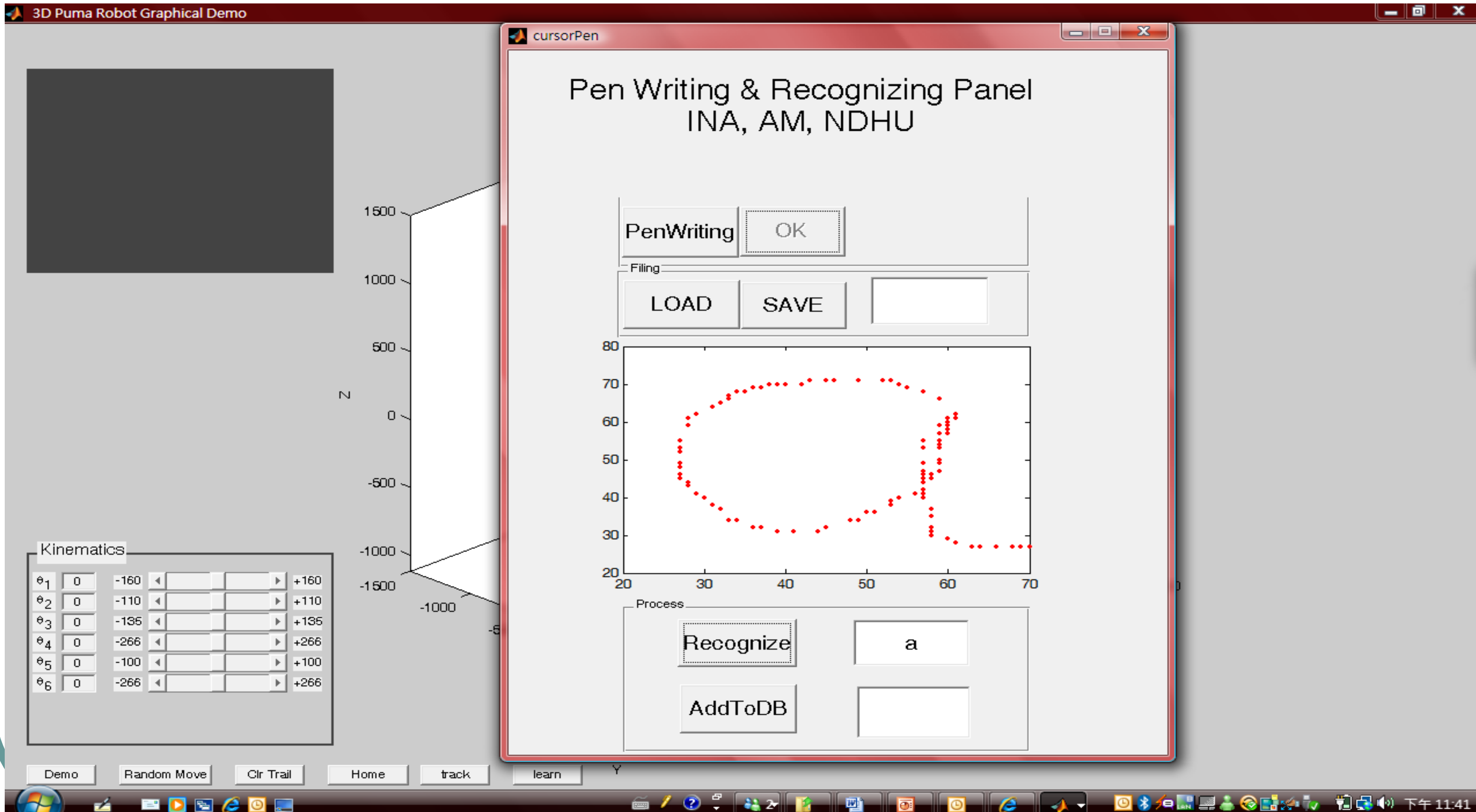


Figure 12

Robot arm control



Pen writing recognition



Advanced Intelligence Computations

- ***Function decomposition***
- ***Covariance matrix analysis (Wu & Lin S.H. 2011)***
- ***Inverse neural systems (Wu & Lin I.L. 2012)***
  - ***One-to-many function approximation***
- ***Automata embedded neural systems (Wu & Chen J.Q. 2012)***
- ***Function integration***
- ***Density support approximation***
- ***Density function approximation***

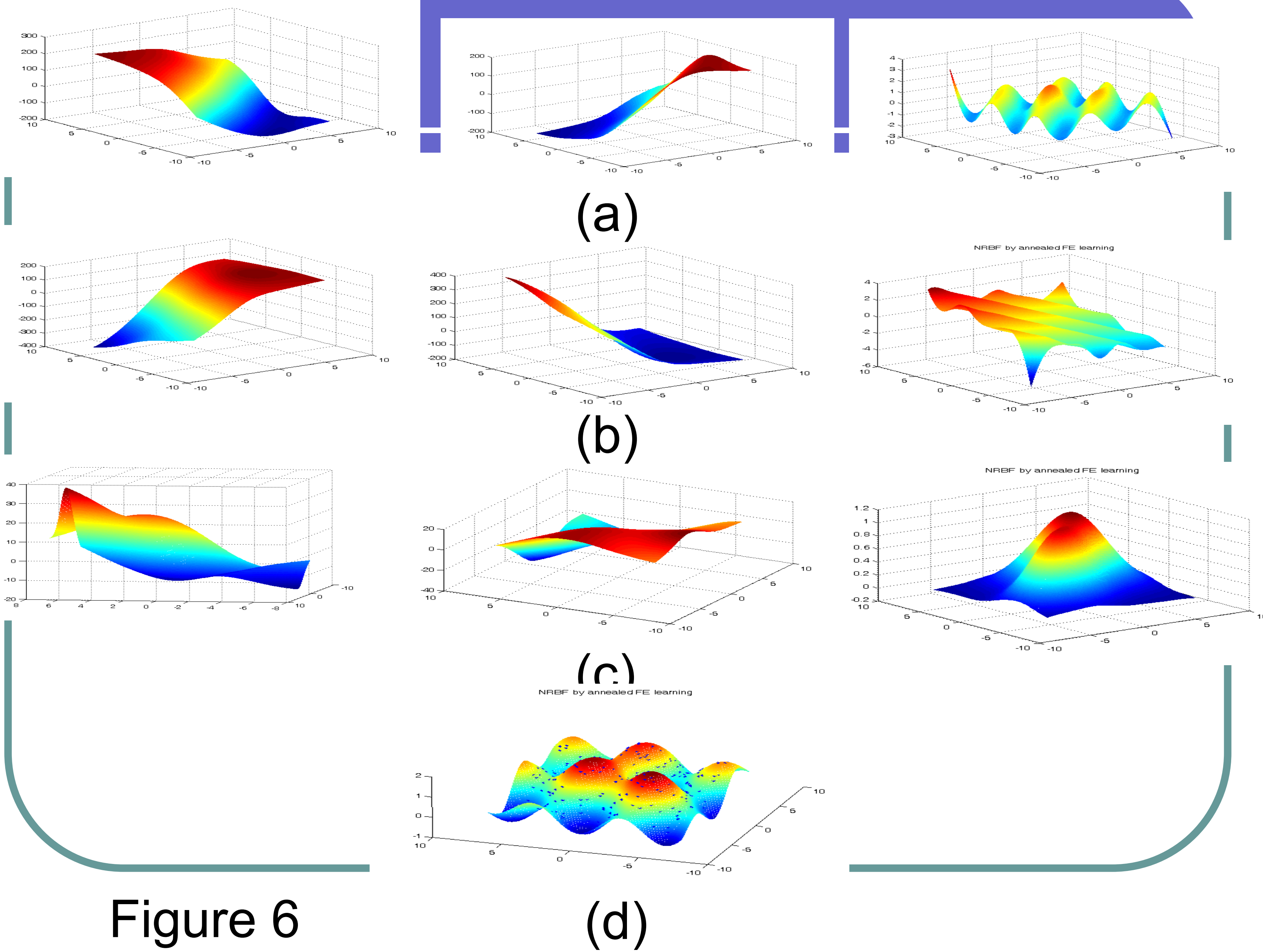
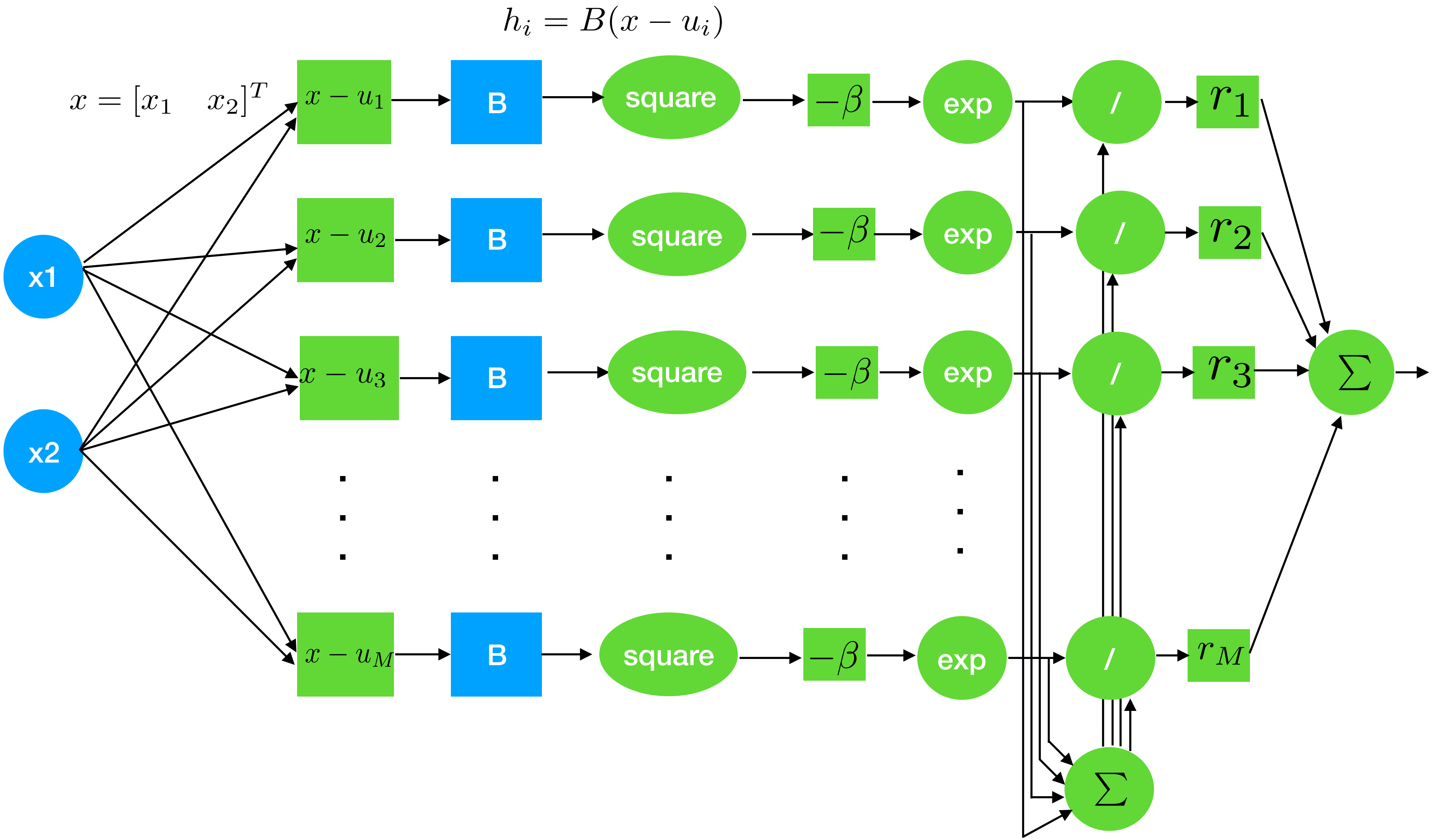
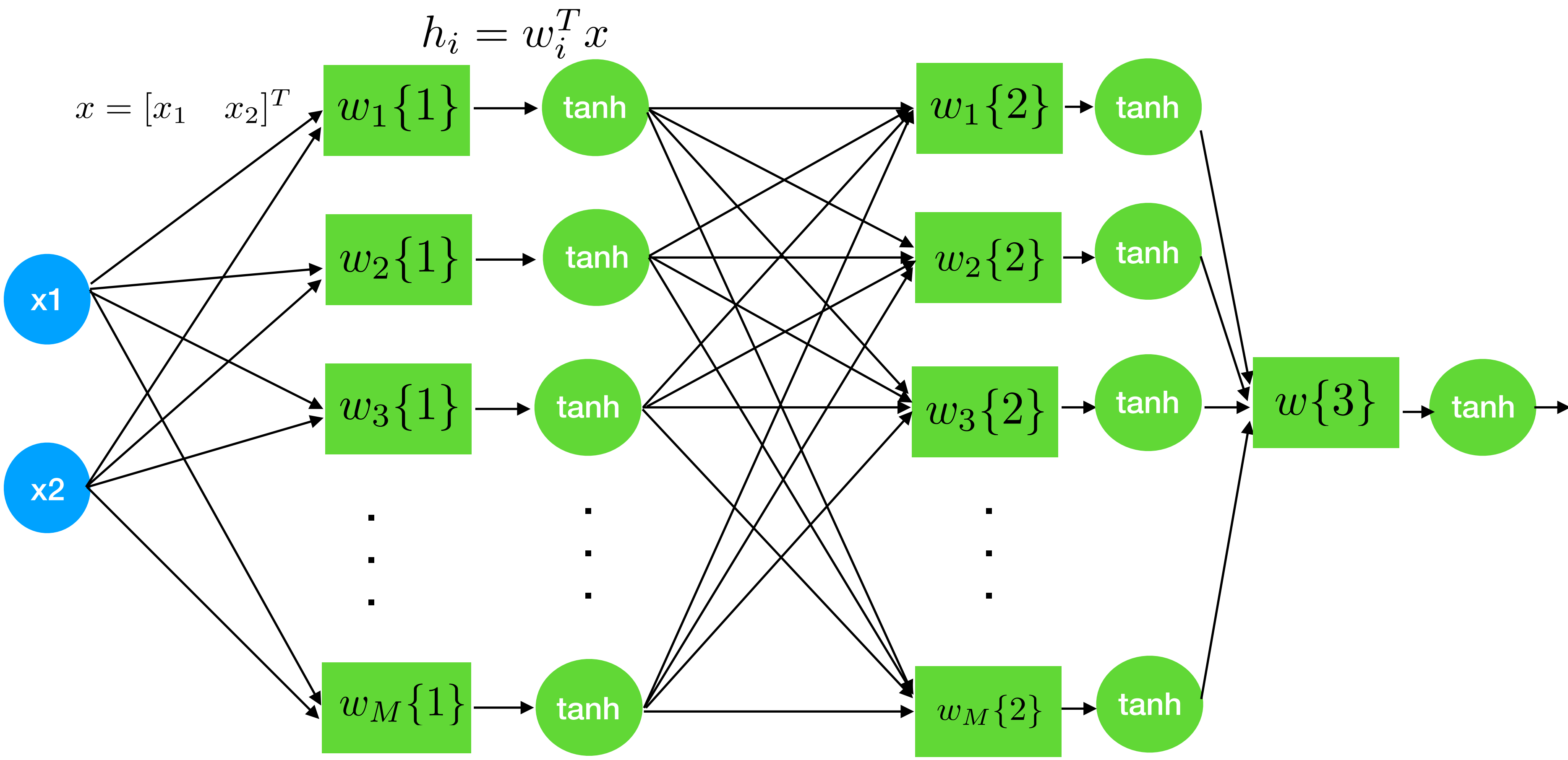

(a)
NRBF by annealed FE learning
(b)
NRBF by annealed FE learning
(c)
NRBF by annealed FE learning
(d)

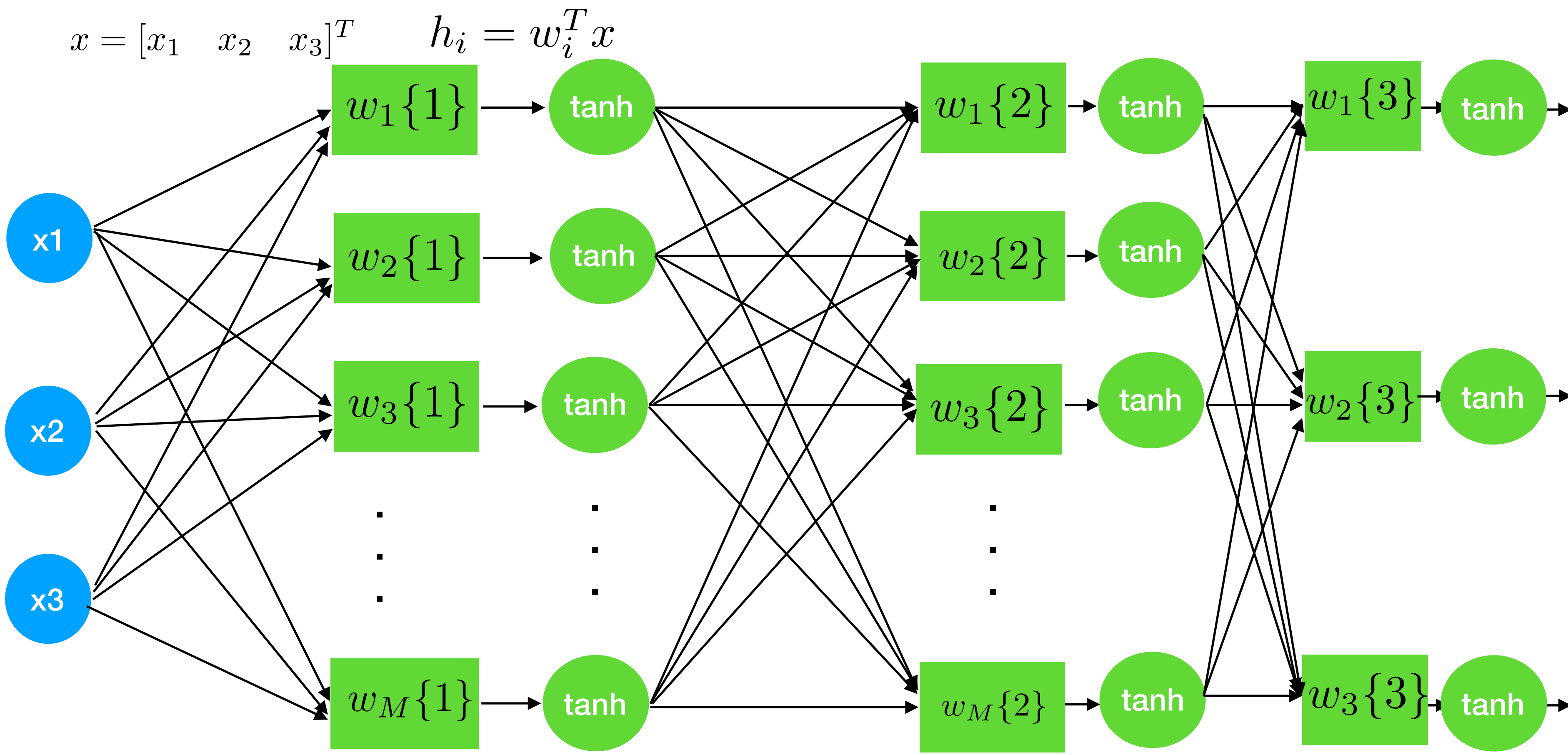
Figure 6



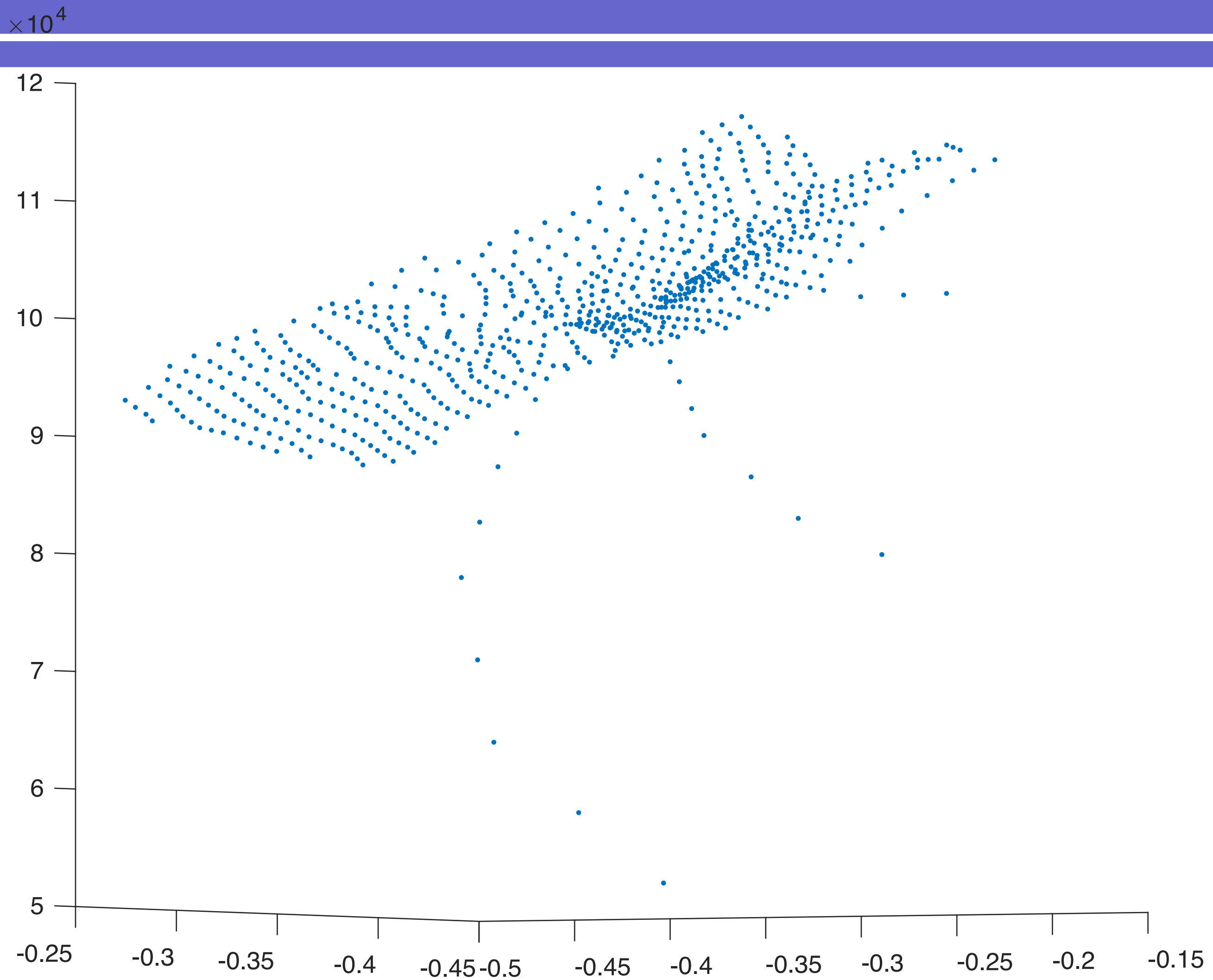
$h_i = B(x - u_i)$
$x = [x_1 \quad x_2]^T$
x1
x2
$x - u_1$
$x - u_2$
$x - u_3$
$x - u_M$
B
B
B
B
square
square
square
square
$-\beta$
$-\beta$
$-\beta$
$-\beta$
exp
exp
exp
exp
/
/
/
/
$\sum$
r_1
r_2
r_3
r_M
$\sum$



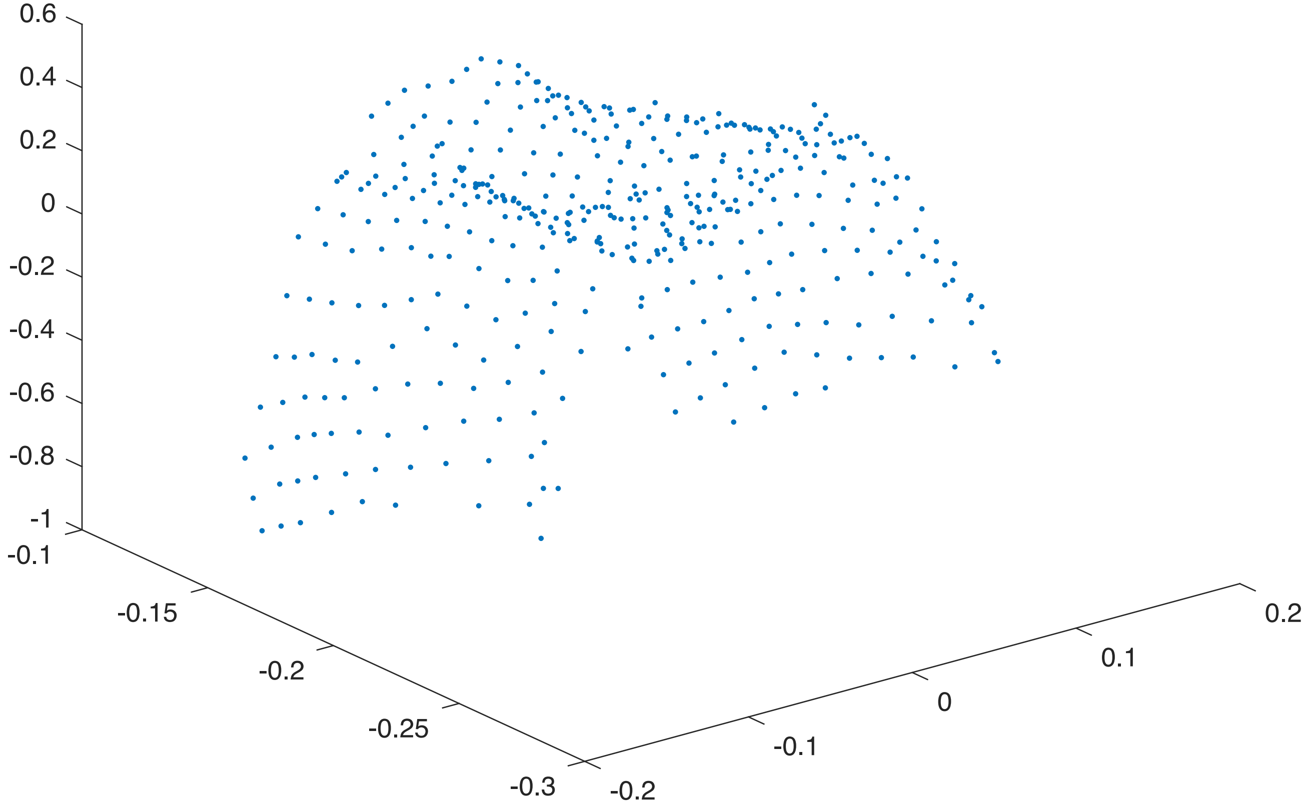
$h_i = w_i^T x$
$x = [x_1 \quad x_2]^T$
x1
x2
$w_1\{1\}$
$w_2\{1\}$
$w_3\{1\}$
$w_M\{1\}$
tanh
$w_1\{2\}$
$w_2\{2\}$
$w_3\{2\}$
$w_M\{2\}$
$w\{3\}$



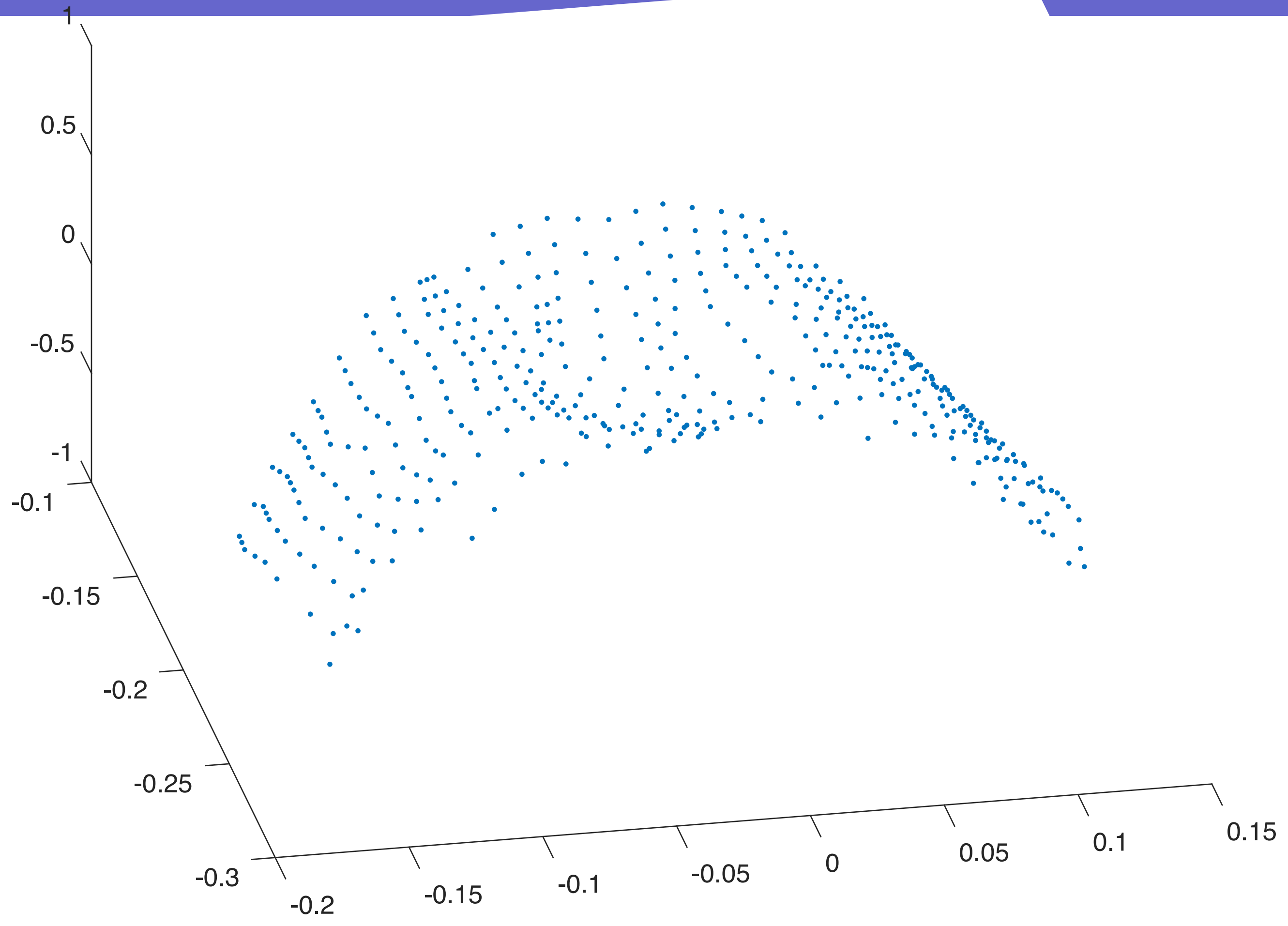
$x = [x_1 \quad x_2 \quad x_3]^T$
$h_i = w_i^T x$
x1
x2
x3
$w_1\{1\}$
$w_2\{1\}$
$w_3\{1\}$
$w_M\{1\}$
tanh
$w_1\{2\}$
$w_2\{2\}$
$w_3\{2\}$
$w_M\{2\}$
$w_1\{3\}$
$w_2\{3\}$
$w_3\{3\}$



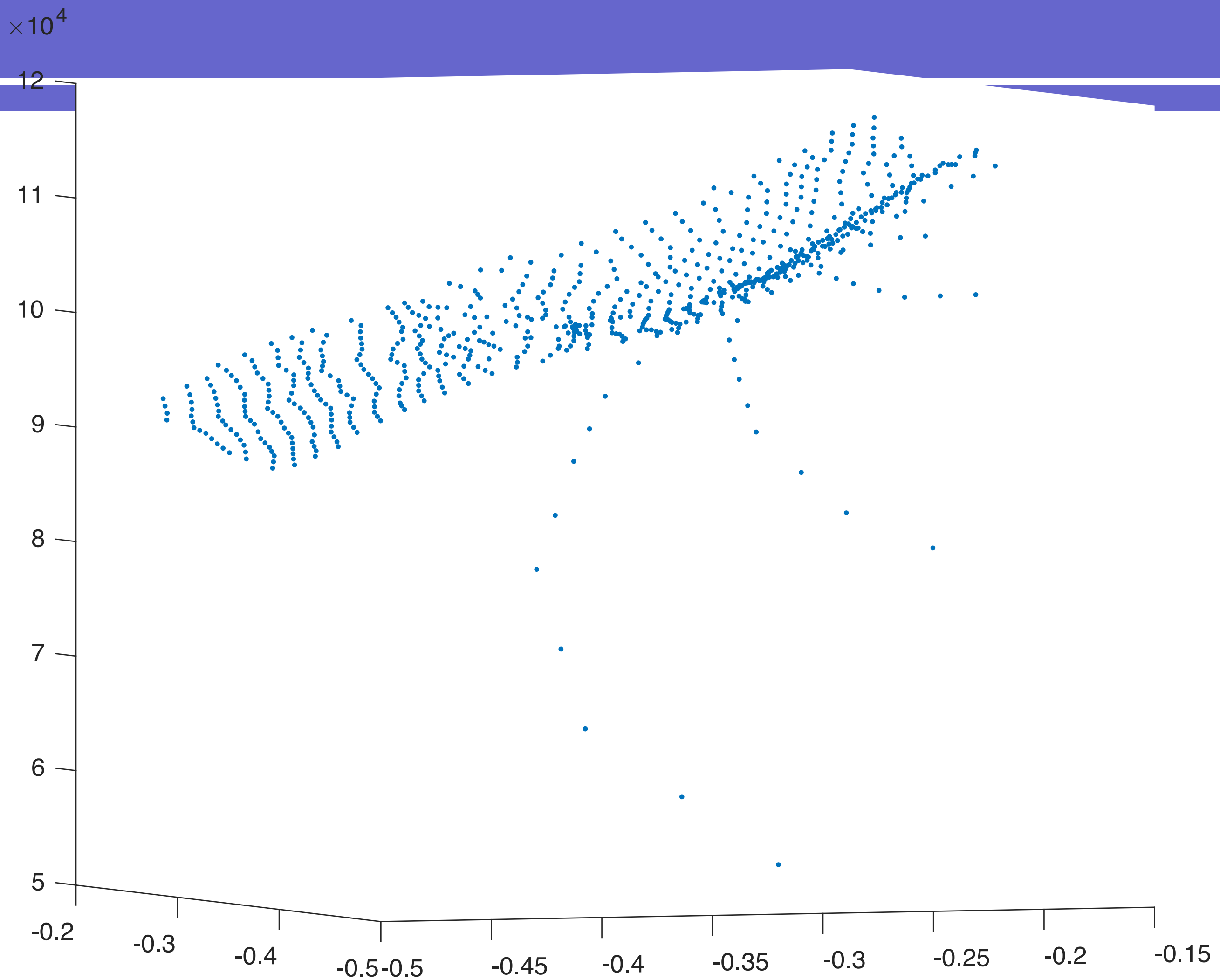
×10^4
12
11
10
9
8
7
6
5
-0.25
-0.3
-0.35
-0.4
-0.45
-0.5
-0.45
-0.4
-0.35
-0.3
-0.25
-0.2
-0.15

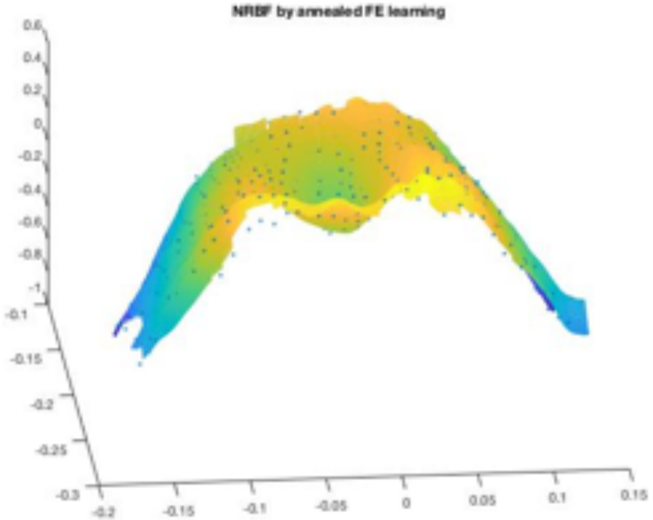


0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.1
-0.15
-0.2
-0.25
-0.3
-0.2
-0.1
0
0.1
0.2

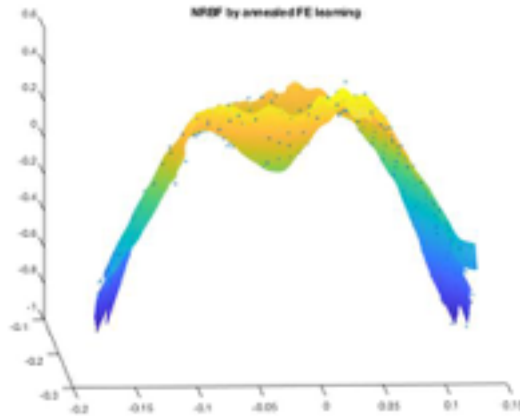


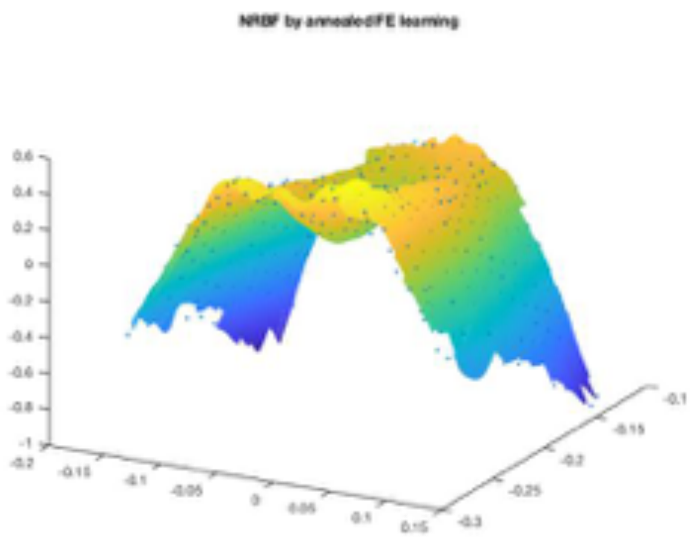
1
0.5
0
-0.5
-1
-0.1
-0.15
-0.2
-0.25
-0.3
-0.2
-0.15
-0.1
-0.05
0
0.05
0.1
0.15

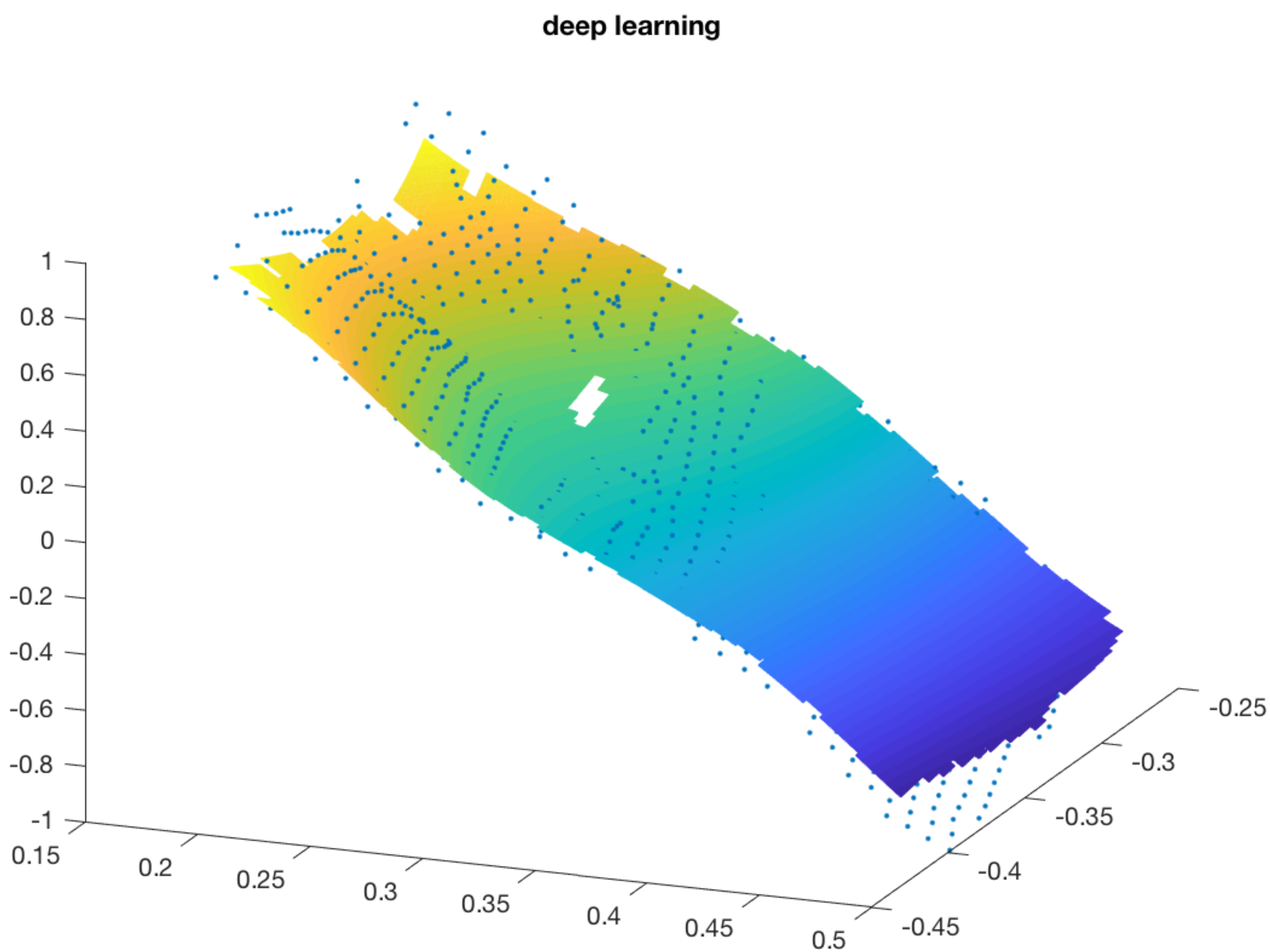

×10^4
12
11
10
9
8
7
6
5
-0.2
-0.3
-0.4
-0.5
-0.5
-0.45
-0.4
-0.35
-0.3
-0.25
-0.2
-0.15

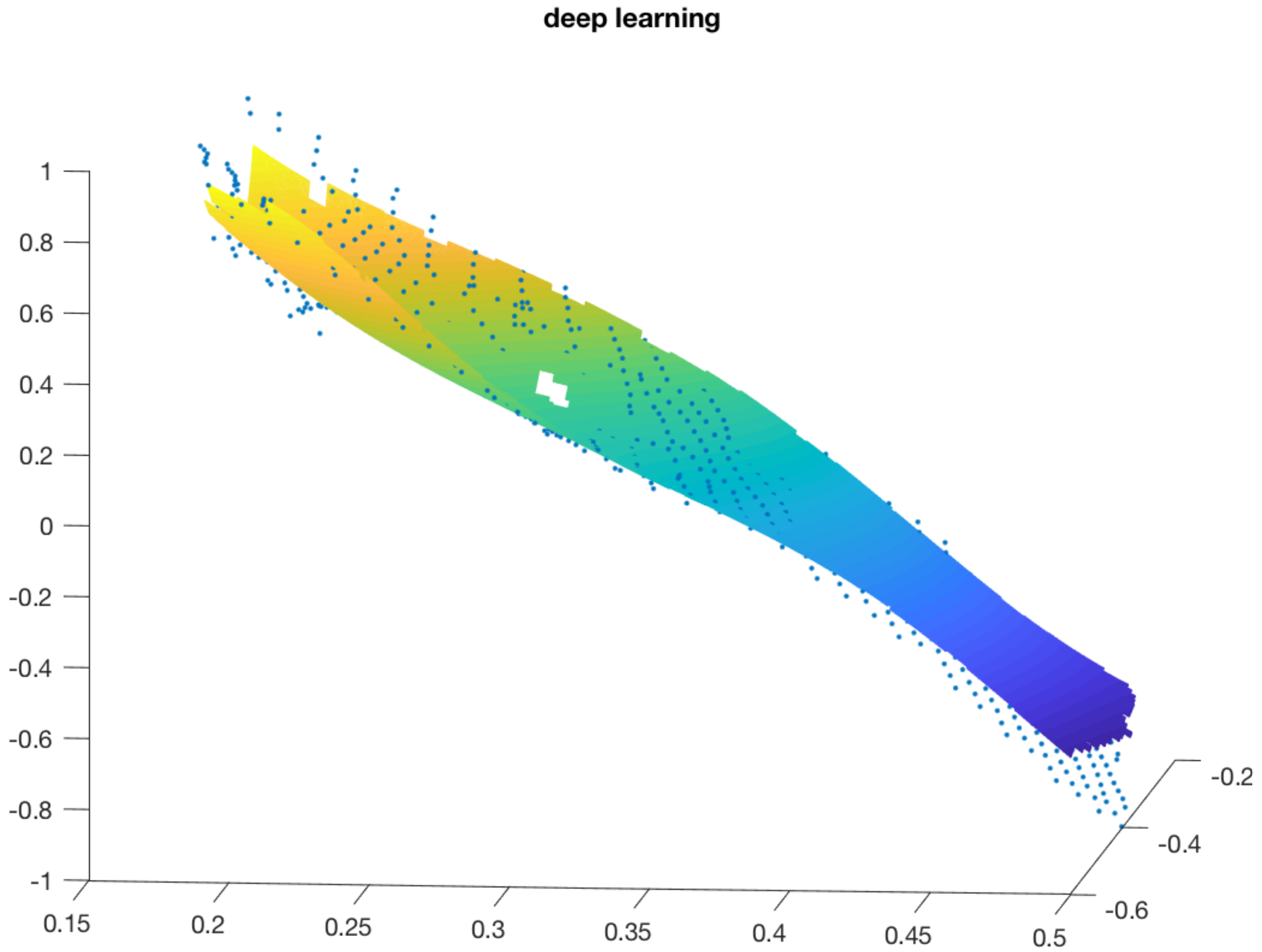


NRBF by annealed FE learning

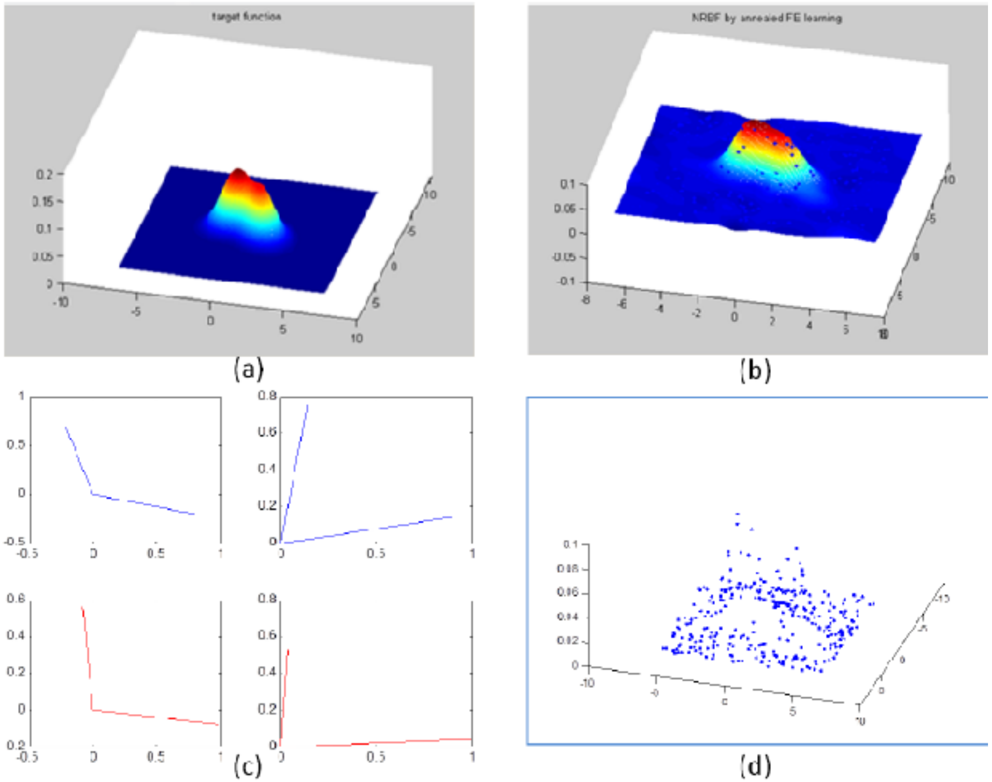

NRBF by annealed FE learning
-0.2
-0.15
-0.1
-0.05
0
0.05
0.1
0.15


NRBF by annealedFE learning
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.2
-0.15
-0.1
-0.05
0
0.05
0.1
0.15
-0.3
-0.25
-0.2
-0.15
-0.1


deep learning
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
-0.25
-0.3
-0.35
-0.4
-0.45

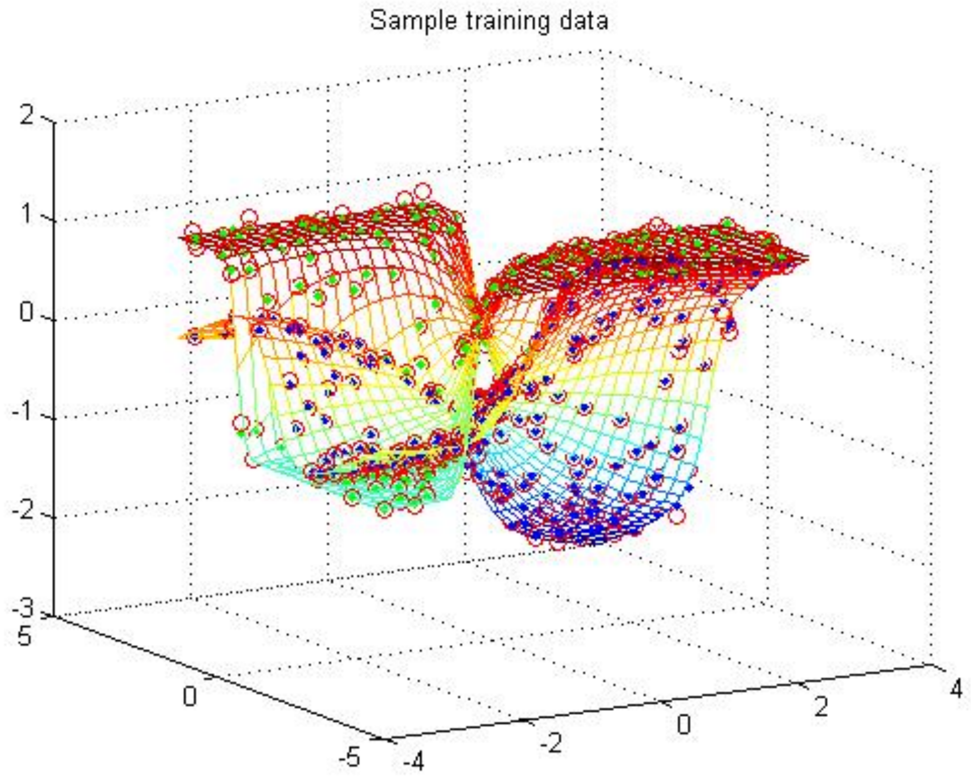

deep learning
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
-0.2
-0.4
-0.6

Covariance Matrix Analysis

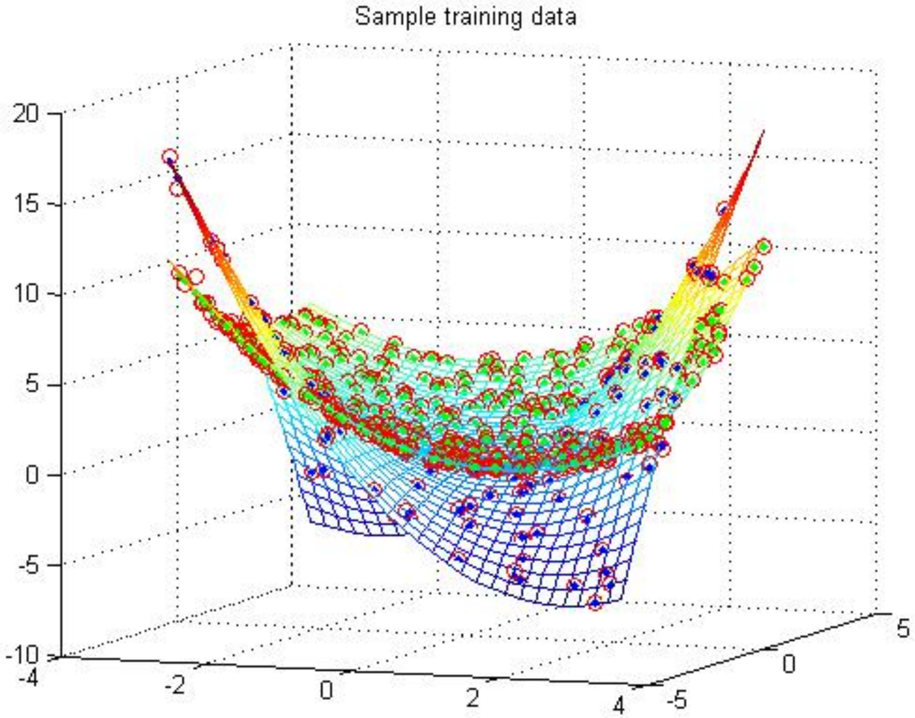


Function approximation v.s One-to-many mapping

	function approximation	one-to-many mapping
type	an RBF network	multiple high-order RBF networks
approximation	one-to-one mapping	one-to-many mapping
order	single-order posterior interconnections	high-order posterior interconnections
learning	supervised learning	supervised learning
data type	paired data without weights	paired data with weights
control system	forward kinematics	inverse kinematics
modular type	single module	multiple modules
objective function	mean square error	weighted square error



- P=2,k=25
- 2 tanh function
- mse=0.0022



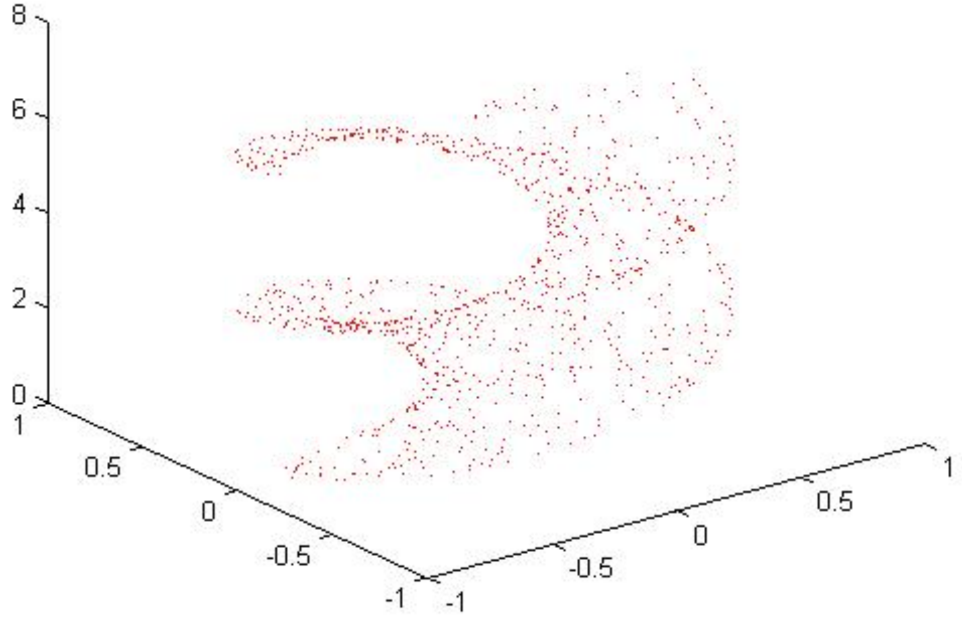
- P=2,k=25
- 2quadratic function
- Mse=0.0089

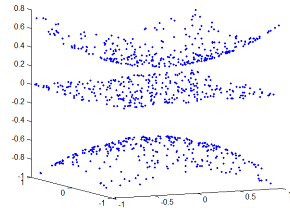
One-to-many function approximation to g_1

- P=5;k=M=30;N=1000

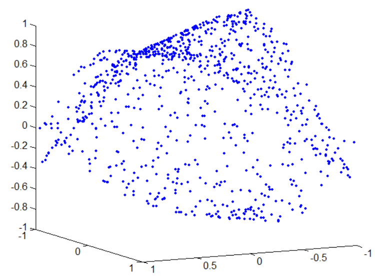
$$a_1 = g_1(q_1, q_2)$$

$$a_1 \in (0, 2\pi)$$

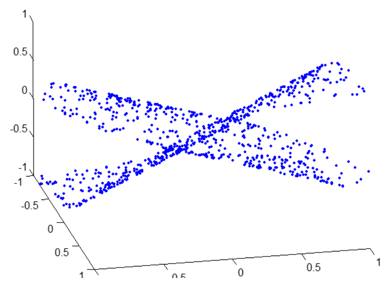




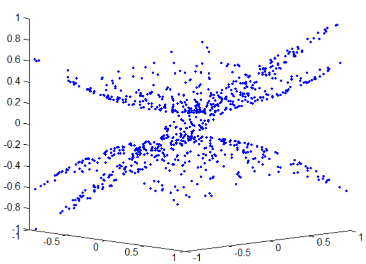
$$f_1(x_1,x_2) = x_1^2 + x_2^2 + 1$$
$$f_2(x_1,x_2) = -x_1^2 - x_2^2 - 2$$
$$f_3(x_1,x_2) = 0.1x_1 + 0.1x_2$$



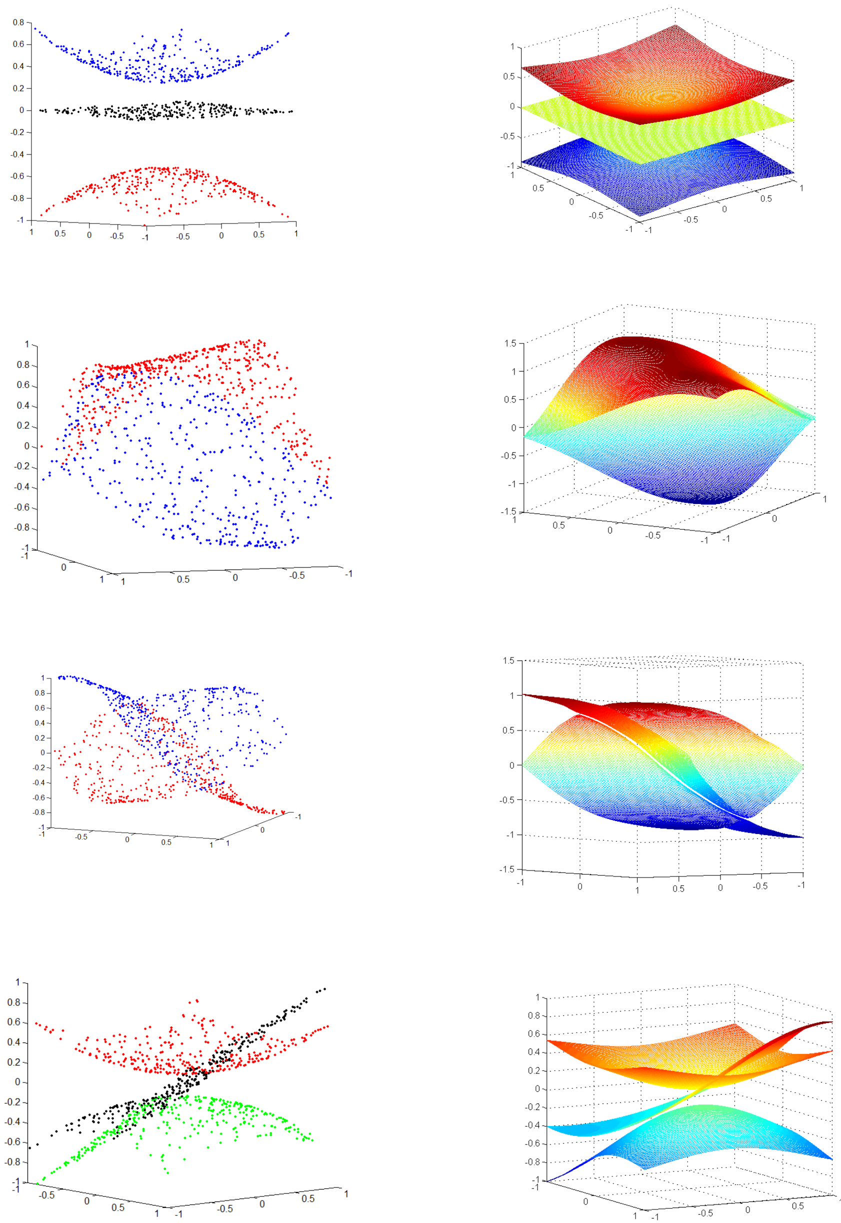
$$f_1(x_1,x_2) = \sin(x_1 + x_2)$$
$$f_2(x_1,x_2) = \cos(x_1 - x_2)$$



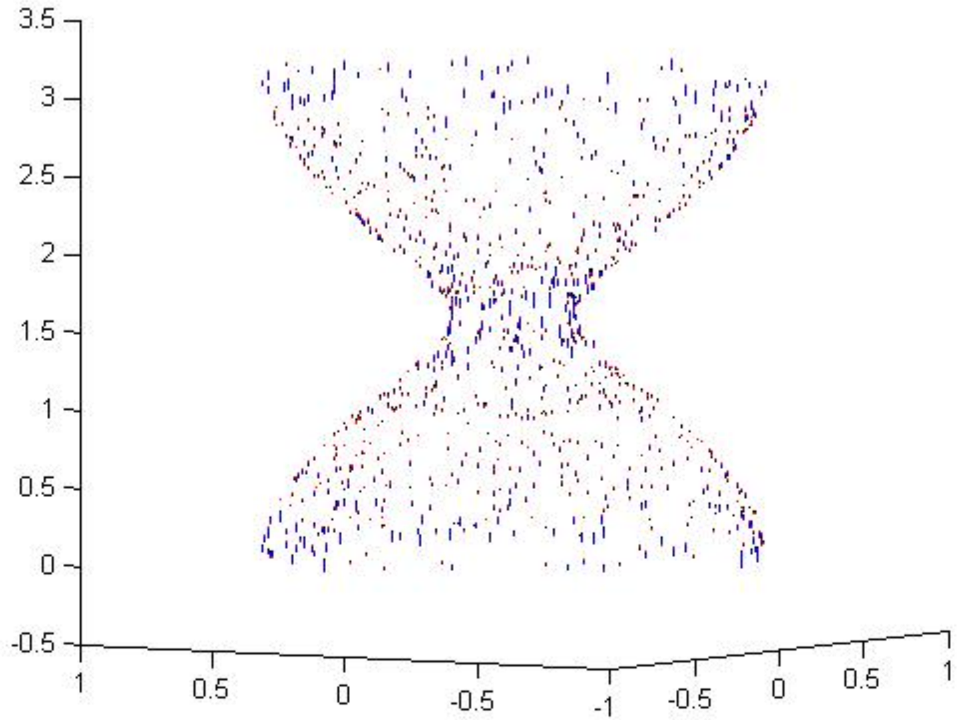
$$f_1(x_1,x_2) = \tanh(x_1 + x_2)$$
$$f_2(x_1,x_2) = \tanh(x_1 - x_2)$$

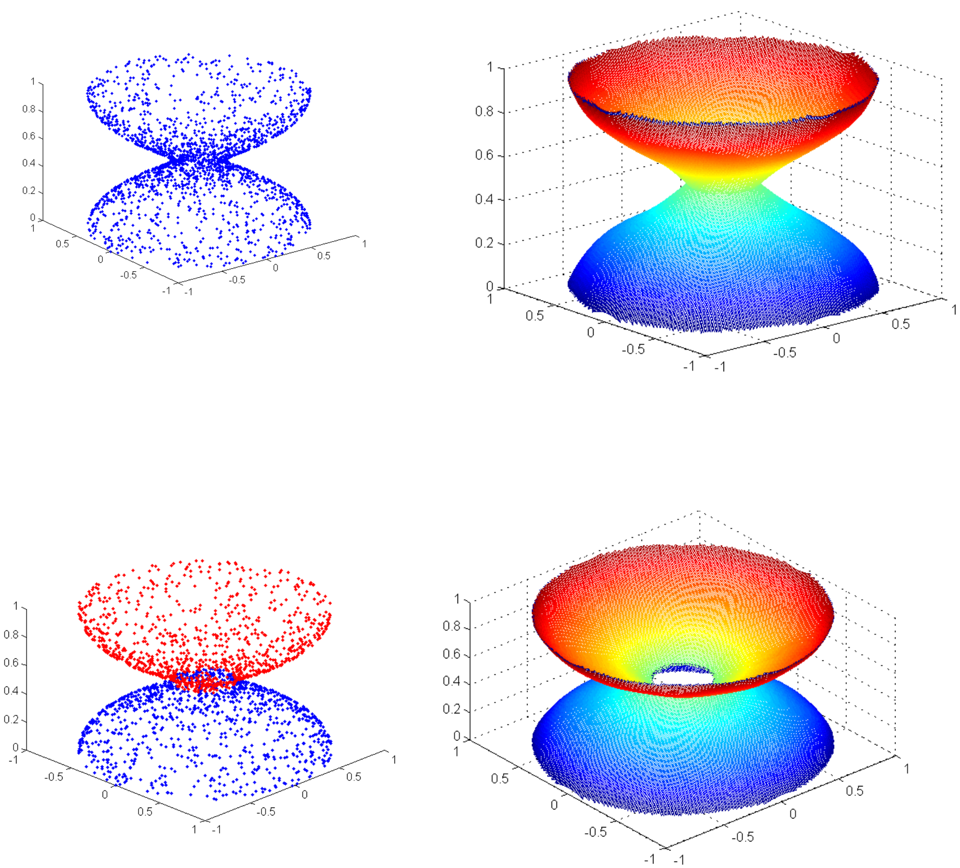


$$f_1(x_1,x_2) = x_1^2 + x_2^2 + 0.5$$
$$f_2(x_1,x_2) = -x_1^2 - x_2^2 - 0.5$$
$$f_3(x_1,x_2) = 2x_1 + 2x_2$$



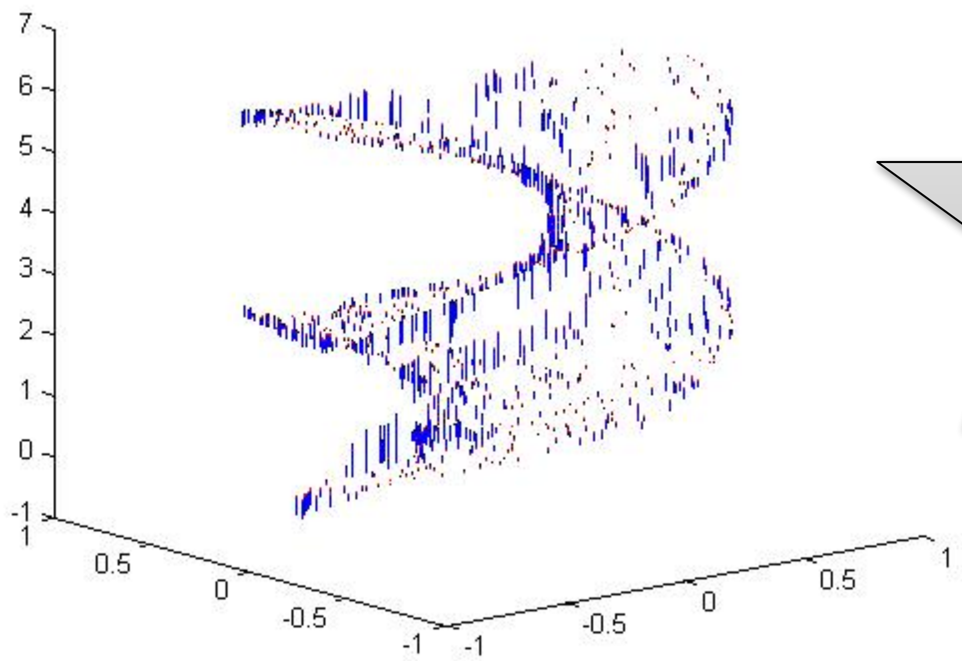
Approximating functions

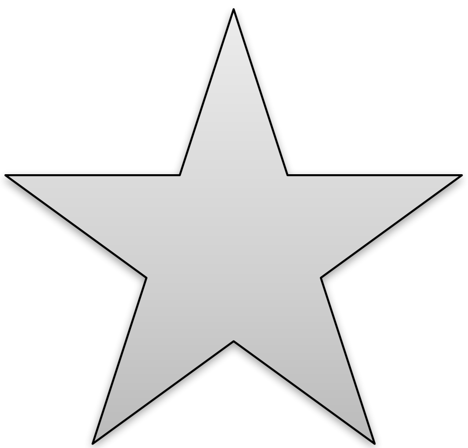


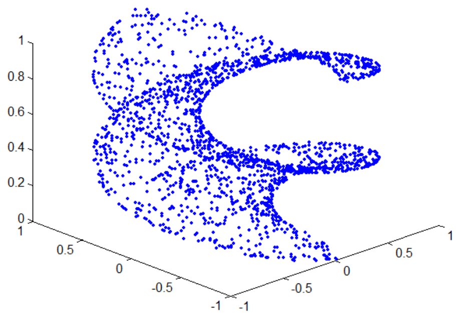


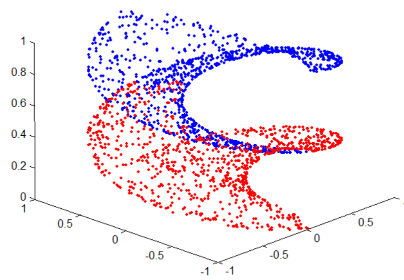
1
0.8
0.6
0.4
0.2
0
1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1
1
0.8
0.6
0.4
0.2
0
1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1
1
0.8
0.6
0.4
0.2
0
-1
-0.5
0
0.5
1
-1
-0.5
0
0.5
1
1
0.8
0.6
0.4
0.2
0
1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1

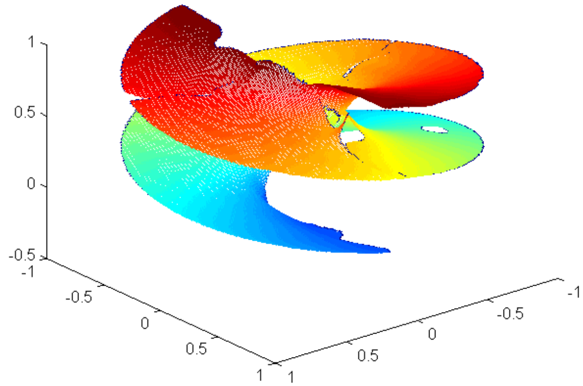
Approximating functions

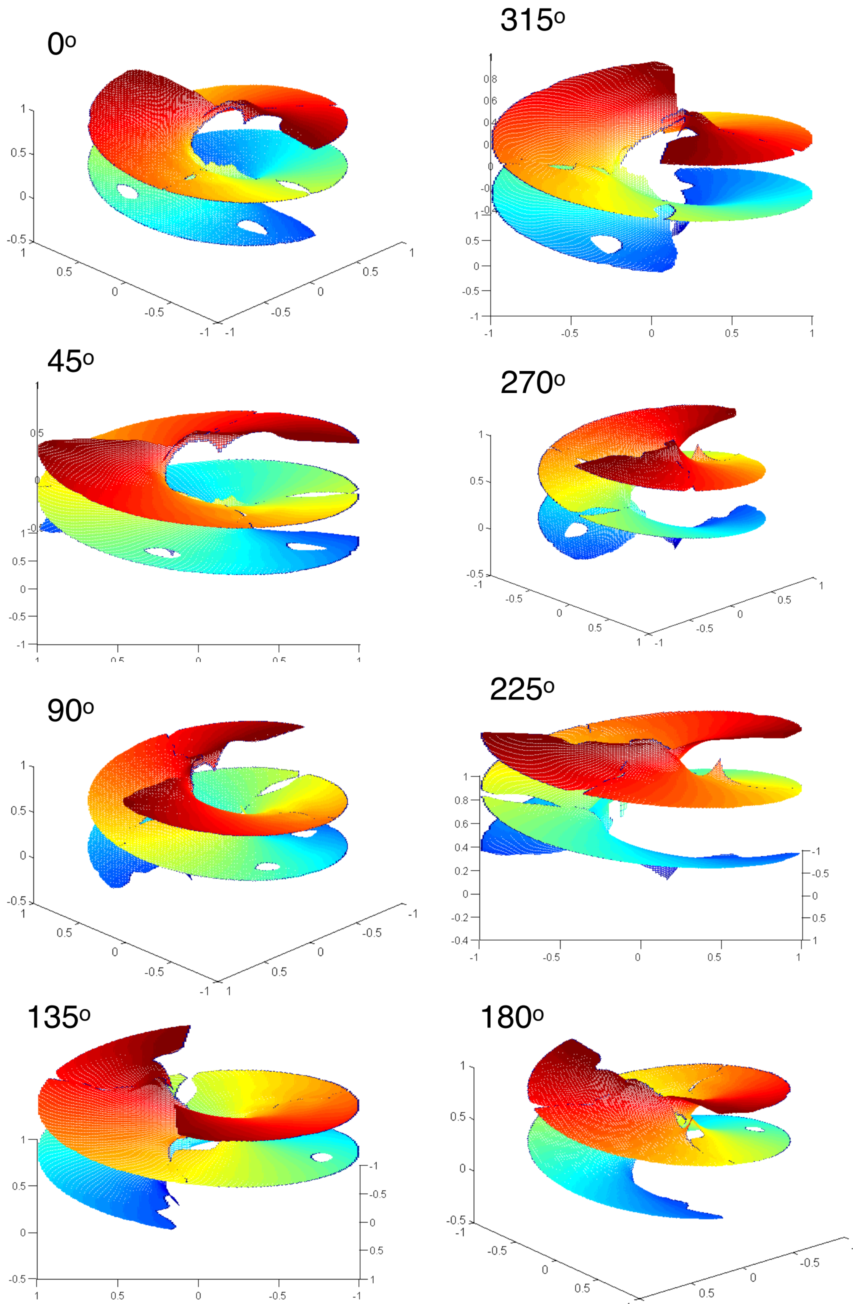




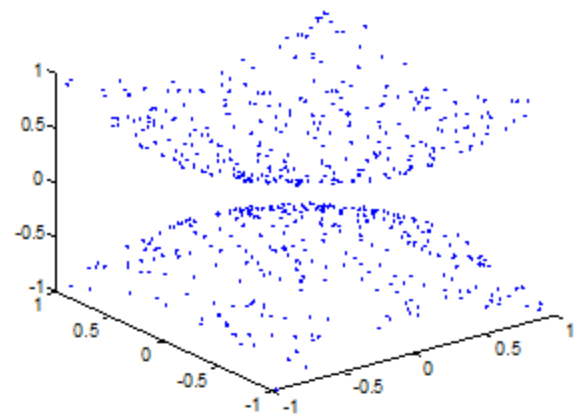
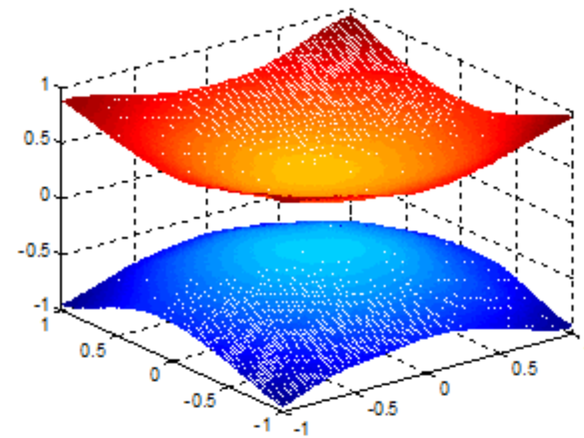

1
0.8
0.6
0.4
0.2
0
1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1


1
0.8
0.6
0.4
0.2
0
1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1


1
0.5
0
-0.5
-1
-0.5
0
0.5
1
1
0.5
0
-0.5
-1

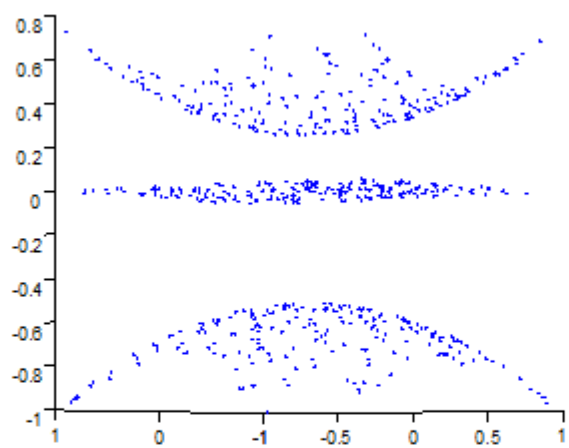
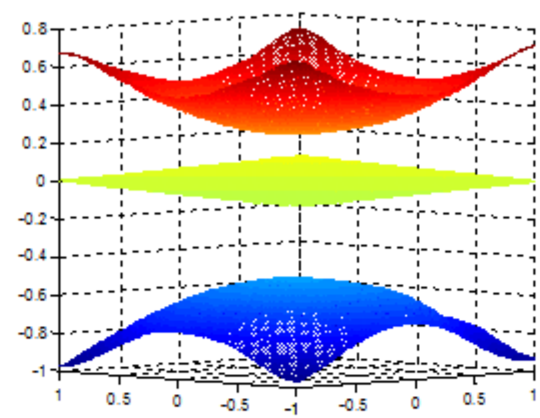

0o
315o
45o
270o
90o
225o
135o
180o

2-type deterministic transition

Original data	deterministic rule
	

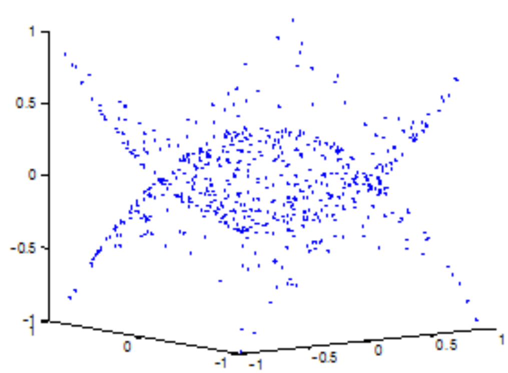
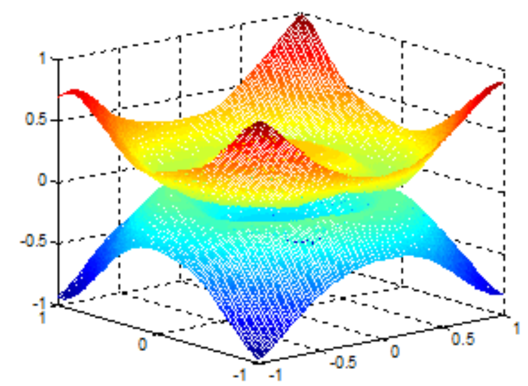
	target prediction (learning)	state inference	target prediction (aenn)
error	0.000257	0.000758	0.000281

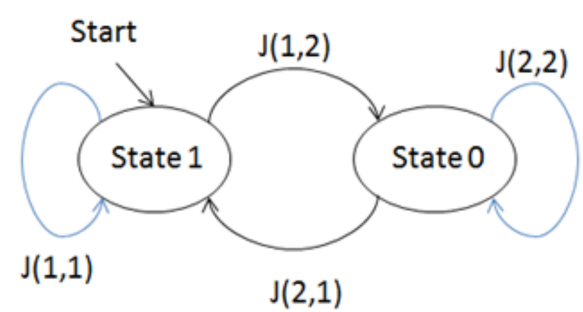
3-type deterministic transition

Original data	deterministic rule
	

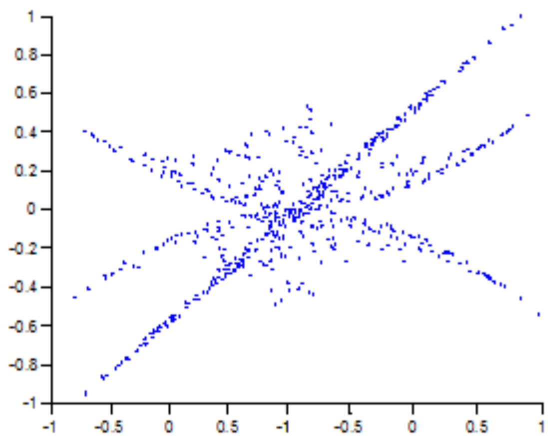
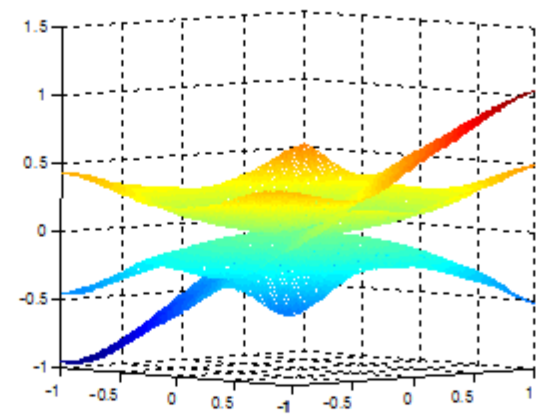
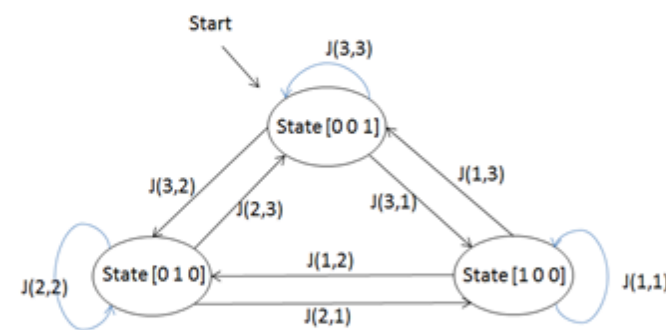
	target prediction (learning)	state inference	target prediction (aenn)
error	0.000211	0.000911	0.000230

2-type Stochastic transition

<table>
<tr><td>Original data</td><td>Stochastic rule</td></tr>
<tr><td></td><td></td></tr>
<tr><td>Probability</td><td>Stochastic automata state transition</td></tr>
<tr><td>J =

0.4045 0.5955
0.6345 0.3655</td><td>
</td></tr>
</table>

3-type Stochastic transition

Original data	Stochastic rule
	
Probability	**Stochastic automata state transition**
J = $\begin{matrix} 0.1181 & 0.7014 & 0.1806 \\ 0.2925 & 0.7075 & 0 \\ 0.0645 & 0.7742 & 0.1613 \end{matrix}$	

Differential function approximation

- Neural Networks

$$v(t) = \frac{dx}{dt}$$

$$= F(v(t), x(t))$$

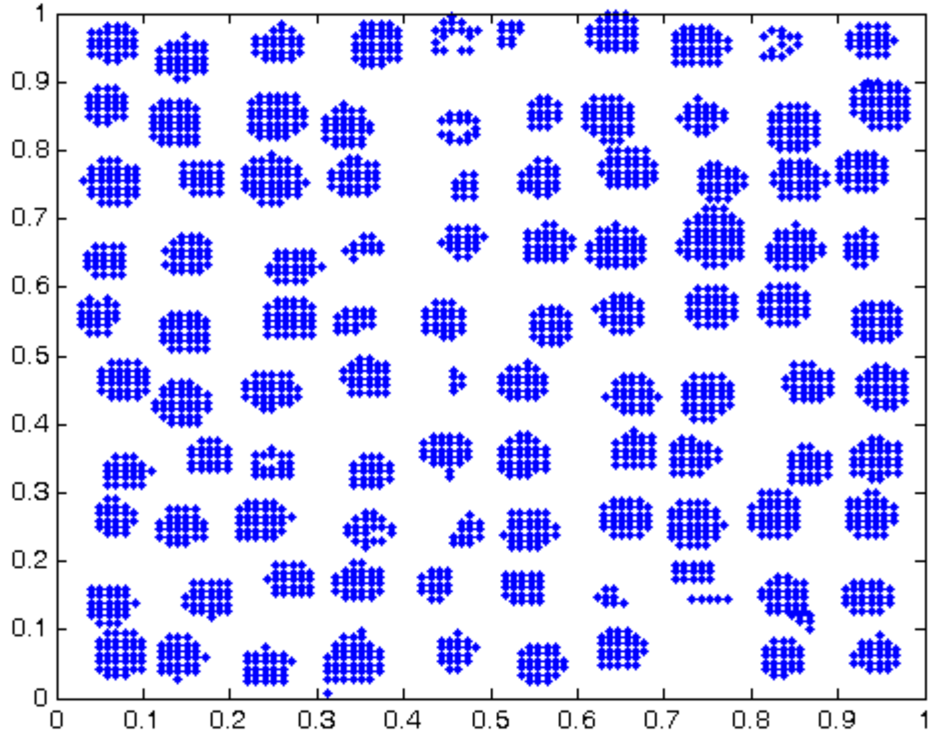
Learning neural networks to approximate differential equations

Unsupervised Data Analysis

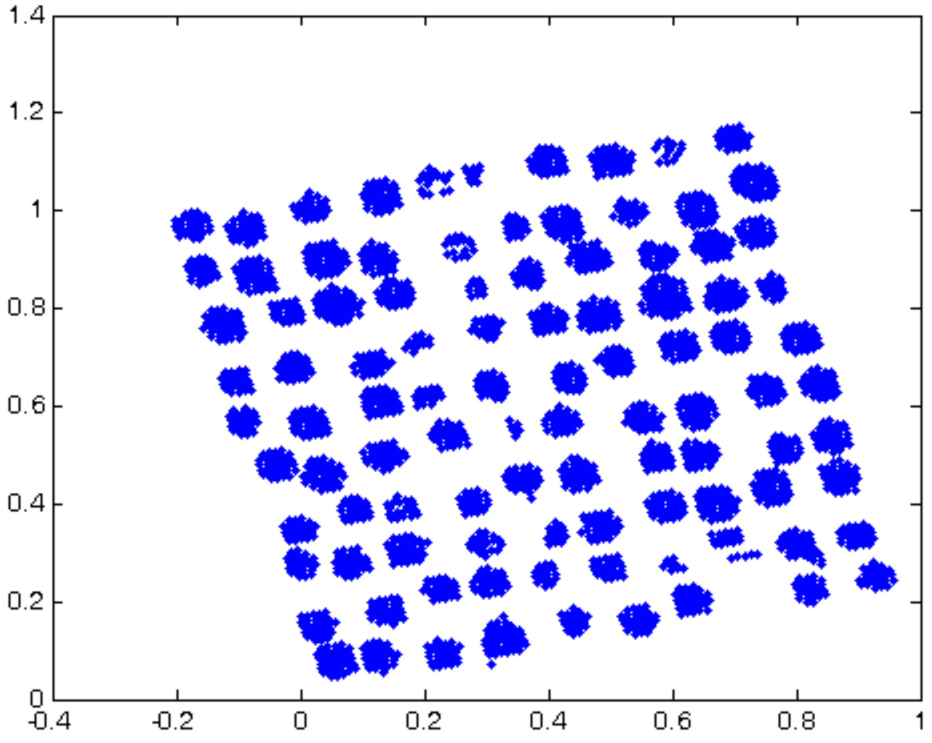
- No time-arrow
- Unsupervised data analysis
  - Topological and statistical Analysis
    - 2D sorting
    - Data clustering
  - Density function approximation
  - Reconstruction of generative models

Clustering analysis

Find data clusters

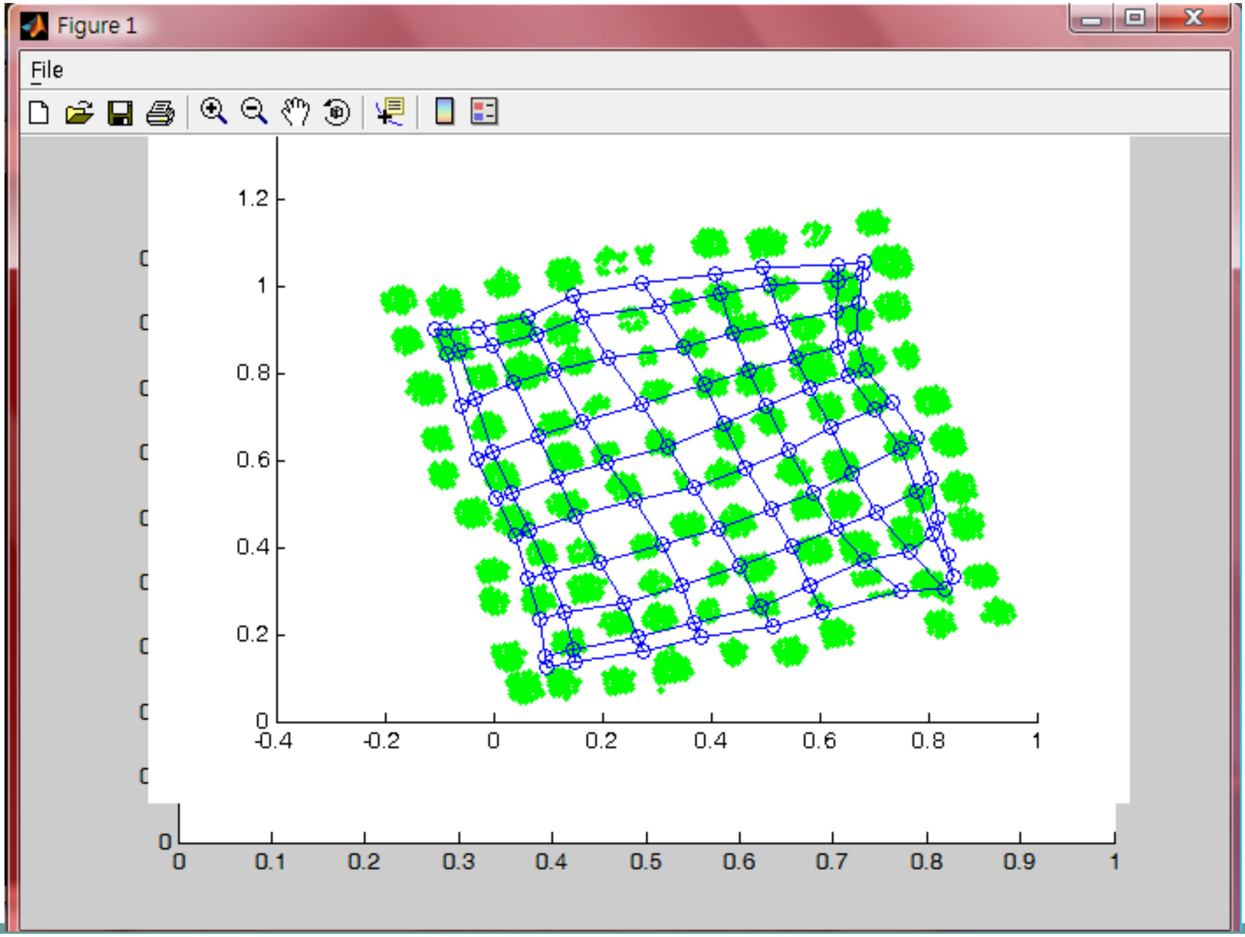


Rotated distributed clusters

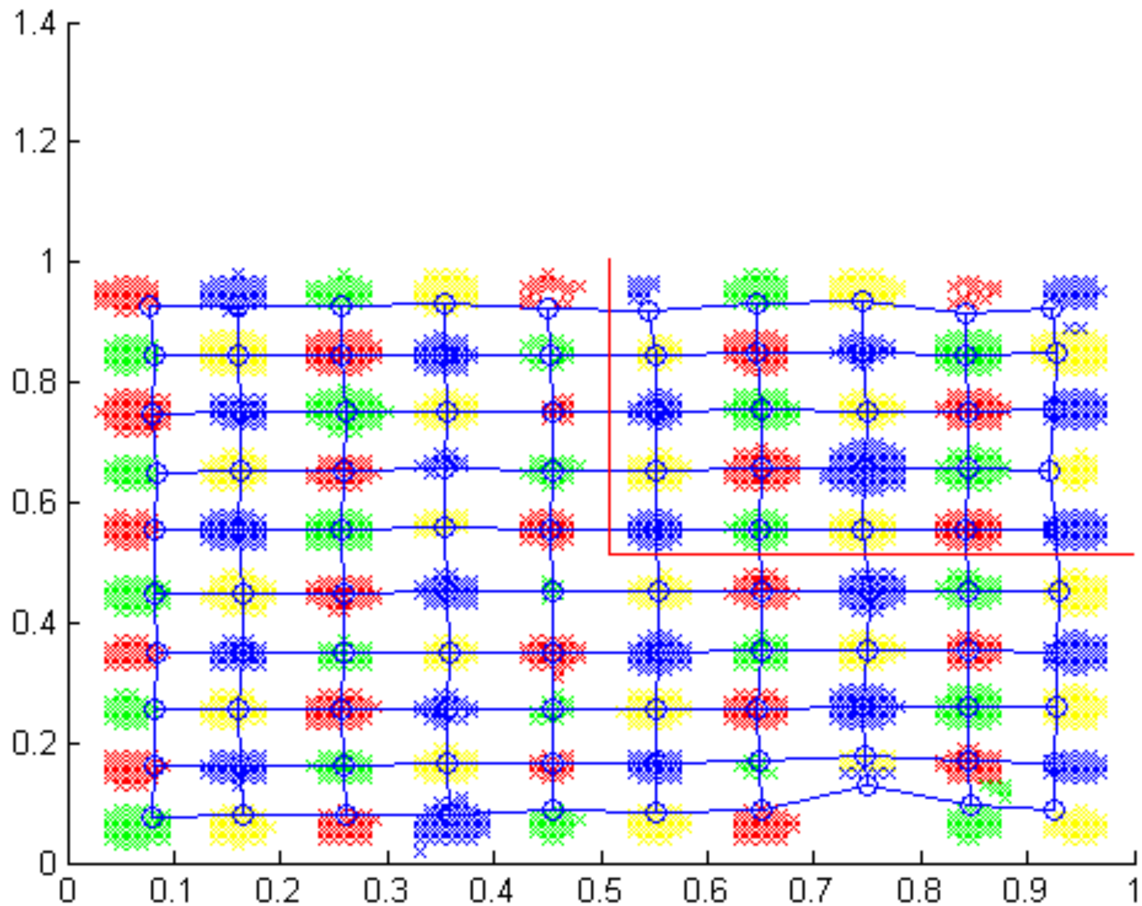


Deformable gridding

Place a lattice to structure distributed clusters



Self-organization



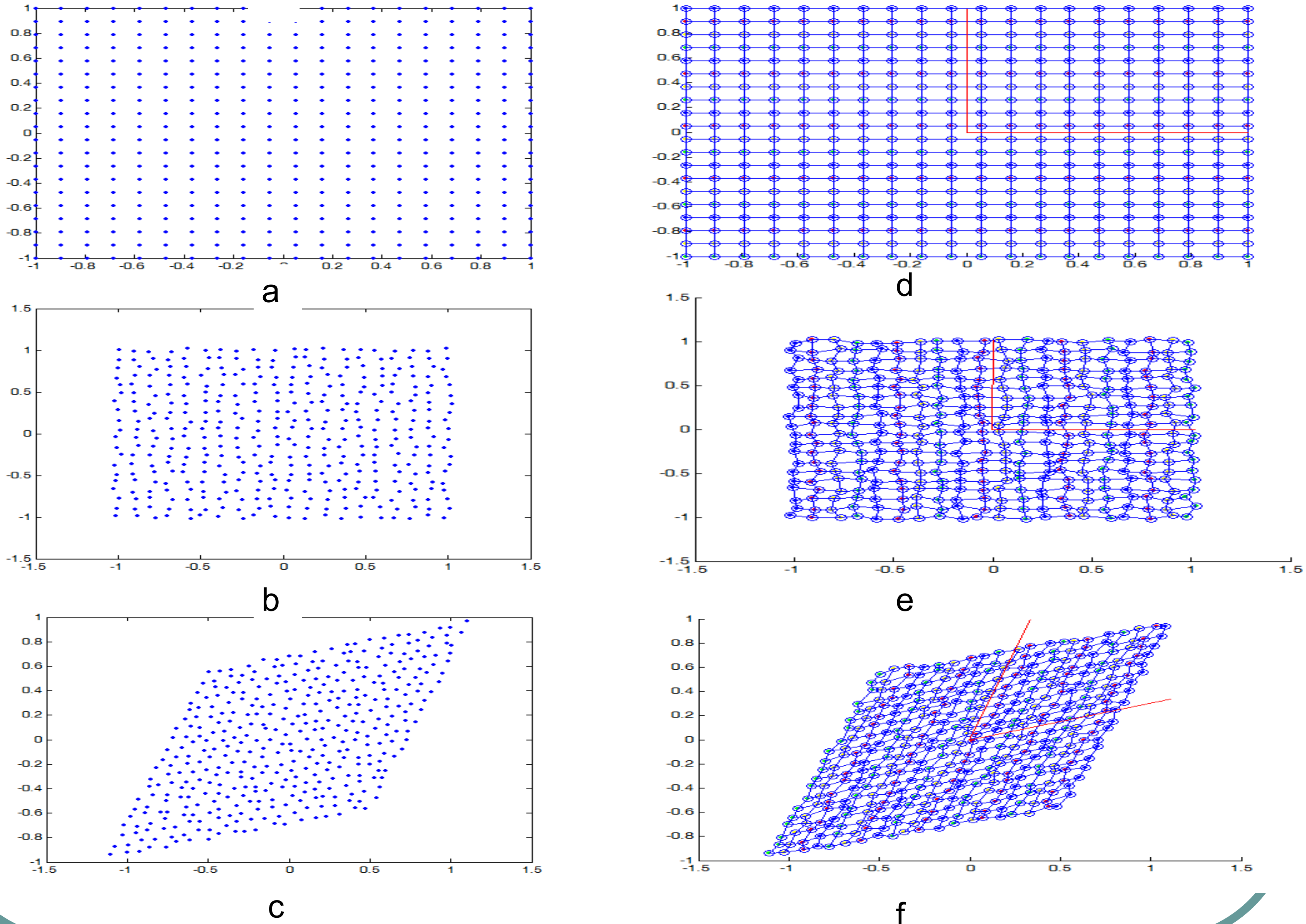


Figure 3

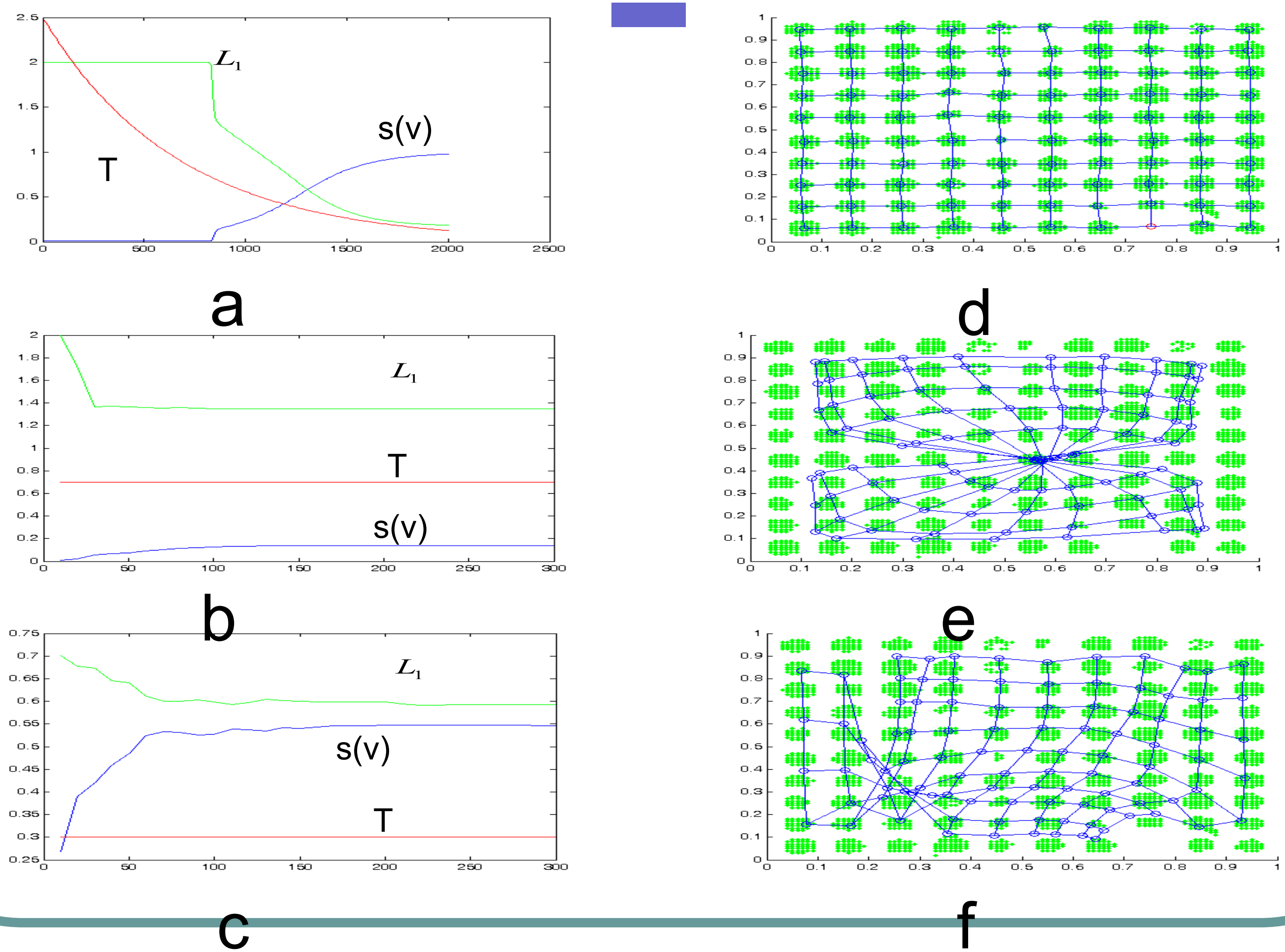


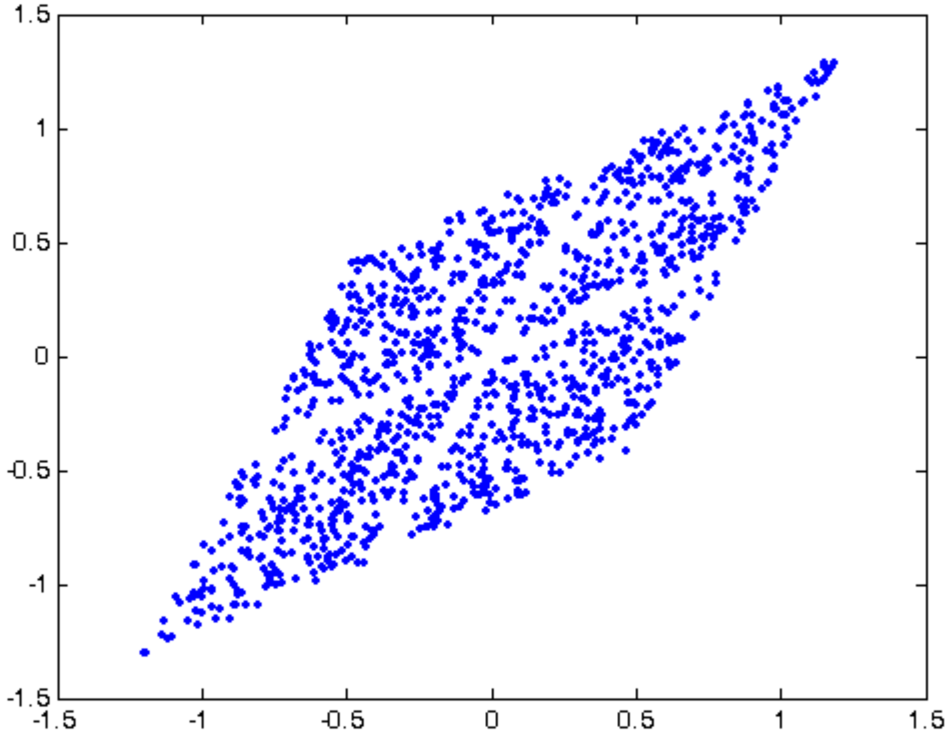
Figure 6

- Goodness-of-fit
- Objective functions
- Global minima versus Local minima

Neurocomputing: Learning, Architectures and Modeling

Rotated distributed clusters

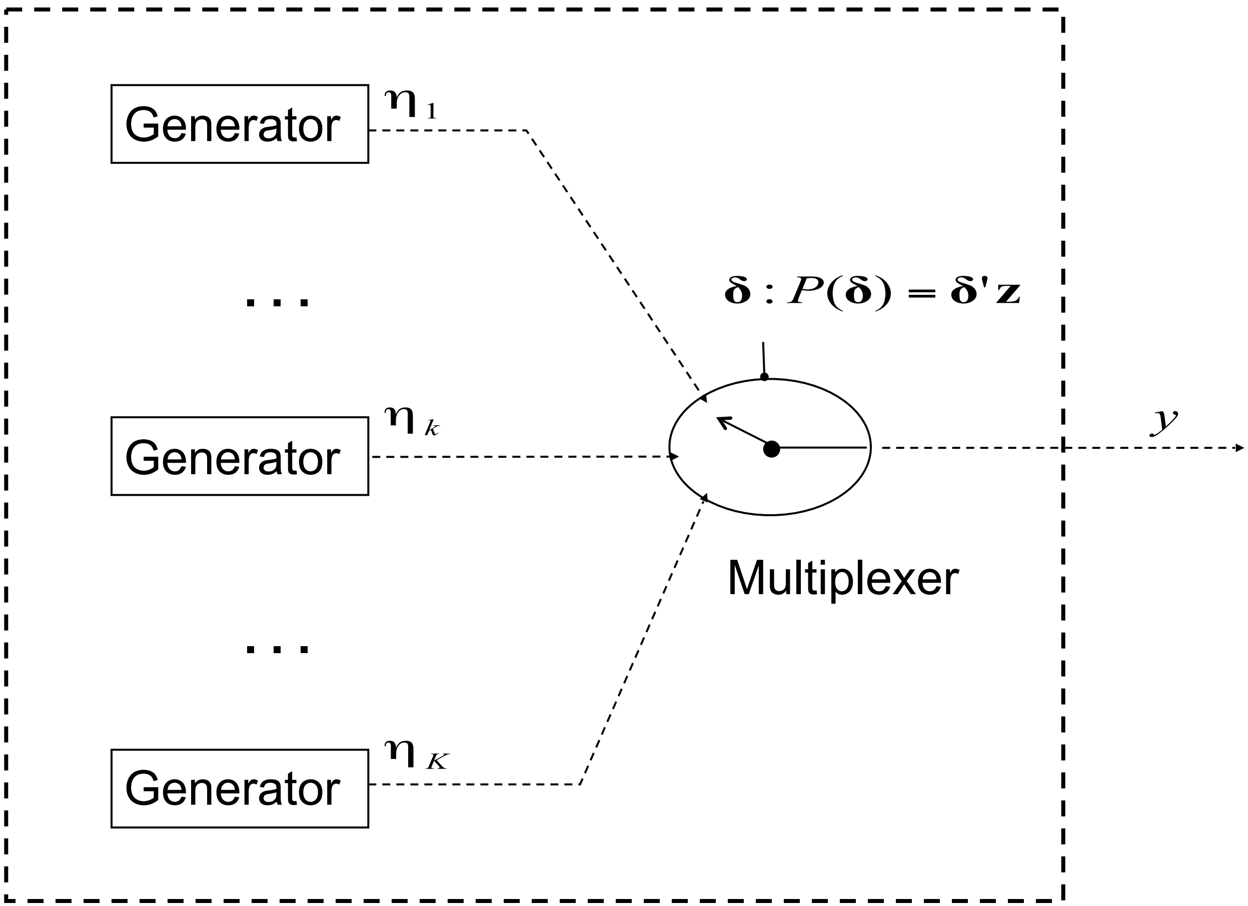
Data structure:
Local means
Rotation matrix



Density function approximation

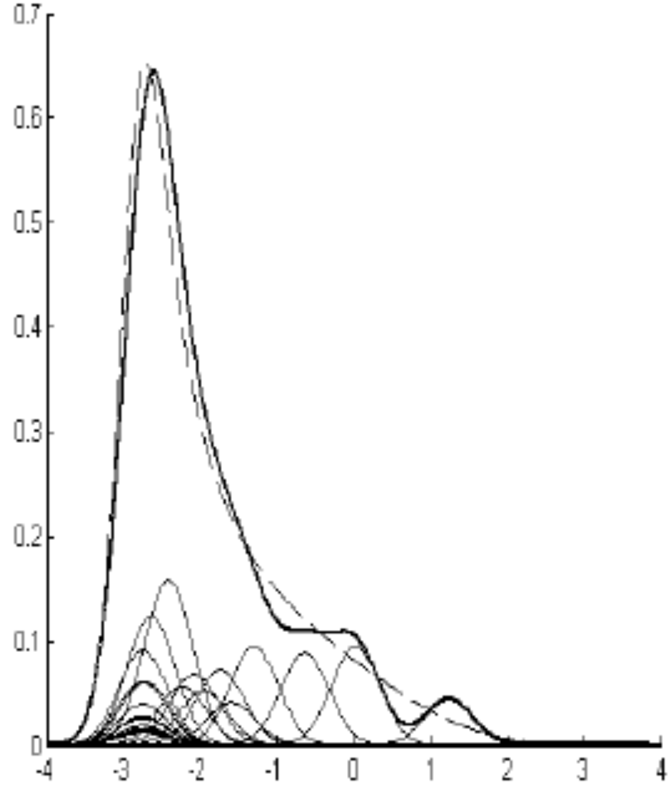
- Unbounded non-kernel space
  - Model based DFA
- Bounded non-kernel space
  - Boundary approximation
  - DFA within bounded non-kernel space

A generative model for Gaussian mixtures



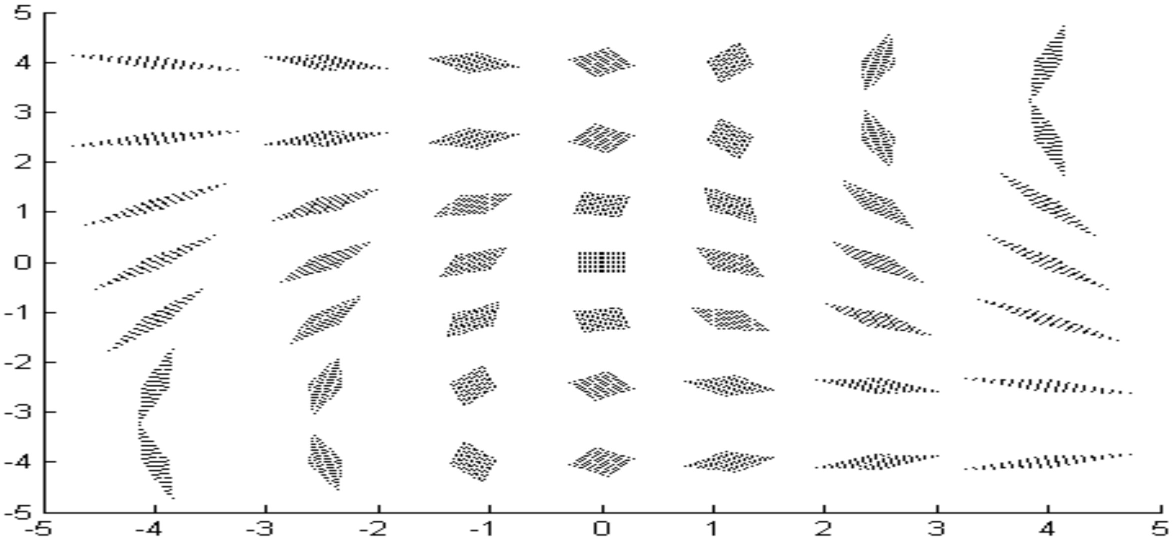
Density estimation

Density estimation by weighted Parzen windows

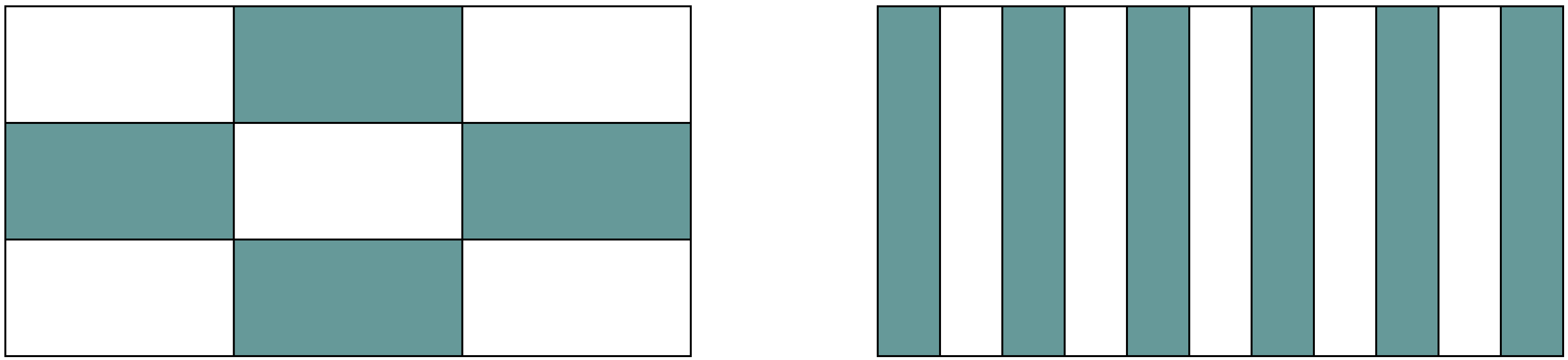


Example : Distributed clusters

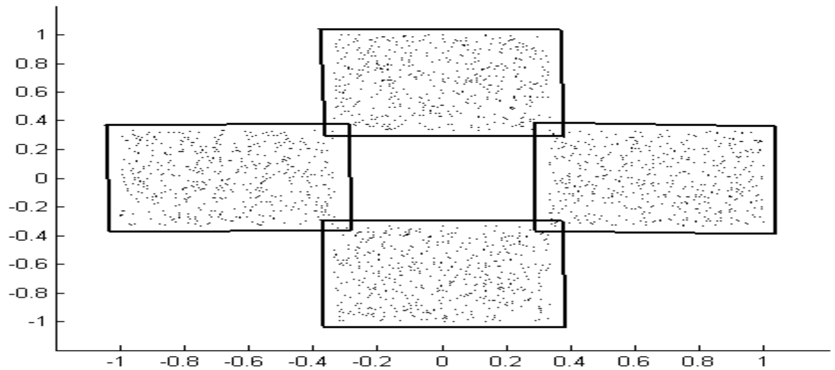
Different rotating matrices

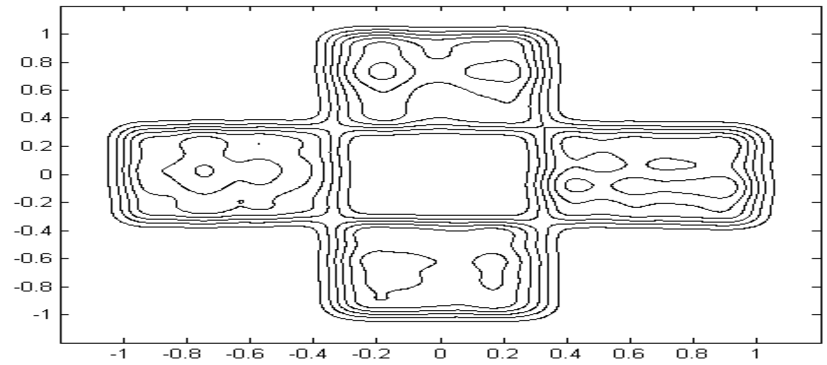


Example: Distributed columns

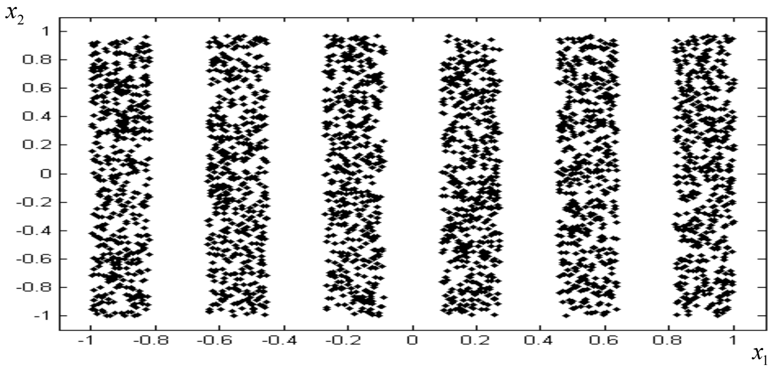


Density estimation

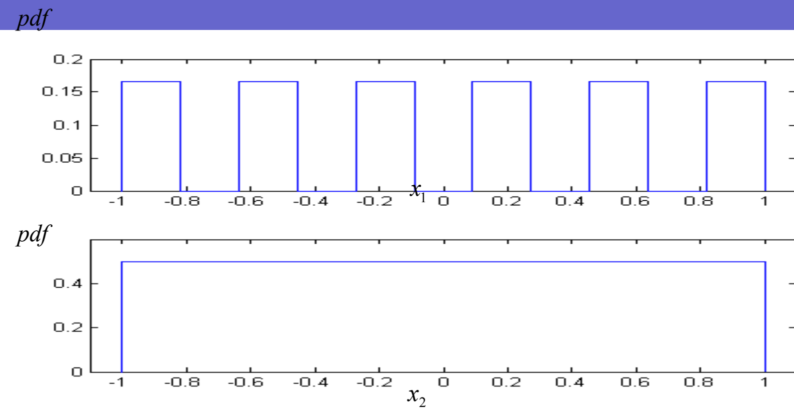




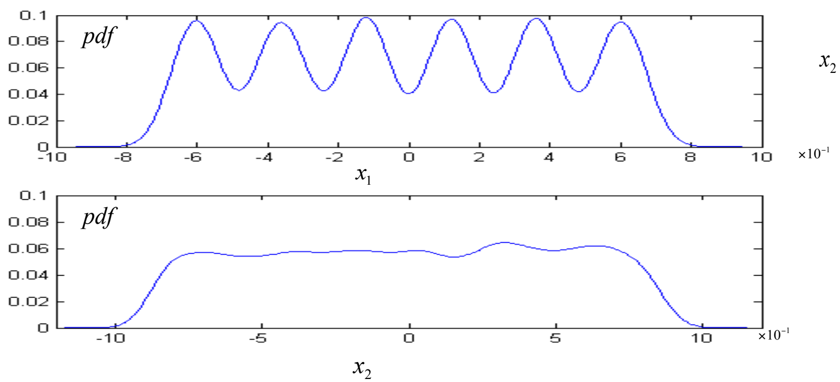
Distributed columns



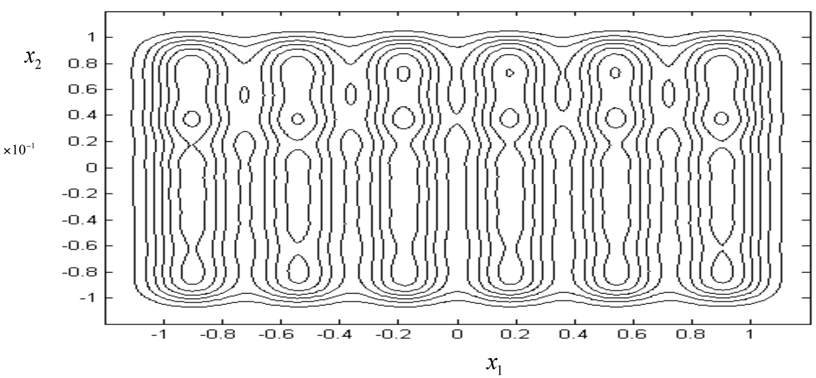
a



b

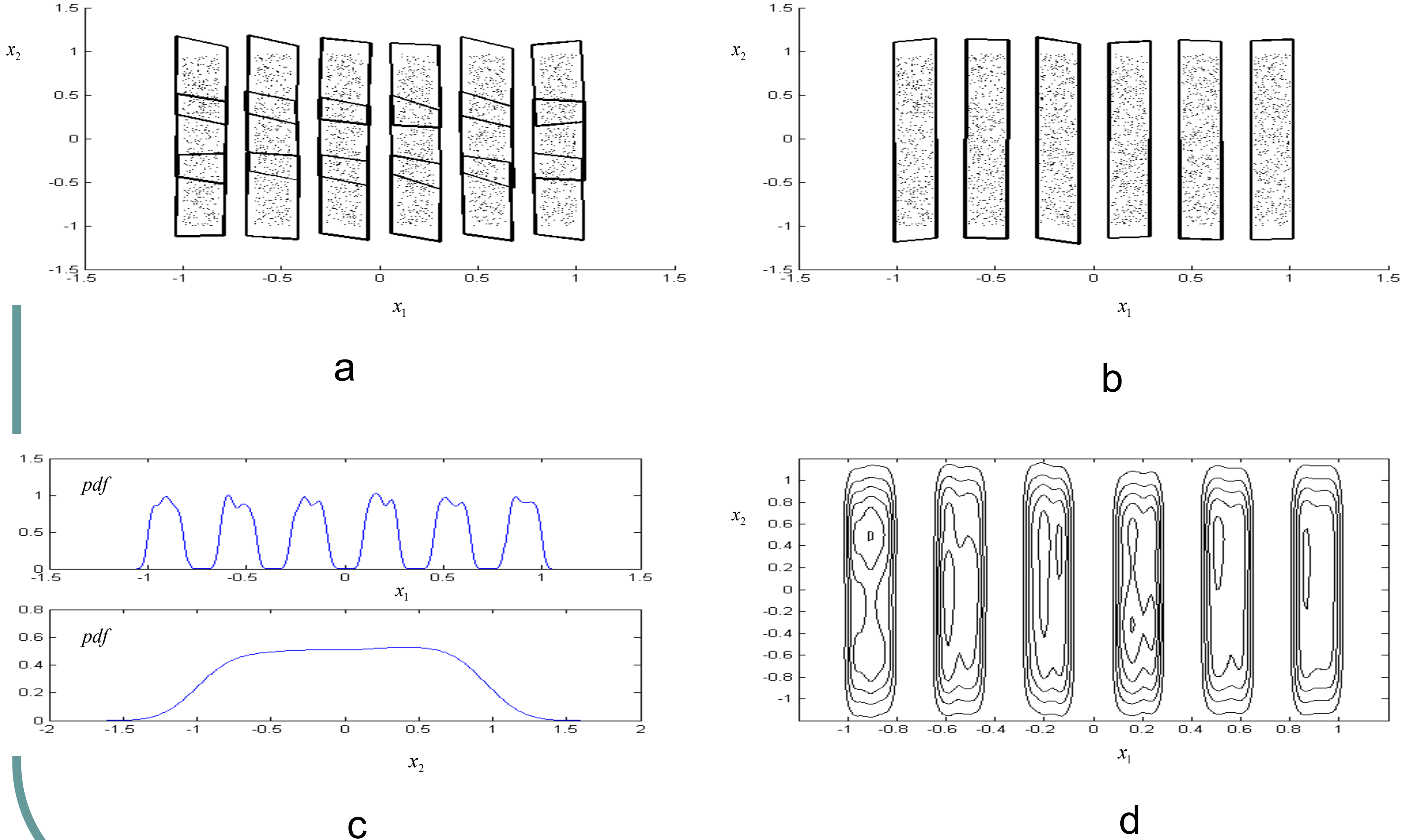


c

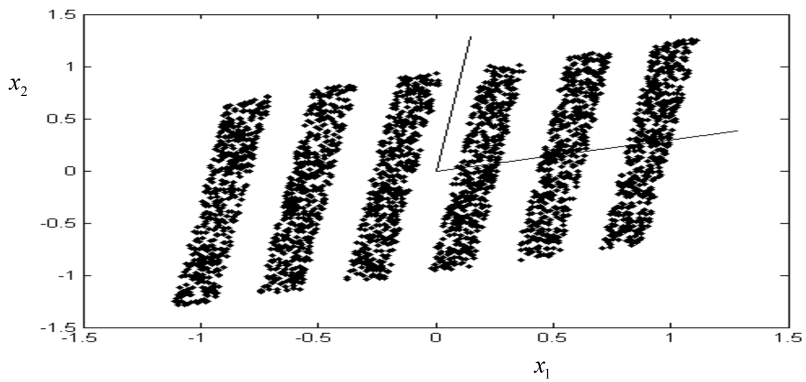


d

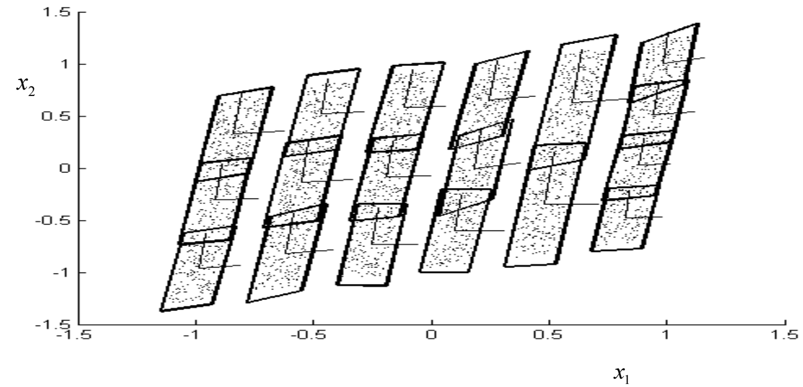
Distributed columns



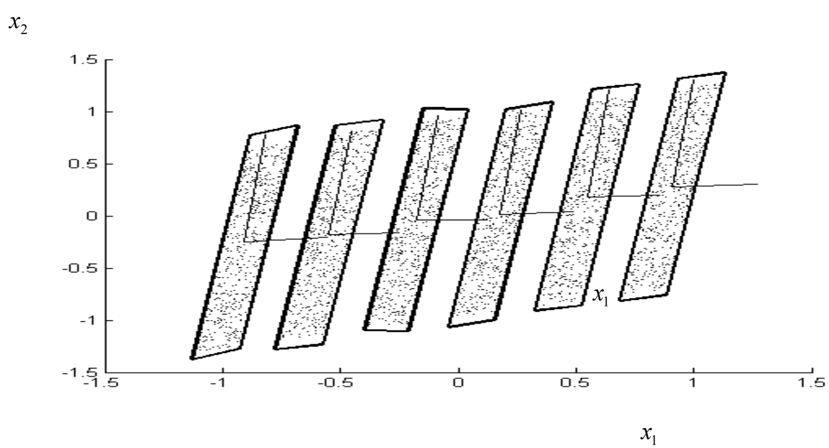
Rotated distributed columns



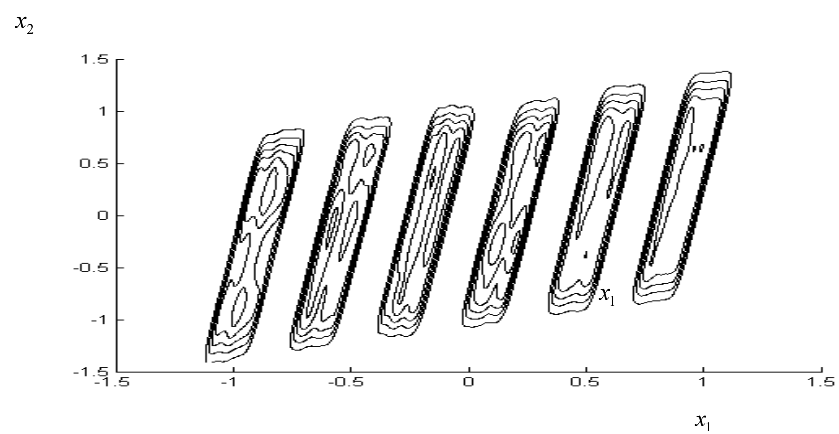
a



b

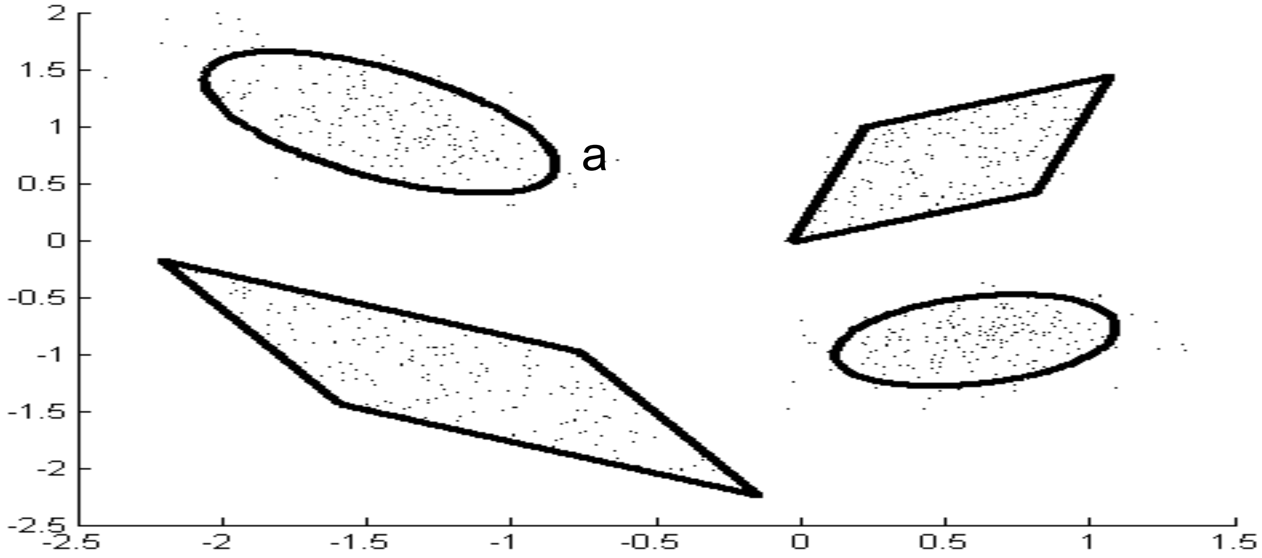


c



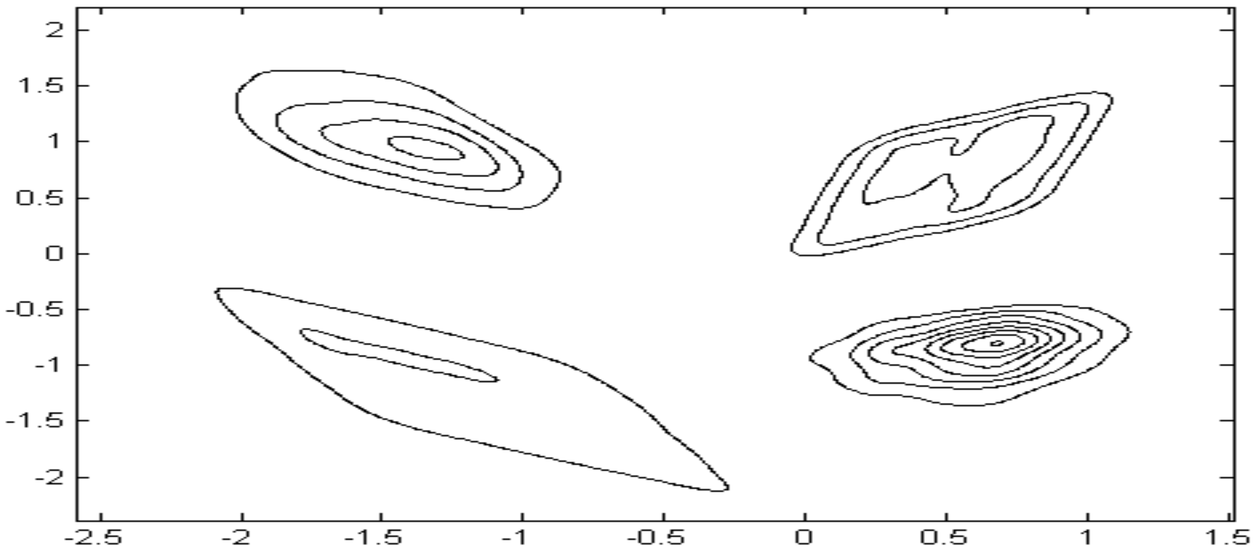
d

Example: Distributed clusters

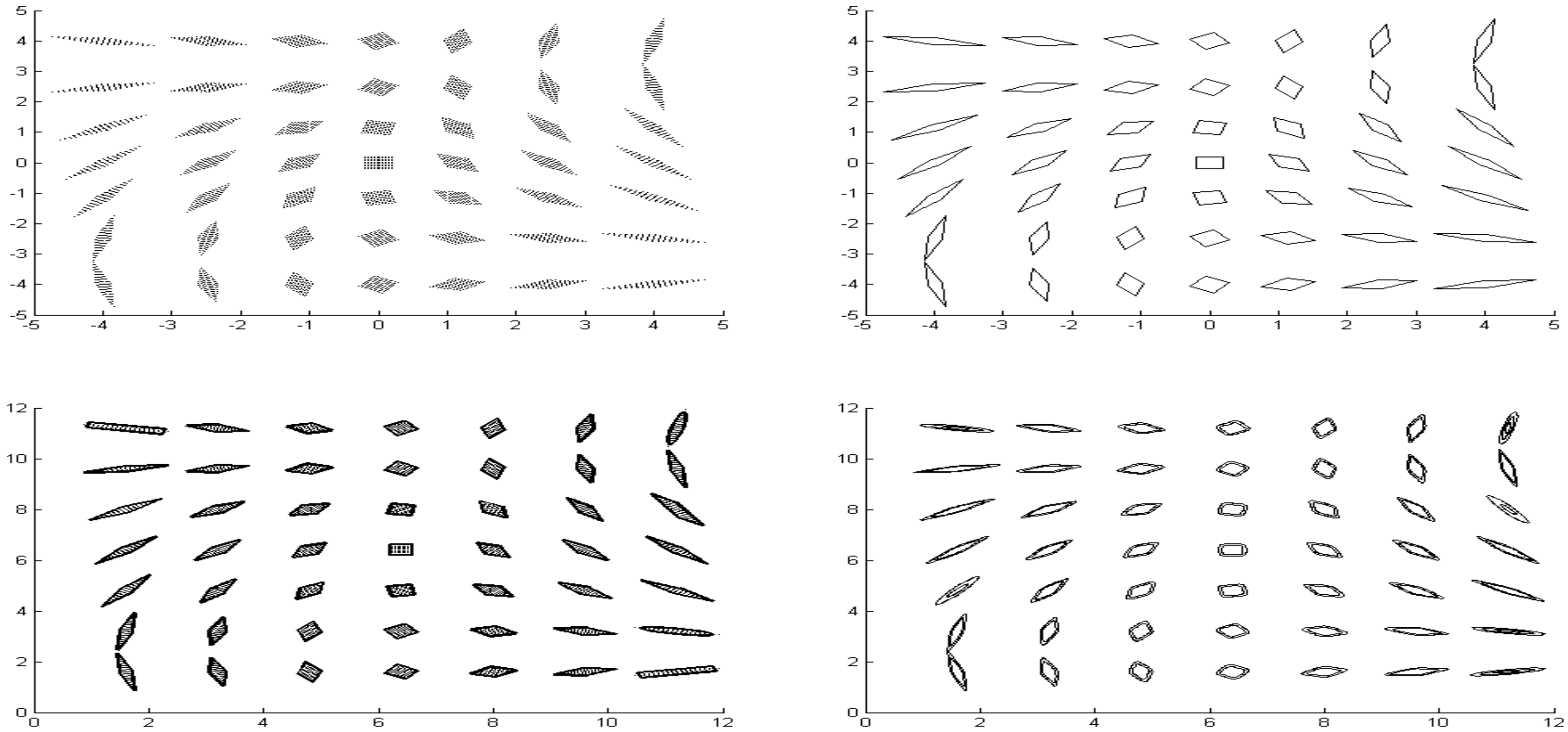


b

Density estimation

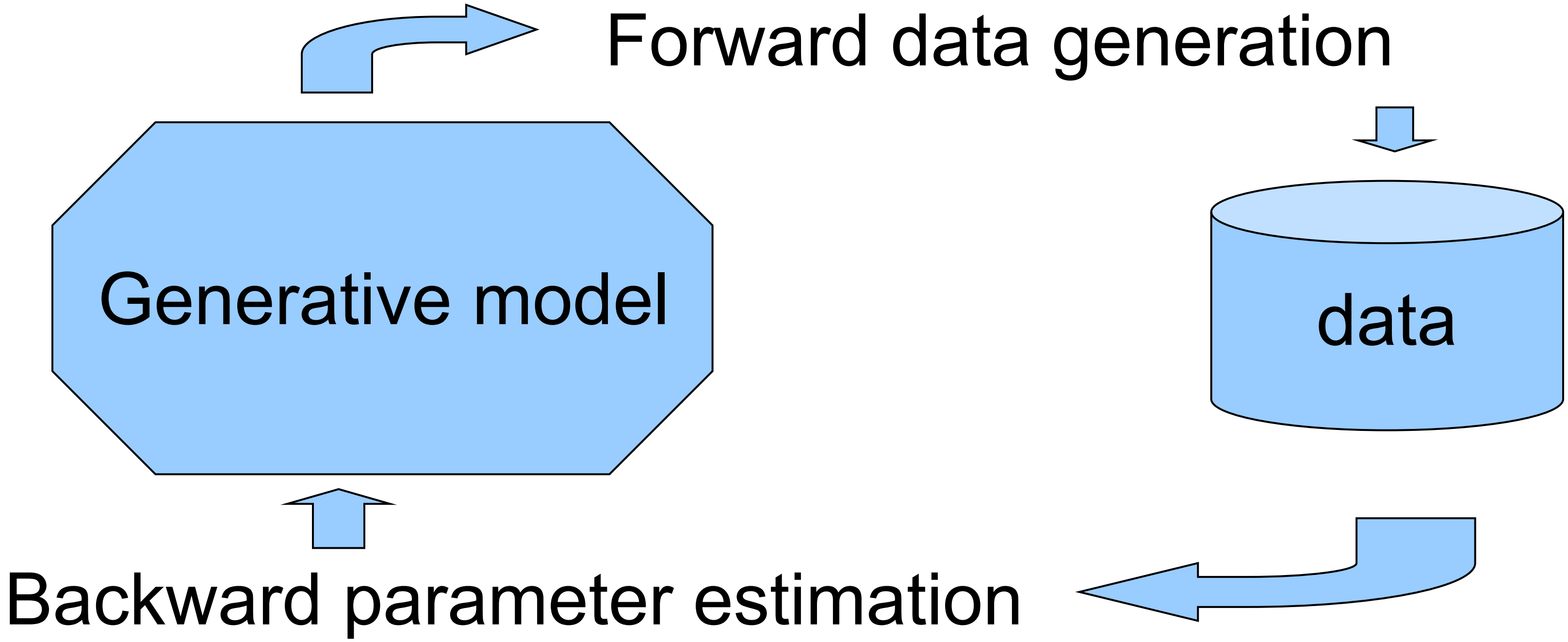


Distributed clusters

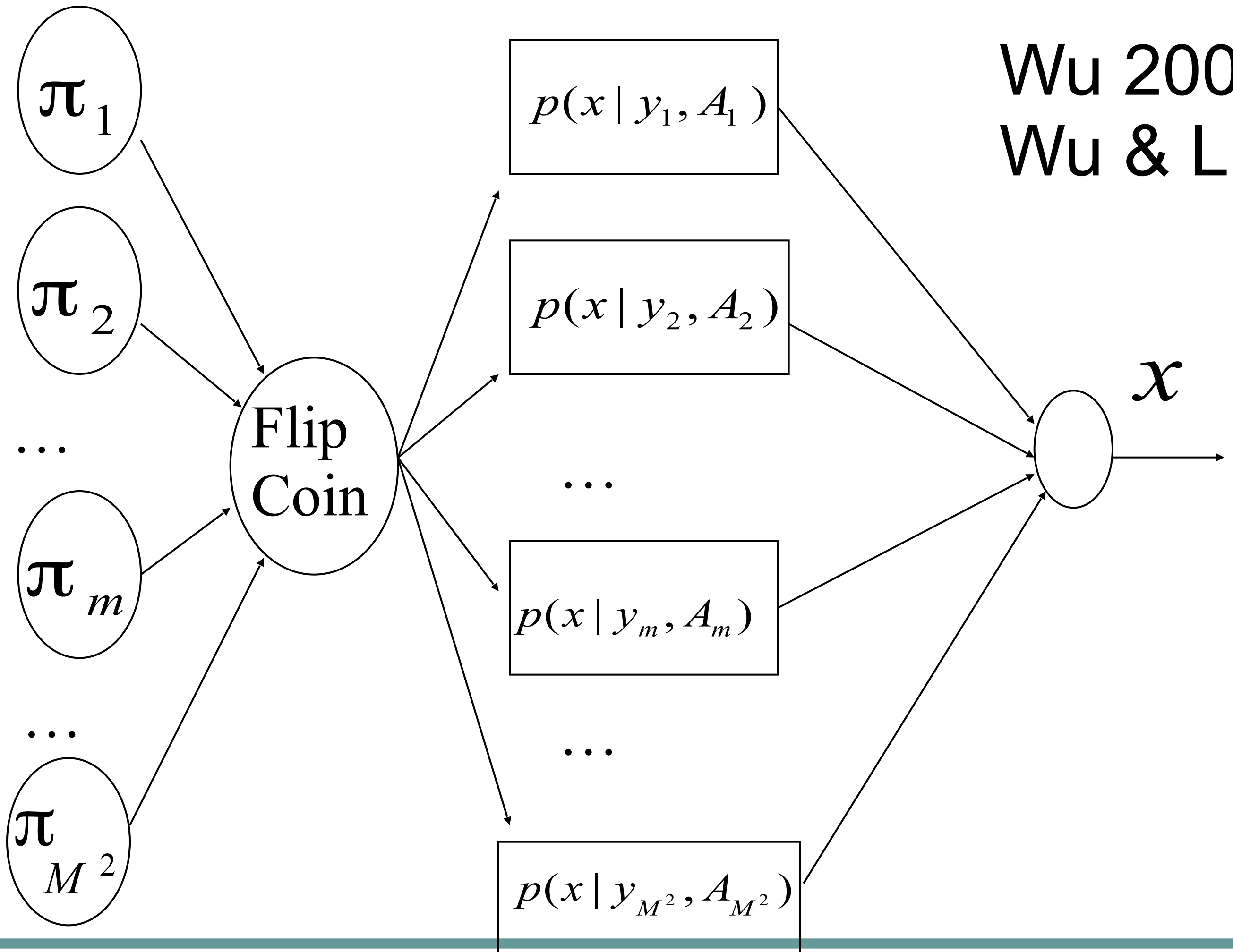


Model based density function approximation

- Generative model
  - A stochastic mechanism for data generation
- Reverse engineering
  - Generative model for emulating data formation
  - Model fitting for adapting bulit-in parameters


Forward data generation
Generative model
data
Backward parameter estimation

Gaussian Mixture Generative model

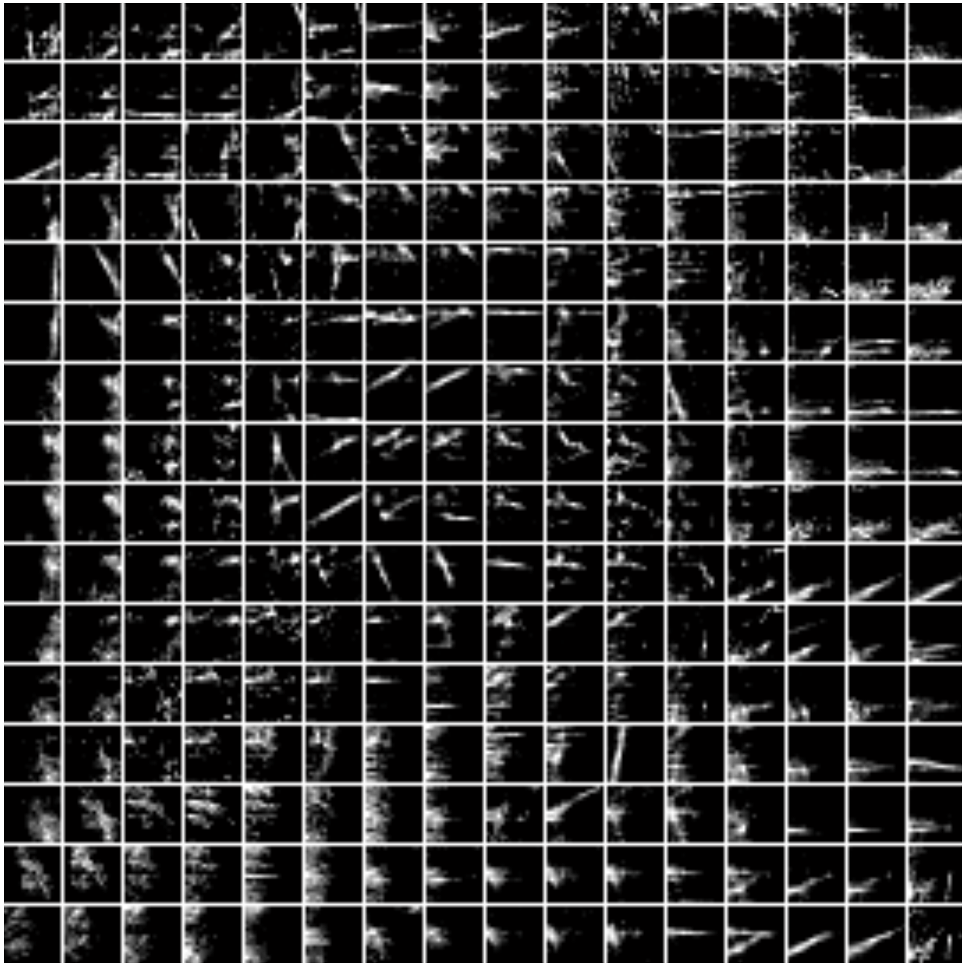


Wu 2002
Wu & Lin 2002

Analysis of Natural images

Generative models of natural images

Local means

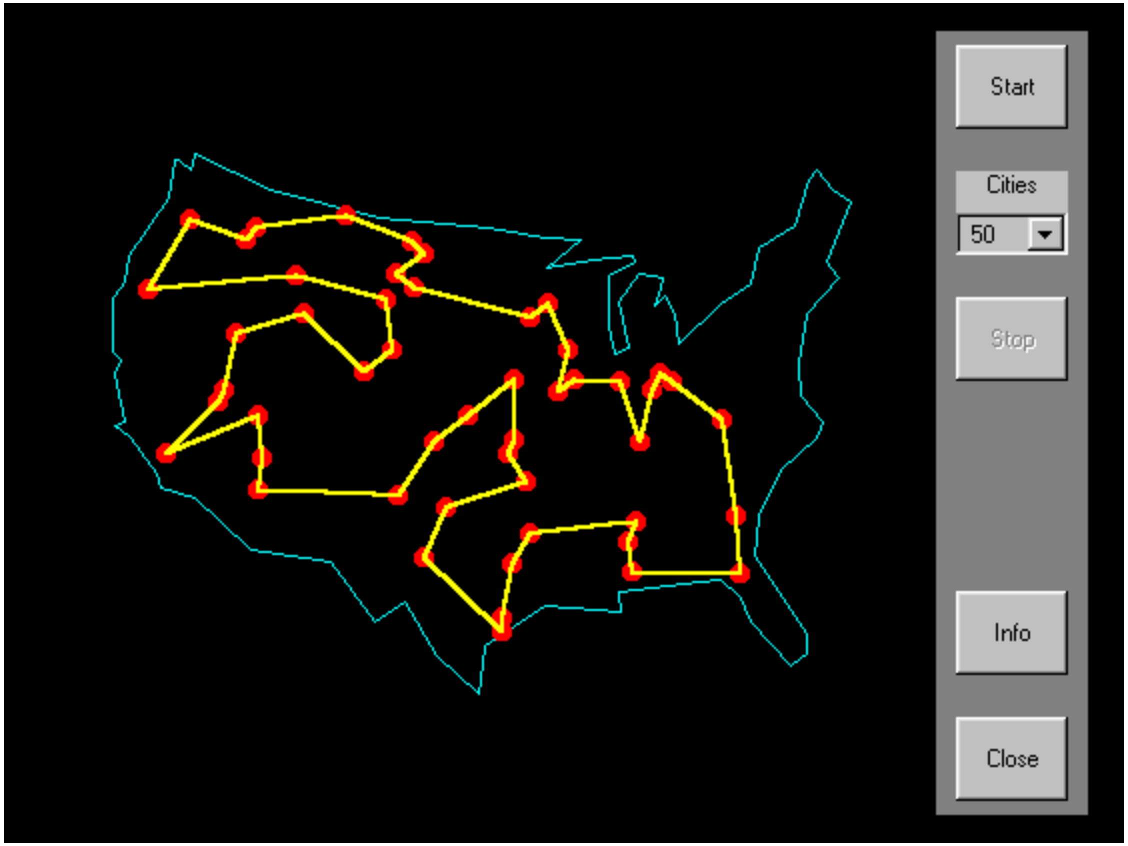


Constrained optimization

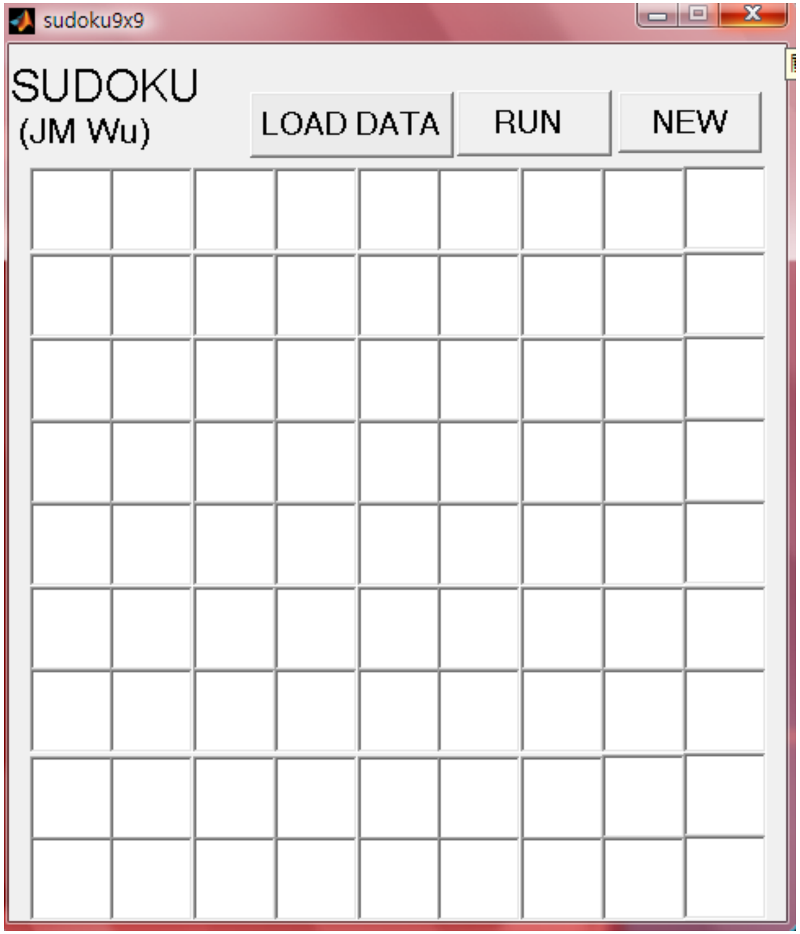
- TSP
- Sudoku
- Magic square, Rubik's Cube

Traveling salesman problems

Planar data ordering on a ring



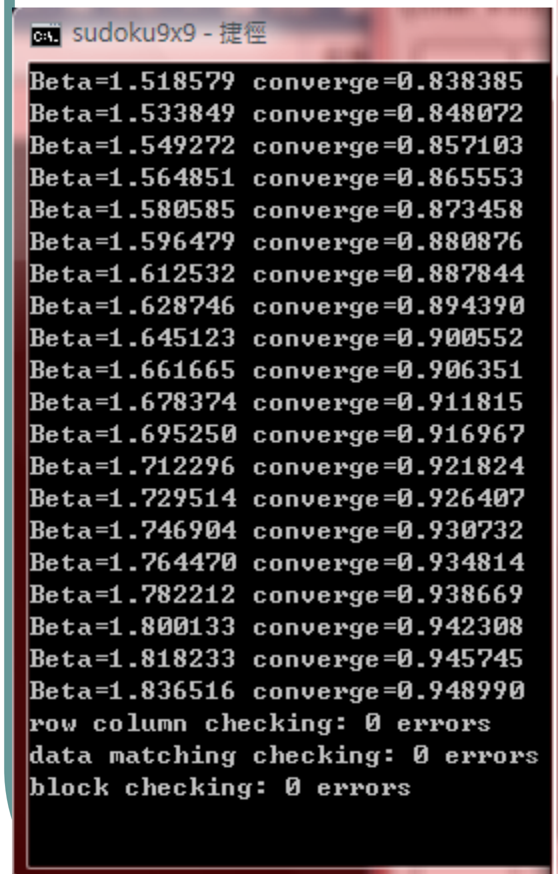
Sudoku



Sudoku


sudoku9x9
SUDOKU
(JM Wu)
LOAD DATA
RUN
NEW
4 5 2
8 6
2 5 8 4
6 7 9
1 3 6
3 1 7
1 9 2 6
5 7
6 3 9

Sudoku


sudoku9x9 - 捷徑
Beta=1.518579 converge=0.838385
Beta=1.533849 converge=0.848072
Beta=1.549272 converge=0.857103
Beta=1.564851 converge=0.865553
Beta=1.580585 converge=0.873458
Beta=1.596479 converge=0.880876
Beta=1.612532 converge=0.887844
Beta=1.628746 converge=0.894390
Beta=1.645123 converge=0.900552
Beta=1.661665 converge=0.906351
Beta=1.678374 converge=0.911815
Beta=1.695250 converge=0.916967
Beta=1.712296 converge=0.921824
Beta=1.729514 converge=0.926407
Beta=1.746904 converge=0.930732
Beta=1.764470 converge=0.934814
Beta=1.782212 converge=0.938669
Beta=1.800133 converge=0.942308
Beta=1.818233 converge=0.945745
Beta=1.836516 converge=0.948990
row column checking: 0 errors
data matching checking: 0 errors
block checking: 0 errors


sudoku9x9
SUDOKU
(JM Wu)
LOAD DATA
RUN
NEW
4 5 8 6 2 9 7 1 3
9 7 1 3 8 4 2 5 6
3 6 2 7 1 5 9 8 4
6 2 4 8 5 7 3 9 1
7 1 9 4 3 2 5 6 8
8 3 5 1 9 6 4 2 7
1 9 3 2 4 8 6 7 5
5 8 6 9 7 3 1 4 2
2 4 7 5 6 1 8 3 9

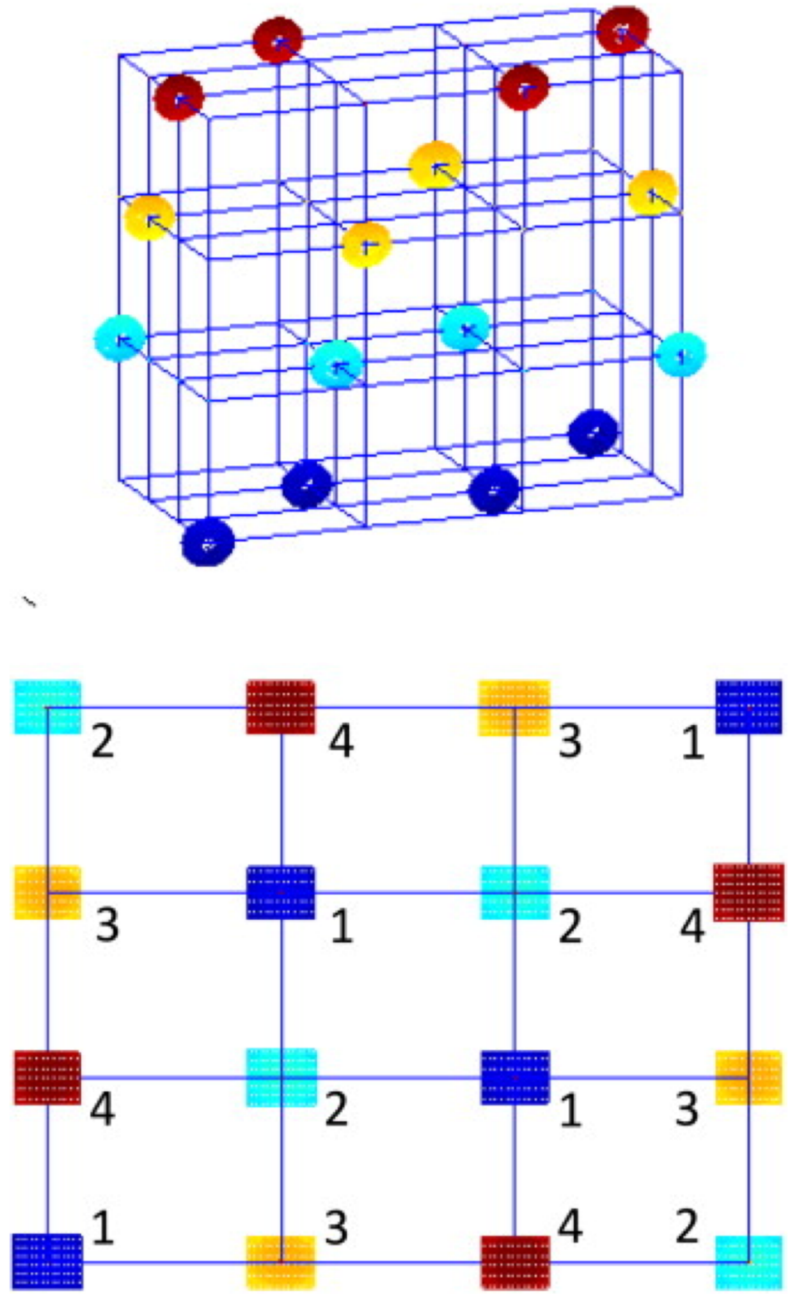


Fig. 6. Bipolar neural activations for Sudoku encoding of $K = 4$.

research direction of Sudoku associative memory. The goal is to achieve automatic error detection, error correction and restoration of Sudoku-rule embedded patterns subject to fewer partial clues, condense clues and perturbed or damaged clues.

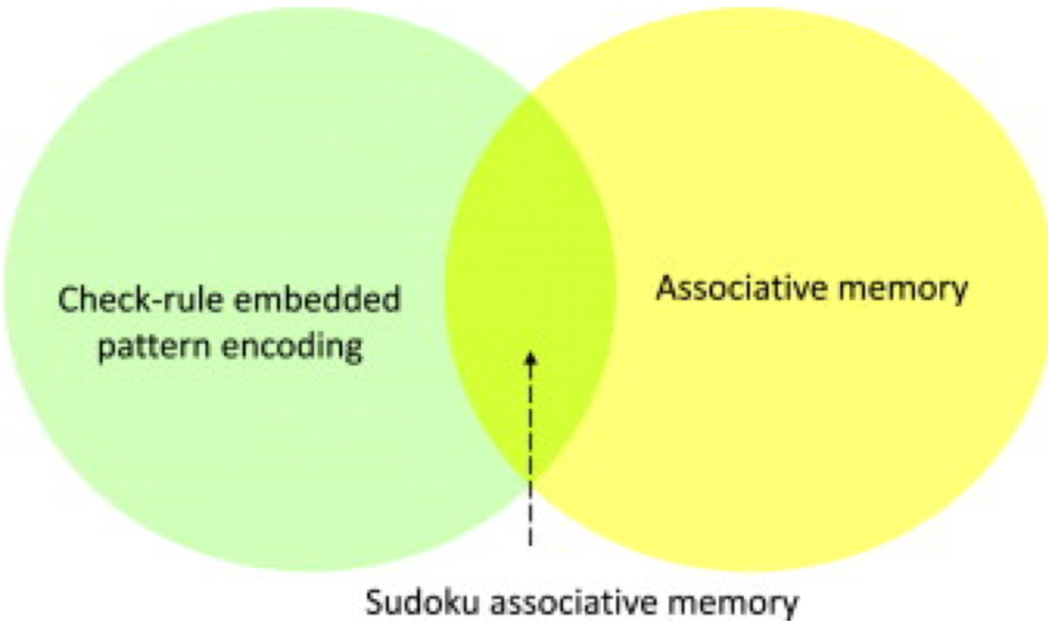
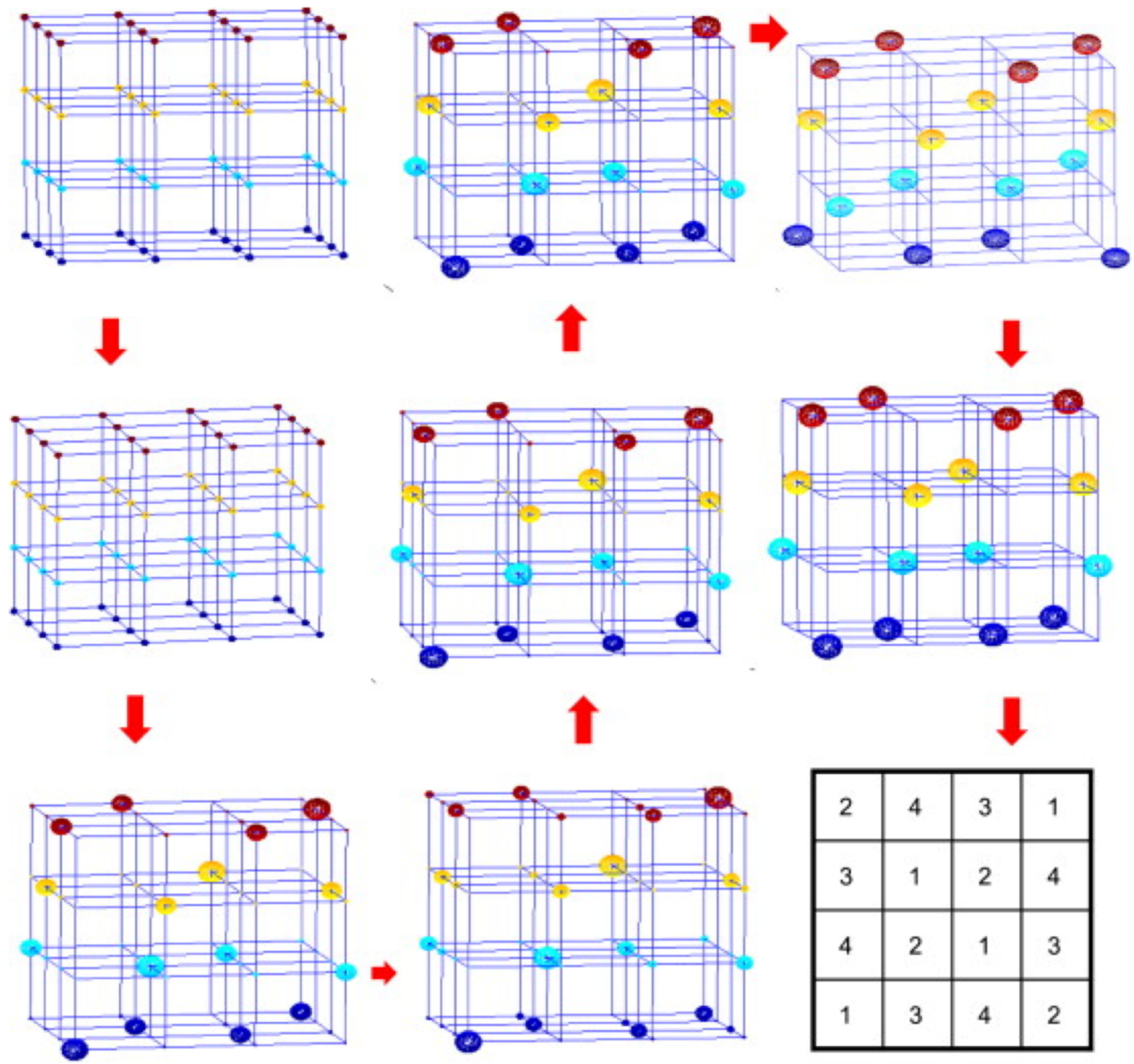


Fig. 7. The concept of developing SAM based on check-rule embedded pattern encoding and associative memory.



2	4	3	1
3	1	2	4
4	2	1	3
1	3	4	2

Fig. 10. Evolution of neural activations for general Sudoku restoration with $K = 4$ along an annealing process.

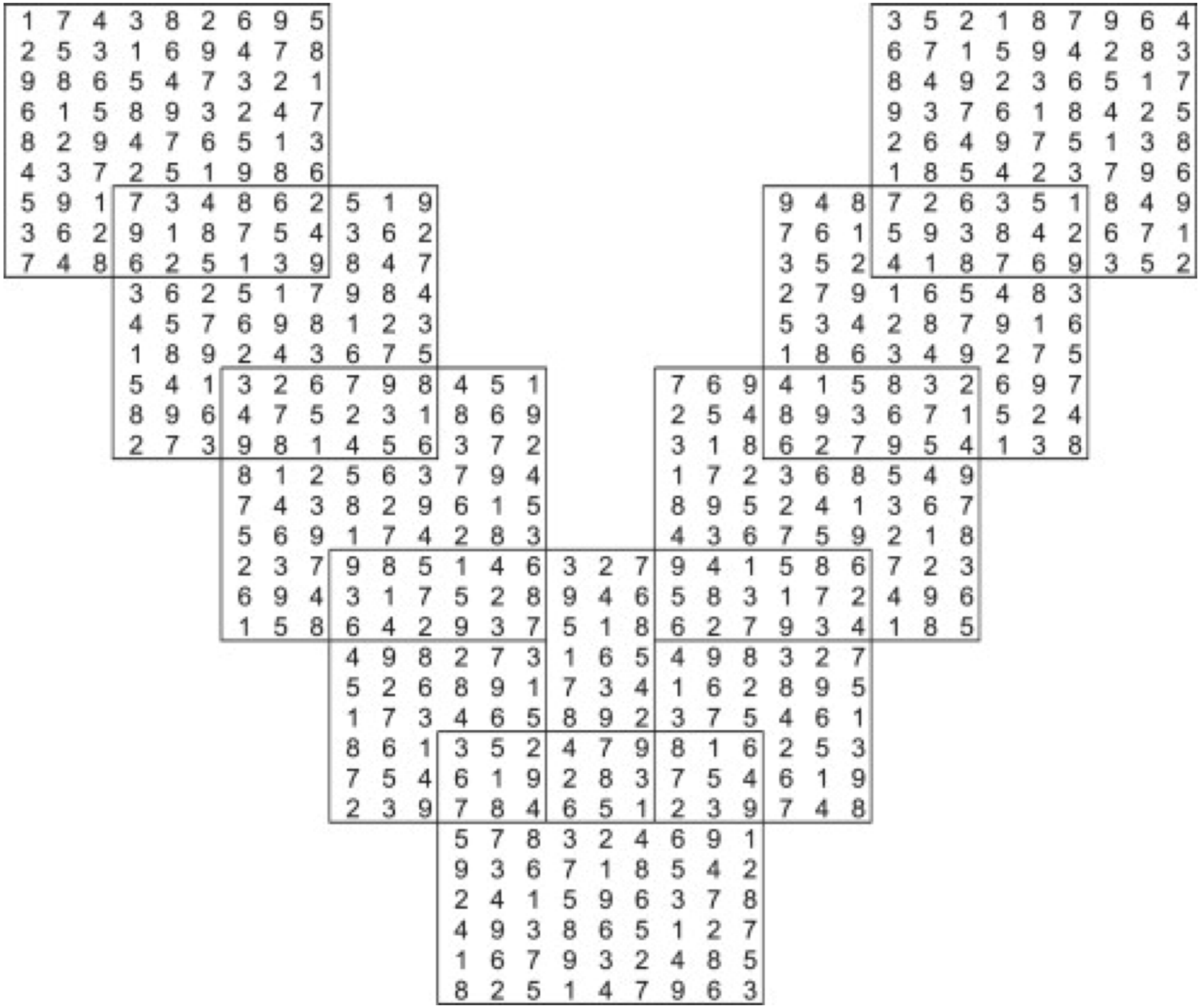


Fig. 12. A V-shape compound Sudoku pattern.

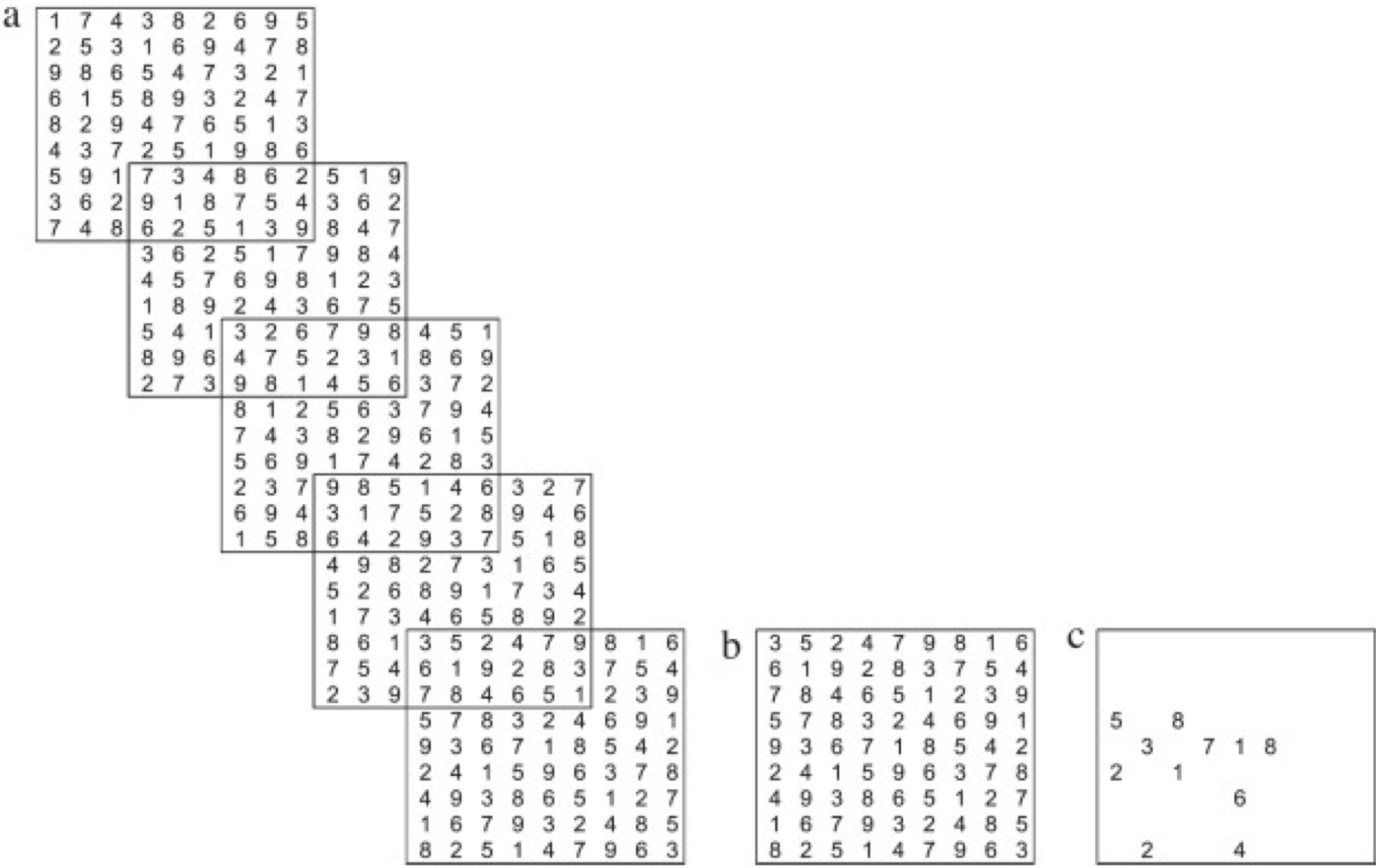


Fig. 13. Different partial clues for V-shape compound Sudoku pattern restoration, (a) a left part of the V-shape compound pattern, (b) a central pattern, and (c) a damaged central pattern.

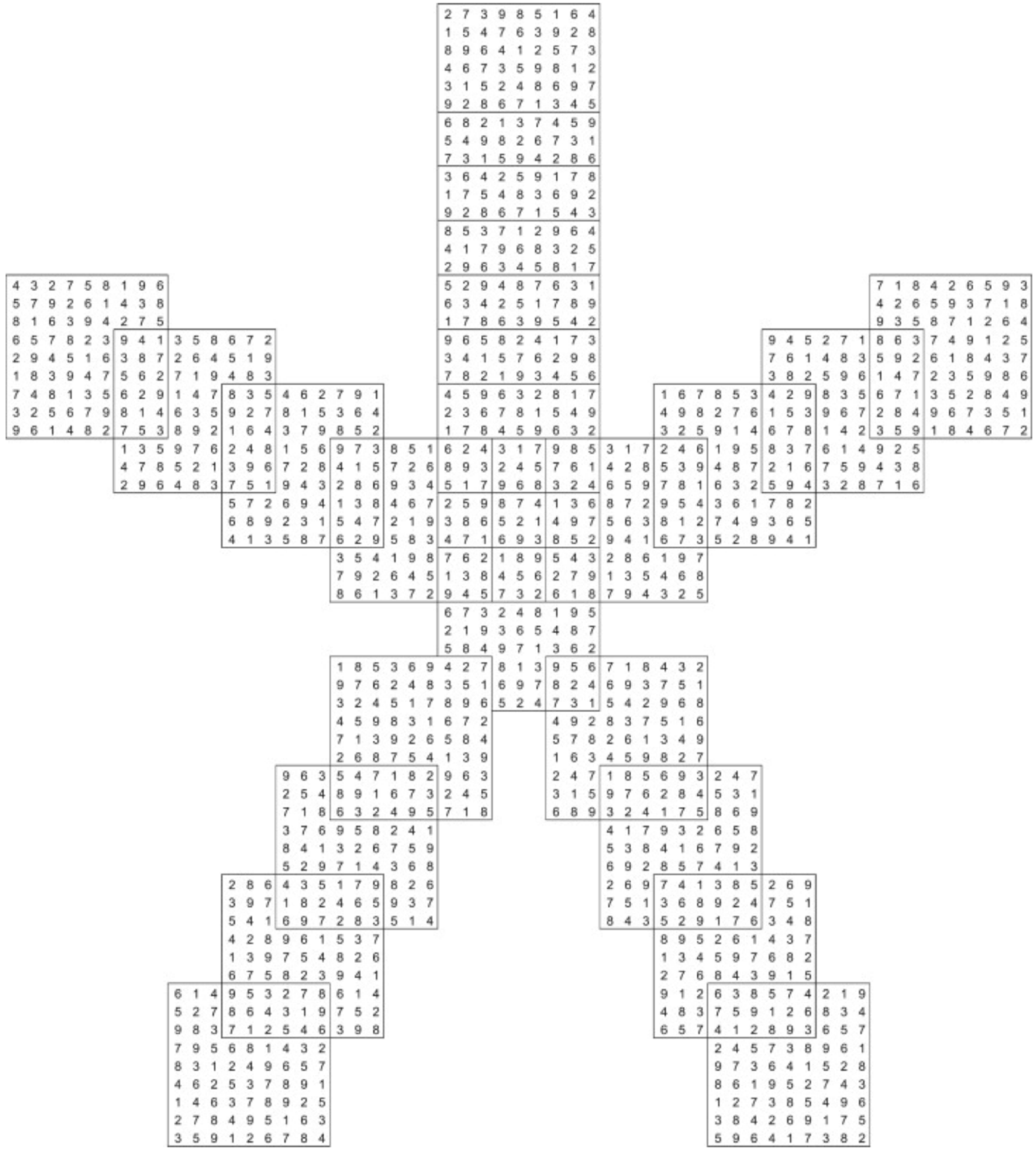


Fig. 14. A starfish-shape compound Sudoku pattern.

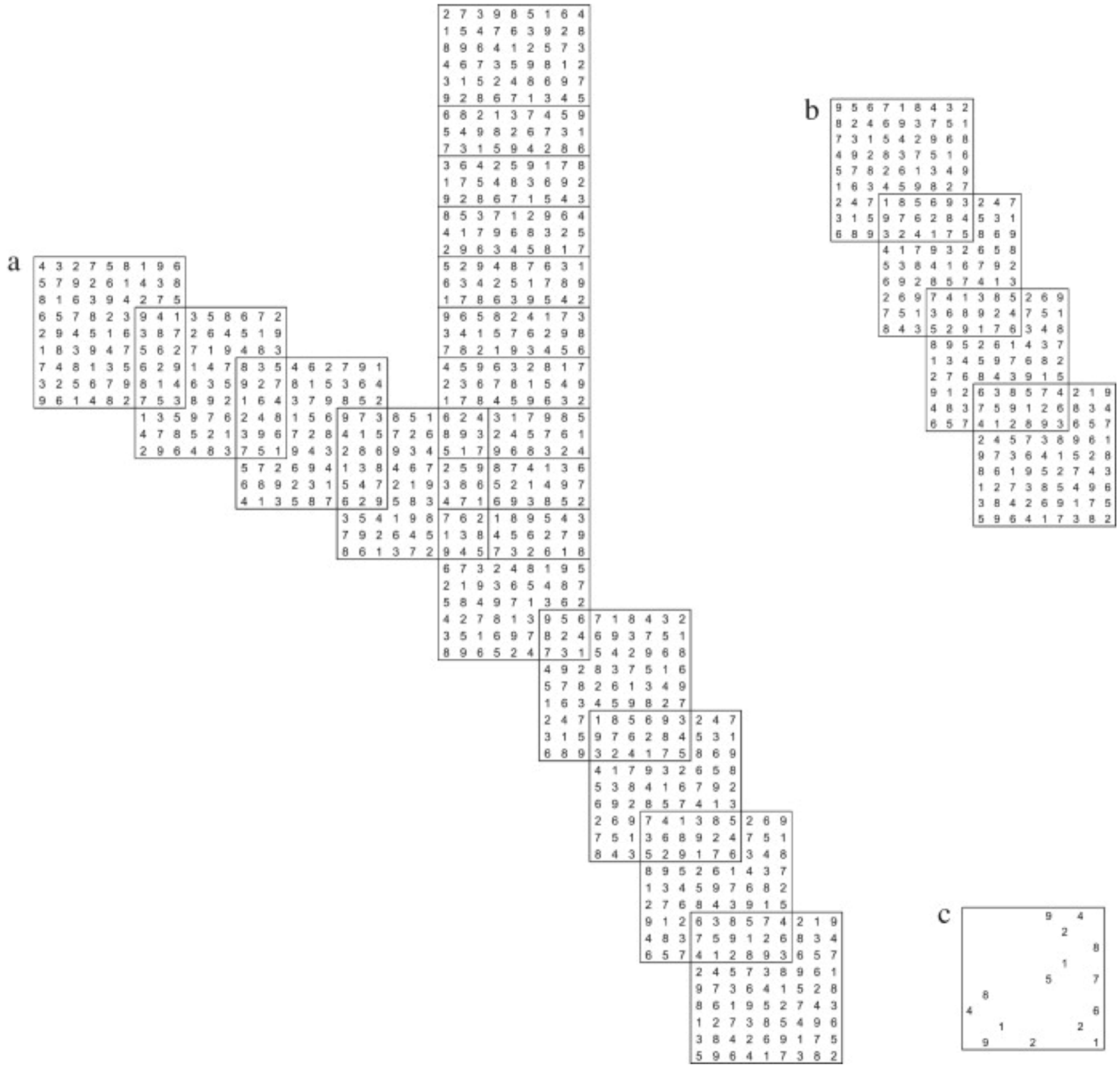


Fig. 15. Different partial clues for starfish shape compound Sudoku pattern restoration, (a) three tentacles, (b) one tentacle, and (c) a damaged seed pattern.

Rubik’s Cube

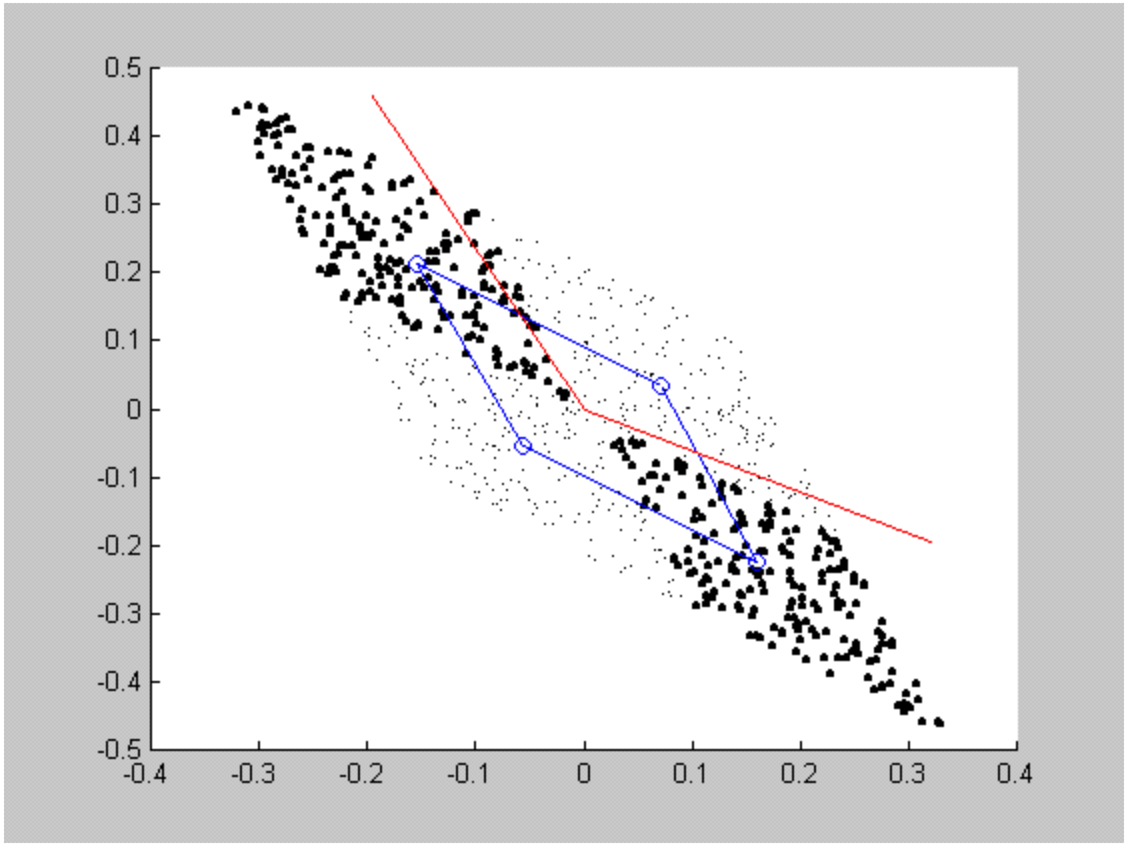
- Rubik's Cube - Wikipedia, the free encyclopedia
- MATLAB Central File Exchange - Rubix Cube

Blind source separation

- Independent component analysis
- Convolutive ICA
- Sound separation
- Fetal ECG extraction
- ERP
- Functional MRI

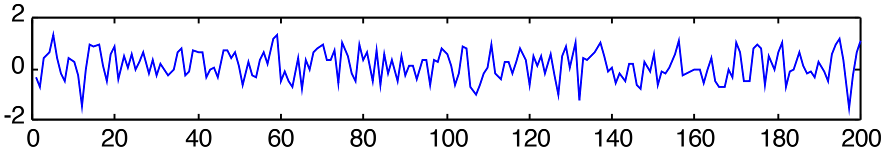
Independent component analysis

Rotated independent data



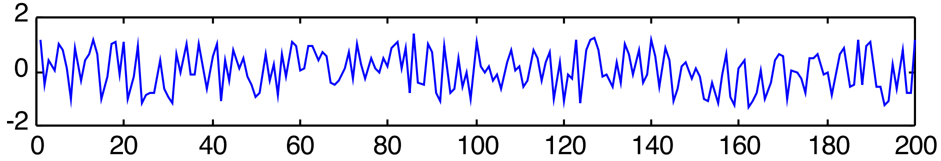
Sources

Observations

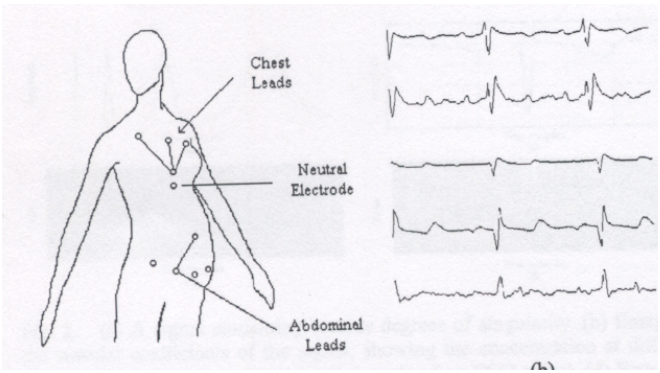


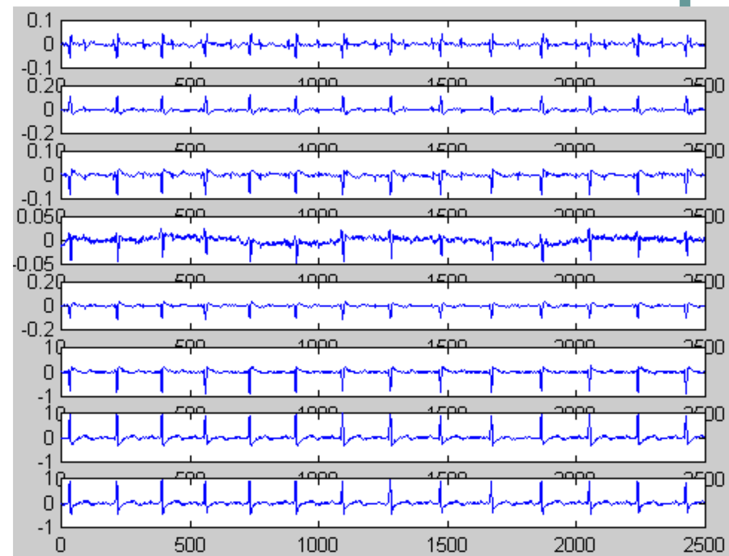
ICs

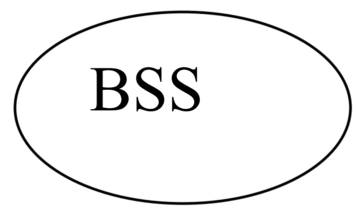
Recovered Sources by ICA

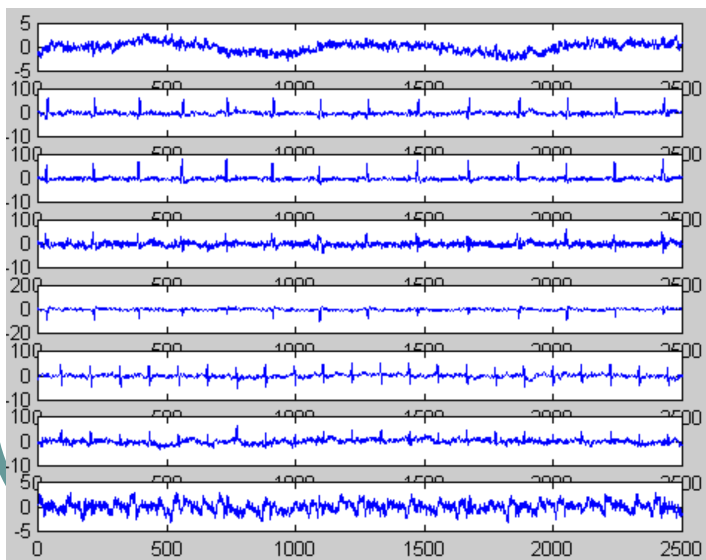


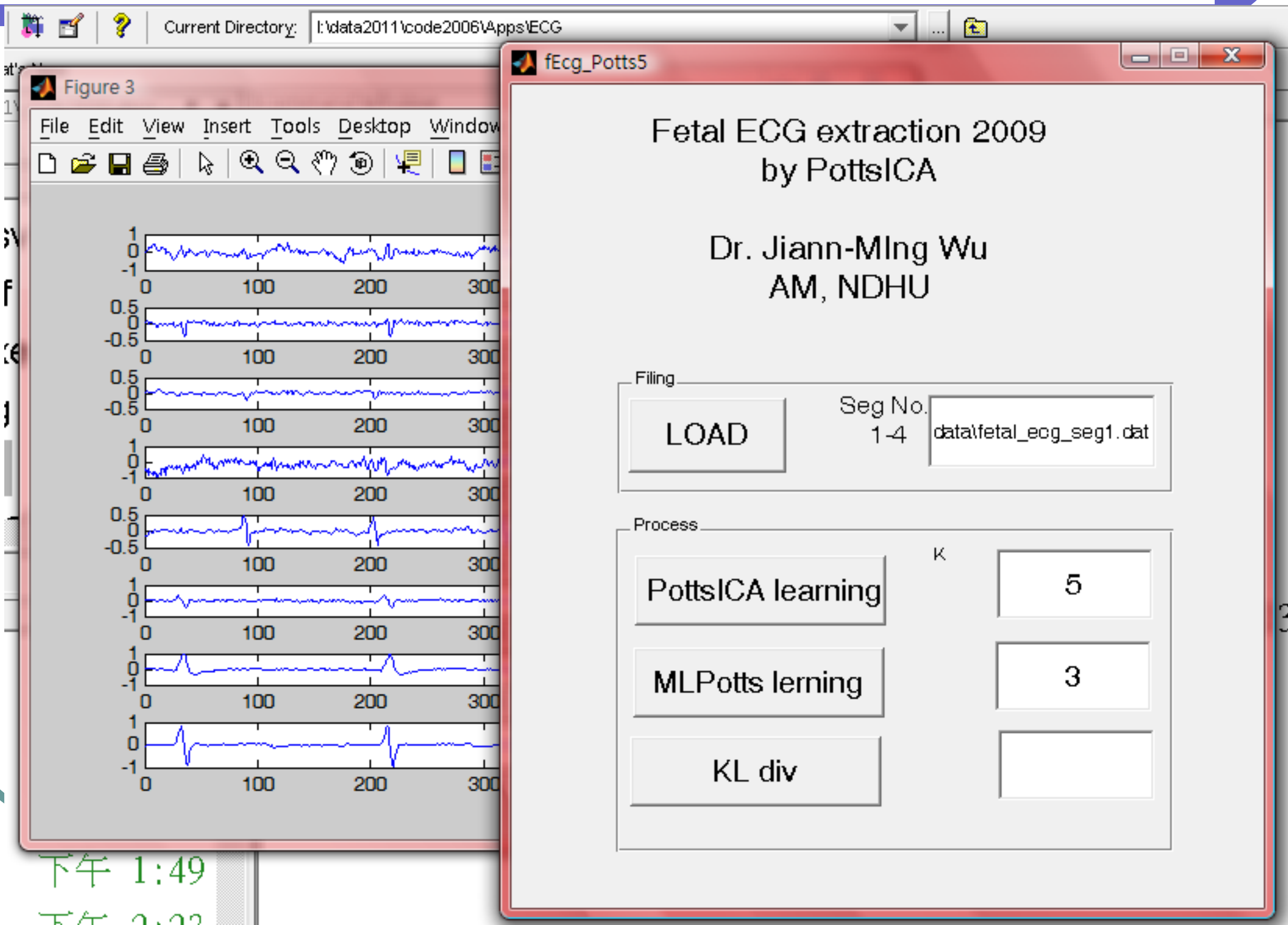
Blind source separation – fetal ECG










Current Directory: I:\data2011\code2006\Apps\ECG
Figure 3
File Edit View Insert Tools Desktop Window
fEcg_Potts5
Fetal ECG extraction 2009
by PottsICA
Dr. Jiann-MIng Wu
AM, NDHU
Filing
LOAD
Seg No.
1-4
data\fetal_ecg_seg1.dat
Process
K
PottsICA learning
5
MLPotts lerning
3
KL div
下午 1:49

Mixed Facial images

Wu et al 2008

ERP(event related potential)

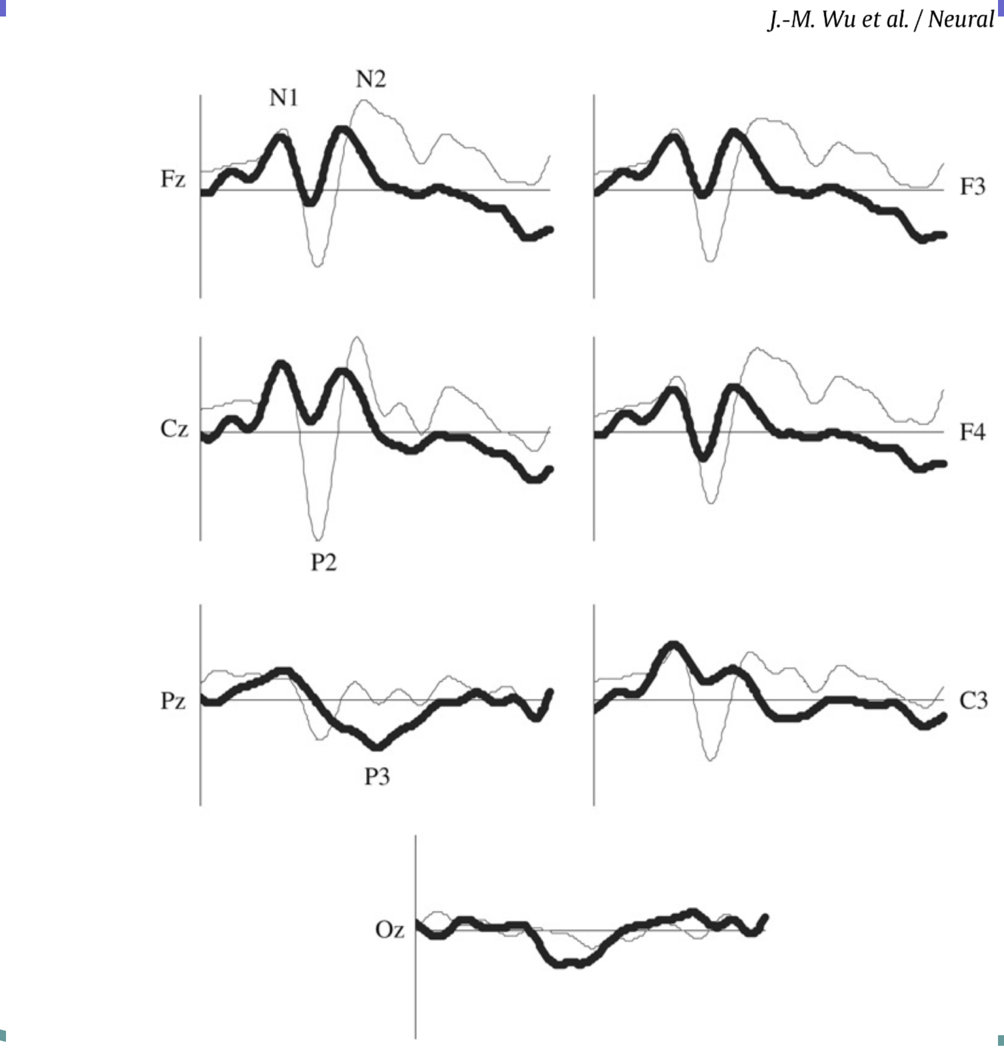


Fig. 11. Observed ERPs.

ICs of ERP (Wu et al 2008)

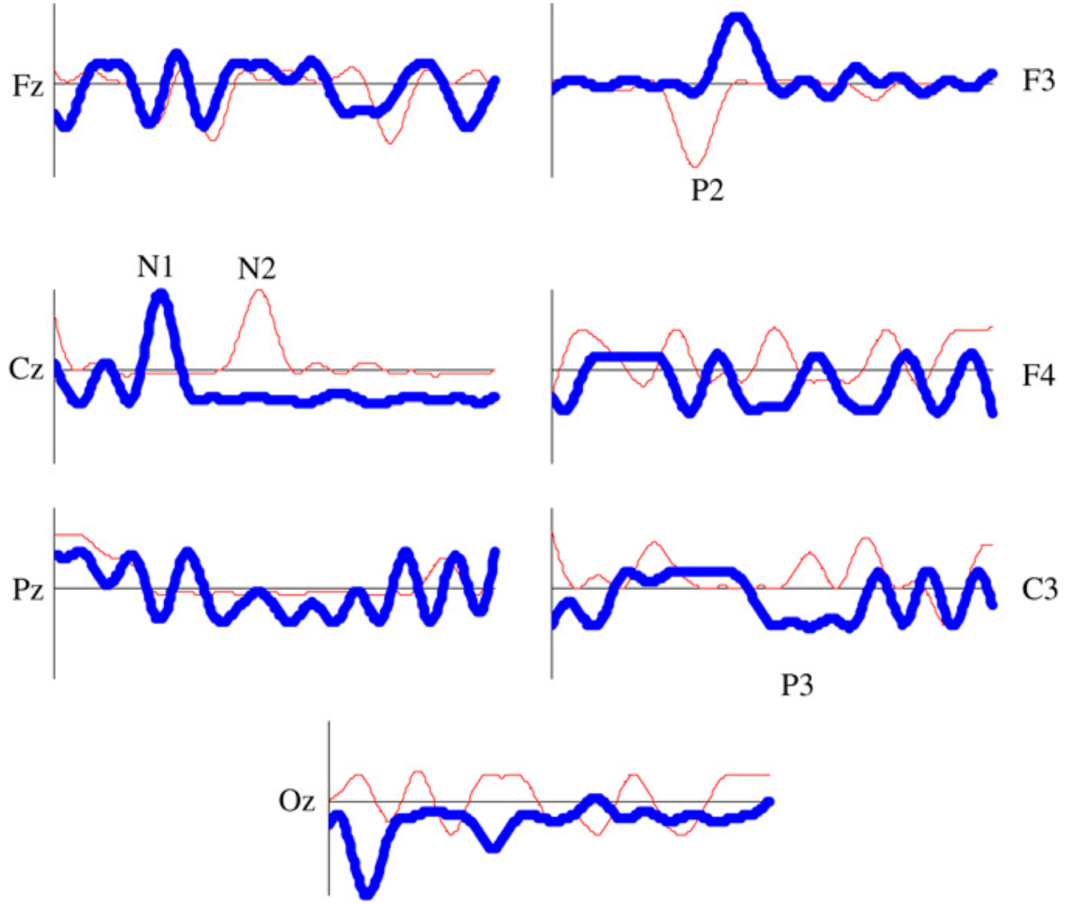
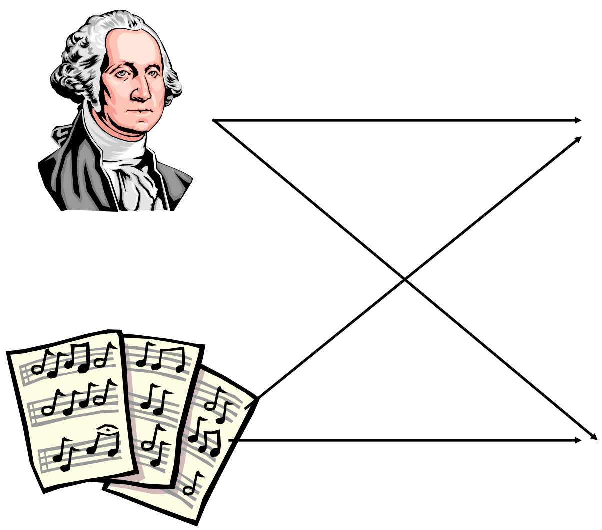
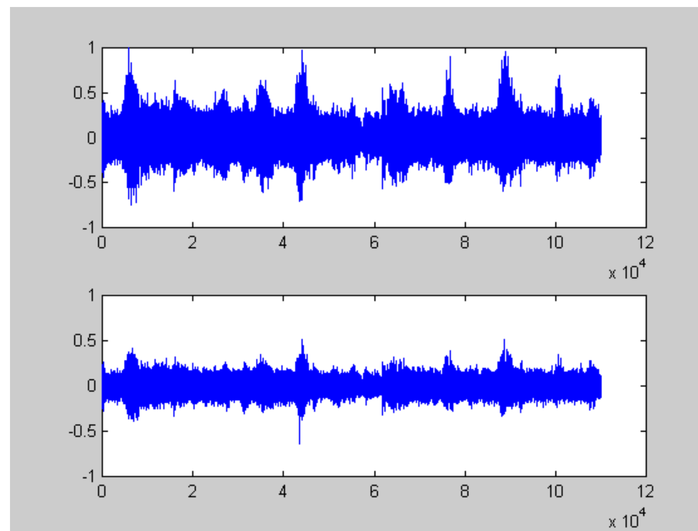


Fig. 12. Independent components obtained by AemICA for blind separation of ERPs.

Blind source separation by convolutive ICA

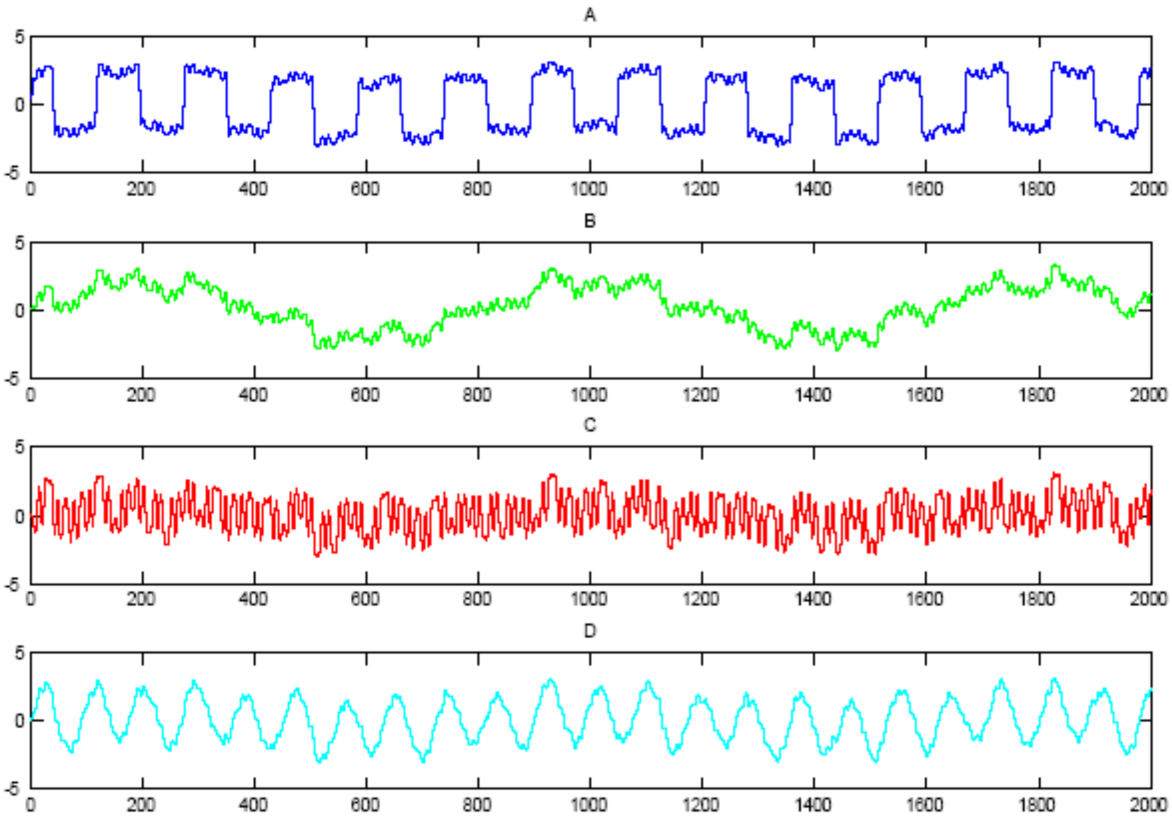
music and speech



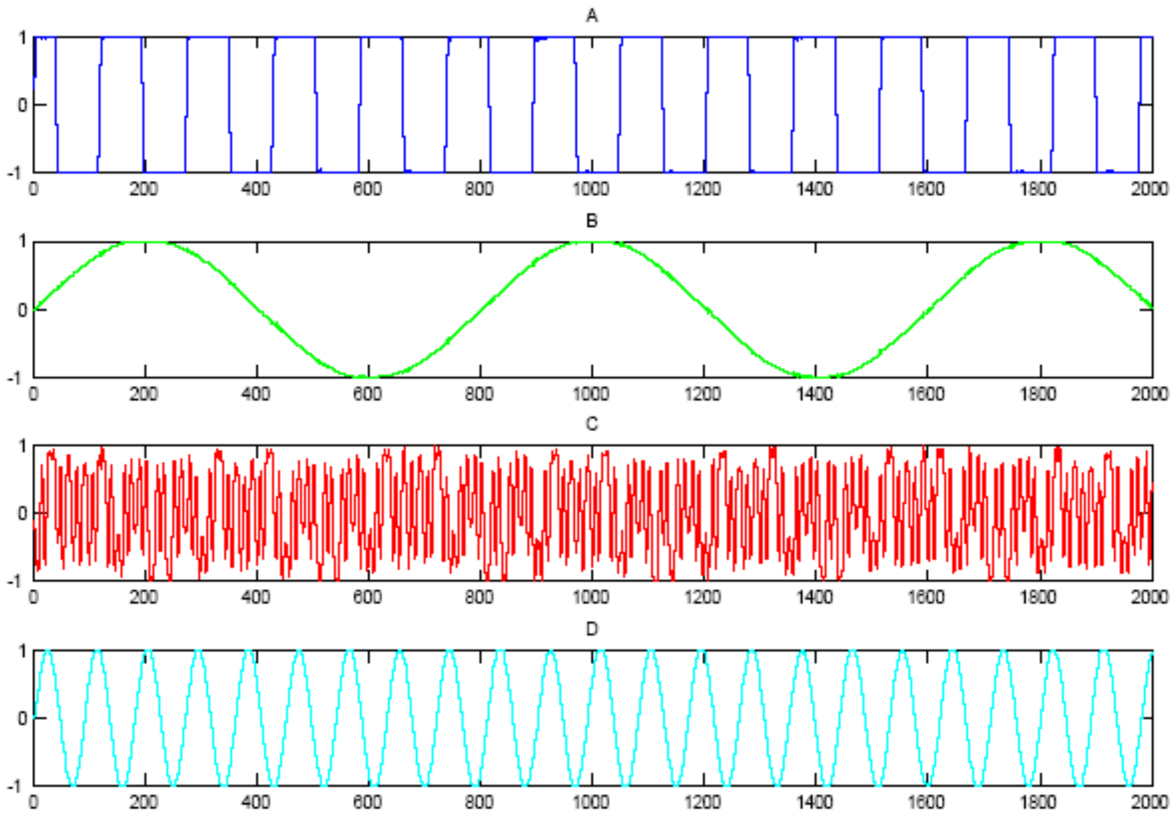


Convolutive mixtures

$\tau = 5$



Recovered sources



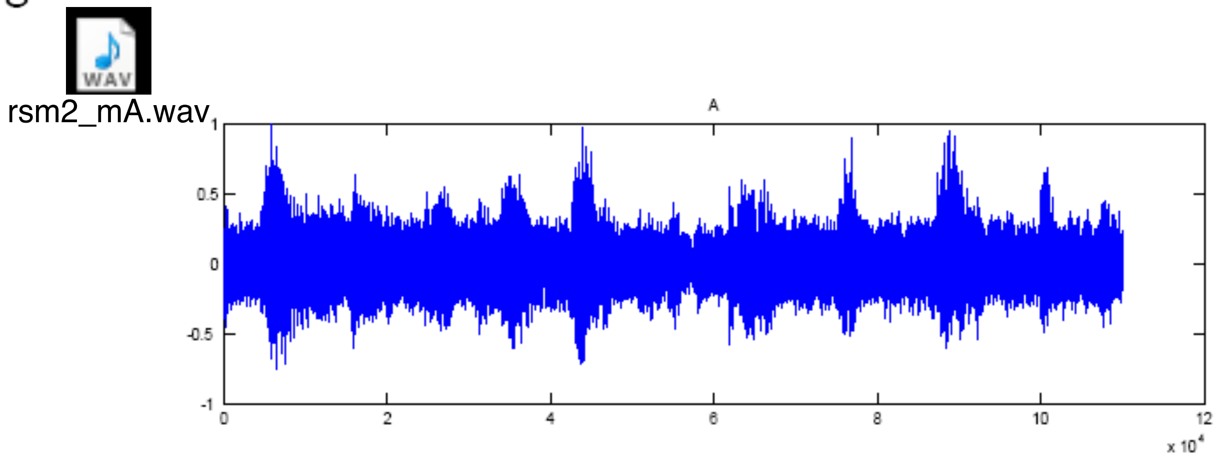
Blind separation of real world signals

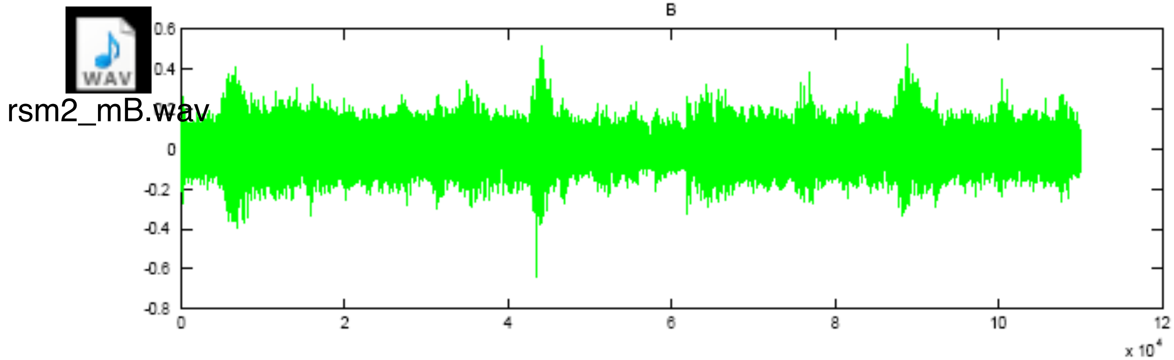
The experiment results as follows:

- Two-microphone recordings of music and speech.
  - Channel-1 Sound
  - Channel-2 Sound
- Blind separation of recordings of music and speech.
  - Channel-1 Sound
  - Channel-2 Sound

Recording of two microphones

The recordings of two microphones are shown in the following figure.

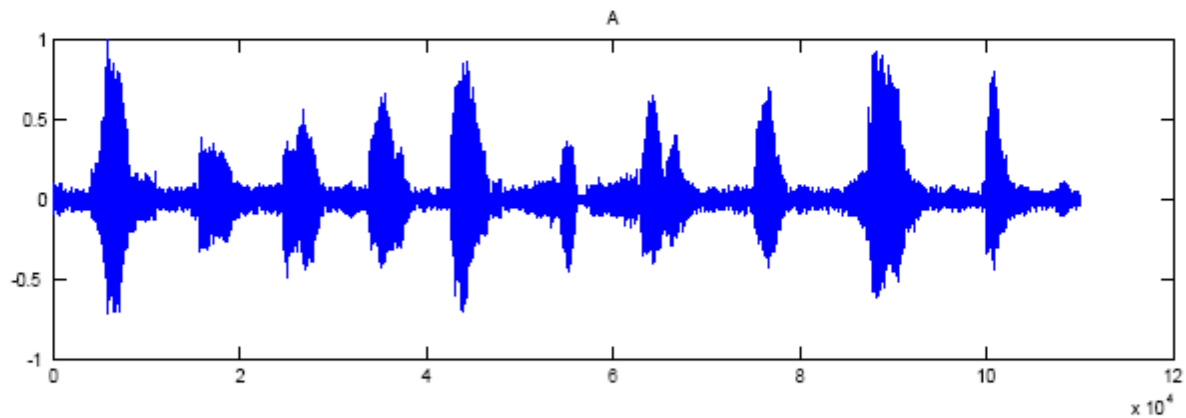




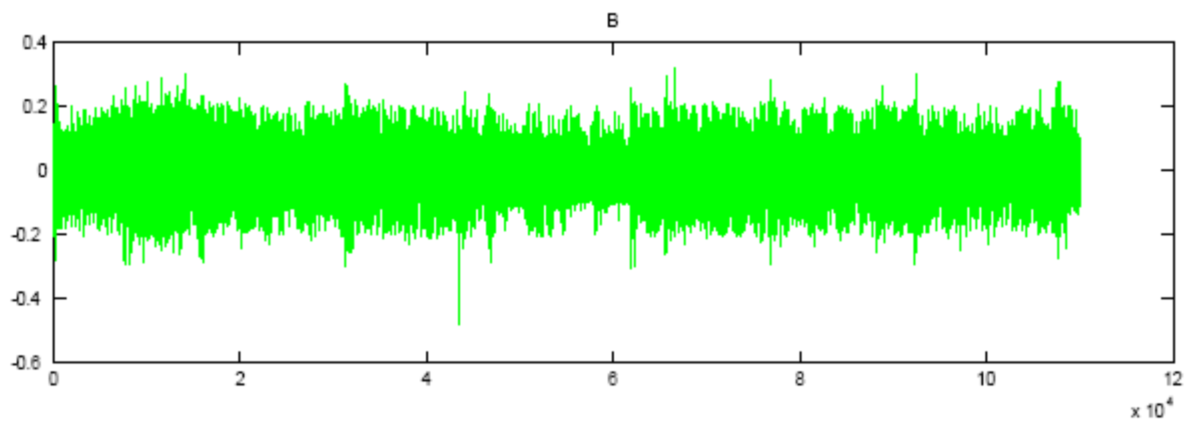
BSS

The blind separation of music and speech are shown in the following figure.

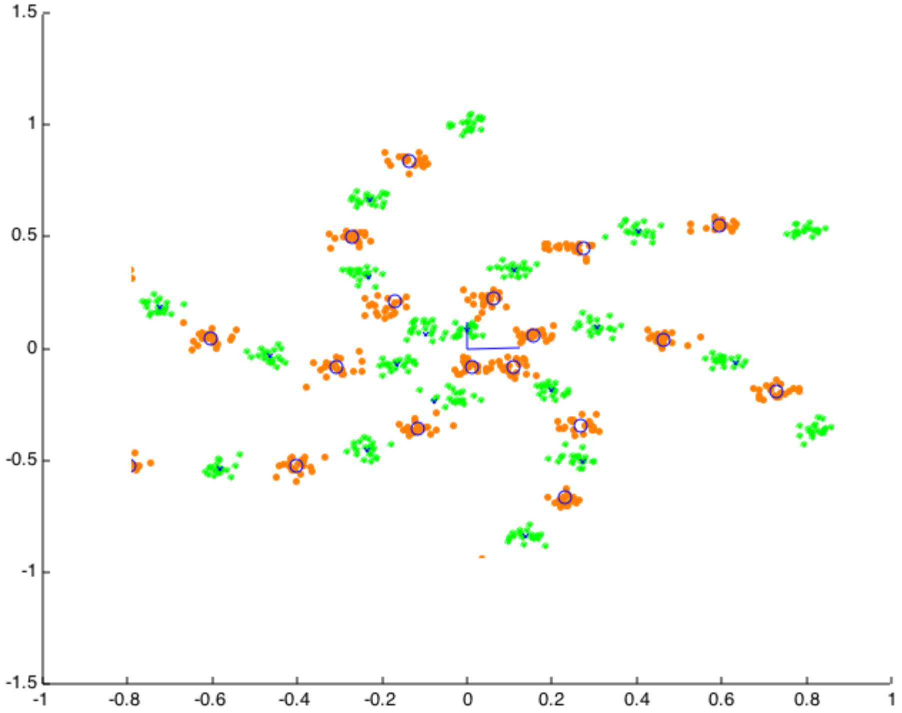
ic_a1.wav



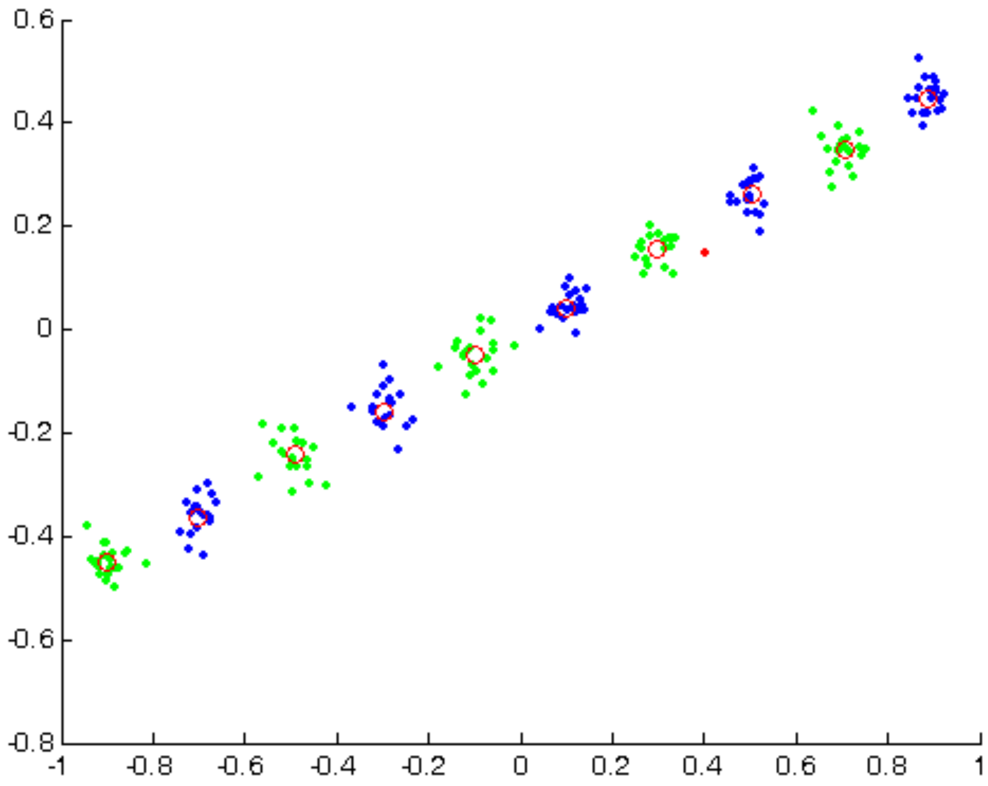
ic_b1.wav

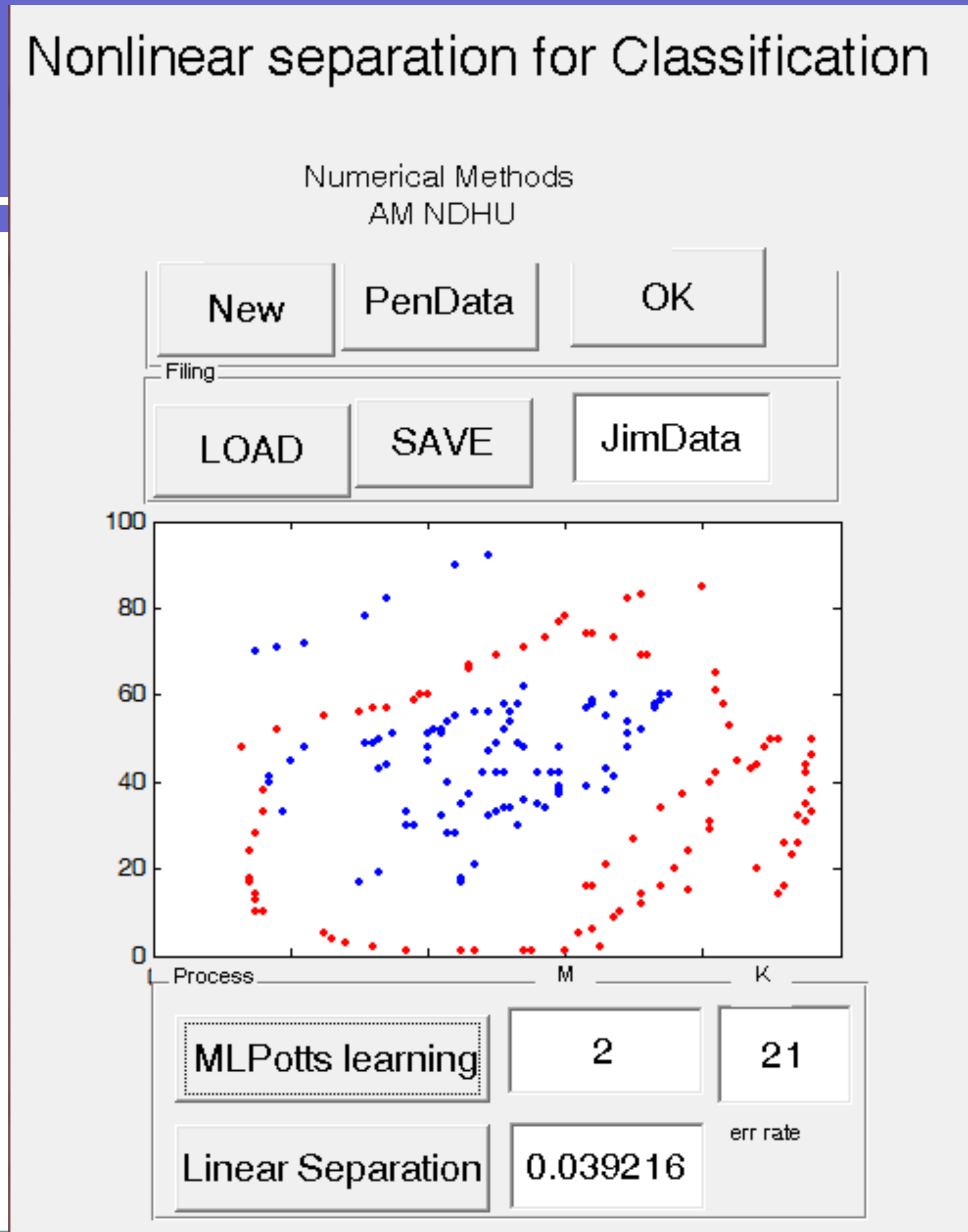


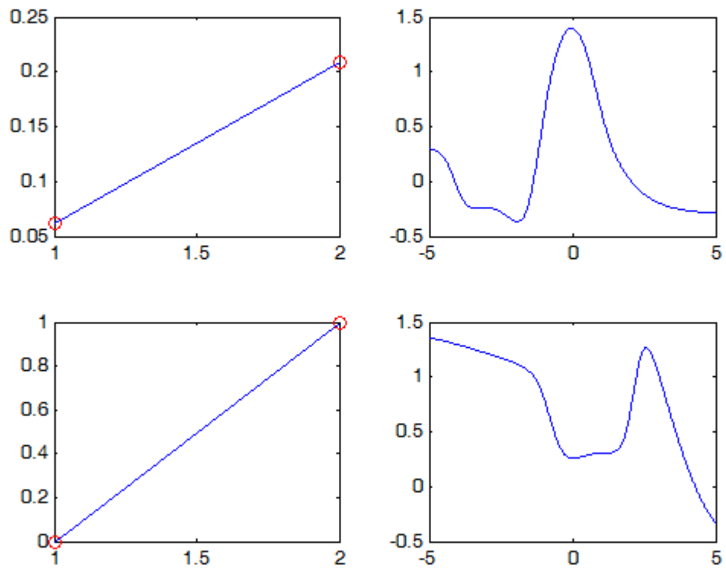
Classification (Discriminate analysis

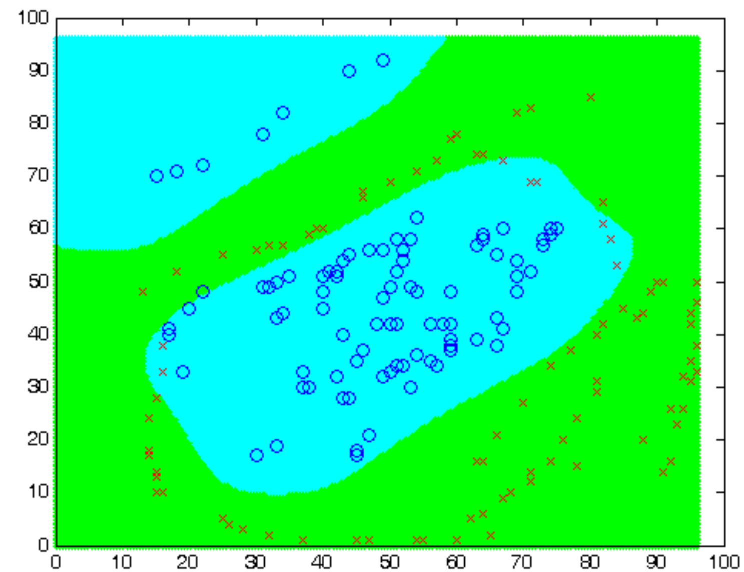


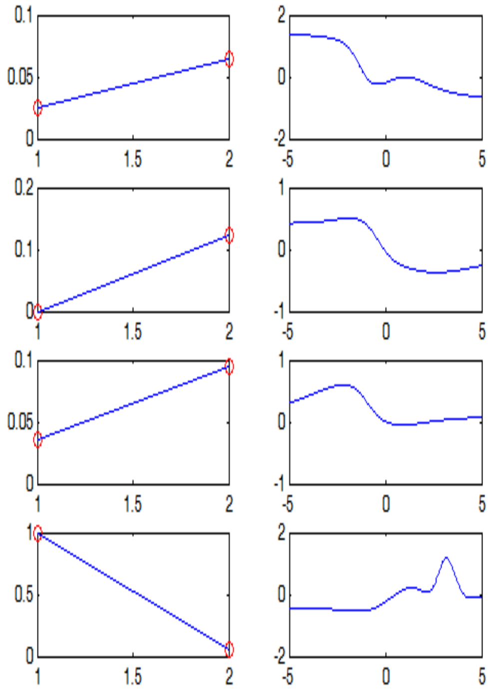
Example : DA



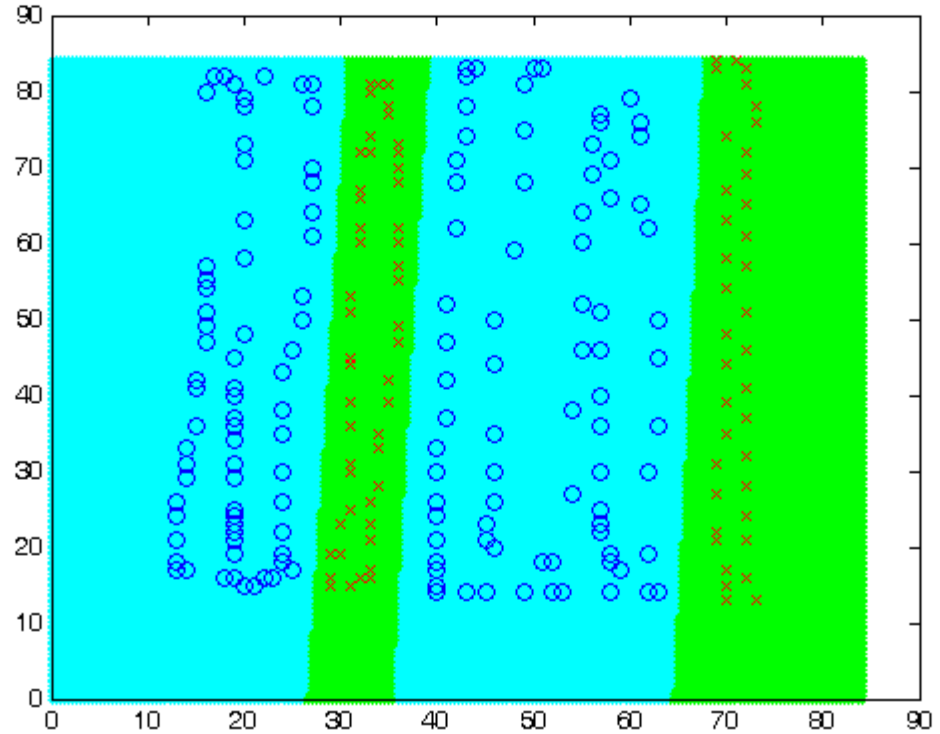

Nonlinear separation for Classification
Numerical Methods
AM NDHU
New
PenData
OK
Filing
LOAD
SAVE
JimData
100
80
60
40
20
0
Process
M
K
MLPotts learning
2
21
err rate
Linear Separation
0.039216


0.25
0.2
0.15
0.1
0.05
1
1.5
2
1.5
1
0.5
0
-0.5
-5
0
5
1
0.8
0.6
0.4
0.2
0
1
1.5
2
1.5
1
0.5
0
-0.5
-5
0
5


100
90
80
70
60
50
40
30
20
10
0
0
10
20
30
40
50
60
70
80
90
100



0.1
0.05
0
1
1.5
2
2
0
-2
-5
0
5
0.2
0.1
0
1
-1
0.1
0.05
1
0
-1
1
0.5
0
2
0
-2



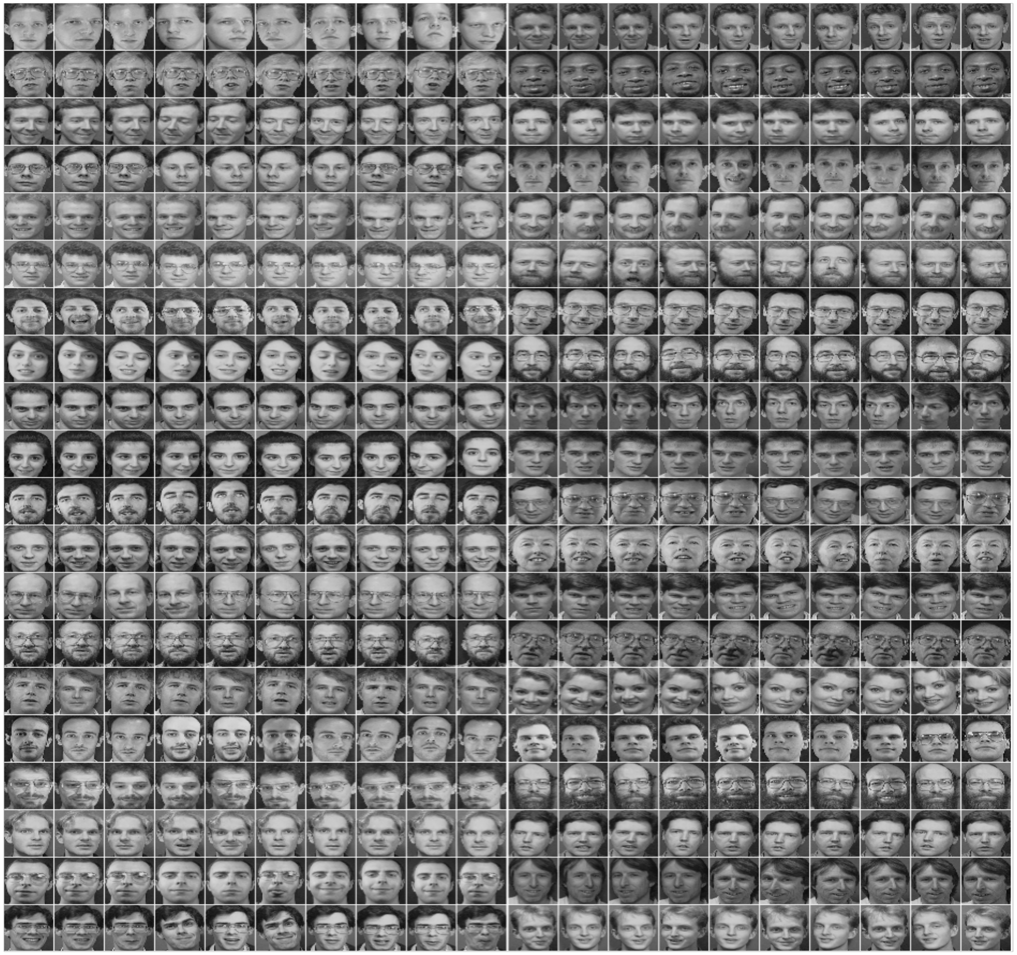
90
80
70
60
50
40
30
20
10
0
0
10
20
30
40
50
60
70
80
90

Classification- Face detection

Classification- Face detection

Classification- Face detection

Classification- Face recognition



Classification – breast cancer diagnosis