



AI Based Fault detection on Industrial Controllers

18. September 2025

Dr.-Ing. Rainer Mümmler

Application Engineer

rmuemmler@mathworks.com



Conrado Ramirez Garcia

Application Engineer

cramirez@mathworks.com





Monitor Processes

- Improve asset availability
- Monitor energy use
- Manage inventory



Improve Quality

- Diagnose defects on production line
- Optimize yield

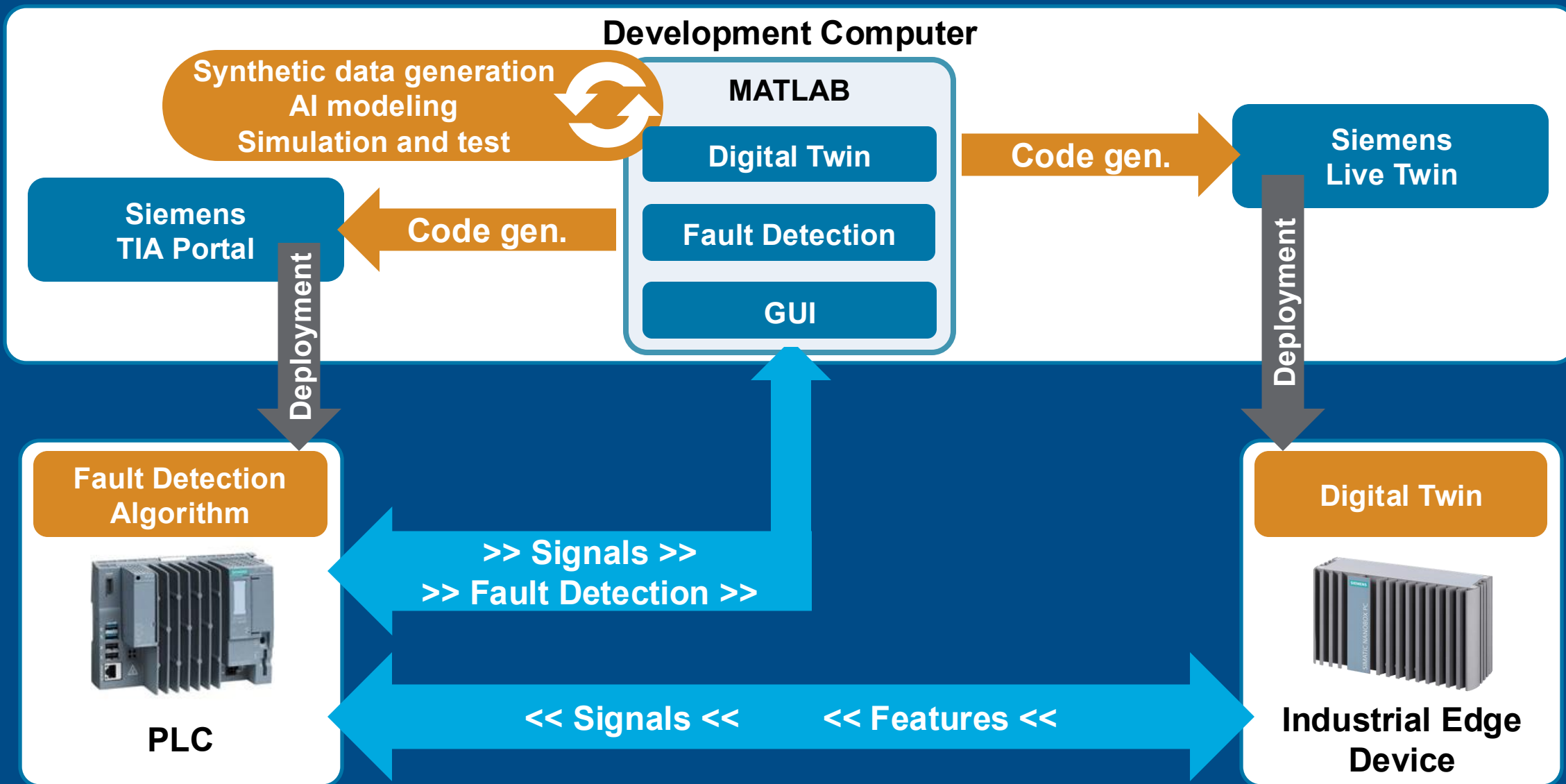


Schedule Maintenance

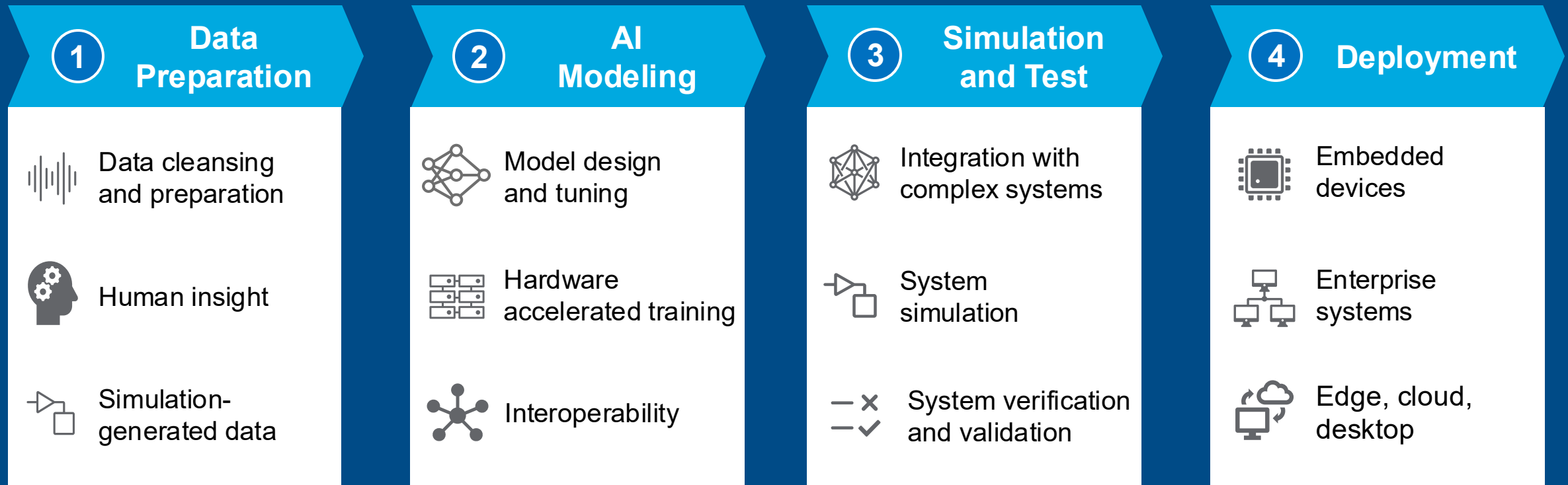
- Predict faults in equipment
- Decrease downtime

Why do companies care about anomaly detection for industrial processes and machinery?





Demonstrator setup



End-to-End Workflows for Artificial Intelligence
Software Development



1

Data
Preparation

2

AI
Modeling

3

Simulation
and Test

4

Deployment

MATLAB Help Center Community Learning

Documentation Examples Functions Apps Videos Answers

Industrial Cooling Fan Anomaly Detection Algorithm Development for Deployment to a Microservice Docker Image R2024b

Detection of anomalies from sensor measurements is an important part of an equipment condition monitoring process for preventing catastrophic failure. This example shows the first part of a two-part end-to-end workflow for creating a predictive maintenance application from the development of a machine learning algorithm that can ultimately be deployed as a trained machine learning model in a Docker® microservice.

The example uses a Simulink® model to synthetically create measurement data for both normal and anomalous conditions. A support vector machine (SVM) model is trained for detecting anomalies in load, fan mechanics, and power consumption of an industrial cooling fan. The model detects a load anomaly when the system is working on overloaded conditions, that is, when the system work demand exceeds designed limits. A fan anomaly is detected when a fault occurs within the mechanical subsystem of the motor or the fan. Finally, a power supply anomaly can be detected by a drop in the voltage.

The companion example, [Deploy Industrial Cooling Fan Anomaly Detection Algorithm as Microservice](#) (MATLAB Compiler SDK), shows the actual deployment of the model in a Docker environment.

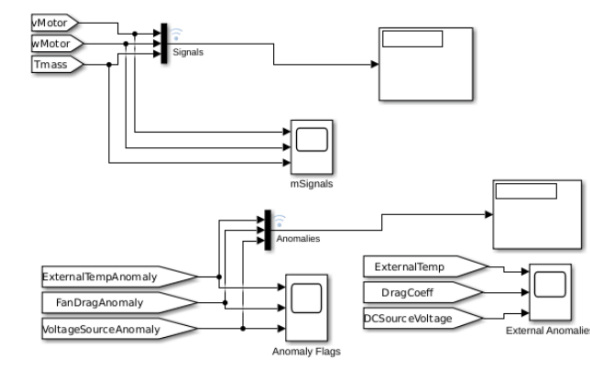
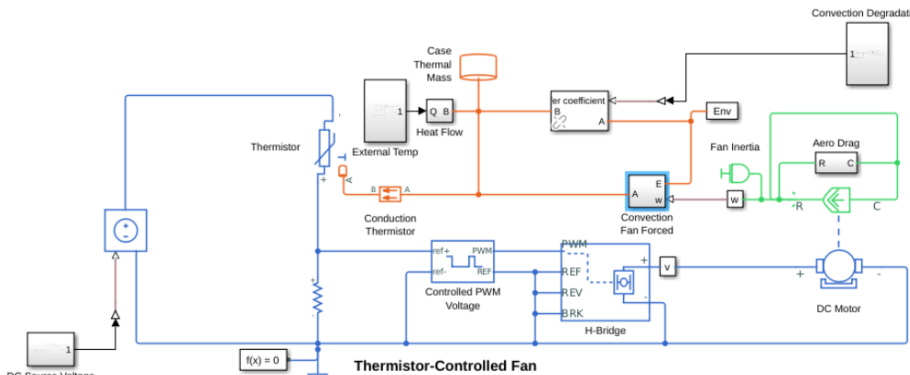
Data Generation

In this example, a thermistor-controlled fan model defined using Simscape™ Electrical™ blocks, generates the measurements. This Simulink model includes thermal, mechanical and electrical components of a fan. You can learn more about this model in [Thermistor-Controlled Fan](#) (Simscape Electrical).

```
addpath('Data_Generator/', 'Data_Generator/VaryingConvectionLib/');  
mdl = 'CoolingFanWithFaults';  
open_system(mdl)
```

Thermistor-Controlled Fan

1. [Plot temperature](#) and other conditions in system ([see code](#))
2. [Explore simulation results](#) using [sscexplore](#)
3. [Learn more](#) about this example



<https://www.mathworks.com/help/predmaint/ug/industrial-cooling-fan-anomaly-detection-with-docker-deployment.html>

Industrial Fan Shipping Demo: Digital Twin



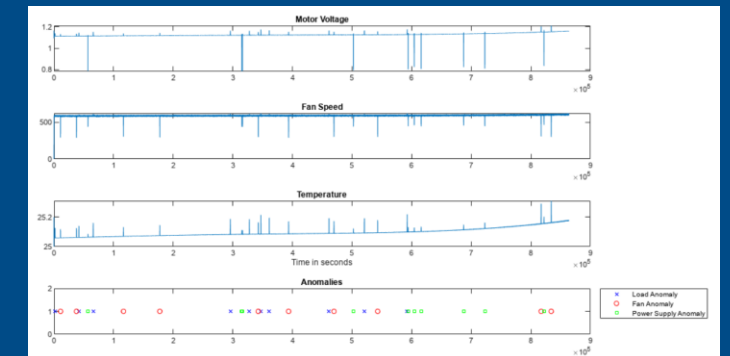
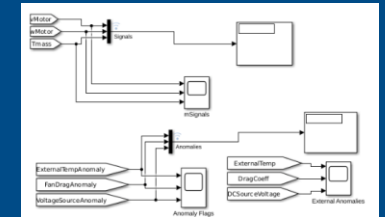
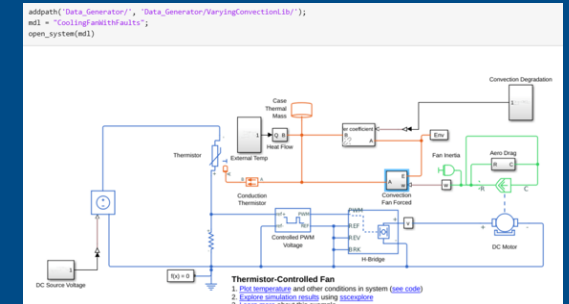
① Data Preparation

② AI Modeling

③ Simulation and Test

④ Deployment

- Simulink/Simscape model simulates fan with electrical, thermal, and mechanical components
- Anomalies injected: Load, Fan Mechanics, Power Supply
- Data collected from voltage, power, and temperature sensors



Industrial Fan Shipping Demo: Digital Twin



1

Data
Preparation

2

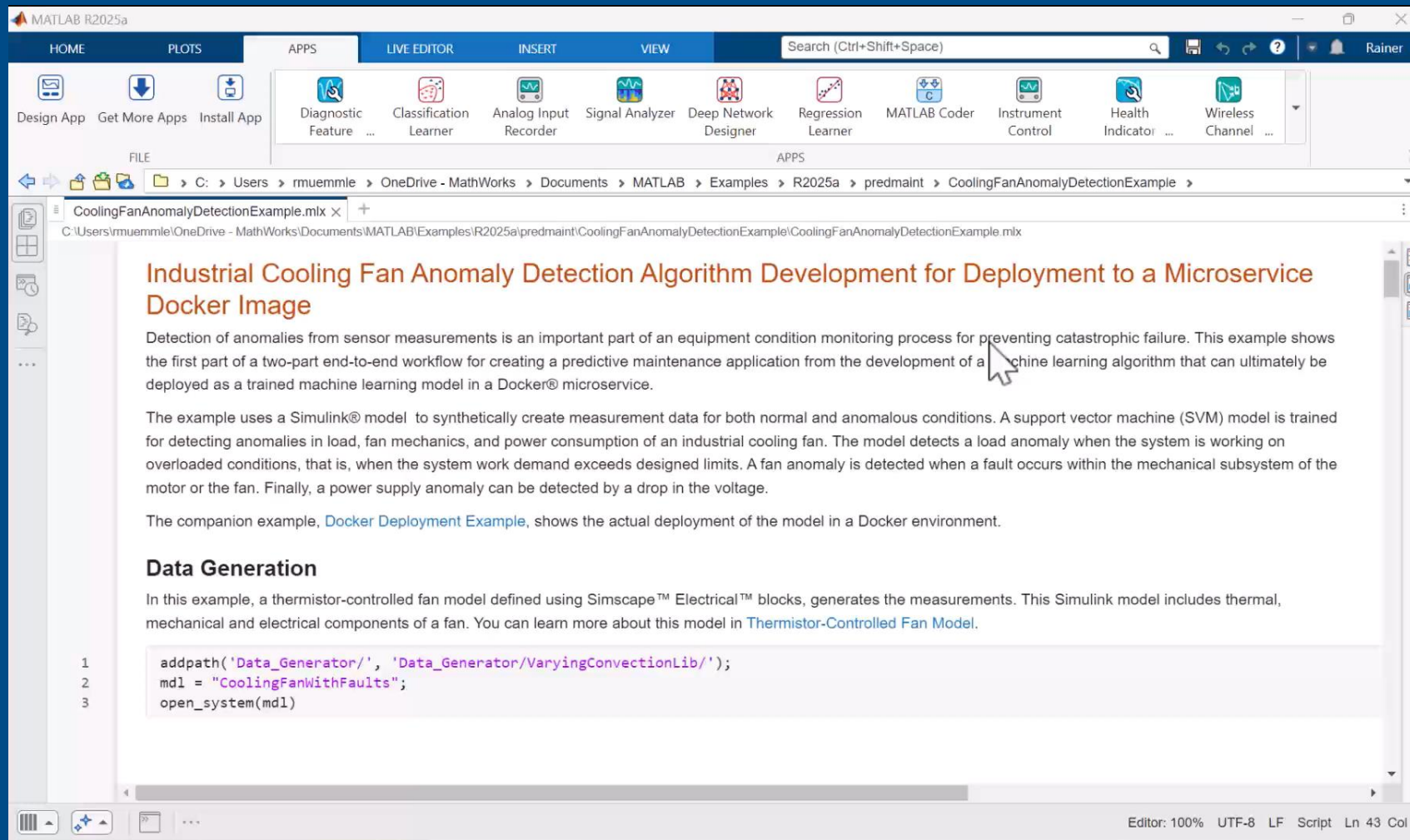
AI
Modeling

3

Simulation
and Test

4

Deployment



Data Generation and Preprocessing

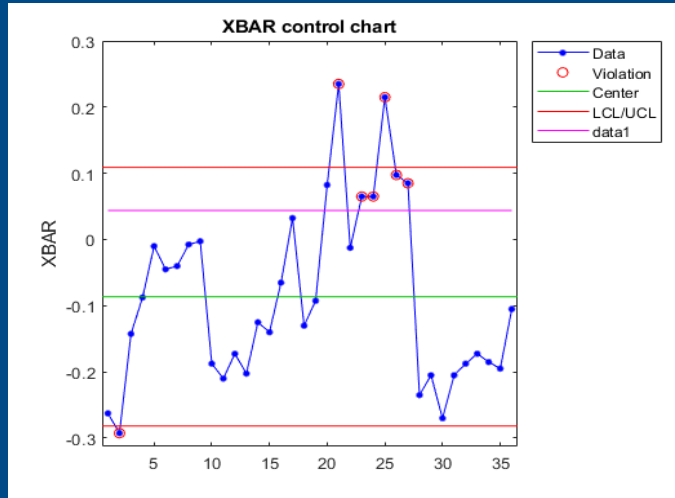


① Data Preparation

② AI Modeling

③ Simulation and Test

④ Deployment

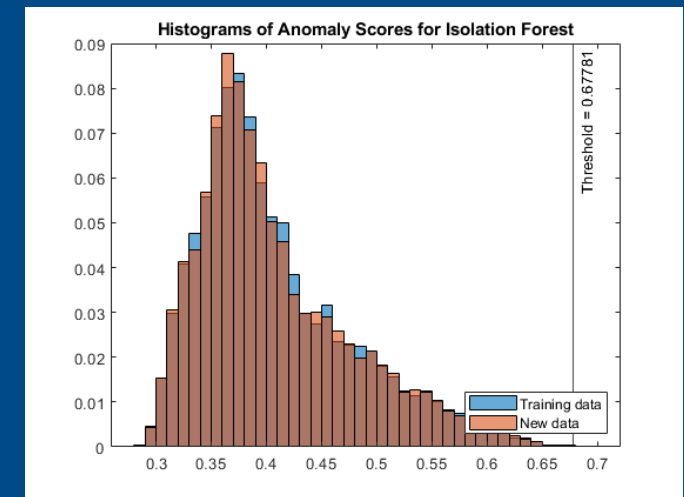


Supervised Methods

Works with plenty available data

Simple Statistics

e.g., Outlier Detector



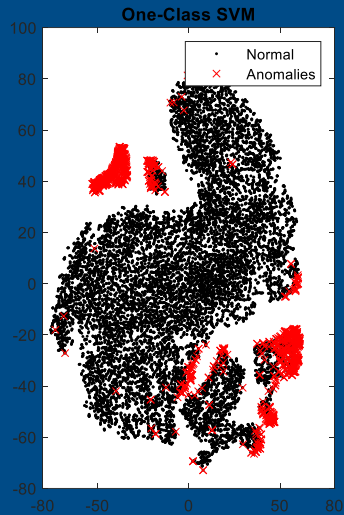
Anomaly Detection Approaches

① Data Preparation

② AI Modeling

③ Simulation and Test

④ Deployment



Unsupervised Methods

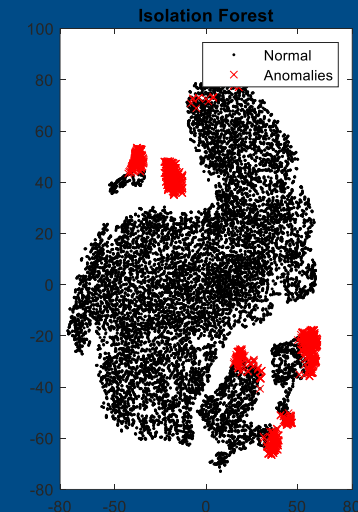
“Normal Only” data available

Supervised Methods

Works with plenty available data

Simple Statistics

e.g., Outlier Detector



Anomaly Detection Approaches

1

Data
Preparation

2

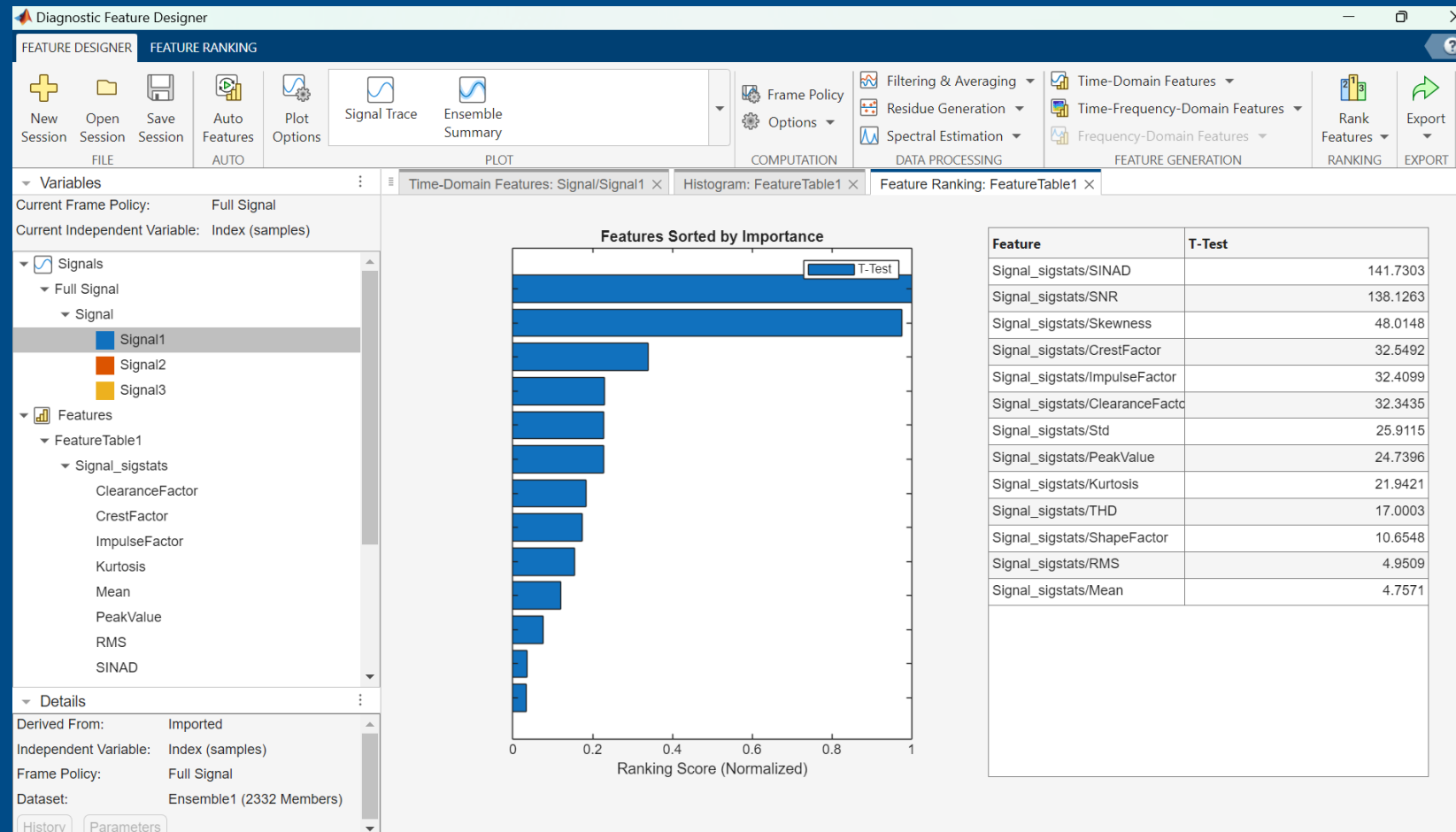
AI
Modeling

3

Simulation
and Test

4

Deployment



Feature Extraction / Fault Classification



1

Data
Preparation

2

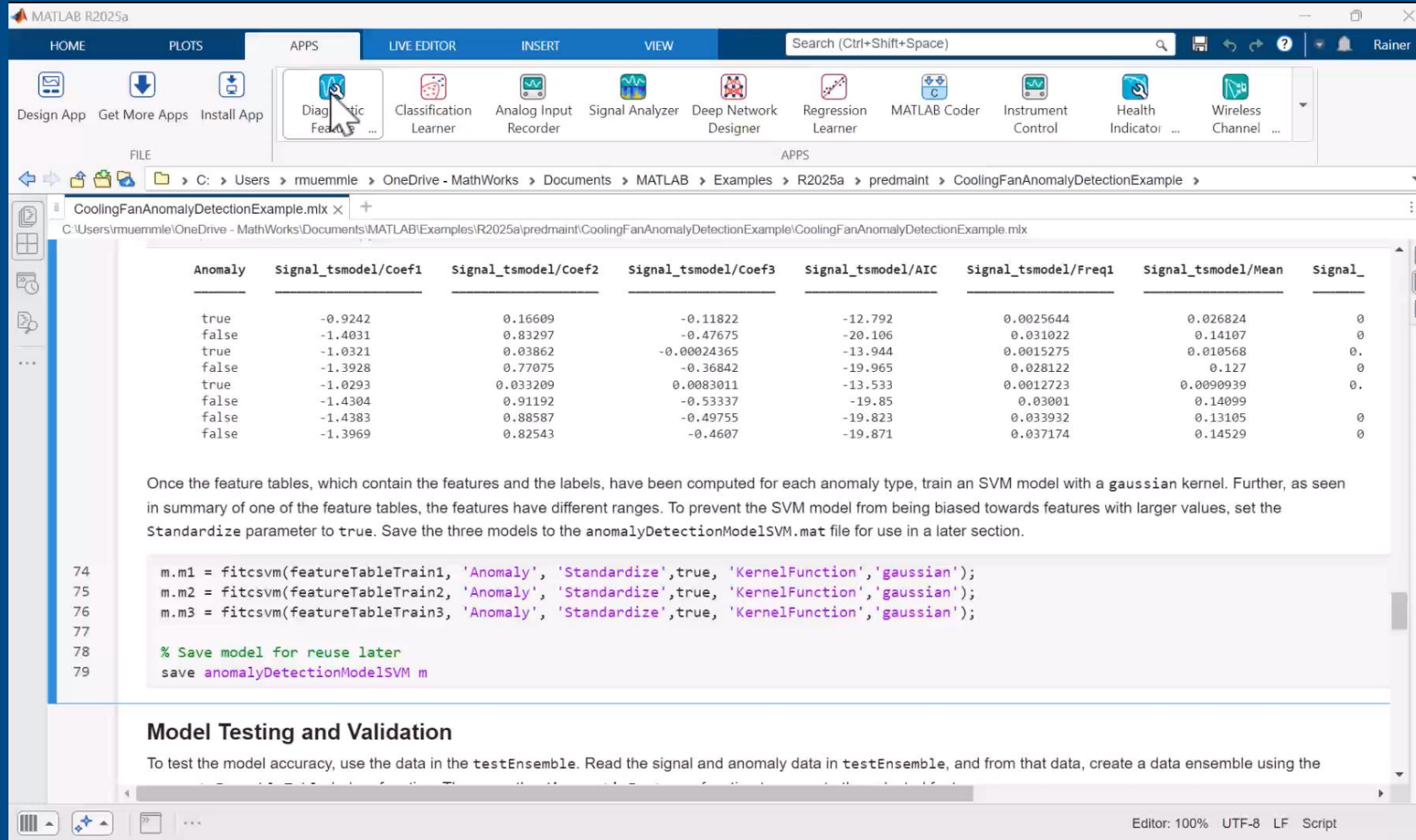
AI
Modeling

3

Simulation
and Test

4

Deployment



Once the feature tables, which contain the features and the labels, have been computed for each anomaly type, train an SVM model with a gaussian kernel. Further, as seen in summary of one of the feature tables, the features have different ranges. To prevent the SVM model from being biased towards features with larger values, set the Standardize parameter to true. Save the three models to the anomalyDetectionModelSVM.mat file for use in a later section.

```
74 m.m1 = fitcsvm(featureTableTrain1, 'Anomaly', 'Standardize',true, 'KernelFunction','gaussian');
75 m.m2 = fitcsvm(featureTableTrain2, 'Anomaly', 'Standardize',true, 'KernelFunction','gaussian');
76 m.m3 = fitcsvm(featureTableTrain3, 'Anomaly', 'Standardize',true, 'KernelFunction','gaussian');
77
78 % Save model for reuse later
79 save anomalyDetectionModelSVM m
```

Model Testing and Validation

To test the model accuracy, use the data in the testEnsemble. Read the signal and anomaly data in testEnsemble, and from that data, create a data ensemble using the

Feature Extraction using Diagnostic Feature Designer App



1

Data
Preparation

2

AI
Modeling

3

Simulation
and Test

4

Deployment

The screenshot shows the MATLAB R2025a interface. The 'LIVE EDITOR' tab is active, displaying a table of feature data and code for training SVM models.

Anomaly	Signal_tsmodel/coef1	Signal_tsmodel/Coef2	Signal_tsmodel/Coef3	Signal_tsmodel/AIC	Signal_tsmodel/Freq1	Signal_tsmodel/Mean	Signal_
true	-0.9242	0.16609	-0.11822	-12.792	0.0025644	0.026824	0
false	-1.4031	0.83297	-0.47675	-20.106	0.031022	0.14107	0
true	-1.0321	0.03862	-0.00024365	-13.944	0.0015275	0.010568	0.
false	-1.3928	0.77075	-0.36842	-19.965	0.028122	0.127	0
true	-1.0293	0.033209	0.0083011	-13.533	0.0012723	0.0090939	0.
false	-1.4304	0.91192	-0.53337	-19.85	0.03001	0.14099	0
false	-1.4383	0.88587	-0.49755	-19.823	0.037032	0.13105	0
false	-1.3969	0.82543	-0.4607	-19.871	0.037032	0.14529	0

Once the feature tables, which contain the features and the labels, have been computed for each anomaly type, train an SVM model with a gaussian kernel. Further, as seen in summary of one of the feature tables, the features have different ranges. To prevent the SVM model from being biased towards features with larger values, set the Standardize parameter to true. Save the three models to the anomalyDetectionModelSVM.mat file for use in a later section.

```

74 m.m1 = fitcsvm(featureTableTrain1, 'Anomaly', 'Standardize',true, 'KernelFunction','gaussian');
75 m.m2 = fitcsvm(featureTableTrain2, 'Anomaly', 'Standardize',true, 'KernelFunction','gaussian');
76 m.m3 = fitcsvm(featureTableTrain3, 'Anomaly', 'Standardize',true, 'KernelFunction','gaussian');
77
78 % Save model for reuse later
79 save anomalyDetectionModelSVM m

```

Model Testing and Validation

To test the model accuracy, use the data in the testEnsemble. Read the signal and anomaly data in testEnsemble, and from that data, create a data ensemble using the

Model Testing and Fault Classification



①

Data
Preparation

②

AI
Modeling

③

Simulation
and Test

④

Deployment

- Features extracted using **Diagnostic Feature Designer APP**
- Three SVM classifiers trained for each anomaly type
- Deployment options:
 - Microservice (Docker)
 - **Code Generation to be implemented on PLC/Edge Device**

Feature Extraction, Classifier training and Fault Classification

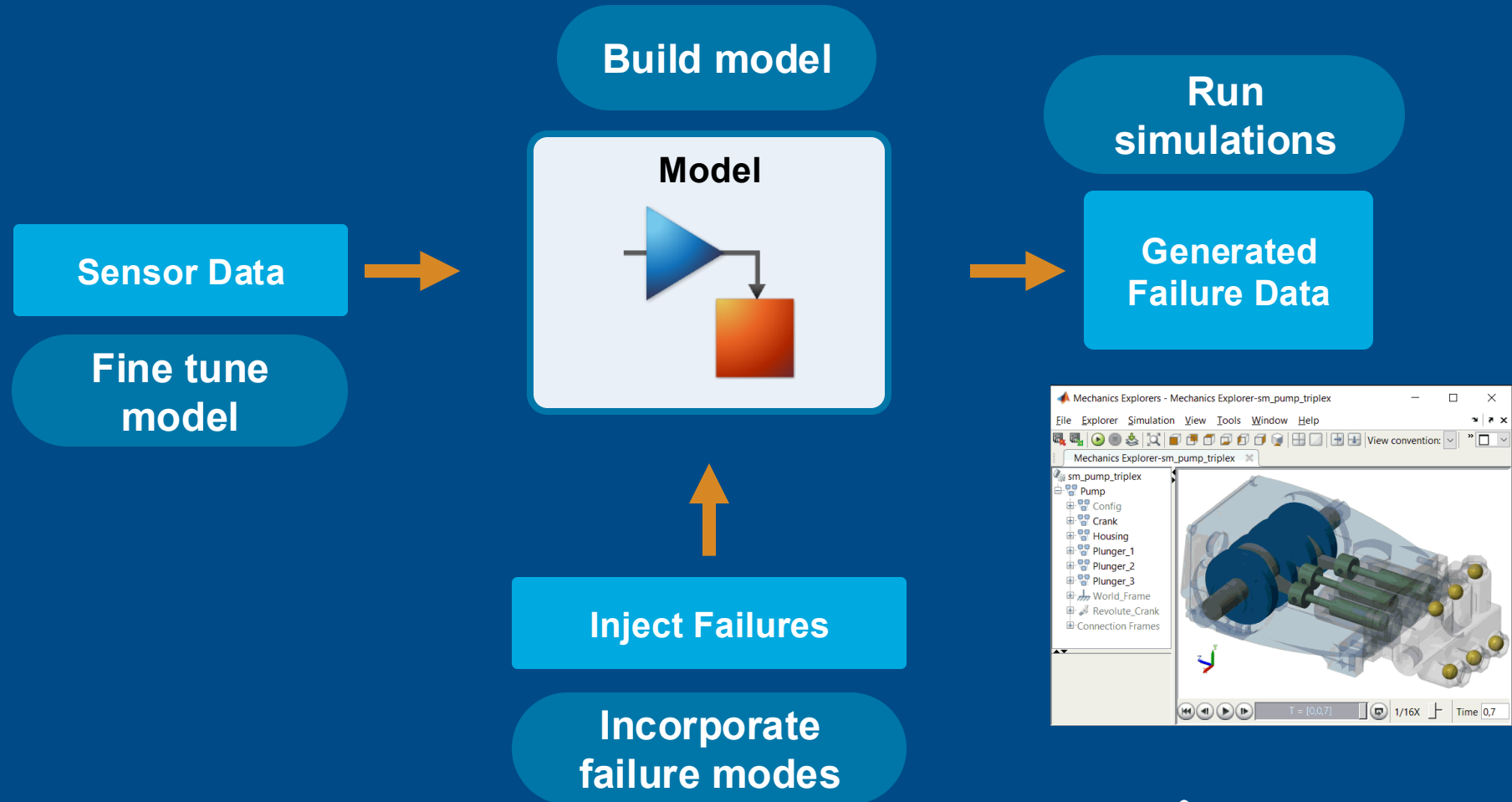


① Data Preparation

② AI Modeling

③ Simulation and Test

④ Deployment



Use data from sensors and simulations

1

Data
Preparation

2

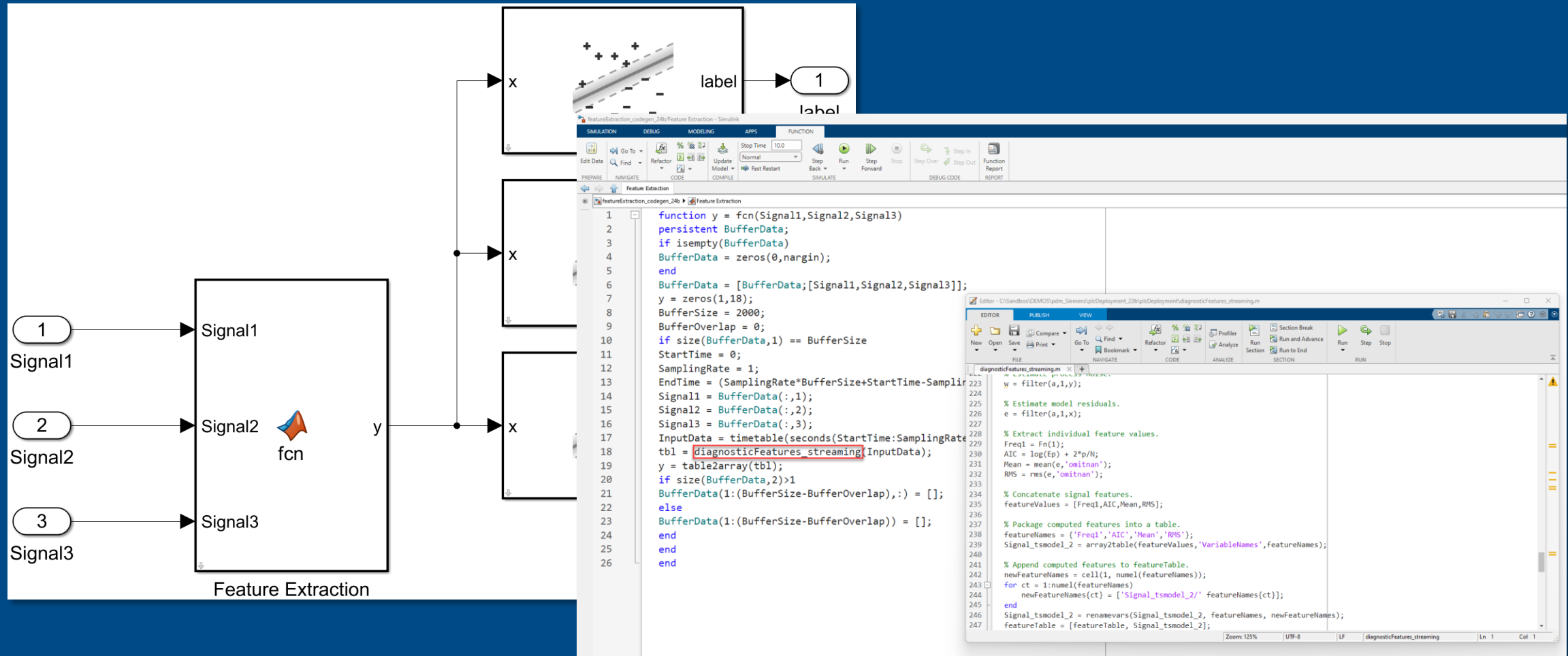
AI
Modeling

3

Simulation
and Test

4

Deployment



Feature Extraction / Fault Classification

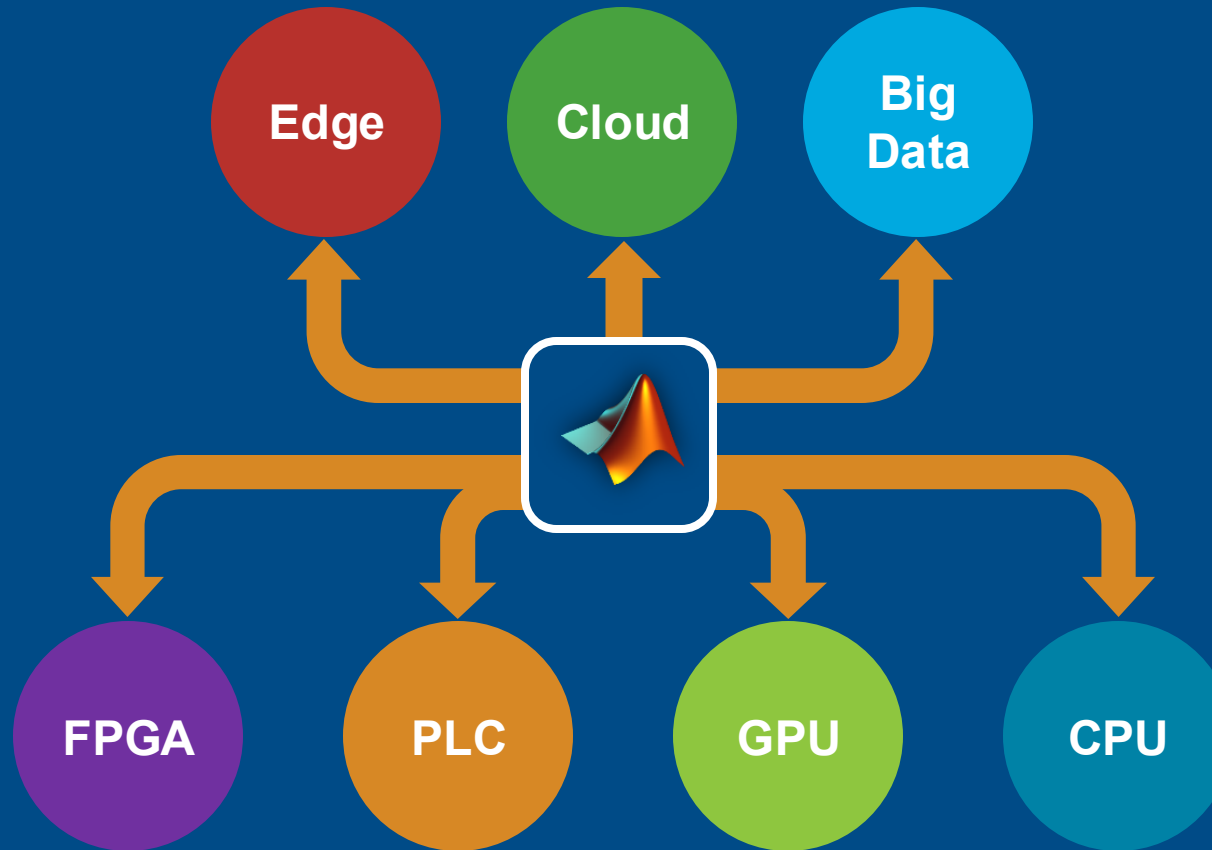


① Data Preparation

② AI Modeling

③ Simulation and Test

④ Deployment



Deploy with zero coding errors



①

Data
Preparation

②

AI
Modeling

③

Simulation
and Test

④

Deployment

Development Computer

MATLAB

Digital Twin

Fault Detection

GUI

Siemens
TIA Portal

Codegen

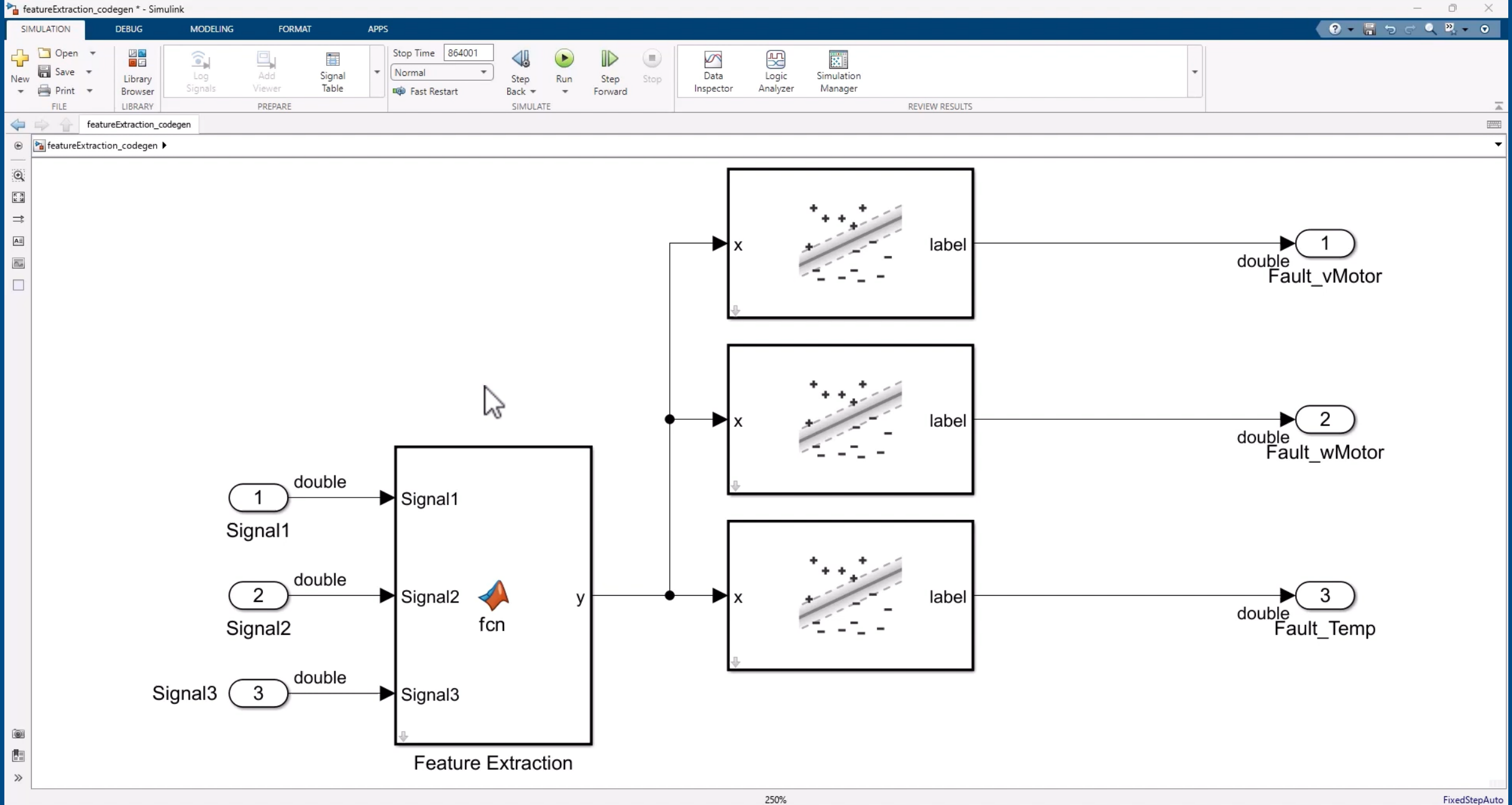
Deployment

Fault Detection
Algorithm

PLC

Code Generation: Feature Extraction
and Fault Classification





①

Data
Preparation

②

AI
Modeling

③

Simulation
and Test

④

Deployment

Development Computer

MATLAB

Digital Twin

Fault Detection

GUI

Codegen

Siemens
Live Twin

Deployment

Digital Twin

Industrial Edge
Device

Code Generation: Digital Twin



Simulink interface for the **CoolingFanWithFaults** model. The top toolbar includes tabs for SIMULATION, DEBUG, MODELING, FORMAT, APPS, C++ CODE, and TESTS. The C++ CODE tab is active, showing options like Generic Code - C++, Quick Start, C/C++ Code Advisor, and Settings. The Model Browser on the left lists components such as Aero Drag, Square law, Drag Coefficient, Convection Fan Forced, Convection Degradation, DC Source Voltage, Environment, External Temp, Heat Flow, Meas 1, Sensing Speed, and Subsystem.

The main workspace displays the **CoolingFanWithFaults** model diagram. The diagram illustrates a thermistor-controlled fan system. It includes a DC Source Voltage block, a Thermistor, a Conduction Thermistor, a Case Thermal Mass, a Heat Flow block, a Convection Fan Forced block, a Convection Degradation block, a Fan Inertia block, an Aero Drag block, and a DC Motor. The system is controlled by a PWM block and a Controlled PWM Voltage block. The diagram also shows various signal blocks and a block labeled $f(x) = 0$.

Below the diagram, the text "Thermistor-Controlled Fan" is followed by three numbered instructions:

1. [Plot temperature](#) and other conditions in system ([see code](#))
2. [Explore simulation results](#) using [sscexplore](#)
3. [Learn more](#) about this example

On the right side of the workspace, there are two signal processing blocks. The top block, labeled "Signals", receives inputs from vMotor, wMotor, and Tmass and outputs to mSignals. The bottom block, labeled "Anomaly Flags", receives inputs from ExternalTempAnomaly, FanDragAnomaly, and VoltageSourceAnomaly and outputs to ExternalTemp, DragCoeff, and DCSourceVoltage, which are then fed into ExternalAnomalies.

The bottom status bar shows "Ready", "View 1 error", "109%", and "auto(ode14x)".

①

Data
Preparation

②

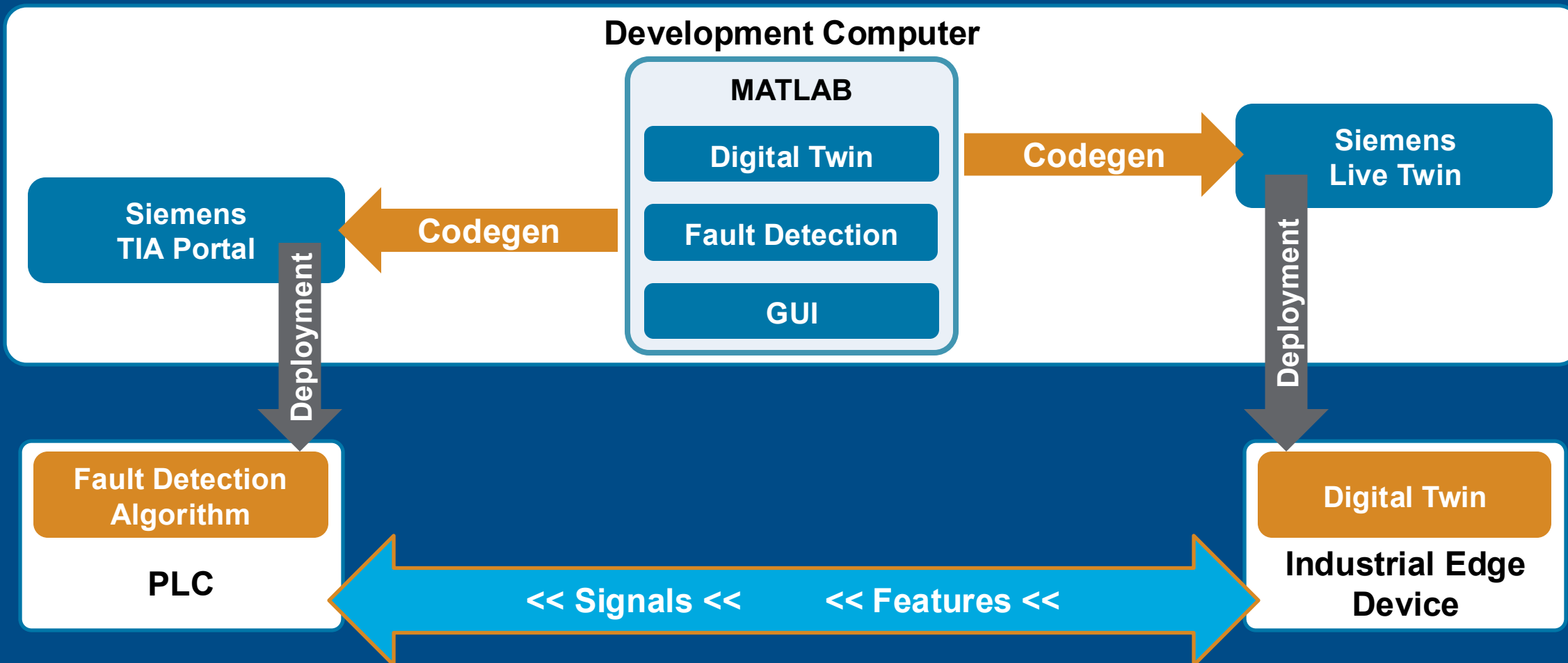
AI
Modeling

③

Simulation
and Test

④

Deployment



Algorithm Export and Integration
TIA Portal and Live Twin

Siemens - C:\Sandbox\DEMOS\PLC Projects\TIA Portal\program_test\program_test

Totally Integrated Automation PORTAL

Project tree: ...1 [CPU 1505SP F] > Program blocks > Cyclic interrupt [OB30]

Block interface: "featureExtraction_codegenOne Step"

Network 3:

Network 4:

Details view: Data

Name	Value
Temp_Anomaly	0.0

Properties: General, Cross-references, Compile, Syntax

Portal view: Overview, Cyclic inter..., Data (DB4)

LiveTwin

Not secure | https://172.26.81.217/livetwin/?1736857651719#/Models/filters?focus=Project

Import favorites | Lenovo Support | Lenovo | McAfee

SIEMENS

Hello | LiveTwin

Fan_Digital_Twin x Fan_Digital_Twin_0 x

Project - Fan_Digital_Twin

Created: 10/25/2024, 11:56:40 AM - Modified: 10/29/2024, 11:30:31 AM

Project Settings

Template: Plan_Fan | Template Version: 1 | Project Type: LiveTwin | Model Type: Simulink

Project Name: Fan_Digital_Twin

Simulation Step (ms): 100 | Project Cyclic Time (ms): 100 | Simulation Rate: 1 | Real time

Startup: Manual

Model Info

Instances

+ Add Instances | Remove Instances

Status	Name	Uptime	Mapped Inputs	Mapped Outputs	Parameters Set	Actions
RUNNING	Fan_Digital_Twin_0	43m 18s	0 / 0	3 / 9	Custom Set	Export Remove Save & Close Apply

①

Data
Preparation

②

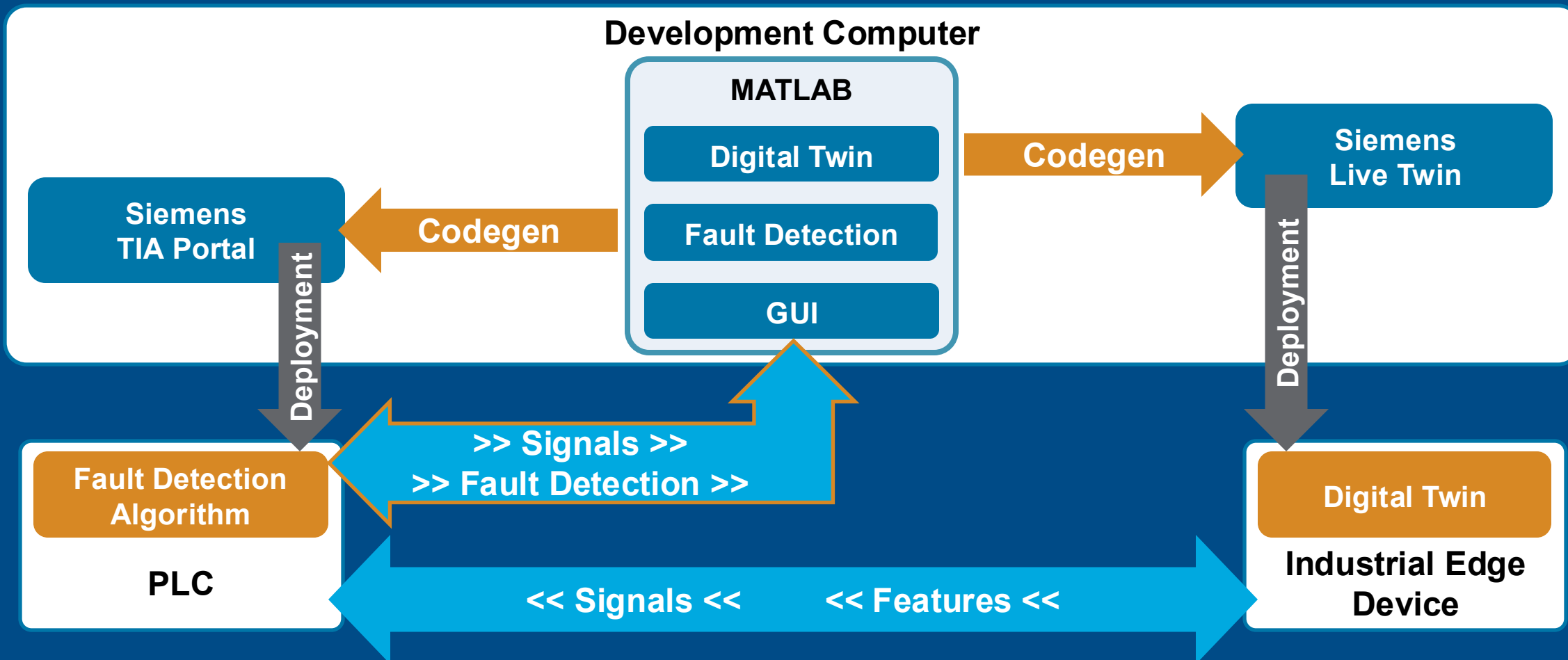
AI
Modeling

③

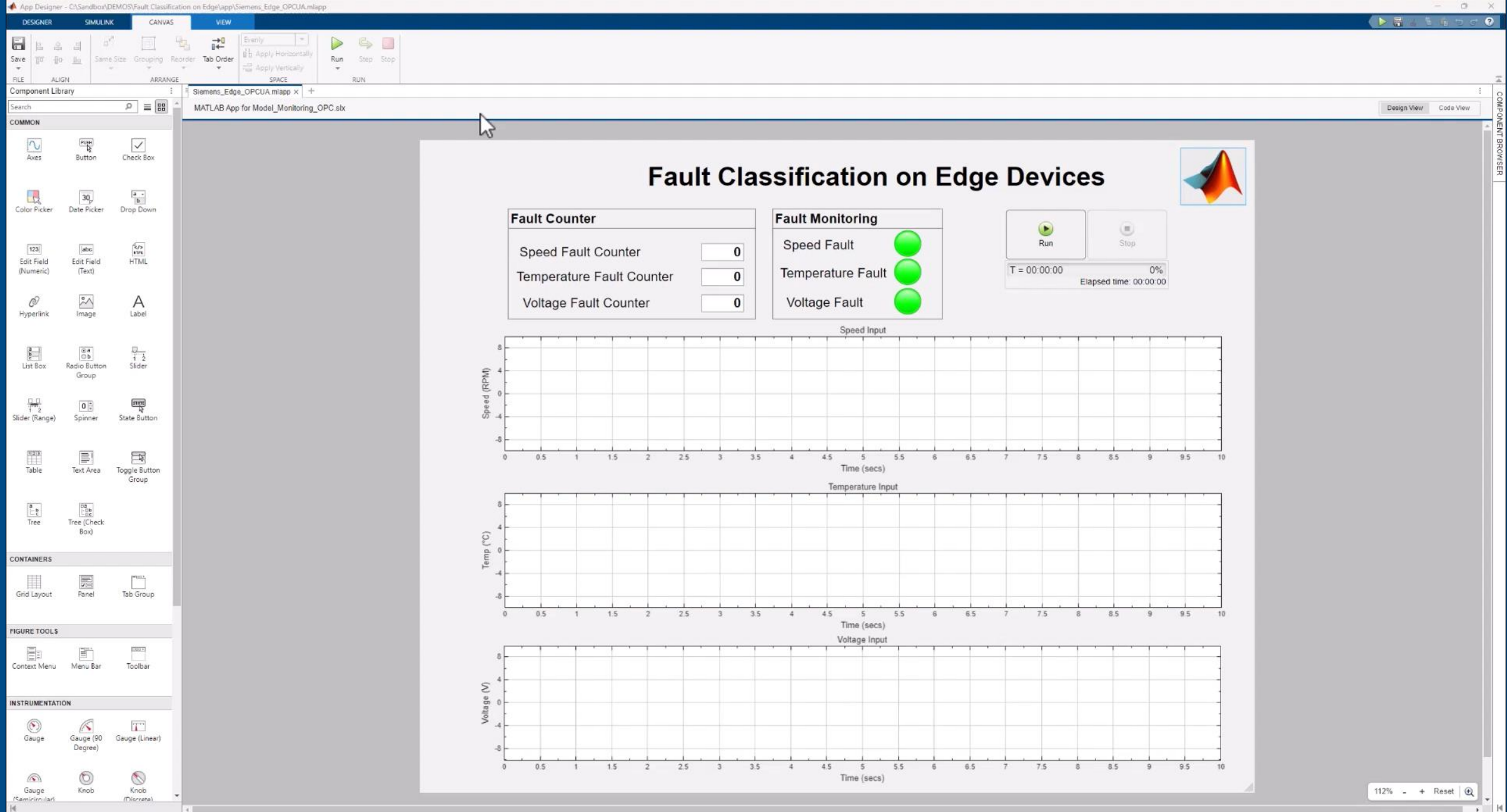
Simulation
and Test

④

Deployment



Industrial Communication – OPC UA



Siemens - C:\Sandbox\DEMOS\PLC Projects\TIA Portal\program_test

Totally Integrated Automation PORTAL

Project tree: Devices, Program blocks, Data [DB4]

Block interface: Network 3, Network 4

Details view: Data, Temp_Anomaly

LiveTwin: https://172.26.81.217/livetwin/71736857651719#/Models/filters?focus=Project

Fault Classification on Edge Devices

STATUS: RUNNING [Stop Update]

Fault Counter

Speed Fault Counter	4
Temperature Fault Counter	8
Voltage Fault Counter	12

Fault Monitoring

Speed Fault	<input checked="" type="checkbox"/>
Temperature Fault	<input checked="" type="checkbox"/>
Voltage Fault	<input checked="" type="checkbox"/>

Elapsed time = 00:05:38

Buttons: Pause, Stop

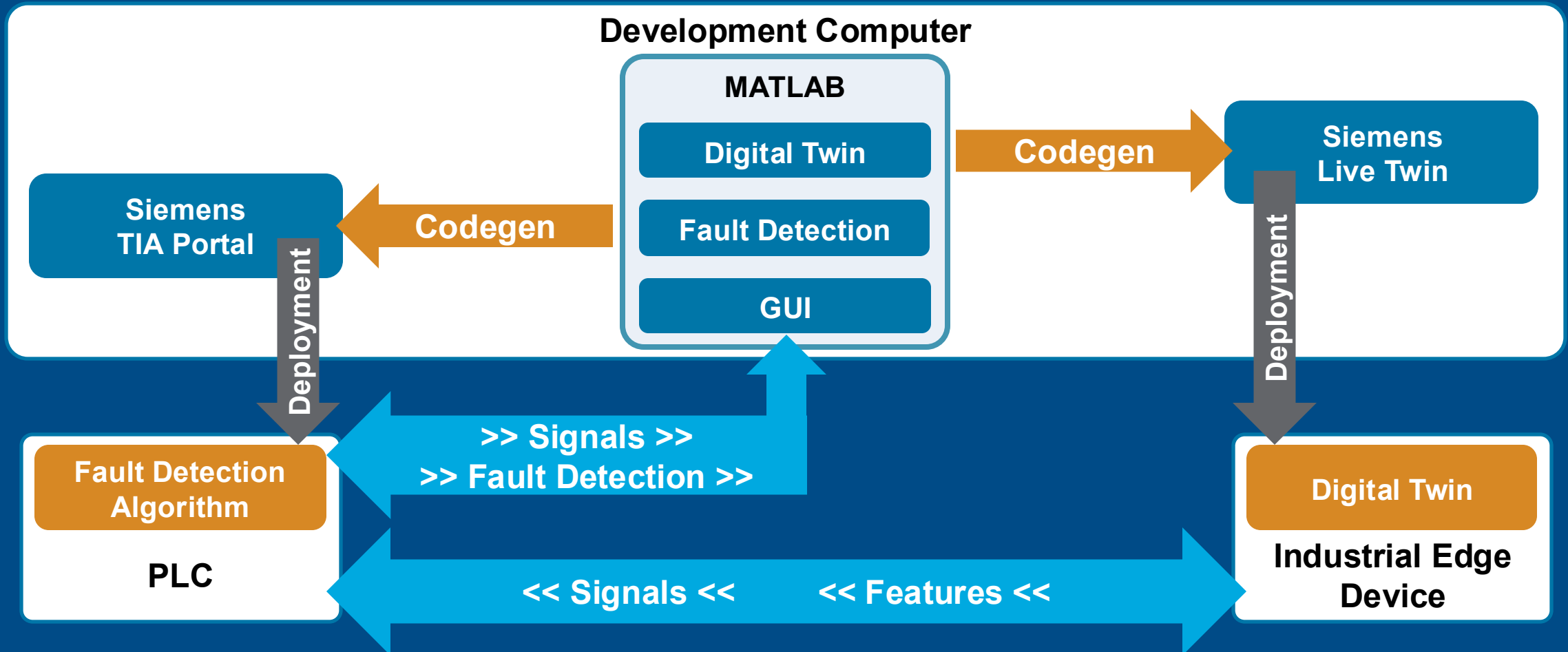
Speed Input

Temperature Input

Voltage Input

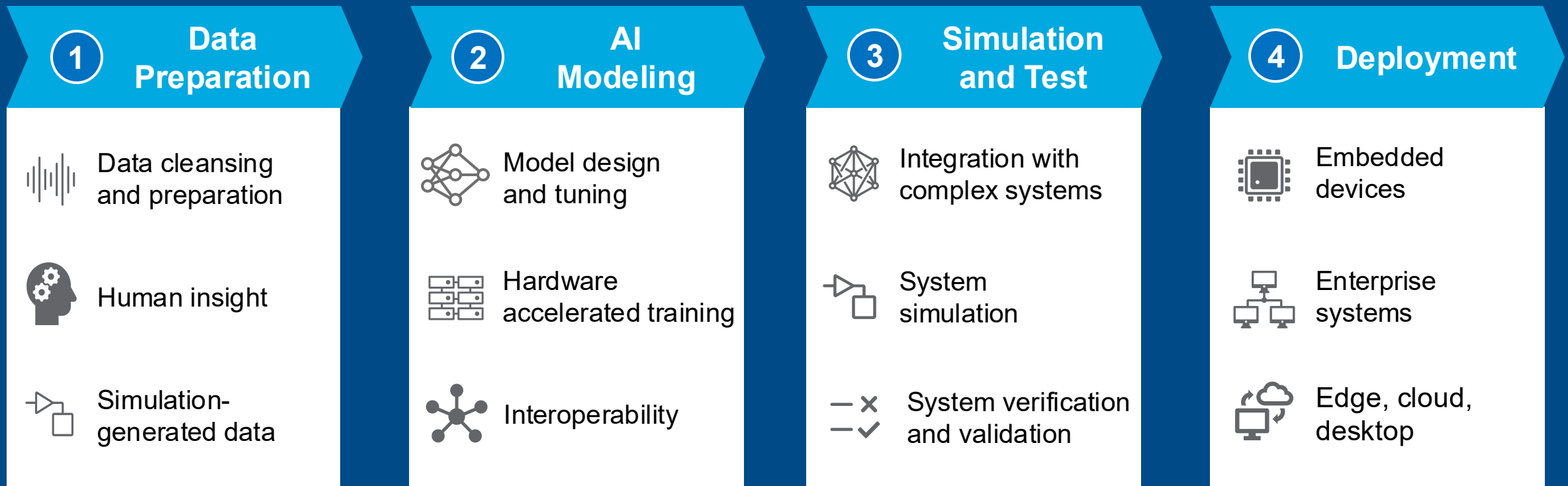
Simulation Time (mm:ss): 54:15 to 55:45

CPU Consumption: 0.89% Memory Consumption: 15.77MB Execution Step Time: 3.33ms Uptime: 55m 51s



AI Based Fault Detection on Industrial Controllers

- Empower domain experts, including ones with limited AI experience
- Build better data sets with domain-specific tools
- Use modeling and simulation to tackle integration challenges and reduce risk
- Deploy AI models to wherever you need them



Lessons learned: Use End-to-End Workflows
for Artificial Intelligence

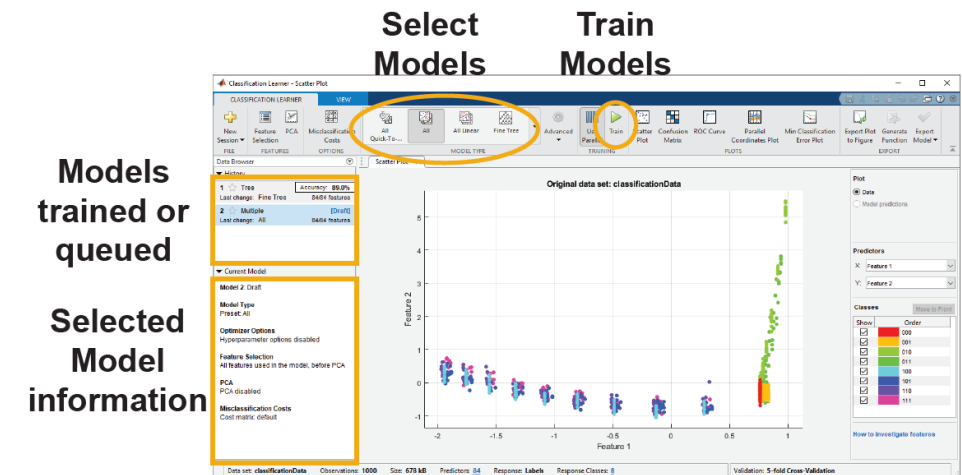
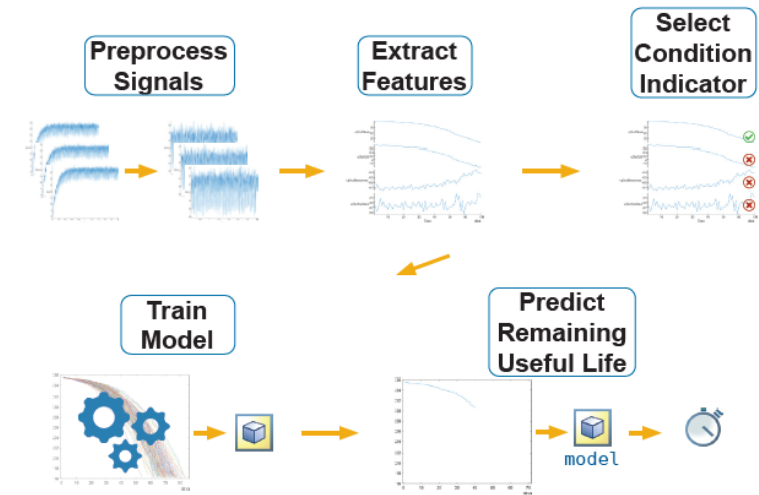
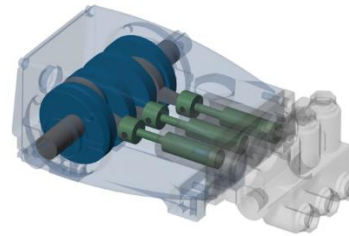
MathWorks Training & Consulting



Predictive Maintenance with MATLAB

Topics included in this 2-day course:

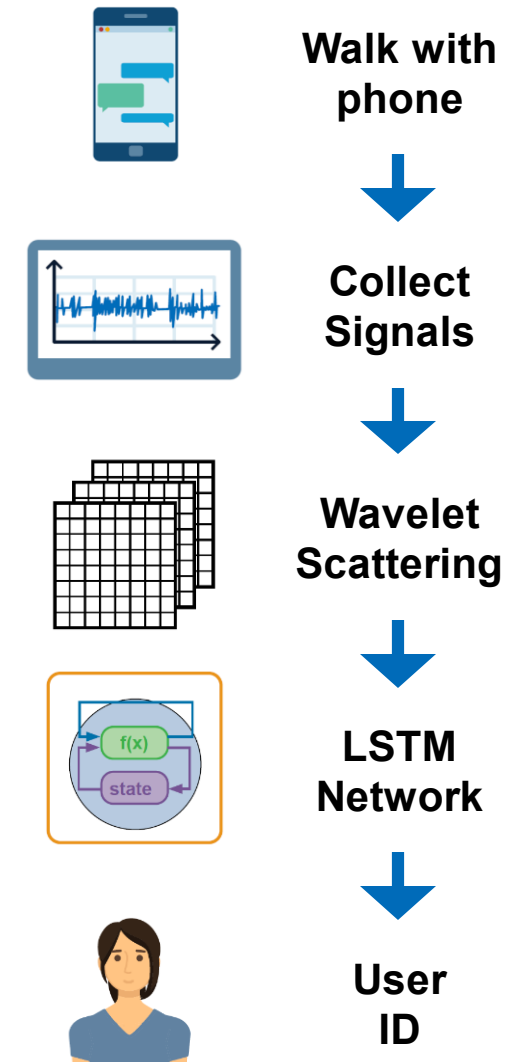
- Importing and organizing data
- Creating custom visualizations
- Fault Detection/Classification
- Preprocessing to improve data quality, and extract time and frequency domain features
- Estimating Remaining Useful Life (RUL)
- Interactive workflows with apps



Deep Learning for Signals in MATLAB

After this 1-day training you will be able to:

- Import and label signal data
- Use CNNs for signal classification
- Create custom LSTMs for signal classification
- Apply Deep Learning for anomaly detection





Training Services Portfolio

Getting Started

[Prepare for training with introductory tutorials](#)

Over 30 hours of free, online training across many topics

Courses

[Bridge the skills gap](#)

Over 70 courses across 9 Focus Areas, addressing Fundamental, Intermediate and Advanced Training needs

Custom Curricula

[Tailored content](#)

We can accommodate tight schedules by combining course content and tailoring content to address unique training needs

Format

[Accommodate learning styles](#)

8 of the courses are online (self-paced) and all are available as ILT (virtual and in-person, in public and private settings)

Packaging

[Flexible options for whatever works best for you](#)

All courses are individually priced. Self-paced courses are available as a 12-month subscription.

MATLAB and Simulink Consulting Services

Achieve Results Faster

Work with MathWorks to speed up your MATLAB and Simulink Projects.

**Advisory
Services**

**Process
Assessments**

**Custom
Projects**

**Strategic
Partnership**



Transparent Approach



Customized Engagements



Return on Investment

For more information see our Proven-Solutions [website](#).

Training and Consulting Services



Training and Consulting Services



MATLAB EXPO

Tuesday, October 21, 2025 | The Westin Grand Munich

Register at [MATLAB EXPO Deutschland](#)



© 2025 The MathWorks, Inc.

