Chapter 2 Learning Processes

授課教師: 張傳育 博士 (Chuan-Yu Chang Ph.D.)
E-mail: chuanyu@yuntech.edu.tw
Tel: (05)5342601 ext. 4516
Office: EB212

Introduction

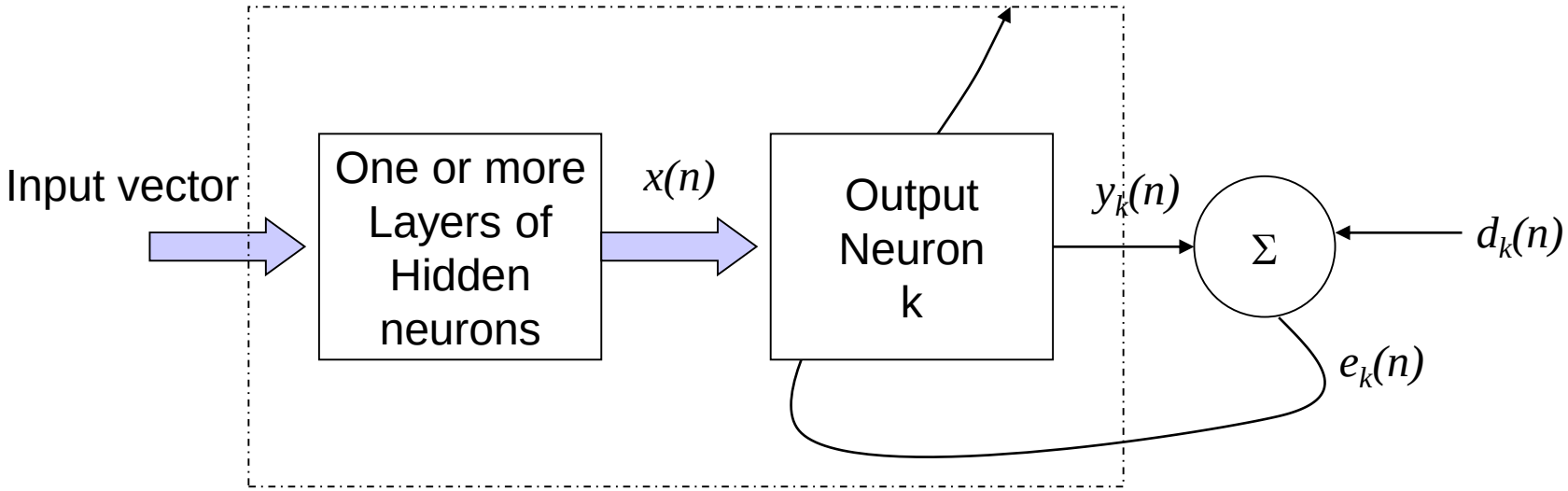
- The neural network has the ability to learn from its environment, and to improve its performance through learning.
  - A neural network learns about its environment through an interactive process of adjustments applied to its synaptic weights and bias levels.
  - Learning in Neural Networks
    - Stimulated by an environment
    - Undergoes changes as a results of this stimulation
    - Because of the changes, responds in a new way to the environment
  - The solution of a learning problem is called a *learning algorithm*.

Adjustments to the synaptic weights

- The algorithm starts from an arbitrary setting of the neuron's synaptic weights
- Adjustments to the synaptic weights are made on a continuous basis
- Computation of adjustments are completed inside a time interval

Error Correction Learning

- The output signal is compared to a *desired response* .
- The corrective adjustments are designed to make the output signal $y_k(n)$ come closer to the desired response $d_k(n)$ in a step-by-step manner.



Error Correction Learning

- The error signal

$$e_k(n) = d_k - y_k(n)$$

- Minimizing a *cost function*

$$\xi(n) = \frac{1}{2} e_k^2(n)$$

  - The step-by-step adjustments to the synaptic weights of neuron k are continued until the system reaches a steady state.

- According to the *delta rule*, the adjustment $w_{kj}(n)$ applied to the synaptic weight w_{kj} at time step n is defined by

Learning rate

$$\Delta w_{kj}(n) = \eta e_k(n) x_j(n)$$

Error Correction Learning

- The delta rule
  - The adjustment made to a synaptic weight of a neuron is proportional to the product of the error signal and the input signal of the synapse in question.
  - The updated value of synaptic weight w_{kj} is

$$w_{kj}(n+1) = w_{kj}(n) + \Delta w_{kj}(n)$$

the unit-delay operator

$$w_{kj} = z^{-1}[w_{kj}(n+1)]$$

Memory-Based Learning

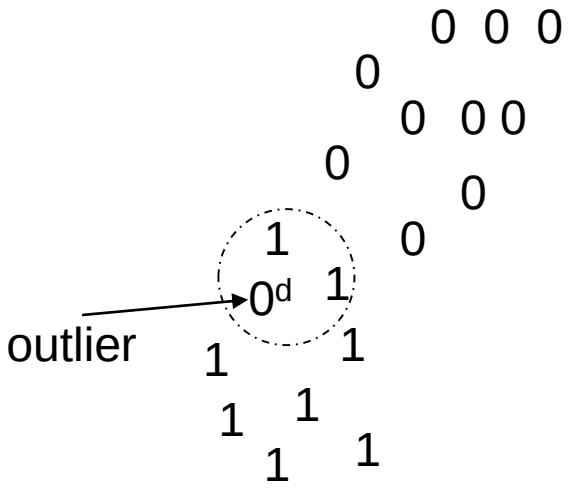
- All of the past experiences are explicitly stored in a large memory of correctly classified input-output examples $\{(\mathbf{x}_i, \mathbf{d}_i)\}$
- 所有memory-based learning algorithm會有兩個重要成分
  - 定義鄰近區域(local neighborhood)的準則
    - Nearest neighbor rule: the training example that lies in the immediate neighborhood of the test vector $\boldsymbol{x}_{test}$.
    - 如果 $\mathbf{x}'_n \in \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$為$\boldsymbol{x}_{test}$的最近鄰居，需滿足

$$\min_i d(\mathbf{x}_i, \mathbf{x}_{test}) = d(\mathbf{x}'_N, \mathbf{x}_{test})$$

  - 用於訓練樣本的學習法則

Memory-Based Learning

- K-nearest neighbor classifier
  - Identify the k-classified patterns that lie nearest to the test vector $\mathbf{x}_{test}$ for some integer k.
  - Assign $\mathbf{x}_{test}$ to the class that is most frequently represented in the k nearest neighbors to $\mathbf{x}_{test}$.



雖然d點為outlier,
但因其最近的三個點
中,最多的”1”,因此d
點將被歸類為”1”

Hebbian Learning

- Hebbian Learning
  - [Donald Hebb] The strength of a synapse between cells A and B increased slightly for the situation when the firing in A was followed by firing in B with a very small time delay.
    - For two neurons on either side of a synapse that are *synchronously* activated, then the strength of the synapse is increased.
  - [Stent] expanded Hebb's original statement to include the case when two neurons on either side of a synapse are *asynchronously* activated, leading to a weakened synapse.
  - [Rumelhart] Adjust the strength of the connection between units A and B is proportion to the product of their simultaneous activation.
    - If the product of the activations is positive, the modification to the synaptic connection is more excitatory.
    - If the product of the activations is negative, the modification to the synaptic connection is more inhibitory.

Hebbian Learning (cont.)

- Hebbian synapse
  - Uses a highly local, time-dependent, and strongly interactive mechanism to increase synaptic efficiency as a function of the correlation between the presynaptic and postsynaptic activity levels.

Hebbian Learning (cont.)

- Four key properties of a Hebbian synapse
  - Time-dependent mechanism
    - To change in a Hebbian synapse that depend on the precise time of occurrence of the presynaptic and postsynaptic activity levels.
  - Local mechanism
    - Within a synapse, ongoing activity levels in the presynaptic and postsynaptic units are used by a hebbian synapse to produce an input-dependent, local synaptic modification.
  - Interactive mechanism
    - Any form of hebbian learning depends on the interaction between presynaptic and postsynaptic activities.
  - Conjunctional (correlational) mechanism
    - The “co-occurrence” of presynaptic and postsynaptic activities within a relatively short time interval is sufficient to produce a synaptic modification.

Hebbian Learning (cont.)

- Synaptic activities can be categorized as
  - Hebbian
    - A Hebbian synapse increases its strength with positively correlated presynaptic and postsynaptic signals, and its strength is decreased when the activities are either uncorrelated or negatively correlated.
  - Anti-Hebbian
    - An anti-Hebbian synapse enhance negatively correlated presynaptic or postsynaptic activities and weakens positively correlated activities.
  - Non-Hebbian
    - A non-Hebbian synapse does not involve the strongly interactive, highly local, time-dependent mechanism.

Hebbian Learning

- The adjustment applied to the synaptic weight w_{kj} at time step n is expressed in the general form

$$\Delta w_{kj}(n) = F\left(y_k(n), x_j(n)\right)$$

  - where $F(.)$ is a function of both presynaptic and postsynaptic signals
- The simplest form of Hebbian learning (activity product rule)

$$\Delta w_{kj}(n) = \eta y_k(n) x_j(n)$$

  - The repeated application of the input signal x_j leads to an increase in y_k and therefore exponential growth that final drives the synaptic connection into saturation.
  - At that point no information will be stored in the synapse and selectivity is lost.

Competitive Learning

- The output neurons of a neural network compete among themselves to become active (fire)
- Only a single output neuron is active at any one time.
- There are three basis elements to a competitive learning rule
  - A set of neurons that are all the same except for some randomly distributed synaptic weights, and which therefore respond differently to a given set of input patterns.
  - A limit imposed on the “strength” of each neuron.
  - A mechanism that permits the neurons to compete for the right to respond to a given subset of inputs, such that only one output neuron, or only one neuron per group, is active at a time.
    - The neuron that wins the competition is called a *winner-takes-all* neuron.

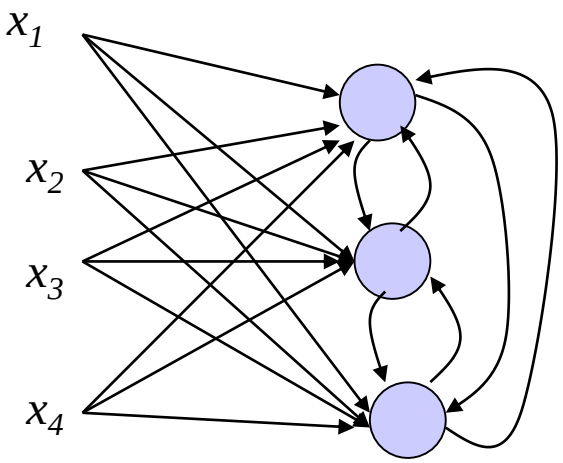
Competitive Learning

- For a neuron k to be the winning neuron, its induced local field v_k for a specified input pattern $\boldsymbol{x}$ must be the largest among all the neurons in the network

$$y_k = \begin{cases} 1 & \text{if } v_{(k)} > v_j \text{ for all } j, j \neq k \\ 0 & \text{otherwise} \end{cases}$$

- According to the standard competitive learning rule, the change w_{kj} applied to synaptic weight w_{kj} is defined as

$$\Delta w_{kj} = \begin{cases} \eta(x_j - w_{kj}) & \text{if neuron } k \text{ wins the competition} \\ 0 & \text{if neuron } k \text{ loses the competition} \end{cases}$$



Geometric interpretation of the competitive learning process

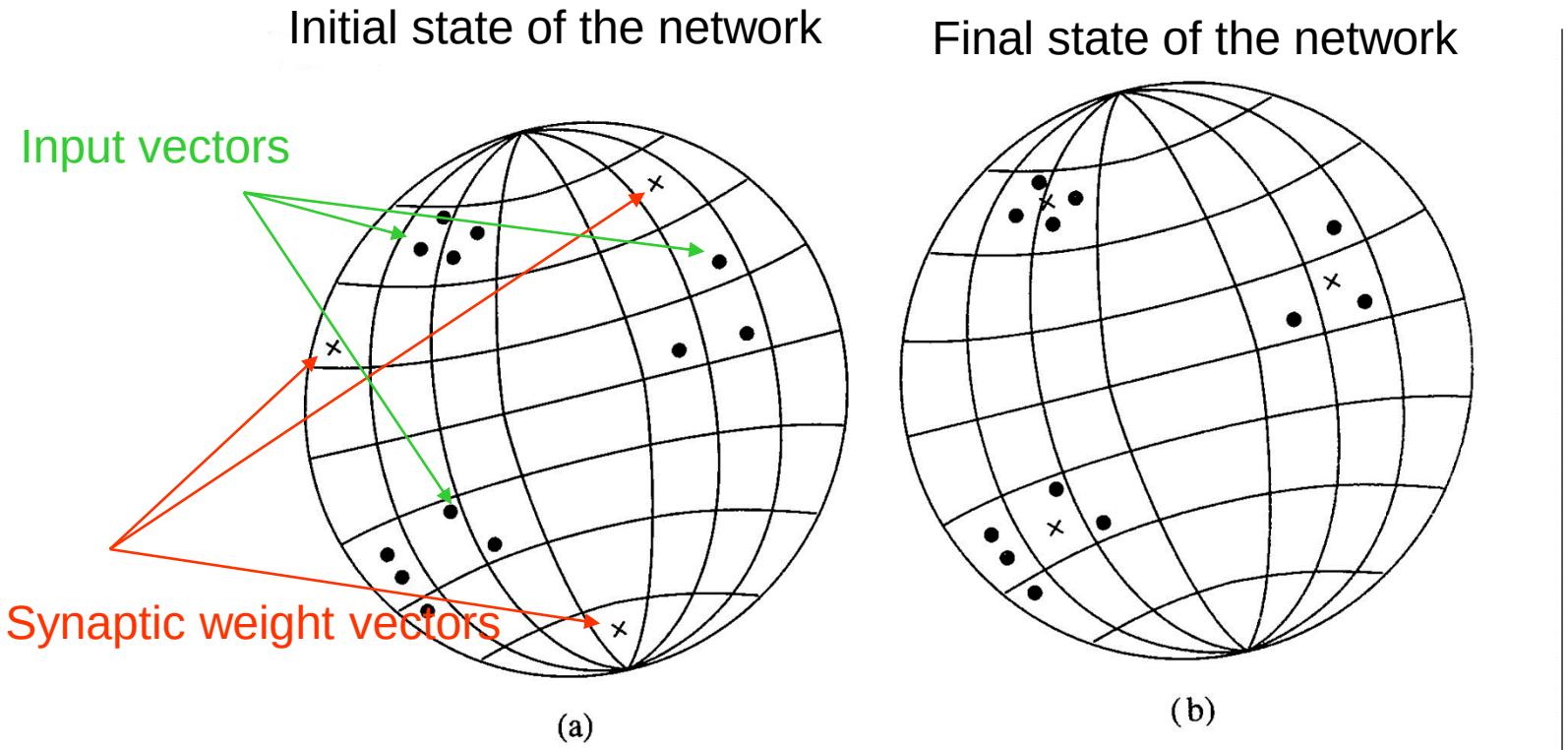


FIGURE 2.5 Geometric interpretation of the competitive learning process. The dots represent the input vectors, and the crosses represent the synaptic weight vectors of three output neurons. (a) Initial state of the network. (b) Final state of the network.

Boltzmann Learning

- In a Boltzmann machine the neurons constitute a recurrent structure, and they operate in a binary manner since they are either in an “on” state denoted by +1 or in an “off” state denoted by -1.
- The machine is characterized by an *energy function*, which is determined by the particular states occupied by the individual neurons of the machine.

$$E = -\frac{1}{2}\sum_{j}\sum_{k} w_{kj} x_k x_j$$

$j \neq k$

神經元j和神經元k間的權重

Neuron j的狀態

表示無self-feedback

Boltzmann Learning (cont.)

- The machine operates by choosing a neuron at random (for example, neuron k) at some step of the learning process,
- Then flipping the state of neuron k from state x_k to state $-x_k$ at some temperature T with probability.

$$p(x_k \rightarrow -x_k) = \frac{1}{1+\exp(-\Delta E_K / T)}$$

- If this rule is applied repeatedly, the machine will reach thermal equilibrium.

Boltzmann Learning (cont.)

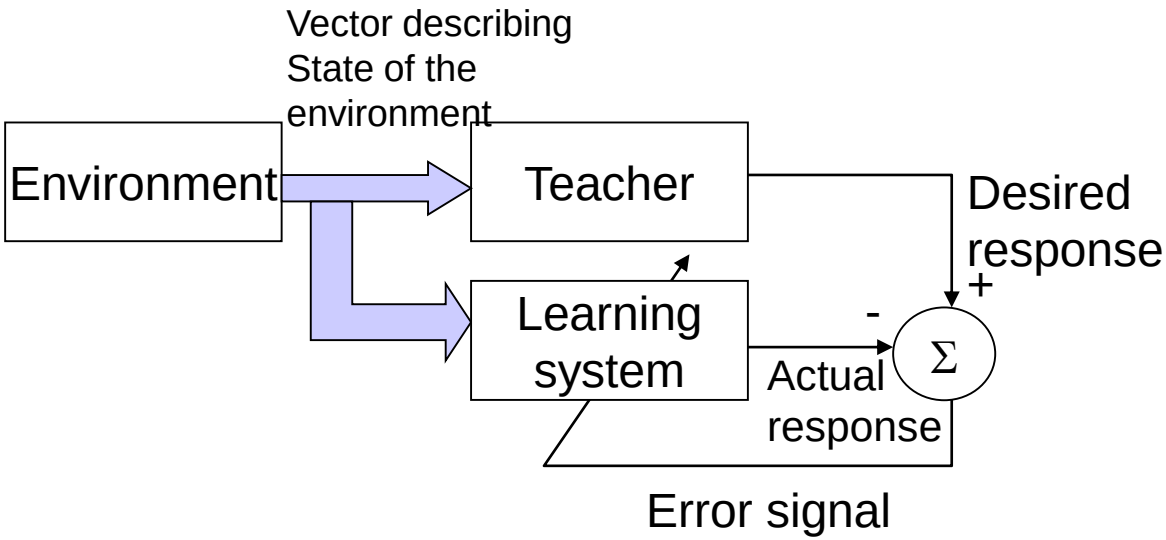
- The neurons of a Boltzmann machine partition into two functional groups:
  - Visible neuron
    - Provide an interface between the network and the environment in which it operates.
  - Hidden neuron
    - Always operate freely.
- There are two modes of operations to be considered:
  - Clamped condition
    - The visible neurons are all clamped onto specific states determined by the environment.
  - Free-running condition
    - All the neurons (visible and hidden) are allowed to operate freely.

Credit-assignment Problem

- The credit-assignment problem
  - Assigning credit or blame for overall outcomes to each of the internal decisions made by a learning machine and which contributed to those outcomes.
  - 可將credit-assignment problem分成兩個子問題
    - The assignment of credit for outcomes to actions.
      - Temporal credit-assignment problem, the instants of time when the actions that deserve credit were actually taken.
      - 當一學習機器採取許多action而獲得某一結果，我們必須決定這些action和這個結果的責任關係。
    - The assignment of credit for actions to internal decision.
      - Structural credit-assignment problem, assigning credit to the internal structures of actions generated by the system.
      - 對一個多成分學習機器(multi-component learning machine)我們需知道系統的哪一個成分應被修改，修改多少？以提升系統整體效能。
  - Multi-layer feedforward neural network的error-correction learning即是credit-assignment problem的範例。

Learning with a teacher

- Supervised learning
  - 藉由input-output examples來訓練網路
  - 網路參數的調整是在輸入向量和錯誤訊號(error signal)的影響下進行
  - Error signal定義成desired response和actual response間的差異
  - 參數的調整是step-by-step反覆的進行。
  - Error-performance surface, error surface
  - Gradient Steepest descent

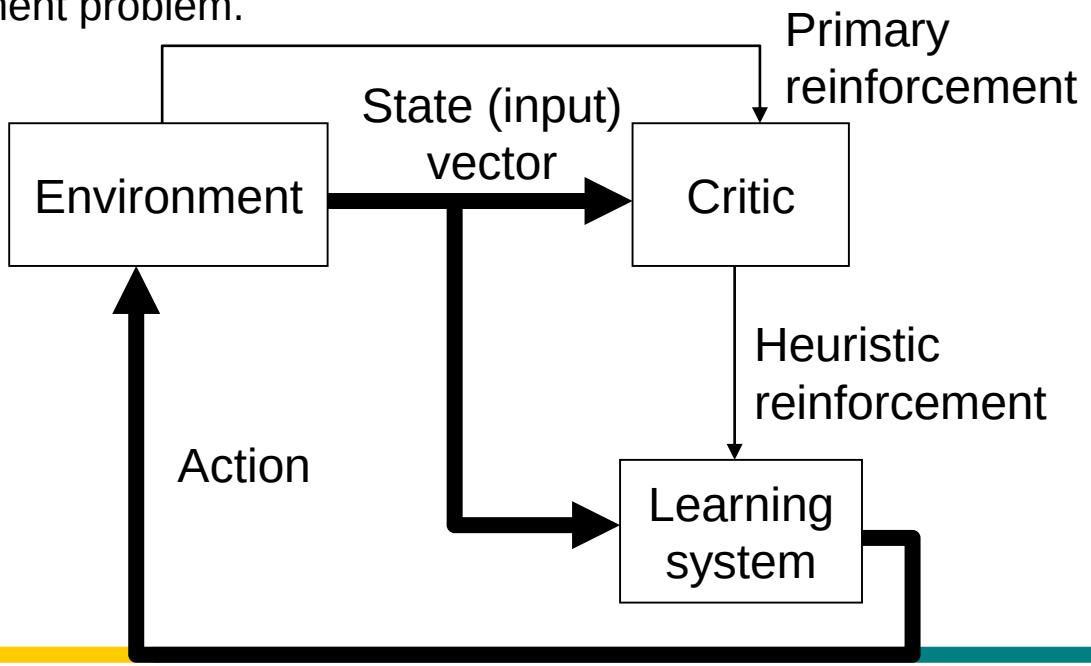


Learning without a teacher

- There is no teacher to oversee the learning process.
  - There are no labeled examples of the function to be learned by the network.
  - 有兩種典型：
    - 1. Reinforcement learning/Neurodynamic programming
      - The learning of an input-output mapping is performed through continued interaction with the environment in order to minimize a scalar index of performance.

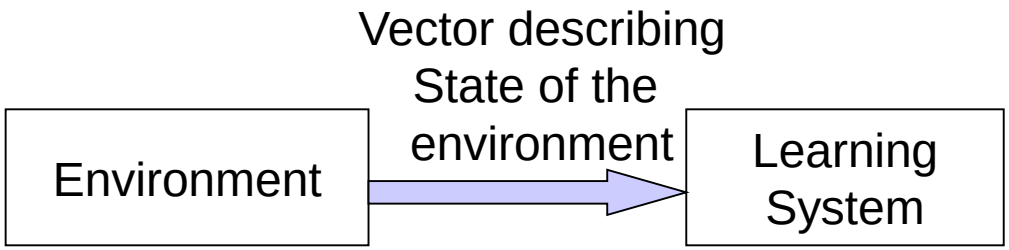
Block diagram of reinforcement learning

- Delayed reinforcement
  - 系統觀察一段時間的環境刺激，最後產生heuristic reinforcement signal.
  - 此種學習的目的在於將一cost-to-go function最小化，亦即將一連串的步驟行動的累積成本期望值最小化。
  - 在執行上有兩個難題
    - 在每個學習階段都沒有teacher提供desired response
    - Primary reinforcement 信號的取得意味著learning machine必須解決temporal credit assignment problem.



Learning without a teacher

- Unsupervised learning
  - There is no external teacher to oversee the learning process.
  - The free parameters of the network are optimized with respect to the task-independent measure.
  - Once the network has become tuned to the statistical regularities of the input data, it develops the ability to form internal representations for encoding features of the input.



Learning Tasks

- Pattern association
  - An associative memory is a brain-like distributed memory that learns by association.
  - Association有兩種形式
    - Auto-association
      - A neural network is required to store a set of patterns by repeatedly presenting them to the network.
      - The network is subsequently presented as partial description version of an original pattern.
      - The task is to retrieve (recall) that particular pattern
      - unsupervised
    - Hetero-association
      - An arbitrary set of input patterns is paired with another arbitrary set of output pattern.
      - supervised

Learning Tasks (cont.)

- 令$\mathbf{x}_k$為key pattern，$\mathbf{y}_k$為memorized pattern
- The pattern association 可表示成

$$\mathbf{x}_k \rightarrow \mathbf{y}_k \quad k = 1,2,...q \qquad \text{Eq.(2.18)}$$

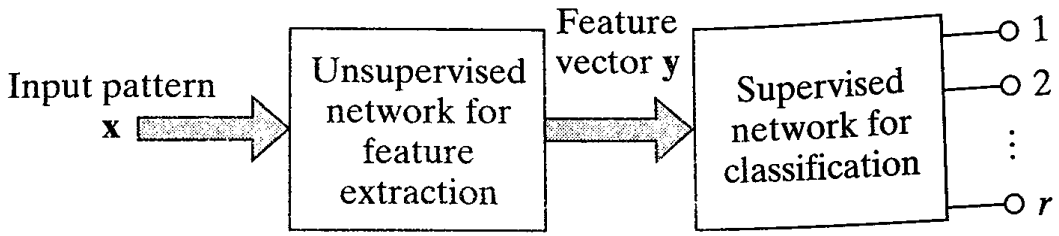
- Associative memory包含兩個階段
  - Storage phase
    - Training of the network in accordance with Eq.(2.18)
  - Recall phase
    - The retrieval of a memorized pattern in response to the presentation of a noisy or distorted version of a key pattern to the network.

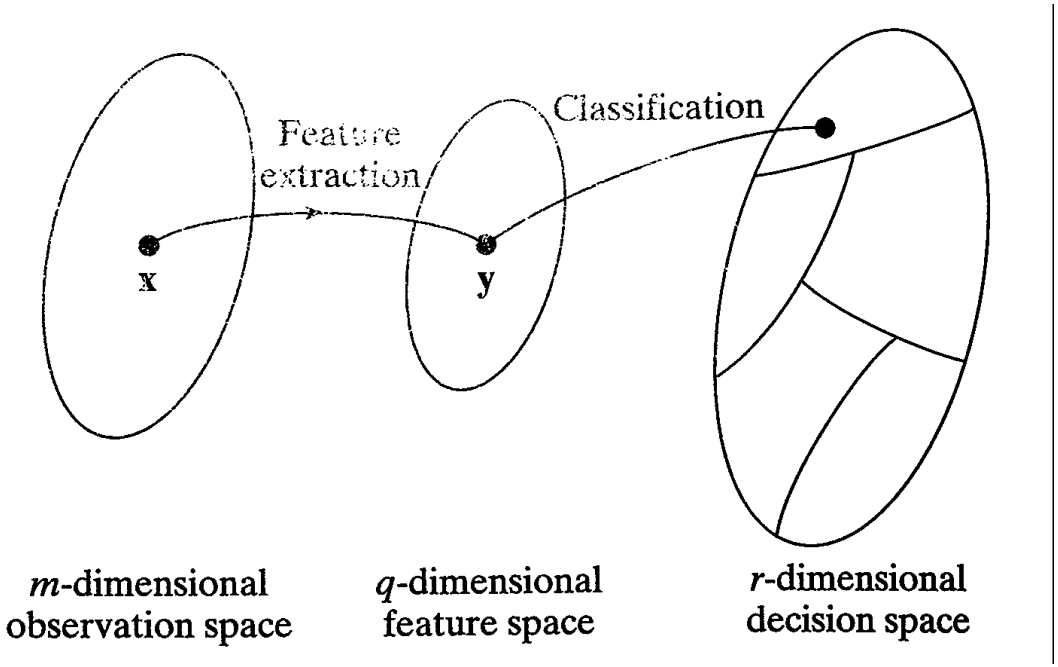
Learning Tasks (cont.)

- Pattern recognition
  - The process whereby a received pattern/signal assigned to one of a prescribed number of classes.
  - Patterns being represented by points in a multidimensional decision space.
  - The decision space is divided into regions, each one of which is associated with a class.
  - The decision boundaries are determined by the training process.
  - 一般的類神經網路的PR機器會採下列兩種形式
    - 一個非監督式網路進行特徵擷取，一個監督式網路進行分類(classification)
    - 使用監督式的Multilayer feedforward network。

Pattern classification

- The classical approach to pattern classification





Learning Tasks (cont.)

- Function Approximation
  - Consider a nonlinear input-output mapping described by the functional relationship

$$\mathbf{d} = \mathbf{f}(\mathbf{x})$$

where vector $\mathbf{x}$ is the input
vector $\mathbf{d}$ is the output
the function $\mathbf{f}(.)$ is assumed to be unknown

  - Given the set of labeled examples

$$\mathfrak{I} = \{(\mathbf{x}_i, \mathbf{d}_i)\}_{i=1}^{N}$$

Learning Tasks (cont.)

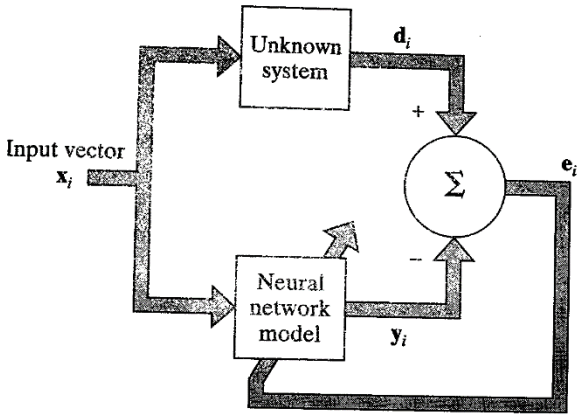
- The requirement is to design a neural network that approximates the unknown function **f**(.) such that the function **F**(.) describing the input-output mapping actually realized by the network is close enough to **f**(.) in a Euclidean sense over all inputs

$$\|\mathbf{F}(\mathbf{x}) - \mathbf{f}(\mathbf{x})\| < \varepsilon \text{ for all } \mathbf{x}$$

- 提供足夠多的training pattern,網路提供適當數目的自由參數,則可獲得一足夠小的近似誤差

Learning Tasks (cont.)

- The ability of a neural network to approximate an unknown input-output mapping may be exploited in two ways:
  - System identification
    - Unknown memoryless multiple input-multiple output
  - Inverse system
    - Known memoryless MIMO
    - To construct an inverse system that produces the vector $\mathbf{x}$ in response to the vector $\mathbf{d}$.



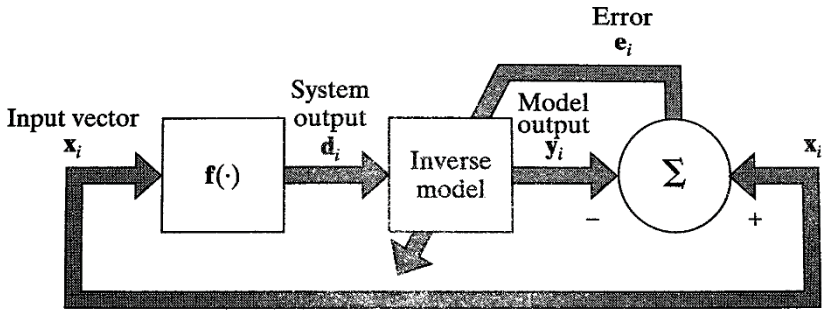
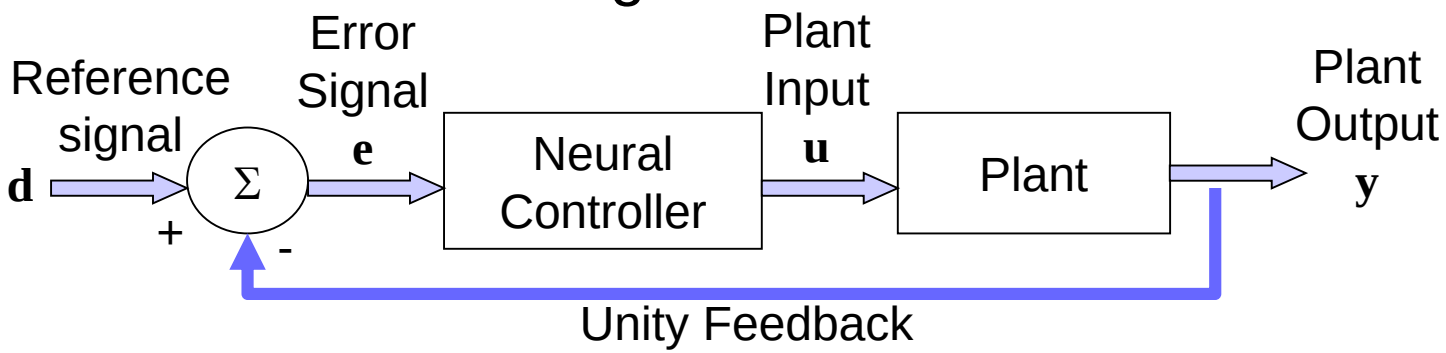


FIGURE 2.12 Block diagram of inverse system modeling.

Learning Tasks (cont.)

- Control
  - Feedback control system
    - The plant output is fed back directly to the input.
    - The plant output **y** is subtracted from a reference signal **d** supplied from an external source.
    - The error signal **e** is applied to a neural controller is used for adjusting free parameters
  - The main objective of the controller is to supply appropriate inputs to the plant to make its output **y** track the reference signal **d**



Learning Tasks (cont.)

- To perform adjustments on the free parameters of the plant in accordance with an error-correction learning algorithm, we need to know the Jacobian matrix.

$$\mathbf{J} = \left\{ \frac{\partial y_k}{\partial u_j} \right\}$$

where y_k is an element of the plant output $\mathbf{y}$ and u_j is an element of the plant input $\mathbf{u}$.

- The partial derivatives for various k and j depend on the operating point of the plant and are therefore not known.
- We may use one of the two approached to account for them
  - Indirect learning
    - 使用機器(plant)的實際I-O值，建立一個 NN能複製該機器的功能，依次使用該NN模型以獲的Jacobian matrix **J**的估計值，最後再以**J**來計算neural controller的修正量。
  - Direct learning
    - 以個別符號來近似Jacobian內的偏微分，她們的絕對值可作為neural controller的自由參數，因此可直接用來調整自由參數。

Learning Tasks (cont.)

- Filtering
  - A device or algorithm used to extract information about a prescribed quantity of interest from a set of noisy data.
  - 我們可使用filter來執行三種資訊處理的工作
    - Filtering
      - Extraction of information about a quantity of interest at discrete time n by using data measured up to and including time n.
    - Smoothing
      - The quantity of interest need not be available at time n, and data measured *later than* time n can be used in obtaining this information.
    - Prediction
      - To derive information about what the quantity of interest will be like at some time $n+n_0$ in the future.

Learning Tasks (cont.)

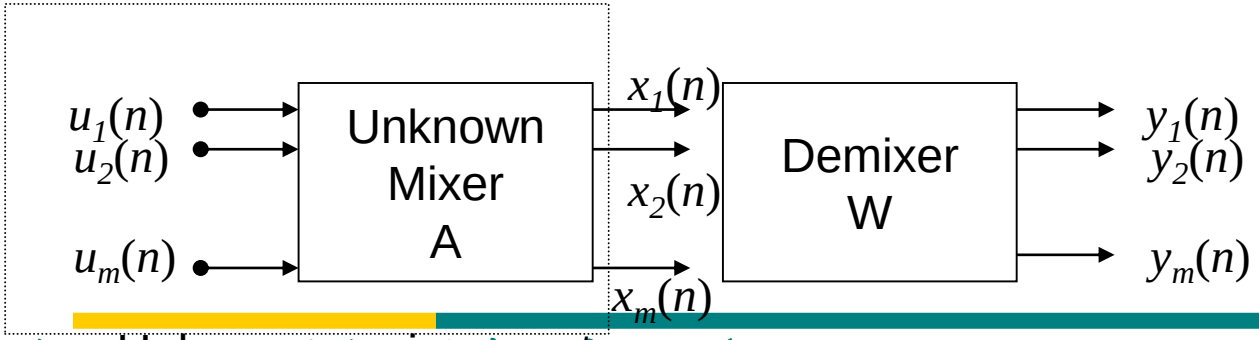
- Cocktail party problem
  - 在鷄尾酒會的吵雜環境中，說話的聲音夾雜各種雜訊，因此須有preattentive, preconscious analysis。
  - 在類神經網路中有個類似的問題稱為blind signal separation.
  - 假設某訊號經由一未知的感應器處理後得

$$\mathbf{x}(n) = \mathbf{A}\mathbf{u}(n)$$

其中　原始訊號　$\mathbf{u}(n) = [u_1(n), u_2(n), ..., u_m(n)]^T$

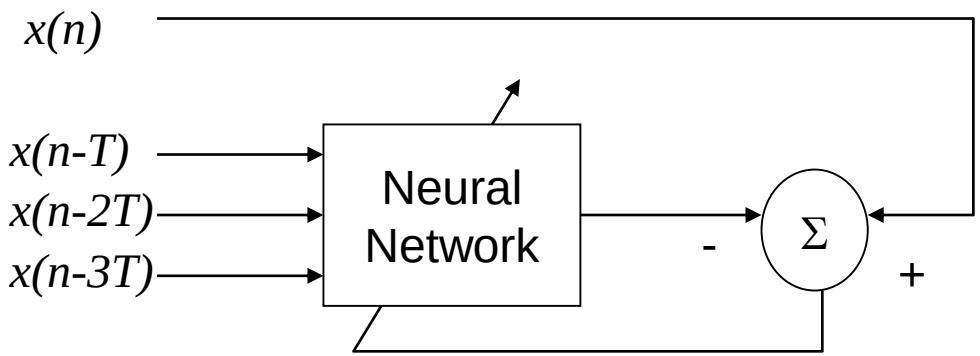
觀察得到的訊號　$\mathbf{x}(n) = [x_1(n), x_2(n), ..., x_m(n)]^T$

  - 目的在於使用非監督式的學習法來還原原始訊號



Learning Tasks (cont.)

- Prediction problem
  - Given past values of the process $x(n\text{-}T)$, $x(n\text{-}2T), \ldots, x(n\text{-}mT)$ to predict the present value $x(n)$ of a process.
  - Prediction may be solved by using error-correction learning in an unsupervised manner.



https://MIPL.yuntech.edu.tw

Learning Tasks (cont.)

- Beamforming
  - Beamforming is a signal processing technique used with arrays of transmitting or receiving transducers that control the directionality of, or sensitivity to, a radiation pattern.
  - When receiving a signal, beamforming can increase the receiver sensitivity in the direction of wanted signals and decrease the sensitivity in the direction of interference and noise.
  - When transmitting a signal, beamforming can increase the power in the direction the signal is to be sent.
  - Beamforming is a spatial form of filtering and is used to distinguish between the spatial properties of a target signal and background noise.
  - Beamforming is commonly used in radar and sonar system
    - The primary task is to detect and track a target of interest in the combined present of receiver noise and interfering signals.

Memory

- Memory refers to the relatively enduring neural alterations induced by the interaction of an organism with its environment.
- An activity pattern must initially be stored in memory through a learning process.
- Memory可分為
  - Short-term memory
    - A compilation of knowledge representing the current state of the environment
  - Long-term memory
    - Knowledge stored for a long time or permanently.

Associative Memory Networks

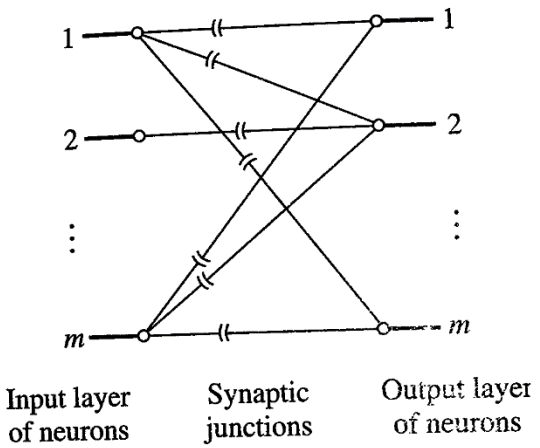
- 記憶容量(memory capability)對於資訊系統而言，記憶(remember)並且降低(reduce)所儲存的資訊量是很重要的。
- 儲存的資訊必須被適當的存放在網路的記憶體中，也就是說，給予一個關鍵的(key)輸入或刺激(stimulus)，將從關聯式記憶體(associative memory)中擷取記憶化的樣本(memorized pattern)，以適當的方式輸出。

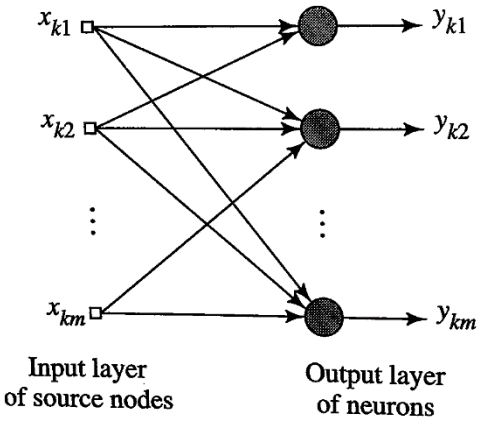
Associative Memory Networks (cont.)

- 在神經生物學的系統中，記憶(memory)的概念與神經因環境和有機結構的互動改變有關。如果變化不存在，則不會有記憶存在。
- 如果記憶是有用的，則它必須是可被存取的(從神經系統)，所以會發生學習(learning)，而且也可以擷取(retrieval)資訊。
- 一個樣本(pattern)可經由學習的過程(learning process)存在記憶體。

General Linear Distributed Associative Memory

- General Linear Distributed Associative Memory 的學習過程是給網路一個key input pattern (vector)，然後記憶體轉換這個向量(vector)成一個儲存(記憶)的pattern.





General Linear Distributed Associative Memory (cont.)

- Input (key input pattern)

$$\mathbf{x}_k = [x_{k1}, x_{k2}, ..., x_{km}]^T$$

- Output (memorized pattern)

$$\mathbf{y}_k = [y_{k1}, y_{k2}, ..., y_{km}]^T$$

- 對一個n-dimension的神經架構，可聯想(associate) h個pattern。$h<=n$。實際上是$h < n$。

General Linear Distributed Associative Memory (cont.)

- Key vector $\mathbf{x}_k$ 和記憶向量$\mathbf{y}_k$間的線性對應可寫成：

$$\mathbf{y}_k = \mathbf{W}(k)\mathbf{x}_k \tag{2.27}$$

其中$\mathbf{W}(k)$為權重矩陣(weight matrix)

- Memory matrix M
  - describes the sum of the weight matrices for every input/output pair. This can be written as

$$M = \sum_{k=1}^{q} W(k) \tag{2.32}$$

the memory matrix can be thought of as representing the collective experience.

$$M_k = M_{k-1} + W(k), \quad \text{for } k = 1,2,\ldots,h \tag{2.33}$$

General Linear Distributed Associative Memory (cont.)

- Association between the pattern $\mathbf{x_k}$ and $\mathbf{y_k}$

$$\mathbf{x}_k = [x_{k1}, x_{k2}, \ldots, x_{km}]^T$$
$$\mathbf{y}_k = [y_{k1}, y_{k2}, \ldots, y_{km}]^T$$

- $\mathbf{x_k}$和$\mathbf{y_k}$的關聯可表示成

$$\mathbf{y}_k = \mathbf{W}(k)\mathbf{x}_k, \quad k = 1,2,\ldots,q \tag{2.27}$$

其中$\mathbf{W}(k)$為權重矩陣

$$y_k = \sum_{j=1}^{m} w_{ij}(k)x_{kj}, \quad i = 1,2,\ldots,q \tag{2.28}$$

$$y_{ki} = [w_{i1}(k), w_{i2}(k), \ldots, w_{im}(k)] \begin{bmatrix} x_{k1} \\ x_{k2} \\ \cdot \\ \cdot \\ \cdot \\ x_{km} \end{bmatrix}, \quad i = 1,2,\ldots,m \tag{2.29}$$

General Linear Distributed Associative Memory (cont.)

m-by-1 stored vector $\mathbf{y}_k$

$$\begin{bmatrix} y_{k1} \\ y_{k2} \\ \cdot \\ \cdot \\ \cdot \\ y_{km} \end{bmatrix} = \begin{bmatrix} w_{11}(k) & w_{12}(k) & \ldots & w_{1m}(k) \\ w_{21}(k) & w_{22}(k) & \ldots & w_{2m}(k) \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ w_{m1}(k) & w_{m2}(k) & \ldots & w_{mm}(k) \end{bmatrix} \begin{bmatrix} x_{k1} \\ x_{k2} \\ \cdot \\ \cdot \\ \cdot \\ x_{km} \end{bmatrix} \tag{2.30}$$

m-by-*m* weight matrix $\mathbf{W}(k)$

$$\mathbf{W}(k) = \begin{bmatrix} w_{11}(k) & w_{12}(k) & \ldots & w_{1m}(k) \\ w_{21}(k) & w_{22}(k) & \ldots & w_{2m}(k) \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ w_{m1}(k) & w_{m2}(k) & \ldots & w_{mm}(k) \end{bmatrix} \tag{2.31}$$

m-by-*m* memory matrix

$$M = \sum_{k=1}^{q} \mathbf{W}(k) \tag{2.32}$$

左式亦可表示成

$$\mathbf{M}_k = \mathbf{M}_{k-1} + \mathbf{W}(k), \quad k = 1,2,\ldots,q \tag{2.33}$$

General Linear Distributed Associative Memory (cont.)

- The initial value $\mathbf{M_0}$ is zero
  - The synaptic weights in the memory are all initially zero.
- The final value $\mathbf{M_q}$ is identically equal to M as Eq(2.32).
- When $\mathbf{W}(k)$ is added to $\mathbf{M_{k-1}}$, the increment $\mathbf{W(k)}$ loses its distinct identity among the mixture of contributions that form $\mathbf{M_k}$.
- As the number q of stored patterns increases, the influence of a new pattern on the memory as a whole is progressively reduced.

Correlation Matrix Memory

- Estimation of the memory matrix $\mathbf{M}$
  - Suppose that the associative memory has learned the memory matrix $\mathbf{M}$ through the associations of key and memorized patterns described by $\mathbf{x}_k \rightarrow \mathbf{y}_k$.
  - We may denoting an estimate of the memory matrix $\mathbf{M}$ in terms of these patterns as

$$\hat{M} = \sum_{k=1}^{q} y_k x_k^T \qquad (2.34)$$

  - Outer product of the key pattern and the memorized pattern.
  - The local learning process may be viewed as a generalization of Hebb's postulate of learning.
    - Also referred to as the *outer product rule* in recognition of the matrix operation used to construct the memory matrix $\mathbf{M}$.
    - *Correlation matrix memory*

Correlation Matrix Memory (cont.)

- Eq(2.34) can be represented as matrix form as

$$\hat{\mathbf{M}} = \begin{bmatrix} \mathbf{y}_1 & \mathbf{y}_2, \ldots \mathbf{y}_q \end{bmatrix} \begin{bmatrix} \mathbf{x}_1^t \\ \mathbf{x}_2^t \\ \cdot \\ \cdot \\ \cdot \\ \mathbf{x}_q^t \end{bmatrix} = \mathbf{Y}\mathbf{X}^t \qquad (2.35)$$

$$\mathbf{X} = \left[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_q\right] \qquad (2.36)$$

$$\mathbf{Y} = \left[\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_q\right] \qquad (2.37)$$

- The matrix $\mathbf{X}$ is an m-by-q matrix composed of the entire set of key patterns used in the learning process,
  - called the *key matrix*.
- The matrix $\mathbf{Y}$ is an m-by-q matrix composed of the corresponding set of memorized patterns
  - called the *memorized matrix*
- Equation (2.35) may be restructured as

$$\hat{\mathbf{M}}_k = \hat{\mathbf{M}}_{k-1} + \mathbf{y}_k \mathbf{x}_k^T, \quad \text{for } k = 1,2,\ldots,q \qquad (2.38)$$

- Comparing Eq(2.38) with Eq(2.33), the outer product represents an estimate of the weight matrix $\mathbf{W}(k)$.

Correlation Matrix Memory (cont.)

- Recall
  - 當某key pattern輸入時，希望memory matrix能recall適當的memorized pattern，假設estimated memory matrix已經由(2.34)學習了m次$(\mathbf{x}_k \rightarrow \mathbf{y}_k)$，則輸入第$j$個key input $\mathbf{x}_j$，欲求memorized $\mathbf{y}$

$$\mathbf{y} = \hat{M}\mathbf{x}_j = \sum_{k=1}^{m} \mathbf{y}_k \mathbf{x}_k^T \mathbf{x}_j = \sum_{k=1}^{m} (\mathbf{x}_k^T \mathbf{x}_j)\mathbf{y}_k \qquad (2.39,\ 2.40)$$

Eq. (2.39) can be rewritten as (將$x^T_j x_j$內積獨立出來)

$$\mathbf{y} = (\mathbf{x}_j^T \mathbf{x}_j)\mathbf{y}_j + \sum_{\substack{k=1 \\ k \neq j}}^{m} (\mathbf{x}_k^T \mathbf{x}_j)\mathbf{y}_k \qquad (2.41)$$

Assume that each key input vector has normalized unit length.

$$\mathbf{x}_k^T \mathbf{x}_k = 1 \quad \text{for} \quad k = 1,2,\ldots,q \qquad (2.42)$$

Correlation Matrix Memory (cont.)

- Eq.(2.41) can be represented as

$$\mathbf{y} = \mathbf{y}_j + \sum_{\substack{k=1 \\ k \neq j}}^{m} (\mathbf{x}_k^T \mathbf{x}_j)\mathbf{y}_k = \mathbf{y}_j + \mathbf{v}_j \qquad (2.43)$$

Desired response

Noise, crosstalk

where

$$\mathbf{v}_j = \sum_{\substack{k=1 \\ k \neq j}}^{m} (\mathbf{x}_k^T \mathbf{x}_j)\mathbf{y}_k \qquad (2.44)$$

- Eq(2.43)右手邊第一項為 "desired" $\mathbf{y}_j$.
- 第二項為noise vector, 由key vector $\mathbf{x}_j$ 和所有其他儲存在記憶體中的key vector的*crosstalk*所造成。

Correlation Matrix Memory (cont.)

- 在線性訊號空間中，兩向量的夾角可定義成

$$\cos(\mathbf{x}_k, \mathbf{x}_j) = \frac{\mathbf{x}_k^T \mathbf{x}_j}{\|\mathbf{x}_k\| \|\mathbf{x}_j\|} \tag{2.45}$$

其中$\|\mathbf{x}_k\|$表示向量$\mathbf{x}_k$的Euclidean norm

$$\begin{aligned} \|\mathbf{x}_k\| &= (\mathbf{x}_k^T \mathbf{x}_k)^{1/2} \\ &= E_k^{1/2} \end{aligned} \tag{2.46}$$

- 因此，根據Eq(2.42) ，Eq(2.45)可簡化寫成

$$\cos(\mathbf{x}_k, \mathbf{x}_j) = \mathbf{x}_k^T \mathbf{x}_j \tag{2.47}$$

- 同樣的Eq(2.44)可改寫成

$$\mathbf{v}_j = \sum_{\substack{k=1 \\ k \neq j}}^{m} \cos(\mathbf{x}_k, \mathbf{x}_j) \mathbf{y}_k \tag{2.48}$$

Correlation Matrix Memory (cont.)

- 從Eq(2.49)可發現，若key vectors為orthogonal
  則 $\cos(\mathbf{x}_k, \mathbf{x}_j) = 0, \quad k \neq 1$
- 因此，noise vector $\mathbf{v}_j$將等於零。
- 從(2.44)當網路引進associated key pattern ($\mathbf{y}=\mathbf{y}_j$, $\mathbf{z}_j=0$)
  - 若$\mathbf{y}=\mathbf{y}_j$表示輸入$\mathbf{x}_j$可精確的得到其對應的輸出$\mathbf{y}_j$。
  - 若z_j不為0，表示有noise或crosstalk。
- 若key input之間為orthogonal，則crosstalk為0
  - 因此可得到perfect memorized pattern。

$$\mathbf{x}_k^T \mathbf{x}_j = \begin{cases} 1, & k = j \\ 0, & k \neq j \end{cases} \qquad (2.50)$$

Correlation Matrix Memory (cont.)

- What is the limit on the storage capacity of the associative memory?
  - The storage capacity of the associative memory is

$$\rho\left(\hat{M}\right) \leq m$$

    - The storage limit depends on the rank of the memory matrix.
      - The number of independent columns (rows) of the matrix.
    - The number of patterns that can be reliably stored in a correlation matrix memory can never exceed the input space dimensionality.

Correlation Matrix Memory (cont.)

- 在實際應用上,key pattern之間很難完全orthogonal，因此Eq(2.34)的記憶體矩陣會存在許多錯誤。也就是該記憶體會經常需要辨識或聯想未曾看過的pattern
- 假設一組key patterns和相對應的記憶pattern

$$\{\mathbf{x}_{key}\}: \mathbf{x}_1, \mathbf{x}_{2,...,}\mathbf{x}_q$$

$$\{\mathbf{y}_{mem}\}: \mathbf{y}_1, \mathbf{y}_{2,...,}\mathbf{y}_q$$

- community
  - The community of the set of patterns $\{\mathbf{x}_{key}\}$ as the lower bound on the inner products $\mathbf{x}_k^T\mathbf{x}_j$ of any two patterns $\mathbf{x}_j$ and $\mathbf{x}_k$ in the set.
  - 若已從一組 key vector $\{\mathbf{x}_{key}\}$及其對應的memorized pattern $\{\mathbf{y}_{kmem}\}$，根據Eq(2.34)獲得$\mathbf{M}$(head)
  - 若從key vector中選取一個輸入，根據Eq(2.39)可獲得記憶體的輸出$\mathbf{y}$

$$\mathbf{x}_k^T\mathbf{x}_j \geq \gamma \quad \text{for } k \neq j$$

Correlation Matrix Memory (cont.)

- Example 2.1
  - Autoassociative memory

    The memory is trained with three key vectors:

$$x_1 = \begin{bmatrix} -0.3333 \\ 0.7778 \\ 0.5329 \end{bmatrix} \quad x_2 = \begin{bmatrix} 0.4444 \\ -0.5556 \\ 0.7027 \end{bmatrix} \quad x_3 = \begin{bmatrix} 0.4969 \\ 0.6667 \\ 0.5556 \end{bmatrix}$$

Each of these vectors has unit length.

Correlation Matrix Memory (cont.)

- The angles between these vectors:

$$\theta_{12} = \cos^{-1}\frac{x_1 x_2^T}{\|x_1\|_2 \|x_2\|_2} = 101.9^o$$

$$\theta_{13} = \cos^{-1}\frac{x_1 x_3^T}{\|x_1\|_2 \|x_3\|_2} = 49.5^o$$

$$\theta_{23} = \cos^{-1}\frac{x_2 x_3^T}{\|x_2\|_2 \|x_3\|_2} = 76.1^o$$

- These three vectors are far from being mutually orthogonal.

Correlation Matrix Memory (cont.)

- The memory matrix of the autoassociative network

$$\hat{M} = x_1 x_1^T + x_2 x_2^T + x_3 x_3^T = \begin{bmatrix} 0.5555 & -0.1749 & 0.4107 \\ -0.1749 & 1.3582 & 0.3945 \\ 0.4107 & 0.3945 & 1.0865 \end{bmatrix}$$

- Using (3.19),

$$x_1 = \begin{bmatrix} -0.3333 \\ 0.7778 \\ 0.5329 \end{bmatrix} \quad x_2 = \begin{bmatrix} 0.4444 \\ -0.5556 \\ 0.7027 \end{bmatrix} \quad x_3 = \begin{bmatrix} 0.4969 \\ 0.6667 \\ 0.5556 \end{bmatrix}$$

the estimate of the input key patterns

$$\hat{x}_1 = \hat{M}x_1 = \begin{bmatrix} -0.1023 \\ 1.3249 \\ 0.7489 \end{bmatrix} \quad \hat{x}_2 = \hat{M}x_2 = \begin{bmatrix} 0.6326 \\ -0.5551 \\ 0.7268 \end{bmatrix} \quad \hat{x}_3 = \hat{M}x_3 = \begin{bmatrix} 0.3876 \\ 1.0378 \\ 1.0707 \end{bmatrix}$$

Correlation Matrix Memory (cont.)

- The Euclidean distance of the response vector from each of the key vectors.

$$\delta_{11} = \|x_1 - \hat{x}_1\|_2 = 0.6319 \qquad \delta_{21} = \|x_2 - \hat{x}_1\|_2 = 1.9589$$

$$\delta_{31} = \|x_3 - \hat{x}_1\|_2 = 0.9108$$

$$\delta_{12} = \|x_1 - \hat{x}_2\|_2 = 1.6575 \qquad \delta_{22} = \|x_2 - \hat{x}_2\|_2 = 0.1898$$

$$\delta_{32} = \|x_3 - \hat{x}_2\|_2 = 1.2412$$

$$\delta_{13} = \|x_1 - \hat{x}_3\|_2 = 0.9363 \qquad \delta_{23} = \|x_2 - \hat{x}_3\|_2 = 1.6363$$

$$\delta_{33} = \|x_3 - \hat{x}_3\|_2 = 0.6442$$

有較小的error

Correlation Matrix Memory (cont.)

- Another key vector with unit length

$$x_1 = \begin{bmatrix} 0.1309 \\ -0.9779 \\ -0.1629 \end{bmatrix} \qquad x_2 = \begin{bmatrix} -0.7548 \\ 0.0587 \\ -0.6533 \end{bmatrix} \qquad x_3 = \begin{bmatrix} -0.6354 \\ -0.2370 \\ 0.7349 \end{bmatrix}$$

- The angles between these vectors

$$\theta_{12} = \cos^{-1} \frac{x_1 x_2^T}{\|x_1\|_2 \|x_2\|_2} = 92.9^o$$

$$\theta_{13} = \cos^{-1} \frac{x_1 x_3^T}{\|x_1\|_2 \|x_3\|_2} = 88.3^o$$

$$\theta_{23} = \cos^{-1} \frac{x_2 x_3^T}{\|x_2\|_2 \|x_3\|_2} = 90.8^o$$

Correlation Matrix Memory (cont.)

- The memory matrix is

$$\hat{M} = x_1 x_1^T + x_2 x_2^T + x_3 x_3^T = \begin{bmatrix} 0.9906 & -0.0217 & 0.0048 \\ -0.0217 & 1.0159 & -0.0532 \\ 0.0048 & -0.0532 & 0.9934 \end{bmatrix}$$

- The estimate of input key patterns

$$\hat{x}_1 = \hat{M}x_1 = \begin{bmatrix} 0.1501 \\ -0.9876 \\ -0.1092 \end{bmatrix} \quad \hat{x}_2 = \hat{M}x_2 = \begin{bmatrix} -0.7521 \\ 0.1108 \\ -0.6558 \end{bmatrix} \quad \hat{x}_3 = \hat{M}x_3 = \begin{bmatrix} -0.6207 \\ -0.2661 \\ 0.7396 \end{bmatrix}$$

Correlation Matrix Memory (cont.)

- The Euclidean distance of the response vector from each of the key vectors.

$$\delta_{11} = \|x_1 - \hat{x}_1\|_2 = 0.0579 \qquad \delta_{21} = \|x_2 - \hat{x}_1\|_2 = 1.4865$$

$$\delta_{31} = \|x_3 - \hat{x}_1\|_2 = 1.3758$$

$$\delta_{12} = \|x_1 - \hat{x}_2\|_2 = 1.4859 \qquad \delta_{22} = \|x_2 - \hat{x}_2\|_2 = 0.0522$$

$$\delta_{32} = \|x_3 - \hat{x}_2\|_2 = 1.4382$$

$$\delta_{13} = \|x_1 - \hat{x}_3\|_2 = 1.3734 \qquad \delta_{23} = \|x_2 - \hat{x}_3\|_2 = 1.4365$$

$$\delta_{33} = \|x_3 - \hat{x}_3\|_2 = 0.0329$$

Lower error。

Correlation Matrix Memory (cont.)

- An error correction approach for correlation matrix memories
  - The drawback of associative memory is the relatively large number of errors that can occur during recall.
  - The simplest correlation matrix memory has no provision for correcting errors
    - Lack of feedback from the output to input.

Correlation Matrix Memory (cont.)

- The objective of the error correction approach is to have the associative memory reconstruct the memorized pattern in an optimal sense.
- The error vector is defined as

$$e_k(\tau) = y_k - \hat{M}(\tau)x_k \tag{3.33}$$

where y_k is the desired pattern with the key input pattern x_k.

- The discrete-time learning rule based on steepest descent

$$\hat{M}(\tau+1) = \hat{M}(\tau) - \mu\nabla_{\hat{M}}\varepsilon(\hat{M}) \tag{3.34}$$

Correlation Matrix Memory (cont.)

- where the energy function is defined as

$$\varepsilon(\hat{M}) = \frac{1}{2}\|e_k\|_2^2 = \frac{1}{2}\left\|y_k - \hat{M}(\tau)x_k\right\|_2^2 \qquad (3.35)$$

- Computing the gradient of (3.35)

$$\nabla_{\hat{M}}\varepsilon(\hat{M}) = -y_k x_k^T + \hat{M}x_k x_k^T \qquad (3.36)$$

- 將(3.36)代入(3.34)

$$\hat{M}(\tau+1) = \hat{M}(\tau) + \mu[y_k - \hat{M}(\tau)x_k]x_k^T \qquad (3.37)$$

Error vector

Correlation Matrix Memory (cont.)

- (3.37)式的第二項中的中括號的內容，為 error vector(參見(3.33)) ，因此具有error correction的效果。
- 將(3.37)展開可改寫成

$$\hat{M}(\tau+1)=\hat{M}(\tau)+\mu y_k x_k^T-\mu\hat{M}(\tau)x_k x_k^T \qquad (3.38)$$

和(3.7)比較起來，多了第三項。

$$\hat{M}_k=\hat{M}_{k-1}+y_k x_k^T,\quad \text{for } k=1,2,\ldots,h \qquad (3.7)$$

- (3.37)式基於error correction的監督式學習是對h個association的每個不同pattern重複訓練所得到。

$$x_k \rightarrow y_k,\quad \text{for } k=1,2,\ldots,h \qquad (3.39)$$

Correlation Matrix Memory (cont.)

- Selecting the learning rate parameter (μ)
  - Fixed learning rate
  - Adjustable learning rate with respect to time
  - For each association, the iterative adjustments to the memory matrix (3.37) continue until the error vector e_k (3.33) becomes negligibly small.
- The initial memory matrix
  - The initial memory matrix $\boldsymbol{M}(0)=0$.
- 和(2.29)比較，發現(3.37)也是LMS。

$$\hat{M}(\tau+1)=\hat{M}(\tau)+\mu[y_k-\hat{M}(\tau)x_k]x_k^T \tag{3.37}$$

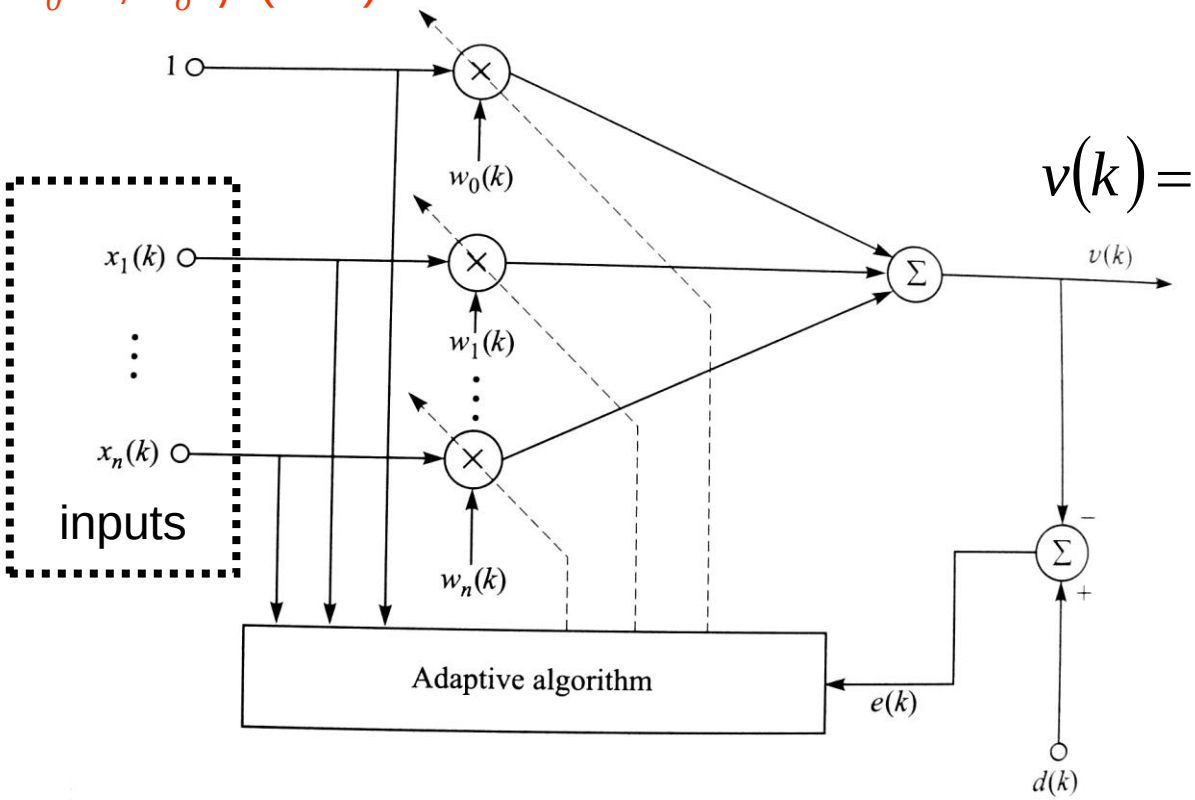
$$w(k+1)=w(k)+\mu\left[-\nabla_w J(w)\right]=w(k)+\mu e(k)x(k) \tag{2.29}$$

Adaline

- Least-Mean-Square (LMS) Algorithm
  - Delta rule
  - The LMS is an adaptive algorithm that computes adjustments of the neuron synaptic weights.
  - The algorithm is based on the method of steepest decent.
  - It adjusts the neuron weights to minimize the mean square error between the inner product of the weight vector with the input vector and the desired output of the neuron.
- Adaline (adaptive linear element)
  - A single neuron whose synaptic weights are updated according to the LMS algorithm.

Simple adaptive linear combiner

x_0=1, w_o=β (bias)



$$v(k) = x^T(k)w(k) = w^T(k)x(k)$$

Simple adaptive linear combiner

- The difference between the desired response and the network response is

$$e(k) = d(k) - v(k) = d(k) - \boldsymbol{w}^T(k)\boldsymbol{x}(k) \tag{2.22}$$

- The MSE criterion can be written as

$$J(\boldsymbol{w}) = \frac{1}{2}E\{e^2(k)\} = \frac{1}{2}E\left\{\left[d(k) - \boldsymbol{w}^T(k)\boldsymbol{x}(k)\right]^2\right\} \tag{2.23}$$

- Expanding Eq(2.23)

$$J(w) = \frac{1}{2}E\{d^2(k)\} - E\{d(k)\boldsymbol{x}^T(k)\}\boldsymbol{w}(k) + \frac{1}{2}\boldsymbol{w}^T(k)E\{\boldsymbol{x}(k)\boldsymbol{x}^T(k)\}\boldsymbol{w}(k) \tag{2.24}$$

$$= \frac{1}{2}E\{d^2(k)\} - \boldsymbol{p}^T\boldsymbol{w}(k) + \frac{1}{2}\boldsymbol{w}^T(k)\boldsymbol{C}_x\boldsymbol{w}(k) \tag{2.25}$$

Simple adaptive linear combiner

- Cross correlation vector between the desired response and the input patterns

$$\boldsymbol{p} = E\{d(k)\boldsymbol{x}(k)\}$$

- Covariance matrix for the input pattern

$$\boldsymbol{C}_x = E\{\boldsymbol{x}(k)\boldsymbol{x}^T(k)\}$$

- $J(\boldsymbol{w})$的MSE表面有一個最小值(minimum)，因此計算梯度等於零的權重值

$$\nabla_w J(\boldsymbol{w}) = \frac{\partial J(\boldsymbol{w})}{\partial \boldsymbol{w}} = -\boldsymbol{p} + \boldsymbol{C}_x \boldsymbol{w}(k) = 0 \qquad (2.26)$$

- 因此，最佳的權重值為

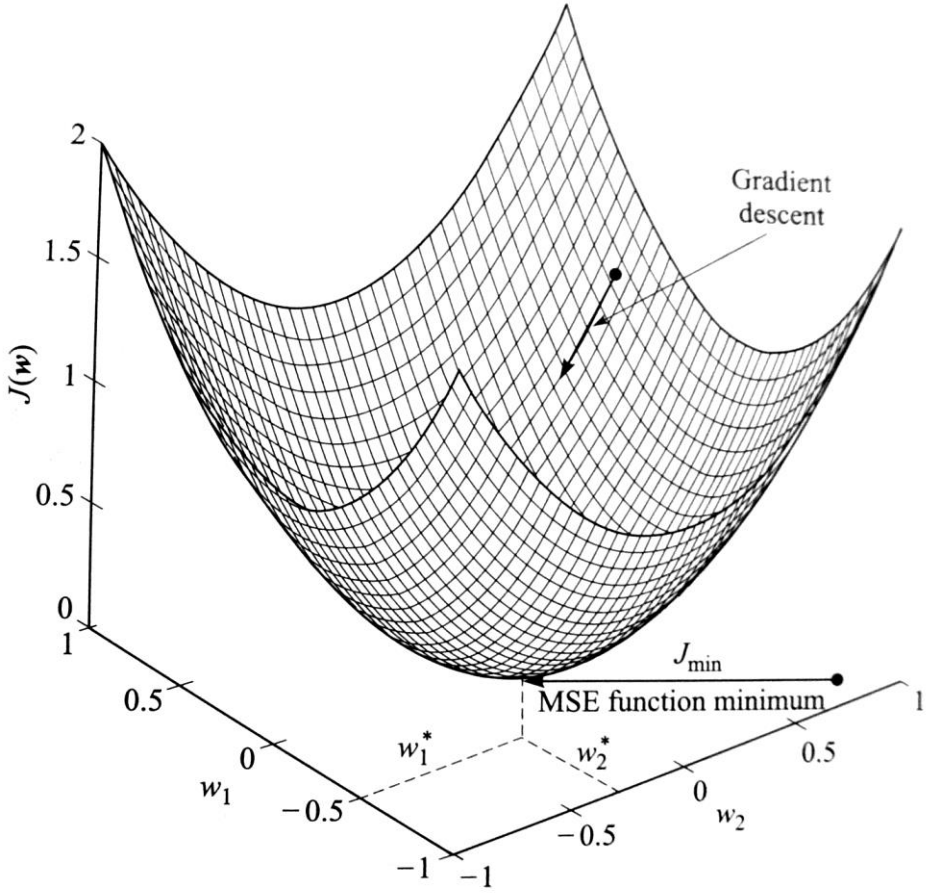
$$\boldsymbol{w}^* = \boldsymbol{C}_x^{-1}\boldsymbol{p} \qquad (2.27)$$

The LMS Algorithm

- 上式的兩個限制
  - 求解covariance matrix的反矩陣很費時
  - 不適合即時的修正權重，因為在大部分情況，covariance matrix和cross correlation vector無法事先知道。
- To solve these problems, Widow and Hoff proposed the LMS algorithm
  - To obtain the optimal values of the synaptic weights when $J(w)$ is minimum.
  - Search the error surface using a *gradient descent* method to find the minimum value.
  - We can reach the bottom of the error surface by changing the weights in the direction of the *negative gradient* of the surface.

The LMS Algorithm

Typical MSE surface of an adaptive linear combiner



The LMS Algorithm

- Because the gradient on the surface cannot be computed without knowledge of the input covariance matrix and the cross-correlation vector, these must be estimated during an iterative procedure.
- Estimation of the MSE gradient surface can be obtained by taking the gradient of the instantaneous error surface.
- The gradient of $J(\boldsymbol{w})$ approximated as

$$\nabla_w J(\boldsymbol{w}) \approx \frac{1}{2}\frac{\partial e^2(k)}{\partial \boldsymbol{w}}\bigg|_{w=w(k)} = -e(k)\boldsymbol{x}(k) \qquad (2.28)$$

- The learning rule for updating the weights using the steepest descent gradients method as

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) + \mu\left[-\nabla_w J(\boldsymbol{w})\right] = \boldsymbol{w}(k) + \mu e(k)\boldsymbol{x}(k) \qquad (2.29)$$

Learning rate specifies the magnitude of the update step for the weights in the negative gradient direction.

The LMS Algorithm

- If the value of μ is chosen to be too small, the learning algorithm will modify the weights slowly and a relatively large number of iterations will be required.
- If the value of μ is set too large, the learning rule can become numerically unstable leading to the weights not converging.

The LMS Algorithm

- The scalar form of the LMS algorithm can be written from (2.22) and (2.29)

$$e(k) = d(k) - \sum_{h=1}^{n} w_h(k) x_h(k) \qquad (2.30)$$

$$w_i(k+1) = w_i(k) + \mu e(k) x_i(k) \qquad (2.31)$$

- From (2.29) and (2.31), we should set an appropriate learning rate range to maintain the stability of the network. (Haykin,1996)

$$0 < \mu < \frac{2}{\lambda_{\max}}$$

The largest eigenvalue of the input covariance matrix $\boldsymbol{C}_x$

The LMS Algorithm

- 為使LMS收斂的最小容忍的穩定性，可接受的learning rate可限定在

$$0 < \mu < \frac{2}{trace\{\mathbf{C}_x\}} \tag{2.33}$$

- (2.33)式是一個近似的合理解法，因為

$$trace\{\mathbf{C}_x\} = \sum_{h=1}^{n} \lambda_h = \sum_{h=1}^{n} c_{xhh} \geq \lambda_{\max} \tag{2.34}$$

The LMS Algorithm

- 從(2.32) 、(2.33)式知道，learning rate的決定，至少得計算輸入樣本的covariance matrix，在實際的應用上是很難達到的。
- 即使可以得到，這種固定learning rate在結果的精確度上是有問題的。
- 因此，Robbin's and Monro's root-finding algorithm提出了，隨時間變動learning rate的方法。(Stochastic approximation)

$$\mu(k) = \frac{\kappa}{k} \qquad (2.35)$$

where κ is a very small constant.

- 缺點：learning rate減低的速度太快。

The LMS Algorithm

- 理想的做法應該是在學習的過程中，learning rate μ應該在訓練的開始時有較大的值，然後逐漸降低。(Schedule-type adjustment)
- Darken and Moody
  - Search-then converge algorithm
    - Search phase: μ is relatively large and almost constant.
    - Converge phase: μ is decrease exponentially to zero.

$$\mu(k) = \frac{\mu_0}{1 + k/\tau} \qquad (2.36)$$

    - $\mu_0 > 0$ and $\tau >> 1$, typically $100 <= \tau <= 500$
  - These methods of adjusting the learning rate are commonly called *learning rate schedules*.

The LMS Algorithm

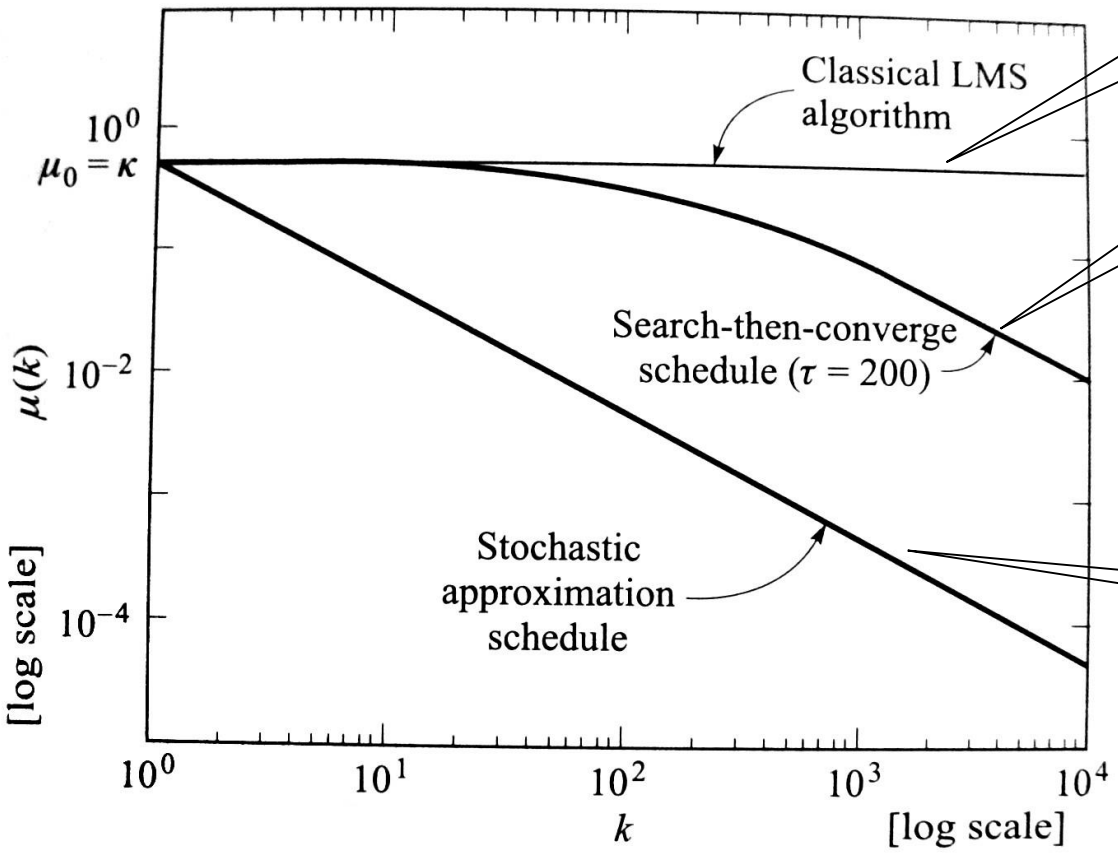
- Adaptive normalization approach (non-schedule-type)
  - μ is adjusted according to the input data every time step

$$\mu(k) = \frac{\mu_0}{\|x(k)\|_2^2} \tag{2.37}$$

  - where μ_0 is a fixed constant.
  - Stability is guaranteed if $0 < \mu_0 < 2$; the practical range is $0.1 <= \mu_0 <= 1$

The LMS Algorithm

- Comparison of two learning rate schedules: stochastic approximation schedule and the search-then-converge schedule.



μ is a constant

Eq.(2.36)

Eq.(2.35)

Summary of the LMS algorithm

- Step 1: set $k=1$, initialize the synaptic weight vector $w(k=1)$, and select values for μ_0 and τ.
- Step 2: Compute the learning rate parameter

$$\mu(k) = \frac{\mu_0}{1 + k/\tau}$$

- Step 3: Computer the error

$$e(k) = d(k) - \sum_{h=1}^{n} w_h(k) x_h(k)$$

- Step 4: Update the synaptic weights

$$w_i(k+1) = w_i(k) + \mu(k) e(k) x_i(k)$$

- Step 5: If convergence is achieved, stop; else set $k=k+1$, then go to step 2.

Adaptation

- Stationary environment
  - The essential statistics of the environment can be learned by the network under the supervision of a teacher.
  - The learning system relies on memory to recall and exploit past experiences.
- Non-stationary
  - The statistical parameters of the information-bearing signals generated by the environment vary with time.
  - It is desirable for a neural network to continually adapt its free parameters to variations in the incoming signals in a real-time fashion.
  - Continuous learning, learning-on-the-fly
    - The learning process encountered in an adaptive system never stops, with learning going on while signal processing is being performed by the system.
    - Ex. Linear adaptive filter
    - How can a neural network adapt its behavior to the varying temporal structure of the incoming signals in its behavioral space?
      - 在一個足夠小的時間周期中，nonstationary process通常變化的非常緩慢，也就是說statistical characteristic幾乎不變，可視為pseudostationary.

Statistical Nature of the Learning Process

- 探討當以類神經網路實現某函數時,"actual" function $\boldsymbol{F}(\mathbf{x},\mathbf{w})$和" target" function之間的偏差(deviation)
- 類神經網路可視為是將有興趣的實際現象或環境參數,經由訓練過程以獲得知識(empirical knowledge)
- 考慮某隨機現象以亂數向量$\boldsymbol{X}$表示(由一組獨立變數所組成),亂數值$\boldsymbol{D}$表示相依變數
- 假設亂數向量$\boldsymbol{X}$與其對應之亂數值$\boldsymbol{D}$有N個實現,分別表示成$\{\mathbf{x}_i\}_{i=1}^{N}$, $\{d_i\}_{i=1}^{N}$,
- 則此實現可表示成

$$\mathfrak{I} = \{(\mathbf{x}_i, d_i)\}_{i=1}^{N} \qquad (2.53)$$

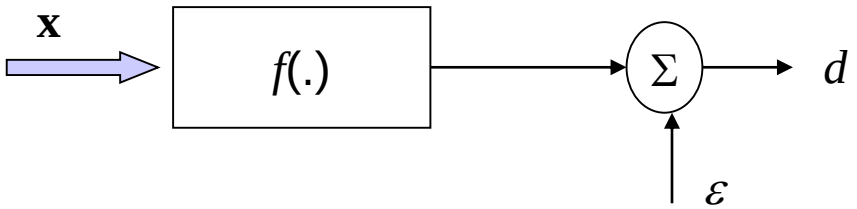
Statistical Nature of the Learning Process

- 在一般情況下,我們無法得知隨機向量$\boldsymbol{X}$和亂數值$\boldsymbol{D}$之間的關係，因此假設成

$$D = f(\mathbf{X}) + \varepsilon \tag{2.54}$$

$f(.)$為deterministic function, ε is a random *expectational error*.

- 上式可視為一個regressive model



使用向量x來解釋或預測相依變數D

Statistical Nature of the Learning Process

- 圖2.20a的regressive model有兩個有用的特性
  - The mean value of the expectational error ε, given any realization $\mathbf{x}$ is zero

$$E[\varepsilon \mid \mathbf{x}] = 0 \tag{2.55}$$

也就是說,當給予一個輸入$\mathbf{X}=\mathbf{x}$時,regression function $f(\mathbf{x})$是output $\mathbf{D}$的條件平均值

$$f(\mathbf{x}) = E[D \mid \mathbf{x}] \tag{2.56}$$

  - Principle of orthogonality: The expectational error ε is uncorrelated with the regression function $f(\mathbf{X})$

$$E[\varepsilon f(X)] = 0 \tag{2.57}$$

    - All the information about D available to us through the input $\mathbf{X}$ has been encoded into the regression function $f(\mathbf{X})$.

Statistical Nature of the Learning Process

- 圖2.20b是藉由訓練範例， 調整神經鍵**w**以獲得由經驗得到的知識

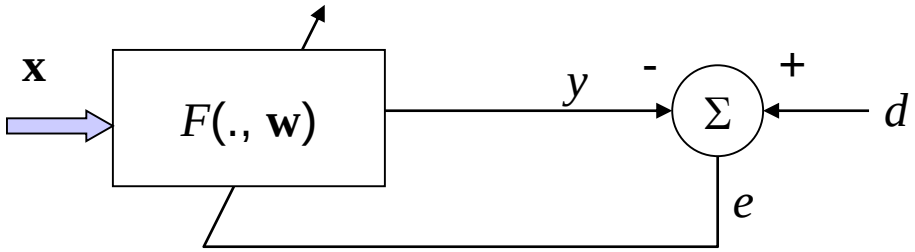
$$\mathcal{T} \longrightarrow \mathbf{w} \tag{2.58}$$

- 類神經網路提供regressive model的一種近似

$$Y = F(\mathbf{X}, \mathbf{w}) \tag{2.59}$$

其中， $F(.,\mathbf{w})$為由類神經網路所實現的input-output function，給予訓練資料$\mathcal{T}$，權重向量**w**是由最小化成本函數(cost function)所得到

$$\xi(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}(d_i - F(\mathbf{x}_i, \mathbf{w}))^2 \tag{2.60}$$



Statistical Nature of the Learning Process

- Let the symbol $E_{\mathfrak{T}}$ denote the *average operator* taken over the entire training sample $\mathfrak{T}$.
- The statistical expectation operator E acts on the whole ensemble of random variables $\mathbf{X}$ and D.
- 根據2.58的描述，我們可以交換使用$\boldsymbol{F}(\mathbf{x},\mathfrak{T})$與$\boldsymbol{F}(\mathbf{x},\mathbf{w})$，因此Eq(2.60)可改寫成

$$\xi(w)=\frac{1}{2}E_{\mathfrak{T}}\left[(d-F(\mathbf{x},\mathfrak{T}))^2\right] \tag{2.61}$$

- 對參數$(d\text{-}F(\mathbf{x},\mathfrak{T}))$同時加及減去$f(\mathbf{x})$，並使用Eq(2.54)可得

$$\begin{aligned} d-F(\mathbf{x},\mathfrak{T}) &= (d-f(\mathbf{x}))+(f(\mathbf{x})-F(\mathbf{x},\mathfrak{T})) \\ &= \varepsilon+(f(\mathbf{x})-F(\mathbf{x},\mathfrak{T})) \end{aligned}$$

Statistical Nature of the Learning Process

- 將上式代入Eq(2.61)並整理，可得

$$\xi(\mathbf{w}) = \frac{1}{2} E_{\mathfrak{I}}[\varepsilon^2] + \frac{1}{2} E_{\mathfrak{I}}\left[(f(\mathbf{x}) - F(\mathbf{x}, \mathfrak{I}))^2\right] + E_{\mathfrak{I}}\left[\varepsilon(f(\mathbf{x}) - F(\mathbf{x}, \mathfrak{I}))\right] \qquad (2.62)$$

- Eq(2.62)等號右邊的最後一項為零
  - The expectational error is uncorrelated with the regression function $f(\mathbf{x})$
- 因此Eq(2.62)可簡化成

$$\xi(\mathbf{w}) = \frac{1}{2} E_{\mathfrak{I}}[\varepsilon^2] + \frac{1}{2} E_{\mathfrak{I}}\left[(f(\mathbf{x}) - F(\mathbf{x}, \mathfrak{I}))^2\right] \qquad (2.63)$$

Intrinsic error與$\mathbf{w}$無關(That is the variance of the expectational error, evaluated over the training sample $\mathfrak{I}$)

- 因此，the natural measure of the effectiveness of $F(\mathbf{x},\mathbf{w})$ as a predictor of the desired response d is defined by

$$L_{av}(f(\mathbf{x}), F(\mathbf{x}, \mathbf{w})) = E_{\mathfrak{I}}\left[(f(\mathbf{x}) - F(\mathbf{x}, \mathfrak{I}))^2\right] \qquad (2.64)$$

Statistical Nature of the Learning Process

- Bias/Variance Dilemma
  - 利用Eq(2.56)可重新定義$f(\mathbf{x})$和$F(\mathbf{x},\mathbf{w})$之間的差距

$$L_{av}(f(\mathbf{x}),F(\mathbf{x},\mathbf{w}))=E_{\mathfrak{I}}\left[(E[D\mid X=\mathbf{x}]-F(\mathbf{x},\mathfrak{I}))^2\right] \quad (2.65)$$

  - 同時對$E[D|\mathbf{X}=\mathbf{x}]$加及減去$E_{\mathfrak{I}}[F(\mathbf{x},\mathfrak{I})]$，可得

$$\begin{aligned}&E[D\mid X=\mathbf{x}]-F(\mathbf{x},\mathfrak{I})\\&=\left(E[D\mid X=\mathbf{x}]-E_{\mathfrak{I}}[F(\mathbf{x},\mathfrak{I})]\right)+\left(E_{\mathfrak{I}}[F(\mathbf{x},\mathfrak{I})]-F(\mathbf{x},\mathfrak{I})\right)\end{aligned}$$

  - 使用類似於從Eq(2.61)推導Eq(2.62)的方法，可將Eq(2.65)重新改寫成

$$L_{av}(f(\mathbf{x}),F(\mathbf{x},\mathfrak{I}))=B^2(\mathbf{w})+V(\mathbf{w}) \quad (2.66)$$

其中

$$B(\mathbf{w})=E_{\mathfrak{I}}[F(\mathbf{x},\mathfrak{I})]-E[D\mid X=\mathbf{x}] \quad (2.67)$$

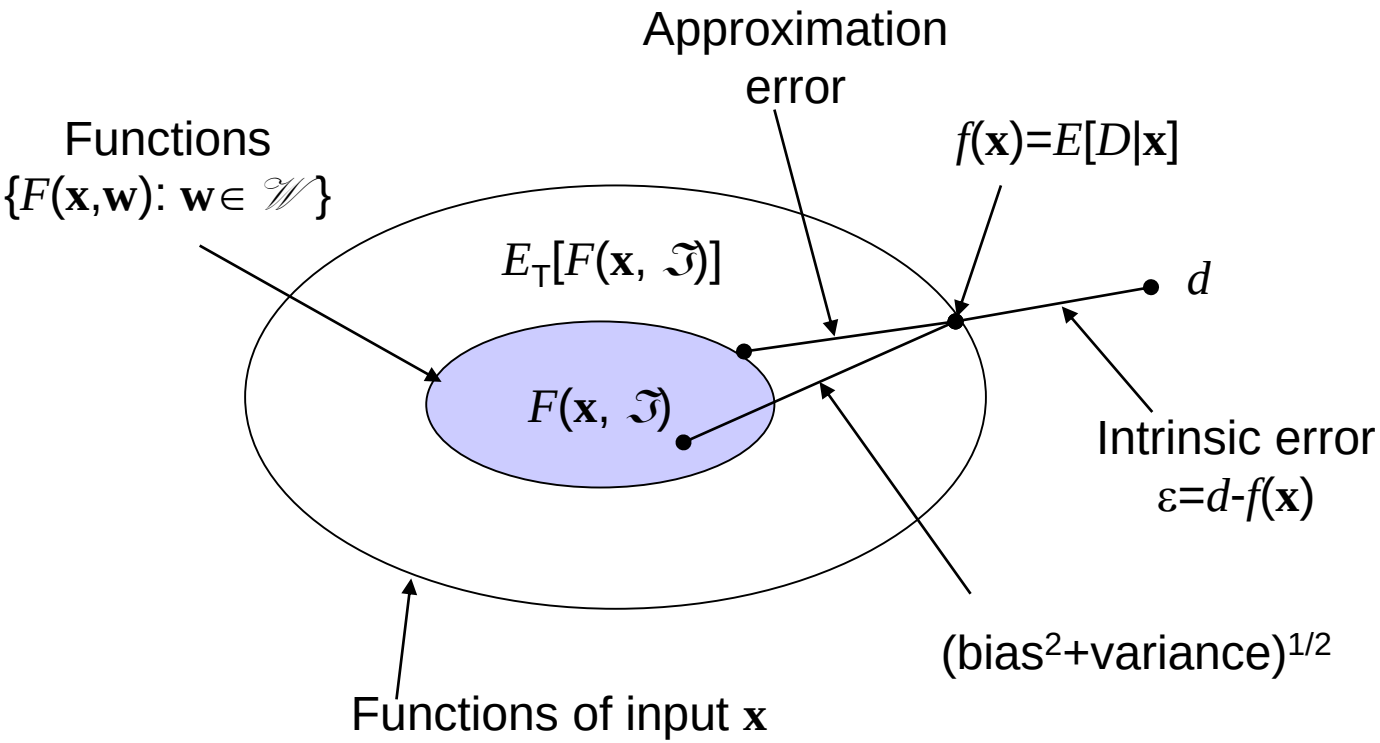
$$V(\mathbf{w})=E_{\mathfrak{I}}\left[(F(\mathbf{x},\mathfrak{I})-E_{\mathfrak{I}}[F(\mathbf{x},\mathfrak{I})])^2\right] \quad (2.68)$$

近似誤差：由類神經網路$F(\mathbf{x},\mathbf{w})$來實現regression function $f(\mathbf{x})$時精確度的誤差。

估計誤差：量測整個訓練範例，所獲得的近似函數$F(\mathbf{x},\mathbf{w})$的variance。包含在訓練樣本內的資料不適當性

Statistical Nature of the Learning Process

- The various sources of error in solving the regression problem



Statistical Learning theory

- 本節將使用數學式來探討如何控制類神經網路的一般性能力(generalization ability)
- Supervised learning由下列元件所組成
  - Environment
    - Stationary, supplying a vector $\mathbf{x}$ with a fixed but unknown cumulative (probability) distribution function $F_{\mathbf{x}}(\mathbf{x})$
  - Teacher
    - Provides a desired response d for every input vector $\mathbf{x}$ received from the environment

$$d = f(\mathbf{x}, v) \quad \text{(2.69)}$$

noise

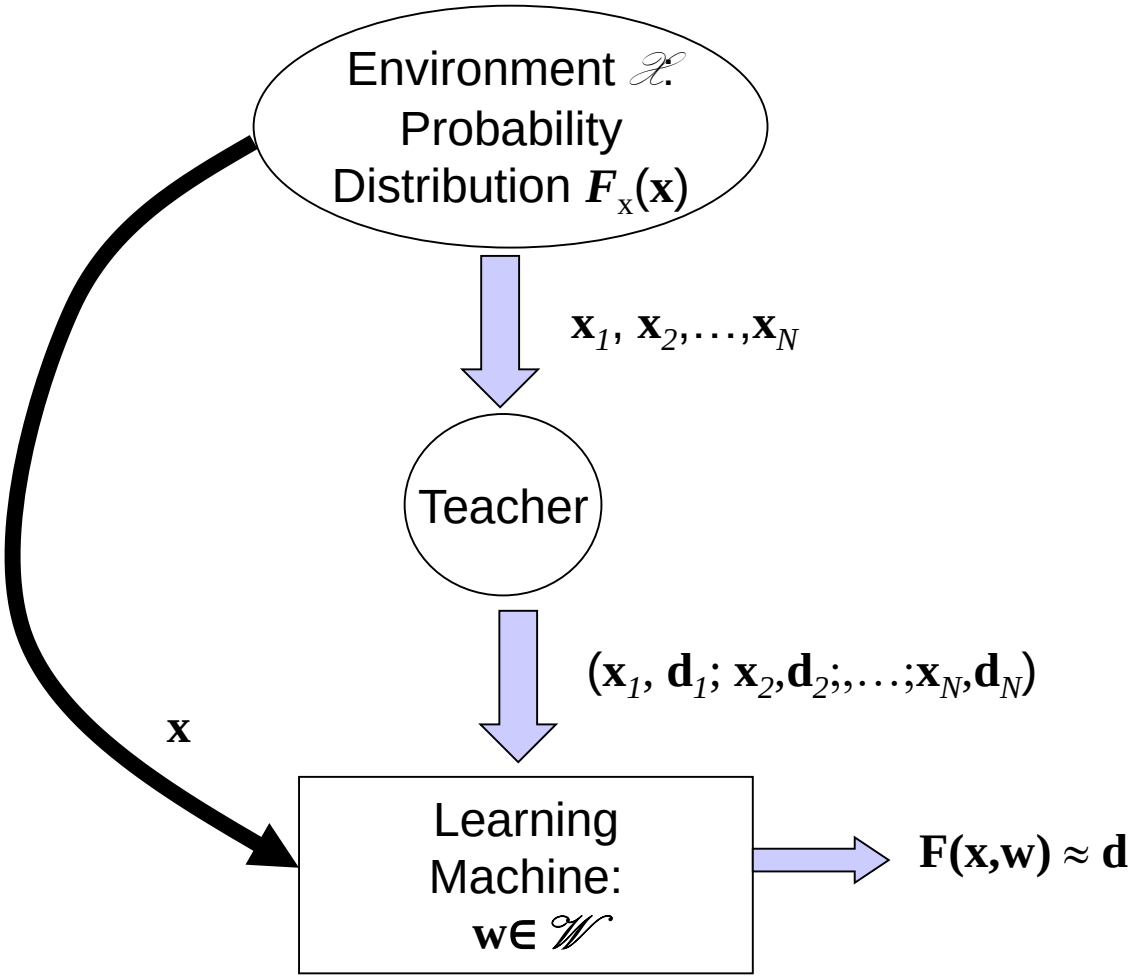
  - Learning machine
    - Neural network is capable of implementing a set of input-output mapping functions describe by

$$y = F(\mathbf{x}, \mathbf{w}) \quad \text{(2.70)}$$

Actual response

A set of free parameters

Statistical Learning theory



Statistical Learning theory

- Supervised learning problem在於選擇某一特定函數$F(\mathbf{x},\mathbf{w})$，能最佳的近似desired response d。
- 其選擇的過程則是從一組N個獨立，且相同分布的訓練範例中去學習

$$\mathfrak{I} = \{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$$

- 因此，supervised learning的問題在於”訓練樣本是否提供足夠的資訊，供learning machine學習generalization?”
- 令$L(d,F(\mathbf{x},\mathbf{w}))$為desired response d和$F(\mathbf{x},\mathbf{w})$的實際輸出的差異(loss function)

$$L(d, F(\mathbf{x},\mathbf{w})) = (d - F(\mathbf{x},\mathbf{w}))^2 \qquad (2.71)$$

- The expected value of the loss is defined by the *risk functional*

$$R(\mathbf{w}) = \int L(d, F(\mathbf{x},\mathbf{w}))\, dF_{\mathbf{X},D}(\mathbf{x}, d)$$

將所有的loss function加總起來，即為risk functional

實際上該項為未知，在supervised learning中所有的資訊都在training set $\mathfrak{I}$中

Statistical Learning theory (cont.)

- To overcome the mathematical difficulty, we use the inductive principle of empirical risk minimization.
  - This principle relies entirely on availability of the training data set $\mathfrak{T}$, which makes it perfectly suited to the design philosophy of neural network.

Some Basic Definitions

- Convergence in probability
  - Consider a sequence of random variables $a_1, a_2, \ldots, a_N$. This sequence of random variables is said to converge in probability to a random variable a_0 if for any δ>0

$$P\left(\left|a_N - a_0\right| > \delta\right) \xrightarrow{P} 0 \quad \text{as } N \to \infty$$

- Supremum and infimum
  - The supremum of a nonempty set $\mathcal{A}$ of scalars, sup $\mathcal{A}$, is defined as the smallest scalar x such that $x \geq y$ for all $y \in \mathcal{A}$.
    - If no such scalar exist, we say that the supremum of the nonempty set $\mathcal{A}$ is ∞.
  - Theinfimum of set $\mathcal{A}$ of scalars, inf $\mathcal{A}$, is defined as the largest scalar x such that $x \leq y$ for all $y \in \mathcal{A}$.
    - If no such scalar exist, we say that the infimum of the nonempty set $\mathcal{A}$ is ∞.

Some Basic Definitions (cont.)

- Empirical risk functional
  - Given the training sample $\mathfrak{I} = \{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$, the empirical risk functional is defined in terms of the loss function as

$$R_{emp}(\mathbf{w}) = \frac{1}{N}\sum_{i=1}^{N} L(d_i, F(\mathbf{x}_i, \mathbf{w})) \qquad (2.74)$$

- Strict Consistency
  - Consider the set $\mathscr{W}$ of functions $L(d, F(\mathbf{x},\mathbf{w}))$ whose underlying distribution is defined by the joint cumulative distribution function $F_{\mathbf{X},\mathbf{D}}(\mathbf{x},d)$.
  - Let $\mathscr{W}(c)$ be any nonempty subset of this set of function

$$\mathscr{W}(c) = \left\{\mathbf{w} : \int L(d, F(\mathbf{x},\mathbf{w})) \geq c\right\} \quad \text{where } c \in (-\infty,\infty) \qquad (2.75)$$

  - The empirical risk function is said to be strictly consistent if for any subset $\mathscr{W}(c)$ the following convergence in probability holds

$$\inf_{\mathbf{w}\in\mathscr{W}(c)} R_{emp}(\mathbf{w}) \xrightarrow{P} \inf_{\mathbf{w}\in\mathscr{W}(c)} R(\mathbf{w}) \text{ as } N \to \infty \qquad (2.76)$$

Statistical Learning theory

- Principle of Empirical Risk Minimization
  - Empirical risk functional
    - 給定訓練範例$\mathcal{T}=\{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$,empirical risk functional定義成

$$R_{emp}(\mathbf{w}) = \frac{1}{N}\sum_{i=1}^{N} L(d_i, F(\mathbf{x}_i, \mathbf{w})) \qquad (2.74)$$

因為loss function可由input-output pair獲得

    - 上式與Eq(2.72)有兩點不同
      - 它是明確的，不相依於未知的分佈$\boldsymbol{F}_{\mathbf{x},D}(\mathbf{x},d)$
      - 理論上，可藉由調整權重向量$\mathbf{w}$，使上式最小化
    - 令$\mathbf{w}_{emp}$和$\boldsymbol{F}(\mathbf{x}, \mathbf{w}_{emp})$分別表示可使$\boldsymbol{R}_{emp}(\mathbf{w})$最小化的權重向量及相對的對應.
    - 令$\mathbf{w}_0$和$\boldsymbol{F}(\mathbf{x}, \mathbf{w}_0)$分別表示可使$\boldsymbol{R}(\mathbf{w})$最小化的權重向量及相對的對應
    - 藉由量測$\boldsymbol{R}(\mathbf{w}_{emp})$與$\boldsymbol{R}(\mathbf{w}_0)$的不匹配值，使approximate mapping $\boldsymbol{F}(\mathbf{x}, \mathbf{w}_{emp})$能接近desired mapping $\boldsymbol{F}(\mathbf{x}, \mathbf{w}_0)$

Statistical Learning theory

- 對某一固定的$\mathbf{w}=\mathbf{w}^*$，the risk functional $\boldsymbol{R}(\mathbf{w}^*)$ determines the mathematical expectation of a random variable

$$Z_{w^*} = L(d, F(\mathbf{x}, \mathbf{w}^*)) \tag{2.77}$$

- 根據大數定理(law of large number),當training sample $\Im$的大小變成無窮大時，經由訓練得到的$\boldsymbol{Z}_{W^*}$的mean將收斂成其期望值
- 如果empirical risk functional $\boldsymbol{R}_{\text{emp}}(\mathbf{w})$近似於原始的risk functional $\boldsymbol{R}(\mathbf{w})$在某個w於精確度$\varepsilon$，則最小的$\boldsymbol{R}_{\text{emp}}(\mathbf{w})$和最小的$\boldsymbol{R}(\mathbf{w})$間的差距不會超過$2\varepsilon$。因此可訂出嚴格的機率關係

$$P(\sup_{\mathbf{w}} | R(\mathbf{w}) - R_{emp}(\mathbf{w}) | > \varepsilon) \to 0 \quad as\ N \to \infty \tag{2.78}$$

當N趨近於無窮大，真實風險函數$R(\mathbf{w})$和empirical風險函數$R_{\text{emp}}(\mathbf{w})$的差異大於$\varepsilon$的最大機率趨近於0，也就是說$R_{\text{emp}}(\mathbf{w})$會往$R(\mathbf{w})$趨近

- 如果Eq(2.78)滿足的話，可說empirical mean risk在權重向量$\mathbf{w}$的情況下將穩定的收斂到其期望值。

Statistical Learning theory

- 同樣的，給予任一事先設定的precision ε，可假設不等式

當N不趨近於無窮大時，$R(\mathbf{w})$和$R_{emp}(\mathbf{w})$>ε的機率小於α

$$P(\sup_{w} | R(\mathbf{w}) - R_{emp}(\mathbf{w}) | > \varepsilon) < \alpha \qquad (2.79)$$

- 下式也同時成立 (將上式移除絕對值符號)

$$P(R(\mathbf{w}_{emp}) - R(\mathbf{w}_0) > 2\varepsilon) < \alpha \qquad (2.80)$$

- 也就是說，當Eq(2.79)成立時，$\boldsymbol{F}(\mathbf{x},\mathbf{w}_{emp})$的解將會最小化empirical risk functional $\boldsymbol{R}_{emp}(\mathbf{w})$的機率最少是(1-α)，而且實際的risk $\boldsymbol{R}(\mathbf{w}_{emp})$和最小可能的$\boldsymbol{R}(\mathbf{w}_0)$間的差距不會超過2ε

Statistical Learning theory

- 若Eq(2.79)成立，表示有至少(1-α)的機率下列兩式會成立

$$R(\mathbf{w}_{emp}) - R_{emp}(\mathbf{w}_{emp}) < \varepsilon \quad (2.81)$$

$$R_{emp}(\mathbf{w}_0) - R(\mathbf{w}_0) < \varepsilon \quad (2.82)$$

上述二式定義真實風險與empirical 風險的差異，當$\mathbf{w}=\mathbf{w}_{emp}$, $\mathbf{w}=\mathbf{w}_0$

- 因為$\mathbf{W}_{emp}$和$\mathbf{w}_0$分別會獲得$\boldsymbol{R}_{emp}(\mathbf{w})$和$\boldsymbol{R}(\mathbf{w})$的最小值，因此

$$R_{emp}(\mathbf{w}_{emp}) \le R_{emp}(\mathbf{w}_0) \quad (2.83)$$

- 將Eq(2.81)+Eq(2.82) ，並使用Eq(2.83)可得

$$R(\mathbf{w}_{emp}) - R(\mathbf{w}_0) < 2\varepsilon \quad (2.84)$$

- 由於Eq(2.81)和Eq(2.82)有(1-α)的機率同時滿足，因此下式不等式有α的機率會成立

$$R(\mathbf{w}_{emp}) - R(\mathbf{w}_0) > 2\varepsilon$$

Statistical Learning theory

- Summaries of principle of empirical risk minimization
  - 取代risk functional $\boldsymbol{R}(\mathbf{w})$ ，建構一empirical risk functional

$$R_{emp}(\mathbf{w}) = \frac{1}{N}\sum_{i=1}^{N} L(d_i, F(\mathbf{x}_i, \mathbf{w}))$$

  - 令$\mathbf{w}_{emp}$表示使empirical risk functional $\boldsymbol{R}_{emp}(\mathbf{w})$最小化的權重向量，則當training sample的大小為無限大時, empirical risk functional $\boldsymbol{R}_{emp}(\mathbf{w})$將會收斂到與actual risk functional $\boldsymbol{R}(\mathbf{w})$一樣。
  - Uniform convergence定義成

$$P(\sup_{w} | R(\mathbf{w}) - R_{emp}(\mathbf{w}) | > \varepsilon) \to 0 \quad as \;\; N \to \infty$$

    為一充分必要條件
  - 在訓練learning machine前，所有的approximating functions可能會很類似；隨著訓練過程的進行，近似函數與訓練資料間的相似度會增加，隨著訓練資料集的增加，輸入空間將更密集，則$\boldsymbol{R}_{emp}(\mathbf{w})$最小值的點將收斂到$\boldsymbol{R}(\mathbf{w})$的最小點。

VC Dimension

- Vapnik-Chervonenkis dimension
  - A measure of the capacity or expressive power of a family of classification functions realized by the learning machine.
  - 假設一個binary classification problem，其desired為$d \in \{0,1\}$
  - 則經由一學習機器所實現的整體二元函數dichotomy為

$$\mathfrak{I} = \left\{ F(\mathbf{x}, \mathbf{w}) : \mathbf{w} \in \mathcal{W}, F : R^m \mathcal{W} \to \{0,1\} \right\} \tag{2.85}$$

  - 令$\mathscr{L}$為輸入向量在m維空間$\mathscr{X}$上N個點所組成的集合

$$\mathcal{L} = \left\{ \mathbf{x}_i \in \mathcal{X}; i = 1,2,..., N \right\} \tag{2.86}$$

VC Dimension

- 二分法(dichotomy)將$\mathcal{S}$分成兩個子集合

$$F(\mathbf{x},\mathbf{w})=\begin{cases}0 & for\ \mathbf{x}\in\mathcal{L}_0\\ 1 & for\ \mathbf{x}\in\mathcal{L}_1\end{cases} \qquad (2.87)$$

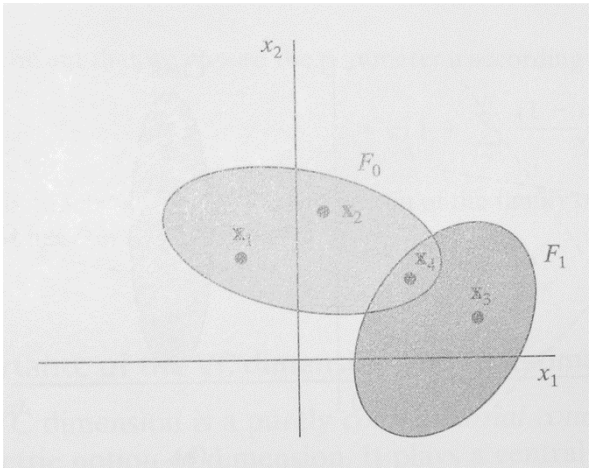
- 令$\Delta_{\mathcal{F}}(\mathcal{S})$表示由學習機器可實現之不同二分函數(dichotomy)的個數。
- 定義 $\Delta_{\mathcal{F}}(\mathcal{S})=\max \Delta_{\mathcal{F}}(l)$ ，其中 $|\mathcal{S}|=l$ ($|\mathcal{S}|$ 為$\mathcal{S}$的元素的個數)
- $|\mathcal{S}|$ is shattered by $\mathcal{F}$ if $\Delta_{\mathcal{F}}(\mathcal{S})=2^{|\mathcal{S}|}$
  - If all posible dichotomies of $|\mathcal{S}|$ can be induced by functions in $\mathcal{F}$. (在size l 的data中所能分類種類個數)
- $\Delta_{\mathcal{F}}(l)$ 稱為growth function
- The VC dimension of an ensemble of dichotomies $\mathcal{F}$ is the cardinality of the largest set $\mathcal{S}$ that is shattered by $\mathcal{F}$.
- The VC dimension of $\mathcal{F}$ is the largest N such that $\Delta_{\mathcal{F}}(N)=2^N$
- 一組分類函數的VC維度為可經由機器學習，而且對所有可能的分類函數的binary labeling沒有錯誤的最大訓練樣本數。

VC Dimension

- Example 2.1
  - 假設F_0為第0類， F_1為第1類
  - 則

$$D_0 = \{L_0 = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_4\}, L_1 = \{\mathbf{x}_3\}\}$$
$$D_1 = \{L_0 = \{\mathbf{x}_1, \mathbf{x}_2\}, L_1 = \{\mathbf{x}_3, \mathbf{x}_4\}\}$$

$$\Delta_{\mathcal{F}}(\mathcal{L}) = 2^4 = 16$$



  - The VC dimension of an ensemble of dichotomies $\mathcal{F}$ is the cardinality of the largest set $\mathcal{L}$ that is shattered by $\mathcal{F}$
  - VCdim($\mathcal{F}$) is the largest N such that $\Delta_{\mathcal{F}}(N)=2^N$

Classification function的VC dimension等於可由機器所學習,而無任何可能錯誤的binary labeling的最大量訓練樣本數

Example 2.2

- 考慮一個在m維輸入向量空間$\mathcal{X}$的簡單的決策規則:

$$\mathcal{F}: \ y = \varphi(w^T x + b) \qquad (2.88)$$

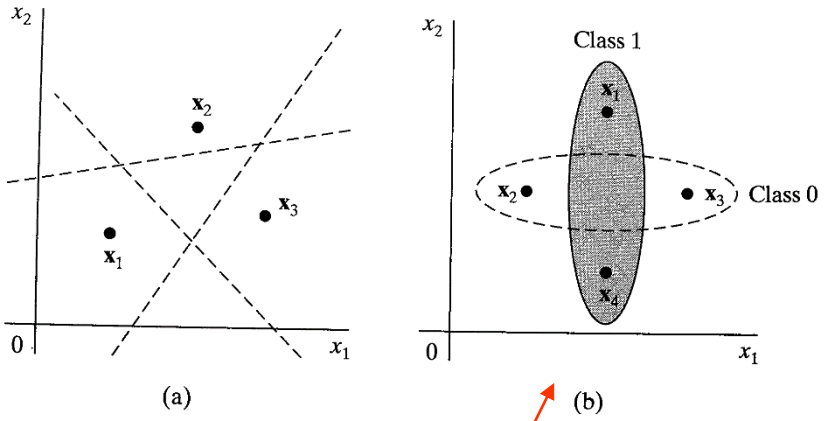
其中激發函數定義為

$$\varphi(v) = \begin{cases} 1, & v \geq 0 \\ 0, & v < 0 \end{cases}$$

- 則決策規則的VC維度為

$$VC\ \dim(\mathcal{F}) = m+1 \qquad (2.89)$$

m=2，因為data在2度空間中，則可shatter到 l =3 個data



當data個數為4時，無論此4點如何改，都無法分到2^4種，如圖中$\mathbf{x}_2,\mathbf{x}_3$和$\mathbf{x}_1,\mathbf{x}_4$無法用一條線分成兩類。

Example 2.3

- 是否# of free parameter大的，其VC dimension就大？而# of free parameter小的，其VC dimension就小？
  - 不盡然
  - 假設選擇任意數目N，尋找N個點可被$f(x,a)$分割開來。
  - 例如：$f(x,a)=\text{sgn}(sin(ax))$只有一個free parameter，但當$x$ sample為$x_i=10^{-i}, i=1,2,...,N$時，將這些資料點分成兩類：$d_1, d_2,...,d_N$ $d_i \in \{-1,1\}$
  - 且當a定為

$$a = \Pi\left(1+\sum_{i=1}^{N}\frac{(1-d_i)10^i}{2}\right)$$

    則無論多少個x_i，均可被分開
  - 因此，其VC dimension無窮大

VC Dimension (cont.)

- Importance of the VC dimension and its Estimation
  - The number of examples needed to learn a class of interest reliably is proportional to the VC dimension of that class.
  - 一般而言，VC dimension越大，所需的learning pattern越多，一般的VC dimension不一定可由分析求得，但其bound可大略知道
  - The VC dimension is determined by the free parameters of a neural network.
    - Let $\mathcal{N}$ denote an arbitrary feed-forward network built up from neurons with a threshold activation function

$$\varphi(v) = \begin{cases} 1 & \text{for } v \geq 0 \\ 0 & \text{for } v < 0 \end{cases}$$

The VC dimension of $\mathcal{N}$ is $O(W \log W)$ where W is the total number of free parameters in the network.

VC Dimension (cont.)

- Let $\mathcal{N}$ denote a multilayer feed-forward network whose neurons used a sigmoid activation function

$$\varphi(v) = \frac{1}{1+\exp(-v)}$$

The VC dimension of N is $O(W^2)$ where W is the total number of free parameters in the network.

- The multilayer feed-forward networks have a finite VC dimension.

Constructive Distribution-free Bounds on the Generalization Ability of Learning Machine.

- 考慮binary pattern classification $d \in \{0,1\}$
  - 其loss function只可能有兩個值

$$L(d, F(\mathbf{x}, \mathbf{w})) = \begin{cases} 0 & \text{if } F(\mathbf{x}, \mathbf{w}) = d \\ 1 & otherwise \end{cases} \tag{2.90}$$

  - 因此Eq(2.72)和Eq(2.74)所定義的risk functional $R(\mathbf{w})$和empirical risk functional $R_{\text{emp}}(\mathbf{w})$在這裡可解釋成
    - $R(\mathbf{w})$ is the *probability of classification error* (error rate) denoted by $P(\mathbf{w})$
    - $R_{\text{emp}}(\mathbf{w})$ is the *training error* (frequency of errors made during the training session) denoted by $v(\mathbf{w})$

Constructive Distribution-free Bounds on the Generalization Ability of Learning Machine.

- 根據law of large numbers, the empirical frequency of occurrence of an event converges almost surely to the actual probability of that event as the number of trials is made infinitely large.

$$P(\left|P(\mathbf{w}) - v(\mathbf{w})\right| > \varepsilon) \to 0 \quad as\ N \to \infty \tag{2.91}$$

- For a training set of sufficiently large size N, the proximity between $v(\mathbf{w})$ and $P(\mathbf{w})$ follows from a stronger condition, which stipulates that the following condition holds for any ε>0

$$P(\sup_{\mathbf{w}} \left| P(\mathbf{w}) - v(\mathbf{w}) \right| > \varepsilon) \to 0 \quad as\ N \to \infty \tag{2.92}$$

- The uniform convergence of the frequency of training errors to the probability that $v(\mathbf{w})=P(\mathbf{w})$

Constructive Distribution-free Bounds on the Generalization Ability of Learning Machine.

VC dimension提供uniform convergence機率的一個bound。

- For the set of classification functions with VC dimension h, the following inequality holds

為達到uniform convergence，當N變大時，該項將獲一小值

$$P(\sup_{\mathbf{w}} |P(\mathbf{w}) - v(\mathbf{w})| > \varepsilon) < \left(\frac{2eN}{h}\right)^h \exp(-\varepsilon^2 N)$$

因為N為有趨近無窮大，所以會有一微小值

where N is the size of the training sample
ε is the base of the natural logarithm

- To make the right-hand side of Eq(2.93) small for large N in order to achieve uniform convergence.
  - 當h為有限值， 且N趨近無窮大時，右手邊那一項將趨近於零。
  - A finite VC dimension is a necessary and sufficient condition for uniform convergence of the principle of empirical risk minimization.

Constructive Distribution-free Bounds on the Generalization Ability of Learning Machine.

- If the input space $\mathscr{X}$ has finite cardinality, any family of dichotomies $\mathscr{F}$ will have finite VC dimension with respect to $\mathscr{X}$.
  - Let α denote the probability of occurrence of the event

$$\sup_{\mathbf{w}} |P(\mathbf{w}) - \nu(\mathbf{w})| \geq \varepsilon \tag{2.93a}$$

有α的機率上式會成立，則有1-α的機率下式會成立

$$P(\mathbf{w}) < \nu(\mathbf{w}) + \varepsilon \tag{2.94}$$

  - 根據Eq(2.93)及Eq(2.93a),我們可設定

$$\alpha = \left(\frac{2eN}{h}\right)^{h} \exp(-\varepsilon^2 N) \tag{2.95}$$

因為Eq(2.93)表示分類錯誤的機率$P(\mathbf{w})$和訓練錯誤的機率$\nu(\mathbf{w})$的最大差異大於某特定ε的機率小於Eq(2.93)的右項

Constructive Distribution-free Bounds on the Generalization Ability of Learning Machine.

- 令$\varepsilon_0(N,h,\alpha)$表示滿足Eq(2.95)的特定$\varepsilon$，則對Eq(2.95)兩邊取log，整理後可獲得

$$\varepsilon_0(N,h,\alpha)=\sqrt{\frac{h}{N}\left[\log\left(\frac{2N}{h}\right)+1\right]-\frac{1}{N}\log\alpha} \qquad (2.96)$$

上式為confidence interval

- E(2.93)適用於$P(\mathbf{w})$>=1/2下，For small $P(\mathbf{w})$, a modification of the inequality Eq(2.93) is obtained as

$$P\left(\sup_{\mathbf{w}}\frac{|P(\mathbf{w})-\nu(\mathbf{w})|}{\sqrt{P(\mathbf{w})}}>\varepsilon\right)<\left(\frac{2eN}{h}\right)^{h}\exp\left(-\frac{\varepsilon^2 N}{4}\right) \qquad (2.97)$$

Constructive Distribution-free Bounds on the Generalization Ability of Learning Machine.

$$P(\mathbf{w}) \le v(\mathbf{w}) + \varepsilon_1(N,h,\alpha,v) \tag{2.98}$$

- 其中$\varepsilon_1(N,h,\alpha,v)$是由confidence interval $\varepsilon_0(N,h,\alpha)$所定義的新confidence interval

$$\varepsilon_1(N,h,\alpha,v) = 2\varepsilon_0^2(N,h,\alpha)\left(1+\sqrt{1+\frac{v(\mathbf{w})}{\varepsilon_0^2(N,h,\alpha)}}\right) \tag{2.99}$$

- The second confidence interval depends on the training error $v(\mathbf{w})$.
- 當$v(\mathbf{w})$=0時，Eq(2.99)可簡化成

$$\varepsilon_1(N,h,\alpha,0) = 4\varepsilon_0^2(N,h,\alpha) \tag{2.100}$$

Constructive Distribution-free Bounds on the Generalization Ability of Learning Machine.

- Summarize the two bound on the rate of uniform convergence:
  - In general, we have the following bound on the rate of uniform convergence:

$$P(\mathbf{w}) \le v(\mathbf{w}) + \varepsilon_1(N, h, \alpha, v)$$

  - For a small training error $v(\mathbf{w})$ close to zero, we have

$$P(\mathbf{w}) \le v(\mathbf{w}) + 4\varepsilon_0^2(N, h, \alpha)$$

  - For a large training error $v(\mathbf{w})$ close to unity, we have the bound

$$P(\mathbf{w}) \le v(\mathbf{w}) + \varepsilon_0(N, h, \alpha)$$

Structural Risk Minimization

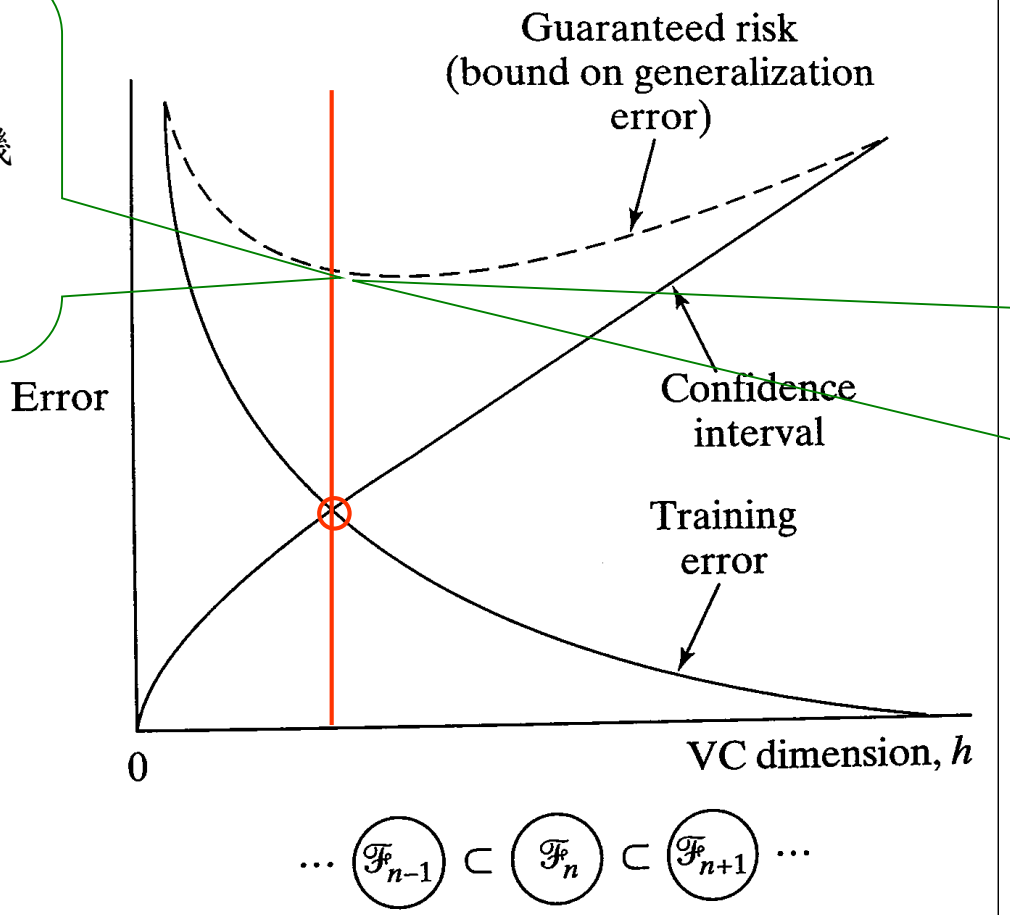
- The *training error* $v_{\text{train}}(\mathbf{w})$ is the frequency of errors made by a learning machine of some weight vector $\mathbf{w}$ during the training session.
- The *generalization error* $v_{\text{gene}}(\mathbf{w})$ is defined as the frequency of errors made by the machine when it is tested with examples not seen before.
- 有1-α的機率，當訓練樣本 N大於 VC維度 h，對所有分類函數 $F(\mathbf{x},\mathbf{w})$ ，其一般性錯誤$v_{\text{gene}}(\mathbf{w})$小於保證風險(guaranteed risk)

$$v_{guarant}(\mathbf{w}) = v_{train}(\mathbf{w}) + \varepsilon_1(N, h, \alpha, v_{train}) \qquad (2.101)$$

- 對一固定數量(N)的訓練樣本，訓練誤差將隨著機器容量或VC維度的增加而降低，同時，信賴區間(confidence interval)也增加。

Structural Risk Minimization

在最低點之前，learning problem為overdetermined，機器容量*h*對training detail的數量而言是太小了。



在最低點之後，learning problem為underdetermined，機器容量*h*對training data的數量而言是太大了。

Structural Risk Minimization

- 求解監督式學習問題的挑戰，在於實現最佳的generalization performance。藉由將問題的機器容量對應到可用的訓練資料量。
- The method of structural risk minimization provides an inductive procedure for achieving this goal by making the VC dimension of the learning machine a control variable.
  - 考慮一pattern classifier，定義一個n的巢狀結構

$$\mathrm{F}_k = \{F(\mathbf{x}, \mathbf{w}); \mathbf{w} \in \mathrm{W}_k\}, \quad k = 1,2,...,n \tag{2.102}$$

  - 我們可獲得(比較大的pattern classifier會包含較小的pattern classifier)

$$\mathcal{F}_1 \subset \mathcal{F}_2 \subset ... \subset \mathcal{F}_n \tag{2.103}$$

  - 個別pattern classifier的VC dimension滿足下面的條件(較大的VC dimension會包含較小的VC dimension)

$$h_1 \le h_2 \le ... \le h_n \tag{2.104}$$

Structural Risk Minimization

- The method of structural risk minimization may proceed as follows
  - The empirical risk (training error) for each pattern classifier is minimized.
  - The pattern classifier $\mathcal{F}^*$ with the smallest guaranteed risk is identified.
- 我們的目的在於找到一個網路結構，能減少VC dimension，而使訓練錯誤(training error)只有些微的增加。
- 可藉由改變隱藏層神經元的數量，來改變VC dimension h。

Probably approximately correct model of learning

- Probably approximately correct (PAC)
  - PAC model is a probabilistic framework for the study of learning and generalization in binary classification systems.
  - 假設*environment* $\mathcal{X}$, $\mathcal{X}$的集合稱為*concept*, $\mathcal{X}$的子集合稱為*concept class*, concept的*example*會有class label，可分成兩種
    - Positive example
      - If the example is a member of the concept.
    - Negative example
      - If the example is not a member of the concept.
  - Training data of length N for a target concept c can be represented as

$$\mathfrak{I} = \{(\mathbf{x}_i, c(\mathbf{x}_i))\}_{i=1}^{N} \qquad (2.105)$$

Probably approximately correct model of learning

- The set of concepts derived from the environment $\mathcal{X}$ is referred to as a concept space C.
  - The concept space may contain “the letter A”, “the letter B”…
  - Each of these concepts may be coded differently to generate a set of positive examples and a set of negative examples.
  - In supervised learning
    - A learning machine typically represents a set of functions, with each function corresponding to a specific state.
    - The machine may be designed to recognize “the letter A”, “the letter B”…
    - The set of all concepts determined by the learning machine is referred to as a hypothesis space $\mathcal{G}$ (假設空間$\mathcal{G}$可能與 concept 空間相等或不相等)

Probably approximately correct model of learning

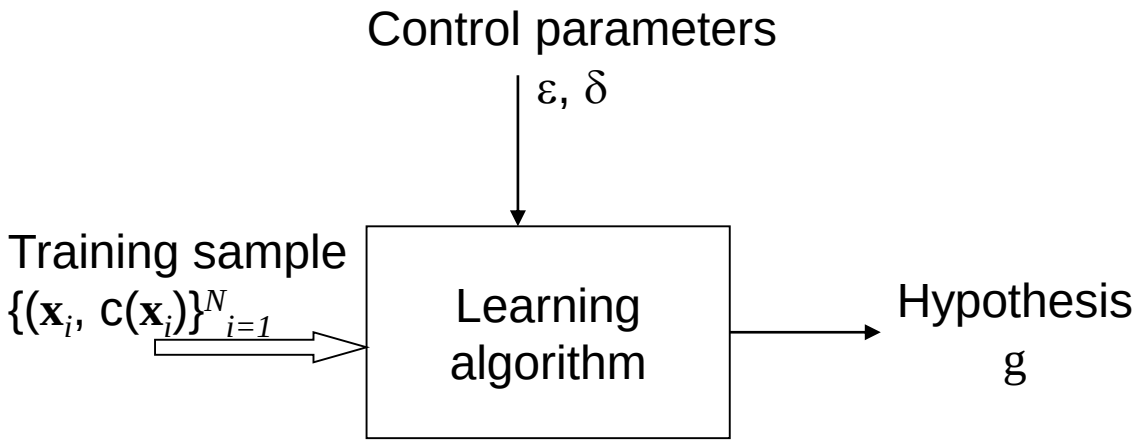
- 假設給定一target concept $c(\mathbf{x}) \in \mathcal{C}$
- 我們希望由data set $\mathcal{T}$訓練類神經網路
- 令$g(\mathbf{x}) \in \mathcal{G}$表示假設的Input/Output對應
- 評估學習是否成功的方法在於量測假設的$g(\mathbf{x})$和目標概念$c(\mathbf{x})$是否相近,當$g(\mathbf{x}) \neq c(\mathbf{x})$時即有誤差產生,訓練誤差的機率定義為

$$v_{train} = P(\mathbf{x} \in \mathcal{X} : g(\mathbf{x}) \neq c(\mathbf{x})) \quad (2.106)$$

- PAC learning的目的在於確保v_{train}經常是小的,此演算法提供兩個控制參數
  - Error parameter $\varepsilon \in (0,1]$:用來訂出target concept $c(\mathbf{x})$和假設的$g(\mathbf{x})$間的容忍誤差
  - Confidence parameter $\delta \in (0,1]$: controls the likelihood of constructing a good approximation

Probably approximately correct model of learning

- Provide that the size N of the training sample $\mathcal{T}$ is large enough, after the neural network has been trained on that data set it is "probably" the case that the input-output mapping computed by the network is "approximately correct".



Probably approximately correct model of learning

- Sample complexity
  - 應該提供多少個隨機樣本給learning machine,才能獲得足夠的資訊來學習一個未知的目標概念c (target concept)?
  - Training set $\mathcal{T}$應該多大才夠?
  - 令 $\mathcal{F} = \{(\mathbf{x}_i, c(\mathbf{x}_i))\}_{i=1}^{N}$ be any set of labeled examples, each $\mathbf{x}_i \in \mathcal{X}$ and each $d_i \in$(0,1), c is a target concept over the environment $\mathcal{T}$
  - Concept c is said to be consistent with the training set $\mathcal{T}$ if for all 1<=i<=N we have $c(\mathbf{x}_i)=d_i$
- Summary
  - Any consistent learning algorithm for that neural network is a PAC learning algorithm
  - There is a constant K such that a sufficient size of training set $\mathcal{T}$ for any such algorithm is

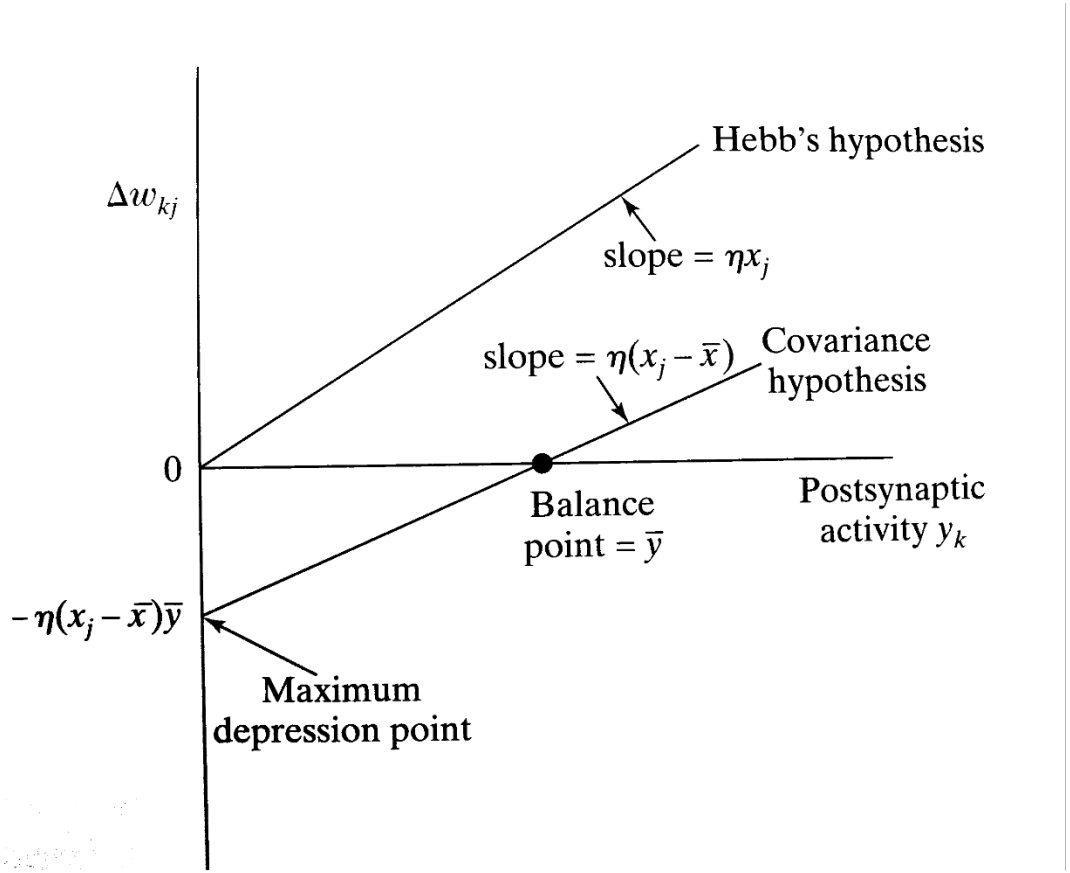
$$N = \frac{K}{\varepsilon}\left(h\log\left(\frac{1}{\varepsilon}\right) + \log\left(\frac{1}{\delta}\right)\right)$$

Probably approximately correct model of learning

- Computational Complexity
  - 估計當給予一組N個labeled examples情況下，訓練一個類神經網路需要花多少時間?(量測需要執行的運算個數)
  - O$(m)^r$
- Error parameter ε
  - For efficient computation, the appropriate condition is to have the running time polynomial in $1/\varepsilon$.

Hebbian Learning (cont.)

- Illustration of Hebb's hypothesis and the covariance hypothesis



Hebbian Learning (cont.)

- Covariance hypothesis
  - To overcome the limitation of Hebb's hypothesis
  - Let $\bar{x}$ and $\bar{y}$ denote the time-averaged values of the presynaptic signal x_j and postsynaptic signal y_k, respectively.
  - The adjustment applied to the synaptic weight w_{kj} is defined by

$$\Delta w_{kj} = \eta(x_j - \bar{x})(y_k - \bar{y})$$

Hebbian Learning (cont.)

- The covariance hypothesis allows for the following:
  - Convergence to a nontrivial state, which is reached when $x_k = \bar{x}$ or $y_j = \bar{y}$
  - Prediction of both synaptic potentiation and synaptic depression.
  - Synaptic weight w_{kj} is enhanced if there are sufficient levels of presynaptic and postsynaptic activities, the conditions $x_j > \bar{x}$ and $y_k > \bar{y}$ are both satisfied.
  - Synaptic weight w_{kj} is depressed if there is either
    - $x_j > \bar{x}$ and $y_k < \bar{y}$
    - $y_k > \bar{y}$ and $x_j < \bar{x}$