

CHEN4011

Advanced Modelling and Control



Curtin University

Dr. Ranjeet Utikar (RU)



Curtin University
Malaysia

Dr. Jobrun Nandong (JN)

ANN Modelling

Outline

- ANN modelling background
- Basics of ANN model
- ANN model applications
- MATLAB – Neural Net Fitting (NFT) toolbox
- Illustrative example – crystallization
- Summary

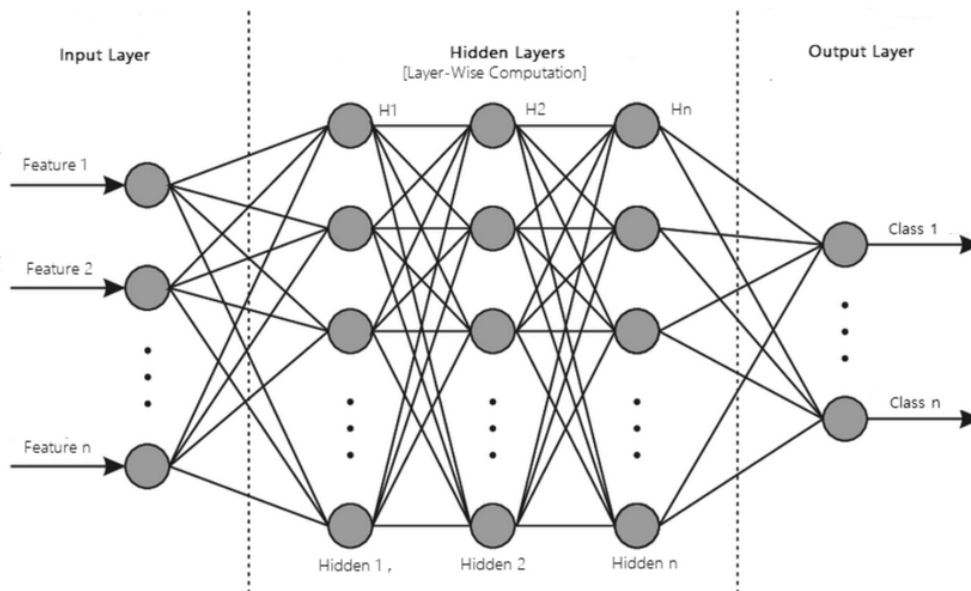
Background

- Modelling is vital to understanding the behaviours of systems, e.g., some hidden dynamics that can affect the production quality.
- Fundamental model based on the principles of conservation and process knowledge e.g., mass transfer, fluid mechanics, etc., might not be practical due to missing information and lack of knowledge.
- Black-box models can be the alternative to fundamental models
- Artificial Neural Network (ANN) model is one of the most popular black-box models derived based on data, i.e., data-driven modelling.
- ANN is a computational model that predicts an output or outputs based on inputs.

Forms of Models

Model Type	Equation type	
	Steady-state model	Dynamic model
Deterministic	Algebraic equations, e.g., steady-state mass balance	ODEs/PDEs, e.g., dynamic mass balance
Stochastic	Algebraic/difference equations	Stochastic ODEs/algebraic difference equations
Lumped Parameter	Algebraic equations	ODEs
Distributed Parameters	Algebraic equations	PDEs
Linear	Linear algebraic equations	Linear ODEs
Nonlinear	Nonlinear algebraic equations	Nonlinear ODEs
Continuous	Algebraic equations	ODEs
Discrete	Difference equations	Difference equations

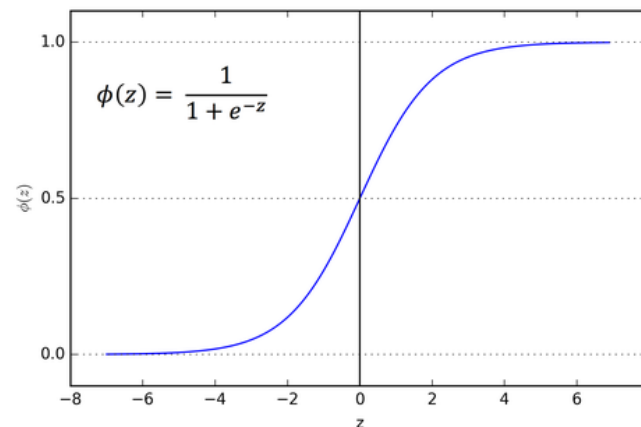
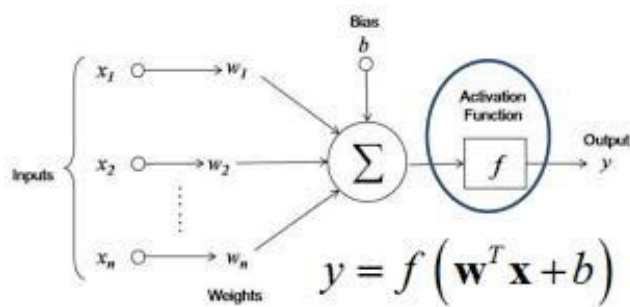
ANN Model basics



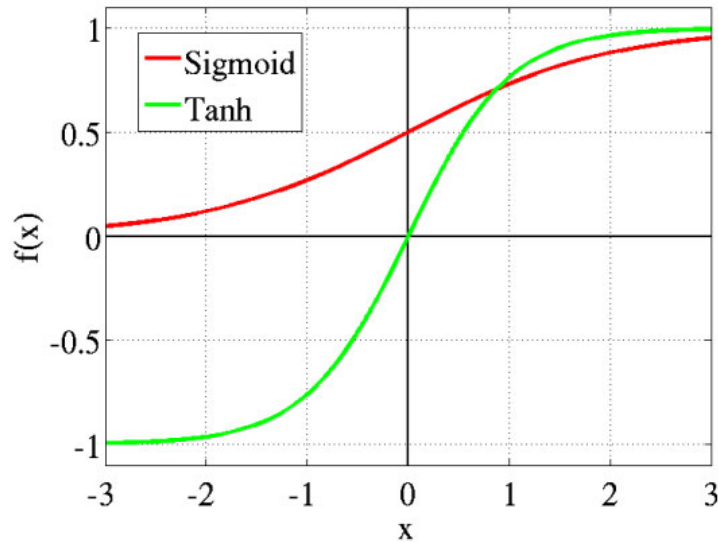
- ANN imitates the brain neurons to enable the computer to learn things
- Consists of 3 types of layers
 - i. Input layer
 - ii. Hidden layer
 - iii. Output layer
- Hidden layers represent the vital component ANN
 - Act as artificial neurons
 - Receives a set of inputs and produce an output via activation function
 - Relatively simple data requires only 1 or 2 hidden layers
 - More complex data requires more number of hidden layers

ANN Hidden Layers – Activation function

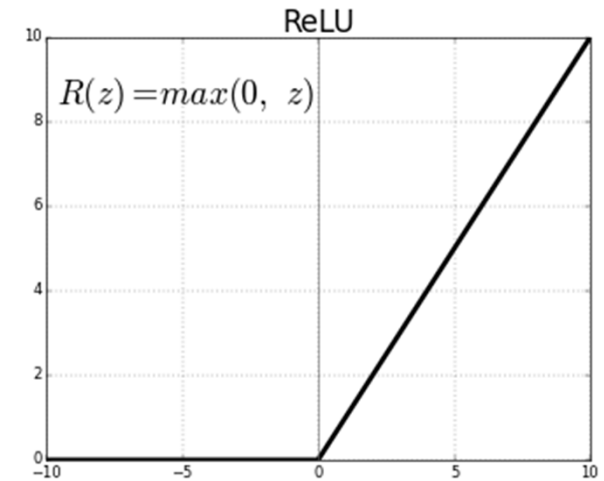
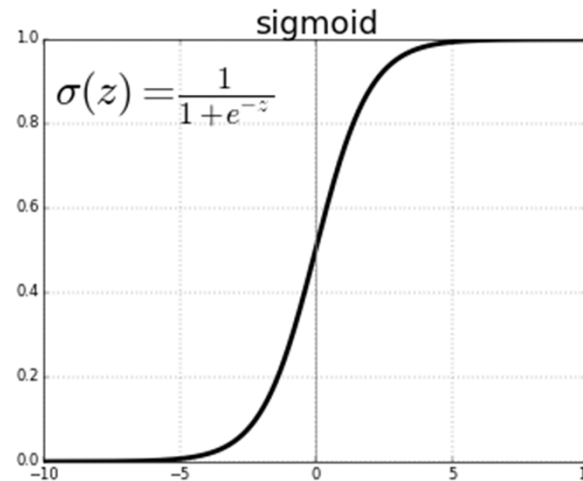
- Hidden layers are used to perform nonlinear transformations of inputs to the network
- Hidden layers required activation function



- **Sigmoid activation function**
- Exists between 0 and 1
- The function differentiable and monotonic



- **Hyperbolic tangent Activation Function**
- Range is between -1 and 1
- **Advantage**
 - Negative inputs will be mapped strongly negative and the zero inputs will be mapped near zero in the tanh graph.



- **ReLU (Rectified Linear Unit) Activation Function**
- $f(z)$ is zero when z is less than zero and $f(z)$ is equal to z when z is above or equal to zero
- Range is between 0 and infinity
- Function and its derivative **both are monotonic**
- The issue is that all the negative values become zero immediately
- Thus, decreases the ability of the model to fit or train from the data properly

ANN applications

■ Artificial System Construction

- The engineering goal of building efficient systems for real applications
- Make machines more powerful and intelligent, relieve humans of tedious tasks and may even improve upon human performance

■ Brain Modelling

- The biological of constructing models of how real brains work
- Potentially can help us understand the nature of perception, actions, learning and memory
- Help formulate medical solutions to brain damaged patients

Advantages and Disadvantages of Neural Network

■ Advantages

- No prior knowledge is required about the system to be modelled.
- Fast development for complex system compared with fundamental modelling approach.

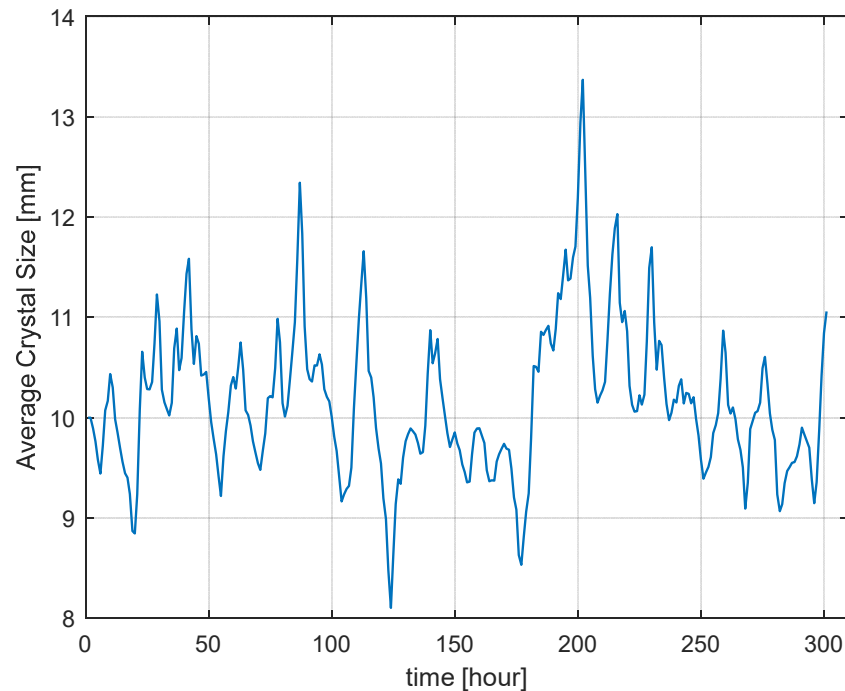
■ Disadvantages

- Requires a lot of data to conduct Training, Validation and Testing. Insufficient data can lead to unreliable neural network model.
- Cannot really be used for a wide range of operating conditions. In particular can be used for extrapolation beyond the ranges of conditions of data used to construct the model.

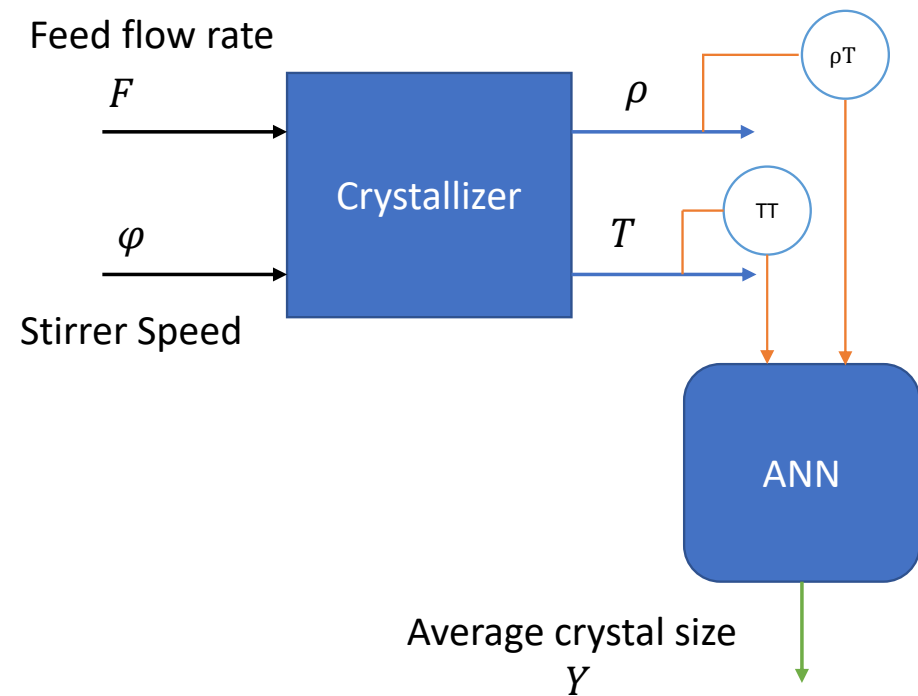
Illustrative Example - Crystallization

- Average crystal size is one of the important parameters to be controlled in a crystallization process
- Assume the average crystal size are affected by two measured variables:
 - i. Mother liquor temperature T
 - ii. Mother liquor density ρ
- Note that average crystal size cannot be measured directly
- Build a model to predict the size using T and ρ measurements

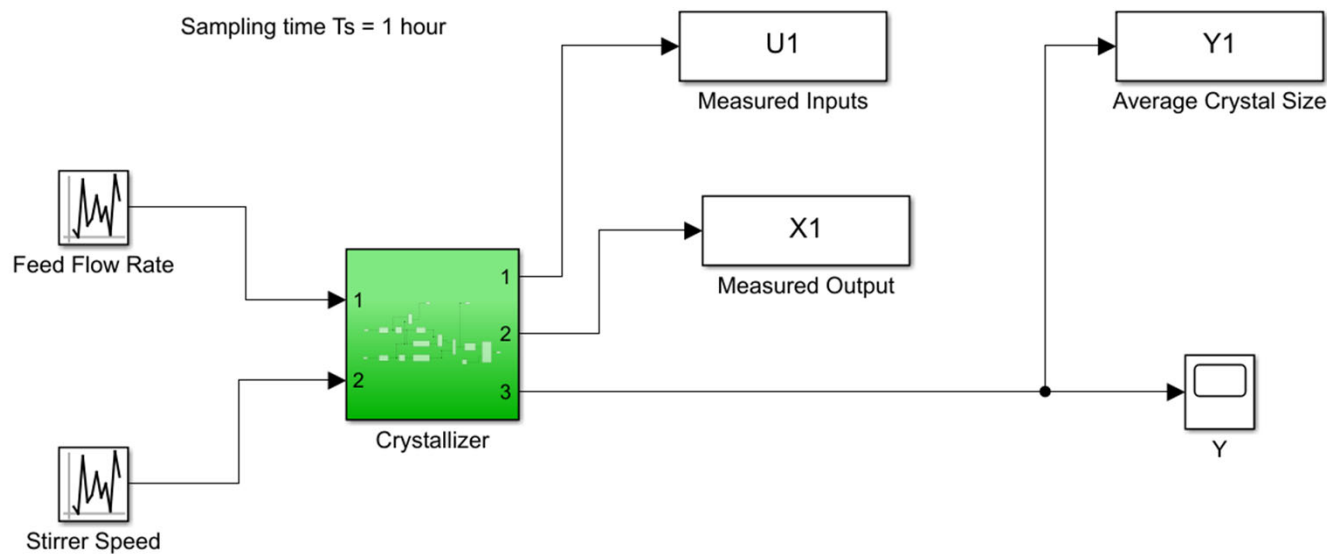
Crystallization – Average Crystal Size ANN prediction



- Average crystal size based on laboratory analysis
- 301 sample points



Crystallization – Average Crystal Size ANN prediction – Simulink model

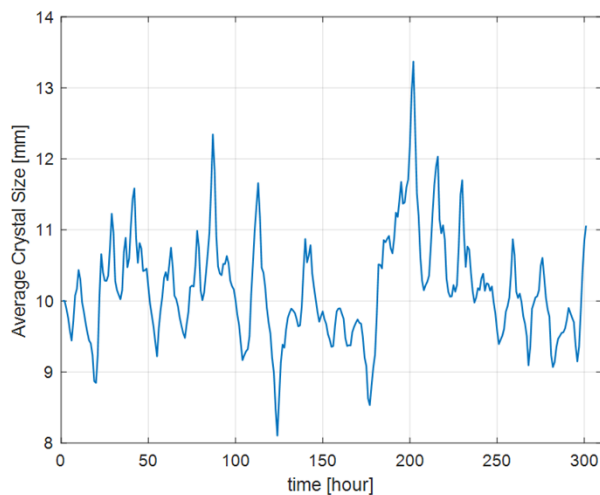


✓ Simulink model for data generation

Assumptions:

- Sampling period $T_s = 1$ hour
- Inputs and outputs are measured
- Inputs change according to uniform random numbers
 - Feed flow rate change is ± 1.2 every 1 hour
 - Stirrer speed change is ± 1.5 every 1 hour

Crystallization – ANN modeling – Neural Net Fitting toolbox



To develop ANN model to predict the average crystal size
Dataset consists of 301 samples

Neural Fitting (nftool)

Welcome to the Neural Network Fitting app.

Solve an input-output fitting problem with a two-layer feed-forward neural network.

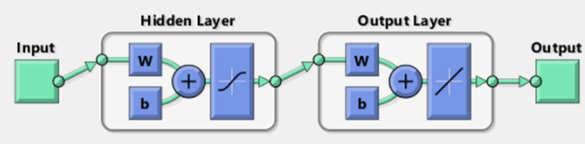
Introduction

In fitting problems, you want a neural network to map between a data set of numeric inputs and a set of numeric targets.

Examples of this type of problem include estimating engine emission levels based on measurements of fuel consumption and speed (*engine_dataset*) or predicting a patient's bodyfat level based on body measurements (*bodyfat_dataset*).

The Neural Fitting app will help you select data, create and train a network, and evaluate its performance using mean square error and regression analysis.

Neural Network



A two-layer feed-forward network with sigmoid hidden neurons and linear output neurons (*fitnet*), can fit multi-dimensional mapping problems arbitrarily well, given consistent data and enough neurons in its hidden layer.

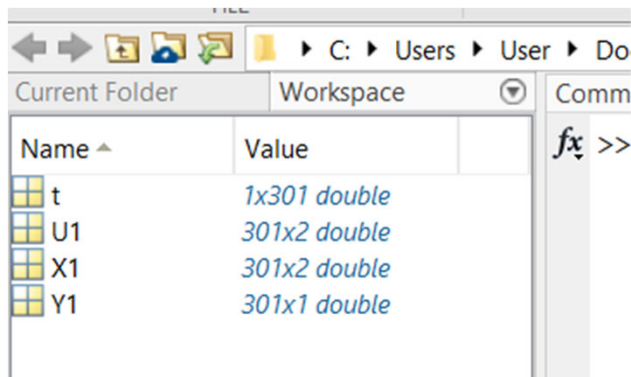
The network will be trained with Levenberg-Marquardt backpropagation algorithm (*trainlm*), unless there is not enough memory, in which case scaled conjugate gradient backpropagation (*trainscg*) will be used.

To continue, click [Next].

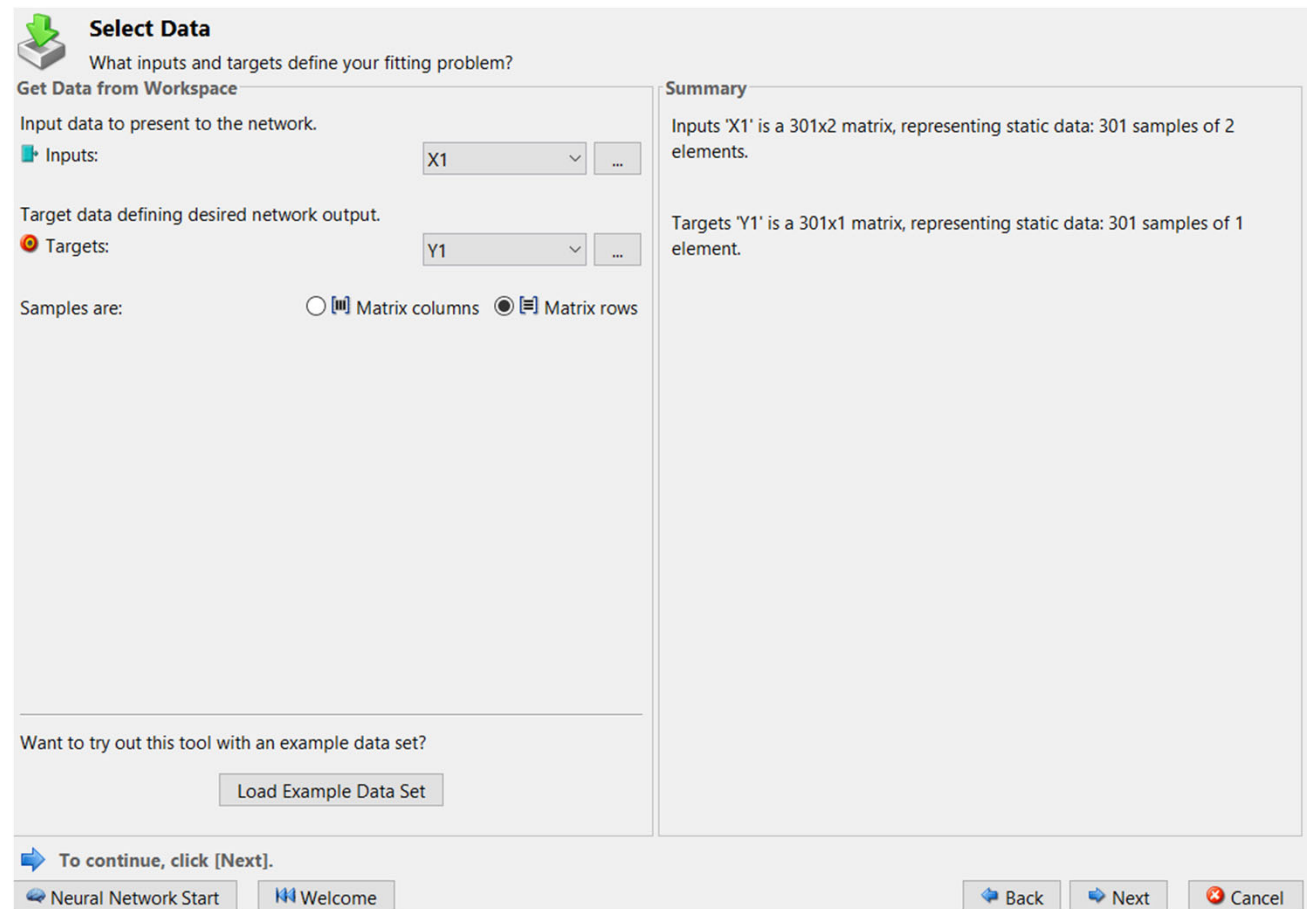
Neural Network Start Welcome Back Next Cancel

Crystallization – Neural Net Fitting

- Upload the dataset to toolbox
- Make sure the dataset already in the MATLAB workspace
 - $X_1 = [\rho, T]$ (301×2 matrix of measured outputs)
 - $Y = ACZ$ (301×1 matrix of average crystal size)



Name	Value
t	1x301 double
U1	301x2 double
X1	301x2 double
Y1	301x1 double



Select Data
What inputs and targets define your fitting problem?


Get Data from Workspace
Input data to present to the network.
Inputs: X1
Target data defining desired network output.
Targets: Y1
Samples are: ☐ Matrix columns ☒ Matrix rows

Summary
Inputs 'X1' is a 301x2 matrix, representing static data: 301 samples of 2 elements.
Targets 'Y1' is a 301x1 matrix, representing static data: 301 samples of 1 element.

Want to try out this tool with an example data set?
[Load Example Data Set](#)

To continue, click [Next].
[Neural Network Start](#) [Welcome](#) [Back](#) [Next](#) [Cancel](#)

Crystallization – Neural Net Fitting (NNF)



Validation and Test Data

Set aside some samples for validation and testing.

Select Percentages

Randomly divide up the 301 samples:

Training:	70%	211 samples
Validation:	15% ▾	45 samples
Testing:	15% ▾	45 samples

Explanation

Three Kinds of Samples:

Training:
These are presented to the network during training, and the network is adjusted according to its error.

Validation:
These are used to measure network generalization, and to halt training when generalization stops improving.

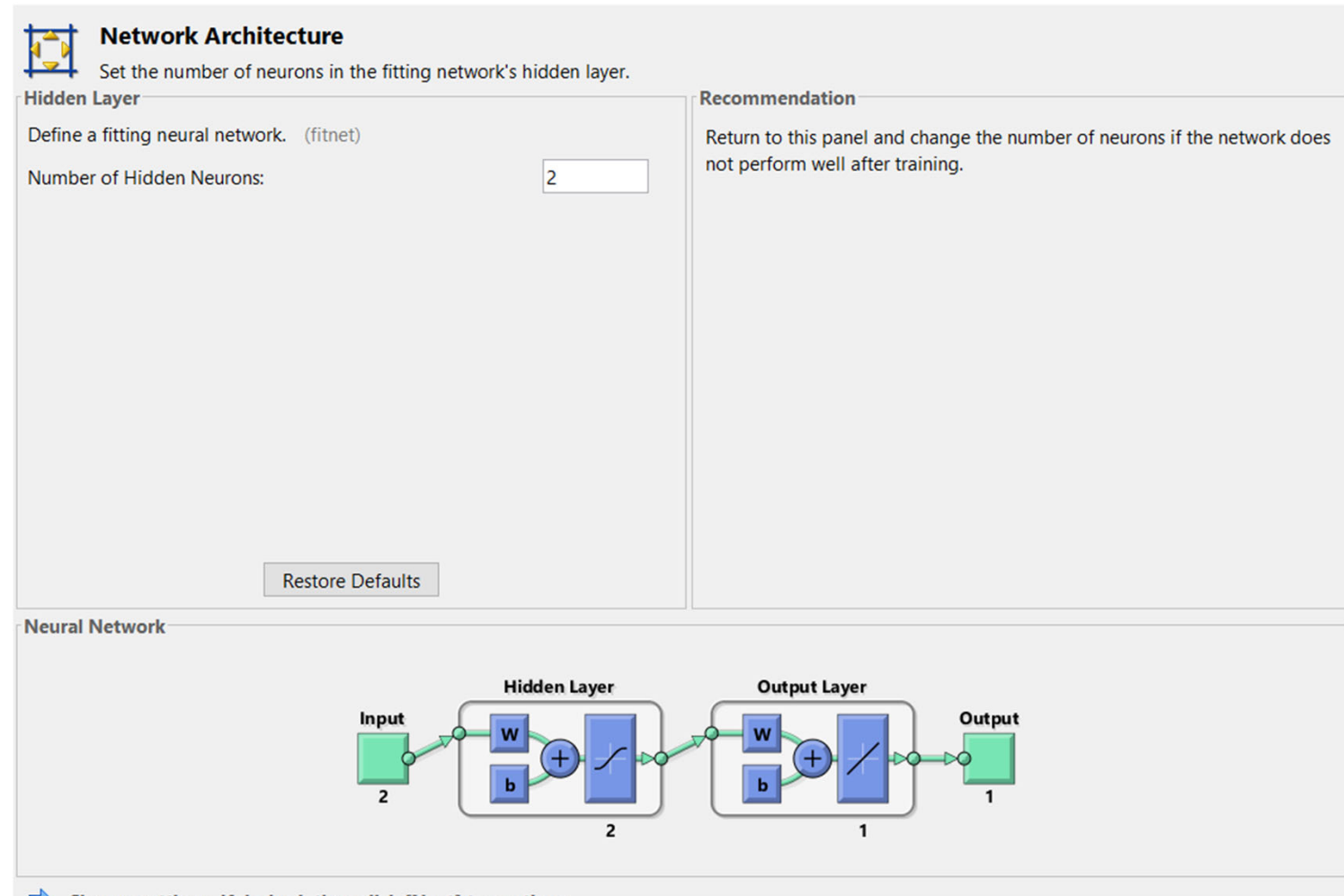
Testing:
These have no effect on training and so provide an independent measure of network performance during and after training.

- Choose the percentages of the dataset uploaded for Training, Validation and Testing
- MATLAB NNT toolbox default selections are **70%, 15% and 15%** for Training, Validation and Testing
- Note the percentages of **data splitting** often affect the **ANN model predictive accuracy**


Crystallization – NNF – Number of Hidden Layer

Note:

- Number of hidden layers affect the ANN model predictive capability
- Try a smaller number first, e.g., 2 hidden layer



Crystallization – NNF – Training Algorithm

**Train Network**
Train the network to fit the inputs and targets.

Train Network
Choose a training algorithm:

Levenberg-Marquardt

This algorithm typically requires more memory but less time. Training automatically stops when generalization stops improving, as indicated by an increase in the mean square error of the validation samples.

Train using Levenberg-Marquardt. (trainlm)

Retrain

Results

	Samples	MSE	R
Training:	211	1.22546e-4	9.99899e-1
Validation:	45	1.40221e-4	9.99834e-1
Testing:	45	2.74736e-4	9.99694e-1

Plot Fit

Plot Error Histogram

Plot Regression

Notes

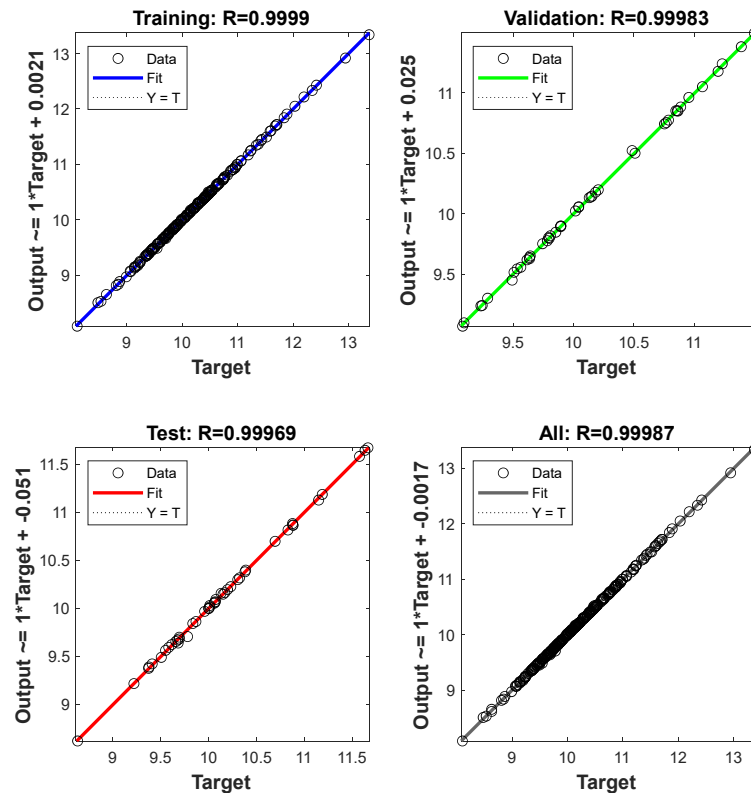
Training multiple times will generate different results due to different initial conditions and sampling.

Mean Squared Error is the average squared difference between outputs and targets. Lower values are better. Zero means no error.

Regression R Values measure the correlation between outputs and targets. An R value of 1 means a close relationship, 0 a random relationship.

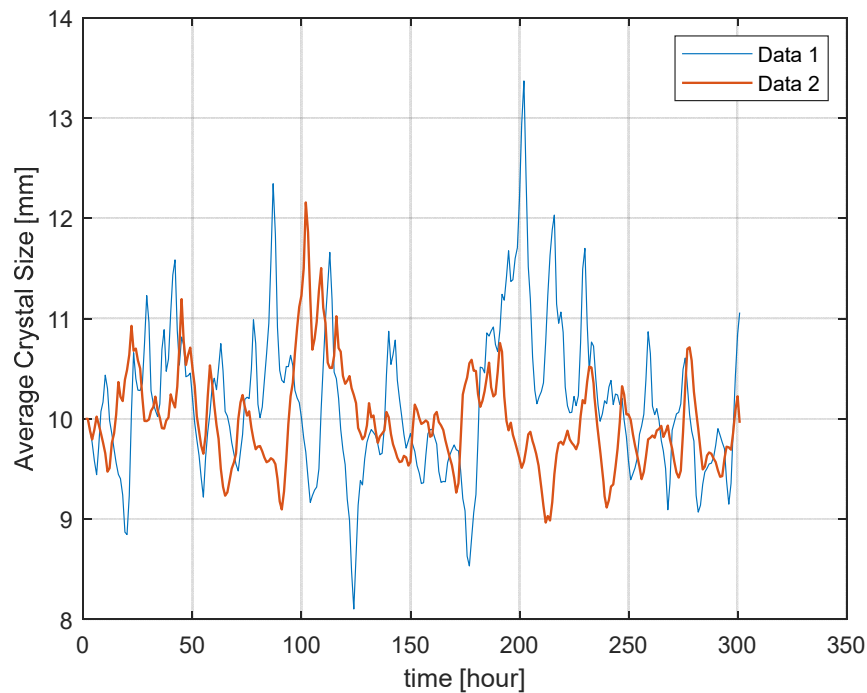
- 3 training algorithms available
- Different algorithms may give different modelling accuracies, e.g., MSE values
- Plot the regression to compare the model predictions versus real data
- R^2 or coefficient of determination is close to unity => ANN model fit the data well

Crystallization – ANN – regression plot



- The plot shows the ANN model with 2 hidden layers can fit the data well
- Model fitness ($R^2 > 0.95$) for both training, validation and test
- We can try the ANN model predictive capability using other sets of data
- We generate a second set of crystallization data with the following assumptions;
 - Feed flow rate change ± 1.2 every 0.5 hours
 - Stirrer speed change ± 1.5 every 0.5 hours


Crystallization – ANN Model – Further Testing

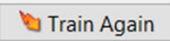


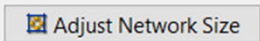
- Data 1 was used to train the ANN model
- Data 2 was generated by same inputs but with different sampling period of their changes applied to the crystallizer
- Can the ANN model developed using Data 1 predict the value in Data 2?

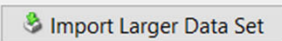
Crystallization – NNF – Testing using Data 2

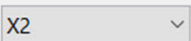
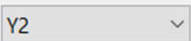
- Click 'Next'
- Upload the second dataset X_2 and Y_2
- Make sure the data already in the workspace
- Click on 'Test Network'
- The coefficient of determination $R^2 > 0.95$
- Thus, the ANN model with two hidden layers can still predict the Data 2 well enough





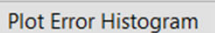

**Evaluate Network**
Optionally test network on more data, then decide if network performance is good enough.

Iterate for improved performance
Try training again if a first try did not generate good results or you require marginal improvement.


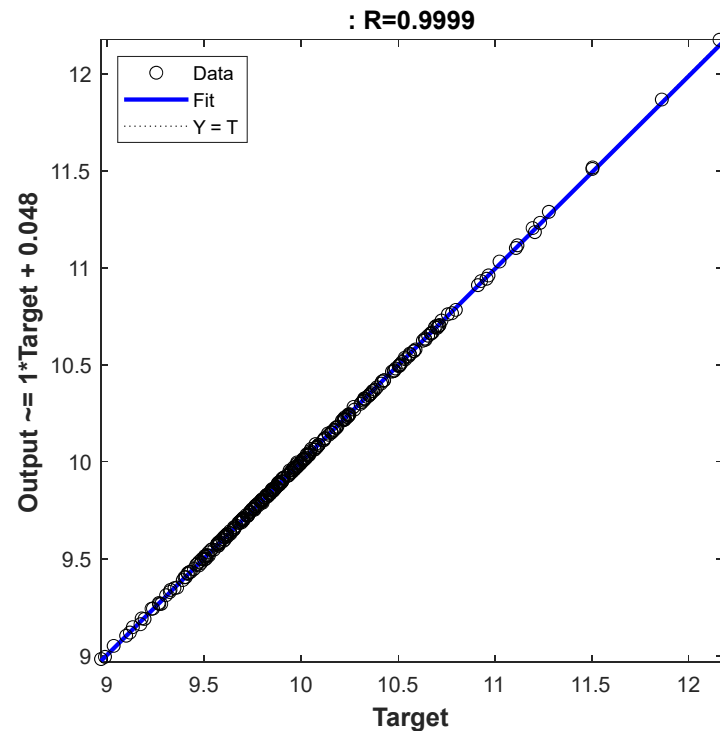
Increase network size if retraining did not help.


Not working? You may need to use a larger data set.


Optionally perform additional tests
 Inputs: X2
 Targets: Y2
Samples are: ☐ Matrix columns ☒ Matrix rows
Inputs 'X2' is a 301x2 matrix, representing static data: 301 samples of 2 elements.
Targets 'Y2' is a 301x1 matrix, representing static data: 301 samples of 1 element.

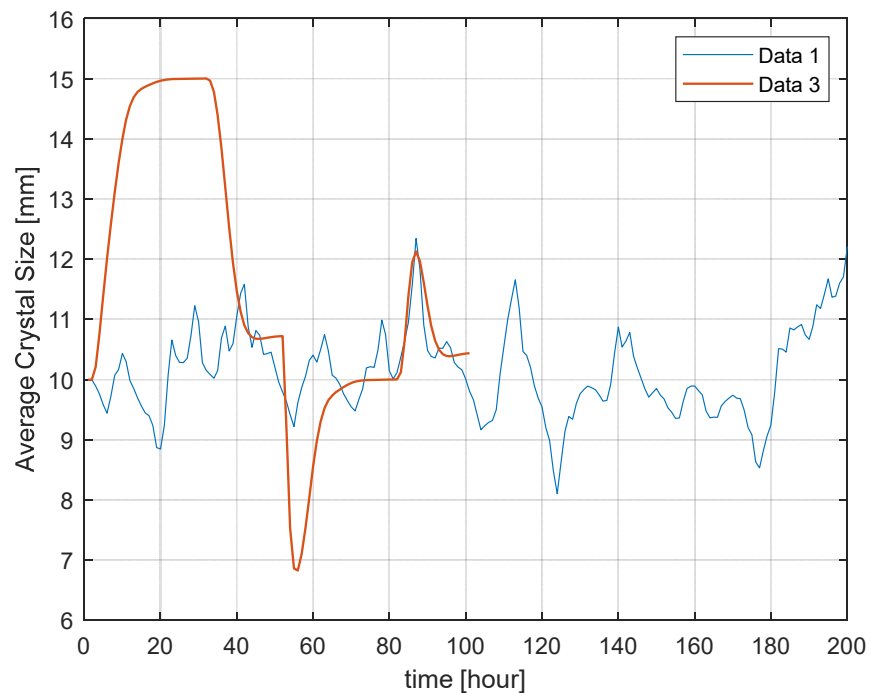

 MSE 5.19962e-5
 R 9.99899e-1
 


Crystallization – ANN Model – regression plot of further testing using Data 2




- The regression plot shows the ANN model can predict Data 2 well
- Note that, the shape of input changes can often affect the prediction capability of the ANN model
- Previously, the ANN model was trained using data generated assuming the input changes are like “uniform random number”
- Let’s generate Data 3 assuming step changes of the inputs

Crystallization – ANN Model – further Testing using Data 3



- Data 3 was generated by step changes in the inputs: feed flow rate and stirrer speed
- Upload X_3 and Y_3 to test the model network

Crystallization – ANN model – further testing using Data 3

**Evaluate Network**
Optionally test network on more data, then decide if network performance is good enough.

Iterate for improved performance
Try training again if a first try did not generate good results or you require marginal improvement.

Train Again

Increase network size if retraining did not help.

Adjust Network Size

Not working? You may need to use a larger data set.

Import Larger Data Set

Optionally perform additional tests

Inputs: X3

Targets: Y3

Samples are: ☐ Matrix columns ☒ Matrix rows

Inputs 'X3' is a 101x2 matrix, representing static data: 101 samples of 2 elements.

Targets 'Y3' is a 101x1 matrix, representing static data: 101 samples of 1 element.

Test Network

☒ MSE

5.77986e-0

☒ R

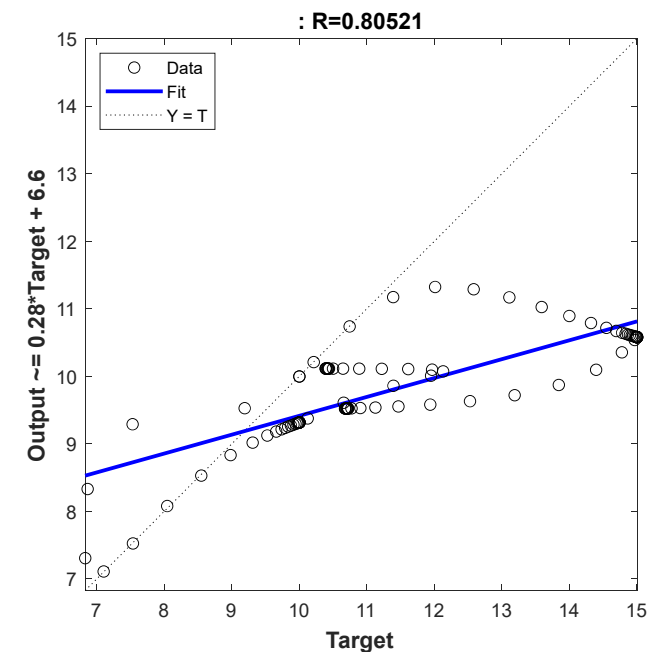
8.05214e-1

Plot Fit

Plot Error Histogram

Plot Regression

- Notice that $R^2 < 0.95$
- The model predictive capability is lower for this Data 3 type



Crystallization – ANN model – increase number of hidden layers

Network Architecture

Set the number of neurons in the fitting network's hidden layer.

Hidden Layer

Define a fitting neural network. (fitnet)

Number of Hidden Neurons:

Recommendation

Return to this panel and change the number of neurons if the network does not perform well after training.

Restore Defaults

Neural Network

The diagram illustrates a neural network architecture with three layers: an input layer with 2 neurons, a hidden layer with 5 neurons, and an output layer with 1 neuron. Each layer is represented by a box containing 'W' (weights) and 'b' (biases), followed by an addition sign and an activation function symbol. The input layer is connected to the hidden layer, which is connected to the output layer. The final output is shown as a single neuron.

Train Network

Train the network to fit the inputs and targets.

Choose a training algorithm:

Levenberg-Marquardt

This algorithm typically requires more memory but less time. Training automatically stops when generalization stops improving, as indicated by an increase in the mean square error of the validation samples.

Train using Levenberg-Marquardt. (trainlm)

Retrain

Results

	Samples	MSE	R
Training:	211	1.31017e-8	9.99999e-1
Validation:	45	6.94961e-8	9.99999e-1
Testing:	45	7.71859e-9	9.99999e-1

Plot Fit Plot Error Histogram Plot Regression

Notes

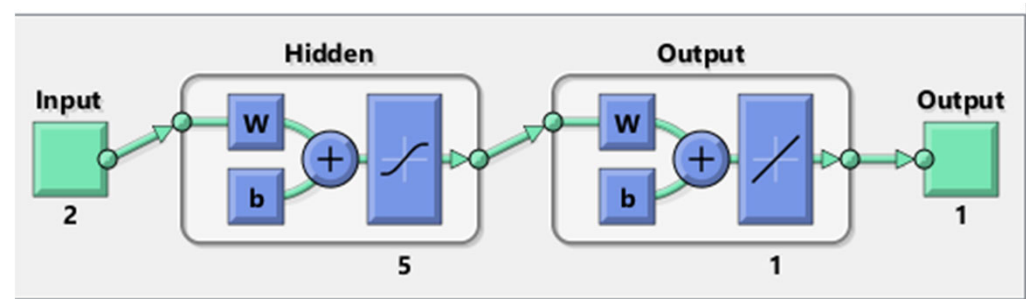
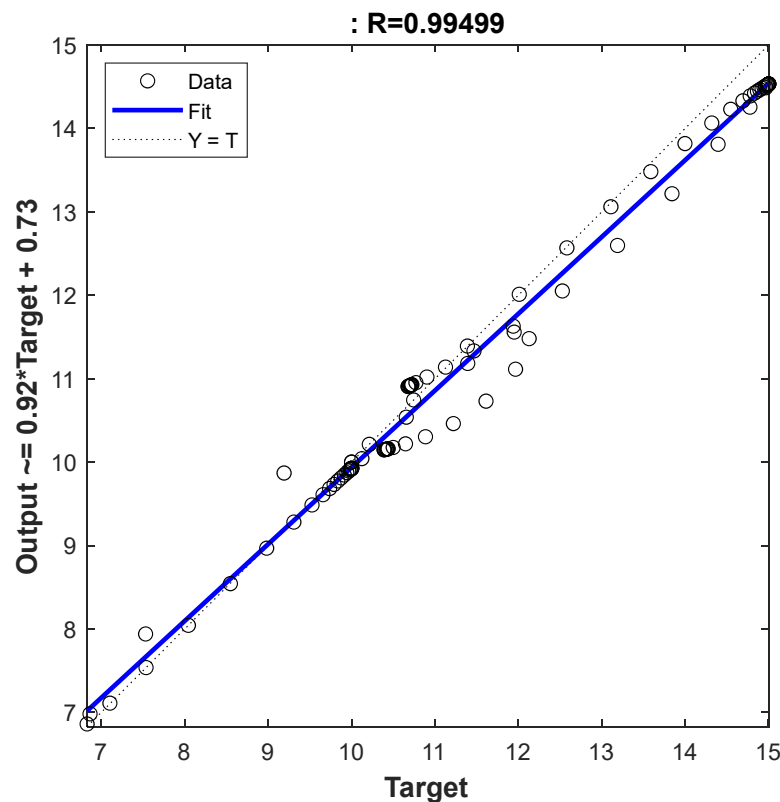
Training multiple times will generate different results due to different initial conditions and sampling.

Mean Squared Error is the average squared difference between outputs and targets. Lower values are better. Zero means no error.

Regression R Values measure the correlation between outputs and targets. An R value of 1 means a close relationship, 0 a random relationship.


- Try the new ANN model with 5 hidden layers
- Can it predict the Data 3 more accurately now?


Crystallization – ANN model – five hidden layers





- The model predictive accuracy looks satisfactory for the Data 3
- We can try to improve it further by adjusting the number of hidden layer and retraining


Crystallization -Generate Deployable ANN Model

 **Deploy Solution**
Generate deployable versions of your trained neural network.

Application Deployment
Prepare neural network for deployment with MATLAB Compiler and Builder tools.
Generate a MATLAB function with matrix and cell array argument support: (genFunction)  MATLAB Function

Code Generation
Prepare neural network for deployment with MATLAB Coder tools.
Generate a MATLAB function with matrix-only arguments (no cell array support): (genFunction)  MATLAB Matrix-Only Function

Simulink Deployment
Simulate neural network in Simulink or deploy with Simulink Coder tools.
Generate a Simulink diagram: (gensim)  Simulink Diagram

Graphics
Generate a graphical diagram of the neural network: (network/view)  Neural Network Diagram

- We can generate deployable versions of trained ANN:
 1. MATLAB function
 2. MATLAB code using MATLAB coder tools
 3. Simulink Deployment

Crystallization – ANN model – generate MATLAB function

- MATLAB function generation
- Save and rename the generated MATLAB function
- Use in Command Window

```
myANN.m x +
1 function [Y,Xf,Af] = myANN(X)
2 %MYNEURALNETWORKFUNCTION neural network simulation function.
3 %
4 % Auto-generated by MATLAB, 26-Sep-2022 19:04:57.
5 %
6 % [Y] = myNeuralNetworkFunction(X,~,~) takes these arguments:
7 %
8 % X = 1xTS cell, 1 inputs over TS timesteps
9 % Each X{1,ts} = Qx2 matrix, input #1 at timestep ts.
10 %
11 % and returns:
12 % Y = 1xTS cell of 1 outputs over TS timesteps.
13 % Each Y{1,ts} = Qx1 matrix, output #1 at timestep ts.
14 %
15 % where Q is number of samples (or series) and TS is the number of timesteps
16
17 %#ok<*RPMT0>
18
19 % ===== NEURAL NETWORK CONSTANTS =====
20
21 % Input 1
22 x1_step1.xoffset = [-1.0446261962651;-0.775832290325625];
23
```

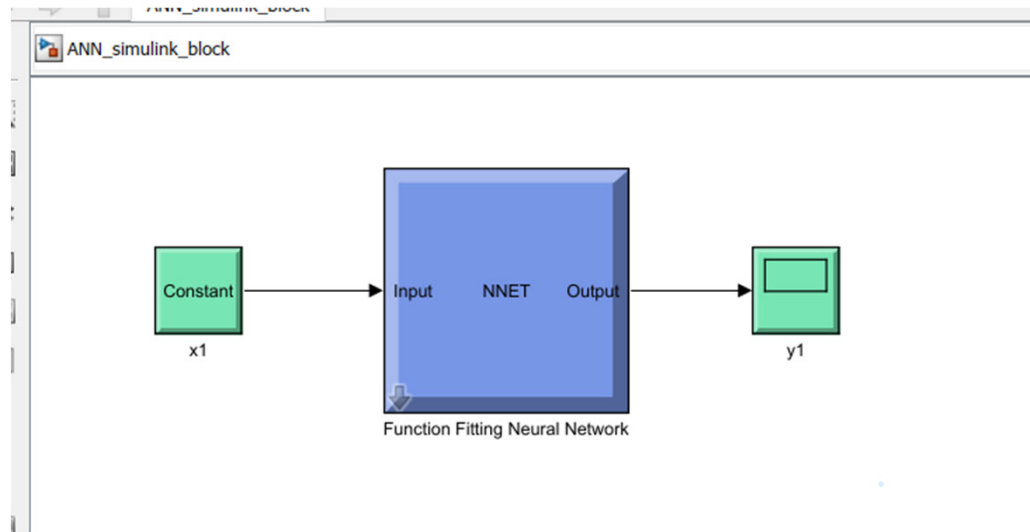
Command Window

```
>> xa = X3(1:10,:);
>> ya = Y3(1:10);
>> [Y,Xf,Af] = myANN(xa);
>> Y
```

```
Y =

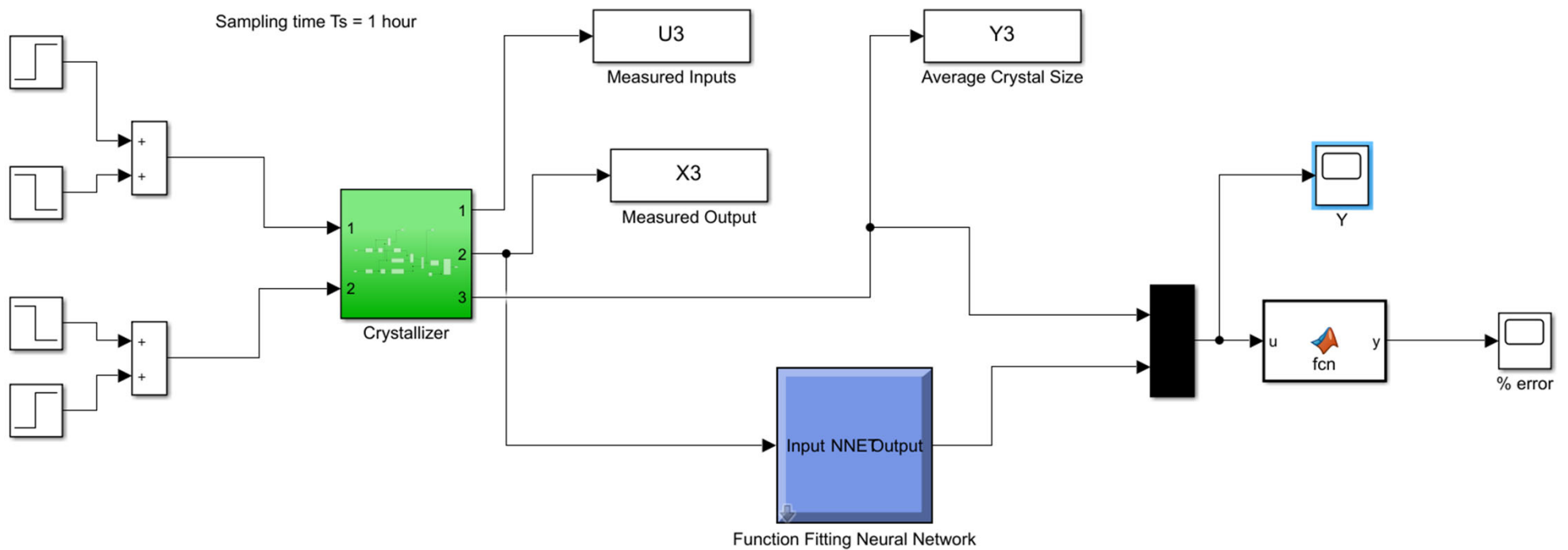
    10.0000
    10.0000
    10.2129
    10.7448
    11.3917
    12.0117
    12.5686
    13.0615
    13.4829
    13.8192
```

Crystallization – ANN model – generate Simulink ANN block



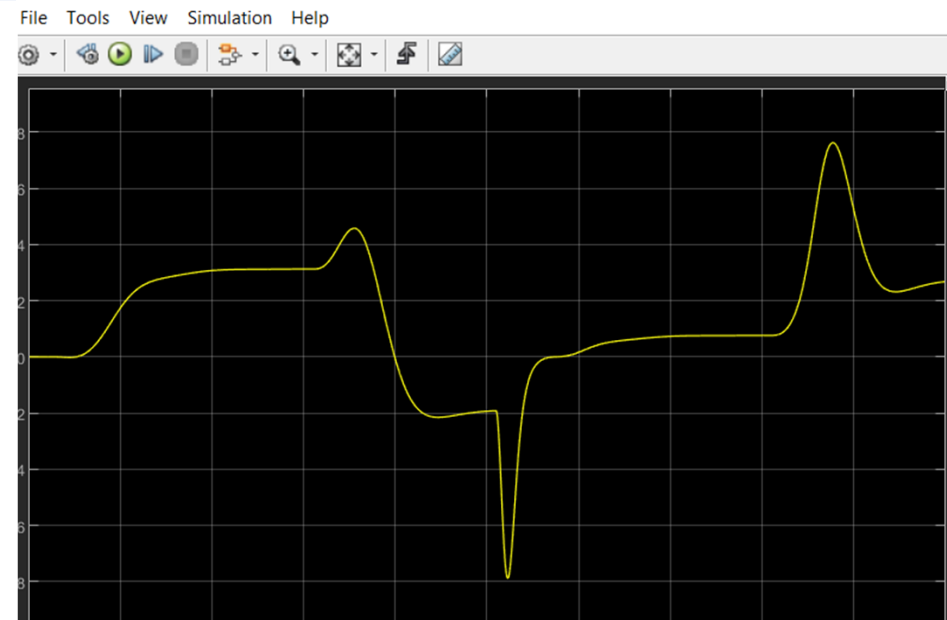
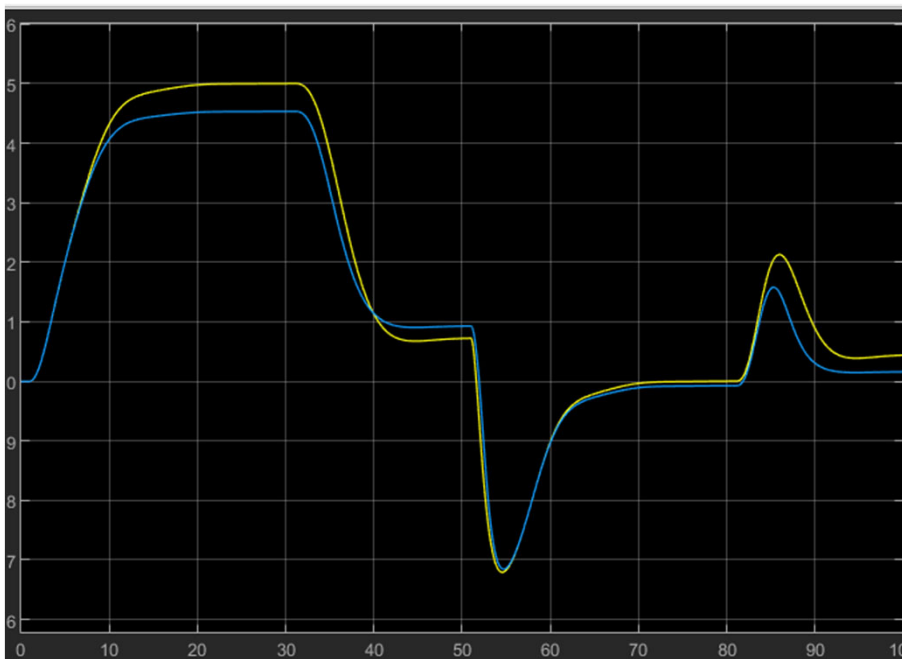
- We can use the generated Simulink ANN block in our Simulink model
- Note that x_1 is a vector of length 2, i.e., two inputs, and x_2 is the predicted output, i.e., average crystal size
- Copy and paste the block directly to our Simulink model

Crystallization – ANN model implementation



Crystallization – ANN model – serves as a soft sensor

- Figure on the left shows:
 - Blue line = actual value
 - Yellow line = predicted value
- Figure below indicates the prediction error, i.e., within $\pm 8\%$

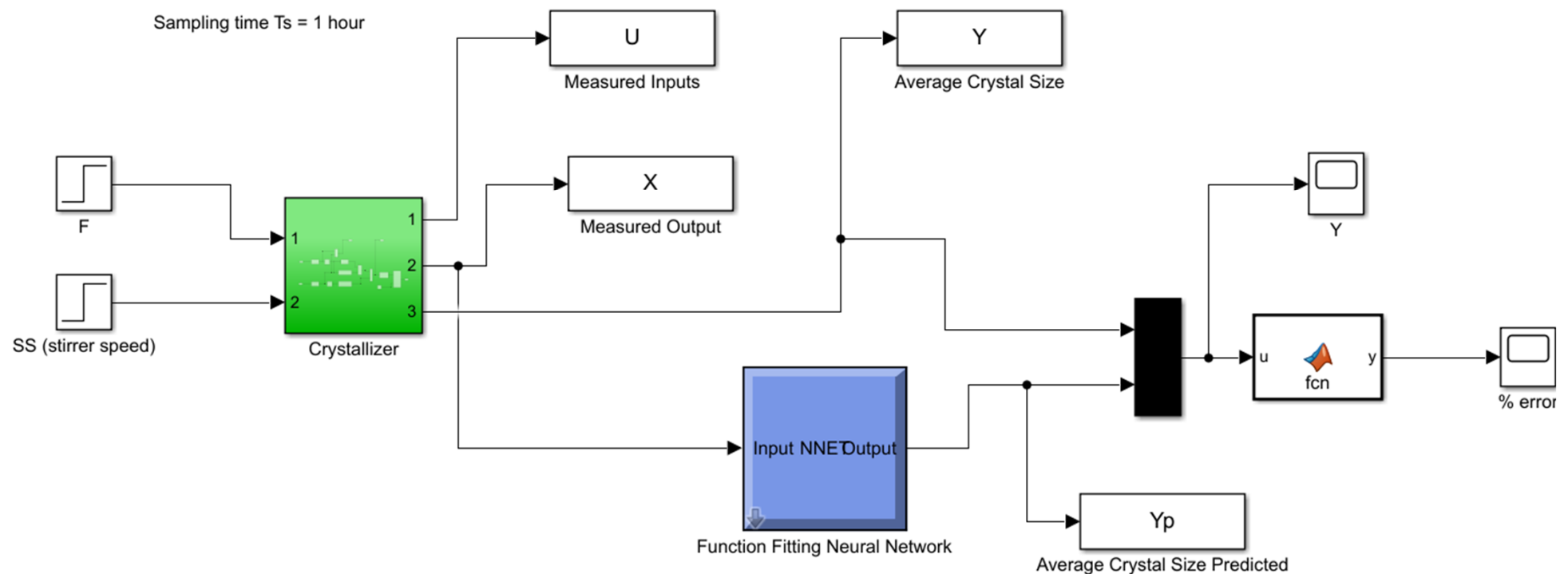


Crystallization – PI controller using ANN soft sensor

- Let's use the ANN soft sensor to control the average crystal size
- Choose the feed flow rate as the manipulated variable
- Use System Identification to obtain a transfer function
- 95% model fitness – System Identification gives the following transfer function:

$$G_p = \frac{Y_p(s)}{F(s)} = \frac{14.542(1 + 0.197s)\exp(-0.06s)}{(3.5899s + 1)(0.058s + 1)}$$

Crystallization – ANN soft sensor – transfer function development – step test in F



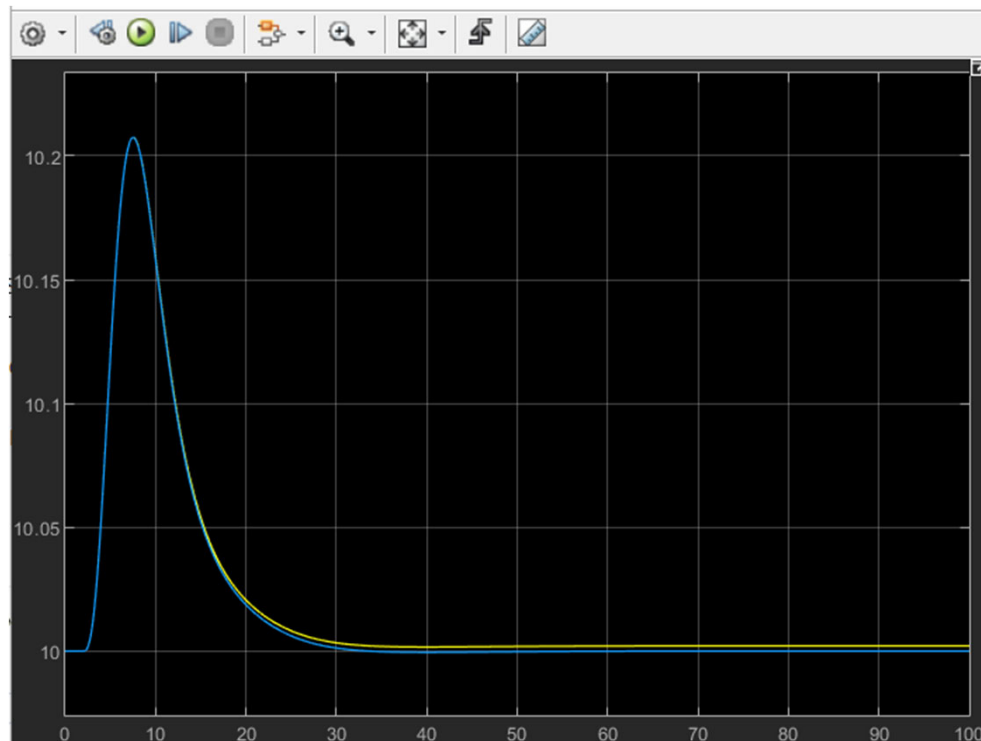
Crystallization – ANN Soft Sensor – PI controller tuning

- Use Control System Design and choose 'Robust response time' tuning for the PI controller:

$$G_c(s) = 0.0683 \left(1 + \frac{1}{1.1s} \right)$$

- Gain Margin (GM) = 31.6 dB and Phase Margin (PM) = 60 deg
- PI controller is very robust, e.g., target GM about 8 to 9 dB and PM about 50 – 60 deg
- Considering the ANN model might not capture certain dynamics of the process, it is reasonable to apply a robust tuning to accommodate the potential model/process mismatch

Crystallization – ANN Soft Sensor with PI controller – closed-loop response to step change in Stirrer Speed



- Let's apply a step change of 0.5 magnitude to the stirrer speed
- Figure on the left shows the closed-loop response
 - Blue line = actual average crystal size
 - Yellow line = predicted average crystal size

Crystallization – ANN Soft Sensor with PI controller – closed-loop response to step change in setpoint

Fig. 1

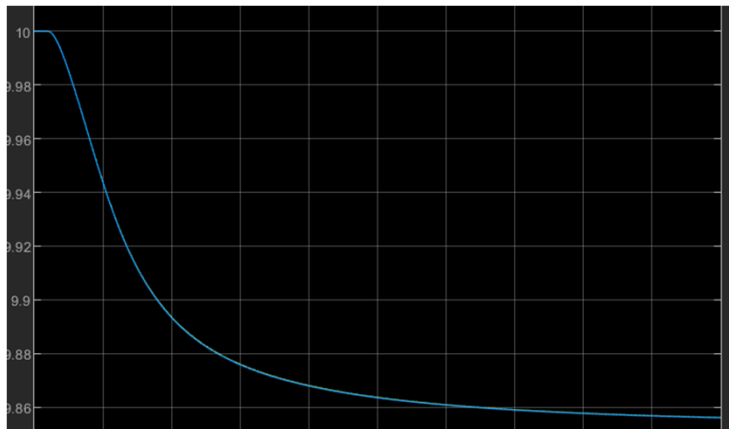
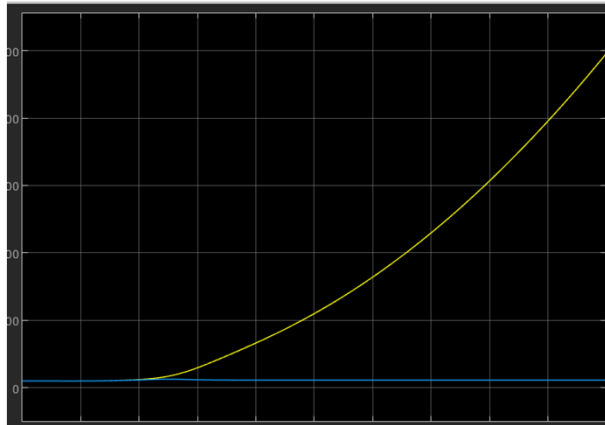


Fig. 2



- For small change in setpoint i.e., from 10 mm to 9.85 mm, the setpoint tracking is stable (Fig. 1)
 - Very small error from the actual value of average crystal size
- For a larger setpoint change, i.e., from 10 mm to 9 mm, the response of actual average crystal size is unstable (Fig. 2)
- For this unstable case, we can apply a supervisor controller using the actual average crystal size
 - Note that in practice the analysis from the lab can be used to supervise the PI controller based on predicted value
 - Use nonlinear controller

Summary

- ANN modelling have received widespread attentions across different science and engineering fields
- MATLAB offers several toolboxes for machine learning, e.g., Neural Net Fitting, Deep Learning, etc.
- ANN model can be used as a soft sensor to measure a variable that is difficult to measure directly, e.g., molecular weight of polymer, crystal size, etc.
- Number of hidden layers affect the ANN modelling predictive capability, e.g., complex data requires more hidden layers