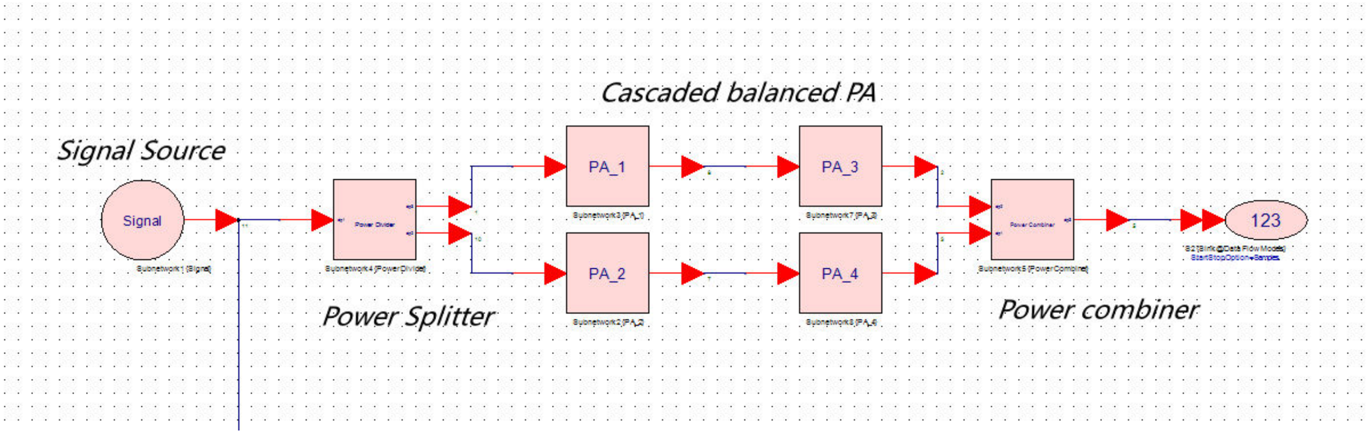
# Supplementary Information for Final Lab

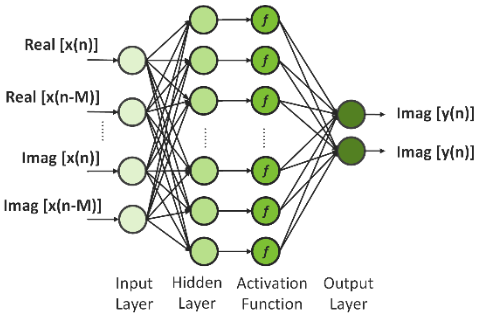
- **Practical Application Scenario – Nonlinear System with balanced PA**



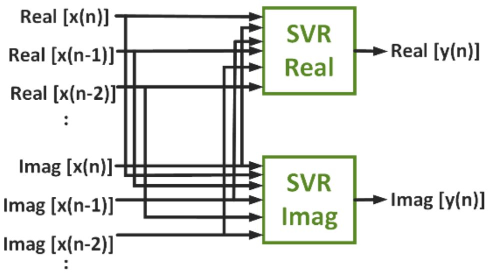
- **Behavioral Modeling Based on Machine Learning**

Behavioral modeling plays a critical role in predicting nonlinearity. Many models have been proposed, such as Volterra, memory polynomial (MP) models. To further enhance modeling accuracy, models based on machine learning are proposed, such as real-valued time-delay neural network (RVTDNN), time-delay support vector regression (SVR) and so on.

**Real-Valued Time-Delay Neural Network**

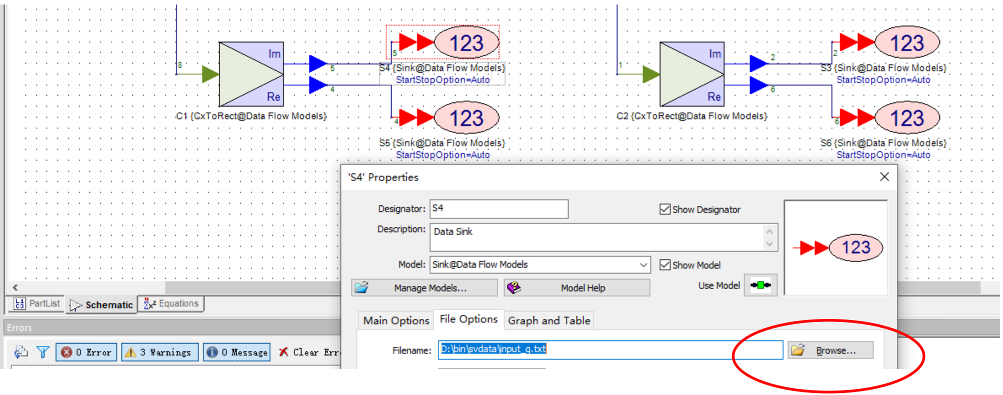


**Time-Delay Support Vector Regression**



- ## Obtain Input/Output Signals in *SystemVue*

Open "**DPD Extraction**" schematic, **Sink** is used to collect input signals and output signals into txt files for extracting DPD coefficients in Matlab. Choosing **file path** by click Browse, then clicking **Run Analyses**.



'S4' Properties

Designator: S4

☑ Show Designator

Description: Data Sink

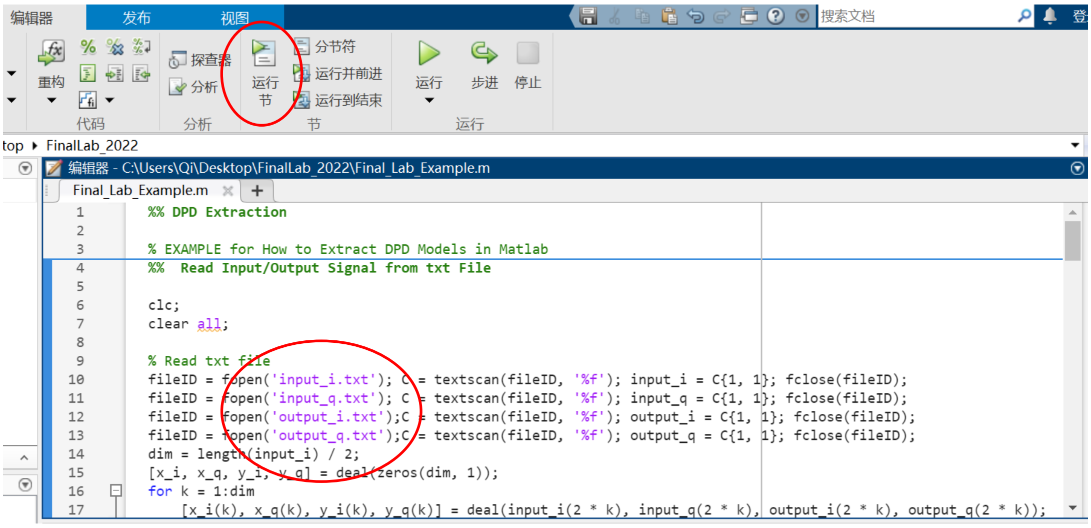
Model: Sink@Data Flow Models

☑ Show Model

Filename: D:\bin\svdata\input_q.txt

- ## DPD Extraction in *Matlab*

- Read Input/Output Signal from txt File, Click **Run Section**.



```
%% DPD Extraction

% EXAMPLE for How to Extract DPD Models in Matlab
%%  Read Input/Output Signal from txt File

clc;
clear all;

% Read txt file
fileID = fopen('input_i.txt'); C = textscan(fileID, '%f'); input_i = C{1, 1}; fclose(fileID);
fileID = fopen('input_q.txt'); C = textscan(fileID, '%f'); input_q = C{1, 1}; fclose(fileID);
fileID = fopen('output_i.txt');C = textscan(fileID, '%f'); output_i = C{1, 1}; fclose(fileID);
fileID = fopen('output_q.txt');C = textscan(fileID, '%f'); output_q = C{1, 1}; fclose(fileID);
dim = length(input_i) / 2;
[x_i, x_q, y_i, y_q] = deal(zeros(dim, 1));
for k = 1:dim
    [x_i(k), x_q(k), y_i(k), y_q(k)] = deal(input_i(2 * k), input_q(2 * k), output_i(2 * k), output_q(2 * k));
```

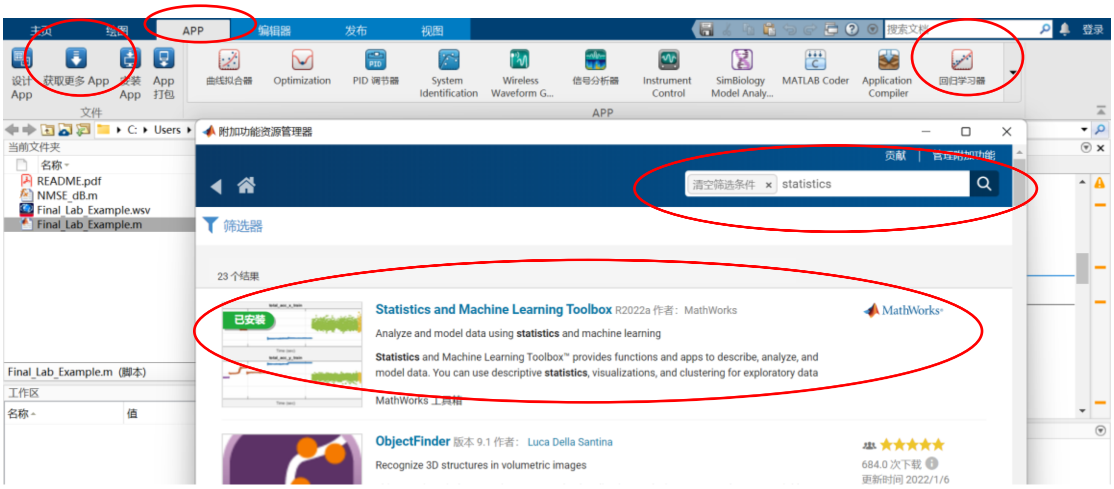
- ## <u>Example I: Feedforward Neural Network</u>

Using the function of **feedforwardnet**, which created a single hidden layer neural network. The number of neuron in hidden layer can be adjusted. (What other parameters of the model can be changed and what impact will changing them have?)

```
•  Set Input/Output Matrix;

•  Adjust Parameters and Train Models;

•  Check Modeling Performance and Save Models;
```

- <u>**Example II: Regression Learner**</u>

If you have not installed **Regression Learner**. Click **Get More Apps**, search **Statistics and Machine Learning Toolbox** and install it.



23 个结果

Statistics and Machine Learning Toolbox R2022a 作者: MathWorks

Analyze and model data using statistics and machine learning

Statistics and Machine Learning Toolbox™ provides functions and apps to describe, analyze, and model data. You can use descriptive statistics, visualizations, and clustering for exploratory data

MathWorks 工具箱

ObjectFinder 版本 9.1 作者: Luca Della Santina

Recognize 3D structures in volumetric images

684.0 次下载

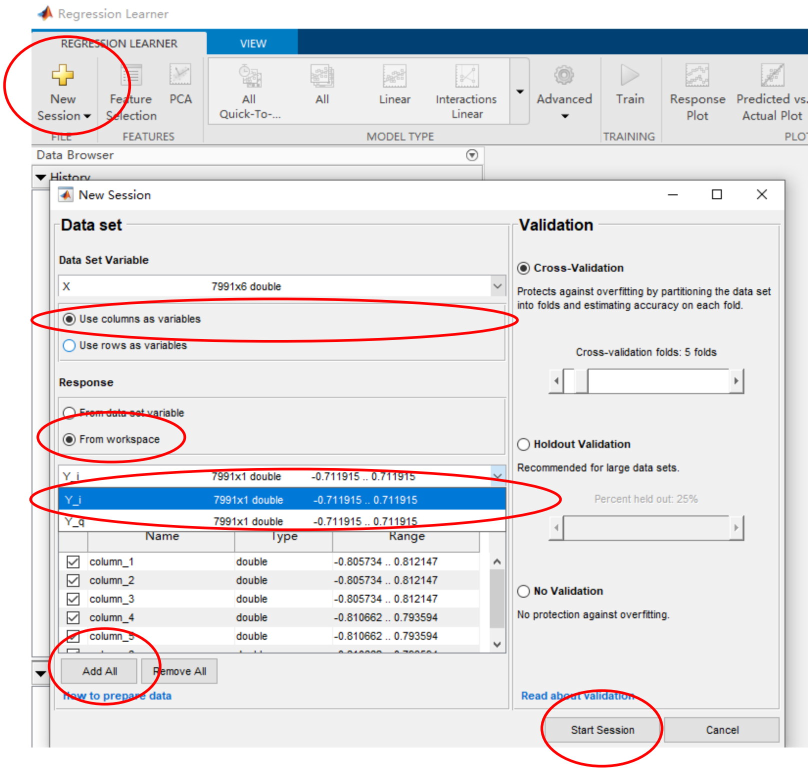
更新时间 2022/1/6

```
•   Set Input/Output Matrix;

•   Train Models in App and Save to Workspace;

•   Check Modeling Performance and Save Models;
```

- **Train Models in App**

Train models in **Regression Learner**. Click New Session and select data set variable. Click Start Session. (**Note**: **Two** models need to be trained, one output is **Y_i**, the other output is **Y_q**.)

**For R2020a or Newer Versions:**



**Data set**

Data Set Variable

X 7991x6 double

● Use columns as variables

○ Use rows as variables

Response

○ From data set variable

● From workspace

| | | |
|---|---|---|
| Y_i | 7991x1 double | -0.711915 .. 0.711915 |
| Y_i | 7991x1 double | -0.711915 .. 0.711915 |
| Y_q | 7991x1 double | -0.711915 .. 0.711915 |

| | Name | Type | Range |
|---|---|---|---|
| ☑ | column_1 | double | -0.805734 .. 0.812147 |
| ☑ | column_2 | double | -0.805734 .. 0.812147 |
| ☑ | column_3 | double | -0.805734 .. 0.812147 |
| ☑ | column_4 | double | -0.810662 .. 0.793594 |
| ☑ | column_5 | double | -0.810662 .. 0.793594 |

**Validation**

● Cross-Validation

Protects against overfitting by partitioning the data set into folds and estimating accuracy on each fold.

Cross-validation folds: 5 folds

○ Holdout Validation

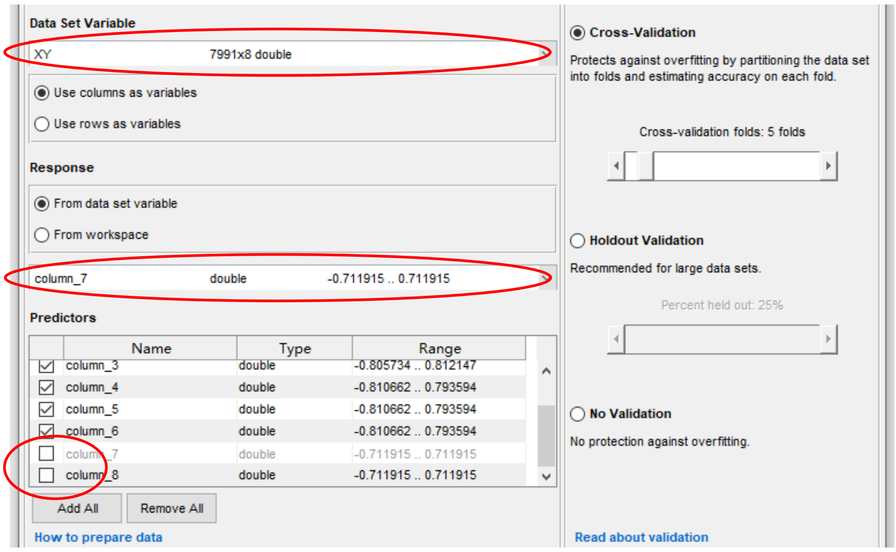
Recommended for large data sets.

Percent held out: 25%

○ No Validation

No protection against overfitting.

### For R2019b or Earlier Version



Data Set Variable

XY 7991x8 double

● Use columns as variables

○ Use rows as variables

Response

● From data set variable

○ From workspace

column_7 double -0.711915 .. 0.711915

Predictors

| | Name | Type | Range |
|---|---|---|---|
| ☑ | column_3 | double | -0.805734 .. 0.812147 |
| ☑ | column_4 | double | -0.810662 .. 0.793594 |
| ☑ | column_5 | double | -0.810662 .. 0.793594 |
| ☑ | column_6 | double | -0.810662 .. 0.793594 |
| □ | column_7 | double | -0.711915 .. 0.711915 |
| □ | column_8 | double | -0.711915 .. 0.711915 |

Add All Remove All

How to prepare data

● Cross-Validation

Protects against overfitting by partitioning the data set into folds and estimating accuracy on each fold.

Cross-validation folds: 5 folds

○ Holdout Validation

Recommended for large data sets.

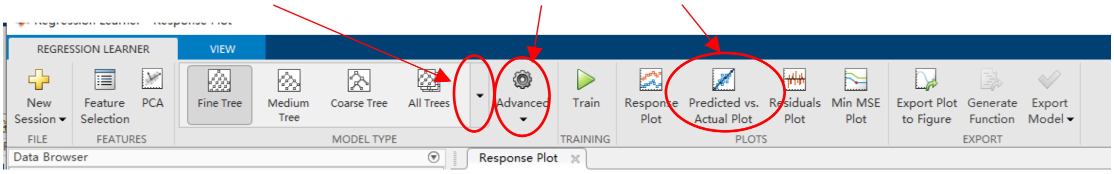
Percent held out: 25%

○ No Validation

No protection against overfitting.

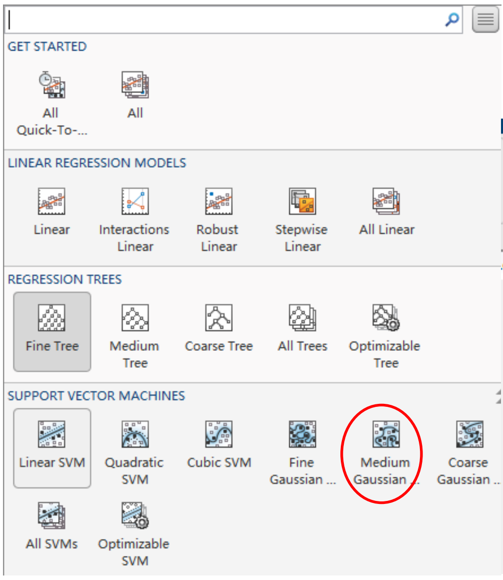
Read about validation

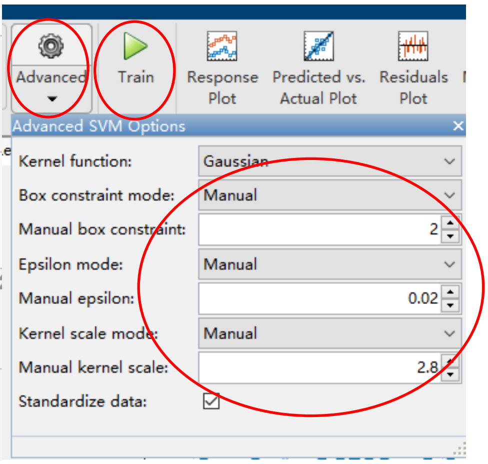
Choose different model. Adjust advanced options. Plot.



### AN EXAMPLE using SVM

Select **Medium Gaussian SVM**, adjust advanced options as following graph. Click Train.





Advanced SVM Options

Kernel function: Gaussian

Box constraint mode: Manual

Manual box constraint: 2

Epsilon mode: Manual

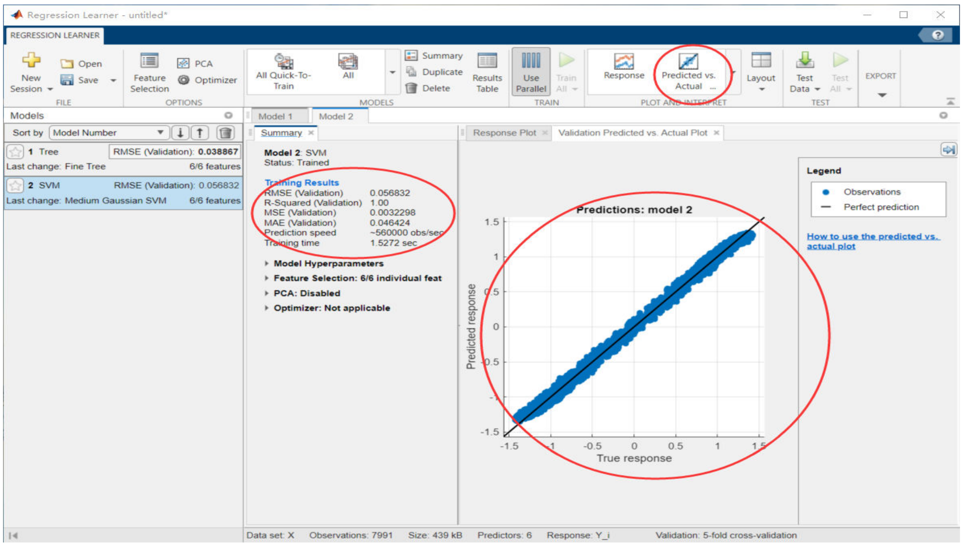
Manual epsilon: 0.02

Kernel scale mode: Manual

Manual kernel scale: 2.8

Standardize data: ☑

Check by RMSE and diagrams.



Sort by Model Number

1 Tree — RMSE (Validation): **0.038867**
Last change: Fine Tree — 6/6 features

2 SVM — RMSE (Validation): 0.056832
Last change: Medium Gaussian SVM — 6/6 features

**Model 2**: SVM
Status: Trained

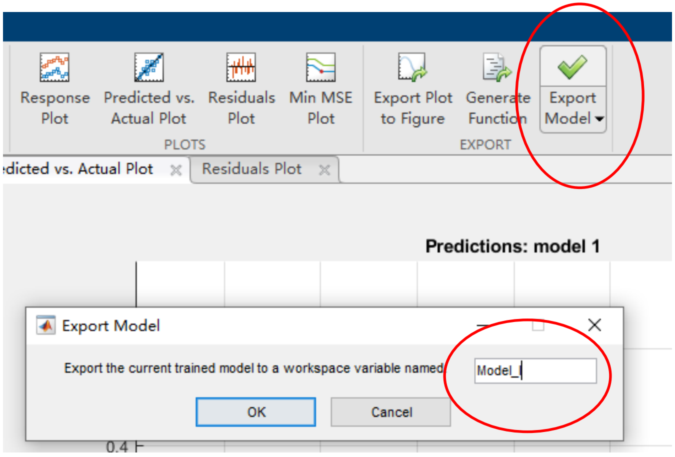
Training Results

| | |
|---|---|
| RMSE (Validation) | 0.056832 |
| R-Squared (Validation) | 1.00 |
| MSE (Validation) | 0.0032298 |
| MAE (Validation) | 0.046424 |
| Prediction speed | ~560000 obs/sec |
| Training time | 1.5272 sec |

- **Model Hyperparameters**
- **Feature Selection: 6/6 individual feat**
- **PCA: Disabled**
- **Optimizer: Not applicable**

Data set: X  Observations: 7991  Size: 439 kB  Predictors: 6  Response: Y_i  Validation: 5-fold cross-validation

Export Model to Workspace.



Export Model

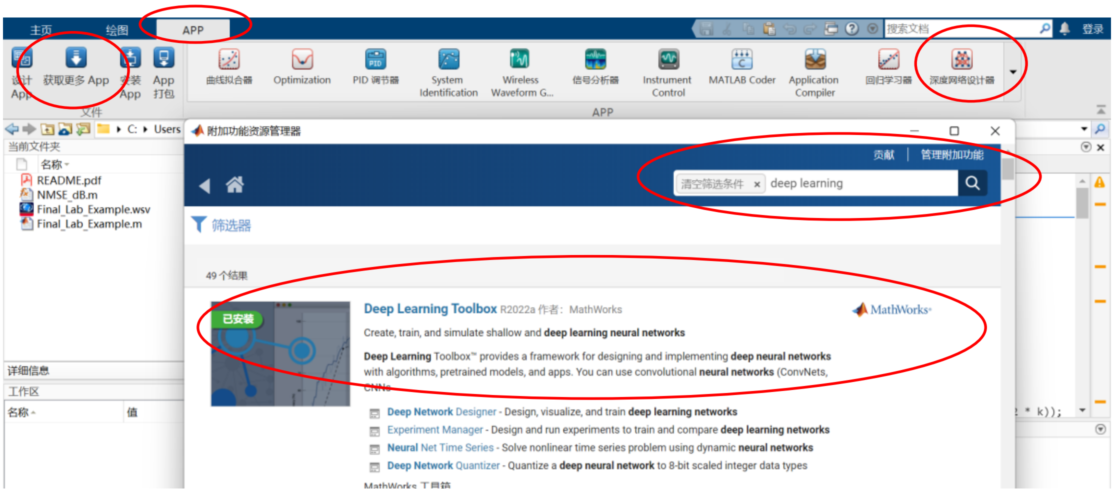
Export the current trained model to a workspace variable named: Model_I

OK  Cancel

Repeat the above steps, train another model of Y_q.

- ## <u>Example III: Deep Network Designer</u>

Using **Deep Network Designer**, which creates deep neural network. If you have not installed **Deep Network Designer**. Click **Get More Apps**, search **Deep Learning Toolbox** and install.



清空筛选条件 × deep learning

49 个结果

Deep Learning Toolbox R2022a 作者：MathWorks

Create, train, and simulate shallow and **deep learning neural networks**

**Deep Learning** Toolbox™ provides a framework for designing and implementing **deep neural networks** with algorithms, pretrained models, and apps. You can use convolutional **neural networks** (ConvNets, CNNs

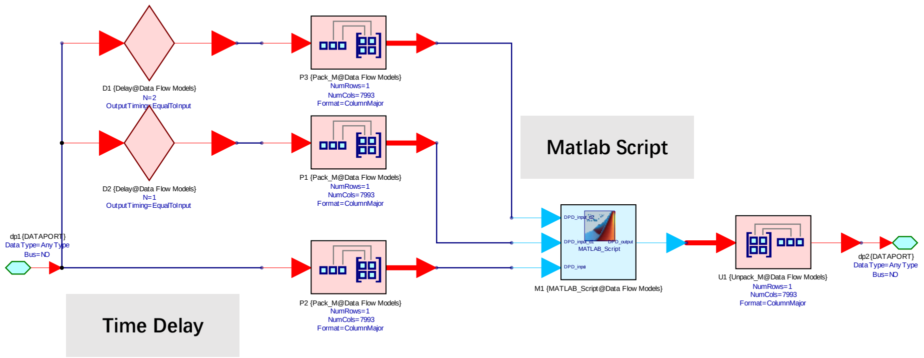
- Deep Network Designer - Design, visualize, and train **deep learning networks**
- Experiment Manager - Design and run experiments to train and compare **deep learning networks**
- Neural Net Time Series - Solve nonlinear time series problem using dynamic **neural networks**
- Deep Network Quantizer - Quantize a **deep neural network** to 8-bit scaled integer data types

MathWorks 工具箱

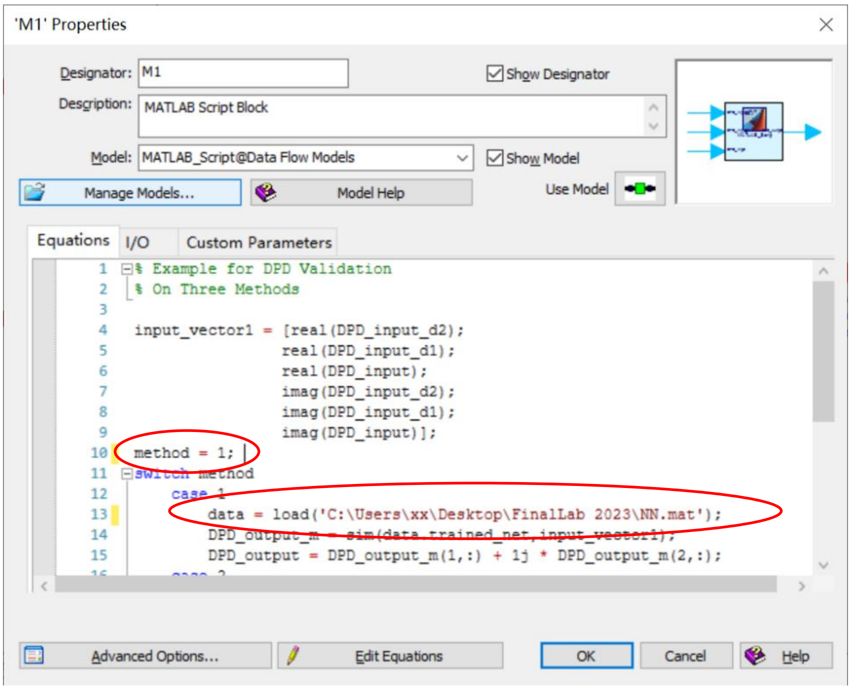
```
•  Design Network and Set Input/Output Matrix;

•  Adjust Parameters and Train Models;

•  Check Modeling Performance and Save Models;
```

- **DPD Validation in *SystemVue***

Open "**DPD Validation**" schematic, and open **DPD(Model)** schematic.



Set **method = 1** and change the file path in **Matlab Script**. Observe DPD results by the NMSE.



Designator: M1 ☑ Show Designator

Description: MATLAB Script Block

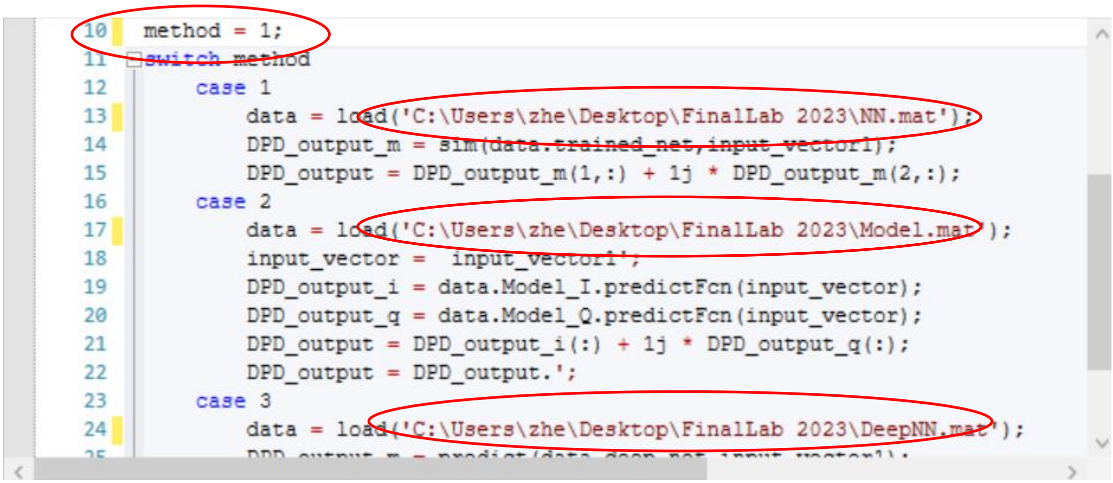
Model: MATLAB_Script@Data Flow Models ☑ Show Model

```
% Example for DPD Validation
% On Three Methods

input_vector1 = [real(DPD_input_d2);
                 real(DPD_input_d1);
                 real(DPD_input);
                 imag(DPD_input_d2);
                 imag(DPD_input_d1);
                 imag(DPD_input)];
method = 1;
switch method
    case 1
        data = load('C:\Users\xx\Desktop\FinalLab 2023\NN.mat');
        DPD_output_m = sim(data.trained_net,input_vector1);
        DPD_output = DPD_output_m(1,:) + 1j * DPD_output_m(2,:);
    case 2
```

In total, we provide examples of 3 methods.

```
•  method=1 Feedforward Neural Network;

•  method=2 Regression Learner;

•  method=3 Deep Network Designer;
```

Run **Example II & III** in ***Matlab*** as before first, and then change the number of **method** to 2 or 3 and the file path in Matlab script in "DPD Validation" schematic in ***SystemVue***.



```
method = 1;
switch method
    case 1
        data = load('C:\Users\zhe\Desktop\FinalLab 2023\NN.mat');
        DPD_output_m = sim(data.trained_net,input_vector1);
        DPD_output = DPD_output_m(1,:) + 1j * DPD_output_m(2,:);
    case 2
        data = load('C:\Users\zhe\Desktop\FinalLab 2023\Model.mat');
        input_vector =  input_vector1';
        DPD_output_i = data.Model_I.predictFcn(input_vector);
        DPD_output_q = data.Model_Q.predictFcn(input_vector);
        DPD_output = DPD_output_i(:) + 1j * DPD_output_q(:);
        DPD_output = DPD_output.';
    case 3
        data = load('C:\Users\zhe\Desktop\FinalLab 2023\DeepNN.mat');
        DPD_output_m = predict(data.deep_net,input_vector1);
```

# Tips

- We recommend that you use the Copyrighted Version of **MATLAB**, otherwise you may encounter problems. Most errors are caused by the wrong file path of the model.
- Verification of DPD performance in SystemVue may take a long time. When optimizing the parameters of the model, you can refer to the modeling performance calculated in MATLAB. In general, the two results are very close.
- Different models have different performance, so try more. Instead of using the example method, we would like to see your own method, such as, GAN, LSTM, and so on.
- The input of the model can be changed. Using more input information may get better performance. Which means you can change the input matrix.
- Have difficulties? Ask for help in the MATLAB official Help or ask TAs.
- Good luck to get excellent results.