ĐẠI HỌC CẦN THƠ

MẠNG NƠ-RON NHÂN TẠO

Mã số: CT384, 3 Tín chỉ
(KT Điện tử VT, KT Điều khiển và Cơ – Điện tử)

TS. Nguyễn Chí Ngôn
Bộ môn Tự Động Hóa
Khoa Kỹ thuật Công nghệ
Email: ncngon@ctu.edu.vn

---2008---

Nội Dung

- **Chương 1: Tổng quan về mạng nơ-ron nhân tạo (ANN)**
- **Chương 2: Cấu trúc của ANN**
- Chương 3: Các giải thuật huấn luyện ANN
- Chương 4: Một số ứng dụng của ANN (MATLAB)
- Đồ án môn học
- Chương 5: Mạng nơ-ron mờ (Fuzzy-Neural Networks)
- Chương 6: Một số định hướng nghiên cứu (Case Studies)
- Ôn tập và thảo luận

Tham khảo:

1. *Nguyễn Chí Ngôn, Điều khiển mô hình nội và Neural network: Chương 2 – Mạng nơ-ron nhân tạo, Luận án cao học, ĐHBK Tp. HCM, 2001.*
2. *Nguyễn Đình Thúc, Mạng nơ-ron – Phương pháp và ứng dụng, NXBGD, 2000.*
3. *Simon Haykin, Neural Networks a comprehensive foundation, Prentice Hall, 1999.*
4. *Howard Demuth, Mark Beale and Martin Hagan, Neural Networks toolbox 5 – User's Guide, The Matworks Inc., 2007.*

Tổ chức môn học

- Thời lượng môn học: 3TC
  - 2 TC lý thuyết và bài tập trên lớp
  - 1 TC Đồ án môn học (03 SV thực hiện 1 đề tài)
- Lịch học:
  - Tuần 1: Chương 1
  - Tuần 2 – 3: Chương 2 + Bài tập
  - Tuần 4 – 5: Chương 3 + Bài tập
  - Tuần 6 – 7: Chương 4 + Bài tập
  - Tuần 8 – 12: Đồ án môn học
  - Tuần 13: Chương 5 + 6
  - Tuần 14: Ôn tập
  - Tuần 15: Thi cuối kỳ (45 phút, 55%)
- Đánh giá
  - Đồ án môn học: 45%
  - Thi hết môn: 55%

Chương 1
Tổng quan về ANN

Giới thiệu

Lịch sử phát triển

Khả năng ứng dụng của ANN

Minh họa ứng dụng của ANN

Khái niệm về ANN

Nơ-ron sinh học

Mô hình nơ-ron nhân tạo

Mô phỏng nơ-ron nhân tạo bằng MATLAB

Bài tập

Giới thiệu

- Mạng nơ-ron nhân tạo được dịch từ cụm Artificial Neural Network (ANN), gọi tắt là mạng nơ-ron.
- Tìm hiểu sơ lược về lịch sử phát triển và các lĩnh vực ứng dụng của ANN.
- Minh họa một số ví dụ về ứng dụng của ANN.
- Khái niệm về ANN
- Hoạt động của nơ-ron sinh học
- Mô hình nơ-ron 1 ngõ vào
- Mô hình nơ-ron nhiều ngõ vào
- Mô phỏng nơ-ron bằng công cụ nnet của MATLAB.

Lịch sử phát triển ANNs

Cuối 1980s -nay: Ứng dụng trong nhiều lĩnh vực

1982: Mạng Hopfield 1 lớp, các nghiên cứu được tiếp tục

1970s: Các nghiên cứu đột nhiên lắng dịu

1969: Các bài báo của Minsky & Papert, Perceptrons

Cuối 50s-60s: Nhiều nổ lực, AI & Neural Computing Fields được thành lập

1956: Dartmouth Summer Research Project

1950s: Phần mềm phát triển mạnh

1949: Giải thuật huấn luyện của Donald Hebb

1943: Mcculloch & Pitts công bố về mô hình neuron đơn giản

1936: Turing dùng bộ não như mô hình xử lý thông tin

1890: Khái niệm của William James.

Khả năng ứng dụng của ANNs (1)

- **Không gian vũ trụ:**

  Phi thuyền không người lái, mô phỏng đường bay, tăng cường khả năng điều khiển, mô phỏng các chi tiết trong máy bay, phi thuyền, dự báo hỏng hóc …

- **Giao thông:**

  Hướng dẫn lưu thông tự động, phân tích cảnh báo tình trạng giao thông, xác định đường đi tối ưu.

- **Ngân hàng:**

  Kiểm soát các hóa đơn chứng từ và các tài liệu khác, dự báo chứng khoán, kiểm tra thẻ tín dụng, …

- **Quân sự:**

  Vũ khí tự động, truy tìm mục tiêu, phân biệt đối tượng, nhận dạng tín hiệu và hình ảnh, các ứng dụng trong tàu ngầm, xử lý tín hiệu radar, …

Khả năng ứng dụng của ANNs (2)

- **Điện tử:**

  Giải mã, dự báo lỗi chip, tổng hợp âm thanh, mô hình hóa hệ thống.

- **Giải trí:**

  Phim hoạt hình, kỹ xảo điện ảnh.

- **Tài chính:**

  Định giá bất động sản, tư vấn nợ, thế chấp, phân tích khả năng tài chính của công ty, …

- **Công nghiệp:**

  Kiểm soát các lò nung, kiểm soát các qui trình công nghiệp, phân tích và thiết kế sản phẩm, dự báo chất lượng sản phẩm, …

- **Y học:**

  Phân tích tế bào ung thư, thiết kế các bộ phận giả cho cơ thể, …

Khả năng ứng dụng của ANNs (3)

- **Dầu khí:**

  Thăm dò quặng.

- **Robotics:**

  Điều khiển tay máy, camera robots, …

- **Ngôn ngữ:**

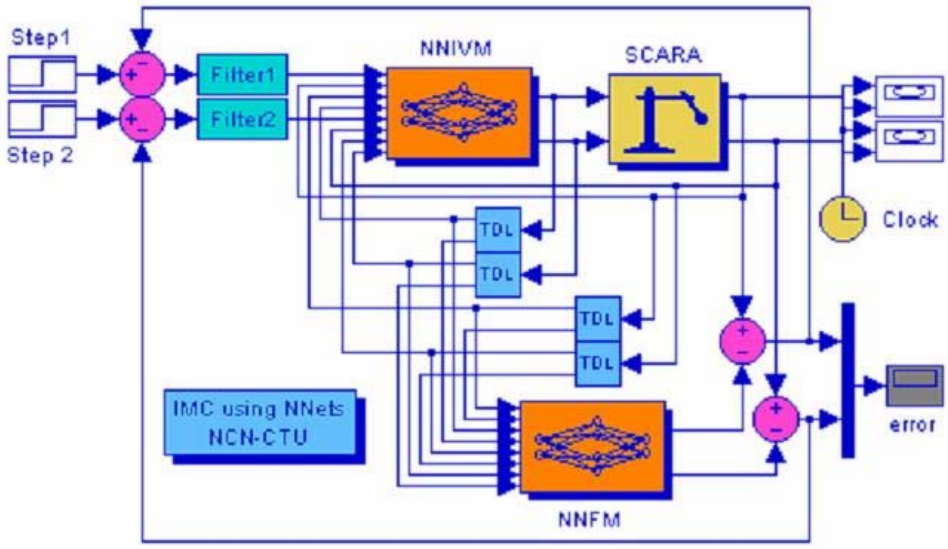
  Nhận dạng giọng nói, tổng hợp âm thanh và chữ viết, nén âm thanh, phân loại nguyên âm.

- **Thông tin:**

  Nén âm thanh và hình ảnh, dịch máy, …

Minh họa ứng dụng của ANN (1)

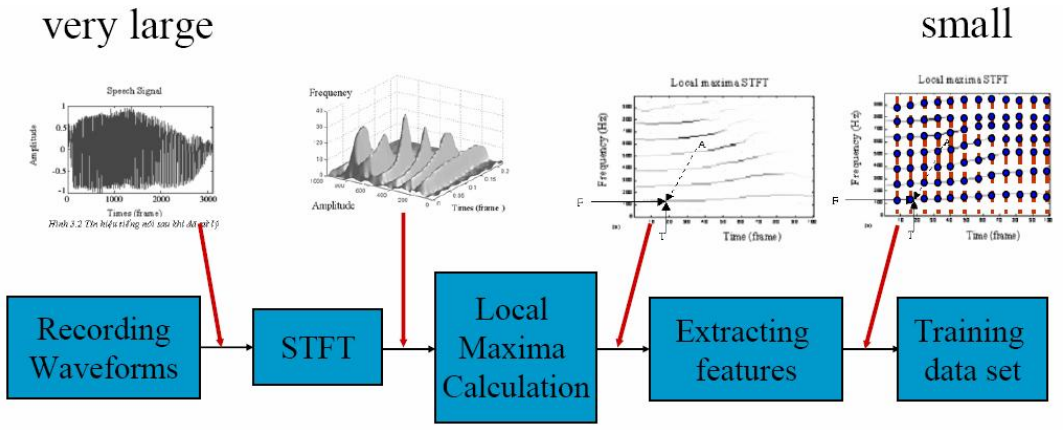
- Điều khiển dùng mô hình nội mạng Neuron áp dụng vào robot SCARA *(Tác giả: Nguyễn Chí Ngôn & Dương Hoài Nghĩa, đăng trên Tạp chí Phát triển Khoa học Công nghệ, ĐHQG Tp. HCM. trang 65-71, tập 4, số 8&9/2001 - PDF file available)*



MATLAB-Simulink

Minh họa ứng dụng của ANN (2)

- Nhận dạng tiếng Việt:
  - Chi-Ngon Nguyen, Thanh-Hung Tran, Thanh-Tuyen T. Truong, and Thai-Nghe Nguyen, "A method of control system by Vietnamese speech using Neural Networks," Proc. 3rd Inter. Conf. in Computer Science: Research, Innovation & Vision of the Future, RIVF'05, Cantho University, Vietnam, February 21-24, 2005, pp. 315-317.
  - Nguyễn Chí Ngôn, Trần Thanh Hùng, Trương Thị Thanh Tuyền, và Nguyễn Thái Nghe, "Ứng dụng mạng nơ-ron nhân tạo để điều khiển thiết bị bằng giọng nói tiếng Việt", Tạp chí Khoa học, Đại học Cần thơ, trang 96-103, số 03, 2005.



✍ **Case study:** Xe lăn điều khiển bằng giọng nói tiếng Việt.

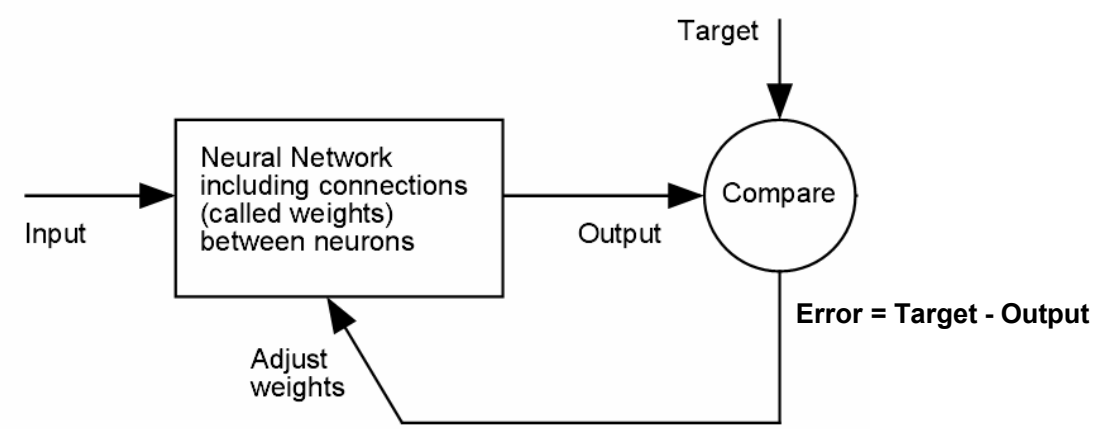
ANN là gì? (1)

- ANN là một mô hình mạng chứa nhiều đơn vị xử lý, giả lập quá trình học tập và tính toán của bộ não con người.
- Mỗi đơn vị xử lý gọi là một nơ-ron nhân tạo
- Mỗi nơ-ron nhân tạo giả lập một nơ-ron sinh học, gồm một ngưỡng kích hoạt (bias) và một hàm kích hoạt (hay hàm truyền – transfer function), đặc trưng cho tính chất của nơ-ron.
- Các nơ-ron nhân tạo được liên kết với nhau bằng các kết nối. Mỗi kết nối có một độ lợi, gọi là trọng số kết nối (weight), đặc trưng cho khả năng nhớ của ANN.
- Quá trình huấn luyện ANN là 1 quá trình điều chỉnh các ngưỡng kích hoạt và các trọng số kết nối, dựa trên dữ liệu học.

✍ **Nhiệm vụ:** Xây dựng một cấu trúc ANN và huấn luyện nó dựa trên dữ liệu thu thập được (gọi là tập dữ liệu học).

ANN là gì? (2)

- Quá trình huấn luyện ANN có thể mô tả như sau:



Dựa trên sự sai biệt (Error) giữa ngõ ra mong muốn (Target) và ngõ ra thực tế của ANN (Output), giải thuật huấn luyện sẽ điều chỉnh các trọng số kết nối của ANN, sao cho: *min{Error}*

Hoạt động của ANN?

- Xét bài toán: Cho 2 vector x và y. Xác định y=f(x)=?

x	1	2	3	4	5
y	2	4	6	8	10

y = f(x) = 2x

Cho x = 7 → y = 14

- Thay đổi dữ liệu:

x	1	2	3	4	5
y	2.5	4.1	6	8.33	10.1

y = f(x) = ?

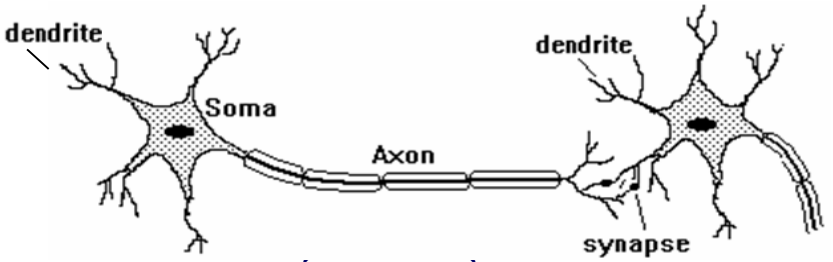
Cho x = 7 → y = ? → khó xác định

- Ứng dụng ANN xác định y=f(x):
  - Thu thập dữ liệu 2 vectors x và y (dữ liệu nhiều, độ chính xác lớn)
  - Xây dựng mạng nơ-ron và huấn luyện mạng dựa trên dữ liệu thu thập
  - Khi đó ANN hoạt động như 1 hàm: $y=f_{ANN}(x)$
  - Với 1 giá trị x_n ta có $y_n = f_{ANN}(x_n) \sim f(x_n)$

✍ ANN có thể được huấn luyện để xấp xỉ một quan hệ phi tuyến bất kỳ

Nơ-ron sinh học

- Bộ não con người chứa khoảng 10^{11} nơ-ron, với hơn 10^{14} liên kết, tạo thành một mạng tế bào thần kinh khổng lồ.
- Mỗi nơ-ron: phần thân (soma), một trục thần kinh ra (axon) và một hệ thống dạng cây chứa các dây thần kinh vào (dendrite).

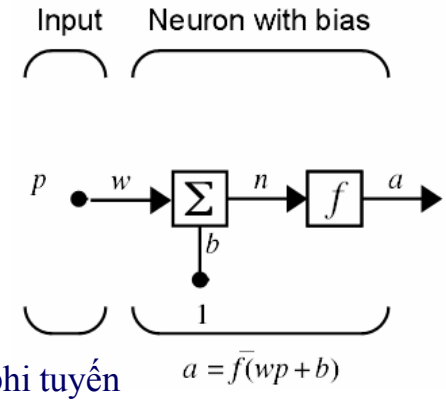


- Hoạt động: Khi điện thế ở dây thần kinh vào vượt quá 1 ngưỡng nào đó, nơ-ron bắt đầu giật (firing), tạo nên một xung điện truyền trên trục thần kinh ra và giải phóng năng lượng cho dây thần kinh vào của các nơ-ron khác qua các khớp nối (synapse)
  - ✍ Có thể mô hình hóa nơ-ron sinh học

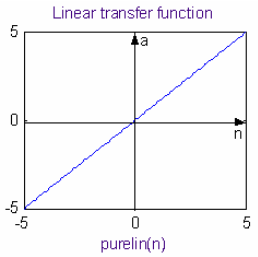
Mô hình nơ-ron 1 ngõ vào (1)

- Cấu tạo nơ-ron gồm:
  - Ngõ vào p; Ngõ ra a - Trọng số kết nối w
  - Ngưỡng kích hoạt b; - Hàm kích hoạt f;
- Tính toán trên nơ-ron:

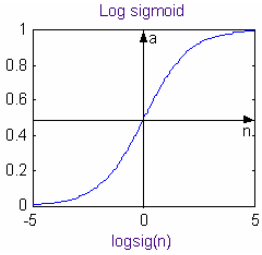
$$a = f(wp + b)$$



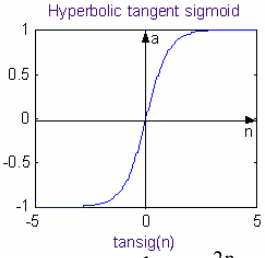
- Hàm kích hoạt (hàm truyền):
  - Có nhiều dạng hàm truyền: Tuyến tính và phi tuyến
  - 3 hàm thông dụng: Purelin, Logsig và Tansig



$f(n) = n$



$$f(n) = \frac{1}{1+e^{-n}}$$

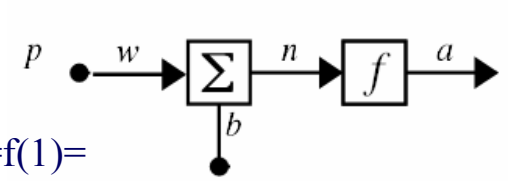


$$f(n) = \frac{1-e^{-2n}}{1+e^{-2n}}$$

Mô hình nơ-ron 1 ngõ vào (2)

- Ví dụ về tính toán trên nơ-ron (1):

  Cho p=0.5, w = 0.2, b=0.9
  và f là hàm Logsig



  Ta có: a = f(wp +b) = f(0.2*0.5+0.9)=f(1)=

$$= \frac{1}{1+e^{-1}} = 0.7311$$

- Ví dụ về tính toán trên nơ-ron (2):

  Cho p=0.5, w = 0.2, b=0.9 và f là hàm Tansig

  Ta có: a = f(wp +b) = f(0.2*0.5+0.9)=f(1)=

$$= \frac{1-e^{-2}}{1+e^{-2}} = 0.7616$$

✍ Quá trình huấn luyện nơ-ron là quá trình điều chỉnh các giá trị trọng số kết nối *w* và giá trị ngưỡng *b*, sao cho ngõ ra *a* gần nhất với giá trị mong muốn.

Các hàm truyền của nơ-ron

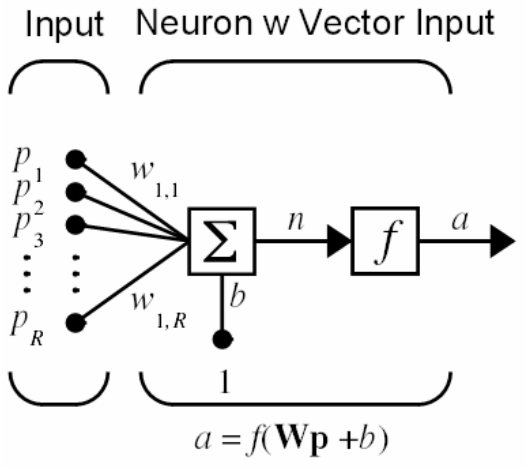
- Sinh viên tự tham khảo: MATLAB - Neural network toolbox

3 hàm thường được sử dụng

TRANSFER FUNCTIONS		
compet	Competitive transfer function.	
hardlim	Hard limit transfer function.	
hardlims	Symmetric hard limit transfer function	
logsig	Log sigmoid transfer function.	
poslin	Positive linear transfer function	
purelin	Linear transfer function.	
radbas	Radial basis transfer function.	
satlin	Saturating linear transfer function.	
satlins	Symmetric saturating linear transfer function	
softmax	Soft max transfer function.	SM
tansig	Hyperbolic tangent sigmoid trans. function.	
tribas	Triangular basis transfer function.	

Mô hình nơ-ron nhiều ngõ vào (1)

- Khi ngõ vào của nơ-ron là một vector có R phần tử:
  $p = [p_1, p_2, ..., p_R]'$
- Cấu trúc:
  - R ngõ vào
  - R trọng số kết nối
    $W=[w_{1,1}, ..., w_{1,R}]$
  - Ngưỡng kích hoạt b
  - Hàm kích hoạt f
  - Ngõ ra a



- Tính toán trên nơ-ron:

$$n = w_{1,1}p_1 + w_{1,2}p_2 + ... + w_{1,R}p_R + b$$

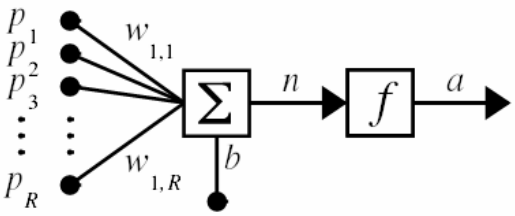
n = W*p + b ⟶ $a = f(\mathbf{W}\mathbf{p} + b)$

Mô hình nơ-ron nhiều ngõ vào (2)

- Ví dụ về tính toán trên nơ-ron nhiều ngõ vào

Cho mô hình nơ-ron như hình vẽ, với:

$$p = [2, 1, 4, 0.5, 10]^T$$
$$W = [0.3, 0.5, 0.2, 0.1, 0.8]$$
$$b = 0.05$$
$$f : Logsig$$

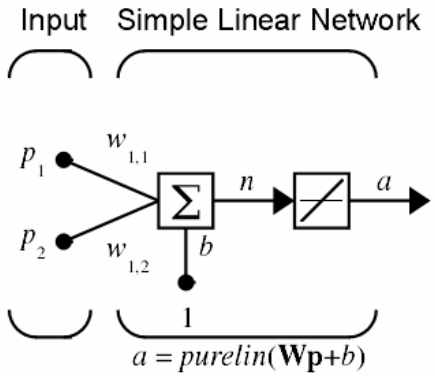


Xác định ngõ ra a:

$$n = Wp + b = [0.3, 0.5, 0.2, 0.1, 0.8] * [2, 1, 4, 0.5, 10]^T + 0.05$$
$$n = 0.3*2 + 0.5*1 + 0.2*4 + 0.1*0.5 + 0.8*10 + 0.05 = 10$$
$$a = f(Wp + b) = f(n) = \frac{1}{1 + e^{-10}} = 0.9999546$$

Mô phỏng nơ-ron bằng MATLAB (1)

- Ví dụ - Xét nơ-ron tuyến tính 2 ngõ vào:
  $a = purelin(n) = purelin(\mathbf{W}p+b) = \mathbf{W}p+b$
  $a = w_{1,1}p_1 + w_{1,2}p_2 + b$
  Giả sử $W = [1, 2]^T$, $p = [0.5, 0.25]$, $b = 0.2$
  ta có: $a = 1.2$



- Mô phỏng bằng MATLAB?
  - Tạo nơ-ron tuyến tính:
    `>> net = newlin([-1 1; -1 1], 1);`
    Trong đó, ma trận *[-1 1; -1 1]* khai báo giới hạn 2 ngõ vào. Tham số thứ hai, *1*, khai báo nơ-ron có 1 ngõ ra.
  - Kiểm tra các trọng số và ngưỡng (mặc nhiên khi mới khởi tạo, các giá trị này bằng 0):
    `>> W = net.IW{1,1};` → W = 0 0 % input weights
    `>> b = net.b{1};` → b = 0 % bias

Mô phỏng nơ-ron bằng MATLAB (2)

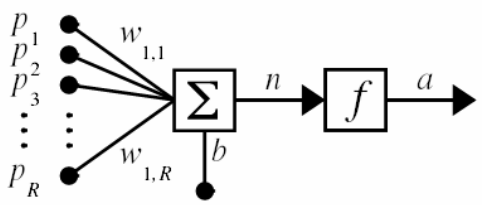
- Gán W và b (trong ứng dụng, các giá trị này có được do huấn luyện):
  `>> net.IW{1,1} = [1 2];`
  `>> net.b{1} = [0.2];`
- Giá trị ngõ vào:
  `>> p = [0.5, 0.25]';` % p' là chuyển vị của p
- Tính ngõ ra của nơ-ron:
  `>> y = sim(net, p)` → y = 1.2

  Hàm *sim* cho phép tính toán ngõ ra ANN có tên là *net* ứng với véctơ ngõ vào *p*.

- **Case study:** Sinh viên hãy thực hiện lại ví dụ trên bằng hàm *newp* (tạo 1 perceptron) và xác định:
  1. Ngõ ra của nơ-ron
  2. Hàm truyền mặc định của nơ-ron được tạo bởi hàm *newp* là gì? Thử thay đổi sang hàm tuyến tính và so sánh ngõ ra của nơ-ron này với kết quả của ví dụ trên.

Bài tập

1. Tìm hiểu các hàm truyền nơ-ron, được hỗ trợ bởi MATLAB. Gọi n là tham số biến thiên: n=-10:0.1:10, vẽ f(n) tương ứng của các hàm này.
2. Cho nơ-ron như hình vẽ, với:
   p=[1 2 3 4 5]';
   W=[1 .5 .25 1 2];
   b=.75;
   f là hàm logsig. Tính ngõ ra *a* của nơ-ron.



3. Tính ngõ ra *a* của nơ-ron trong câu 2, khi *f* là hàm *hardlim*
4. Dùng hàm *newp* của MATLAB – Neural nerwork toolbox để tạo nơ-ron trong câu 3 và so sánh kết quả mô phỏng với kết quả tính toán.
5. Lặp lại câu 4 với *f* là hàm *tansig*.

Chương 2
Cấu trúc của ANN

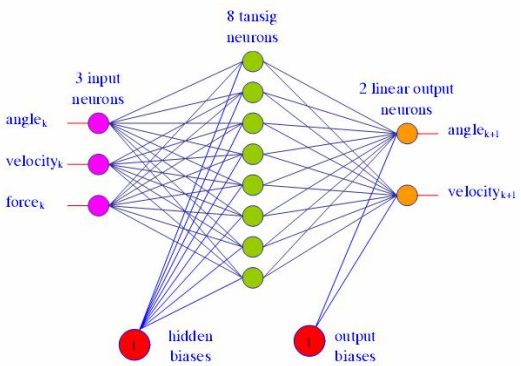
Giới thiệu
Khái niệm về cấu trúc của ANN
Cấu trúc mạng một lớp
Mô phỏng mạng một lớp bằng MATLAB
Cấu trúc mạng nhiều lớp
Mô phỏng mạng nhiều lớp bằng MATLAB
Bài tập

Giới thiệu

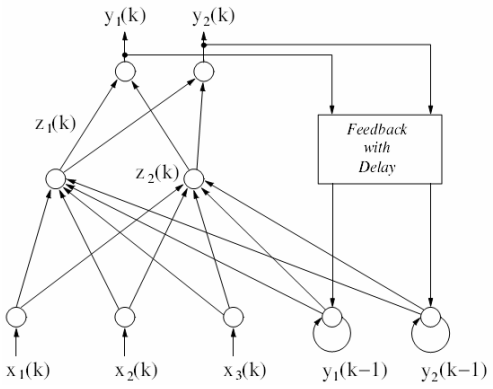
- Khái niệm về cấu trúc của ANN
- Phân loại ANN
- Cấu trúc mạng một lớp
- Tính toán trên mạng 1 lớp
- Mô phỏng mạng một lớp bằng MATLAB
- Cấu trúc mạng nhiều lớp
- Tính toán trên mạng nhiều lớp
- Mô phỏng mạng nhiều lớp bằng MATLAB
- Các ví dụ và bài tập

Khái niệm về cấu trúc ANN

- Mạng nơ-ron là một hệ thống gồm nhiều nơ-ron đơn lẻ kết nối với nhau, để thực hiện một chức năng tính toán nào đó.
- Tính năng của mạng phục thuộc vào: Cấu trúc, các trọng số kết nối và quá trình tính toán tại các nơ-ron đơn lẻ (hàm truyền).
- Mạng nơ-ron có thể học từ dữ liệu mẫu và tổng quát hóa dựa trên các mẫu đã học (mạng có khả năng nhớ).
- Ví dụ về cấu trúc mạng:



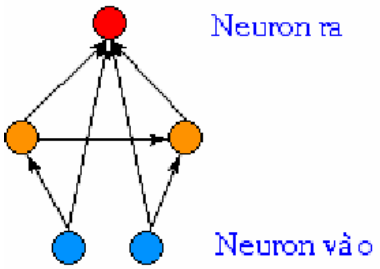
Mạng truyền thẳng



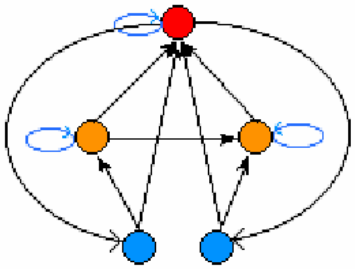
Mạng hồi qui

Phân loại ANN

- Dựa theo kiểu kết nối các nơ-ron: có 2 loại
  - Mạng truyền thẳng (feedforward NN): Kết nối không tạo thành chu trình
  - Mạng hồi qui (recurrent NN): Kết nối tạo thành các chu trình

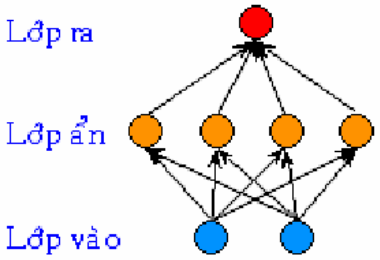


Mạng truyền thẳng



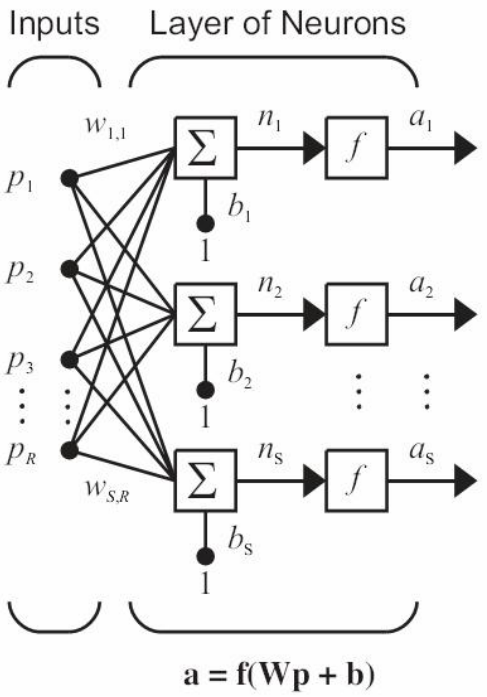
Mạng hồi qui

- Dựa vào số lớp: có 2 loại
  - Mạng 1 lớp
  - Mạng nhiều lớp: Gồm lớp vào, các lớp ẩn và lớp ra. Lớp vào không tính số lớp. Không kết nối trên cùng lớp; không nhảy lớp; không kết nối từ lớp dưới lên lớp trên.



Mạng nơ-ron 1 lớp (1)

- Mô hình mạng 1 lớp với *R* nơ-ron ở lớp vào và *S* nơ-ron ở lớp ra như hình vẽ.
  Lưu ý: lớp vào, chỉ dùng tiếp nhận thông tin, không tham gia quá trình tính toán nên không được kể là 1 lớp mạng.



Mạng nơ-ron 1 lớp

- Tính toán tại ngõ ra thứ i *(i=1,S)*:

$$n_i = \sum_{j=1}^{R} w_{i,j} p_j + b_i$$

$$a_i = f(n_i)$$

với *f* là hàm kích hoạt của nơ-ron.

  - *Thông thường, để đơn giản, các nơ-ron trên cùng lớp sẽ có hàm kích hoạt giống nhau.*
  - *Kí hiệu wi,j là trọng số kết nối từ nơ-ron thứ j đến nơ-ron thứ i*

Mạng nơ-ron 1 lớp (2)

- Biểu diễn ngõ ra của mạng: $\mathbf{a} = f(\mathbf{Wp} + \mathbf{b})$

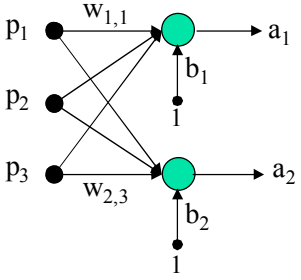
  $\mathbf{a} = [a_1, a_2, ..., a_S]$: vectơ ngõ ra; $\mathbf{p} = [p_1, p_2, ..., p_R]^T$: vectơ ngõ vào

  $\mathbf{b} = [b_1, b_2, ..., b_S]^T$: vectơ ngưỡng kích hoạt

$$W = \begin{vmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,R} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,R} \\ w_{S,1} & w_{S,2} & \cdots & w_{S,R} \end{vmatrix}$$

: Ma trận trọng số kết nối

Lưu ý: Kí hiệu $w_{i,j}$ là trọng số kết nối từ nơ-ron thứ j đến nơ-ron thứ i



- Ví dụ 1: Cho mạng 1 lớp, có: $\mathbf{p}=[2, 1.5, 3]^T$; $\mathbf{b}=[0.5, 0.6]^T$; 2 nơ-ron tuyến tính; ma trận trọng số:

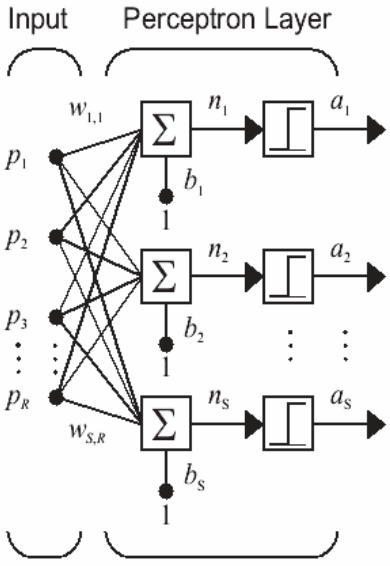
$$W = \begin{vmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \end{vmatrix} = \begin{vmatrix} .1 & .2 & .3 \\ .5 & .4 & .6 \end{vmatrix}$$

Ta có: $\mathbf{Wp}+\mathbf{b} = \begin{vmatrix} .1 & .2 & .3 \\ .5 & .4 & .6 \end{vmatrix} \begin{vmatrix} 2 \\ 1.5 \\ 3 \end{vmatrix} + \begin{vmatrix} .5 \\ .6 \end{vmatrix} = \begin{vmatrix} 1.9 \\ 4.0 \end{vmatrix}$; do f(n)=n $\Rightarrow$ a=f(**Wp**+**b**)=$[1.9, 4.0]^T$

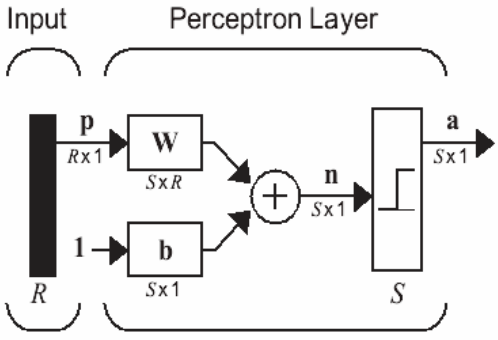
hay a_1= 1.9 và a_2=4.0

Mạng Perceptron 1 lớp (1)

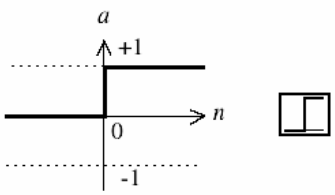
- Cấu trúc: 1 lớp với S nơ-ron, được kết nối với R ngõ vào thông qua ma trận trọng số W. Hàm truyền của các nơ-ron trong ví dụ này là hàm Hard-Limit.



a = hardlim(Wp + b)



a = hardlim(Wp + b)

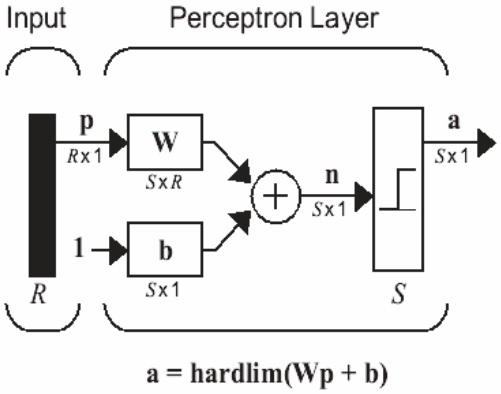


a = hardlim(n)

Hard-Limit Transfer Function

Mạng Perceptron 1 lớp (2)

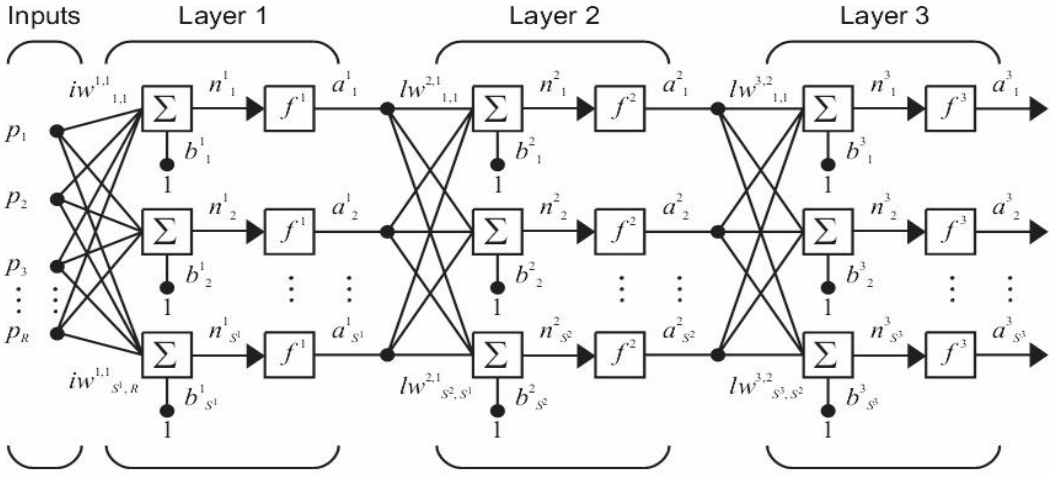
- Bài tập:
  Cho mạng Perceptron 1 lớp như hình vẽ, với:
  p=[1, 2, 0.2, 0.1]';
  b=[0.1, 0.3, 0.5]';
  W=[0.2, 0.1, 0.2, 0.1
  0.1, 0.3, 0.5, 0.1
  0.6, 0.4, 0.2, 0.4];



  Xác định:
  a) Số nơ-ron trên lớp vào và lớp Perceptron.
  b) Vector ngõ ra a.
  c) Sử dụng hàm ***newp*** của MATLAB để mô phỏng mạng này.

Mạng nơ-ron nhiều lớp (1)

- Mạng nơ-ron có thể được tổ chức thành nhiều lớp, với ngõ ra của lớp trước là ngõ vào của lớp sau.
- Quá trình tính toán trên mạng, được thực hiện lần lượt trên từng lớp.
- Tính toán trên mỗi lớp hoàn toàn giống như tính toán trên mạng 1 lớp.

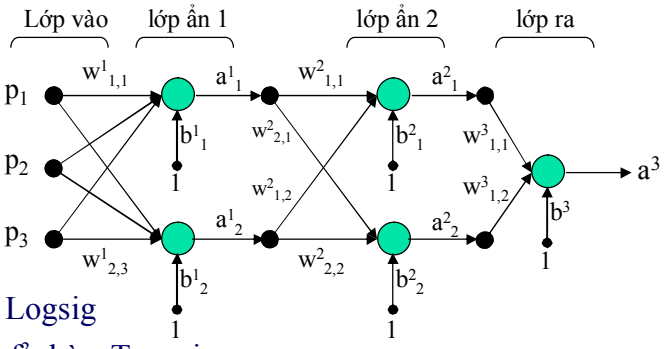


$$\mathbf{a}^3 = \mathbf{f}^3(\mathbf{LW}^{3,2}\mathbf{f}^2(\mathbf{LW}^{2,1}\mathbf{f}^1(\mathbf{IW}^{1,1}\mathbf{p}+\mathbf{b}^1)+\mathbf{b}^2)+\mathbf{b}^3)$$

Các chỉ số 1,2,3: kí hiệu số thứ tự của lớp

Mạng nơ-ron nhiều lớp (2)

- Ví dụ 2: Cho mạng 3 lớp như hình vẽ, tính giá trị ngõ ra, với:
  - Lớp vào: $\mathbf{p} = [2, 1.5, 3]^T$
  - Lớp ẩn 1:
    $\mathbf{b}^1 = [0.5, 0.6]^T$
    $\mathbf{W}^1 = [0.1, 0.2, 0.3; 0.5, 0.4, 0.6]$
    f^1: hàm purelin
  - Lớp ẩn 2:
    $\mathbf{b}^2 = [0.5, 0.6]^T$
    $\mathbf{W}^2 = [0.5, 0.2; 0.1, 0.6]$; f^2: hàm Logsig
  - Lớp ra: $b^3 = 0.2$; $W^3 = [0.4, 0.2]$; f^3: hàm Tangsig



Giải:

- Ngõ ra lớp ẩn 1: Giống Ví dụ ở phần mạng nơ-ron 1 lớp. Ta có a^1_1=1.9 và a^1_2=4
- Ngõ ra lớp ẩn 2: ngõ vào lớp 2 chính là ngõ ra lớp 1

$\mathbf{W}^2\mathbf{p}^2 + \mathbf{b}^2 = [0.5, 0.2; 0.1, 0.6]*[1.9, 4]^T + [0.5, 0.6]^T = [2.25, 3.19]^T$

$$\mathbf{a}^2 = f^2(\mathbf{W}^2\mathbf{p}^2+\mathbf{b}^2) = [f^2(2.25), f^2(3.19)]^T = \left[\frac{1}{1+e^{-2.25}} \quad \frac{1}{1+e^{-3.19}}\right]^T = [.9047, .9605]^T$$

hay: $a^2_1 = 0.9047$ và $a^2_2 = 0.9605$

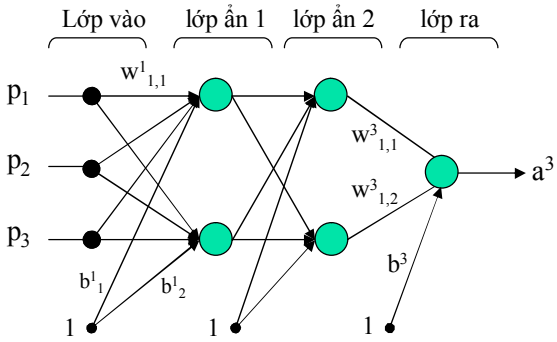
Mạng nơ-ron nhiều lớp (3)

- Ngõ ra lớp ẩn 3: ngõ vào lớp 3
  chính là ngõ ra của lớp 2, ta có:

$\mathbf{W}^3\mathbf{p}^3 + \mathbf{b}^3 = [0.4, 0.2]*[0.9047, 0.9605]^T + 0.2 = 0.7540$

$$a^3 = f^3(\mathbf{W}^3\mathbf{p}^3 + \mathbf{b}^3) = f^3(0.7540) = \frac{1-e^{-2*0.754}}{1+e^{-2*0-754}} = 0.6375$$

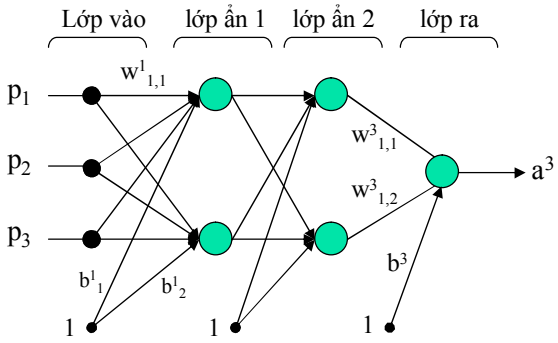
- Một dạng biểu diễn khác của mạng trên:



Mạng trên còn được gọi là mạng truyền thẳng nhiều lớp (Multilayer feedforward neural network). Mục tiêu của Giáo trình tập trung vào loại mạng này.

Mô phỏng mạng nhiều lớp (1)

- Yêu cầu: Mô phỏng mạng 3 lớp ở ví dụ 2 bằng MATLAB
- Giải:
  - Tạo mạng truyền thẳng: 3 ngõ vào, có giá trị nằm trong khoảng [-5, 5]; 2 nơ-ron tuyến tính trên lớp ẩn 1, 2 nơ-ron logsig trên lớp ẩn 2 và 1 nơ-ron tansig trên lớp ra. Dùng hàm *newff* của MATLAB:




```
>> net=newff([-5 5; -5 5; -5 5], [2 2 1], {'purelin', 'logsig', 'tansig'});
```

Giới hạn các ngõ vào | Số nơ-ron trên mỗi lớp | Hàm truyền tương ứng của các lớp

  - Gán tập trọng số $\mathbf{W}^1$ và ngưỡng $\mathbf{b}^1$ cho lớp ẩn 1 (nối từ lớp vào):

```
>> net.IW{1,1} = [0.1, 0.2, 0.3; 0.5, 0.4, 0.6];  % Input Weights
>> net.b{1} = [0.5, 0.6]';
```


Mô phỏng mạng nhiều lớp (2)

- Gán tập trọng số $\mathbf{W}^2$ và ngưỡng $\mathbf{b}^2$ cho lớp ẩn 2:

```
>> net.LW{2,1} = [0.5, 0.2; 0.1, 0.6];  % Layer Weights 1 → 2
>> net.b{2} = [0.5, 0.6]';
```

- Gán tập trọng số $\mathbf{W}^3$ và ngưỡng $\mathbf{b}^3$ cho lớp ra:

```
>> net.LW{3,2} = [0.4, 0.2];          % Layer Weights 2→ 3
>> net.b{3} = [0.2];
```

- Tính toán giá trị ngõ ra (mô phỏng mạng):

```
>> p = [2, 1.5, 3]';
>> y = sim(net, p)
```

→ **y = 0.6375,** đúng với kết quả tính toán

Lưu ý:

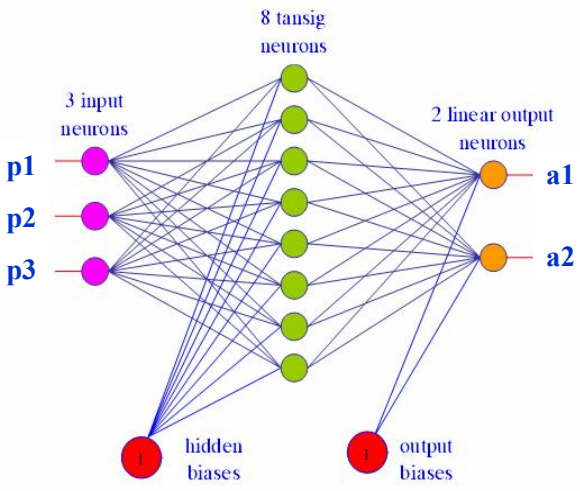
- Các giá trị trọng số và ngưỡng của các nơ-ron có được do quá trình huấn luyện. Vấn đề này sẽ được đề cập ở chương 3.
- Các giá trị này được MATLAB quản lý dưới dạng mãng nhiều chiều. Lớp ẩn thứ nhất được nối trực tiếp với lớp vào, nên các các trọng số của nó được gọi là trọng số lớp vào, net.IW – Input weights. Các trọng số lớp ẩn được kí hiệu là net.LW - Layer weights

Các hàm tạo ANN của MATLAB

New Neural Networks	
newc	Create a competitive layer.
newcf	Create a cascade-forward backpropagation network.
newelm	Create an Elman backpropagation network.
newff	Create a feed-forward backpropagation network.
newfftd	Create a feed-forward input-delay backprop network.
newgrnn	Design a generalized regression neural network.
newhop	Create a Hopfield recurrent network.
newlin	Create a linear layer.
newlind	Design a linear layer.
newlvq	Create a learning vector quantization network
newp	Create a perceptron.
newpnn	Design a probabilistic neural network.
newrb	Design a radial basis network.
newrbe	Design an exact radial basis network.
newsom	Create a self-organizing map.

Bài tập

Cho một mạng truyền thẳng 2 lớp như hình vẽ. Trong đó, lớp vào có 3 nơ-ron nhận giá trị trong khoảng [-1, 1]; lớp ẩn có 8 nơ-ron tansig và lớp ra có 2 nơ-ron tuyến tính.



Giả sử, ta có:

IW{1,1}=[1 -1 2; -1 1 1; -2 -1 1; 1 1 2;
-1 2 1; -2 -1 1; 1 1 2; 2 2 1];

b{1}=[-1 2 -2 1 2 1 -1 1]'; LW{2,1}=[-1 1 2 2 1 1 2 1; -2 -2 -1 1 1 2 -1 1];

b{2}= [-1 1]'; p=[1 -1 0.5]';

a) Tính các ngõ ra a_1 và a_2.

b) Sử dụng hàm *newff* để mô phỏng mạng nơ-ron trên và so sánh ngõ ra mô phỏng với kết quả tính toán ở câu a.

c) Sử dụng lệnh *help init* và cho biết hàm *init* có tác dụng gì đối với ANN.

d) Mỗi lần áp dụng hàm *init* cho mạng ở câu b, thử quan sát ma trận trọng số của lớp ẩn và có nhận xét gì?

e) Lặp lại câu a và b khi thay đổi hàm truyền của 2 nơ-ron ở ngõ ra thành hàm logsig.