Neural Network
Lecture 3

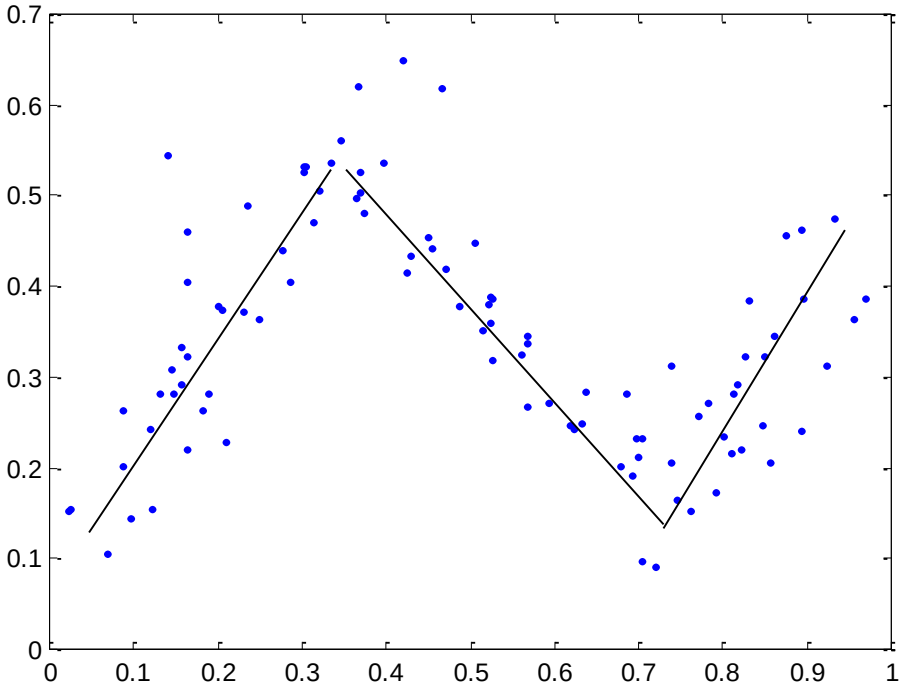
Jeong-Yean Yang

2020/10/22

1 Neural Network Basic Questions Before Learning it

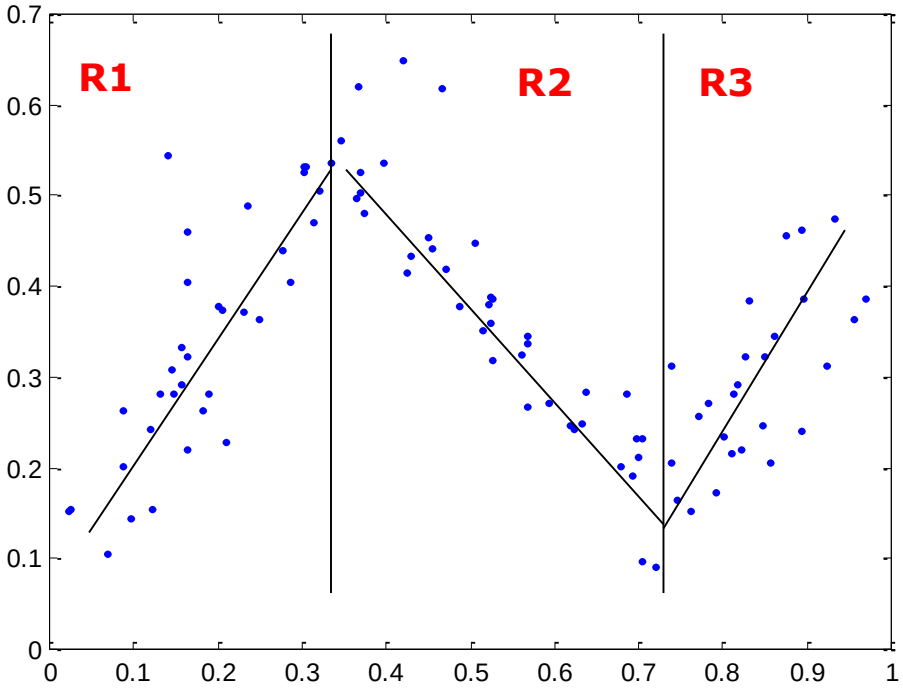
We Learn Regression Model

Question: How we do regression for Next Data?

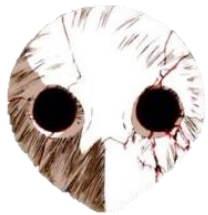


- It is NOT a linear and It is NOT a squared function
- It looks like a sine function but it is NOT.
- How we do? → **Multiple Lines?**

If we divide Three Ranges, We use Regression



???

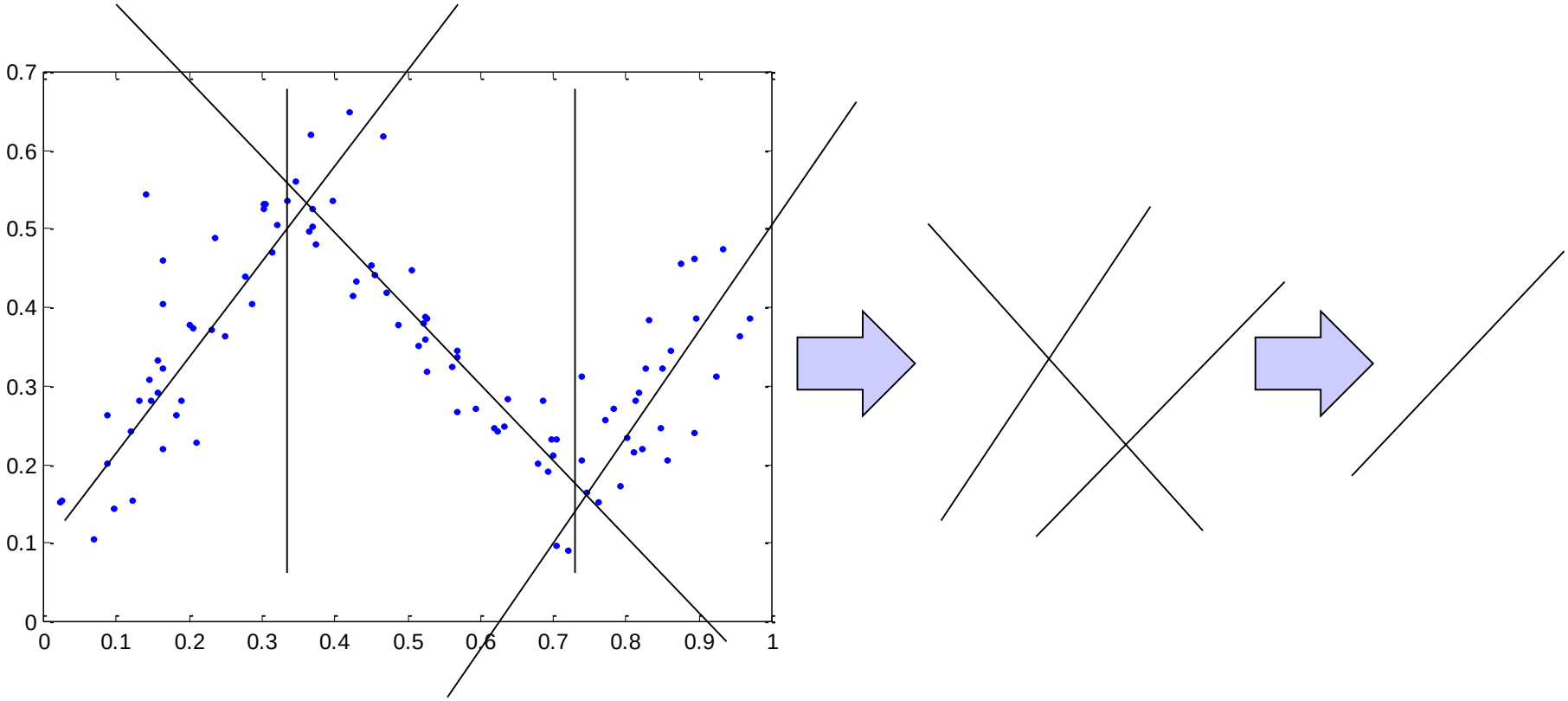


$$J = \sum_{i}^{N} e_i^{\,2} = \sum_{i \in R1}^{N(R1)} \| y_i - (a_1 x_i + b_1) \|^2 + \sum_{i \in R2}^{N(R2)} \| y_i - (a_2 x_i + b_2) \|^2 + \sum_{i \in R3}^{N(R3)} \| y_i - (a_3 x_i + b_3) \|^2$$

Who Determines Three or More Region?
Three Regions are Correct?

How can we do it?

If we add all lines, is it good?



$$J = \sum_{i}^{N} e_i^{\,2} = \sum_{i \in R} \| y_i - (a_1 x_i + b_1) \|^2 + \| y_i - (a_2 x_i + b_2) \|^2 + \| y_i - (a_3 x_i + b_3) \|^2$$

The Sum of Lines is Always a Line

$$J = \sum_{i}^{N} e_i^{\,2} = \sum_{i \in R1}^{N(R1)} \| y_i - (a_1 x_i + b_1) \|^2 + \sum_{i \in R2}^{N(R2)} \| y_i - (a_2 x_i + b_2) \|^2 + \sum_{i \in R3}^{N(R3)} \| y_i - (a_3 x_i + b_3) \|^2$$

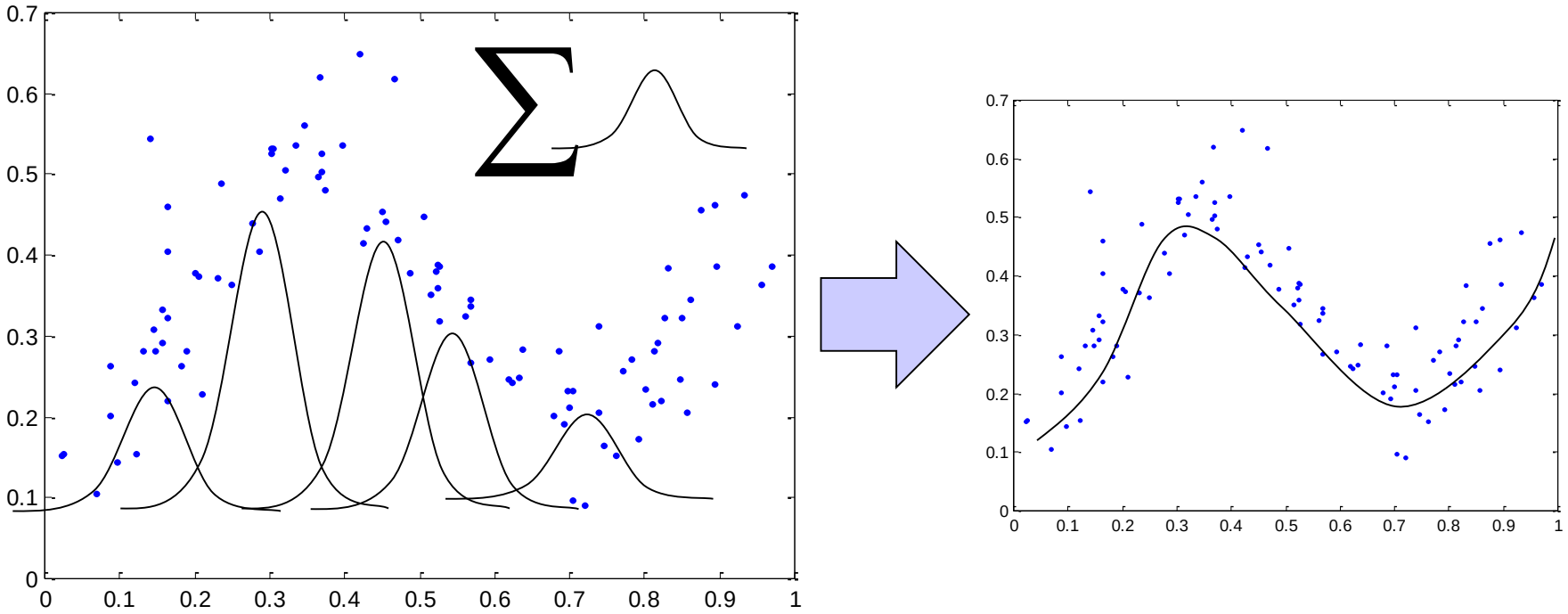
- Given condition)
  - Region,R =R1+R2+R3
  - We cannot determine Proper Region, R1, R2, and R3.
  - We must use R.
- Then, if we use ax+b for every Region, R,

$$J = \sum_{i}^{N} e_i^{\,2} = \sum_{i \in R} \| y_i - \hat{y} \|^2$$

$$\hat{y} = a_1 x + b_1 + a_2 x + b_2 + a_3 x + b_3 = a' x + b'$$

- Thus, we need another Nonlinear function for $\hat{y}$

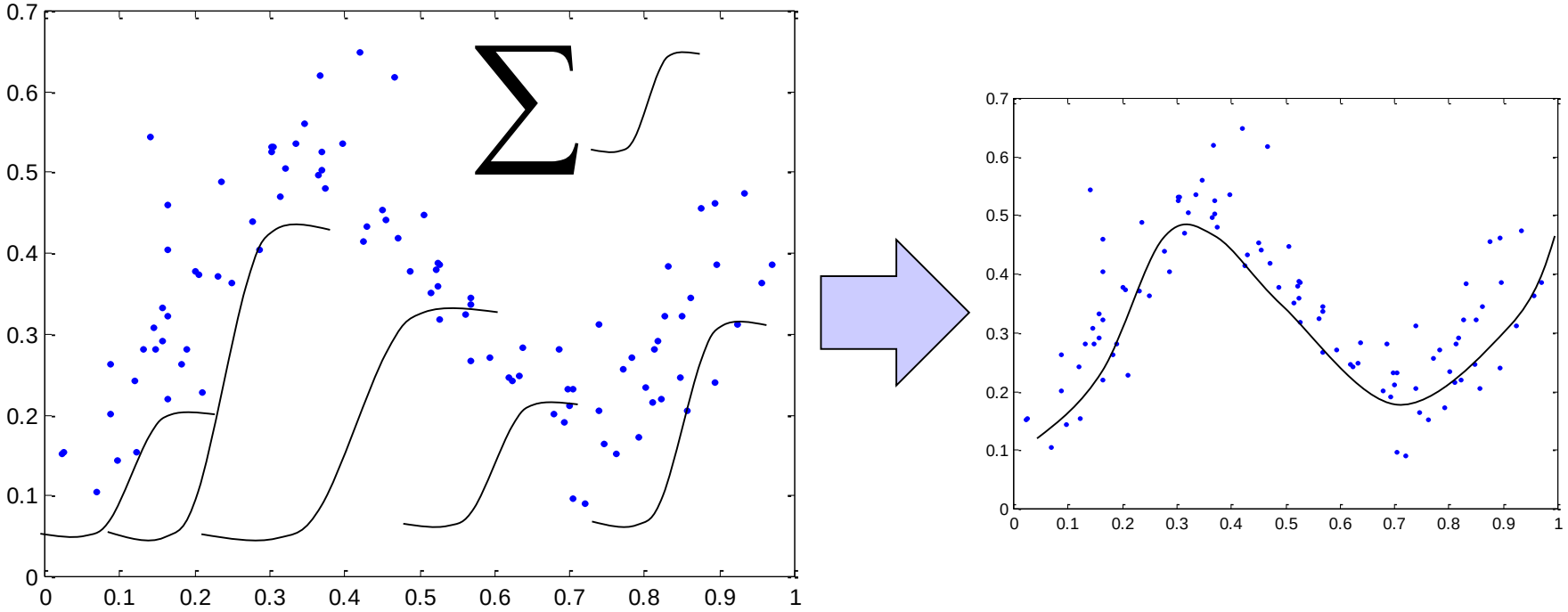
Some Function shapes like,



- Linear function cannot satisfy specific Region
- We Must use every Region → Function must not be Line

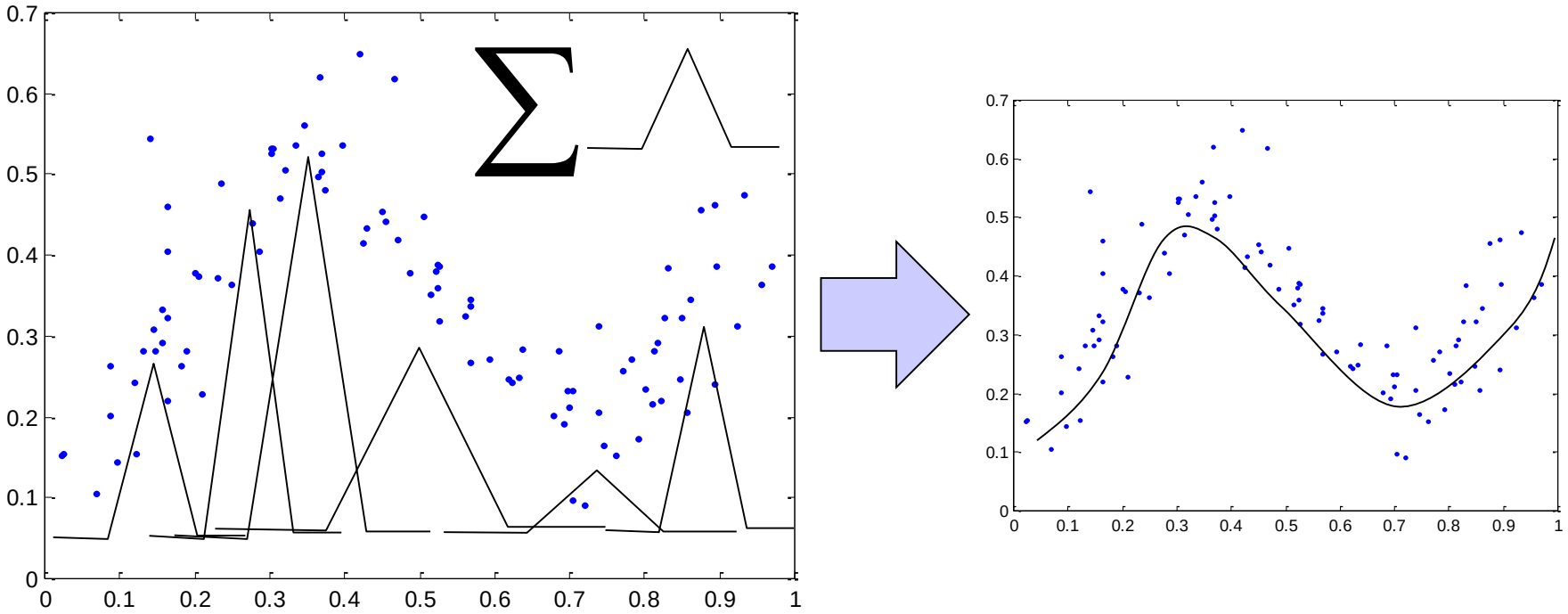
  → Non linear function

Some function shapes like



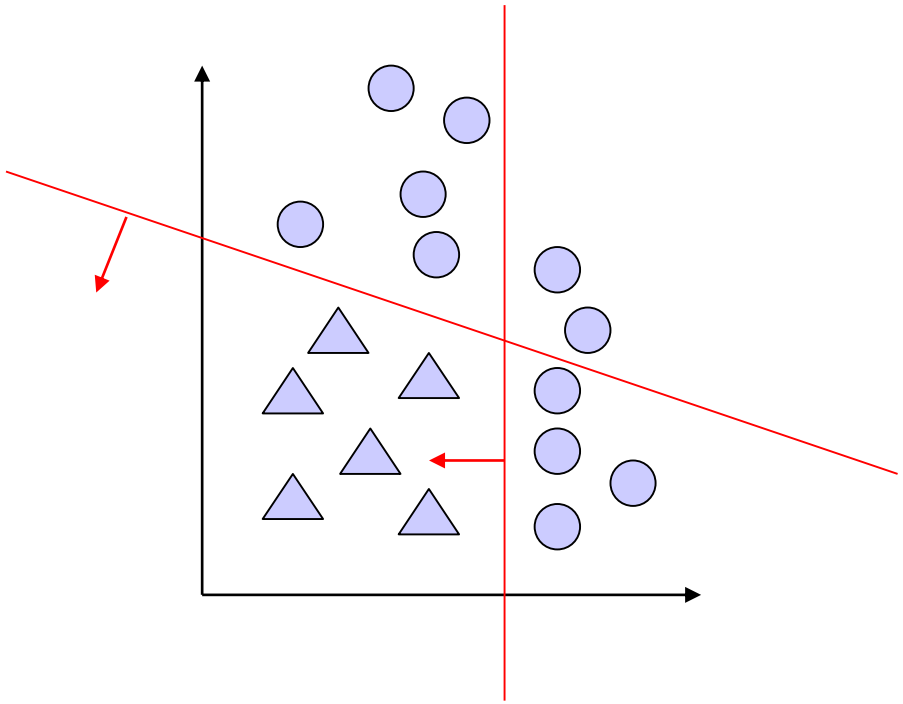
- Non linear functions are the candidate for Neural Network
- Even sine or cosine functions works.

Some function shapes like



- Triangle curves also work
- It is also non linear → Remind linearity condition
  - Triangle curve is NOT equal that av1+bv2 = v

Remind Boundary Decision with Linear Function requires Sign func.

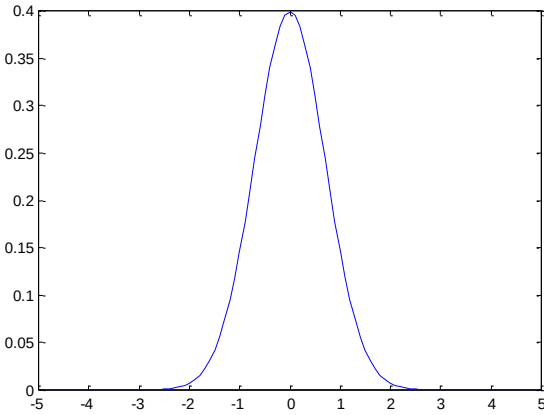


$$Y = \Phi(x) = sign(ax + b)$$

- Sign function is also a nonlinear function Phi, Φ

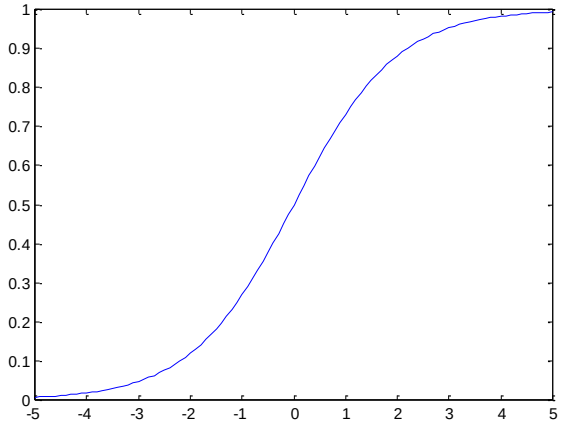
Which Nonlinear Functions are the Best for Kernel?

- No answer, Definitely



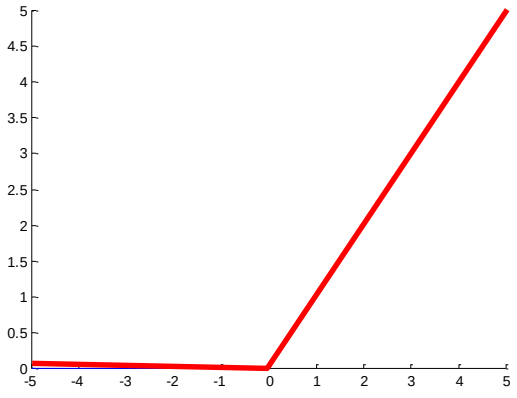
$$\Phi = RBF = \frac{1}{b}\exp\left(-\left(\frac{x-\mu}{b}\right)^2\right)$$

Radial Basis Function



$$\Phi = \frac{1}{1+e^{-x}}$$

Sigmoidal *Function*



$$\Phi = \begin{cases} 0 & x < 0 \\ ax & x \geq 0 \end{cases}$$

Re *LU Function*

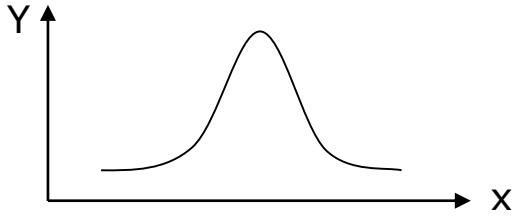
(Rectifier linear Unit)

Define Kernel Function

- These functions are called Kernel function, K

$$Y = \Phi(X)$$

$$K(X, Z) = \Phi(X)^{T}\Phi(Z) \rightarrow \text{linear}$$



X : *Input vector*

$\Phi(X)$: *Nonlinear function*

Kernel function is a dot product of map data

- All Input data are thought as the results of $\Phi(X)$
- Is it good for every cases?
  - 1. In most cases, the results are bad.

    One kernel function cannot satisfy all possible cases
  - 2. Thus, we use many kernel functions → Modern Learnings

Kernel Function K or Φ

- Definition of Kernel function is a dot product of Φ
  - But in many books kernel function K and nonlinear function Pi are in mixed usages.

- Basic of Kernel trick
  - K is a dot product of Pi's, thus K is Scalar function

$$K = \Phi_1 \square \Phi_2 = Scalar$$

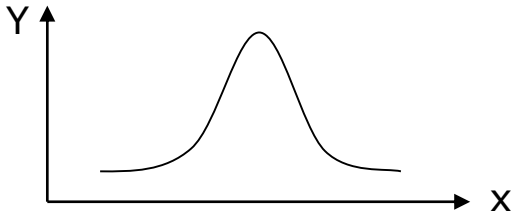
  - Nonlinear function Pi, Φ simplifies high dimensional input into unknown feature space
  - Through Non linear Pi, Kernel can simplifies and linearize a given problem.

2

Kernel Functions are thought as Nonlinear Regression

Kernel Function as Regression

- Use Some function like Gaussian function



- Gaussian function(Probabilistic Density Function)

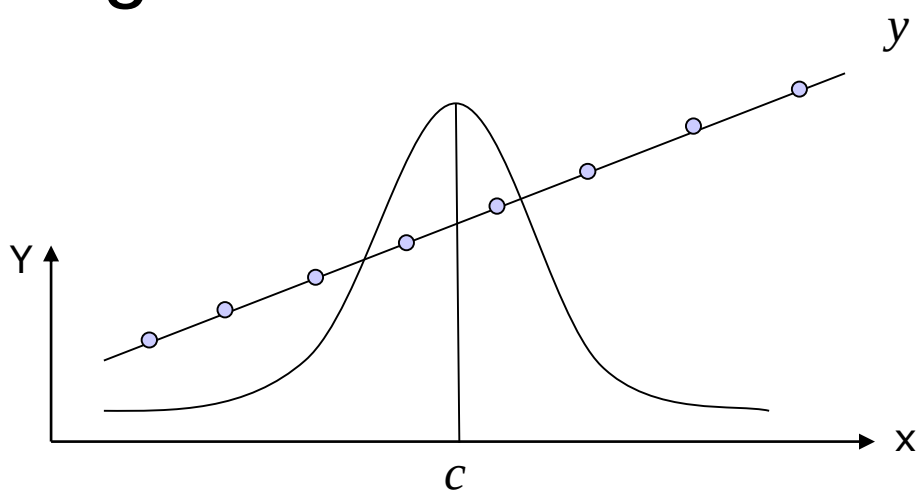
$$pdf(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma} \right)^2 \right)$$

$$x \sim N(\mu, \sigma)$$

- Radial-basis function

$$RBF(x) = \exp\left(-\frac{(x-\mu)^2}{b} \right) = \phi(x)$$

Regression with RBF 1



- Objective Function, J is,

$$J = \sum_i \left(y_i - \phi(x_i)\right)^2 = \sum_i \left(y_i - \exp\left(-\frac{\left(x_i - c\right)^2}{b} \right) \right)^2$$

$$\frac{\partial J}{\partial c} = \frac{\partial}{\partial c} \sum_i \left(y_i - \phi(x_i)\right)^2 = -2 \sum_i \left(y_i - \phi(x_i)\right) \frac{\partial \phi(x_i)}{\partial c}$$

$$= -4 \sum_i \left(y_i - \exp\left(-\frac{\left(x_i - c\right)^2}{b} \right) \right) \exp\left(-\frac{\left(x_i - c\right)^2}{b} \right) \frac{\left(x_i - c\right)}{b}$$

Tip for operation like (x-b)^2

- X=linspace(s,t,number)
  - Ex) x= linspace(0,5,6) =[0,1,2,3,4,5]
- Matrix multiplication has two types.

$$AB = \begin{bmatrix} a & b \\ c & d \end{bmatrix}\begin{bmatrix} a' & b' \\ c' & d' \end{bmatrix} = \begin{bmatrix} aa'+bc' & ab'+bd' \\ ca'+dc' & cb'+dd' \end{bmatrix}$$ General Matrix multiplication

$$A \circ B = \begin{bmatrix} a & b \\ c & d \end{bmatrix}\begin{bmatrix} a' & b' \\ c' & d' \end{bmatrix} = \begin{bmatrix} aa' & bb' \\ cc' & dd' \end{bmatrix}$$ Hadamard Product

- Matlab has two operations
  - Matrix = A*B Hadarmard Product = A.*B
- loop.sys also has two operations
  - Matrix = A*B Hadarmard Product = A.mul(B)

ex/ml/l6rbf1

```
for it in range(0,100):
    p = exp(-(x-c).mul(x-c)/b)

    # Draw current graph
    figure(1)
    graph(2)
    clear(2)
    plot(x,p,'r')

    # get error
    e = y-p
    dJ_dc= -4*sum(e.mul(p).mul(x-c)/b)

    # GDM
    c = c -a*dJ_dc
    e2= e*e.T()
    print(e2)

    # Draw error
    figure(2)
    plot(it,e2);
```

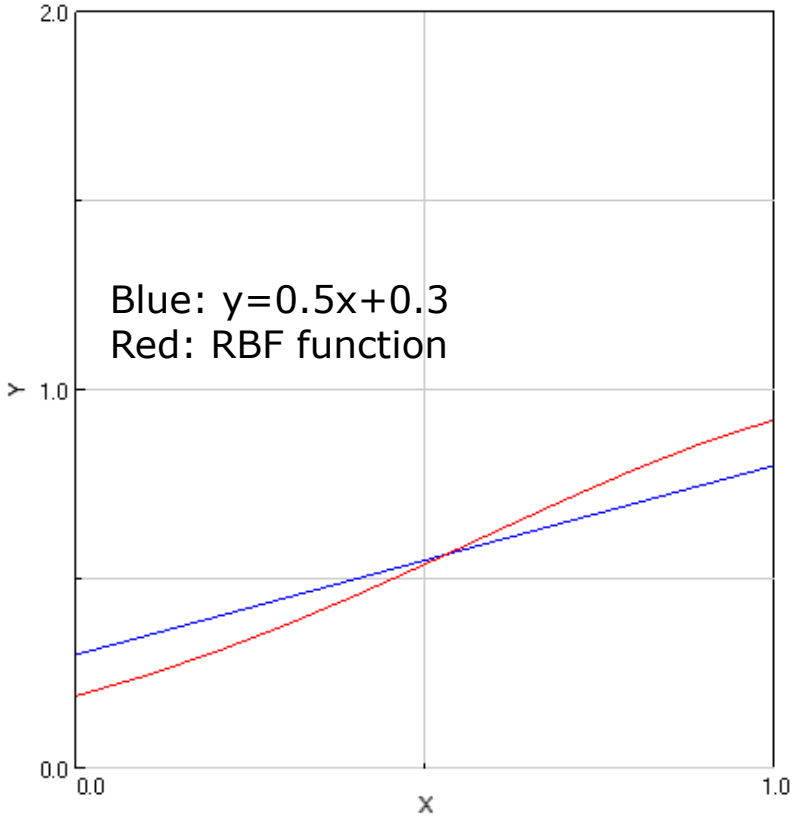
$$J = \sum_i \left(y_i - \phi(x_i)\right)^2 = \sum_i \left(y_i - \exp\left(-\frac{\left(x_i - c\right)^2}{b} \right) \right)^2$$

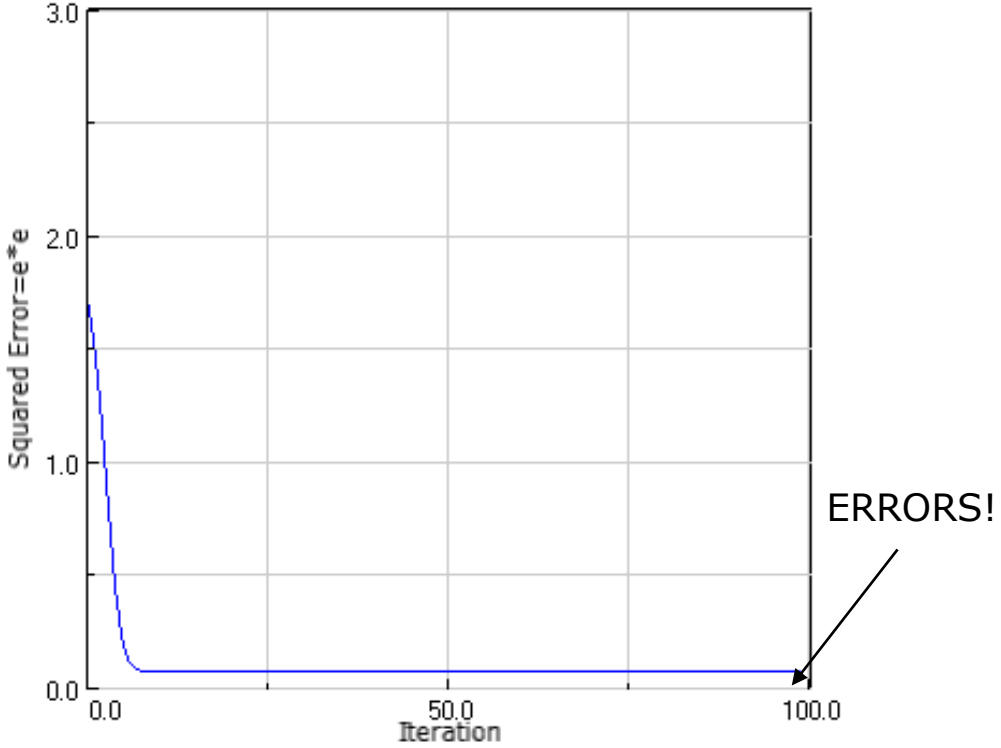
$$e_i = y_i - \phi(x_i)$$

$$\frac{\partial J}{\partial c} = -4 \sum_i \left(y_i - \exp\left(-\frac{\left(x_i - c\right)^2}{b} \right) \right) \exp\left(-\frac{\left(x_i - c\right)^2}{b} \right) \frac{\left(x_i - c\right)}{b}$$

$$= -4 \sum_i \left(e_i\right) \phi(x_i) \frac{\left(x_i - c\right)}{b}$$

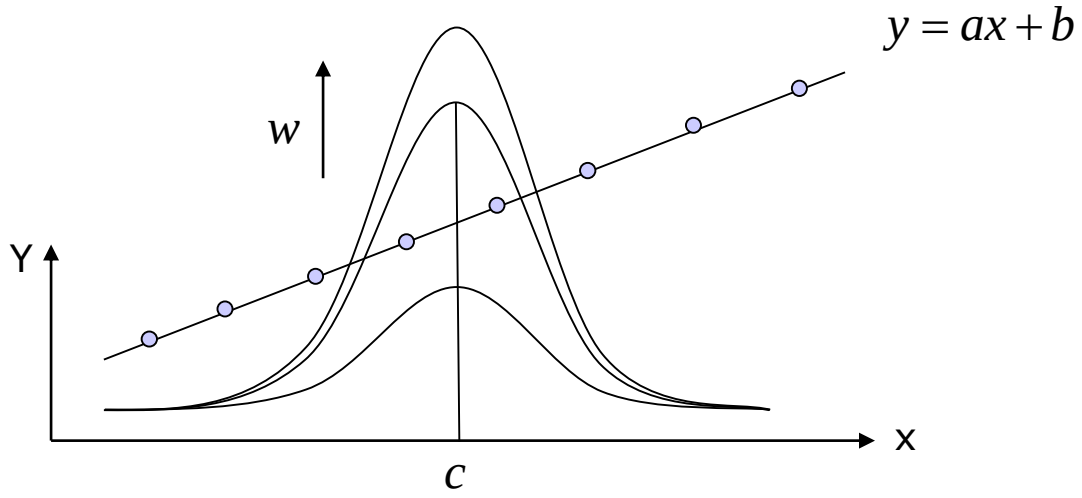
Ex/ml/l6rbf1





- c is the initial center of RBF.
- The result differs with various Initial guess of C.
  - Guess C = 0, 0.5, 1, and so on.

Regression with RBF 2



- Objective Function, J is,

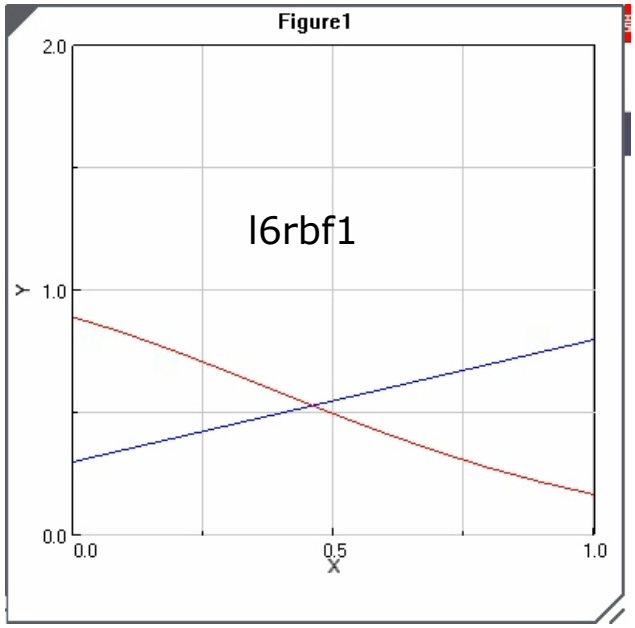
$$J = \sum_i \left(y_i - w \cdot \phi(x_i)\right)^2 = \sum_i \left(y_i - w \exp\left(-\frac{\left(x_i - c\right)^2}{b} \right) \right)^2$$

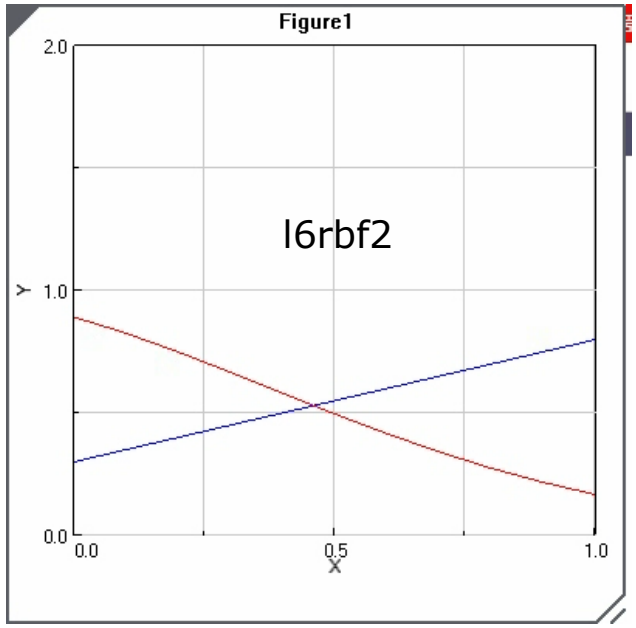
$$\frac{\partial J}{\partial w} = \frac{\partial}{\partial c} \sum_i \left(y_i - w\phi(x_i)\right)^2 = -2 \sum_i \left(y_i - w\phi(x_i)\right) \phi(x_i)$$

$$\frac{\partial J}{\partial c} = \frac{\partial}{\partial c} \sum_i \left(y_i - w\phi(x_i)\right)^2 = -4 \sum_i \left(y_i - w\phi(x_i)\right) w\phi(x_i) \frac{\left(x_i - c\right)}{b}$$

Ex/ml/l6rbf2

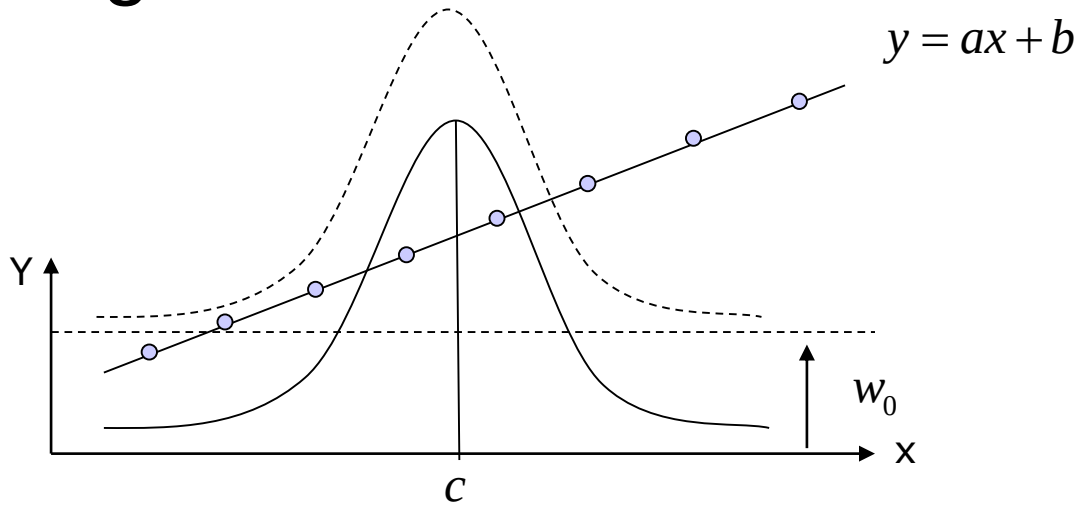
- Result Comparison
  - Use guess, c= 0





- Weight, w works for the better estimation!!

Regression with RBF 3



- Objective Function, J is,

$$J = \sum_i \left(y_i - w \cdot \phi(x_i) - w_0 \right)^2 = \sum_i \left(y_i - w \exp\left(-\frac{(x_i - c)^2}{b} \right) - w_0 \right)^2$$

$$\frac{\partial J}{\partial w} = \frac{\partial}{\partial c} \sum_i \left(y_i - w\phi(x_i) - w_0 \right)^2 = -2 \sum_i \left(y_i - w\phi(x_i) - w_0 \right) \phi(x_i)$$

$$\frac{\partial J}{\partial w_0} = \frac{\partial}{\partial c} \sum_i \left(y_i - w\phi(x_i) - w_0 \right)^2 = -2 \sum_i \left(y_i - w\phi(x_i) - w_0 \right) 1$$

$$\frac{\partial J}{\partial c} = \frac{\partial}{\partial c} \sum_i \left(y_i - w\phi(x_i) - w_0 \right)^2 = -4 \sum_i \left(y_i - w\phi(x_i) - w_0 \right) w\phi(x_i) \frac{(x_i - c)}{b}$$

Ex/ml/l6rbf3

```
for it in range(0,500):
    p = exp(-(x-c).mul(x-c)/b)

    # Draw current graph
    figure(1)
    graph(2)
    clear(2)
    plot(x,p,'r')

    # get error
    e = y-w*p-w0
    dJ_dw= -2*sum(e.mul(p))
    dJ_dw0= -2*sum(e)
    dJ_dc= -4*sum(e.mul(w*p).mul(x-c)/b)

    # GDM
    w = w -a*dJ_dw
    w0= w0-a*dJ_dw0
    c = c -a*dJ_dc
    e2= e*e.T()
    print(e2)

    # Draw error
    figure(2)
    plot(it,e2);
```

$$J = \sum_i \left(y_i - w \cdot \phi(x_i) - w_0 \right)^2 = \sum_i \left(y_i - w \exp\left(-\frac{(x_i - c)^2}{b} \right) - w_0 \right)^2$$

$$\frac{\partial J}{\partial w} = \frac{\partial}{\partial c} \sum_i \left(y_i - w\phi(x_i) - w_0 \right)^2$$

$$= -2 \sum_i \left(y_i - w\phi(x_i) - w_0 \right) \phi(x_i)$$

$$\frac{\partial J}{\partial w_0} = \frac{\partial}{\partial c} \sum_i \left(y_i - w\phi(x_i) - w_0 \right)^2$$

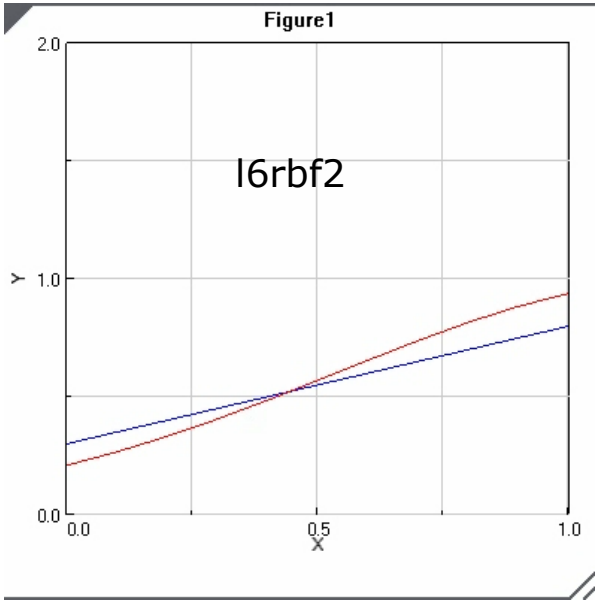
$$= -2 \sum_i \left(y_i - w\phi(x_i) - w_0 \right) 1$$

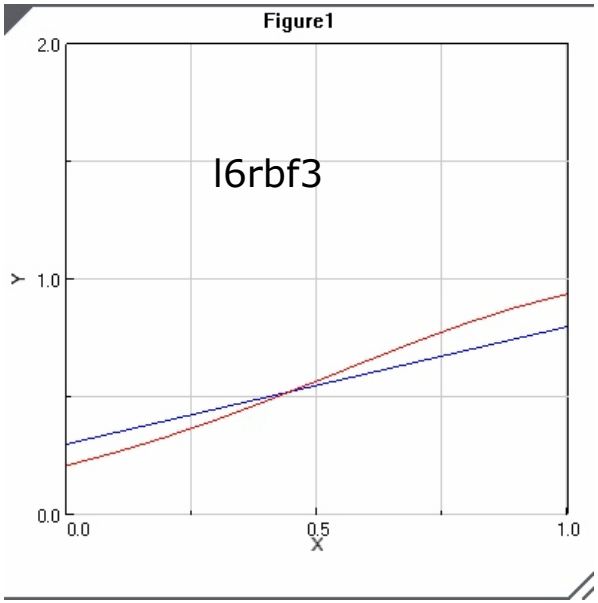
$$\frac{\partial J}{\partial c} = \frac{\partial}{\partial c} \sum_i \left(y_i - w\phi(x_i) - w_0 \right)^2$$

$$= -4 \sum_i \left(y_i - w\phi(x_i) - w_0 \right) w\phi(x_i) \frac{(x_i - c)}{b}$$

Comparison between ex/ml/l6rbf2 and ex/ml/l6rbf3

- Result Comparison
  - Use guess, c= 1





- Both weight, w and bias w0 work for the better estimation.

Definition of Weight and Bias Parameter

- Only Kernel function

$$J = \sum_i \left(y_i - \phi(x_i) \right)^2$$

- Use Weight, w

$$J = \sum_i \left(y_i - w\phi(x_i) \right)^2$$

- Use Bias, w0

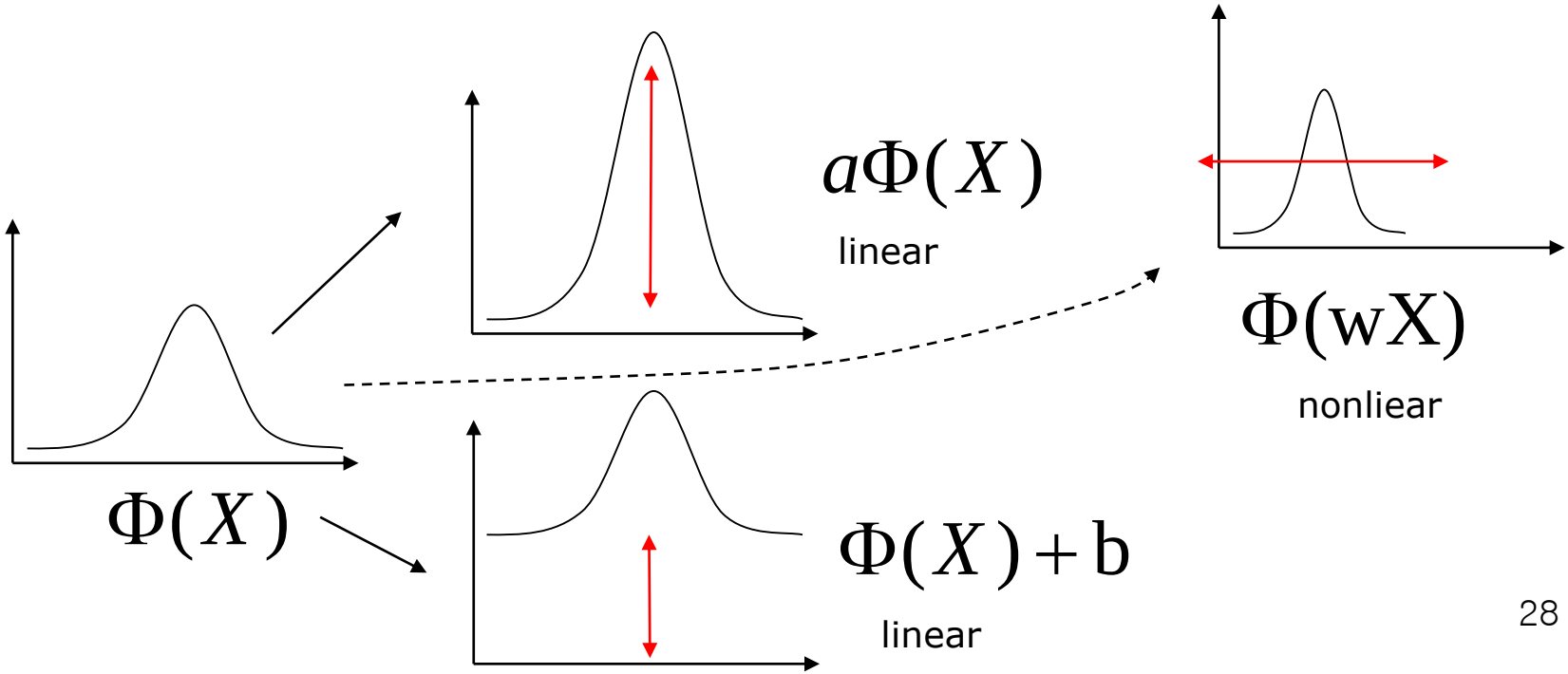
$$J = \sum_i \left(y_i - \left(w\phi(x_i) + w_0 \right) \right)^2$$

3 Neural Network Equation

Weight

- Non linear function Phi, Φ strengthens some value in every input space

- Weight is a linear combination of Phi, Φ



Weight

- Linear weight $Y = a\Phi(X) + b = W\Phi(X)$
- Nonlinear weight $Y = \Phi(wX)$

- Linear weight can be simplified as in Homogeneous Transform

$$Y = a\Phi(X) + b = \begin{bmatrix} a & b \end{bmatrix} \begin{bmatrix} \Phi(X) \\ 1 \end{bmatrix} = W \begin{bmatrix} \Phi(X) \\ 1 \end{bmatrix}$$

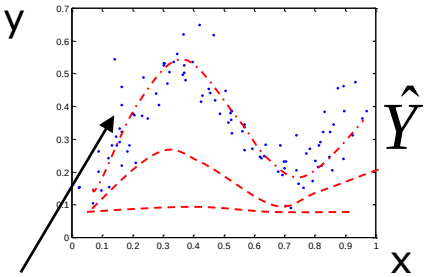
Example

$$Y = \begin{pmatrix} Y_1 \\ Y_2 \end{pmatrix} = \begin{pmatrix} a_1\Phi(\mathrm{X}_1) \\ a_2\Phi(\mathrm{X}_2) \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} = \begin{bmatrix} a_1 & 0 & b_1 \\ 0 & a_2 & b_2 \end{bmatrix} \begin{bmatrix} \Phi(\mathrm{X}_1) \\ \Phi(\mathrm{X}_2) \\ 1 \end{bmatrix} \square\ W\Phi(\mathrm{X})$$

Define Discrimination Function

- Input X
- Output Y:
  - During learning, output of $Y = W\Phi(X)$ is NOT well learned
  - → Approximation of Y = $\hat{Y}$ $\therefore \hat{Y} = W\Phi(X)$

- Weight : our goal parameter
  - like a and b in linear regression
- Cost function, J is a function of Error



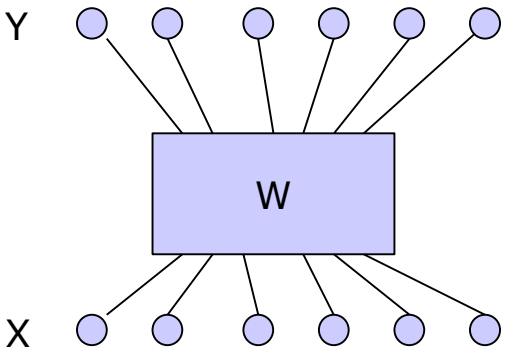
$$J = \sum_i e_i^{\,2} = \sum_{i \in R} \| y_i - \hat{y} \|^2 \longrightarrow \text{Minimization}$$

$$\nabla J = \frac{\partial J}{\partial W} \longrightarrow \text{Gradient Descent Method}$$

$$W' \leftarrow W - \alpha \nabla J$$

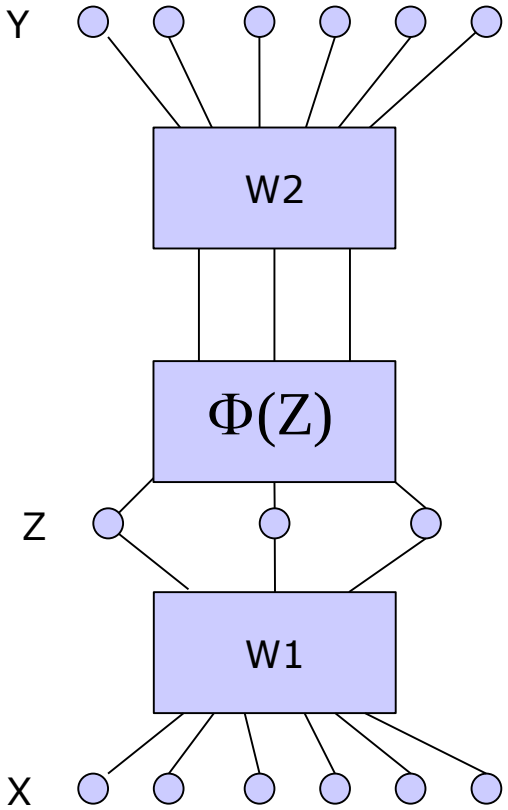
Neural Network Layer

- Input=X and Output= Y
- More Layers are used for creation of hyper space



$$Y = \begin{pmatrix} 1 \\ 4 \\ 9 \end{pmatrix} = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0.2 & 0.1 & 1 & 0.6 \\ 3 & 1 & 1 & 1 \end{bmatrix} \begin{pmatrix} X = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \\ 1 \end{pmatrix}$$

Linear $Y = WX$

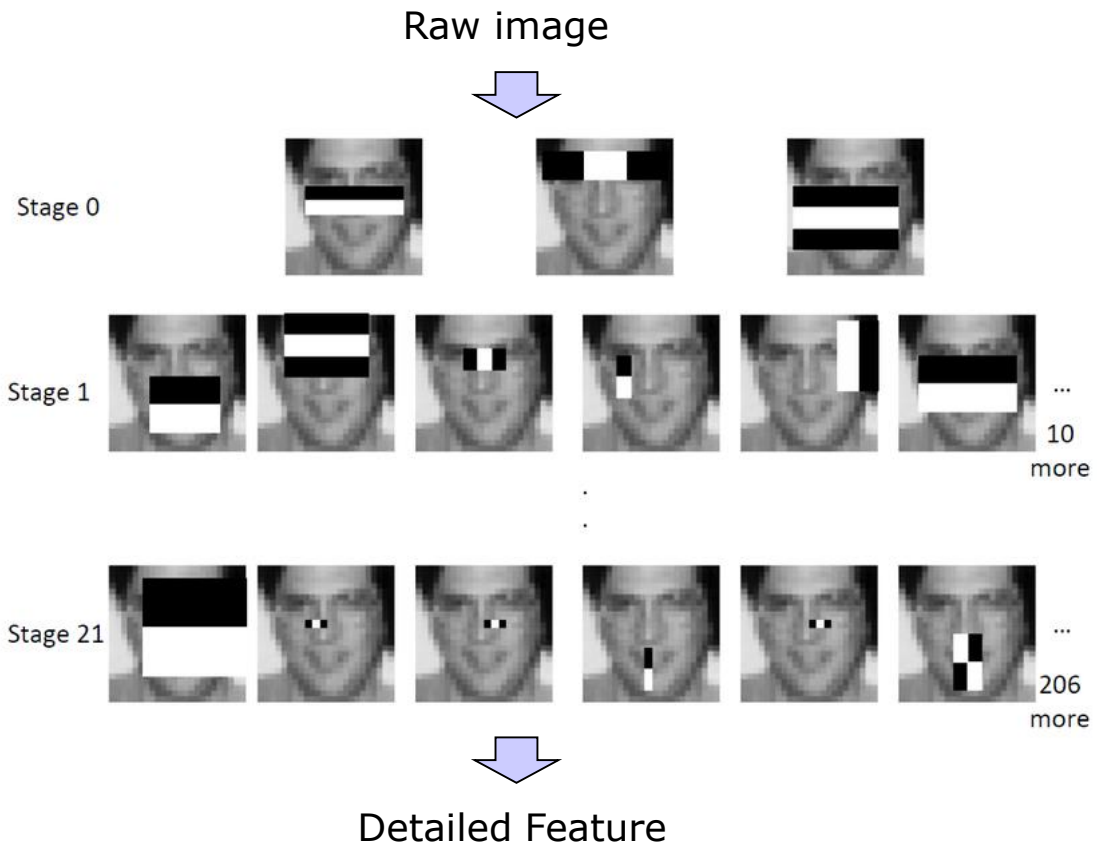


$$Z = W_1 X$$
$$Y = W_2 \Phi(Z)$$
$$= W_2 \Phi(W_1 X)$$

W_2 (6×3) W_1 (3×6)

Z: Hidden Layer

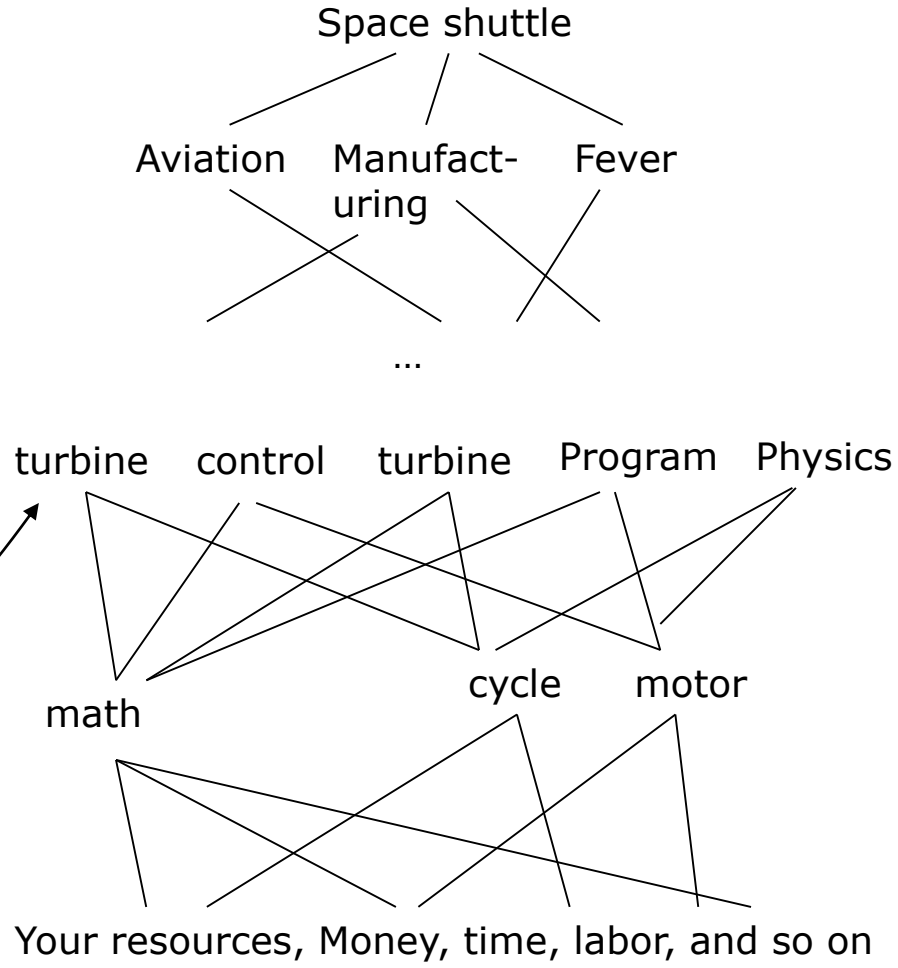
Hidden Layers maps from Lower to Higher level



- Remind
  - Face detection by Haar feature
- Stage 0 maps dominant feature
- Stage 21 maps features in more detailed ways
- Some stages(or layers) seem meaningless

→ Output is meaningful

Build Space Shuttle

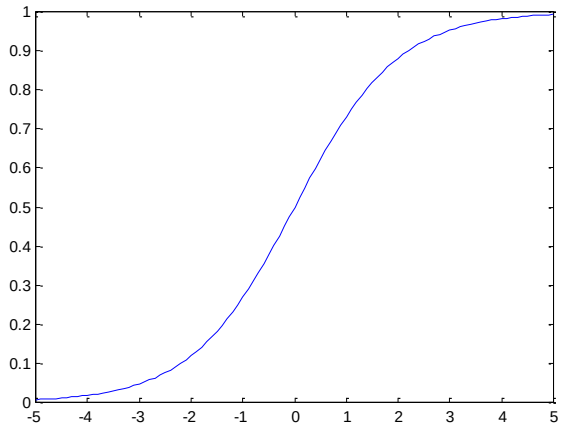


Hidden or Middle layers can be more than input or output

Sigmoidal Function-based Neural Network

$$\hat{Y} = W_2 \Phi(W_1 X)$$

$$= W_2 Z$$



Sigmoidal *Function*, $\Phi = \frac{1}{1+e^{-x}}$

$$\Phi' = \frac{e^{-x}}{\left(1+e^{-x}\right)^2} = \Phi\left(1 - \frac{1}{1+e^{-x}}\right)$$

$$= \Phi(I - \Phi)$$

$$J = \frac{1}{2}\sum_{i \in D} e_i^{\,2} = \frac{1}{2}\sum_{i \in D} \| y_i - \hat{y} \|^2$$

$$\nabla J(W_1, W_2) = \nabla J_{w_1} dw_1 + \nabla J_{w_2} dw_2$$

$$\nabla J_{w_1} = \frac{\partial J}{\partial W_1} = \sum_{i \in D} (y_i - \hat{y}) \left(0 - \frac{\partial \hat{y}}{\partial W_1} \right)$$

$$= -\sum_{i \in D} (y_i - \hat{y}) W_2 \Phi'(W_1 X)\, \mathrm{X}$$

$$\therefore W_1 \leftarrow W_1 - \alpha \nabla J_{w_1}$$

$$\nabla J_{w_2} = \frac{\partial J}{\partial W_2} = \sum_{i \in D} (y_i - \hat{y}) \left(0 - \frac{\partial \hat{y}}{\partial W_2} \right)$$

$$= -\sum_{i \in D} (y_i - \hat{y}) \Phi(W_1 X) = -\sum_{i \in D} (y_i - \hat{y}) Z$$

$$\therefore W_2 \leftarrow W_2 - \alpha \nabla J_{w_2}$$

4

Background of Vector and matrix Differentiation

Differentiation of Neural Network

- Remind Differentiation for Gradient Descent Method

$$J = \frac{1}{2}\sum_{i \in D} e_i^{\,2} = \frac{1}{2}\sum_{i \in D} \| y_i - \hat{y} \|^2$$

$$\nabla J_{w_1} = \frac{\partial J}{\partial W_1} = \sum_{i \in D} (y_i - \hat{y}) \left(0 - \frac{\partial \hat{y}}{\partial W_1} \right)$$

Vector

Matrix

$$= -\sum_{i \in D} (y_i - \hat{y}) W_2 \Phi'(W_1 X)\, \mathrm{X}$$

$$\therefore W_1 \leftarrow W_1 - \alpha \nabla J_{w_1}$$

It looks like
Matrix Multiplication

- **Question:**

  "Differentiation Vector with Matrix", Is it Possible?

Differentiation Scalar, Vector, Matrix

	Scalar y	Vector $\hat{y}$	Matrix Y
Scalar x	$\frac{\partial y}{\partial x}$	$\frac{\partial \hat{y}}{\partial x}$	$\frac{\partial Y}{\partial x}$
Vector $\hat{x}$	$\frac{\partial y}{\partial \hat{x}}$	$\frac{\partial \hat{y}}{\partial \hat{x}}$	
Matrix X	$\frac{\partial y}{\partial X}$		

- Unfortunately,

  Differentiation Matrix by Matrix is Impossible

Differentiation of Matrix 1

- Lemma 1

$$\underset{m\times 1}{y} = \underset{m\times n}{A}\ \underset{n\times 1}{x} \qquad \Rightarrow \frac{\partial y}{\partial x} = A$$

- Proof

$$y_i = \sum_{k=1}^{n} a_{ik} x_k \qquad \frac{\partial y_i}{\partial x_j} = \frac{\partial}{\partial x_j}\sum_{k=1}^{n} a_{ik} x_k = (0+0+..+\frac{\partial a_{ij} x_j}{\partial x_j}+..+0+0) = a_{ij}$$

$$\frac{\partial y_i}{\partial x_j} = a_{ij} \text{ for all i=1...m and j=1... n}$$

$$\therefore \frac{\partial y}{\partial x} = A$$

Differentiation of Matrix 2

- Lemma 2

$$\underset{m\times 1}{y} = \underset{m\times n}{A}\ \underset{n\times 1}{x} \qquad \Rightarrow \frac{\partial y}{\partial z} = \frac{\partial y}{\partial x}\frac{\partial x}{\partial z} = A\frac{\partial x}{\partial z}$$

- Lemma 3

$$\underset{m\times 1}{y} = \underset{m\times n}{A}\ \underset{n\times 1}{x}$$

$$if\ \ c = y^T Ax = y^T y = y \cdot y$$

$$c = c^T = x^T A^T y$$

$$\Rightarrow \frac{\partial c}{\partial x} = \frac{\partial y^T Ax}{\partial x} = y^T A\frac{\partial x}{\partial x} = y^T A\Big|_{1\times n}$$

$$\Rightarrow \frac{\partial c}{\partial y} = \frac{\partial y^T Ax}{\partial y} = ? \qquad \Rightarrow \frac{\partial c}{\partial y} = \frac{\partial c^T}{\partial y} = \frac{\partial x^T A^T y}{\partial y} = x^T A^T$$

Differentiation of Matrix 3

- Lemma 4

$$if \quad c = x^T A x \qquad \Rightarrow \frac{\partial c}{\partial x} = x^T A + x^T A^T$$

- Proof

$$c = \sum_i^n \sum_j^n a_{ij} x_i x_j$$

$$\therefore \frac{\partial c}{\partial x_k} = \sum_j^n a_{kj} x_j + \sum_i^n a_{ik} x_i$$

Differentiation of Matrix 4

- Lemma 5

$$y = y(z), x = x(z) \quad if \;\; \mathrm{c} = y^T x = scalar \quad \Rightarrow \frac{\partial c}{\partial z} = x^T \frac{\partial y}{\partial z} + y^T \frac{\partial x}{\partial z}$$

- Proof

$$\mathrm{c} = y^T x = \mathrm{c}^T = x^T y$$

$$\Rightarrow \frac{\partial c}{\partial z} = \frac{\partial c}{\partial y}\frac{\partial y}{\partial z} + \frac{\partial c}{\partial x}\frac{\partial x}{\partial z}$$

$$= x^T \frac{\partial y}{\partial z} + y^T \frac{\partial x}{\partial z}$$

Differentiation of Matrix 5

- Lemma 6

$$if \;\; \mathrm{c} = x^T x = scalar \qquad \Rightarrow \frac{\partial c}{\partial z} = 2x^T \frac{\partial x}{\partial z}$$

- Lemma7

$$if \;\; \mathrm{c} = y^T A x$$

$$\Rightarrow \frac{\partial c}{\partial z} = \frac{\partial c}{\partial y}\frac{\partial y}{\partial z} + \frac{\partial c}{\partial x}\frac{\partial x}{\partial z}$$

$$= \frac{\partial c^T}{\partial y}\frac{\partial y}{\partial z} + y^T A \frac{\partial x}{\partial z} = x^T A^T \frac{\partial y}{\partial z} + y^T A \frac{\partial x}{\partial z}$$

Hadamard Product

- Matrix Multiplication (what you have learned)

$$AB = \underset{m\times a}{A}\ \underset{a\times n}{B} = \underset{m\times n}{AB}$$

- Hadamard Product (Matlab → A.*B)

$$A \circ B = \underset{m\times n}{A} \circ \underset{m\times n}{B} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix} \circ \begin{bmatrix} b_{11} & b_{12} & \dots & b_{1n} \\ b_{21} & b_{22} & \dots & b_{2n} \\ \dots & \dots & \dots & \dots \\ b_{m1} & b_{m2} & \dots & b_{mn} \end{bmatrix}$$

$$= \begin{bmatrix} a_{11}b_{11} & a_{12}b_{12} & \dots & a_{1n}b_{1n} \\ a_{21}b_{21} & a_{22}b_{22} & \dots & a_{2n}b_{2n} \\ \dots & \dots & \dots & \dots \\ a_{m1}b_{m1} & a_{m2}b_{m2} & \dots & a_{mn}b_{mn} \end{bmatrix} = \underset{m\times n}{AB}$$

Neural Network Multiplication Problems

$$J = \frac{1}{2}\sum_{i \in D} e_i^{\,2} = \frac{1}{2}\sum_{i \in D} \| y_i - \hat{y} \|^2$$

$$\nabla J_{w_1} = \frac{\partial J}{\partial W_1} = \sum_{i \in D} (y_i - \hat{y}) \left(0 - \frac{\partial \hat{y}}{\partial W_1} \right)$$

$$= -\sum_{i \in D} (y_i - \hat{y}) W_2 \Phi'(W_1 X)\, \mathrm{X}$$

- You cannot differentiate NN directly

$$\underset{i \times j}{C} = \underset{i \times k}{A}\ \underset{k \times j}{B}$$

$$c_{ij} = \sum_{k} a_{ik} b_{kj}$$

Matrix Multiplication

$$\underset{i \times j}{C} = \underset{i \times j}{A} \circ \underset{i \times j}{B}$$

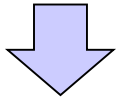
$$c_{ij} = a_{ij} b_{ij}$$

Hadamard Multiplication

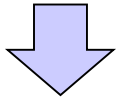
Neural Network Multiplication

5 Neural Network Weight Update

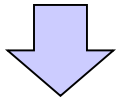
Weight Update by Gradient Descent Method is the key for Neural Network



But, Differentiation is Complex

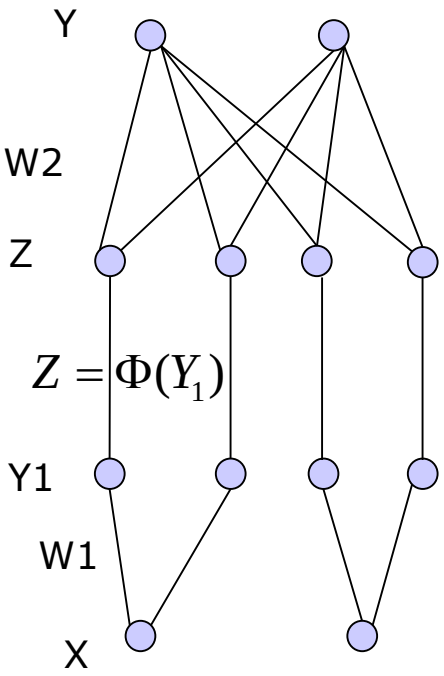


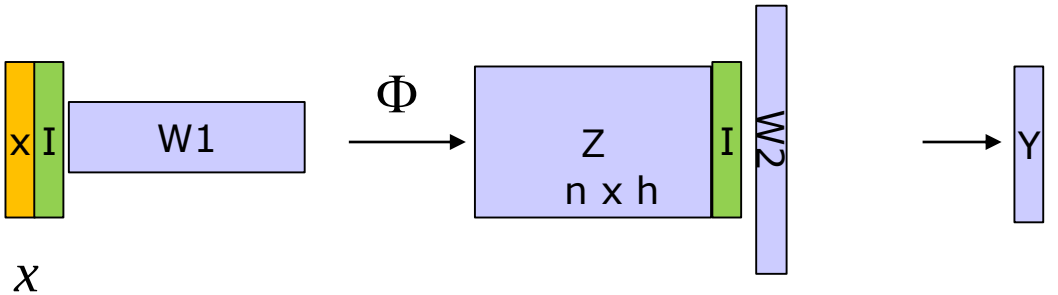
We learn Basic Structure first



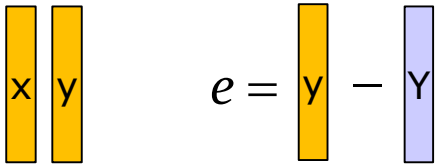
Extend above Neural Network Structure

Network Model by Matrix Expression I





$$\underset{n\times 1}{x} \downarrow \qquad \underset{n\times 2}{X} = [x \quad I] \rightarrow \underset{2\times h}{W_1} \xrightarrow{\Phi} \underset{n\times h}{Z} \downarrow \qquad [\underset{n\times(h+1)}{Z} \quad I] \rightarrow \underset{(h+1)\times 1}{W_2} \rightarrow \underset{n\times 1}{Y}$$



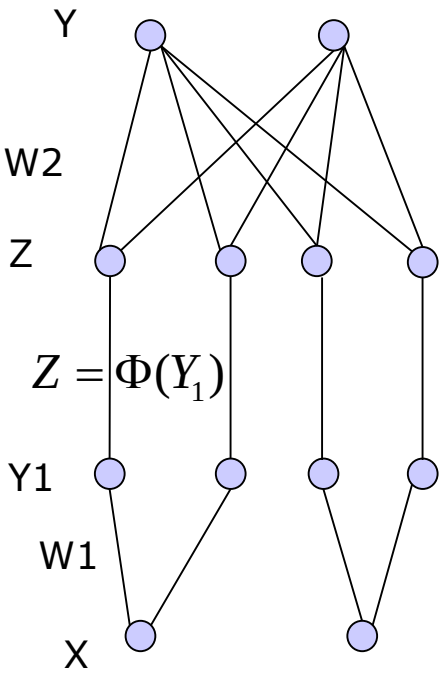
Data

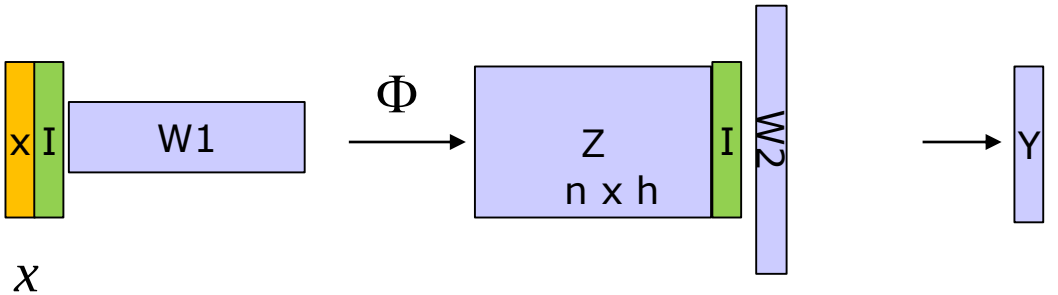
Neural Network

$$Z = \Phi([x \quad I]W_1)$$

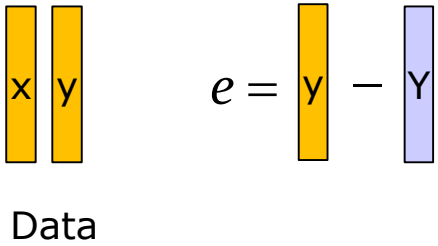
$$Y = [Z \quad I]W_2$$

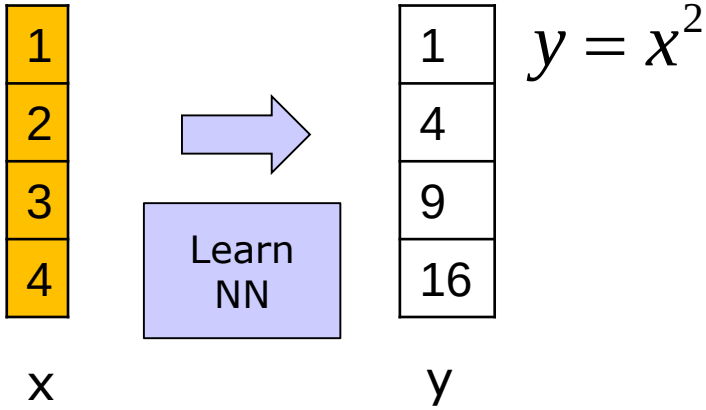
Network Model by Matrix Expression I



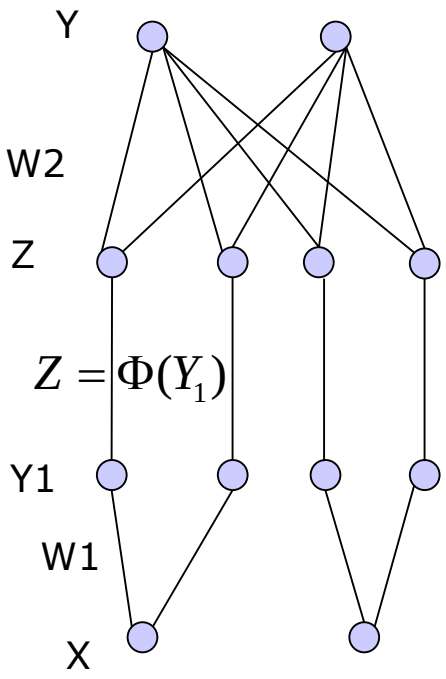


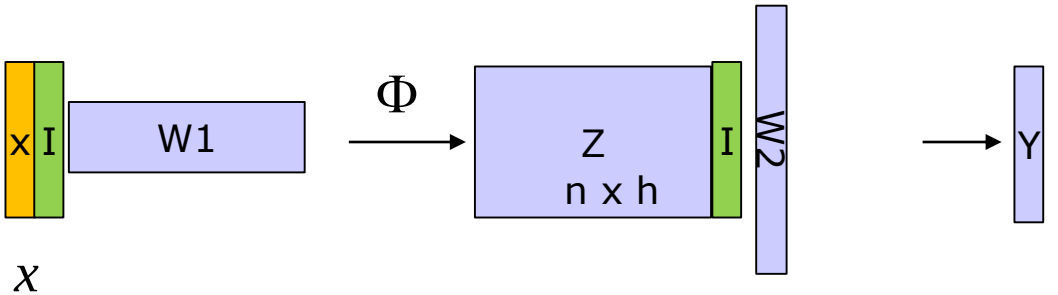
$$\underset{n\times 1}{x} \rightarrow \underset{n\times 2}{X} = [x \quad I] \rightarrow \underset{2\times h}{W_1} \xrightarrow{\Phi} \underset{n\times h}{Z} \rightarrow \underset{n\times(h+1)}{[Z \quad I]} \rightarrow \underset{(h+1)\times 1}{W_2} \rightarrow \underset{n\times 1}{Y}$$





Network Model by Matrix Expression I





$$\underset{n\times 1}{x} \downarrow \qquad \underset{n\times 2}{X} = [x \quad I] \rightarrow \underset{2\times h}{W_1} \xrightarrow{\Phi} \underset{n\times h}{Z} \downarrow \qquad [\underset{n\times(h+1)}{Z} \quad I] \rightarrow \underset{(h+1)\times 1}{W_2} \rightarrow \underset{n\times 1}{Y}$$

n

1	1
2	1
3	1
4	1

[x I]

h

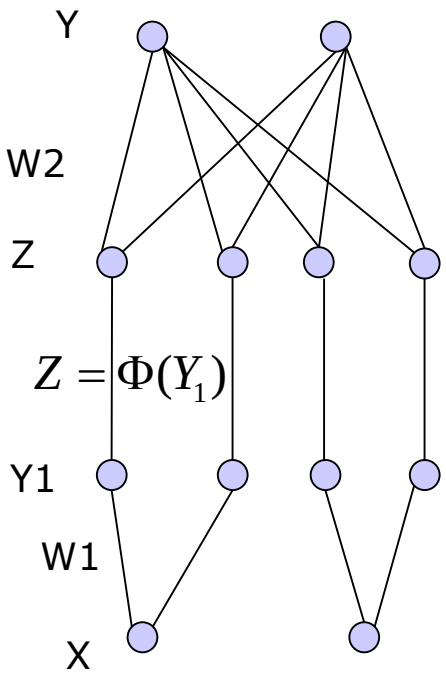
w11	w12	w13	w14	w15
w21	w22	w23	w24	w25

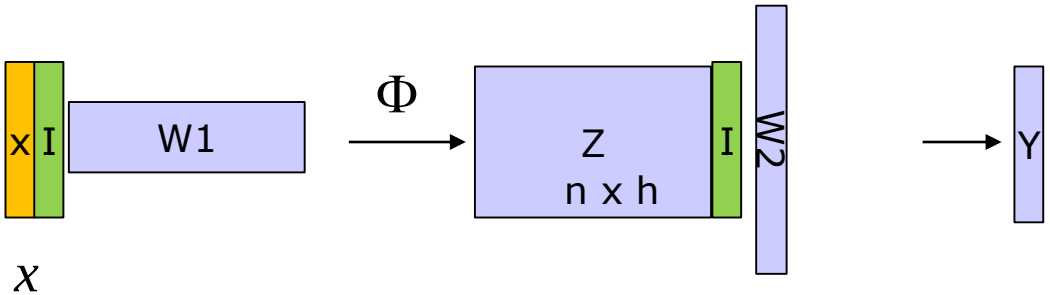
W1

=

w11 +w21	w12 +w22	w13 +w23	w14 +w24	w15 +w25
2w11 +w21	2w12 +w22	2w13 +w23	2w14 +w24	2w15 +w25
3w11 +w21	3w12 +w22	3w13 +w23	3w14 +w24	3w15 +w25
4w11 +w21	4w12 +w22	4w13 +w23	4w14 +w24	4w15 +w25

Network Model by Matrix Expression I





$$\underset{n\times 1}{X}$$

$$\underset{n\times 2}{X} = [x \quad I] \rightarrow \underset{2\times h}{W_1} \xrightarrow{\Phi} \underset{n\times h}{Z}$$

$$\underset{n\times(h+1)}{[Z \quad I]} \rightarrow \underset{(h+1)\times 1}{W_2} \rightarrow \underset{n\times 1}{Y}$$

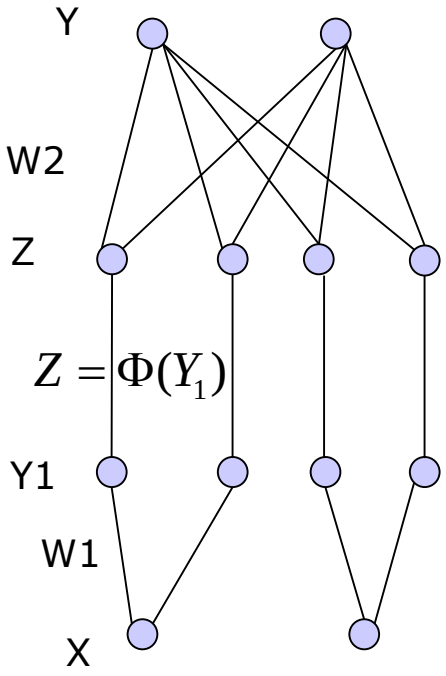
$$Z = \Phi$$

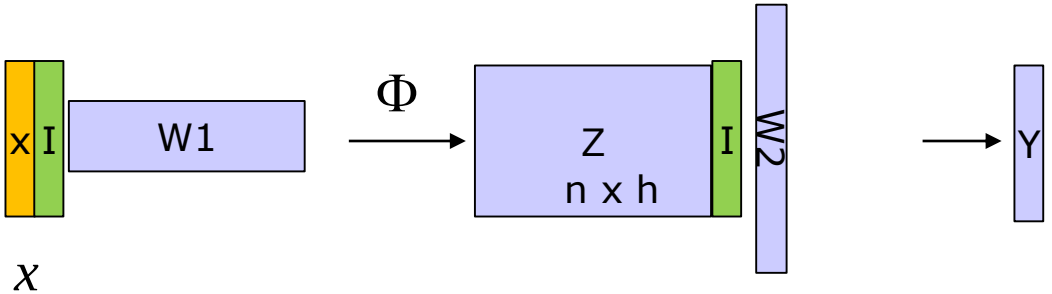
h

n

w11 +w21	w12 +w22	w13 +w23	w14 +w24	w15 +w25
2w11 +w21	2w12 +w22	2w13 +w23	2w14 +w24	2w15 +w25
3w11 +w21	3w12 +w22	3w13 +w23	3w14 +w24	3w15 +w25
4w11 +w21	4w12 +w22	4w13 +w23	4w14 +w24	4w15 +w25

Network Model by Matrix Expression I





$$\underset{n\times 1}{x} \downarrow \qquad \underset{n\times 2}{X} = [x \quad I] \rightarrow \underset{2\times h}{W_1} \xrightarrow{\Phi} \underset{n\times h}{Z} \downarrow \qquad [\underset{n\times(h+1)}{Z} \quad I] \rightarrow \underset{(h+1)\times 1}{W_2} \rightarrow \underset{n\times 1}{Y}$$

h

n

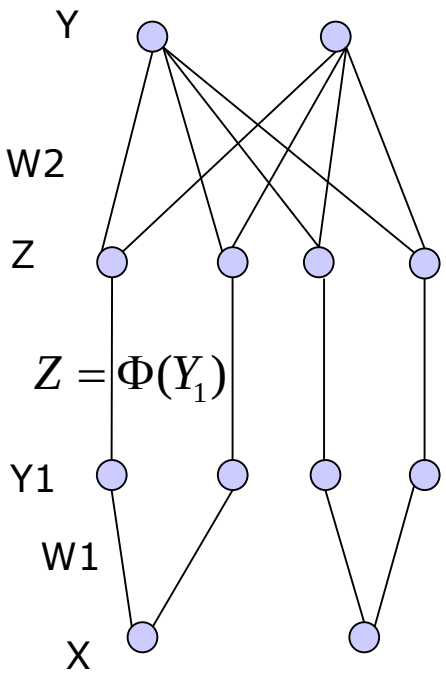
Φ

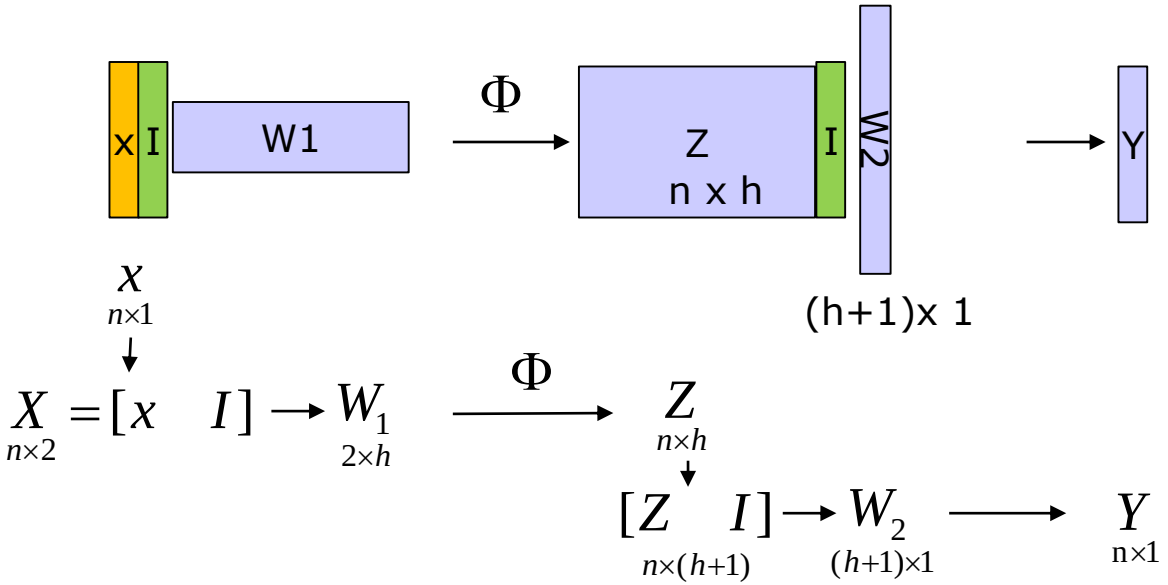
w11 +w21	w12 +w22	w13 +w23	w14 +w24	w15 +w25
2w11 +w21	2w12 +w22	2w13 +w23	2w14 +w24	2w15 +w25
3w11 +w21	3w12 +w22	3w13 +w23	3w14 +w24	3w15 +w25
4w11 +w21	4w12 +w22	4w13 +w23	4w14 +w24	4w15 +w25

1
1
1
1

w2,1
w2,2
w2,3
w2,4
w2,5
w2,6

Network Model by Matrix Expression I





$$\hat{y} = \begin{pmatrix} \Phi(w_{11}+w_{21})w_{2,1}+\Phi(w_{12}+w_{22})w_{2,2}+\Phi(w_{13}+w_{23})w_{2,3}+\Phi(w_{14}+w_{24})w_{2,4}+\Phi(w_{15}+w_{25})w_{2,5}+w_{2,6} \\ \Phi(2w_{11}+w_{21})w_{2,1}+\Phi(2w_{12}+w_{22})w_{2,2}+\Phi(2w_{13}+w_{23})w_{2,3}+\Phi(2w_{14}+w_{24})w_{2,4}+\Phi(2w_{15}+w_{25})w_{2,5}+w_{2,6} \\ \Phi(3w_{11}+w_{21})w_{2,1}+\Phi(3w_{12}+w_{22})w_{2,2}+\Phi(3w_{13}+w_{23})w_{2,3}+\Phi(3w_{14}+w_{24})w_{2,4}+\Phi(3w_{15}+w_{25})w_{2,5}+w_{2,6} \\ \Phi(4w_{11}+w_{21})w_{2,1}+\Phi(4w_{12}+w_{22})w_{2,2}+\Phi(4w_{13}+w_{23})w_{2,3}+\Phi(4w_{14}+w_{24})w_{2,4}+\Phi(4w_{15}+w_{25})w_{2,5}+w_{2,6} \end{pmatrix}$$

Differentiation with W2

$$J = \frac{1}{2}\sum_k {e_k}^2 = \frac{1}{2} e^T e$$

$$\frac{\partial J}{\partial W_2} = e^T \frac{\partial e}{\partial W_2}$$

Lemma 5.

$$c = x^T x$$

$$\frac{\partial c}{\partial z} = 2x^T \frac{\partial x}{\partial z}$$

$$\frac{\partial J}{\partial W_2} = e^T \frac{\partial e}{\partial W_2} = -e^T \frac{\partial Y}{\partial W_2} = -e^T \frac{\partial [Z \quad I] W_2}{\partial W_2} = -e^T{}_{1\times n} [Z \quad I]_{n\times(h+1)}$$

$$Transpose \rightarrow Vector = \left(\frac{\partial J}{\partial W_2}\right)^T_{(h+1)\times 1} = -\left([Z \quad I]_{n\times(h+1)}\right)^T e_{n\times 1} = -[Z \quad I]^T e$$

Differentiation with W1

$$J = \frac{1}{2}\sum_k e_k{}^2$$

$$\frac{\partial J}{\partial w_{11}} = \sum_k e_k \frac{\partial e_k}{\partial w_{11}} = \sum_k e_k \frac{\partial (y_k - \hat{y}_k)}{\partial w_{11}} = -\sum_k e_k \frac{\partial \hat{y}_k}{\partial w_{11}}$$

$$= -e_1 \partial \frac{\Phi(w_{11} + w_{21})w_{2,1} + \Phi(w_{12} + w_{22})w_{2,2} + \Phi(w_{13} + w_{23})w_{2,3} + \Phi(w_{14} + w_{24})w_{2,4} + \Phi(w_{15} + w_{25})w_{2,5} + w_{2,6}}{\partial w_{11}}$$

$$-e_2 \partial \frac{\Phi(2w_{11} + w_{21})w_{2,1} + \Phi(2w_{12} + w_{22})w_{2,2} + \Phi(2w_{13} + w_{23})w_{2,3} + \Phi(2w_{14} + w_{24})w_{2,4} + \Phi(2w_{15} + w_{25})w_{2,5} + w_{2,6}}{\partial w_{11}}$$

$$-e_3 \partial \frac{\Phi(3w_{11} + w_{21})w_{2,1} + \Phi(3w_{12} + w_{22})w_{2,2} + \Phi(3w_{13} + w_{23})w_{2,3} + \Phi(3w_{14} + w_{24})w_{2,4} + \Phi(3w_{15} + w_{25})w_{2,5} + w_{2,6}}{\partial w_{11}}$$

$$-e_4 \frac{\partial \Phi(4w_{11} + w_{21})w_{2,1} + \Phi(4w_{12} + w_{22})w_{2,2} + \Phi(4w_{13} + w_{23})w_{2,3} + \Phi(4w_{14} + w_{24})w_{2,4} + \Phi(4w_{15} + w_{25})w_{2,5} + w_{2,6}}{\partial w_{11}}$$

Differentiation with W1

$$J = \frac{1}{2}\sum_k e_k{}^2$$

$$\frac{\partial J}{\partial w_{11}} = \sum_k e_k \frac{\partial e_k}{\partial w_{11}} = \sum_k e_k \frac{\partial (y_k - \hat{y}_k)}{\partial w_{11}} = -\sum_k e_k \frac{\partial \hat{y}_k}{\partial w_{11}}$$

$$= -e_1 \frac{\partial \Phi(w_{11} + w_{21})w_{2,1}}{\partial w_{11}} - e_2 \frac{\partial \Phi(2w_{11} + w_{21})w_{2,1}}{\partial w_{11}} - e_3 \frac{\partial \Phi(3w_{11} + w_{21})w_{2,1}}{\partial w_{11}} - e_4 \frac{\partial \Phi(4w_{11} + w_{21})w_{2,1}}{\partial w_{11}}$$

$$= -e_1 \Phi'(w_{11} + w_{21})w_{2,1} - e_2 \Phi'(2w_{11} + w_{21})2w_{2,1} - e_3 \Phi'(3w_{11} + w_{21})3w_{2,1} - e_4 \Phi'(4w_{11} + w_{21})4w_{2,1}$$

Can you find the PATTERN?

Differentiation with W1

$$J = \frac{1}{2}\sum_k e_k{}^2$$

$$\frac{\partial J}{\partial w_{1j}} = \sum_k e_k \frac{\partial e_k}{\partial w_{1j}} = \sum_k e_k \frac{\partial (y_k - \hat{y}_k)}{\partial w_{1j}} = -\sum_k e_k \frac{\partial \hat{y}_k}{\partial w_{1j}}$$

$$= -e_1 \frac{\partial \Phi(w_{1j} + w_{2j}) w_{2,\mathrm{j}}}{\partial w_{1j}} - e_2 \frac{\partial \Phi(2w_{1j} + w_{2j}) w_{2,\mathrm{j}}}{\partial w_{1j}} - e_3 \frac{\partial \Phi(3w_{1j} + w_{2j}) w_{2,\mathrm{j}}}{\partial w_{1j}} - e_4 \frac{\partial \Phi(4w_{1j} + w_{2j}) w_{2,\mathrm{j}}}{\partial w_{1j}}$$

$$= -e_1 \Phi'(w_{1j} + w_{2j}) w_{2,\mathrm{j}} - e_2 \Phi'(2w_{1j} + w_{2j}) 2w_{2,\mathrm{j}} - e_3 \Phi'(3w_{1j} + w_{2j}) 3w_{2,\mathrm{j}} - e_4 \Phi'(4w_{1j} + w_{2j}) 4w_{2,\mathrm{j}}$$

$$\frac{\partial J}{\partial w_{2j}} = \sum_k e_k \frac{\partial e_k}{\partial w_{2j}} = \sum_k e_k \frac{\partial (y_k - \hat{y}_k)}{\partial w_{2j}} = -\sum_k e_k \frac{\partial \hat{y}_k}{\partial w_{2j}}$$

$$= -e_1 \frac{\partial \Phi(w_{1j} + w_{2j}) w_{2,\mathrm{j}}}{\partial w_{2j}} - e_2 \frac{\partial \Phi(2w_{1j} + w_{2j}) w_{2,\mathrm{j}}}{\partial w_{2j}} - e_3 \frac{\partial \Phi(3w_{1j} + w_{2j}) w_{2,\mathrm{j}}}{\partial w_{2j}} - e_4 \frac{\partial \Phi(4w_{1j} + w_{2j}) w_{2,\mathrm{j}}}{\partial w_{2j}}$$

$$= -e_1 \Phi'(w_{1j} + w_{2j}) w_{2,\mathrm{j}} - e_2 \Phi'(2w_{1j} + w_{2j}) w_{2,\mathrm{j}} - e_3 \Phi'(3w_{1j} + w_{2j}) w_{2,\mathrm{j}} - e_4 \Phi'(4w_{1j} + w_{2j}) w_{2,\mathrm{j}}$$

Differentiation with W1

$$J = \frac{1}{2}\sum_k {e_k}^2$$

$$\frac{\partial J}{\partial w_{1j}} = \sum_k e_k \frac{\partial e_k}{\partial w_{1j}} = \sum_k e_k \frac{\partial (y_k - \hat{y}_k)}{\partial w_{1j}} = -\sum_k e_k \frac{\partial \hat{y}_k}{\partial w_{1j}}$$

$$= -e_1 \Phi'(w_{1j} + w_{2j}) w_{2,\mathrm{j}} - e_2 \Phi'(2w_{1j} + w_{2j}) 2w_{2,\mathrm{j}} - e_3 \Phi'(3w_{1j} + w_{2j}) 3w_{2,\mathrm{j}} - e_4 \Phi'(4w_{1j} + w_{2j}) 4w_{2,\mathrm{j}}$$

$$= \sum_k -e_k \Phi'(x_k w_{1j} + w_{2j}) x_k w_{2,\mathrm{j}}$$

$$\frac{\partial J}{\partial w_{2j}} = \sum_k e_k \frac{\partial e_k}{\partial w_{2j}} = \sum_k e_k \frac{\partial (y_k - \hat{y}_k)}{\partial w_{2j}} = -\sum_k e_k \frac{\partial \hat{y}_k}{\partial w_{2j}}$$

$$= -e_1 \Phi'(w_{1j} + w_{2j}) w_{2,\mathrm{j}} - e_2 \Phi'(2w_{1j} + w_{2j}) w_{2,\mathrm{j}} - e_3 \Phi'(3w_{1j} + w_{2j}) w_{2,\mathrm{j}} - e_4 \Phi'(4w_{1j} + w_{2j}) w_{2,\mathrm{j}}$$

$$= \sum_k -e_k \Phi'(x_k w_{1j} + w_{2j}) w_{2,\mathrm{j}}$$

Differentiation with W1

$$J = \frac{1}{2}\sum_k e_k{}^2$$

$$\rightarrow X_k = [x_k \quad 1]$$

$$\rightarrow X_{ik}^T = \begin{bmatrix} x_k \\ 1 \end{bmatrix}$$

$$\underset{i\times j}{C} = \underset{i\times k}{A}\ \underset{k\times j}{B} \qquad \underset{i\times j}{C} = \underset{i\times j}{A} \circ \underset{i\times j}{B}$$

$$c_{ij} = \sum_k a_{ik} b_{kj} \qquad c_{ij} = a_{ij} b_{ij}$$

$$\frac{\partial J}{\partial w_{1j}} = \sum_k -x_k e_k \Phi'(x_k w_{1j} + w_{2j}) w_{2,j} = \sum_k -x_k e_k w_{2,j} \Phi'_{kj} = \sum_k -x_k \left(e W_{2,h\times 1}^T\right)_{kj} \Phi'_{kj}$$

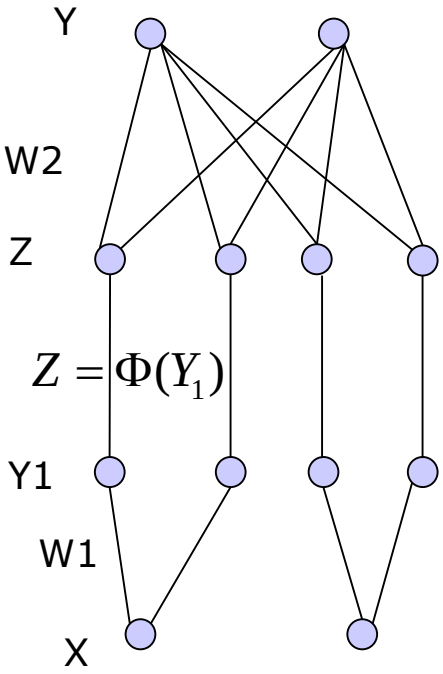
$$\frac{\partial J}{\partial w_{2j}} = \sum_k -1 e_k \Phi'(x_k w_{1j} + w_{2j}) w_{2,j} = \sum_k -1 \left(e W_{2,h\times 1}^T\right)_{kj} \Phi'_{kj}$$

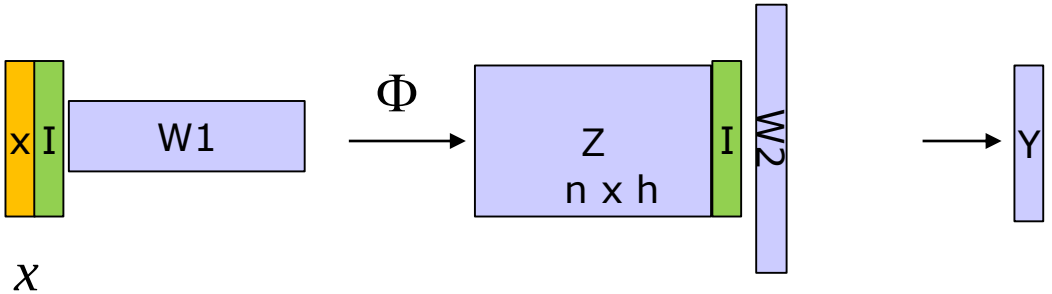
$$\frac{\partial J}{\partial w_{ij}} = \sum_k -X_{ik}^T \left(e W_2^T\right)_{kj} \Phi'_{kj} = \sum_k -X_{ik}^T \left(e W_2^T \circ \Phi'\right)_{kj}$$

$$\therefore \frac{\partial J}{\partial W_1} = -X^T \left(e W_2^T \circ \Phi'\right)$$

Hadamard multiplication

Neural Network Example





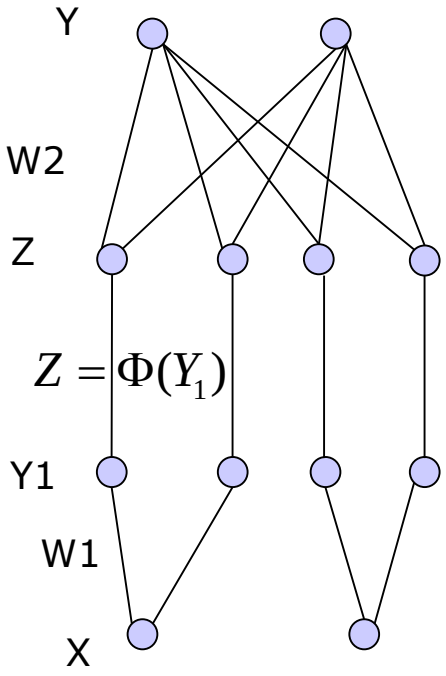
$$\underset{n\times 1}{x} \downarrow \qquad \underset{n\times 2}{X} = [x \quad I] \rightarrow \underset{2\times h}{W_1} \xrightarrow{\Phi} \underset{n\times h}{Z} \downarrow \qquad [\underset{n\times(h+1)}{Z} \quad I] \rightarrow \underset{(h+1)\times 1}{W_2} \rightarrow \underset{n\times 1}{Y}$$

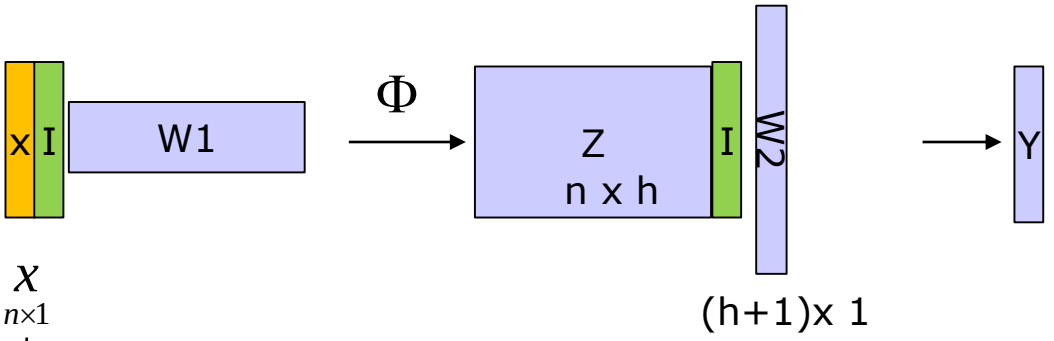
$$J = \frac{1}{2}\sum_k \left(y_k - \hat{y}_k\right)^2 = \frac{1}{2}\sum_k e_k{}^2 = \frac{1}{2}e^T e$$

$$Matrix: \frac{\partial J}{\partial W_1} = -[x \quad I]^T \left(eW_{2,h\times 1}^T \circ \Phi'\right)$$

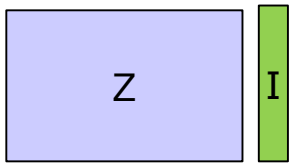
$$Vector: \frac{\partial J}{\partial W_2} = -[Z \quad I]^T e$$

Python Example: l3sig





$$\underset{n\times 1}{x} \downarrow \quad \underset{n\times 2}{X} = [x \quad I] \rightarrow \underset{2\times h}{W_1} \xrightarrow{\Phi} \underset{n\times h}{Z} \downarrow \quad [\underset{n\times(h+1)}{Z} \quad I] \rightarrow \underset{(h+1)\times 1}{W_2} \rightarrow \underset{n\times 1}{Y}$$

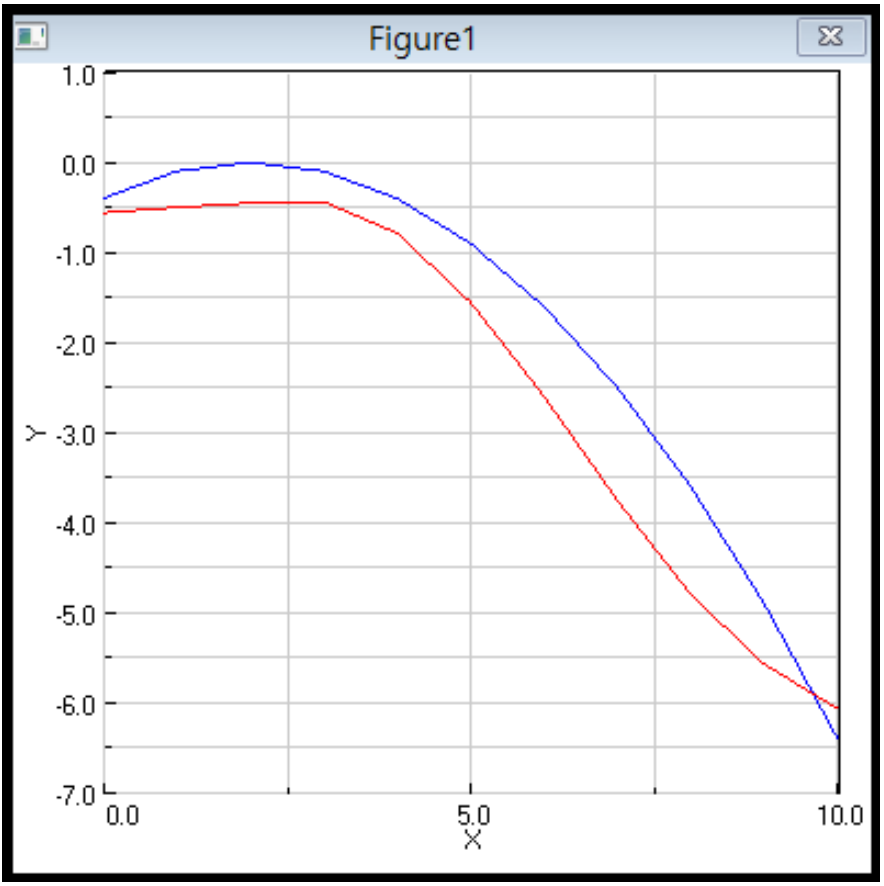



```
n=11
h=20

x= linspace(0,10,n).T()
y= -0.1*pow(x-2,2);

# X
X=x
X[:,2] = 1

W1=randn(2,h)
W2=randn(h+1,1)
```

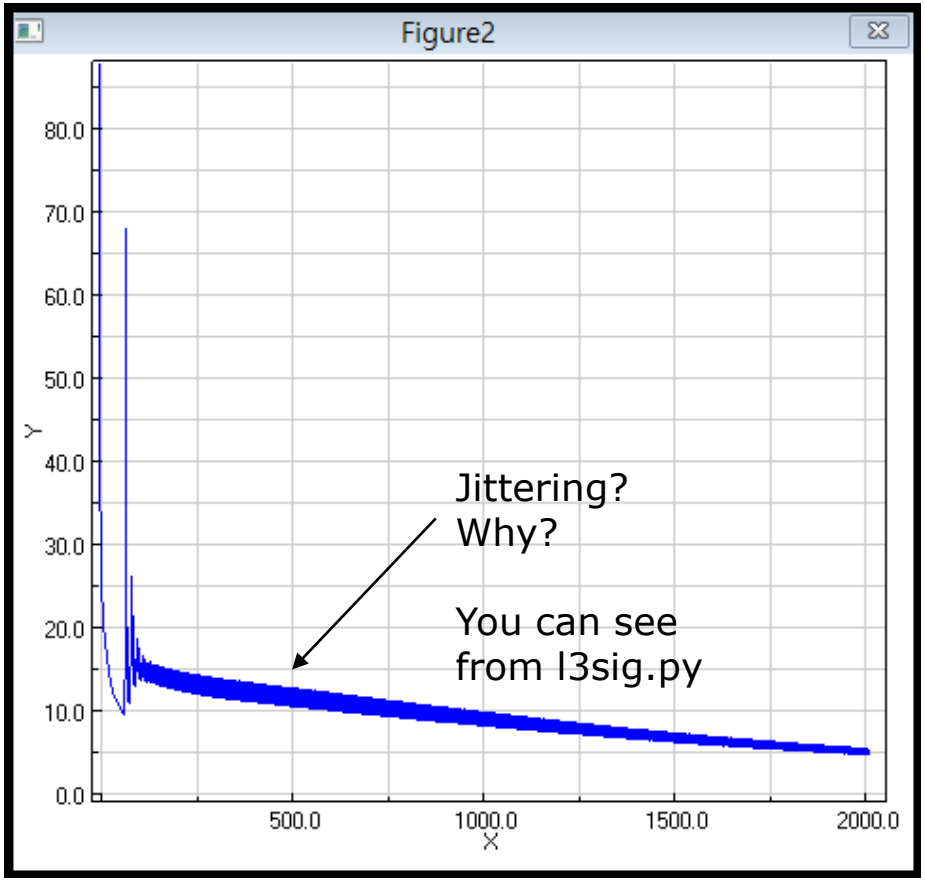
x=
```
[ 0.0000 ]
[ 0.2500 ]
[ 0.5000 ]
[ 0.7500 ]
[ 1.0000 ]
```

$$\underset{n\times 2}{X} = [x \quad I]$$

Python Example: l3sig



Blue: y
Red: Y(est)



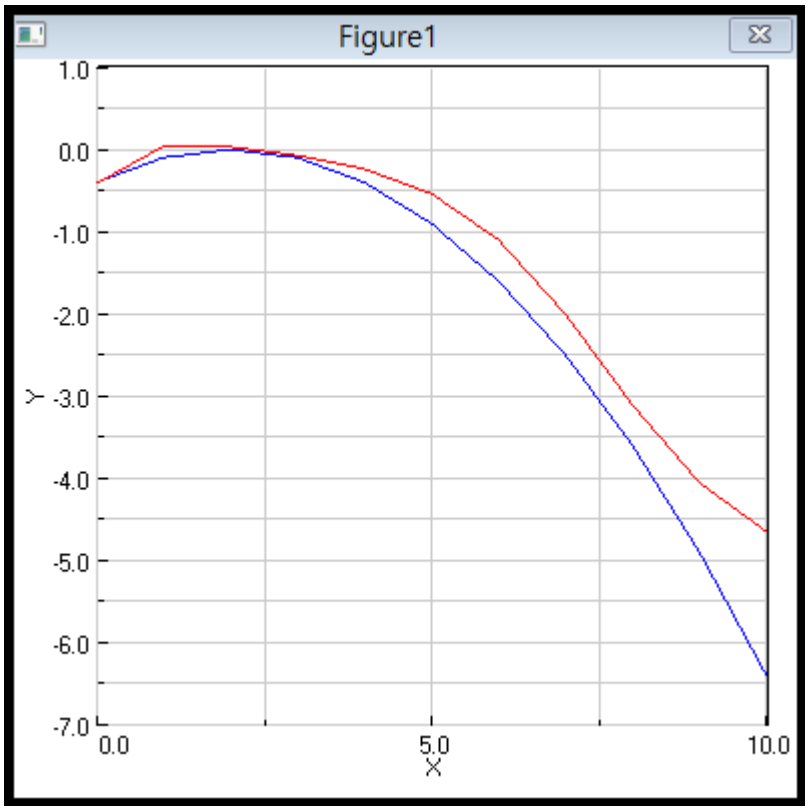
J during 2000 iterations

If we increase Hidden Space?

:Hidden Space Increases Estimation Performance

- h=20 → h=200
- What happens?

```
n=11
h=20

x= linspace(0,10,n).T()
y= -0.1*pow(x-2,2);

# X
X=x
X[:,2] = 1
I=ones(n,h)

W1=randn(2,h)
W2=randn(h+1,1)
```



Hidden layer increases the DOF of Estimation Results