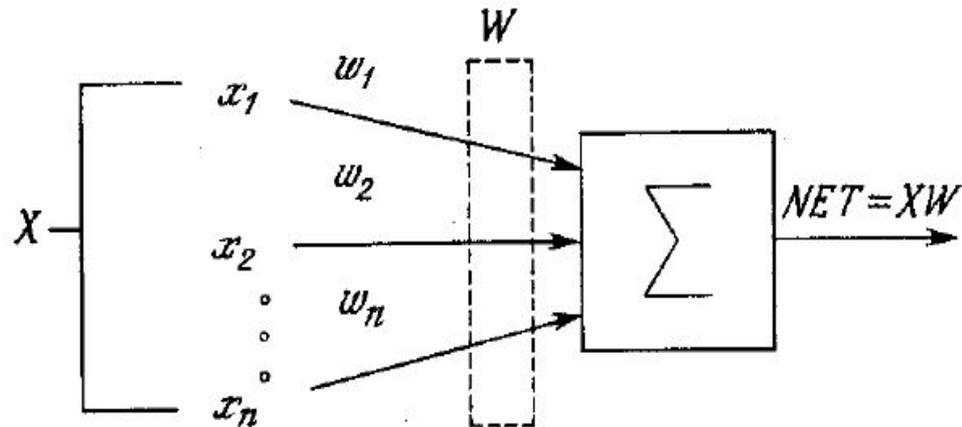


# Нейронные сети

Реализация нейронных сетей в среде MatLab

# Основные понятия. Нейрон.

---



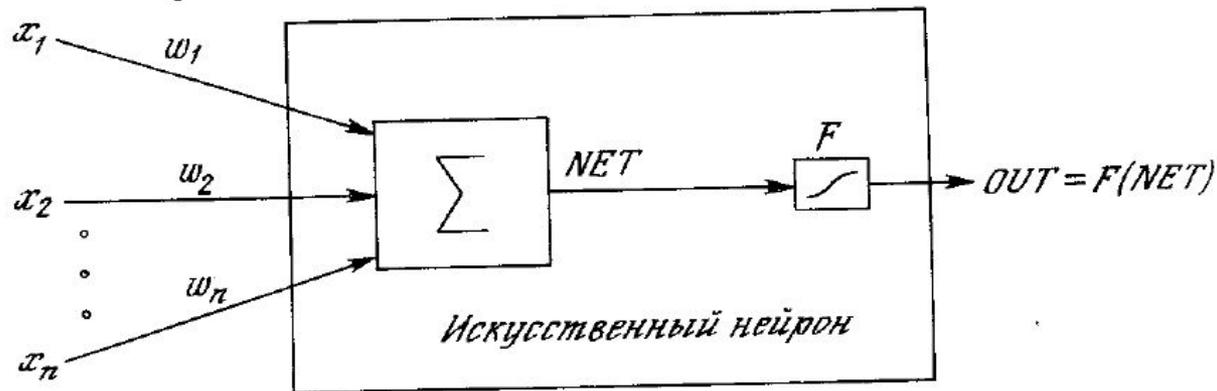
- $X$  – входной вектор;
- $W$  – вектор весов;
- $\Sigma$  – суммирующий блок;
- $NET$  – выход.



# Основные понятия.

## Нейрон.

---



- NET – промежуточный выход;
- F – активационная функция;
- OUT – выход.

Активационная функция должна иметь ограниченное множество значений, к примеру  $[0; 1]$ . Это необходимое условие для нормализации выходного значения.

---

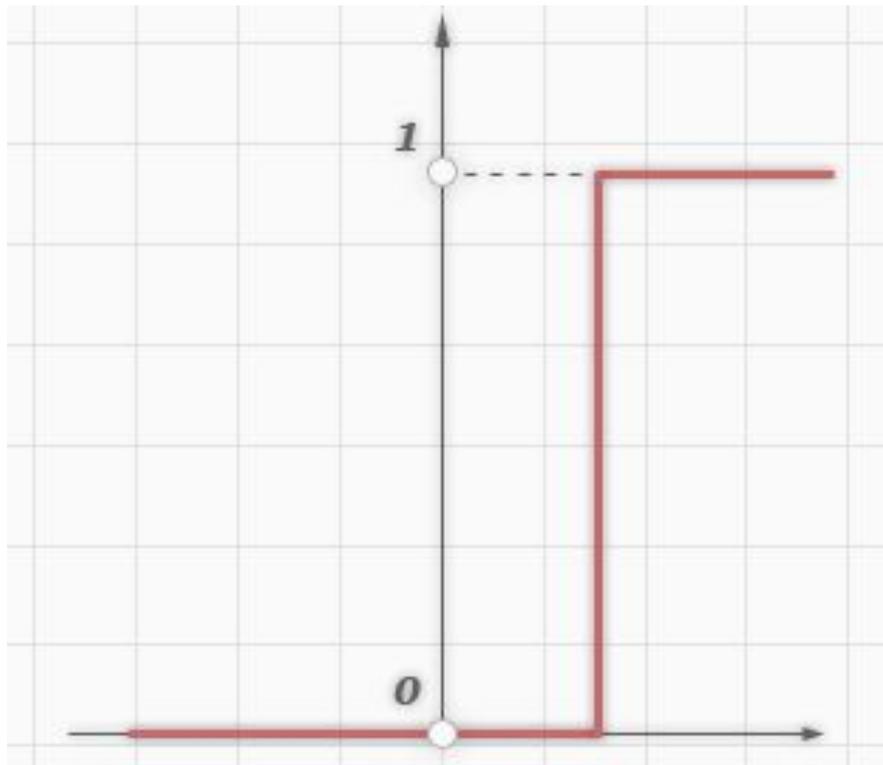


# Основные понятия.

## Примеры активационных функций.

---

- Единичный скачок или жесткая пороговая функция

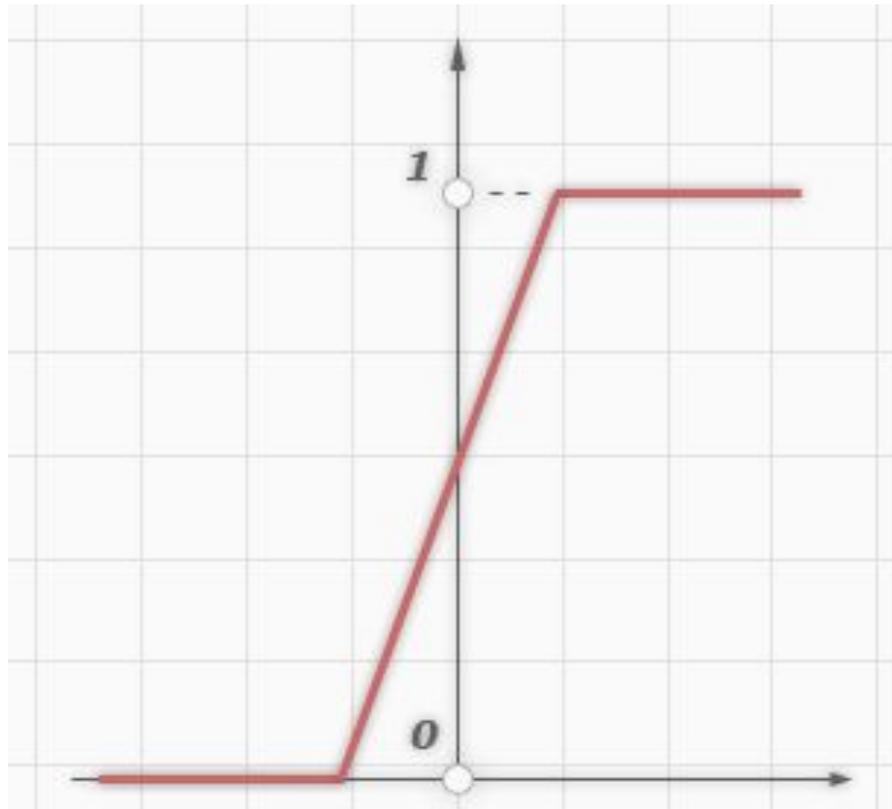


# Основные понятия.

## Примеры активационных функций.

---

□ Линейный порог или гистерезис

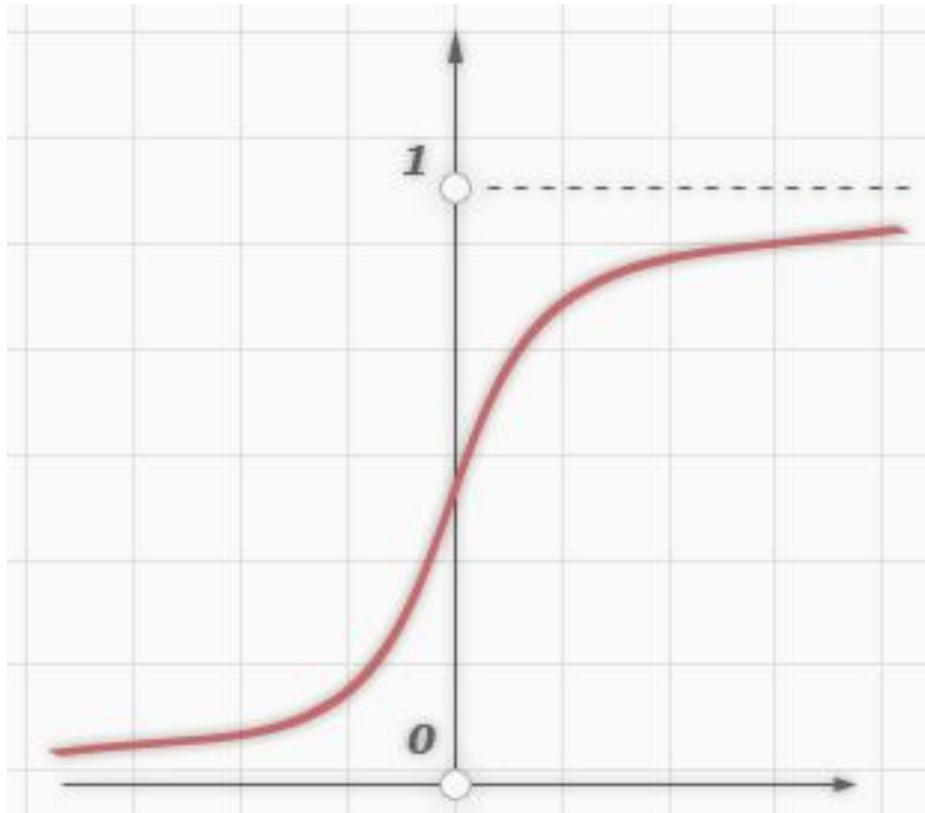


# Основные понятия.

## Примеры активационных функций.

---

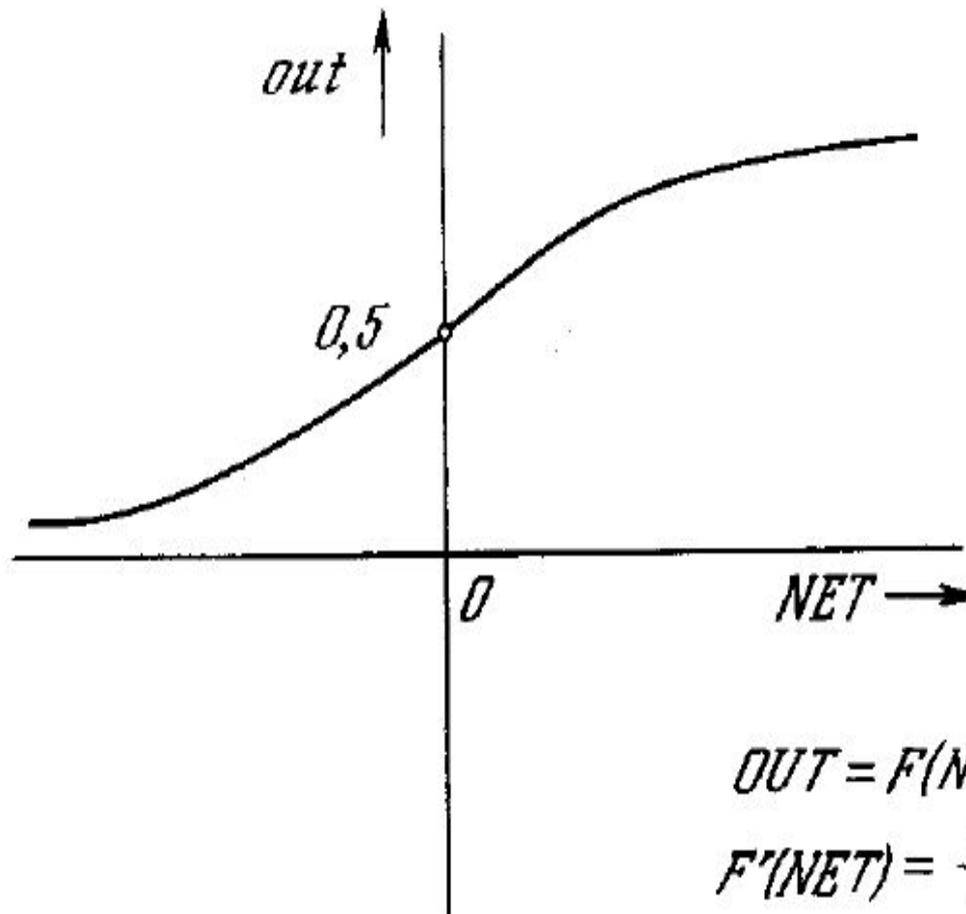
□ Сигмоид



Основные понятия.

Примеры активационных функций.

---



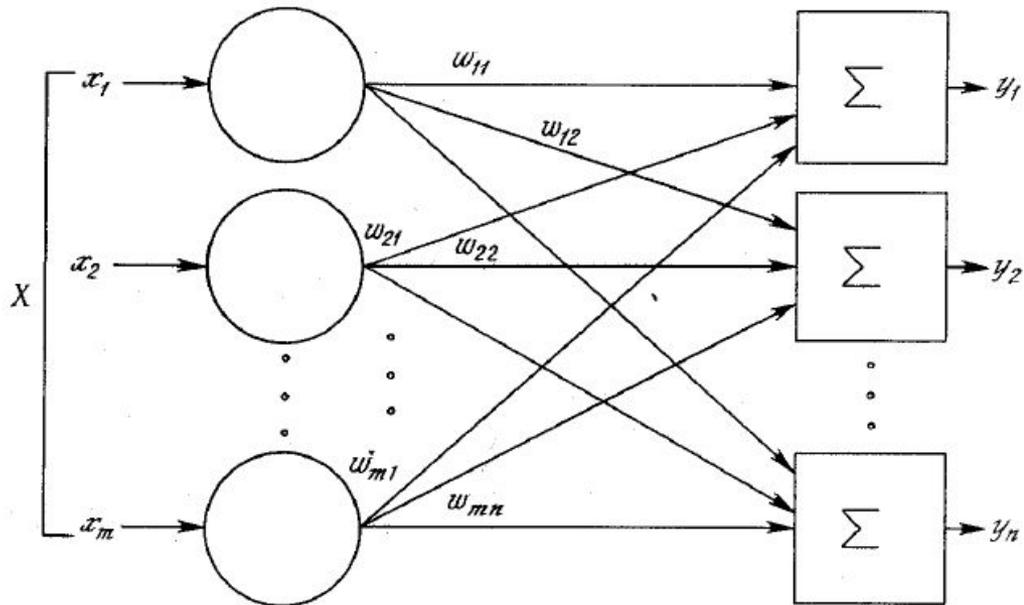
$$OUT = F(NET) = 1 / (1 + e^{-NET})$$
$$F'(NET) = \frac{\partial OUT}{\partial NET} = OUT(1-OUT)$$



# Основные понятия.

## Нейронная сеть. Однослойная сеть.

---

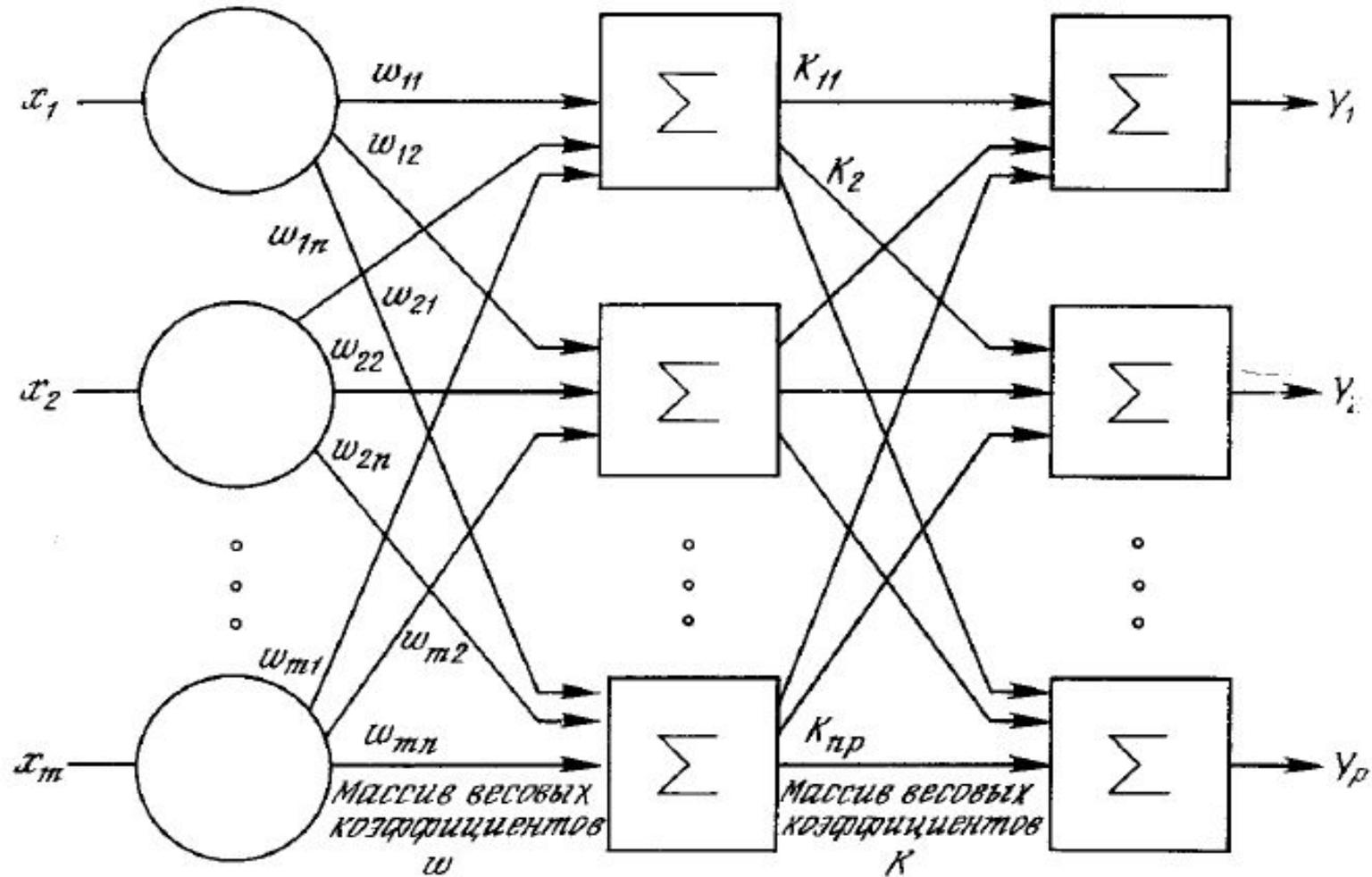


- $X$  – входной вектор;
  - $w_{ij}$  – вес, определенный для значения  $x_i$  у нейрона  $y_j$ ;
  - $Y$  – выходной вектор.
- 



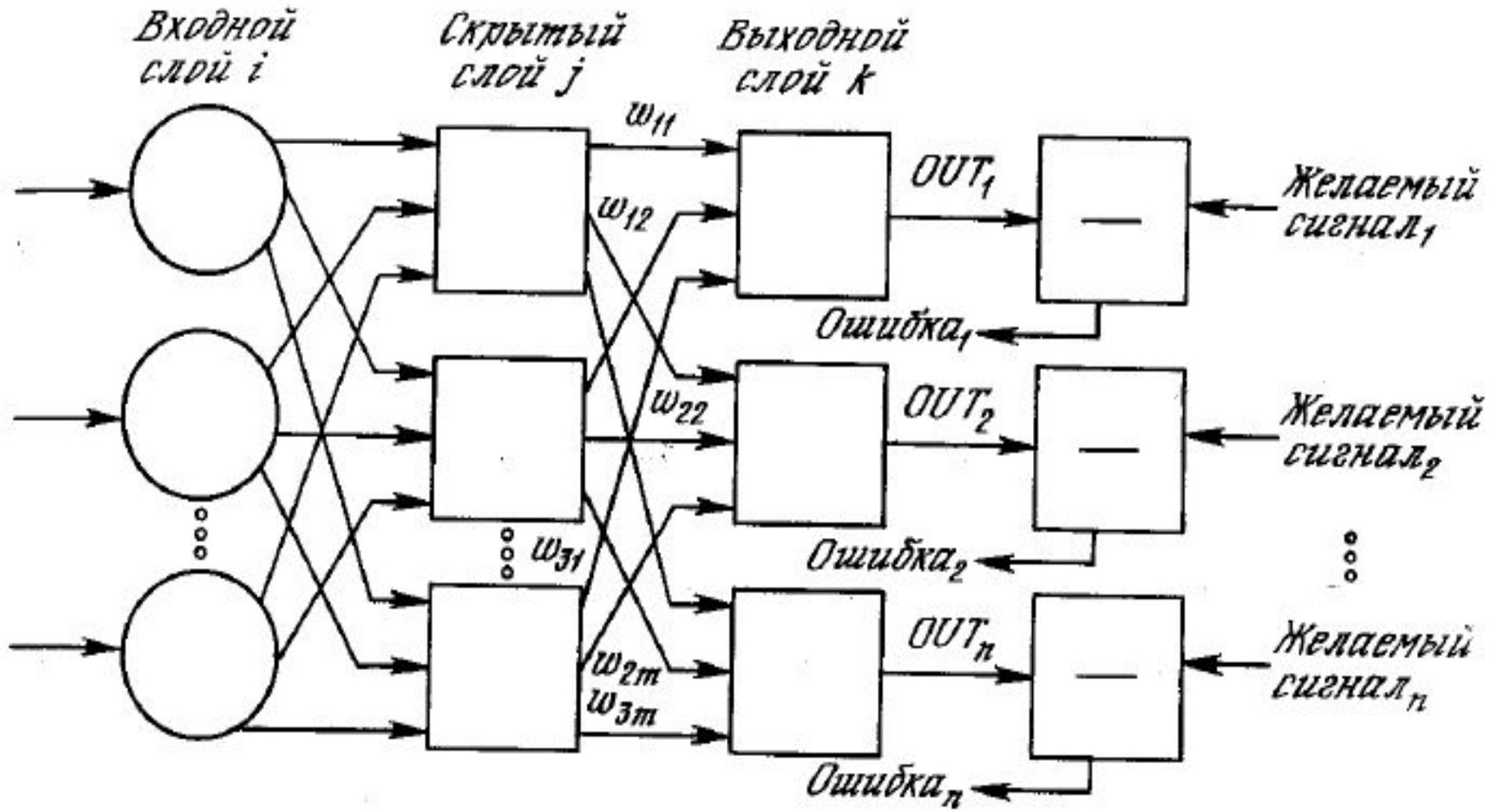
# Основные понятия.

## Нейронная сеть. Многослойная сеть.



Обучение сети.

Нейронная сеть с обратным распространением.



# Обучение сети.

## Алгоритм.

---

- Обучение сети обратного распространения требует выполнения следующих операций:
1. Выбрать очередную обучающую пару из обучающего множества; подать входной вектор на вход сети.
  2. Вычислить выход сети.
  3. Вычислить разность между выходом сети и требуемым выходом (целевым вектором обучающей пары).
  4. Подкорректировать веса сети так, чтобы минимизировать ошибку.
  5. Повторять шаги с 1 по 4 для каждого вектора обучающего множества до тех пор, пока ошибка на всем множестве не достигнет приемлемого уровня.
- 



# Обучение сети.

## Алгоритм (продолжение).

---

- Новое значение весов считается по формуле:

$$W_{pq} = W_{pq} + \mu \delta_q OUT$$

- где  $p$  и  $q$  – номера нейронов.

- Для выходного слоя:

$$\delta = OUT(1 - OUT)(T - OUT)$$

- где  $T$  – желаемый результат, а  $OUT(1-OUT)$  – производная сигмоида.

- Для остальных слоёв:

$$\delta = OUT_{p,i}(1 - OUT_{p,i}) \left[ \sum_q \delta_q W_{pq} \right]$$

---

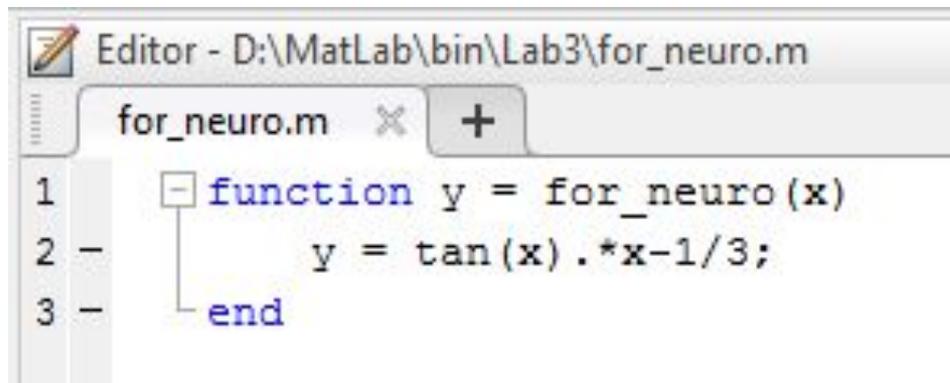


# Работа в пакете MATLAB.

## Подготовка обучающих выборок

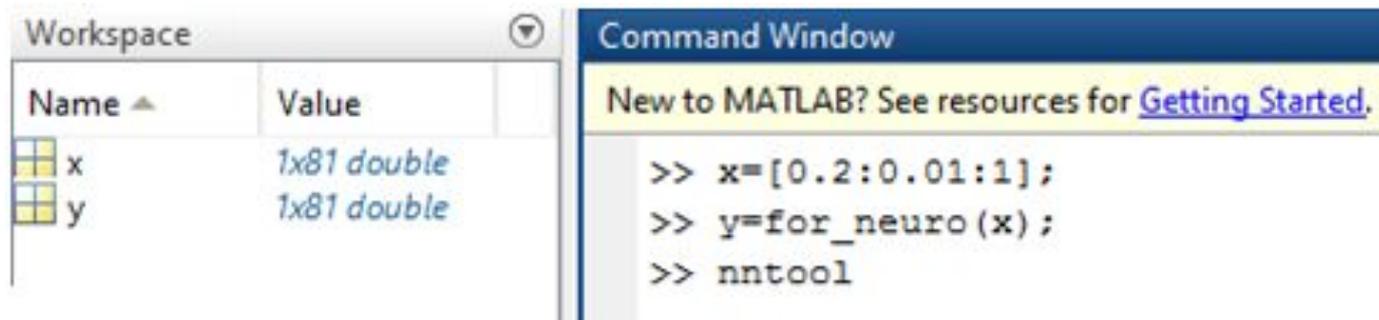
---

- Скрипт с описанием исследуемой функции



```
Editor - D:\MatLab\bin\Lab3\for_neuro.m
for_neuro.m x +
1 function y = for_neuro(x)
2     y = tan(x).*x-1/3;
3 end
```

- Генерация выборок



Workspace

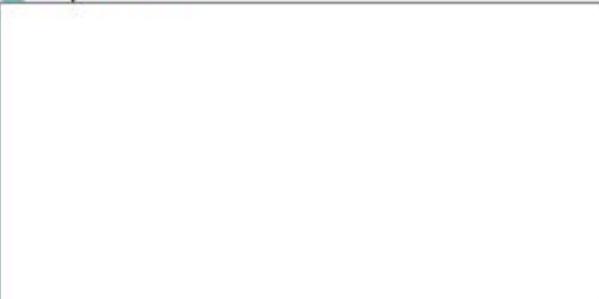
Name	Value
x	1x81 double
y	1x81 double

Command Window

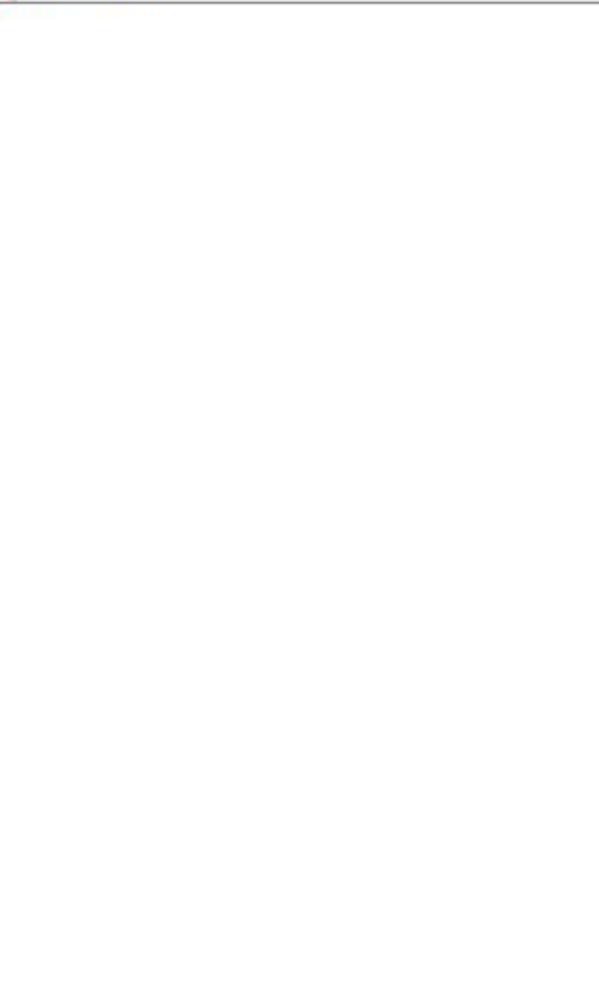
```
New to MATLAB? See resources for Getting Started.
>> x=[0.2:0.01:1];
>> y=for_neuro(x);
>> nntool
```



Input Data:



Networks



Output Data:



Target Data:



Error Data:



Input Delay States:



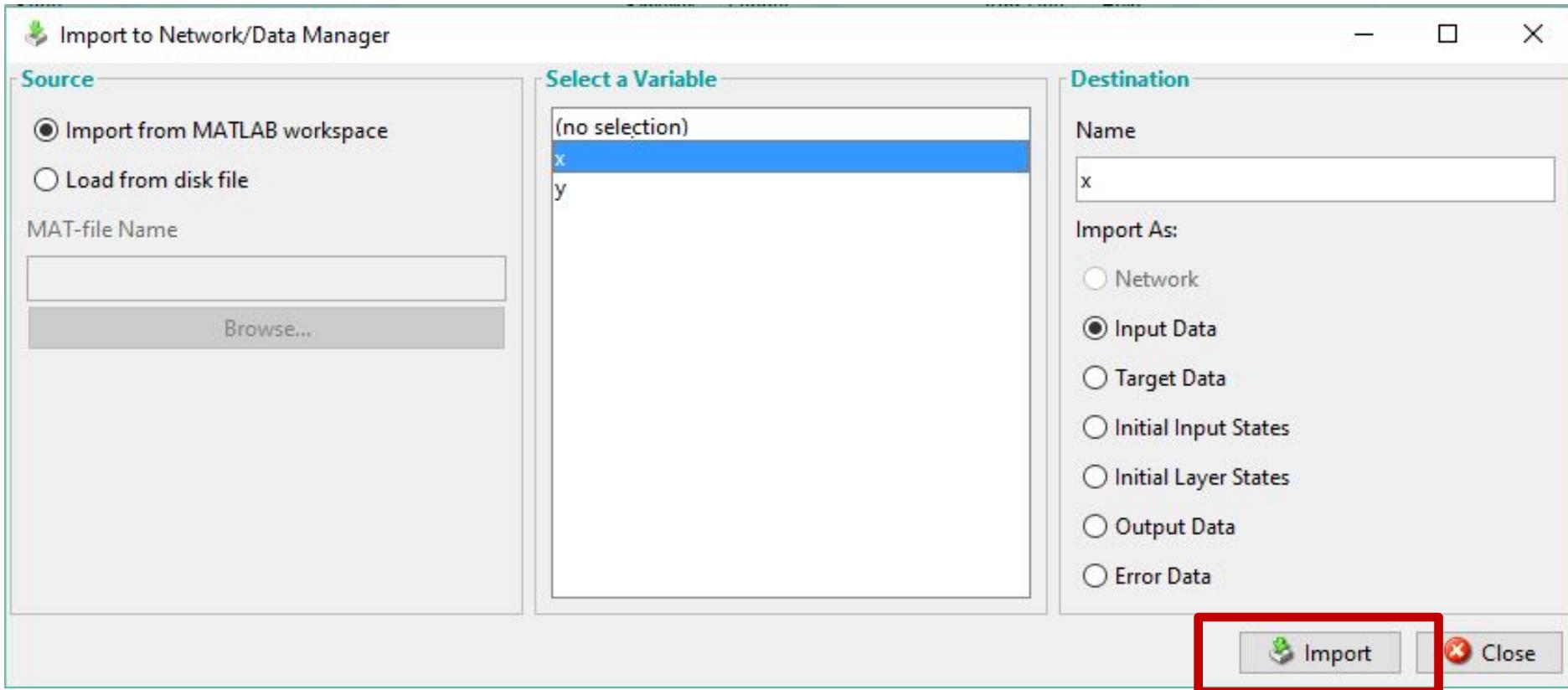
Layer Delay States:



# Работа в пакете MATLAB.

## Импорт данных и создание нейронной сети.

---



Input Data:

x

Target Data:

y

Input Delay States:

Networks

Output Data:

Error Data:

Layer Delay States:

Import...

New...

Open...

Export...

Delete

Help

Close

Name

net

Network Properties

Network Type:

Feed-forward backprop

Input data:

x

Target data:

y

Training function:

TRAINLM

Adaption learning function:

LEARNGDM

Performance function:

MSE

Number of layers:

2

Properties for:

Layer 1

Number of neurons:

5

Transfer Function:

TANSIG

View

Restore Defaults

Help

Create

Close

# Работа в пакете MATLAB.

## Параметры нейронной сети.

---

- **Network type** – список сетей, доступных для работы.
  - **Input Data, Target Data** – данные, представляющие входной и выходной векторы соответственно.
  - **Training function** – обучающая функция (по умол. метод оптимизации Левенберга-Маркара).
  - **Adaptation learning function** - функция, отвечающая за обновление весов и смещений сети в процессе обучения (по умол. метод градиентного спуска)
  - **Performance function** - функция оценки качества обучения (по умол. среднеквадратичная ошибка)
  - **Number of layers** – число слоев сети
- 



# Работа в пакете MATLAB.

## Параметры нейронной сети.

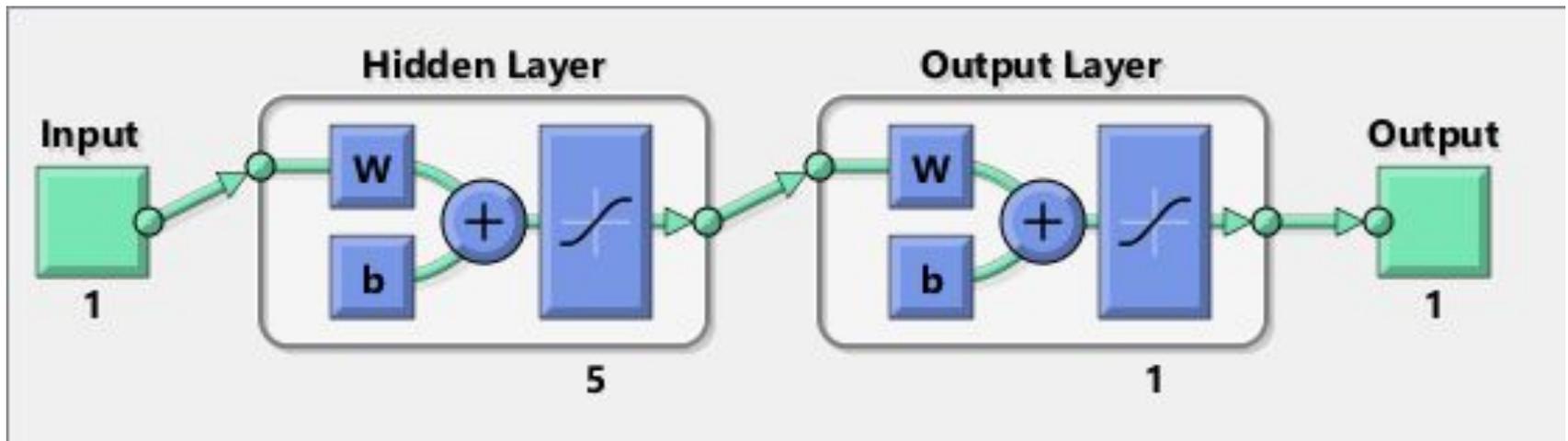
---

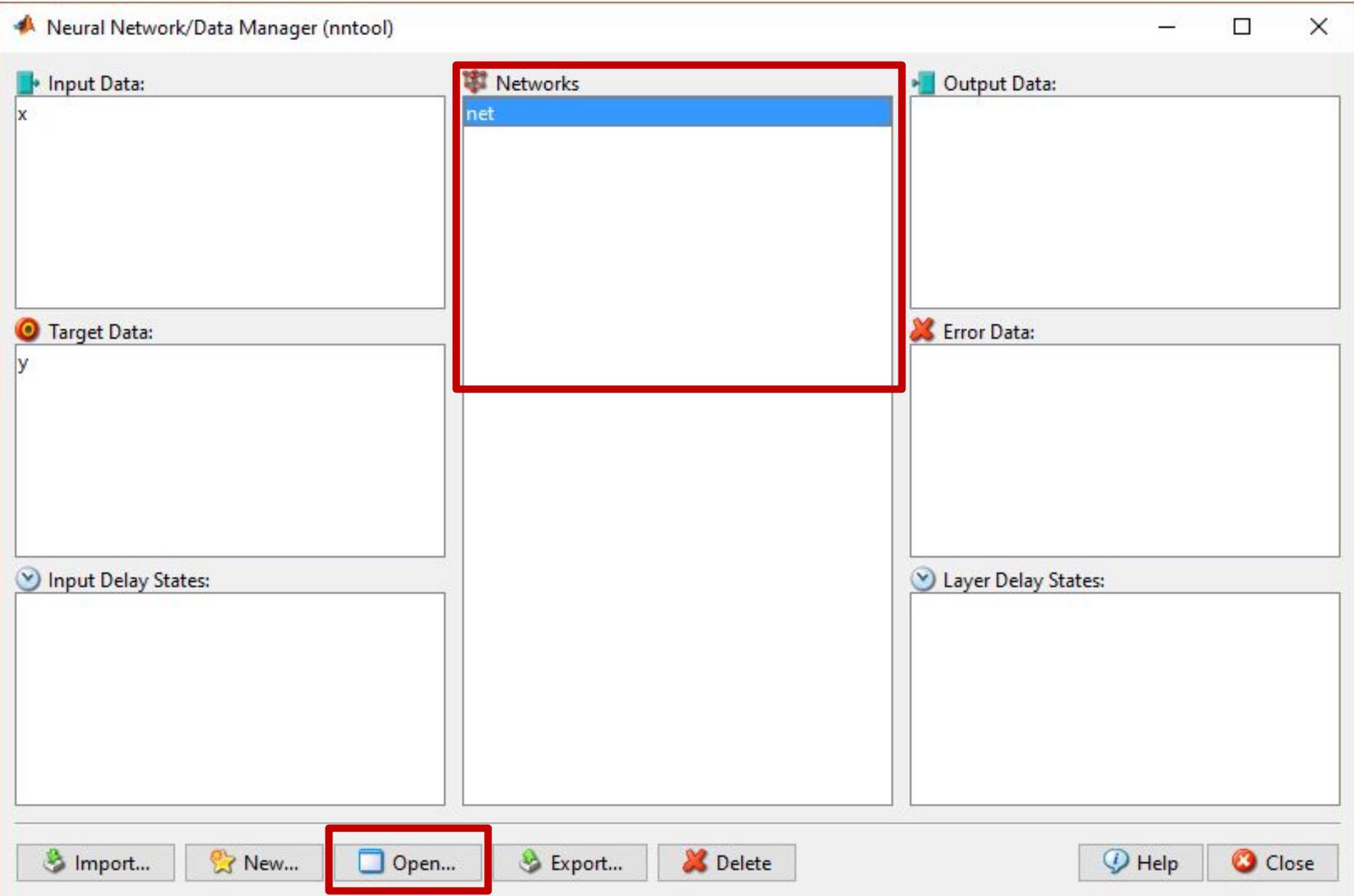
- **Number of neurons** – число нейронов.
- **Transfer function** – активационная функция.



# Работа в пакете MATLAB. Нейронная сеть.

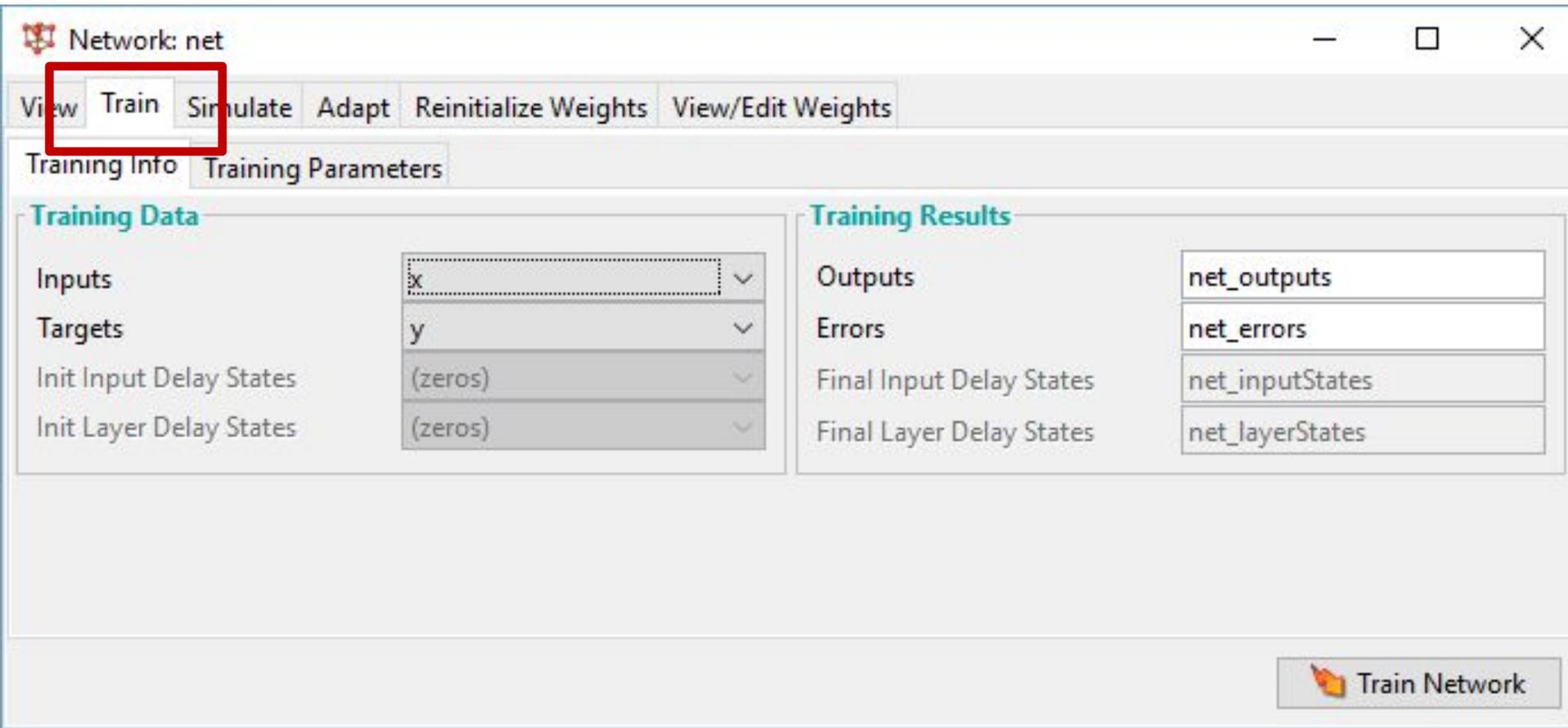
---





# Работа в пакете MATLAB. Обучение нейронной сети.

---



The screenshot shows the MATLAB Neural Network Designer window titled "Network: net". The "Train" button in the top toolbar is highlighted with a red rectangle. Below the toolbar, there are two tabs: "Training Info" and "Training Parameters". The "Training Data" section on the left contains the following settings:

Parameter	Value
Inputs	x
Targets	y
Init Input Delay States	(zeros)
Init Layer Delay States	(zeros)

The "Training Results" section on the right contains the following settings:

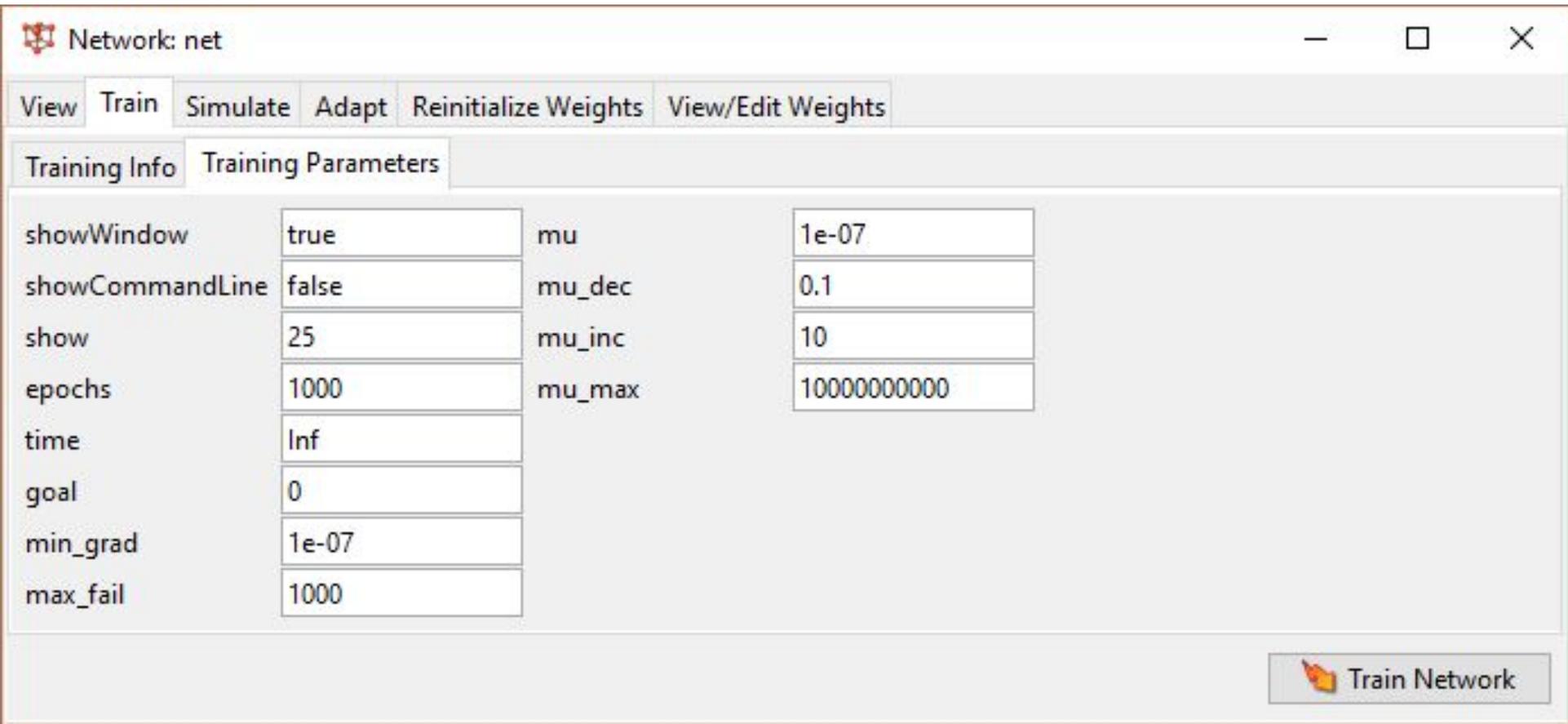
Parameter	Value
Outputs	net_outputs
Errors	net_errors
Final Input Delay States	net_inputStates
Final Layer Delay States	net_layerStates

At the bottom right of the window, there is a "Train Network" button with a flame icon.



# Работа в пакете MATLAB. Обучение нейронной сети.

---



The screenshot shows the MATLAB Network Trainers GUI for a neural network named 'net'. The 'Train' tab is active, and the 'Training Parameters' sub-tab is selected. The parameters are displayed in a table format with input fields for each value.

showWindow	<input type="text" value="true"/>	mu	<input type="text" value="1e-07"/>
showCommandLine	<input type="text" value="false"/>	mu_dec	<input type="text" value="0.1"/>
show	<input type="text" value="25"/>	mu_inc	<input type="text" value="10"/>
epochs	<input type="text" value="1000"/>	mu_max	<input type="text" value="10000000000"/>
time	<input type="text" value="Inf"/>		
goal	<input type="text" value="0"/>		
min_grad	<input type="text" value="1e-07"/>		
max_fail	<input type="text" value="1000"/>		

At the bottom right of the window, there is a button labeled 'Train Network' with a flame icon.



# Работа в пакете MATLAB.

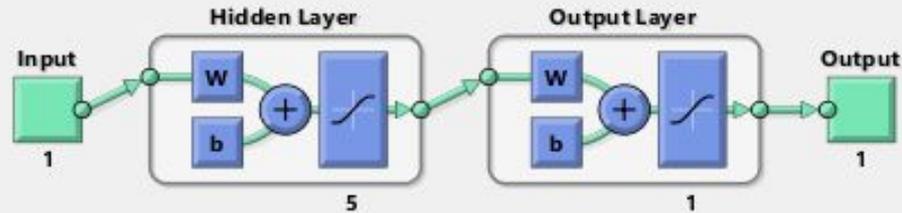
## Параметры обучения нейронной сети.

---

- ▣ **showWindow** – вывод процесса обучения в графическом режиме.
  - ▣ **showCommandLine** - вывод процесса обучения в командную строку.
  - ▣ **show** - период обновления графика кривой обучения, выраженный числом эпох
  - ▣ **epoch** – число эпох, по прошествии которых обучение заканчивается.
  - ▣ **time** – время, по истечении которого обучение прекращается
  - ▣ **goal** – значение функции ошибки, при которой цель будет считаться достигнутой.
  - ▣ **min\_grad** – минимальный градиент.
  - ▣ **max\_fail** – максимальное число ошибок.
  - ▣ **mu** - начальное значение  $\mu$ .
  - ▣ **mu\_dec** - коэффициент убывания  $\mu$ .
  - ▣ **mu\_inc** - коэффициент возрастания  $\mu$ .
  - ▣ **mu\_max** - максимальное значение  $\mu$ .
- 



## Neural Network



## Algorithms

Data Division: Random (dividerand)  
Training: Levenberg-Marquardt (trainlm)  
Performance: Mean Squared Error (mse)  
Calculations: MEX

## Progress

Epoch:	0	1000 iterations	1000
Time:		0:00:02	
Performance:	0.200	3.35e-07	0.00
Gradient:	0.349	6.50e-05	1.00e-07
Mu:	1.00e-07	1.00e-09	1.00e+10
Validation Checks:	0	659	1000

## Plots

- Performance (plotperform)
- Training State (plottrainstate)
- Regression (plotregression)

Plot Interval:  1 epochs

 **Maximum epoch reached.**

 Stop Training

 Cancel

Input Data:  
x

Target Data:  
y

Input Delay States:

Networks  
net

Output Data:  
net\_output

Error Data:  
net\_errors

Layer Delay States:

Import...

New...

Open...

Export...

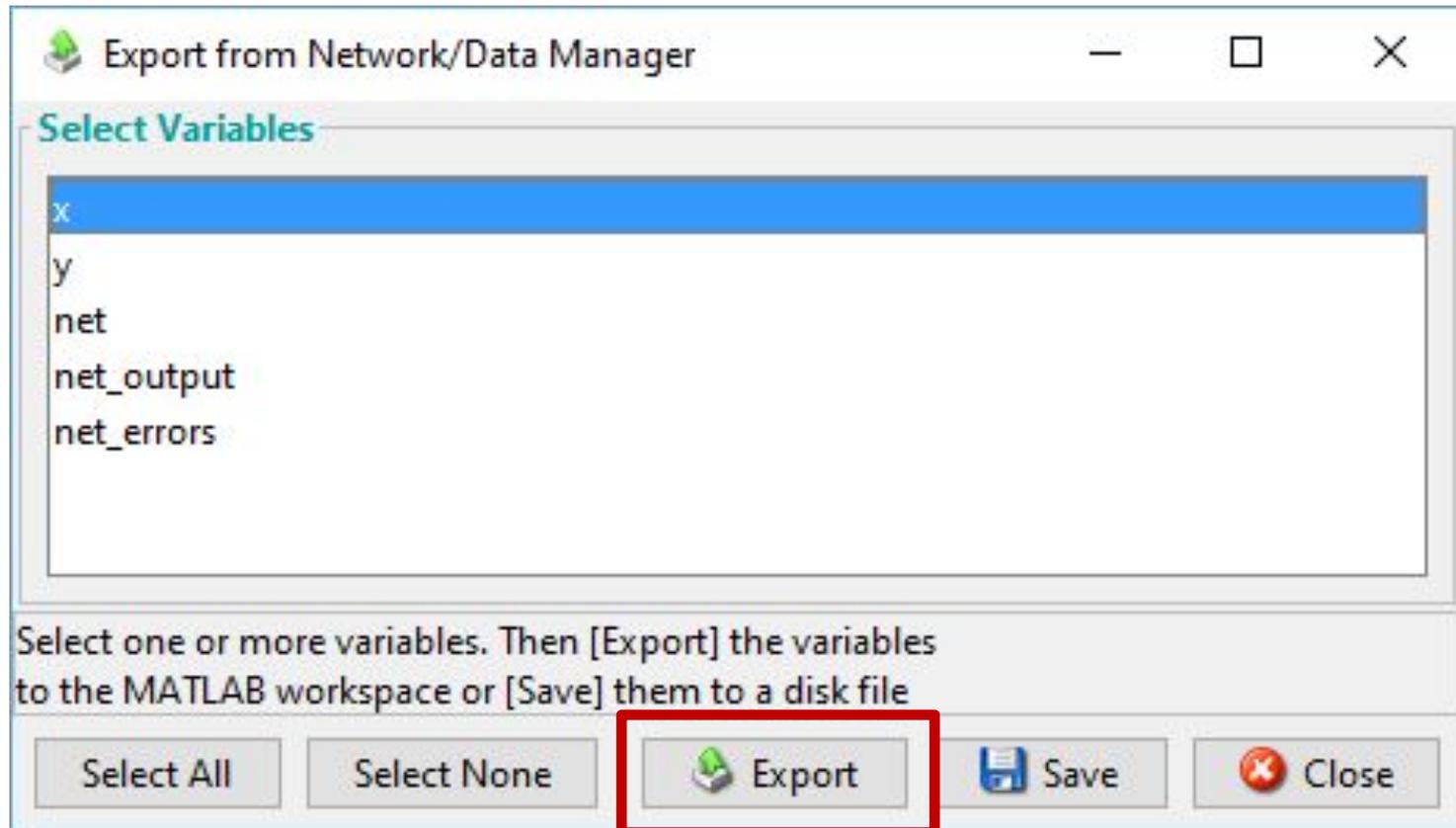
Delete

Help

Close

# Работа в пакете MATLAB. Экспорт данных.

---

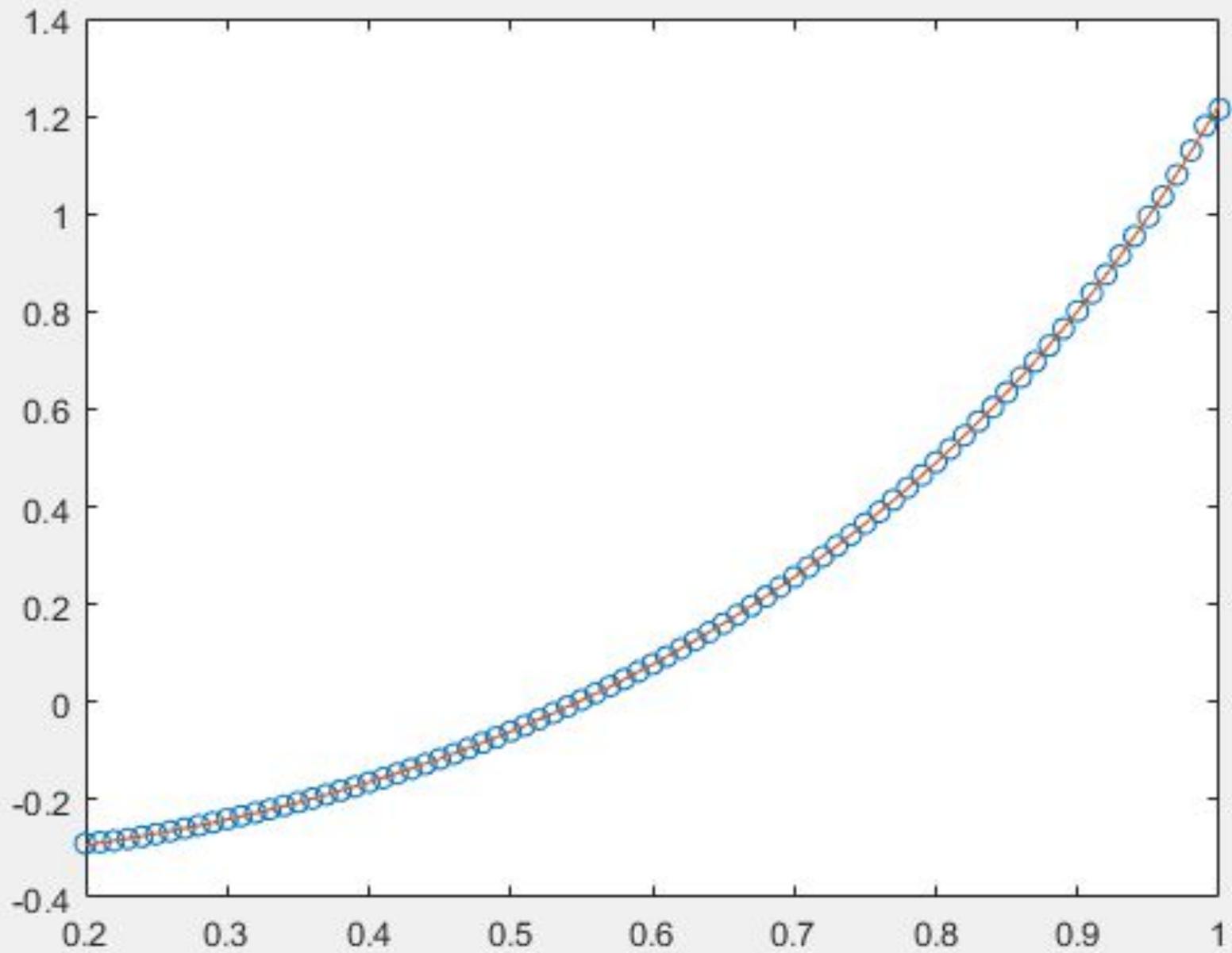


# Работа в пакете MATLAB. Построение графиков.

---

- `plot (x, y);`
- `hold on;`
- `plot (x, net_output, 'o');`





Input Data:  
x1

Target Data:

Input Delay States:

Networks  
net

Output Data:

Error Data:

Layer Delay States:

Import...

New...

Open...

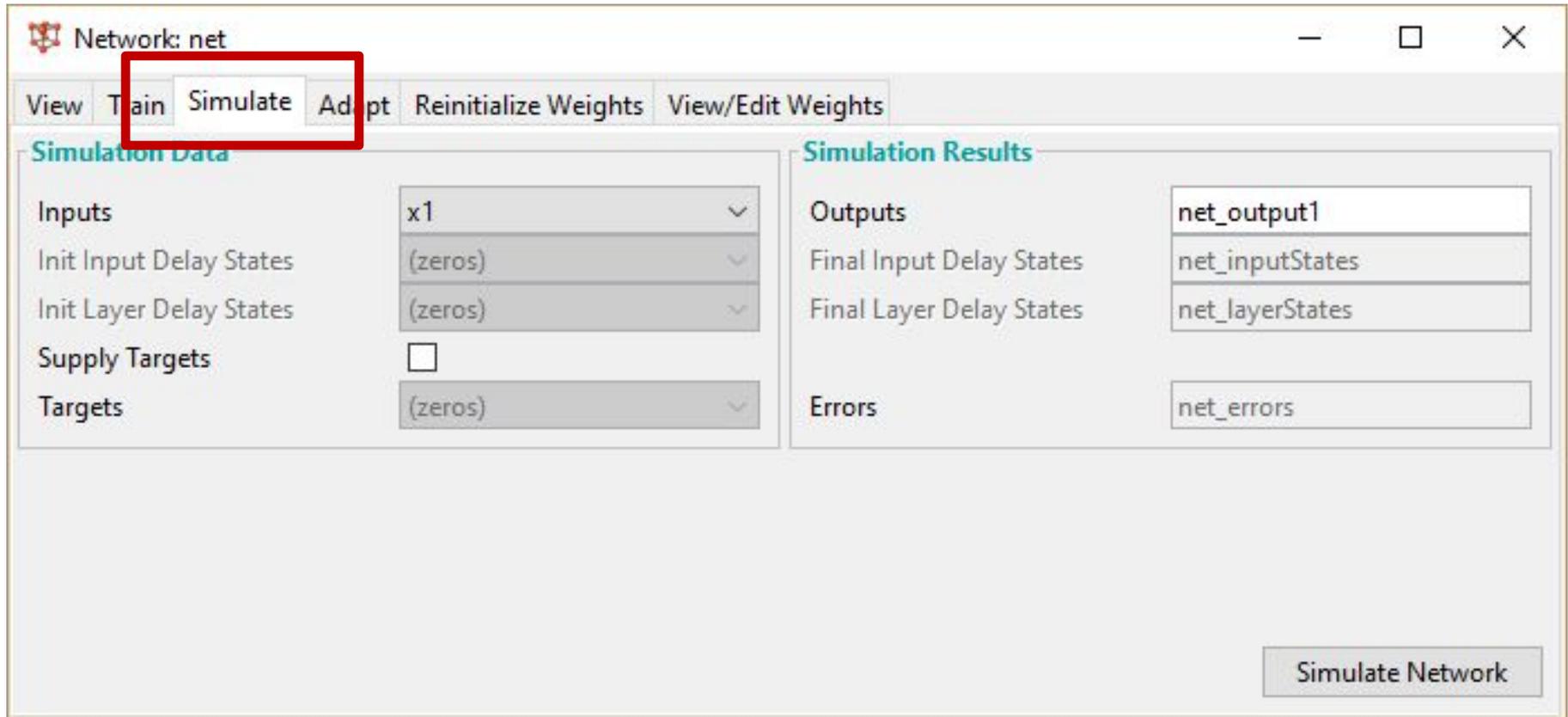
Export...

Delete

Help

Close

# Работа в пакете MATLAB. Проверка нейронной сети.



The screenshot shows the MATLAB Simulink interface for a neural network model named 'Network: net'. The 'Simulate' button in the top toolbar is highlighted with a red rectangle. The window is divided into two main sections: 'Simulation Data' and 'Simulation Results'.

**Simulation Data:**

Inputs	x1
Init Input Delay States	(zeros)
Init Layer Delay States	(zeros)
Supply Targets	<input type="checkbox"/>
Targets	(zeros)

**Simulation Results:**

Outputs	net_output1
Final Input Delay States	net_inputStates
Final Layer Delay States	net_layerStates
Errors	net_errors

A 'Simulate Network' button is located at the bottom right of the window.



Input Data:  
x1

Target Data:

Input Delay States:

Networks  
net

Output Data:  
net\_output1

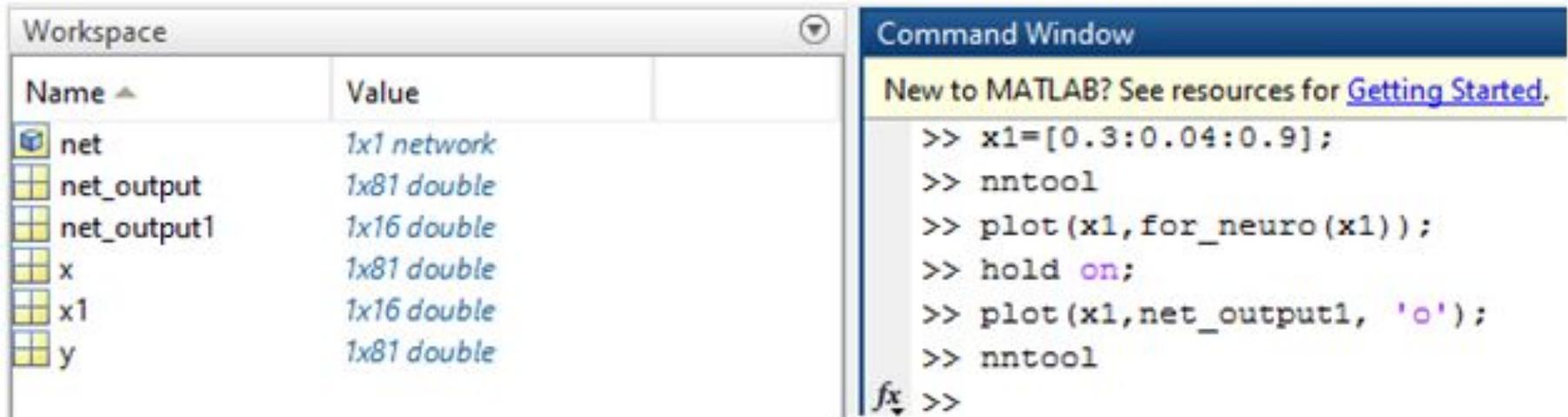
Error Data:

Layer Delay States:

# Работа в пакете MATLAB.

## Проверка нейронной сети.

---



The screenshot displays the MATLAB environment with two main windows: the Workspace and the Command Window.

**Workspace:** A table showing the current variables in the workspace.

Name	Value
net	1x1 network
net_output	1x81 double
net_output1	1x16 double
x	1x81 double
x1	1x16 double
y	1x81 double

**Command Window:** Shows the execution of MATLAB commands for testing a neural network.

```
>> x1=[0.3:0.04:0.9];  
>> nntool  
>> plot(x1,for_neuro(x1));  
>> hold on;  
>> plot(x1,net_output1, 'o');  
>> nntool  
fx >>
```



