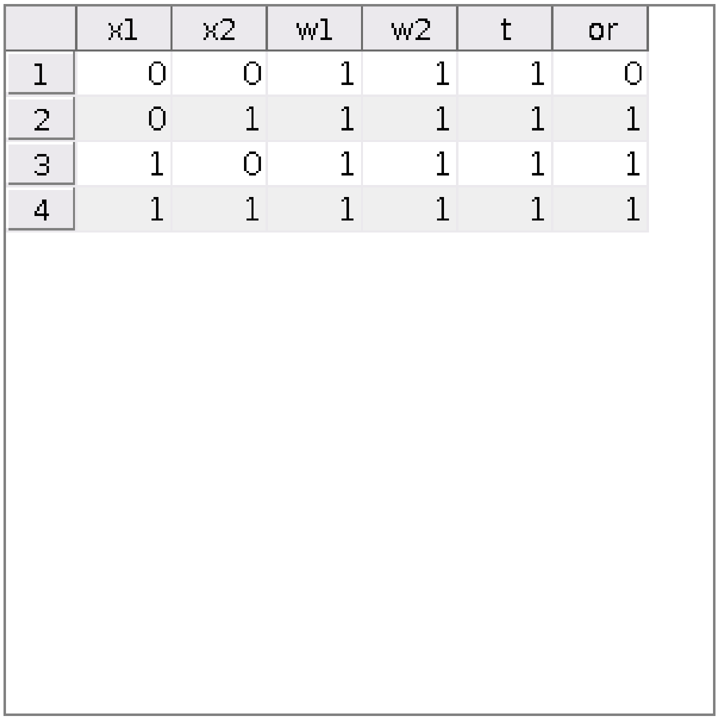
```
% ASSIGNMENT 3
% Akash Rout (21103080)
```

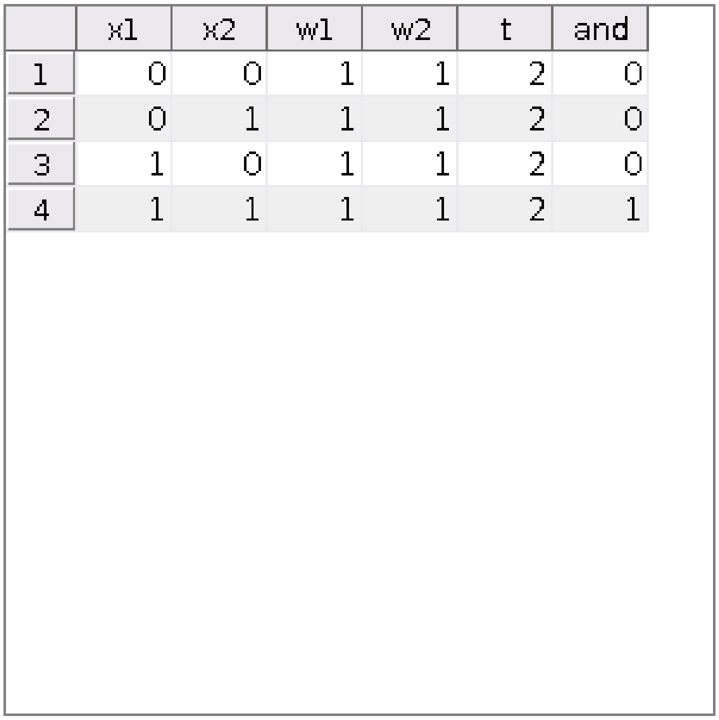
```
% Question 1
```

```
% Part A
% OR Gate

x1 = [0, 0, 1, 1];
x2 = [0, 1, 0, 1];
w1 = [1, 1, 1, 1];
w2 = [1, 1, 1, 1];
t = 1;

or = {'x1' 'x2' 'w1' 'w2' 't' 'or'};
data=[];
for i = 1:length(x1)
    if ( x1(i)*w1(i) + x2(i)*w2(i)) >= t
        datas = [x1(i), x2(i), w1(i), w2(i), t, 1];
    else
        datas = [x1(i), x2(i), w1(i), w2(i), t, 0];
    end
    data = [data; datas];
end

f=figure;
t=uitable(f, 'data', data, 'columnname', or, 'ColumnWidth', {40});
```



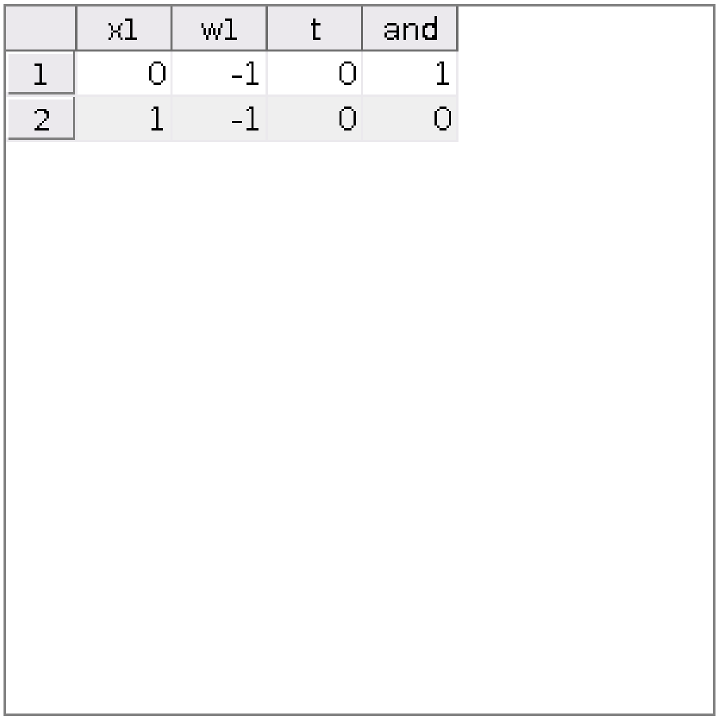
	x1	x2	w1	w2	t	or
1	0	0	1	1	1	0
2	0	1	1	1	1	1
3	1	0	1	1	1	1
4	1	1	1	1	1	1

```
% Part B
% AND Gate

x1 = [0, 0, 1, 1];
x2 = [0, 1, 0, 1];
w1 = [1, 1, 1, 1];
w2 = [1, 1, 1, 1];
t = 2;

and = {'x1' 'x2' 'w1' 'w2' 't' 'and'};
data=[];
for i = 1:length(x1)
    if ( x1(i)*w1(i) + x2(i)*w2(i)) >= t
        datas = [x1(i), x2(i), w1(i), w2(i), t, 1];
    else
        datas = [x1(i), x2(i), w1(i), w2(i), t, 0];
    end
    data = [data; datas];
end

f=figure;
t=uitable(f, 'data', data, 'columnname', and, 'ColumnWidth', {40});
```



	x1	x2	w1	w2	t	and
1	0	0	1	1	2	0
2	0	1	1	1	2	0
3	1	0	1	1	2	0
4	1	1	1	1	2	1

```matlab
% Part B
% NOT Gate

x1 = [0, 1];
w1 = [-1, -1];
t = 0;

not = {'x1' 'w1' 't' 'and'};
data=[];
for i = 1:length(x1)
    if ( x1(i)*w1(i) ) >= t
        datas = [x1(i), w1(i), t, 1];
    else
        datas = [x1(i), w1(i), t, 0];
    end
    data = [data; datas];
end

f=figure;
t=uitable(f, 'data', data, 'columnname', not, 'ColumnWidth', {40});
```



	x1	w1	t	and
1	0	-1	0	1
2	1	-1	0	0

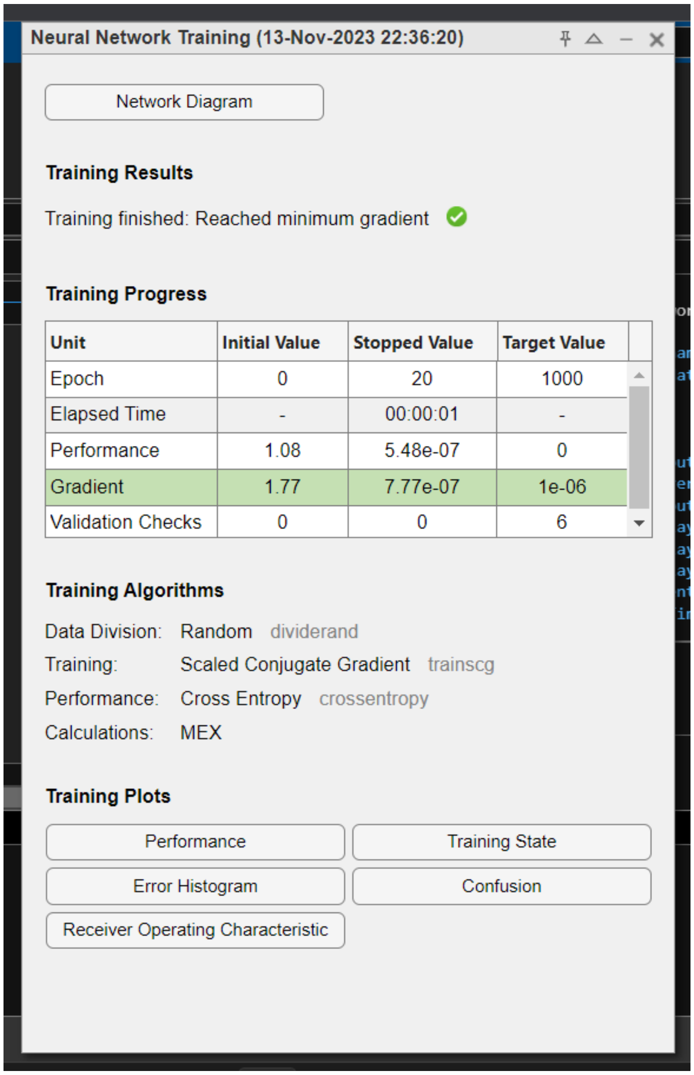
```
% Question 2
```

```
% Part A
% OR Gate

x = [0 0 1 1; 0 1 0 1];
y = [0 1 1 1];

% network architecture
or=patternnet(2);

% train the neural network
nn=train(or, x, y);

% testing
z = [1 1 0 1; 1 0 1 1];
output=round(nn(z))
```

```
output = 1×4
     1     1     1     1
```

```
% Part B
% AND Gate
```

OR GATE



Training Results

Training finished: Reached minimum gradient ✓

Training Progress

Unit	Initial Value	Stopped Value	Target Value
Epoch	0	20	1000
Elapsed Time	-	00:00:01	-
Performance	1.08	5.48e-07	0
Gradient	1.77	7.77e-07	1e-06
Validation Checks	0	0	6

Training Algorithms

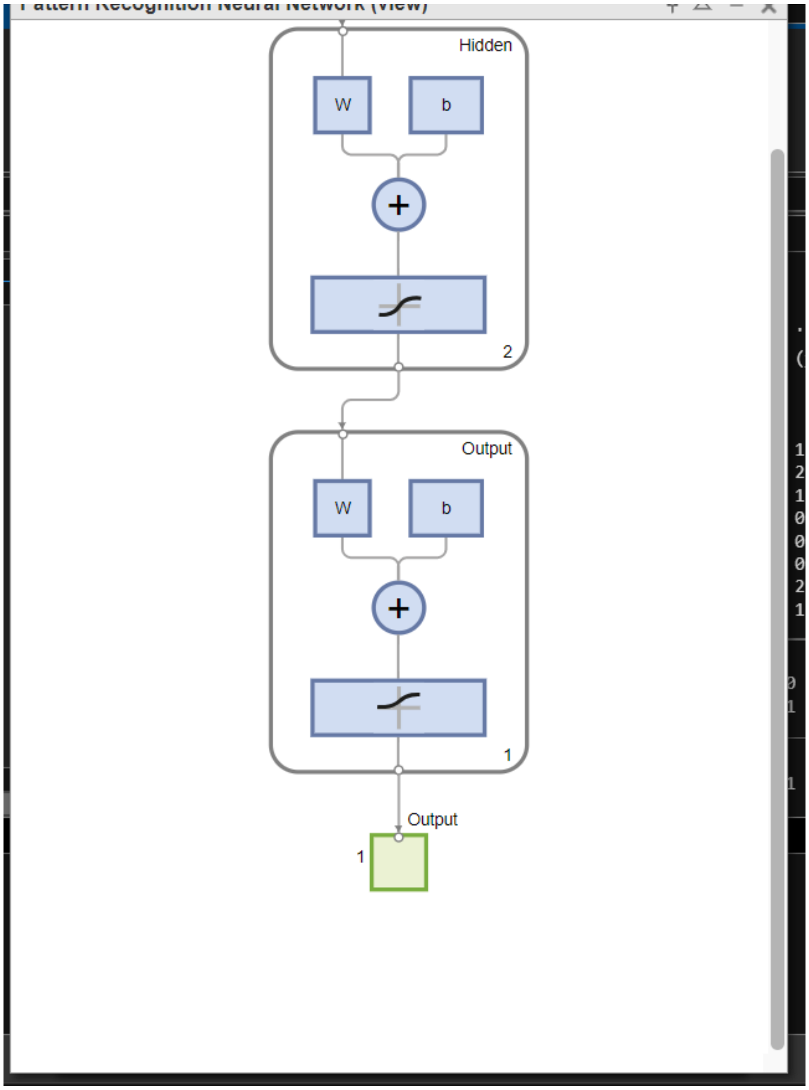
Data Division: Random dividerand

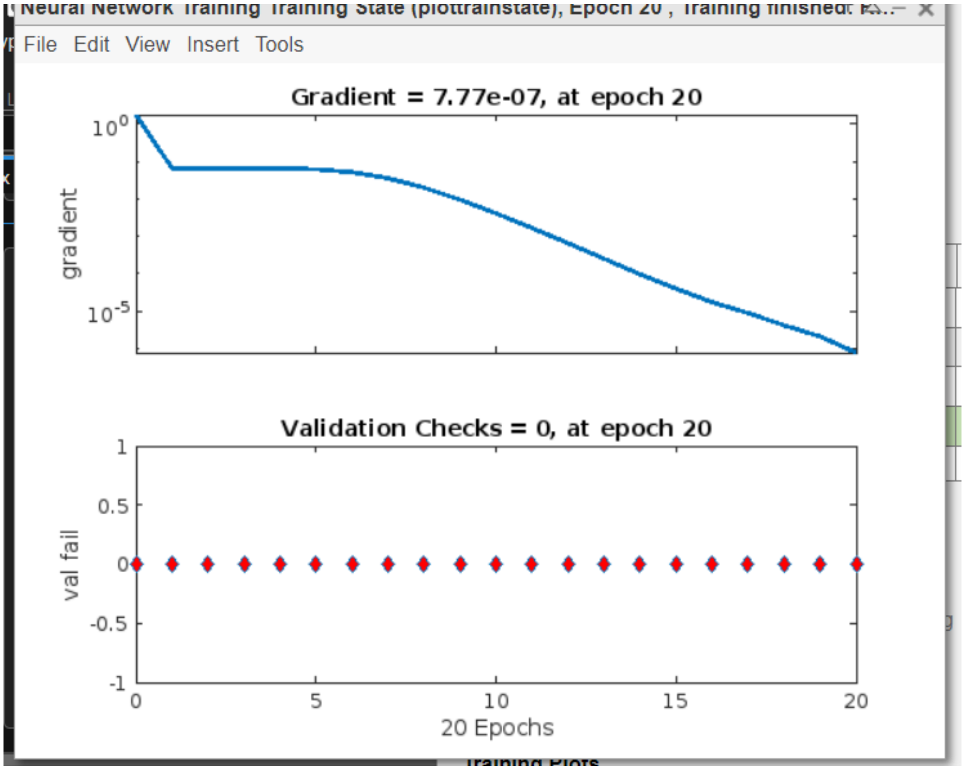
Training: Scaled Conjugate Gradient trainscg

Performance: Cross Entropy crossentropy

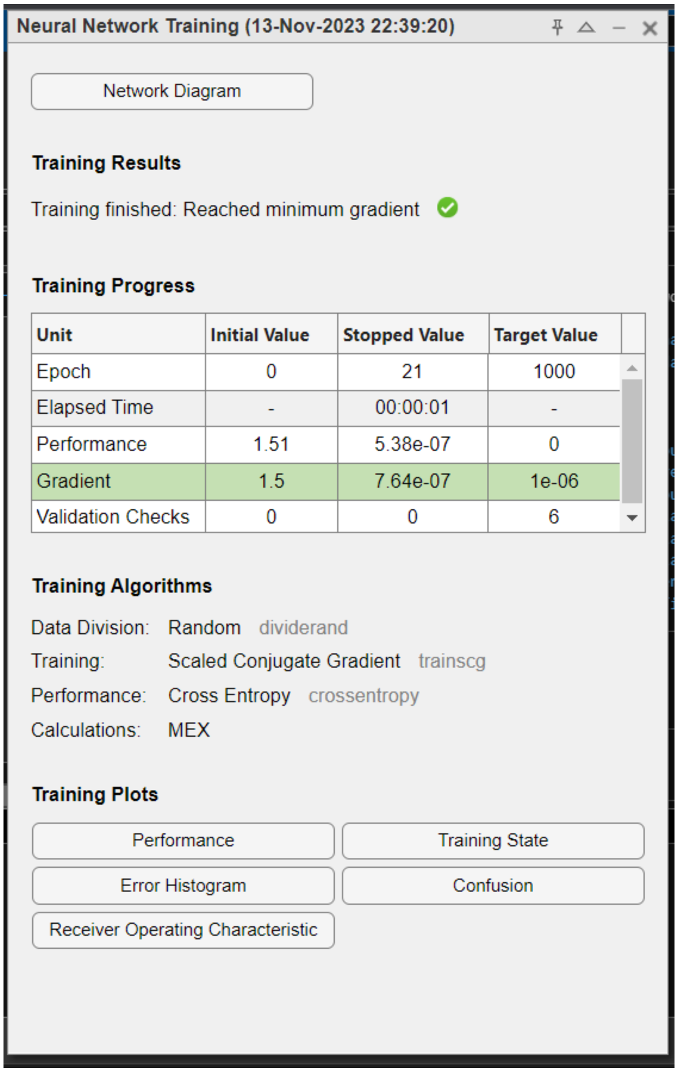
Calculations: MEX

Training Plots





AND GATE



Neural Network Training (13-Nov-2023 22:39:20)

Training Results

Training finished: Reached minimum gradient ✓

Training Progress

Unit	Initial Value	Stopped Value	Target Value
Epoch	0	21	1000
Elapsed Time	-	00:00:01	-
Performance	1.51	5.38e-07	0
Gradient	1.5	7.64e-07	1e-06
Validation Checks	0	0	6

Training Algorithms

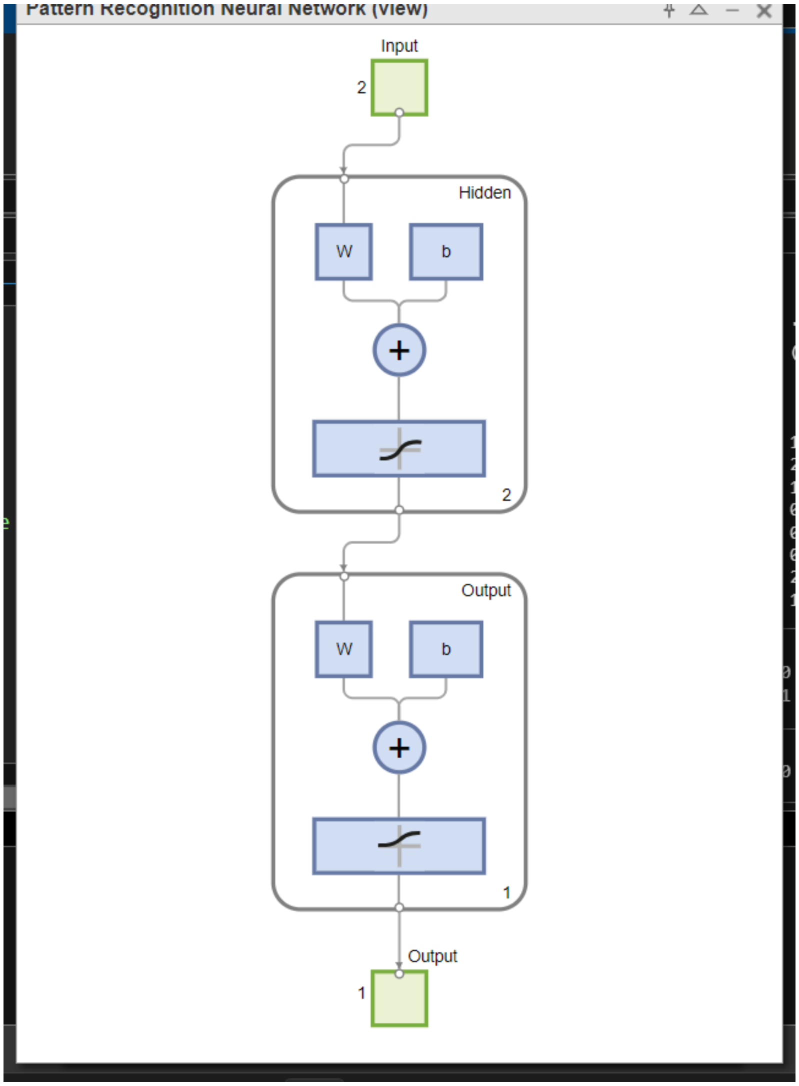
Data Division: Random dividerand

Training: Scaled Conjugate Gradient trainscg

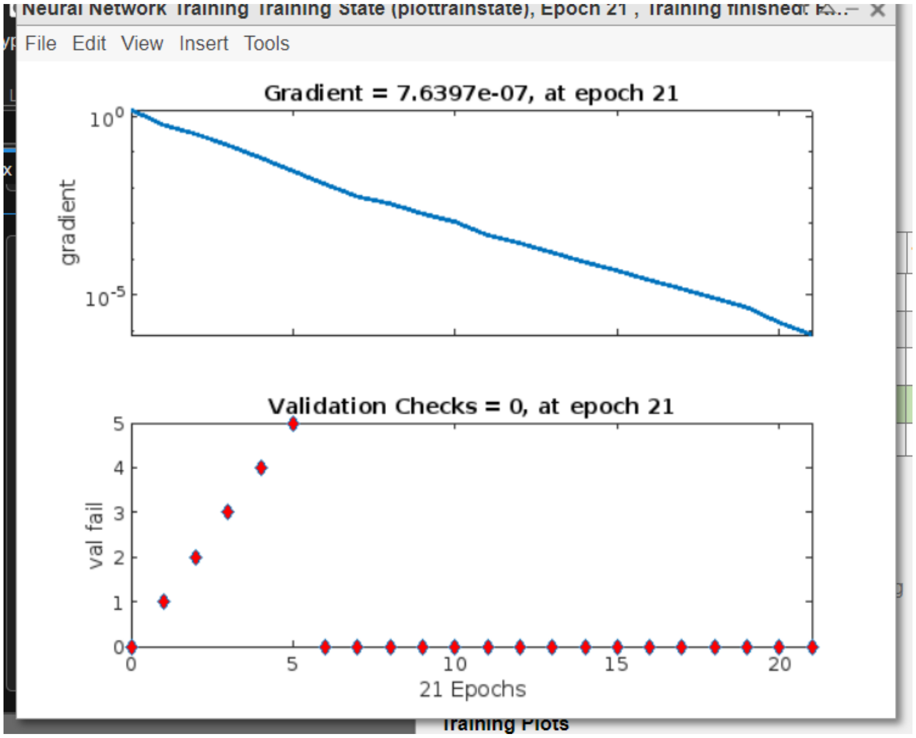
Performance: Cross Entropy crossentropy

Calculations: MEX

Training Plots



Pattern Recognition Neural Network (view)



Neural Network Training Training State (plottrainstate), Epoch 21 , Training finished: Re...

```
x = [0 0 1 1; 0 1 0 1];
y = [0 0 0 1];

% network architecture
and=patternnet(2);

% train the neural network
nn=train(and, x, y);

% testing
z = [1 1 0 1; 1 0 1 1];
output=round(nn(z))
```

```
output = 1×4
     1     0     0     1
```

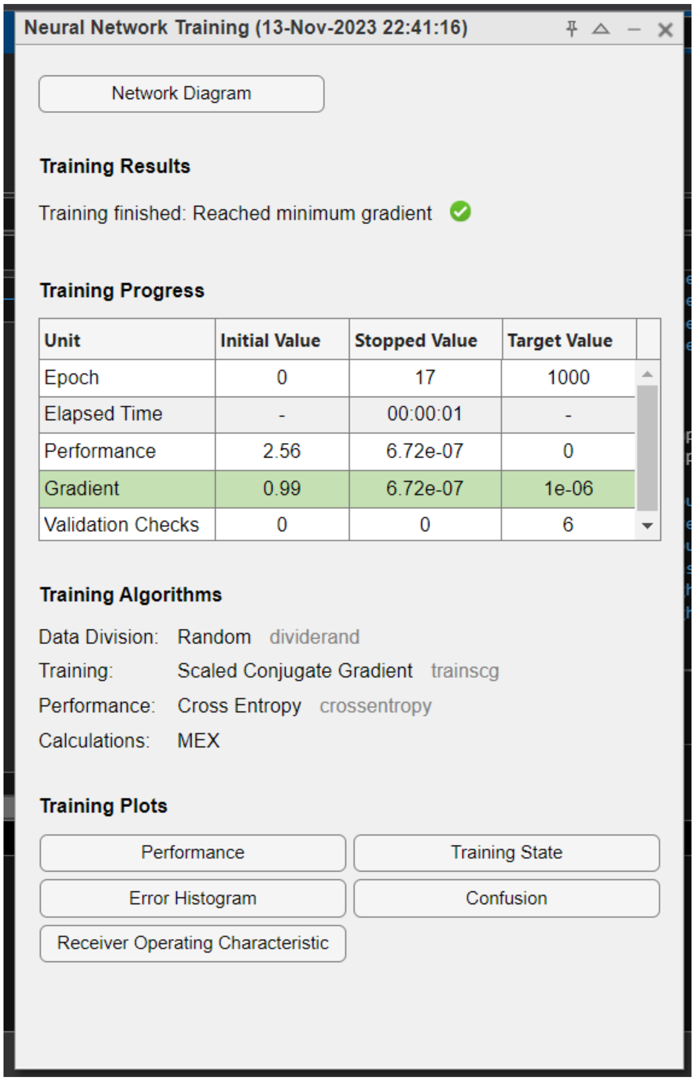
```
% Part C
% NOT Gate

x = [0 1];
y = [1 0];

% network architecture
not=patternnet(1);

% train the neural network
nn=train(not, x, y);

% testing
z = [1 0];
output=round(nn(z))
```

```
output = 1×2
     0     1
```

```
% Question 3
```

```
% Part A
% XOR Gate

x = [0 0 1 1; 0 1 0 1];
y = [0 1 1 0];

% network architecture
xor=feedforwardnet(5);

% train the neural network
nn=train(xor, x, y);
```

NOT GATE



Neural Network Training (13-Nov-2023 22:41:16)

Training Results

Training finished: Reached minimum gradient ✓

Training Progress

Unit	Initial Value	Stopped Value	Target Value
Epoch	0	17	1000
Elapsed Time	-	00:00:01	-
Performance	2.56	6.72e-07	0
Gradient	0.99	6.72e-07	1e-06
Validation Checks	0	0	6

Training Algorithms

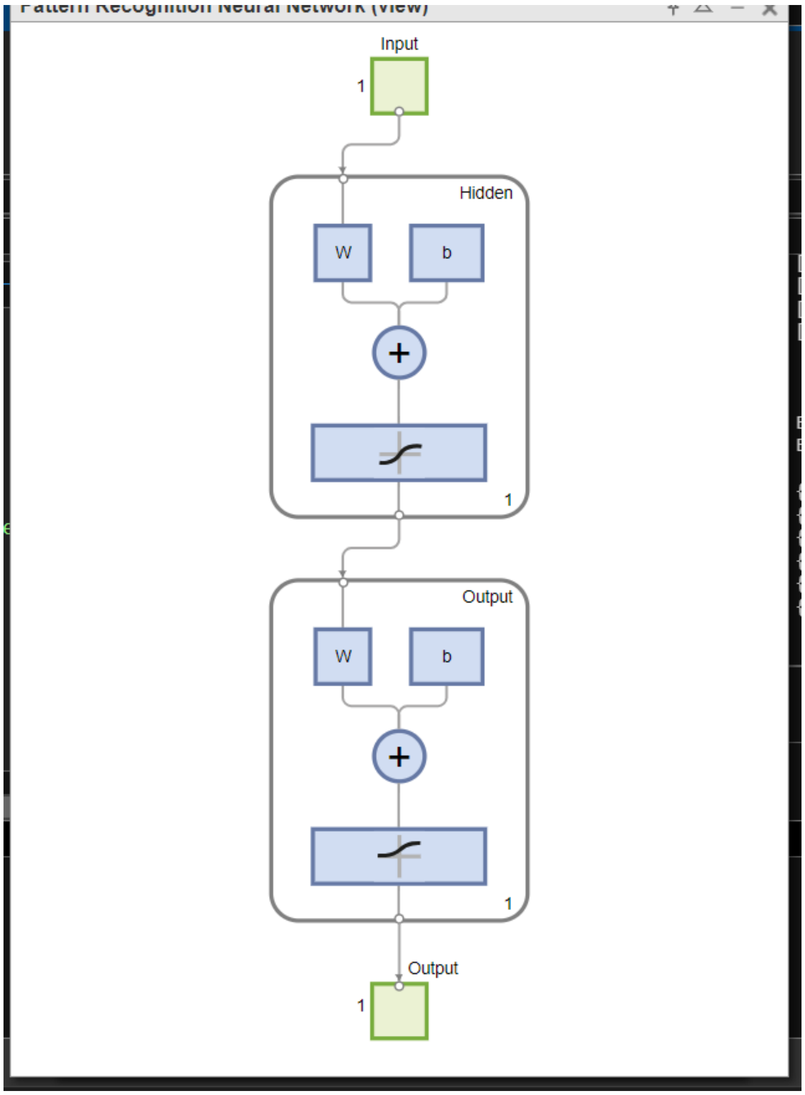
Data Division: Random dividerand

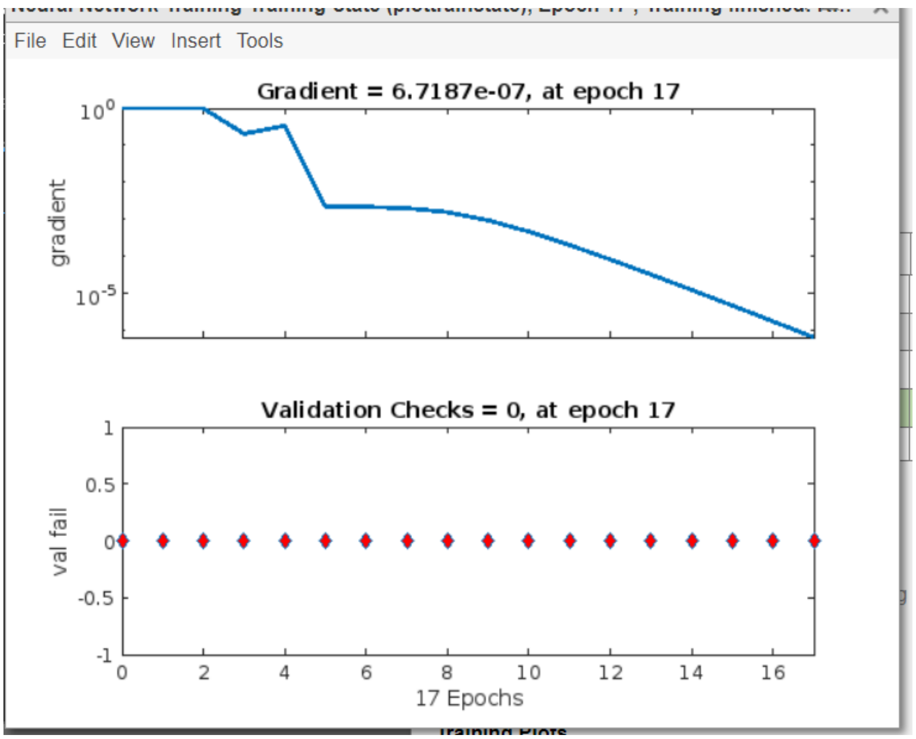
Training: Scaled Conjugate Gradient trainscg

Performance: Cross Entropy crossentropy

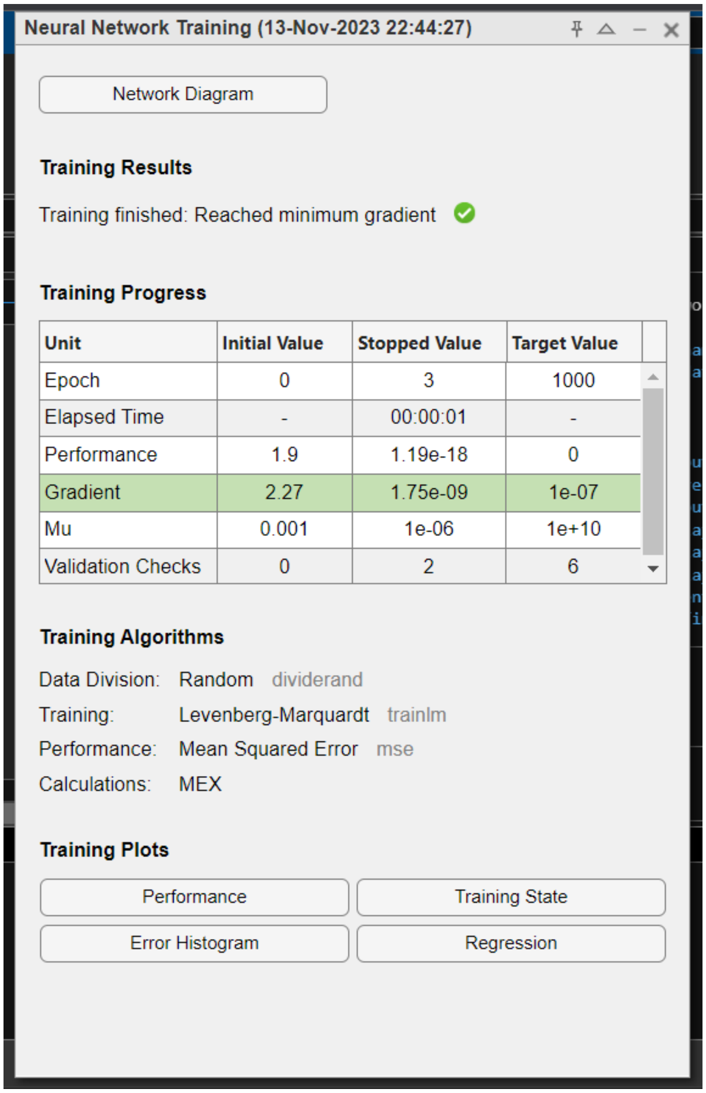
Calculations: MEX

Training Plots





XOR GATE



Neural Network Training (13-Nov-2023 22:44:27)

Training Results

Training finished: Reached minimum gradient ✓

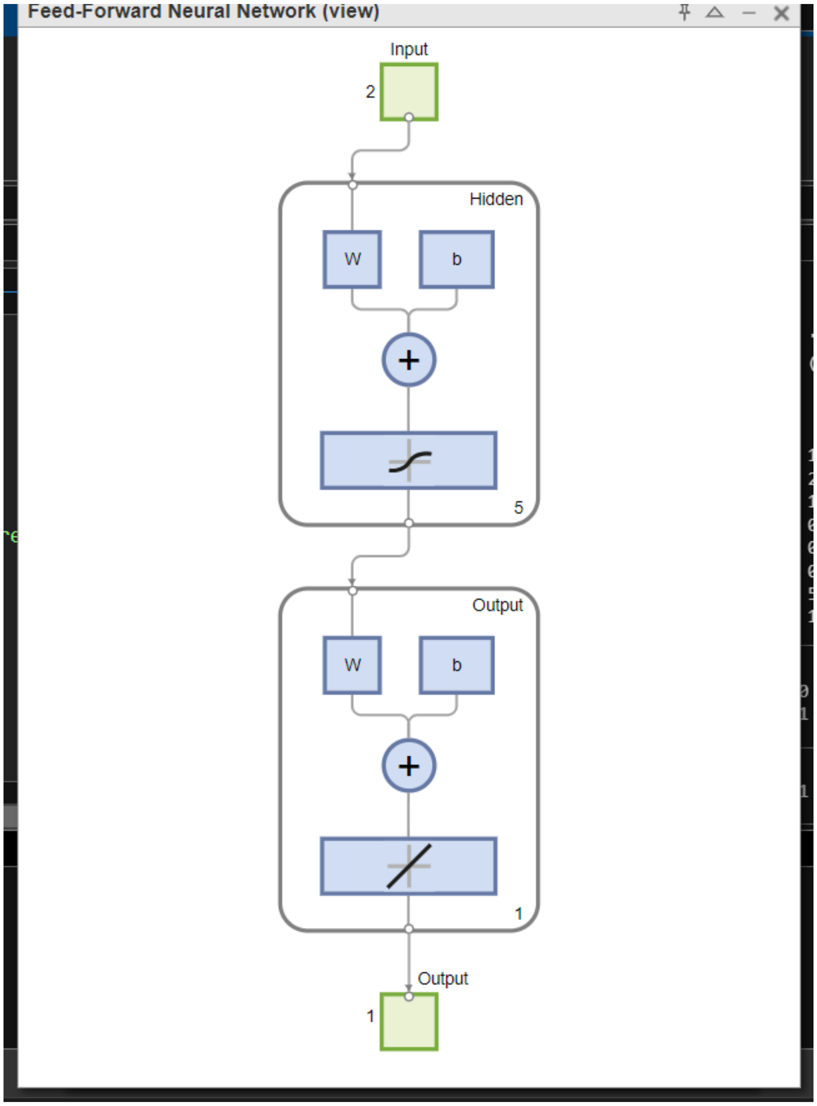
Training Progress

Unit	Initial Value	Stopped Value	Target Value
Epoch	0	3	1000
Elapsed Time	-	00:00:01	-
Performance	1.9	1.19e-18	0
Gradient	2.27	1.75e-09	1e-07
Mu	0.001	1e-06	1e+10
Validation Checks	0	2	6

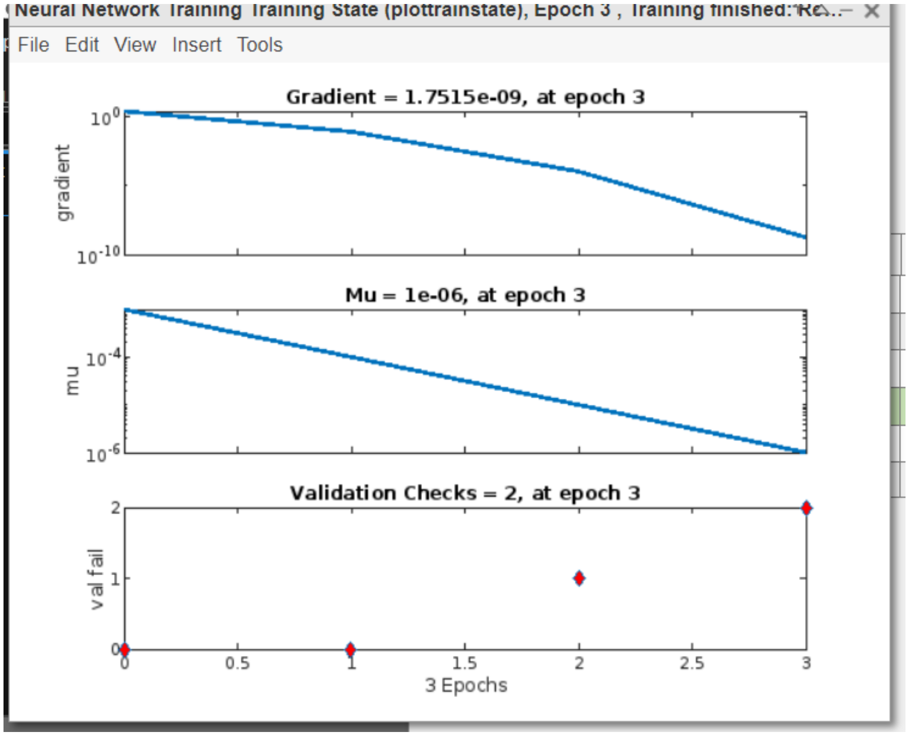
Training Algorithms

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

Training Plots



Feed-Forward Neural Network (view)



Neural Network Training Training State (plottrainstate), Epoch 3 , Training finished: Re...

```

% testing
z = [0 0 1 1; 0 1 1 0];
output=round(nn(z))
```

```
output = 1×4
     0     1     0     1
```

```
% Part B

x = [0 0; 0 1; 1 0; 1 1];
y = [0; 1; 1; 0];

% network architecture
input = 2;
hidden = 5;
output = 1;

% weights and biases
weights_IH=randn(hidden, input);
bias_H=randn(hidden, 1);
weights_HO=randn(output, hidden);
bias_O=randn(output, 1);

learning_rate=0.1
```

```
learning_rate = 0.1000
```

```

% training
for epoch = 1:10000
    % forward pass
    hidden_input=weights_IH*x'+repmat(bias_H, 1, 4);
    hidden_output=sigmoid(hidden_input);
    final_input=weights_HO*hidden_output+bias_O;
    final_output=sigmoid(final_input);
    % backward pass
    error=y'-final_output;
    delta_output=error.*sigmoid_derivative(final_input);

delta_hidden=(weights_HO'*delta_output).*(sigmoid_derivative(hidden_input));
    % update weights and biases
    weights_HO=weights_HO+learning_rate*delta_output*hidden_output';
    bias_output=bias_O+learning_rate*sum(delta_output, 2);
    weights_IH=weights_IH+learning_rate*delta_hidden*x;
    bias_hidden=bias_H+learning_rate*sum(delta_hidden, 2);
end

% testing
```

```
z=round(sigmoid(weights_HO*sigmoid(weights_IH*x'+repmat(bias_H, 1, 4))
+bias_O))
```

```
z = 1×4
     0     1     1     0
```

```
function s = sigmoid(x)
    s=1./(1+exp(-x));
end

function ds = sigmoid_derivative(x)
    s=sigmoid(x);
    ds=s.*(1-s);
end
```