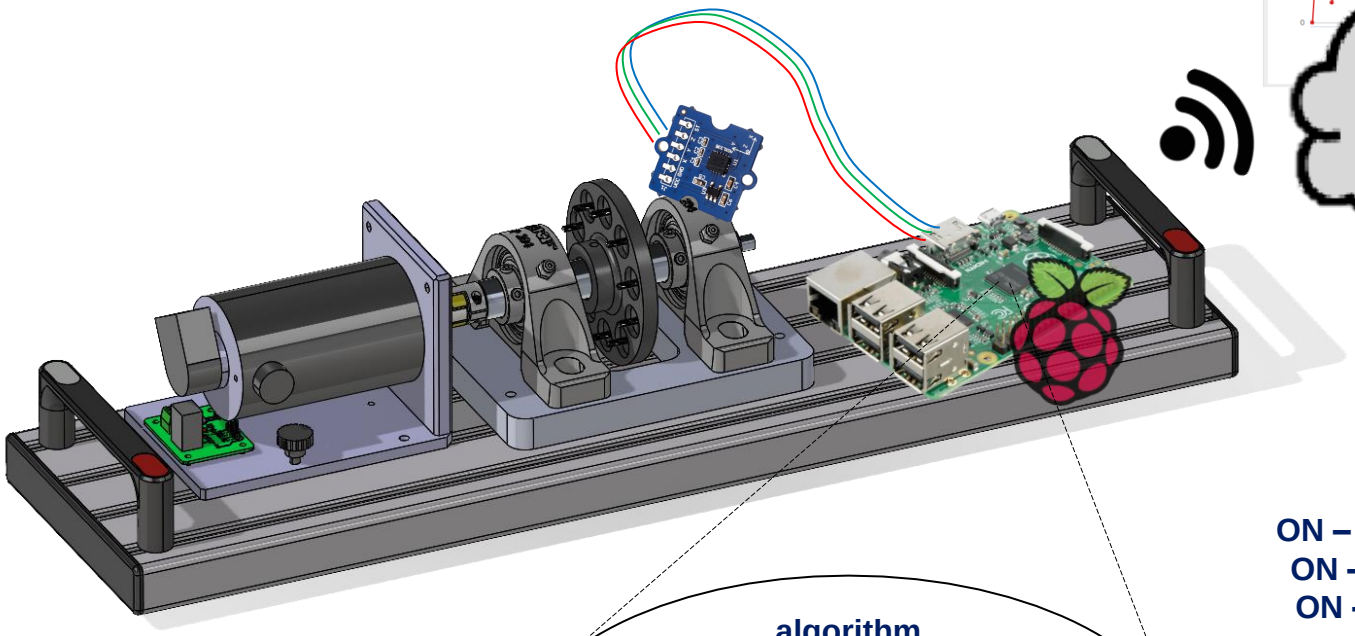
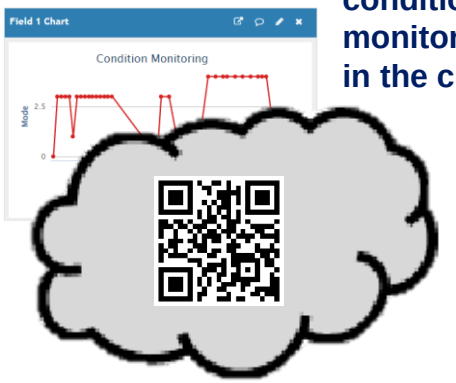
Labor Condition Monitoring – Setup

vibration measurements

accX | accY | accZ





condition monitoring in the cloud

OFF … mode 4
ON – normal operation … mode 3
ON – unbalance HIGH … mode 2
ON – unbalance LOW … mode 1

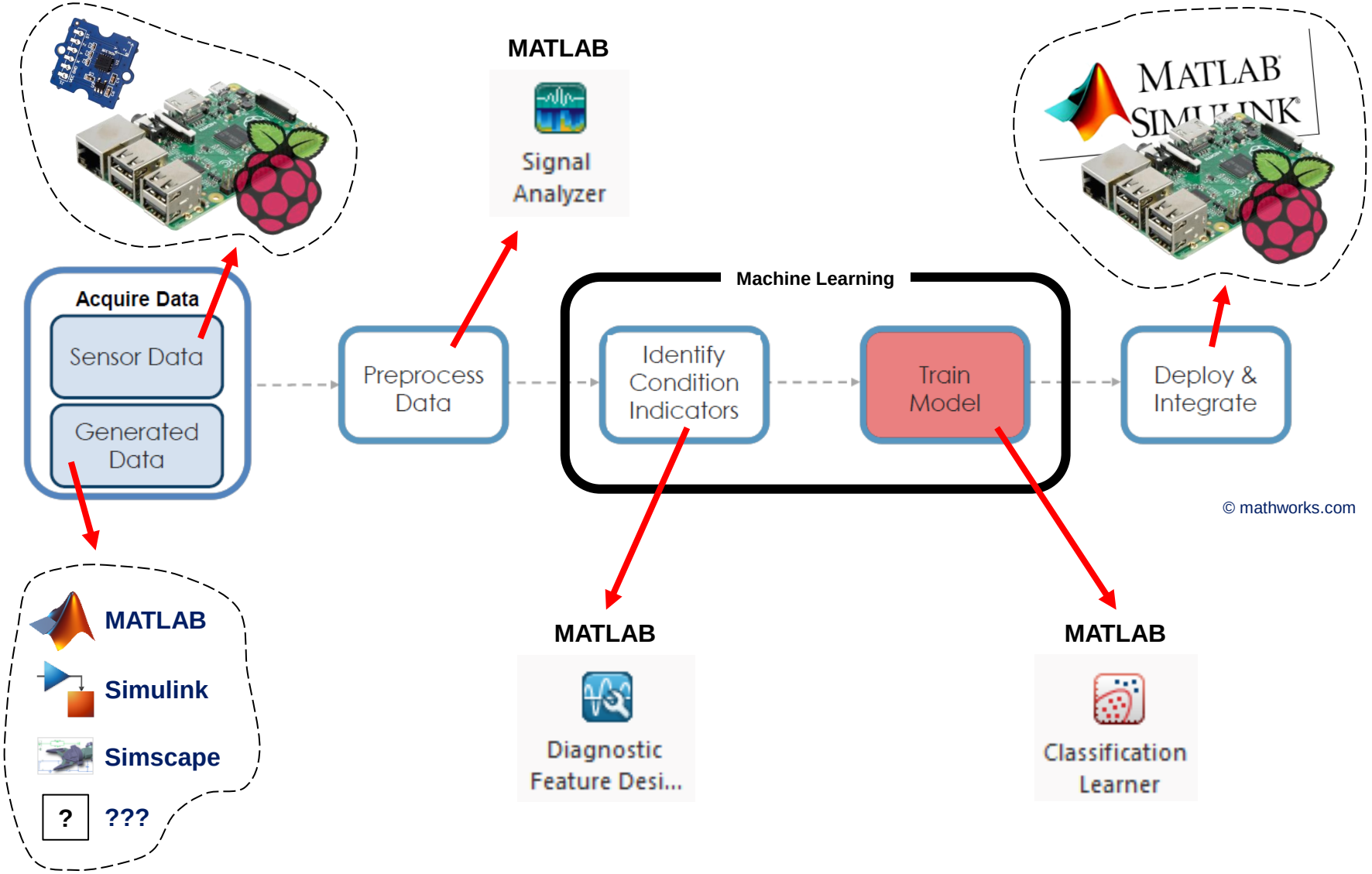
LAB demonstrator consisting of:

- **24V DC-motor**
- **clutch**
- **shaft**
- **bearings**
- **unbalance disc**

algorithm

condition mode determination using ML-model

Condition Monitoring – Goal / Workflow

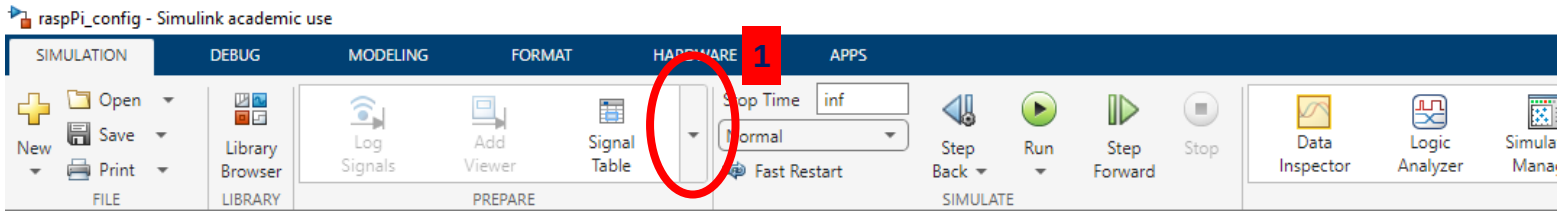


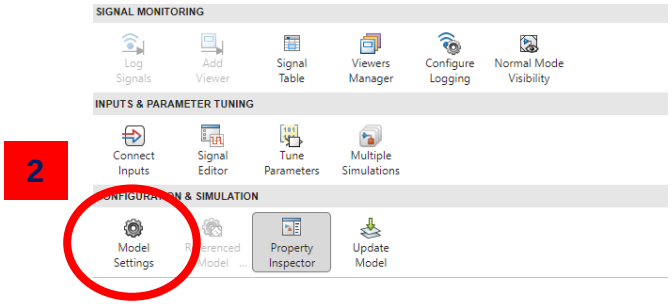
Condition Monitoring – Add-Ons (MATLAB R2024a)

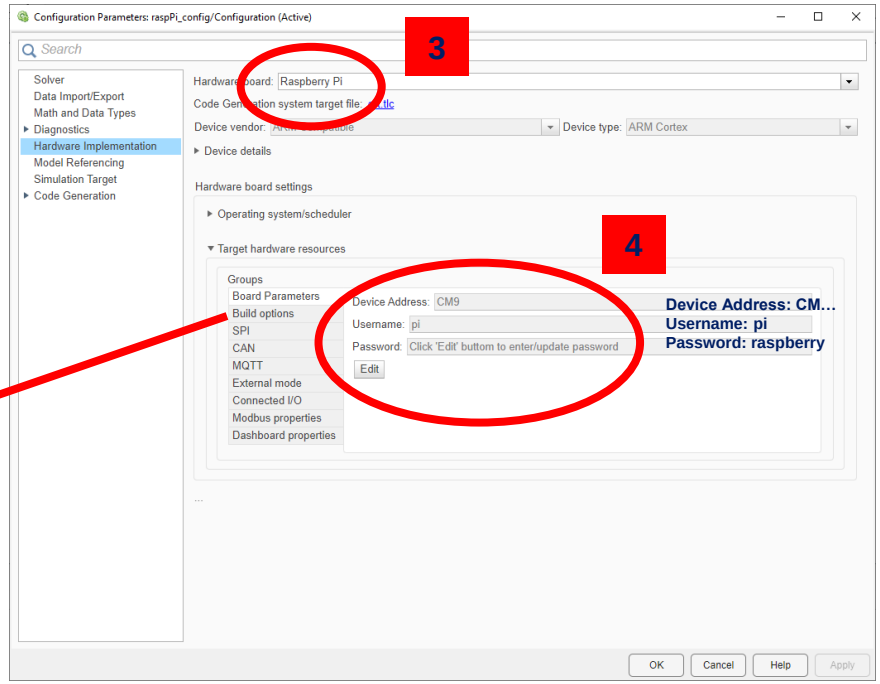
Add-Ons	Requires
Communications Toolbox version 24.1	DSP System Toolbox version 24.1 Signal Processing Toolbox version 24.1
Embedded Coder version 24.1	MATLAB Coder version 24.1
MATLAB Support Package for Raspberry Pi Hardware version 24.1.2	
Predictive Maintenance Toolbox version 24.1	Signal Processing Toolbox version 24.1 Statistics and Machine Learning Toolbox version 24.1 System Identification Toolbox version 24.1
Simulink Coder version 24.1	Simulink version 24.1 MATLAB Coder version 24.1
Simulink Support Package for Raspberry Pi Hardware version 24.1.1	Simulink version 24.1

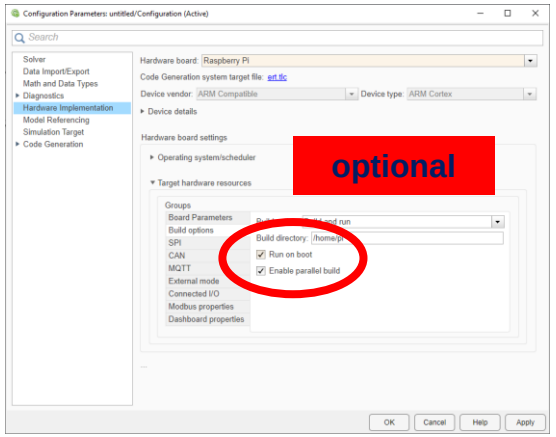
University of Applied Sciences of Upper Austria, Wels

Raspberry Pi Configuration in Simulink (i)

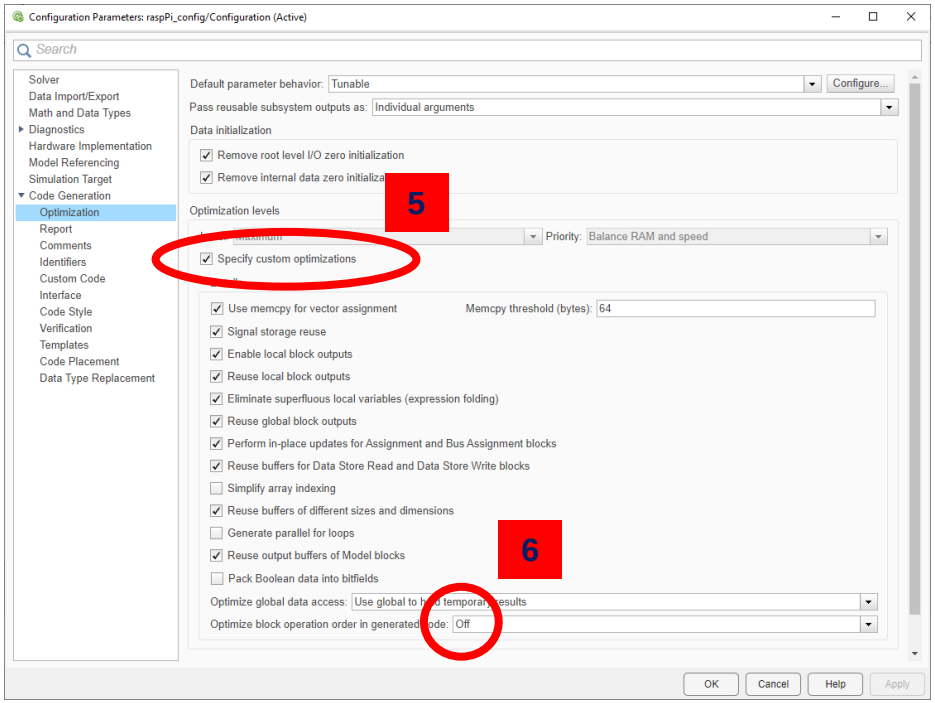


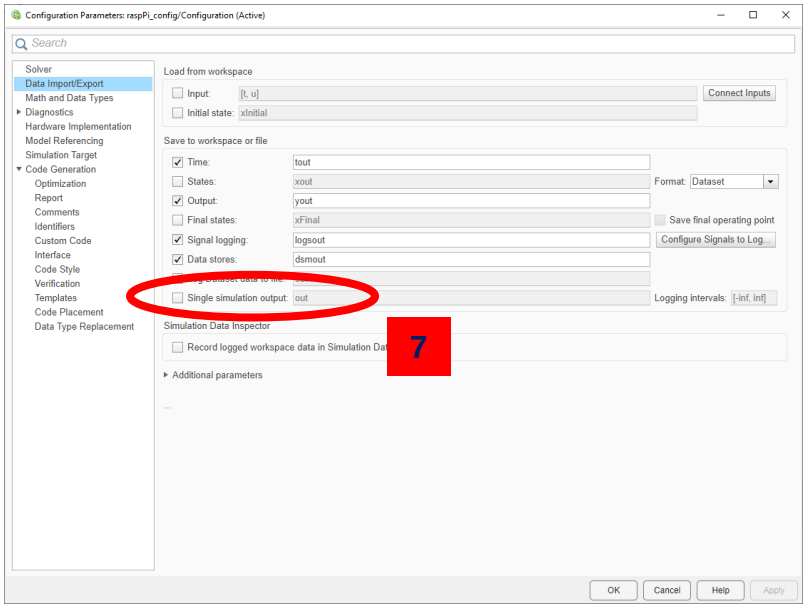




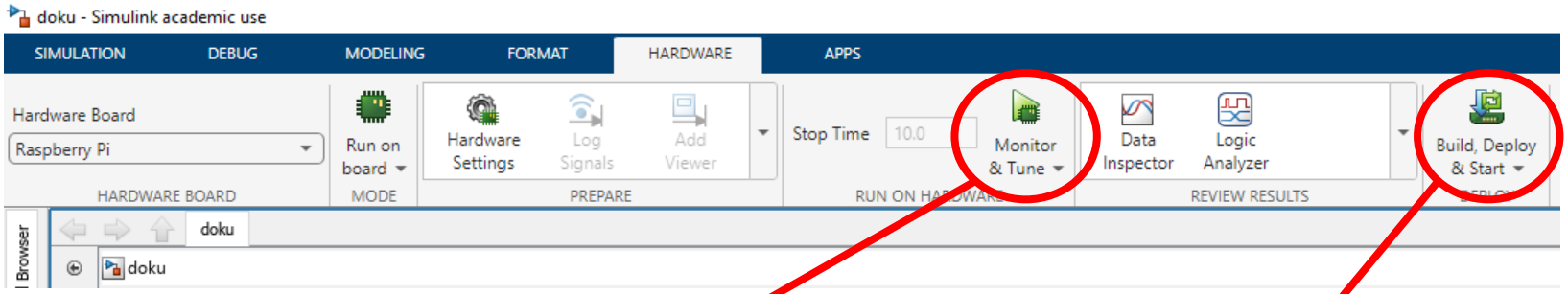


Raspberry Pi Configuration in Simulink (ii)





Raspberry Pi Configuration in Simulink (iii)



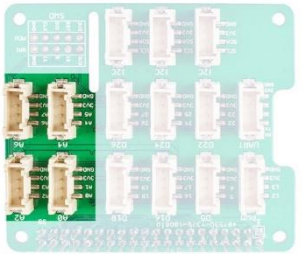
run on RaspPi
& monitor in Simulink

run on RaspPi only

Measuring System / Hardware

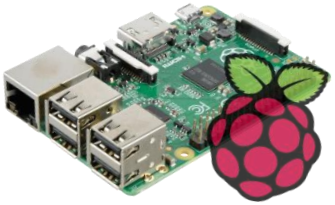
ADXL335

analog



Grove Base HAT

I2C



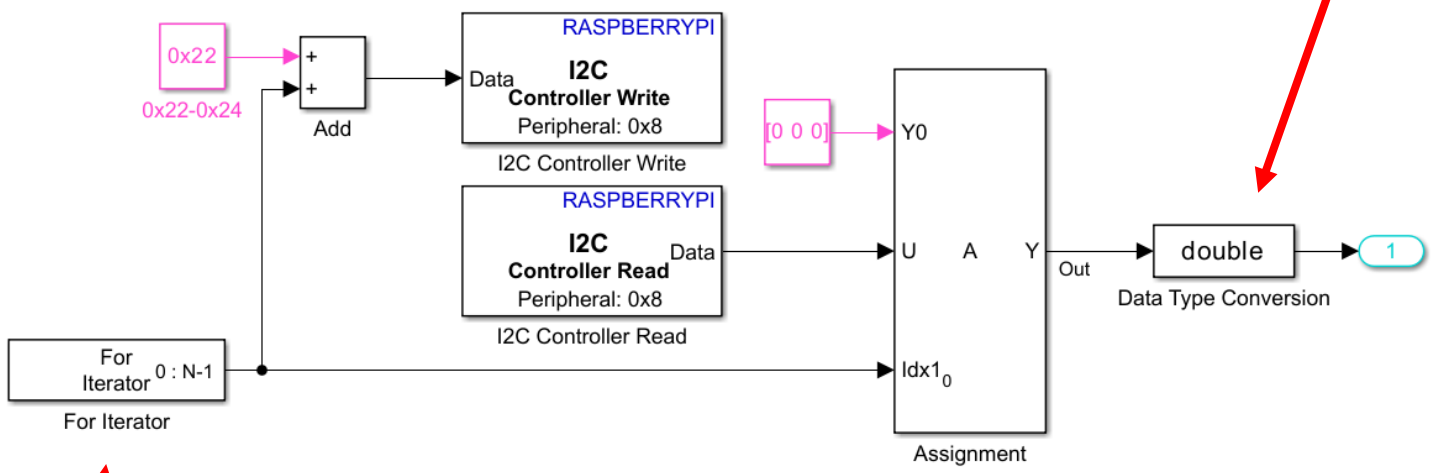
Raspberry Pi 4
(flash card preconfigured)

Measuring System / Software (i)

accSensor values (x,y,z)
in registers 0x22-0x24

in order to avoid implicit data type conversion

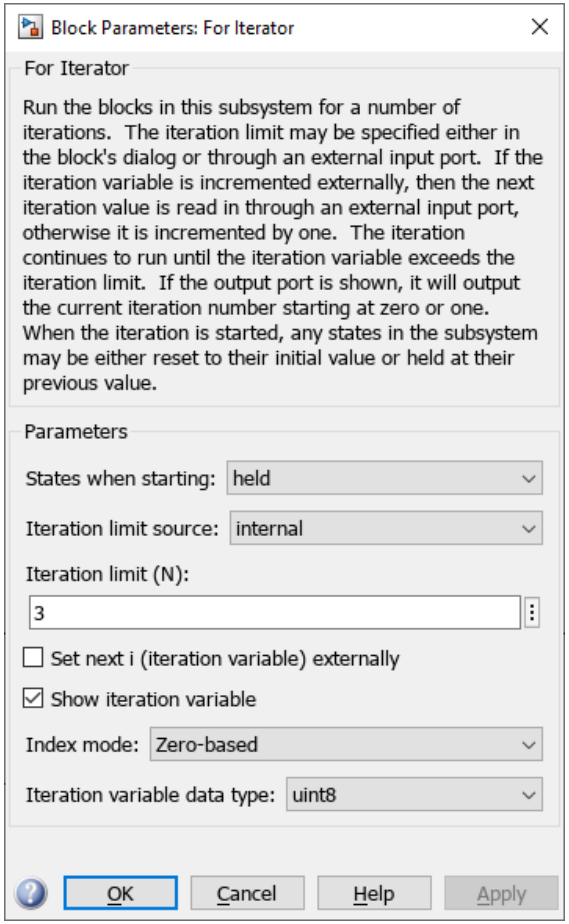
by default, MATLAB/Simulink works with data type double



for loop:
3 cycles for 3 acc-axes

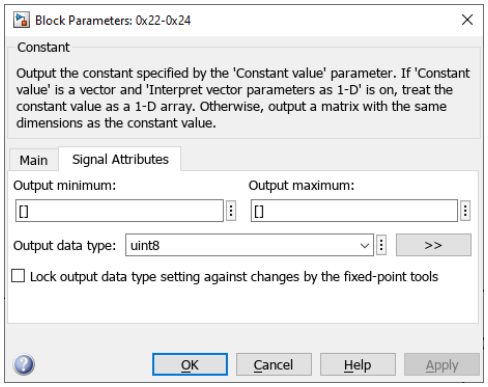
Measuring System / Software (ii)

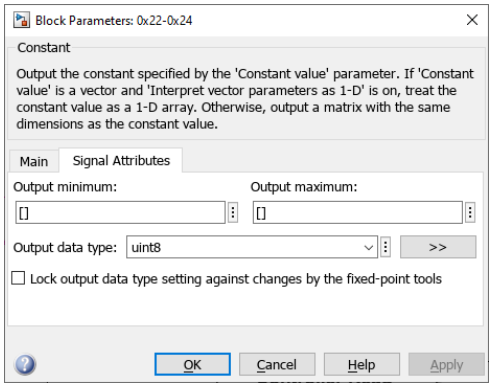
For Iterator



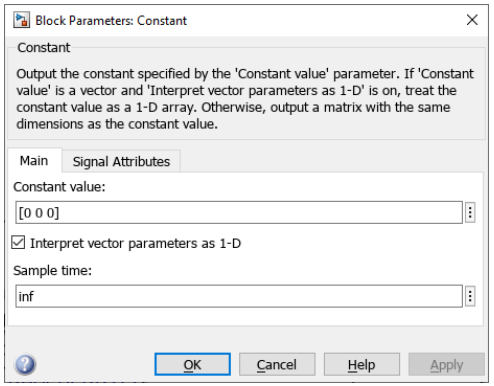
Measuring System / Software (iii)

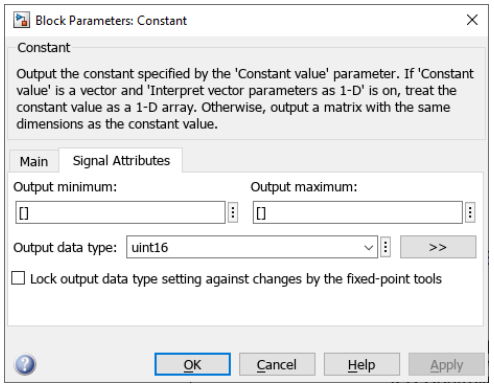
Constant (0x22-0x24)





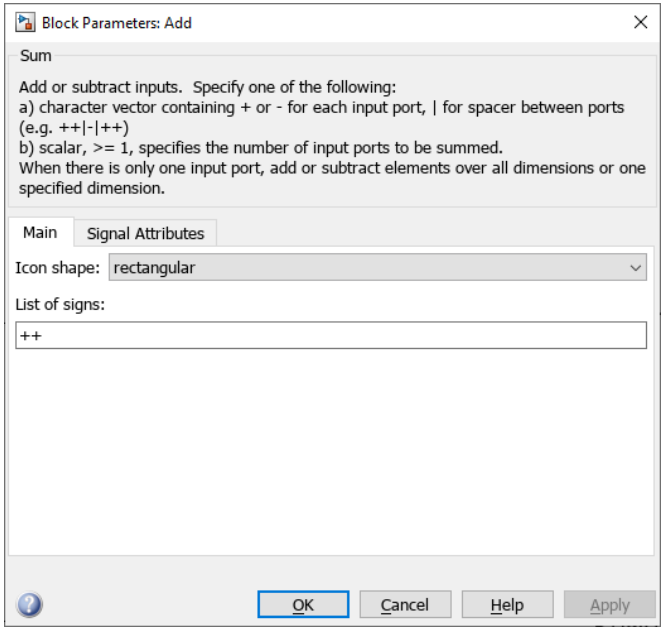
Constant [0 0 0]

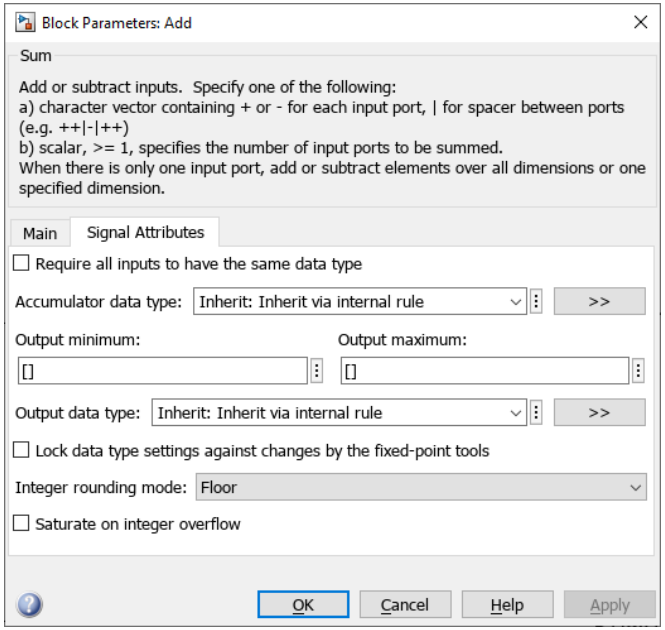




Measuring System / Software (iv)

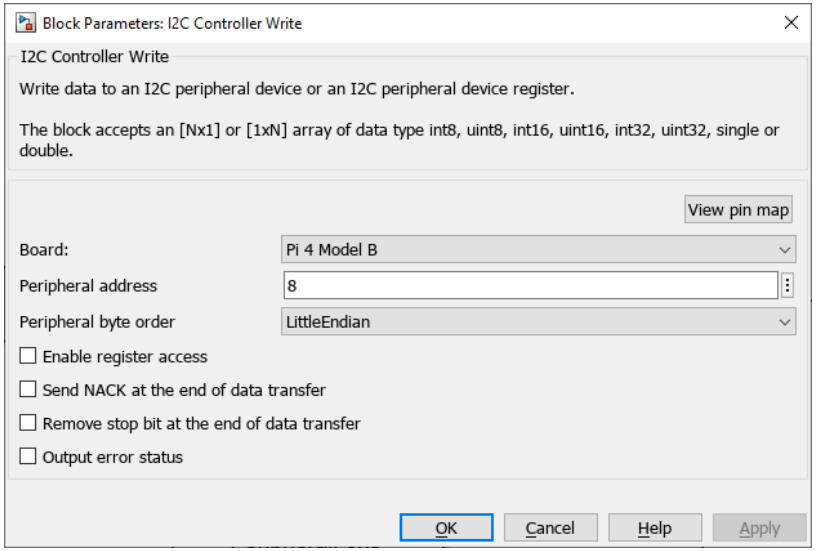
Add

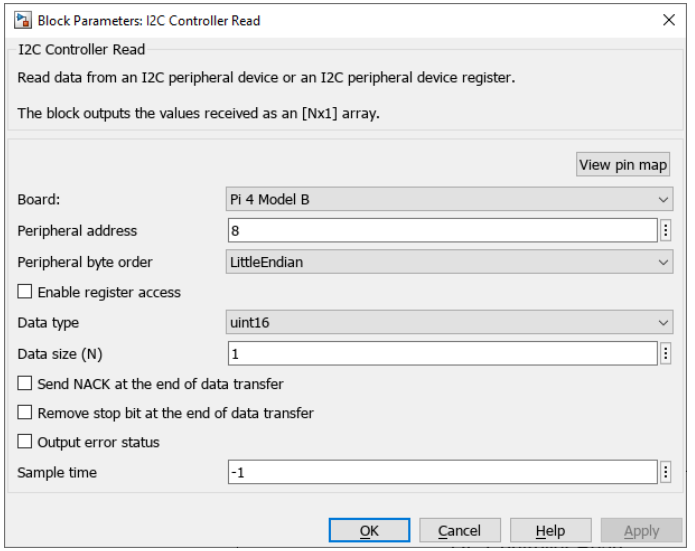




Measuring System / Software (v)

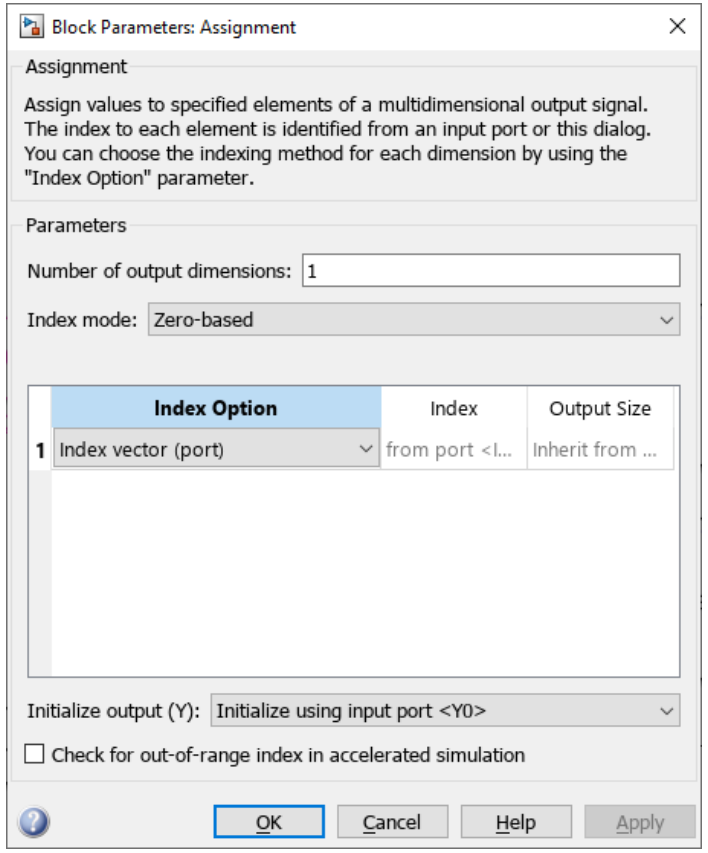
I2C Controller Write/Read





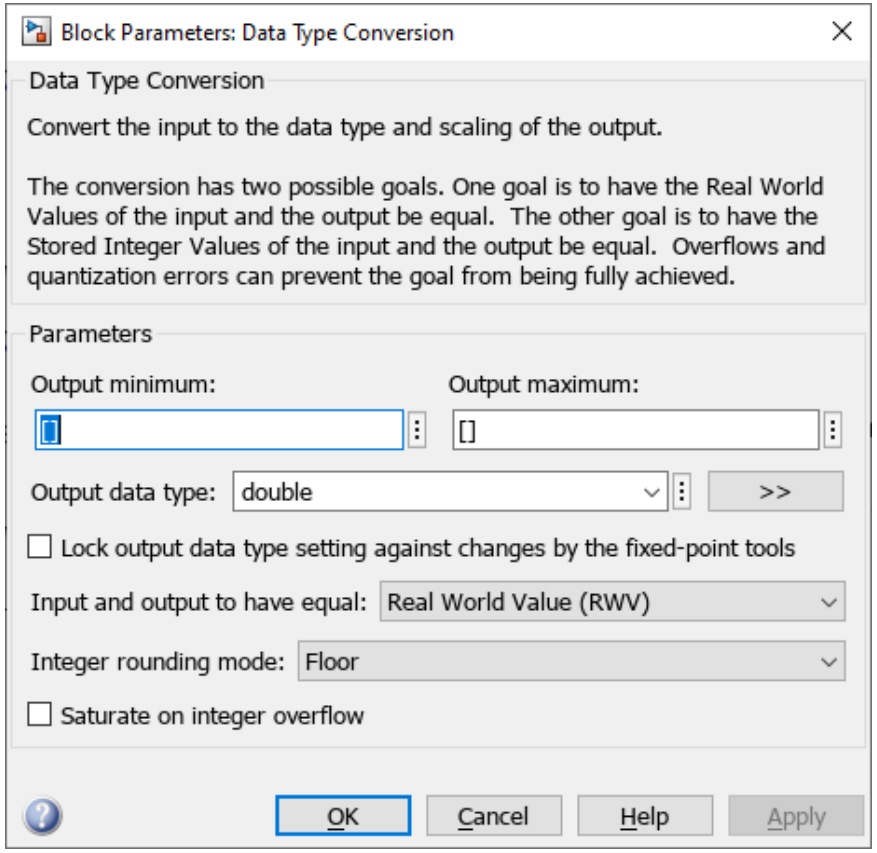
Measuring System / Software (vi)

Assignment



Measuring System / Software (vii)

Data Type Conversion



Signal Analyzer (i)

Signal Analyzer

input format

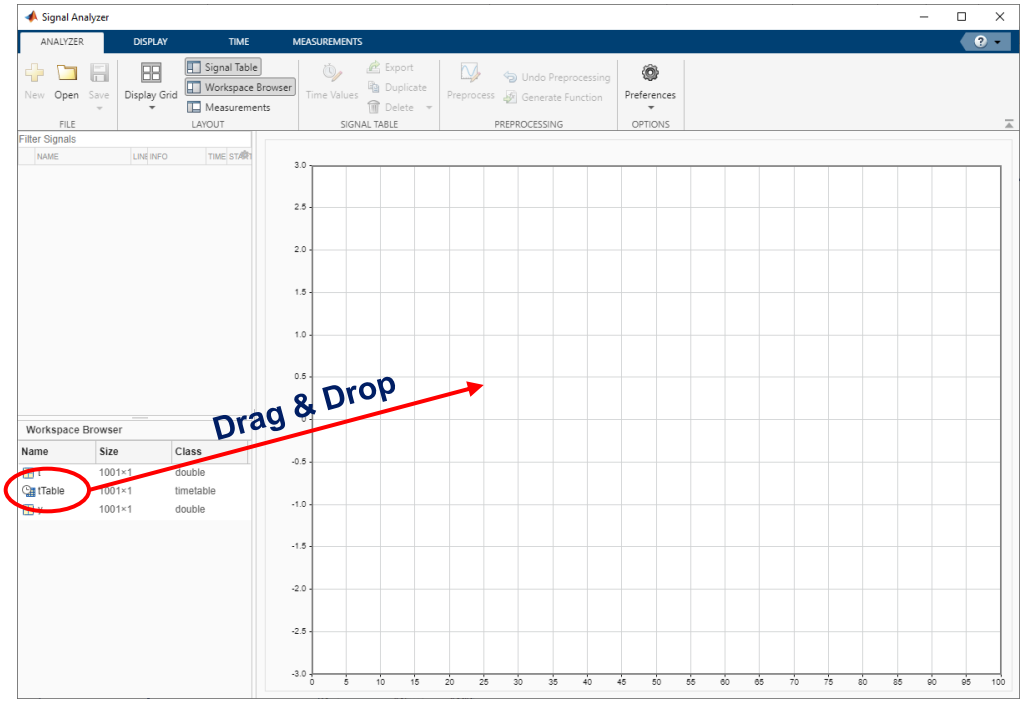
```
t = (0:0.01:10)';
y = sin(t);

% prepare for SignalAnalyzer
x = seconds(t);
tTable = timetable(x,y);
```

tTable

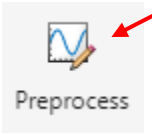
1001x1 timetable

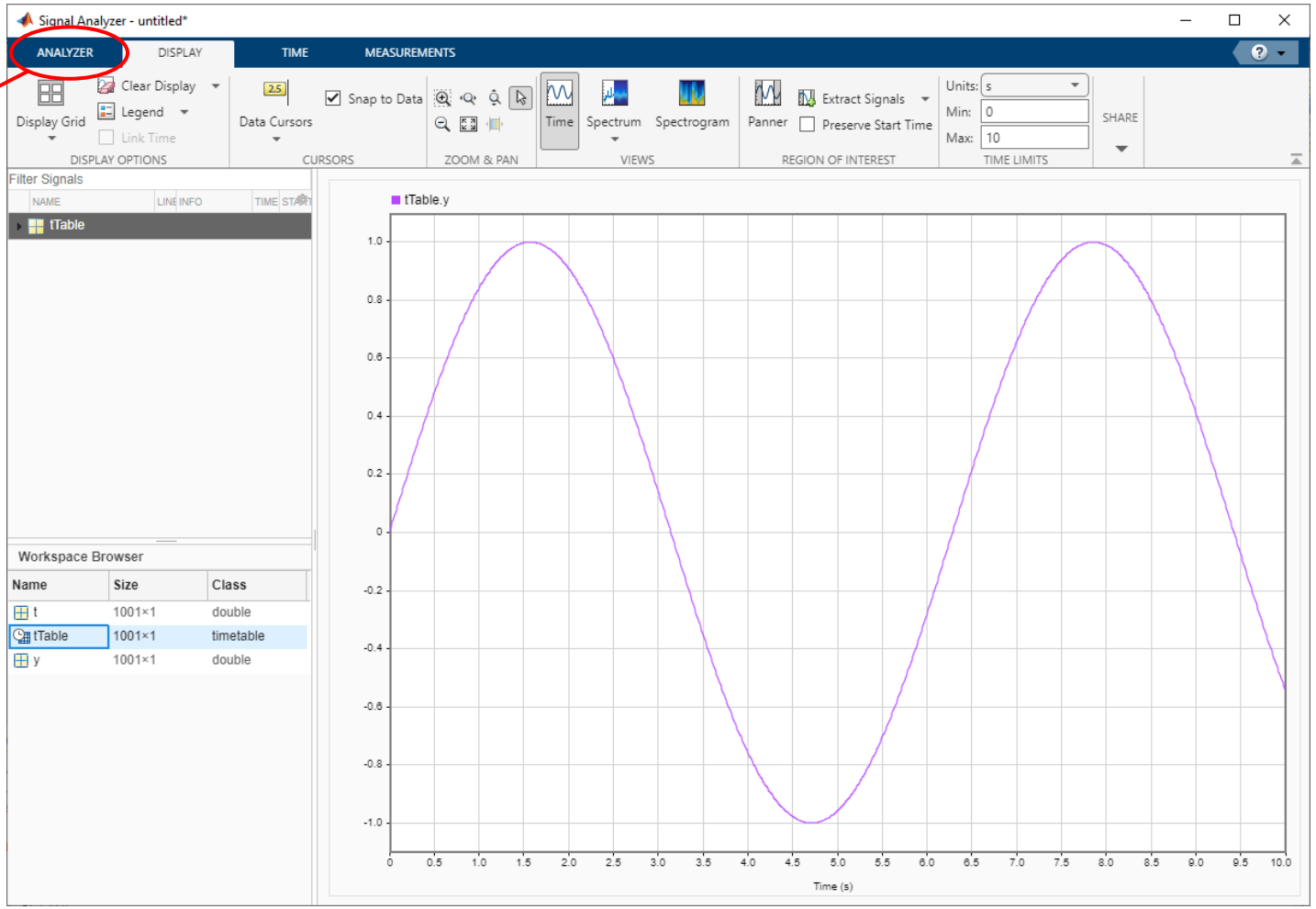
	x	1 y
1	0 sec	0
2	0.01 sec	0.0100
3	0.02 sec	0.0200
4	0.03 sec	0.0300
5	0.04 sec	0.0400
6	0.05 sec	0.0500
7	0.06 sec	0.0600
8	0.07 sec	0.0699
9	0.08 sec	0.0799
10	0.09 sec	0.0899
11	0.1 sec	0.0998
12	0.11 sec	0.1098
13	0.12 sec	0.1197
14	0.13 sec	0.1296
15	0.14 sec	0.1395
16	0.15 sec	0.1494
17	0.16 sec	0.1593
18	0.17 sec	0.1692
19	0.18 sec	0.1790
20	0.19 sec	0.1889
21	0.2 sec	0.1987



Signal Analyzer (ii)

Signal Analyzer





Signal Analyzer (iii)

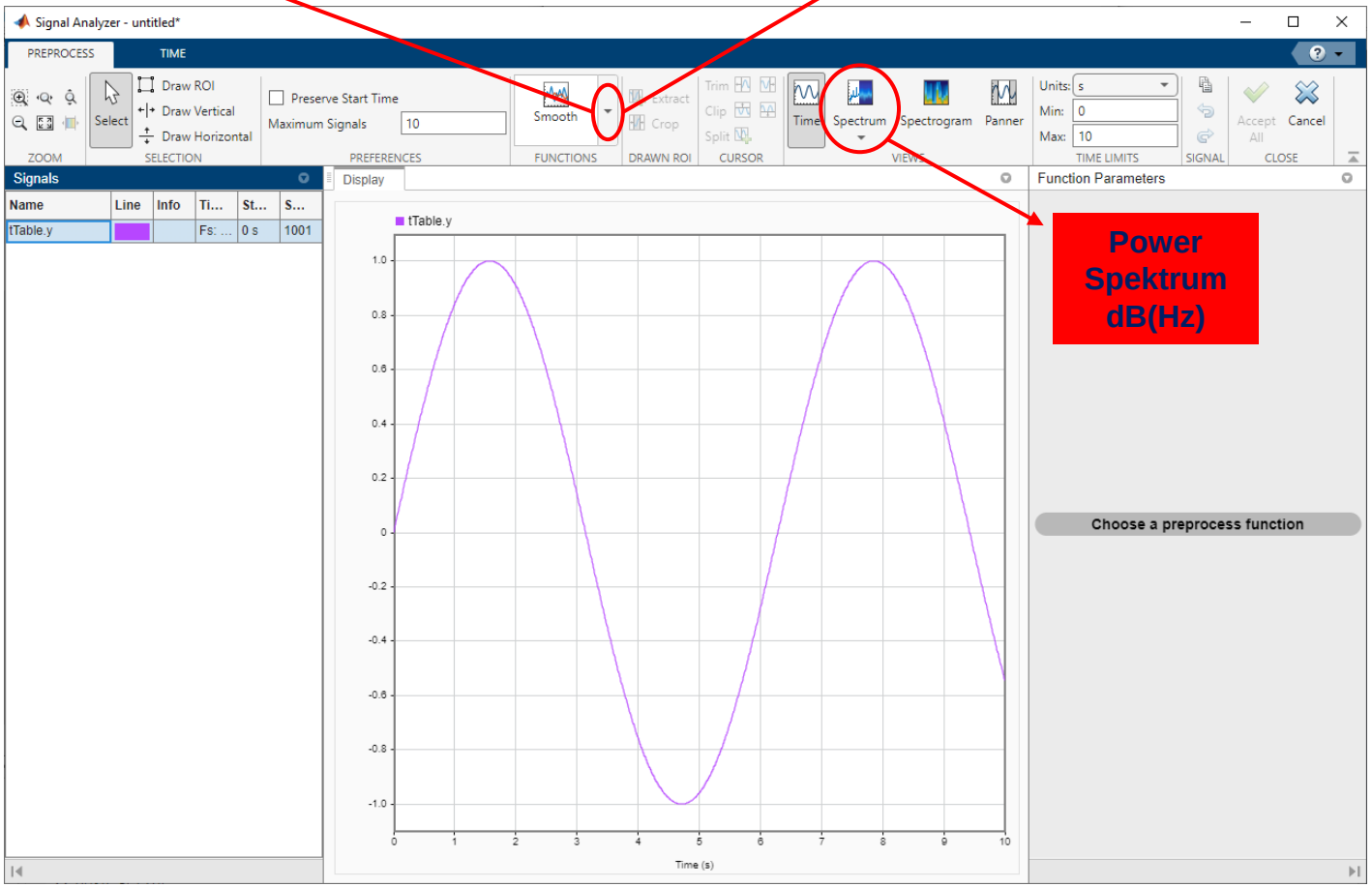
Signal Analyzer

Resample
Change sample rate when uniform or interpolate when nonuniform

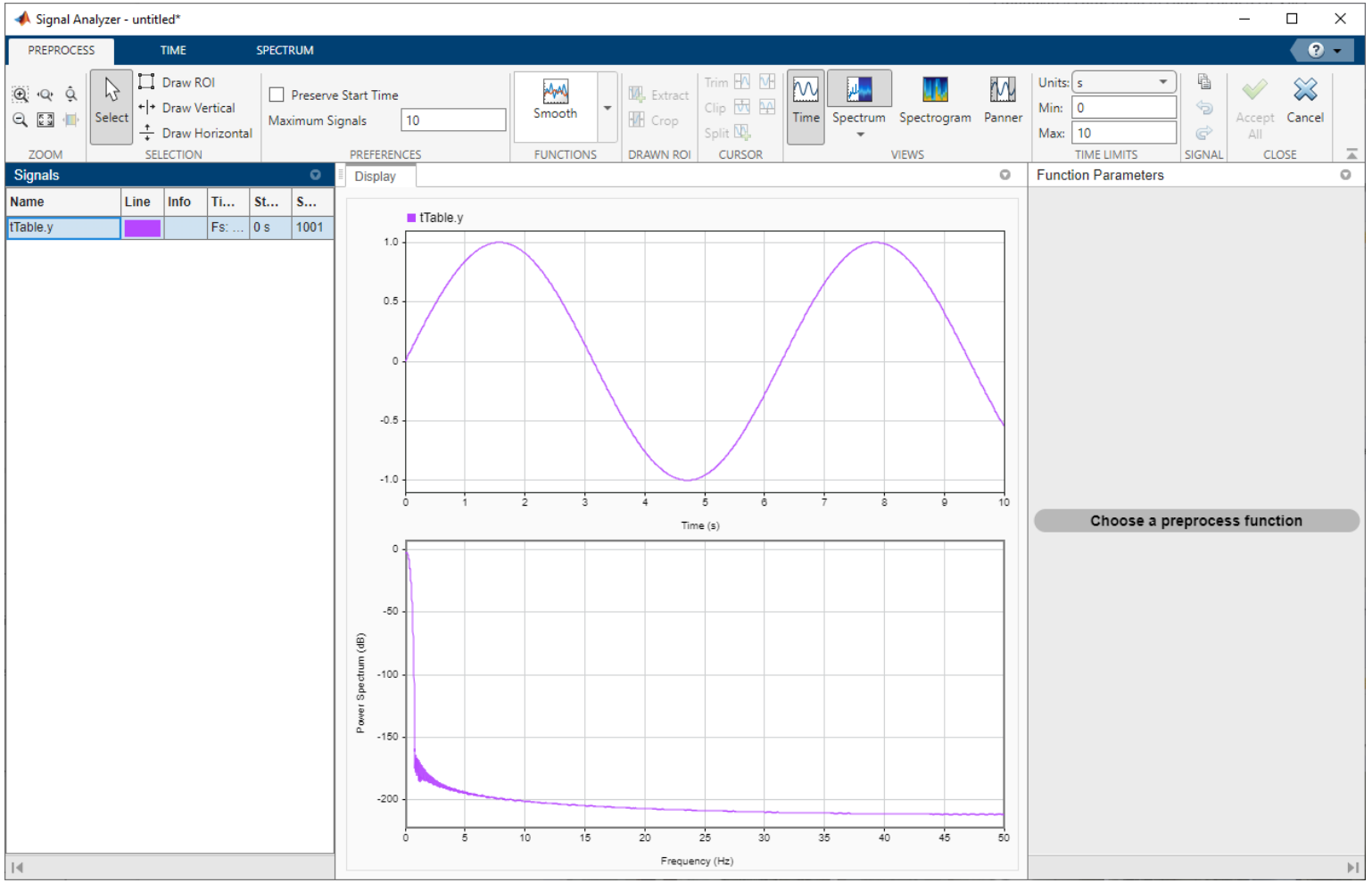
required if data type not consistent (e.g. single vs. double)

Detrend
Remove constant, linear, or piecewise linear trends

required for e.g. offset cleanup



Signal Analyzer (iv)



Signal Analyzer (v)

Signal Analyzer

Example 1: Resample Function

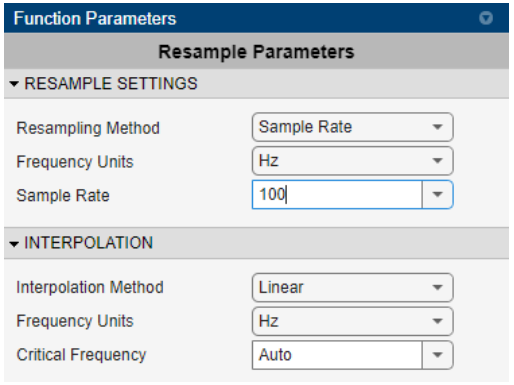
Resample
Change sample rate when uniform or interpolate when nonuniform

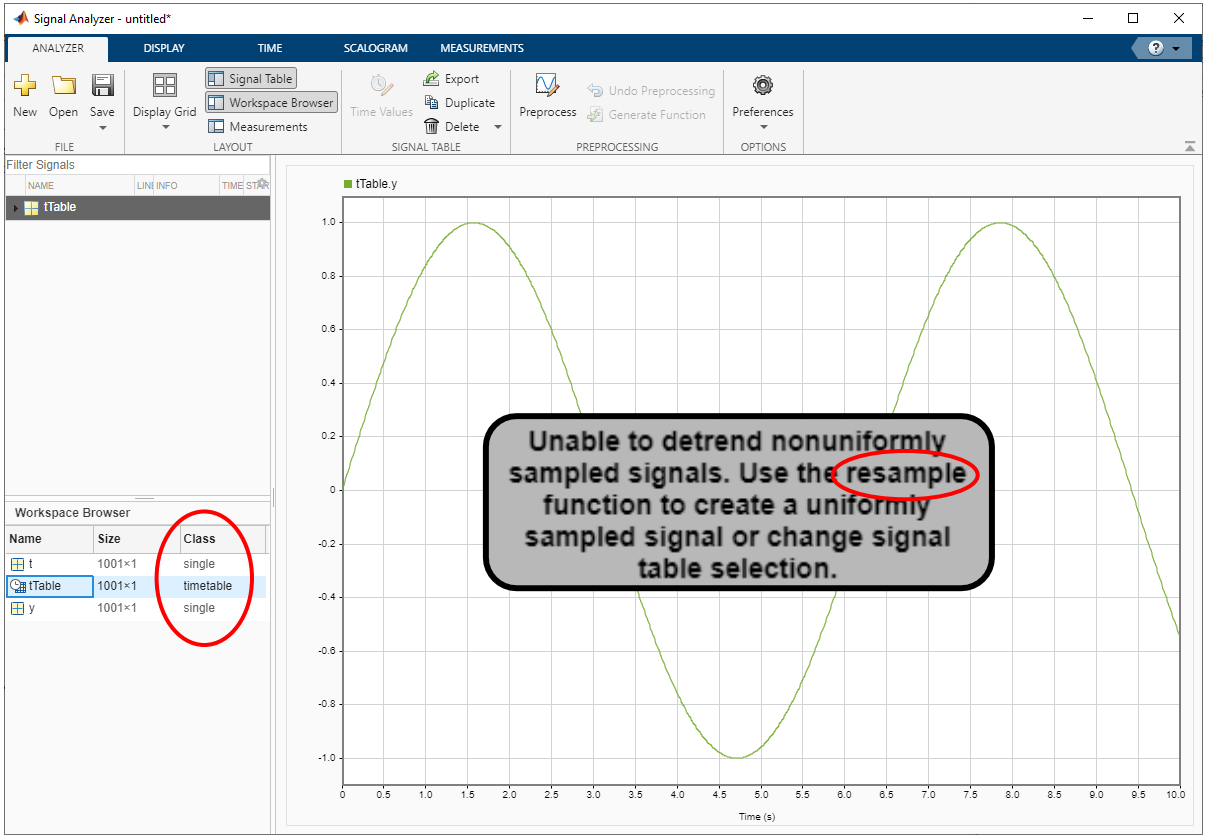
MATLAB function: resample()

generating realistic signal (incl. data type(s) unequal double)

```
t = single(0:0.01:10)';
y = sin(t);

% prepare for SignalAnalyzer
x = seconds(t);
tTable = timetable(x,y);
```





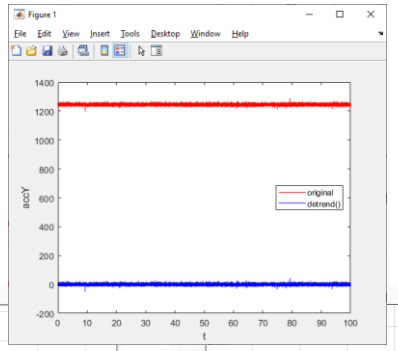
Signal Analyzer (vi)

Signal Analyzer

Example 2: Detrend Function

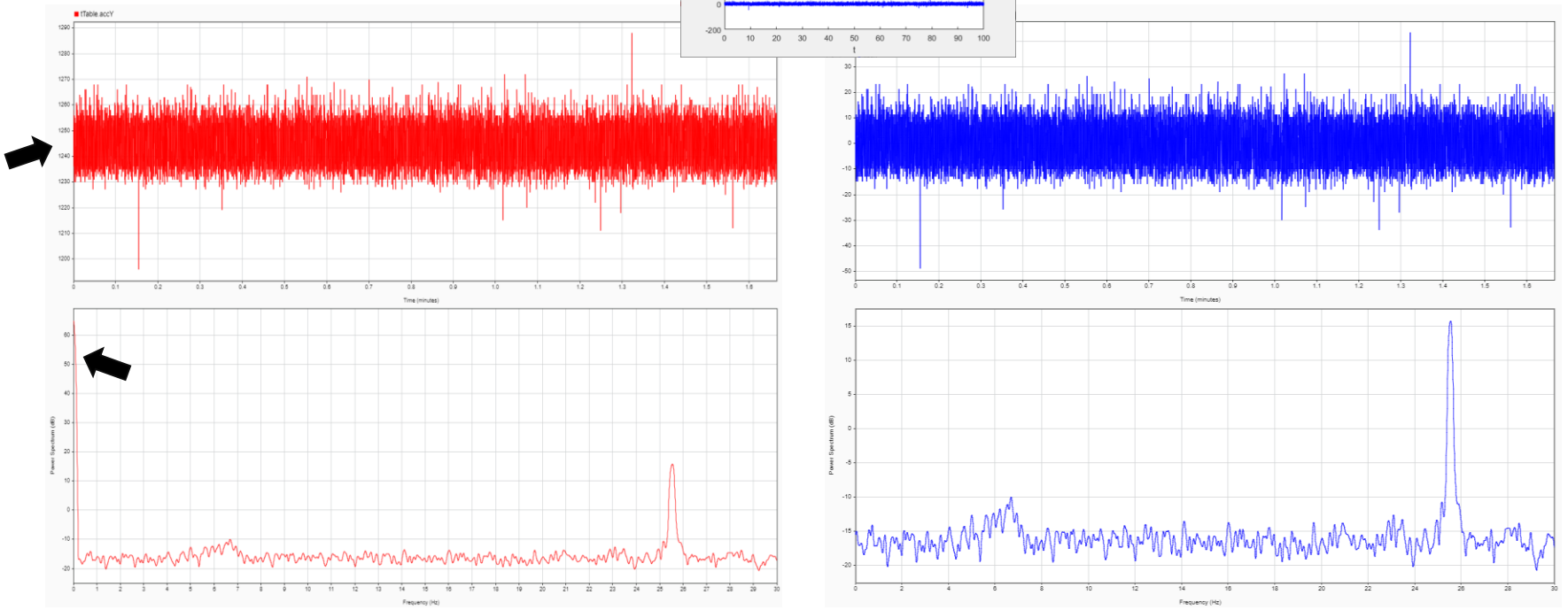
Detrend
Remove constant, linear, or piecewise linear trends

MATLAB function: detrend()



original signal (incl. an offset)

detrended signal



Diagnostic Feature Designer (i)

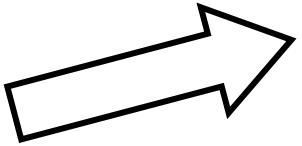
input format (table of timetables)

each row represents one measurement

dataTrainEnsemble

20x4 table

	1 accX	2 accY	3 accZ	4 mode
1	1001x1 time...	1001x1 time...	1001x1 time...	4
2	1001x1 time...	1001x1 time...	1001x1 time...	3
3	1001x1 time...	1001x1 time...	1001x1 time...	1
4	1001x1 time...	1001x1 time...	1001x1 time...	2
5	1001x1 time...	1001x1 time...	1001x1 time...	2
6	1001x1 time...	1001x1 time...	1001x1 time...	1
7	1001x1 time...	1001x1 time...	1001x1 time...	3
8	1001x1 time...	1001x1 time...	1001x1 time...	4
9	1001x1 time...	1001x1 time...	1001x1 time...	4
10	1001x1 time...	1001x1 time...	1001x1 time...	3
11	1001x1 time...	1001x1 time...	1001x1 time...	1
12	1001x1 time...	1001x1 time...	1001x1 time...	2
13	1001x1 time...	1001x1 time...	1001x1 time...	2
14	1001x1 time...	1001x1 time...	1001x1 time...	1
15	1001x1 time...	1001x1 time...	1001x1 time...	3
16	1001x1 time...	1001x1 time...	1001x1 time...	4
17	1001x1 time...	1001x1 time...	1001x1 time...	4
18	1001x1 time...	1001x1 time...	1001x1 time...	3
19	1001x1 time...	1001x1 time...	1001x1 time...	1
20	1001x1 time...	1001x1 time...	1001x1 time...	2

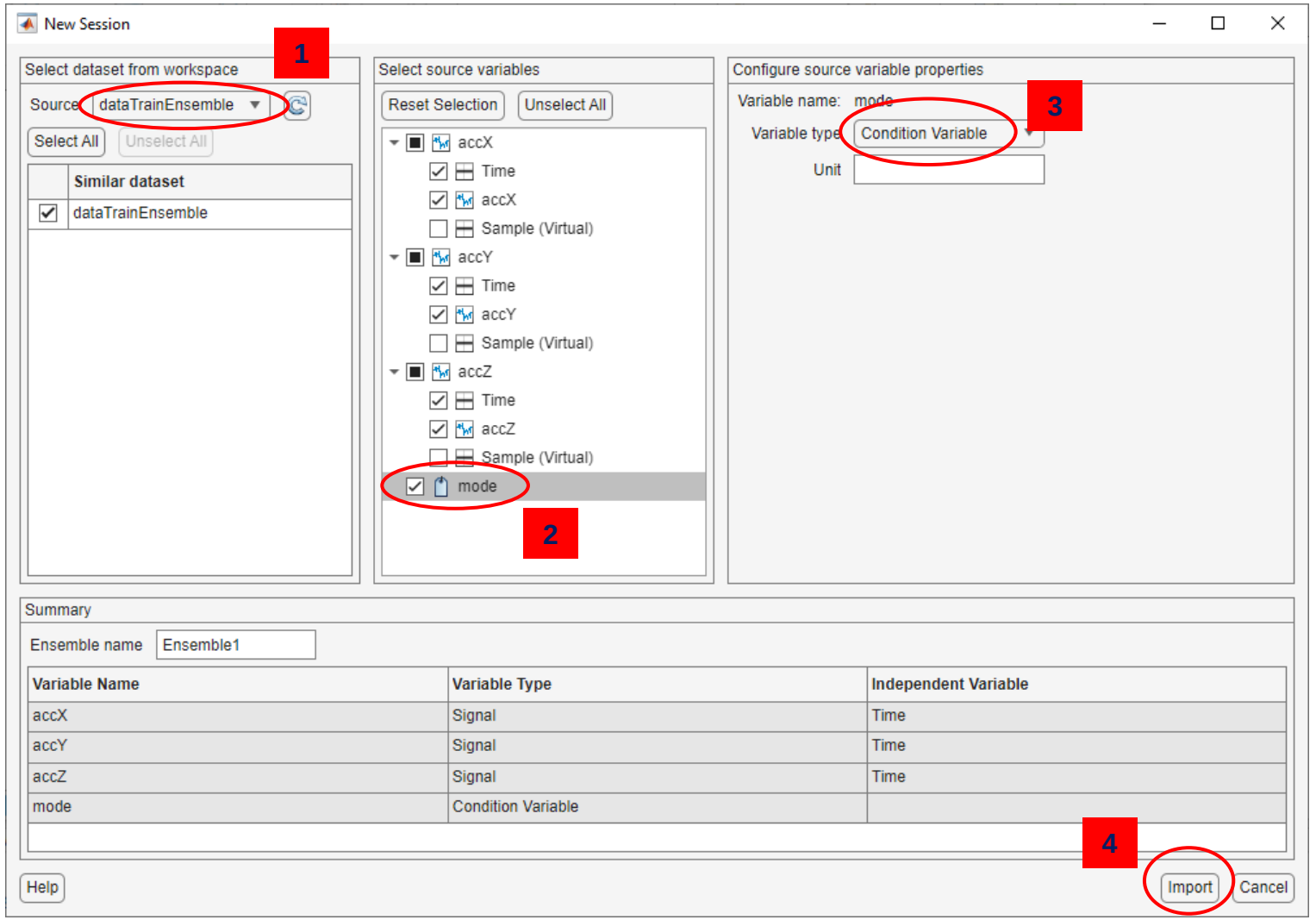


dataTrainEnsemble | dataTrainEnsemble.accY{7,1}

dataTrainEnsemble.accY{7,1}

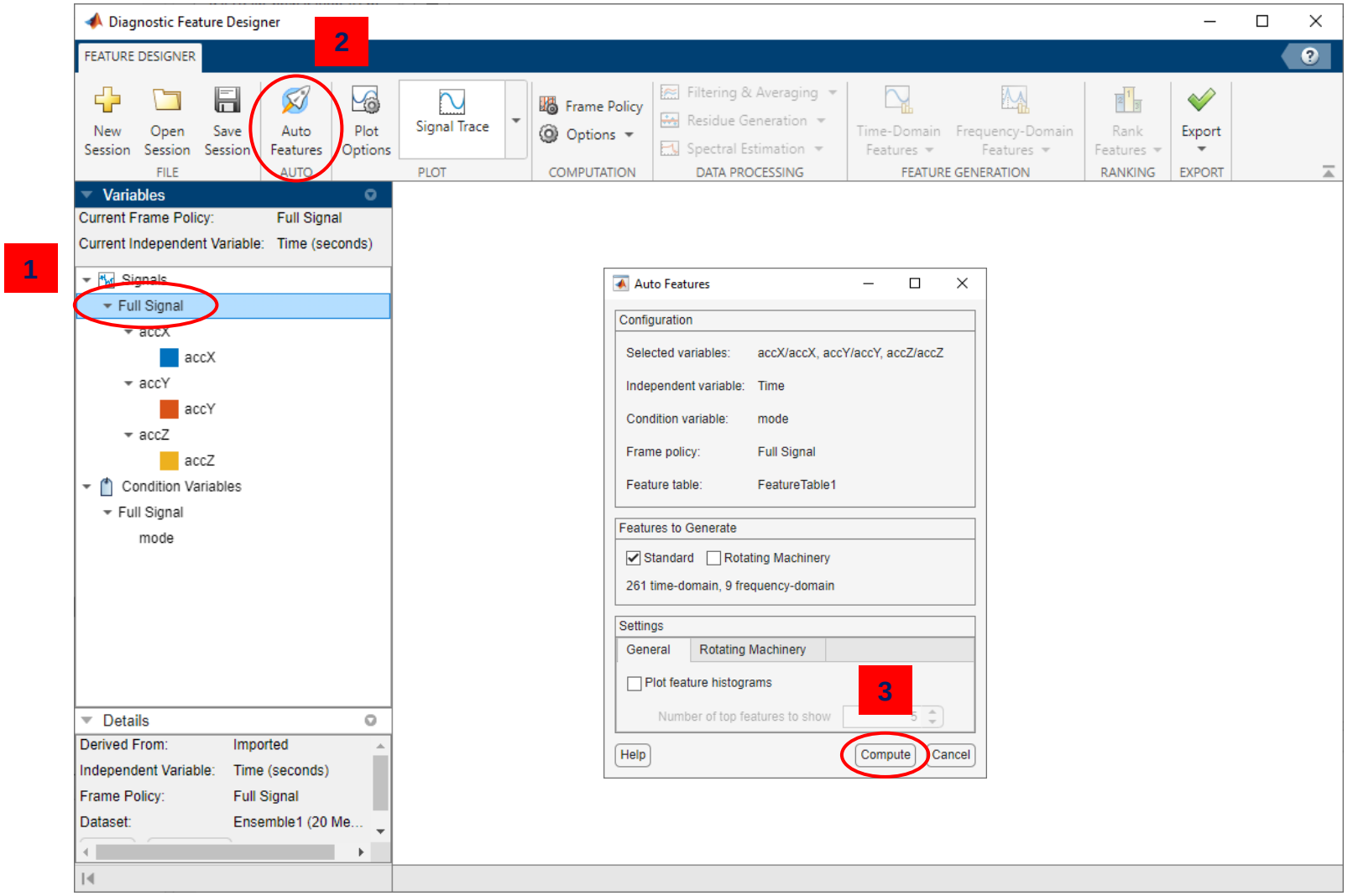
	Time	1 accY	2
1	0 sec	-6.7157	
2	0.01 sec	3.2842	
3	0.02 sec	0.2842	
4	0.03 sec	0.2842	
5	0.04 sec	3.2841	
6	0.05 sec	-3.7159	
7	0.06 sec	4.2840	
8	0.07 sec	0.2840	
9	0.08 sec	0.2839	
10	0.09 sec	0.2839	
11	0.1 sec	2.2839	
12	0.11 sec	1.2838	

Diagnostic Feature Designer (ii)

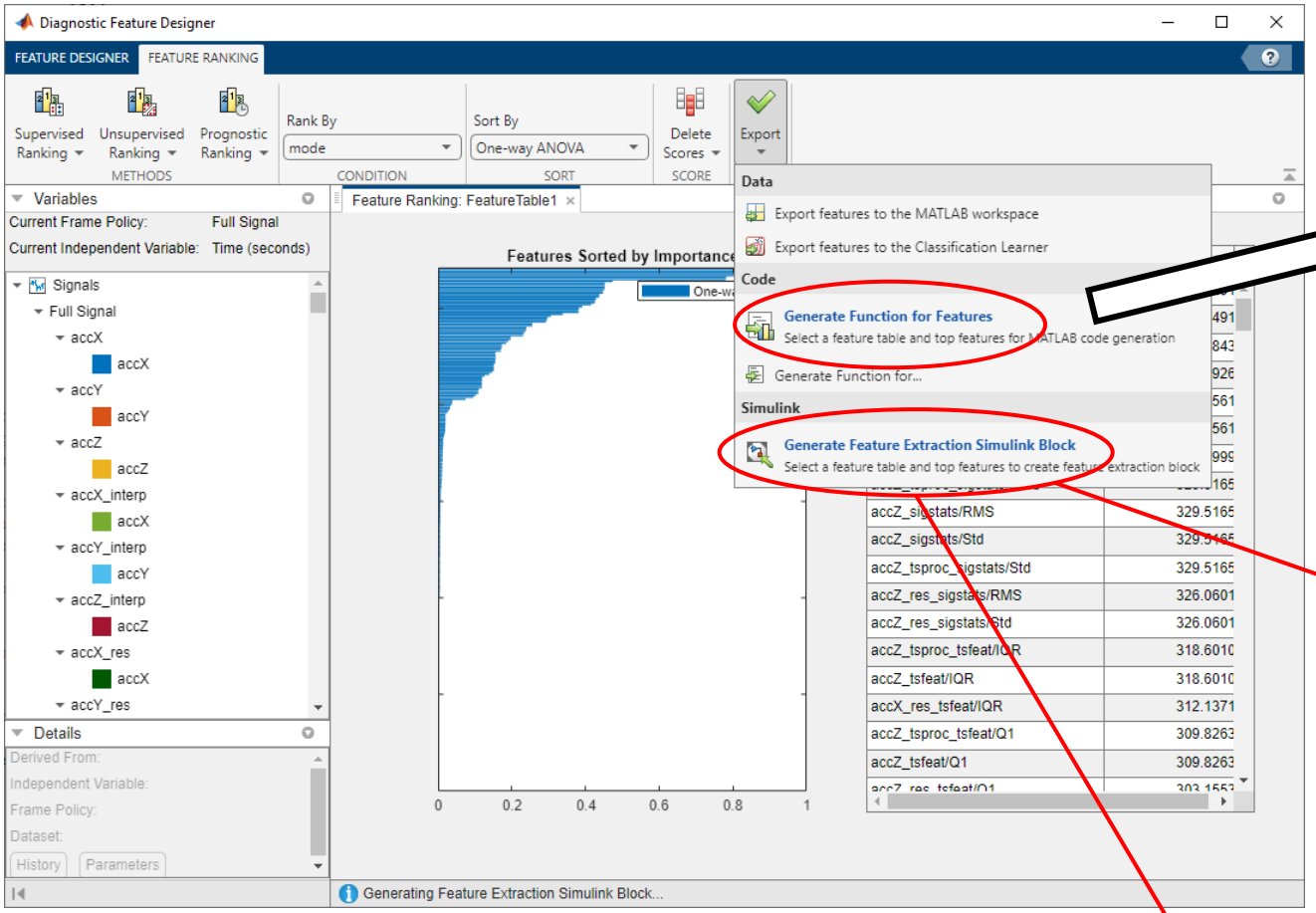


Diagnostic Feature Designer (iii)

Diagnostic Feature Desi...



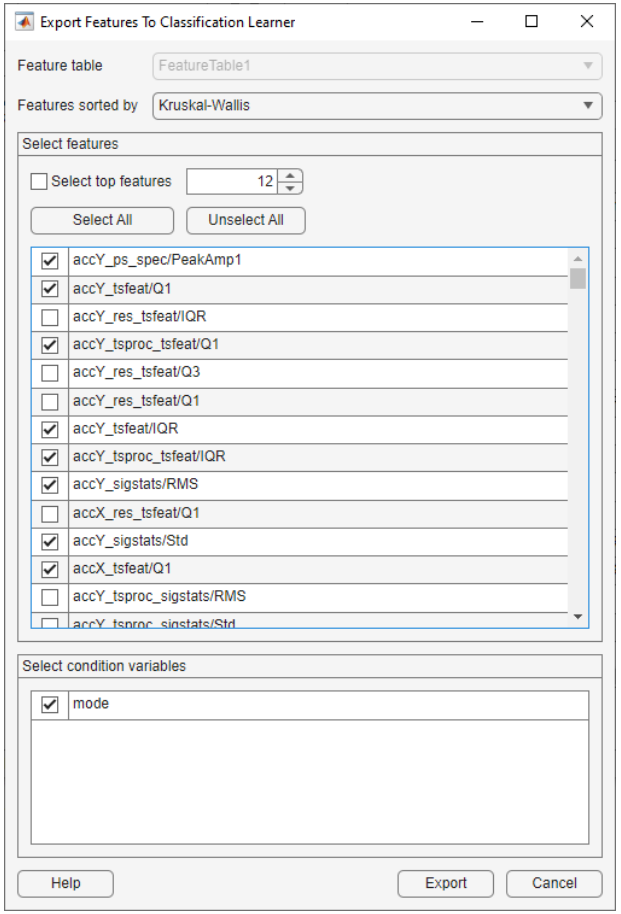
Diagnostic Feature Designer (iv)



Classification Learner (i)

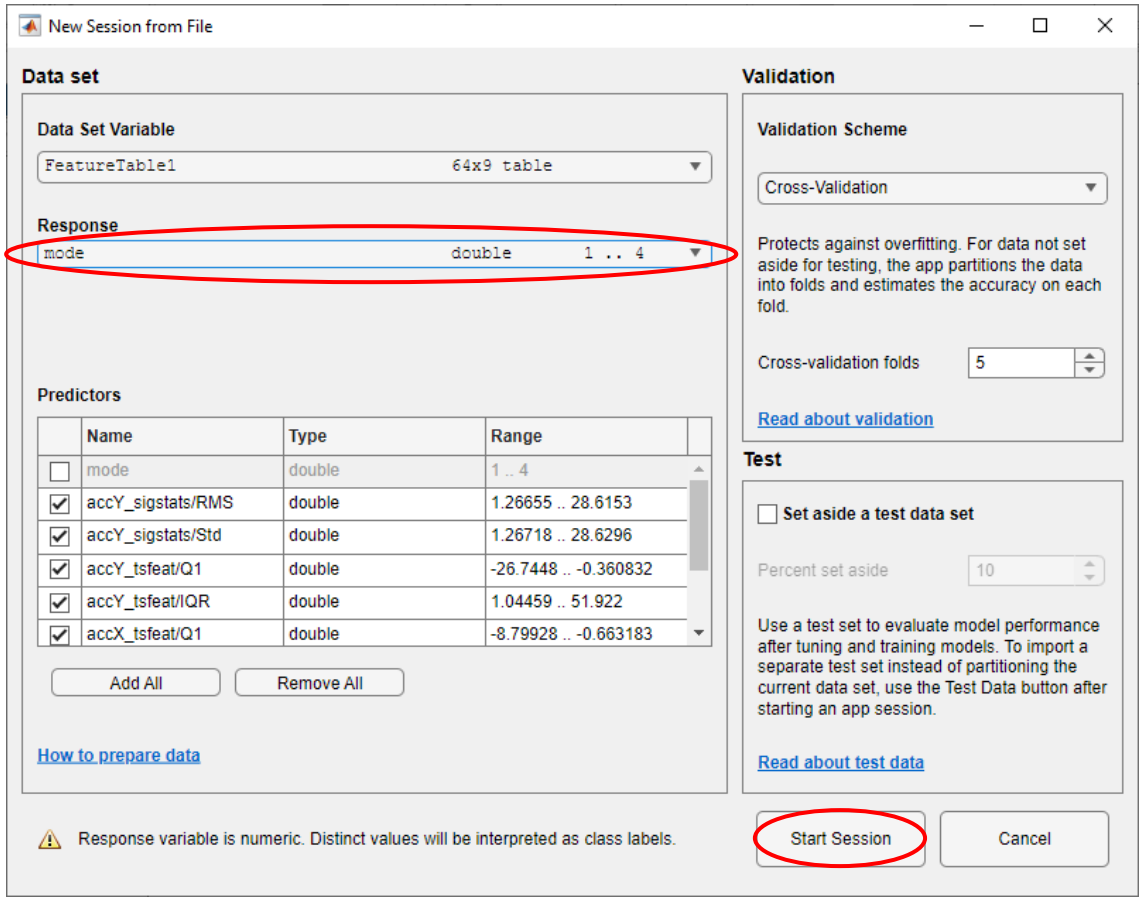
Classification Learner

```
diagnosticFeatures.m
18  % This function computes features:
19  %  accY_sigstats/RMS
20  %  accY_sigstats/Std
21  %  accY_tsfeat/Q1
22  %  accY_tsfeat/IQR
23  %  accX_tsfeat/Q1
24  %  accY_tsproc_tsfeat/Q1
25  %  accY_tsproc_tsfeat/IQR
26  %  accY_ps_spec/PeakAmp1
```

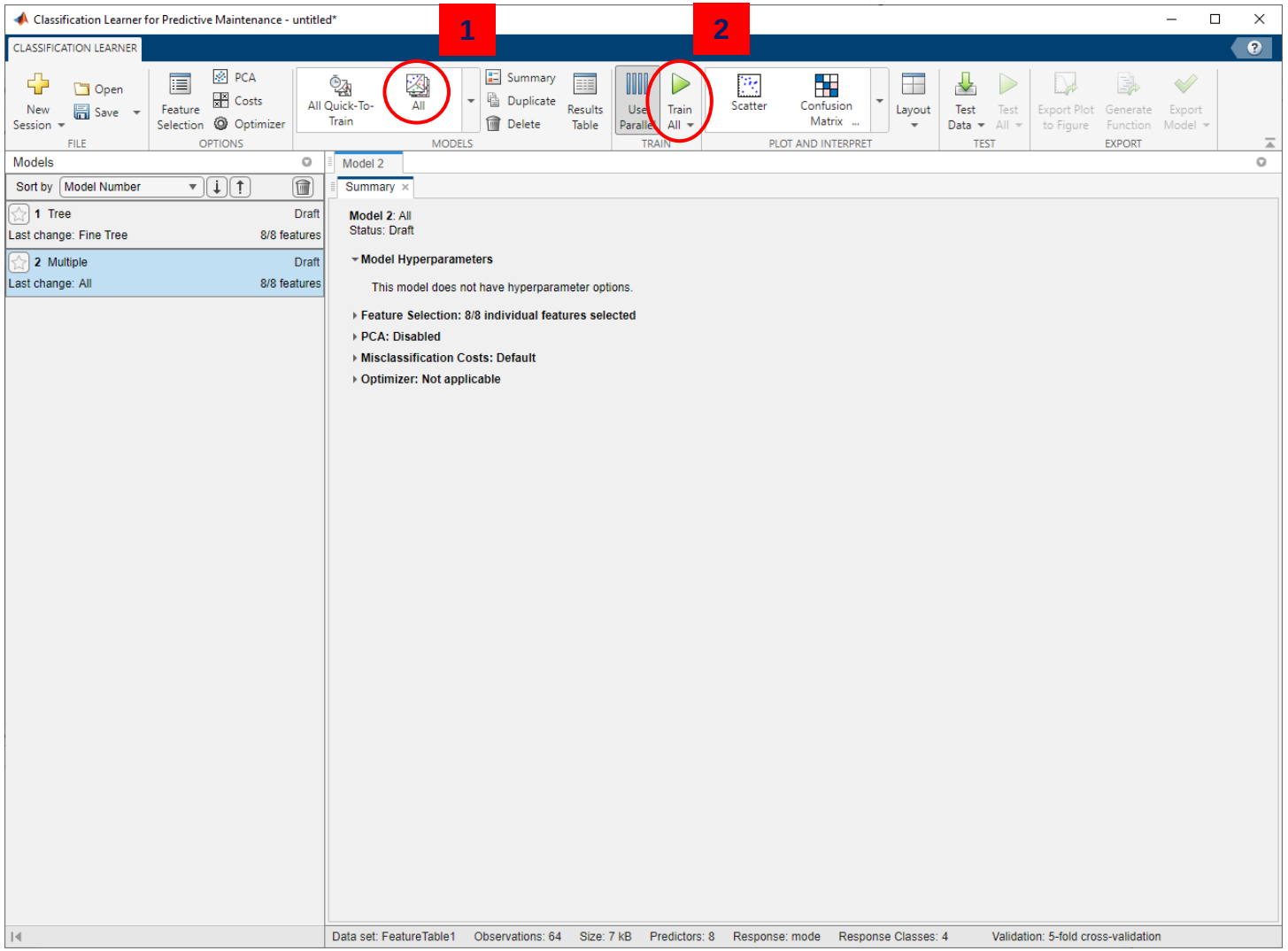


Classification Learner (ii)

Classification Learner



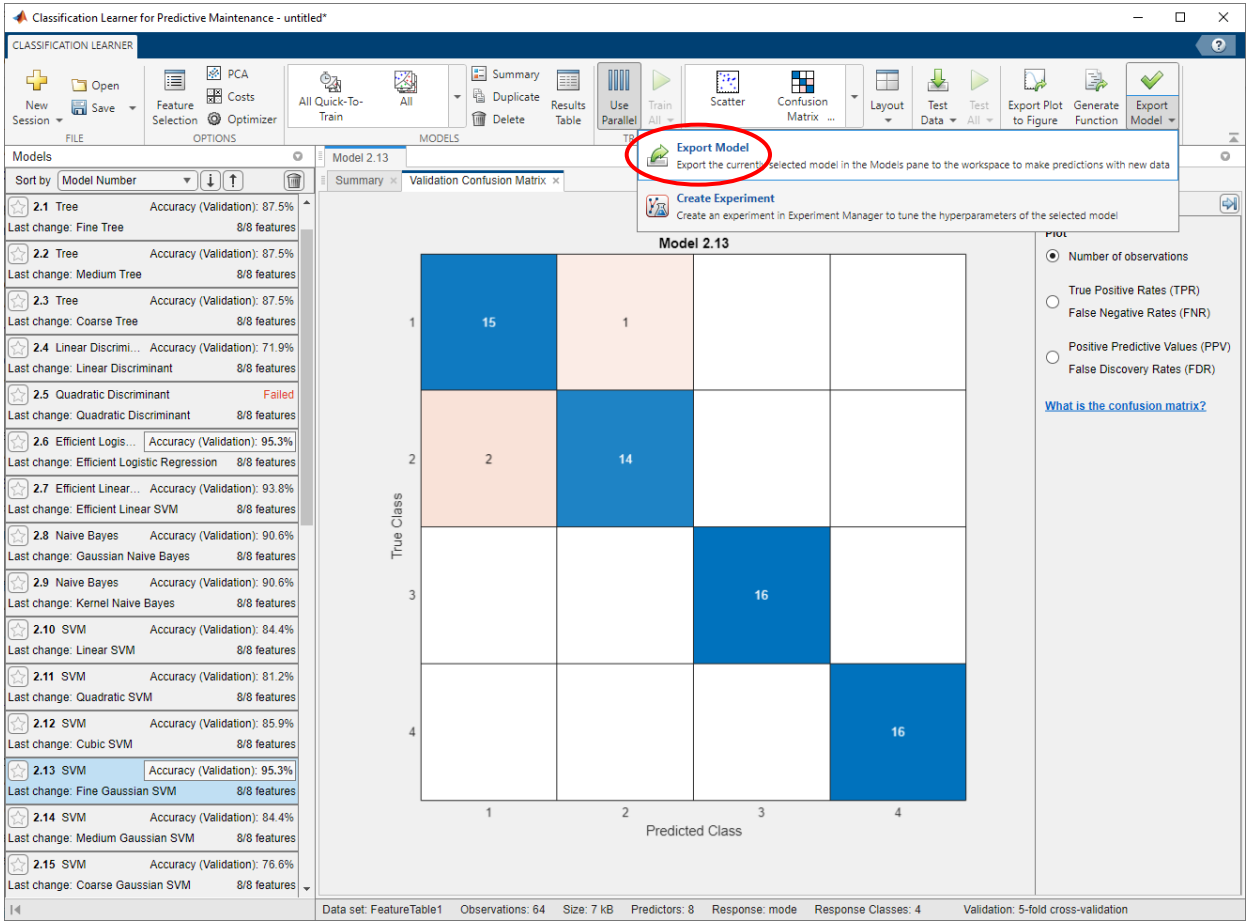
Classification Learner (iii)



Classification Learner (iv)

Classification Learner

chose a model and export it to MATLAB workspace



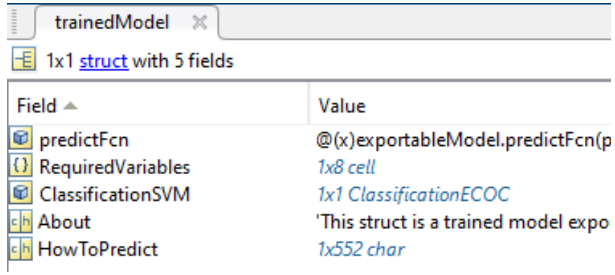
Classification Learner (v)

Classification Learner

commands to use trained models in MATLAB/Simulink:

a) **saveLearnerForCoder(trainedModel.Classification***, 'trainedModel');**

e.g. saveLearnerForCoder(trainedModel.ClassificationSVM, 'trainedModel');

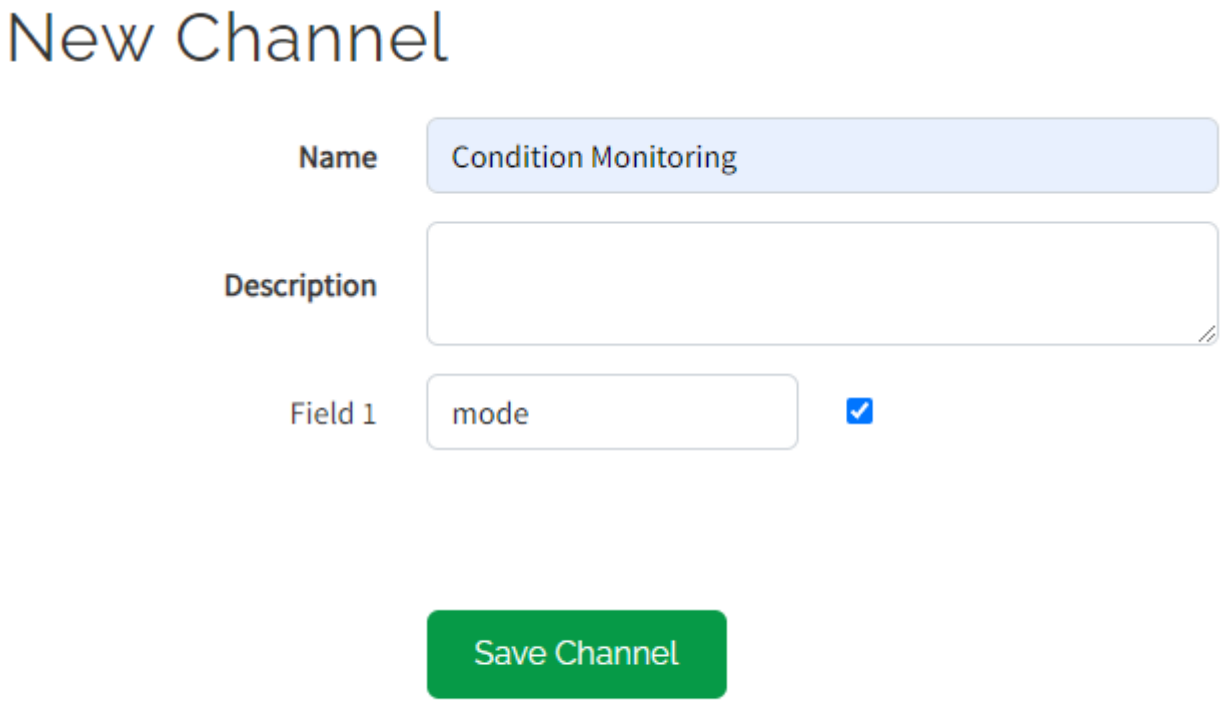


b) **classifier = loadLearnerForCoder("trainedModel.mat");**

c) **mode = predict(classifier, featureTable);**

ThingSpeak (i)

Sign in to ThingSpeak using your MathWorks Account → New Channel



ThingSpeak (ii)

Condition Monitoring

Channel ID: **2485082**
Author: mwa0000032484217
Access: Private

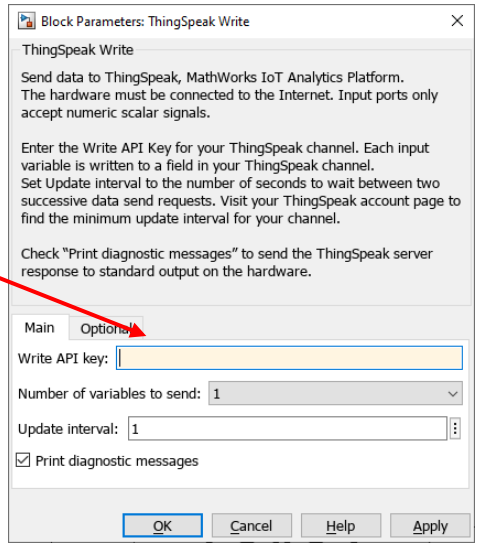
Private View | Public View | Channel Settings | Sharing | API Keys | Data Import

Write API Key

Key: 1SY126GB2QX4EQYD

Generate New Write API Key

copy Write API Key

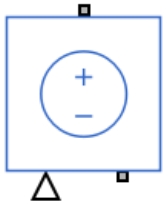


Simscape – Add-Ons (MATLAB R2024a)

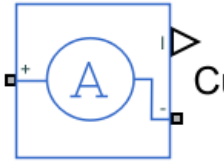
Add-Ons	Requires
Simscape version 24.1	Simulink version 24.1
Simscape Driveline version 24.1	Simulink version 24.1 Simscape version 24.1
Simscape Electrical version 24.1	Simulink version 24.1 Simscape version 24.1

University of Applied Sciences of Upper Austria, Wels

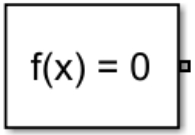
Simscape – Electrical Components



Controlled Voltage Source



Current Sensor

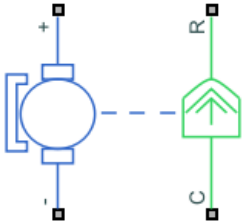


Solver Configuration

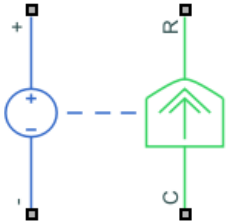
Resistor

Inductor

Electrical Reference

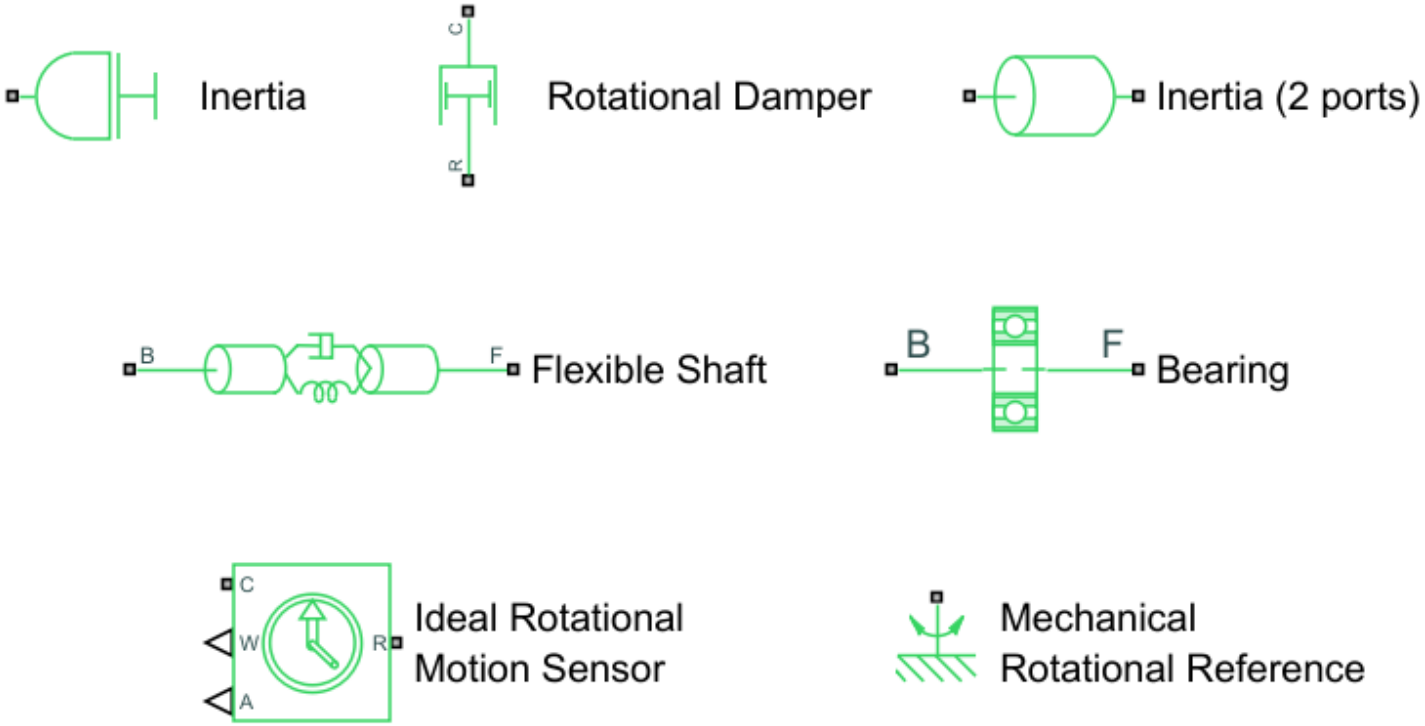


DC Motor

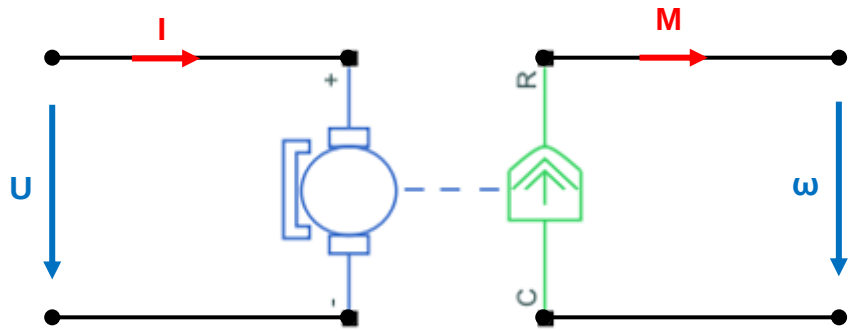


Rotational Electromechanical Converter

Simscape – Mechanical Components



Simscape – Interfaces



Simulink-PS Converter

PS-Simulink Converter