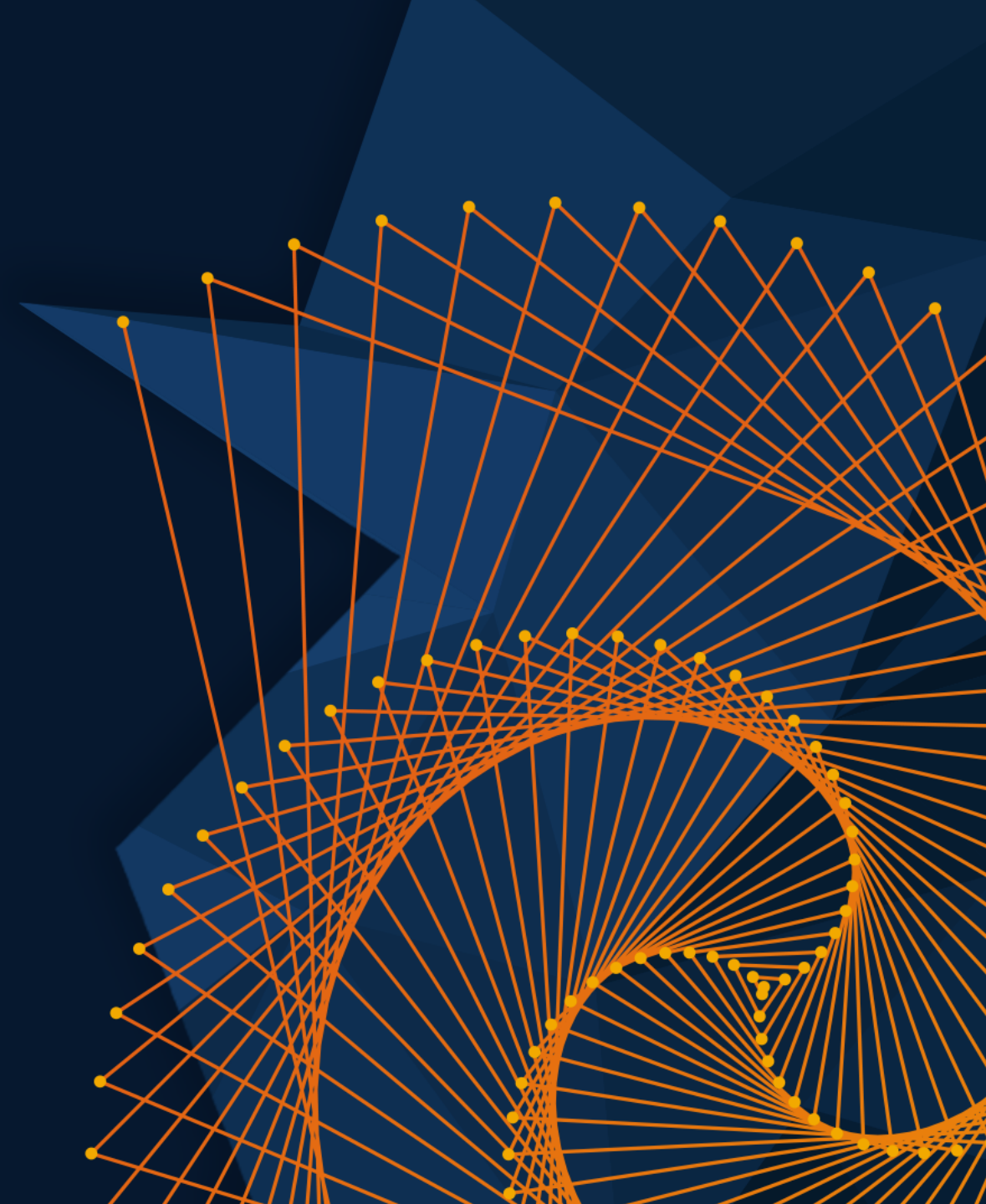


MATLAB EXPO

AI 기반 차수 축소 모델링

엄준상, 김종남 MathWorks

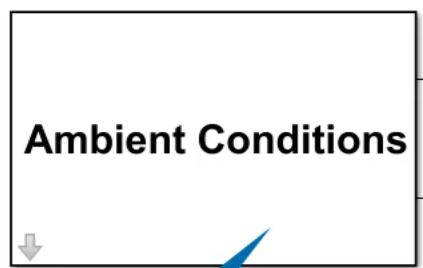


SIMULATION DEBUG MODELING FORMAT APPS **VARIANT SUBSYSTEM**

FILE LIBRARY PREPARE SIMULATE REVIEW RESULTS

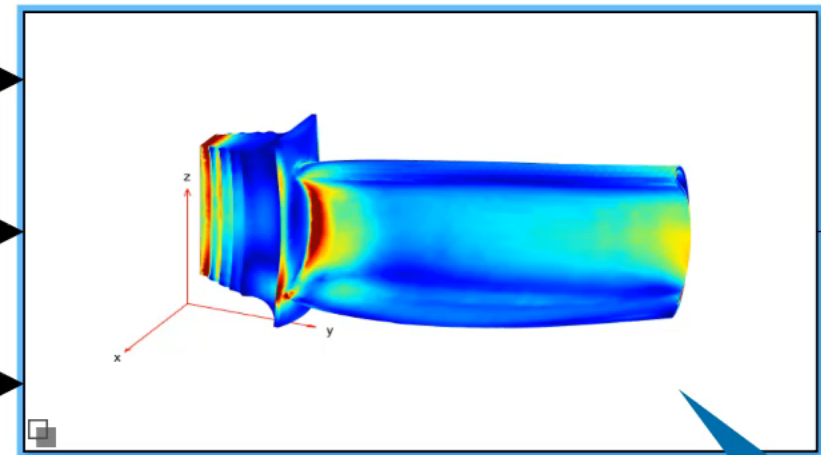
Open Save Print Library Browser Log Signals Add Viewer Signal Table Stop Time 7000 Normal Fast Restart Step Back Run Step Forward Stop Data Inspector Logic Analyzer Simulation Manager

SystemLevelSim_JetEngineBlade Reduced order model



Temperature and pressure conditions

Cooling temperature controller



FEA model to compute maximum tip displacement

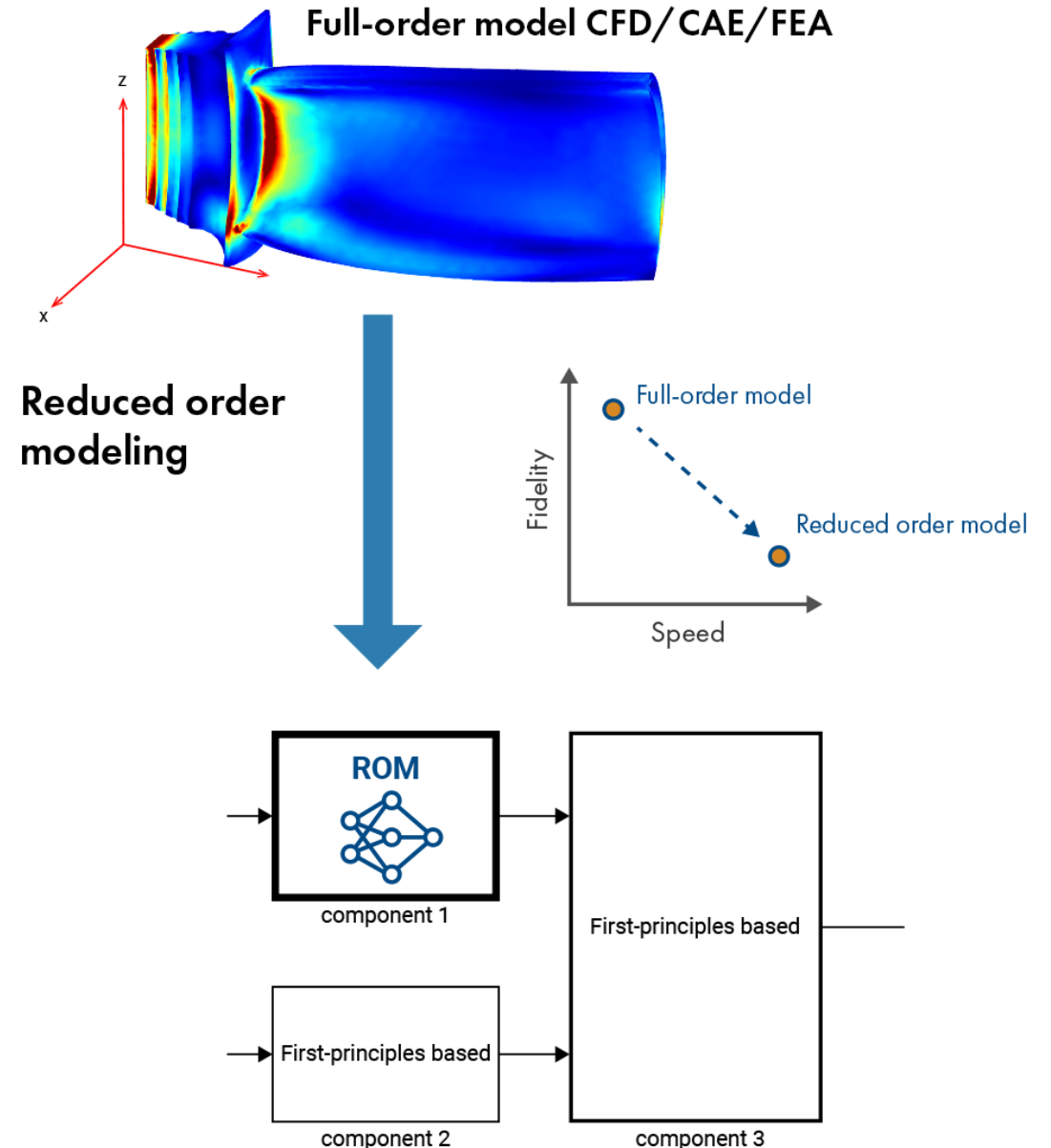
Reduced Order Modeling

What

- Techniques to **reduce the computational complexity** of a computer model
- Provide reduced, but acceptable fidelity**

Why

- Enable simulation of FEA models in Simulink
- Perform hardware-in-the-loop testing
- Perform control design
- Develop virtual sensors, Digital twins
- Enable desktop simulations for orders-of-magnitude longer timescales



Introducing Simulink Add-On for Reduced Order Modeling

Create AI-based reduced order models (ROM)

Set up Design of Experiments (DoE)

Generate input-output data from full-order, high-fidelity subsystems

Train and compare AI-based reduced order models using preconfigured templates

Export trained reduced order models into Simulink or outside of Simulink through FMUs

Reduced Order Modeler App

Products and Services Search MathWorks.com

Reduced Order Modeling with MATLAB and Simulink

Create AI-based reduced order models

Download add-on (beta)

Model: LSTM, Nonlinear ARX, Neural State Space

Signal	Min	Max
1 Ambient	800	2000
2 Cooling	50	250
3 Pressure	450000	850000

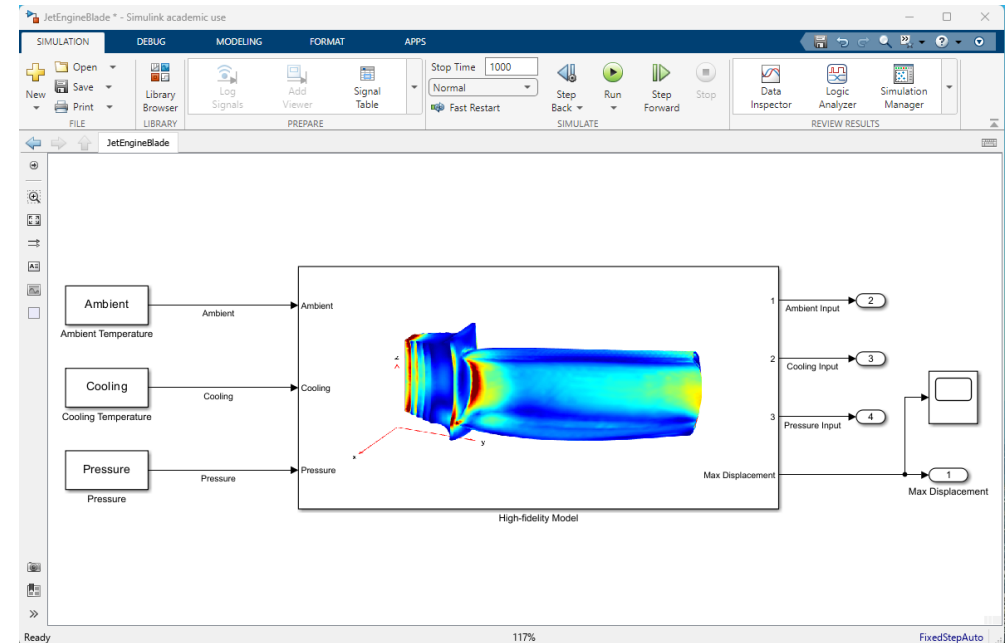
Full-order model CFD/CAE/FEA → Reduced Order Modeling → ROM component 1, First-principles based component 2, component 3

Fidelity vs Speed graph showing Full-order model and Reduced order model.

Generate data for training



Physical system



Simulink/Simscape

Data Preparation

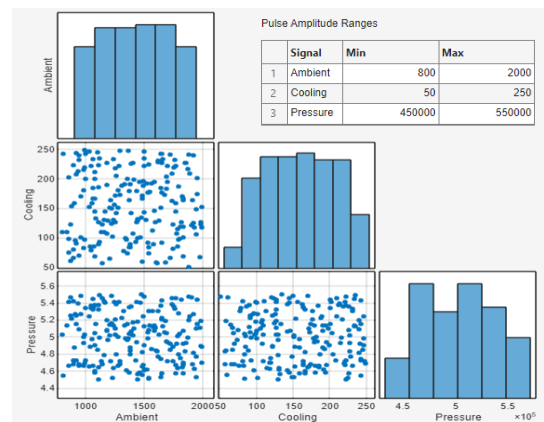
AI Modeling

Simulation & Test

Deployment

Synthetic Data Generation

Design of Experiments

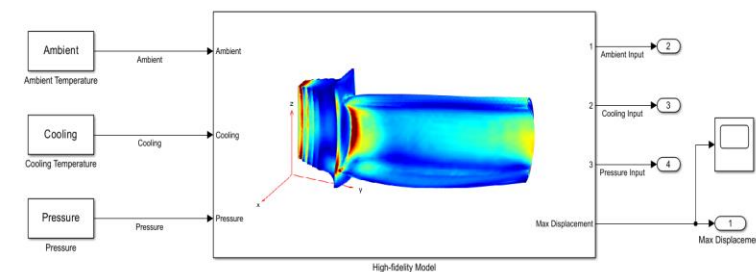
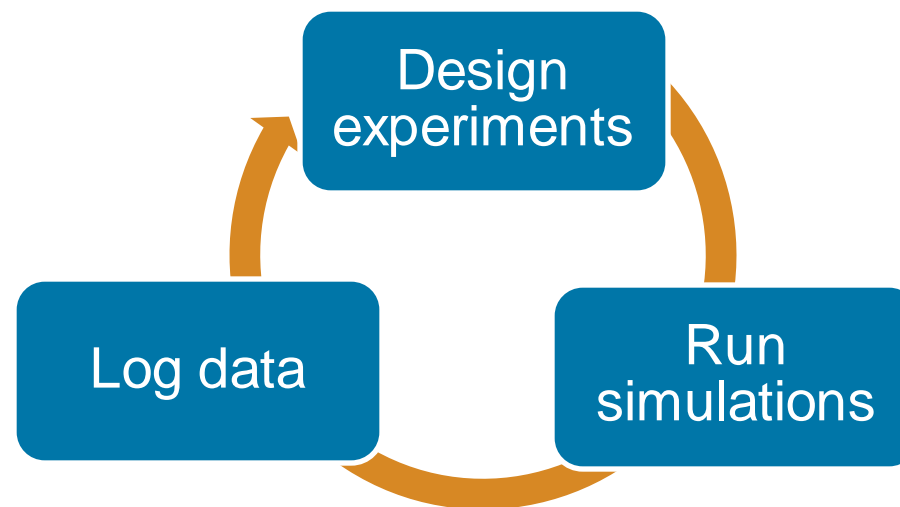


Input features

Ambient Temperature
Ambient Pressure
Cooling Temperature

Response

Max Displacement



Data Preparation

AI Modeling

Simulation & Test

Deployment

Synthetic Data Generation

Design of Experiments

Getting Started with Reduced Order Modeling Support Package

What Is Reduced Order Modeling?
Reduced order modeling is a technique for simplifying full order high-fidelity models by reducing their computational complexity, while preserving their dominant behavior. Working with a reduced order model (ROM) can simplify analysis and control design.

Why Use Reduced Order Modeling?
Using reduced order modeling techniques, you can:

- **Enable use of 3rd party FEA/FEM/CFD models for system-level simulation in Simulink® including hardware-in-the-loop testing** — You can combine multiple complex component-level models, including third-party finite element method (FEM) or finite element analysis (FEA) models, into system-level simulation models in Simulink by replacing the complex models with the corresponding ROMs. ROMs are also useful for hardware in-the-loop testing as they allow real-time simulations. Engineers can create ROMs representing the physical components of the system, which can run on a real-time machine for testing of the control algorithm on embedded hardware. The reduced computational complexity of ROMs make such testing more feasible.
- **Create virtual sensors** — You can use ROMs as virtual sensors for estimating or predicting signals of interest when measuring those signals by using a physical sensor is impractical or impossible.
- **Perform control design** — The reduced complexity of ROMs can make control design tasks more tractable. You can design your controller for the reduced order model of a plant and then validate the controller on the original high-fidelity system. You can also use ROMs for control algorithms that require internal prediction models, such as nonlinear model predictive control.
- **Create digital twins** — You can create or simplify digital twin models using ROMs. Doing so makes the digital twins more computationally efficient and more suitable for periodic updates to represent the current state of the operational asset.

Reduced Order Modeler App Workflow
The general workflow of the Reduced Order Modeler app involves logging data from a Simulink model and using that data to train a ROM. It includes the following steps.

```

graph LR
    A[Open Model and App] --> B[Select Inputs and Outputs]
    B --> C[Specify Experiments]
    C --> D[Run Model]
    D --> E[Create Reduced Order Model]
    E --> F[Export Model]
    F --> G[Replace Blocks in Simulink Model]
  
```

Command Window
fx >>

Data Preparation

AI Modeling

Simulation & Test

Deployment

Inputs/Outputs

ROM Input

- JetEngineBlade/High-fidelity Model/First Order Hold1:1(Ambient_input)
- JetEngineBlade/High-fidelity Model/First Order Hold1:1(Cooling_input)
- JetEngineBlade/High-fidelity Model/First Order Hold2:1(Pressure_input)

ROM Output

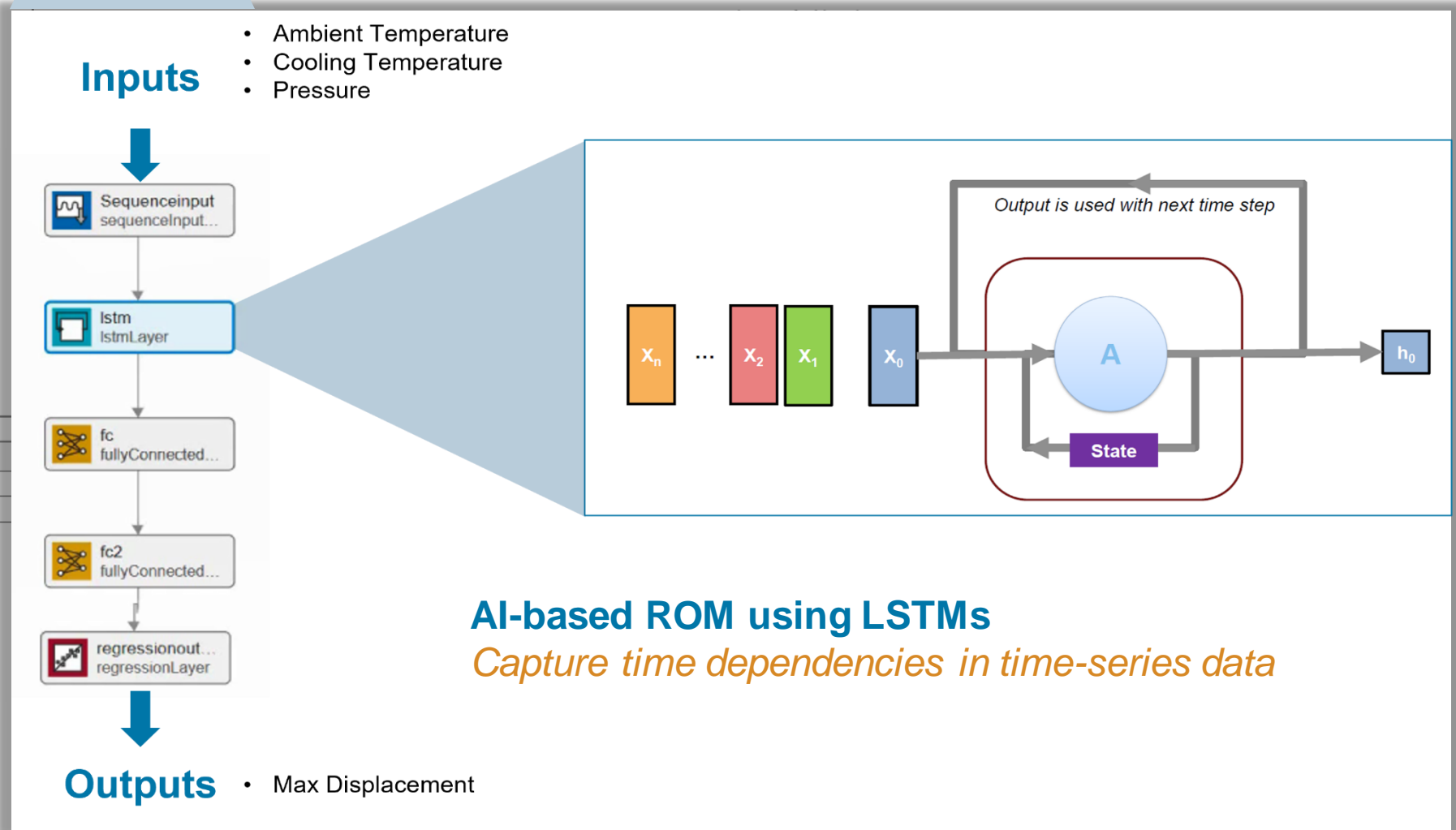
- JetEngineBlade/High-fidelity Model/MATLAB Function:1(maxDisp)

Simulation Input

- JetEngineBlade/Ambient Temperature:1(Ambient)
- JetEngineBlade/Cooling Temperature:1(Cooling)
- JetEngineBlade/Pressure:1(Pressure)

Experiments

	Name	# Sims	Results
<input checked="" type="checkbox"/>	Experiment		1 Data
<input checked="" type="checkbox"/>	Experiment_1		1 Data



Simulation Result 1 of 1

Show output only

Show as scatter plot

SIMULATION RESULTS OPTIONS

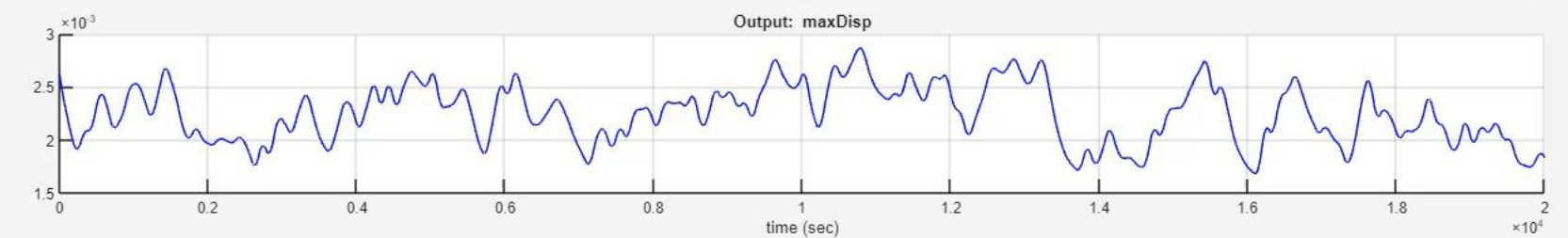
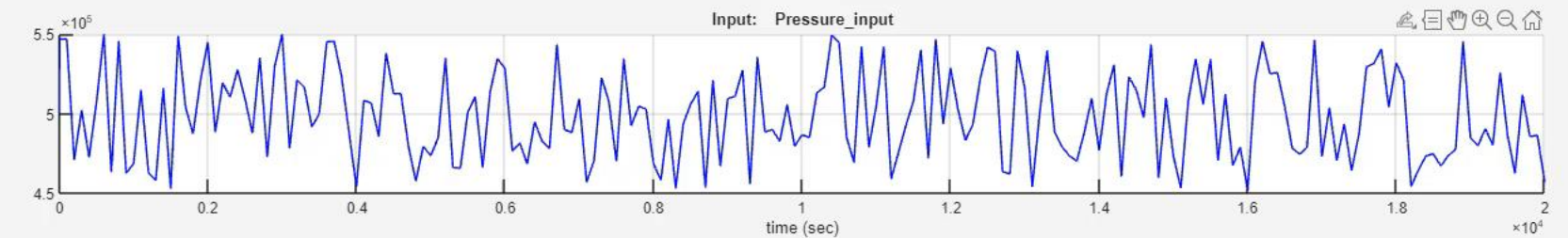
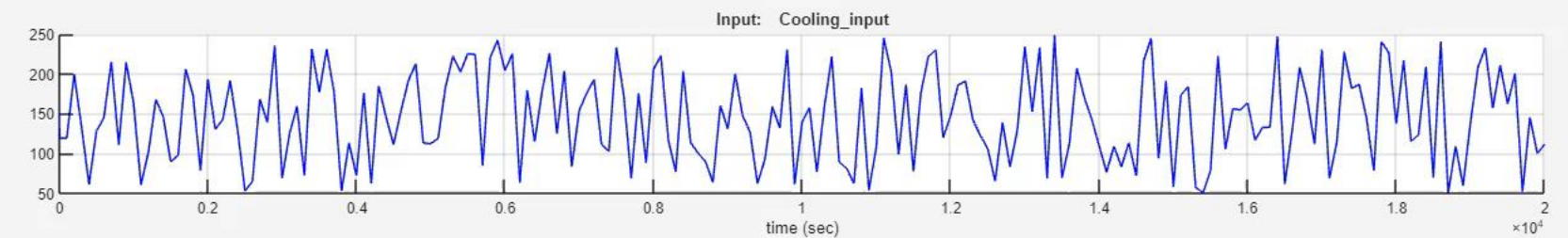
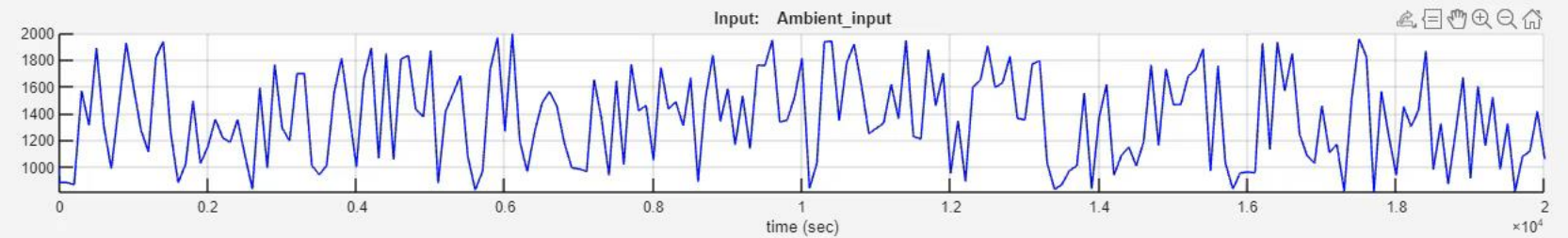
Inputs/Outputs

- ROM Input
 - JetEngineBlade/High-fidelity Model/First Order Hold:1(Ambient_input)
 - JetEngineBlade/High-fidelity Model/First Order Hold1:1(Cooling_input)
 - JetEngineBlade/High-fidelity Model/First Order Hold2:1(Pressure_input)
- ROM Output
 - JetEngineBlade/High-fidelity Model/MATLAB Function:1(maxDisp)
- Simulation Input
 - JetEngineBlade/Ambient Temperature:1(Ambient)
 - JetEngineBlade/Cooling Temperature:1(Cooling)
 - JetEngineBlade/Pressure:1(Pressure)

Experiments

	Name	# Sims	Results
<input checked="" type="checkbox"/>	Experiment	1	Data
<input checked="" type="checkbox"/>	Experiment_1	1	Data

Overview x Result: Experiment x Result: Experiment_1 x Experiment_1 x



Reduced Order Modeler-JetEngineBlade

REDUCED ORDER MODEL SIMULATION RESULT

New Session Open Save Edit Inputs/Outputs New Experiment Simulation Options Run Simulations Open Results LSTM Network Nonlinear ARX **Neural State Space** Export

FILE INPUTS/OUTPUTS COLLECT DATA EXPORT

Inputs/Outputs

ROM Input

- JetEngineBlade/High-fidelity Model/First Order Hold1:1(Ambient_input)
- JetEngineBlade/High-fidelity Model/First Order Hold1:1(Cooling_input)
- JetEngineBlade/High-fidelity Model/First Order Hold2:1(Pressure_input)

ROM Output

- JetEngineBlade/High-fidelity Model/MATLAB Function:1(maxDisp)

Simulation Input

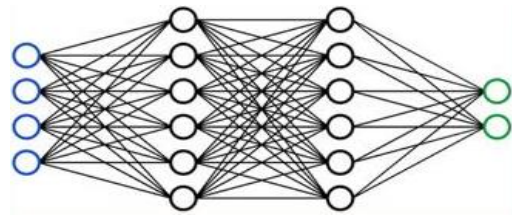
- JetEngineBlade/Ambient Temperature:1(Ambient)
- JetEngineBlade/Cooling Temperature:1(Cooling)
- JetEngineBlade/Pressure:1(Pressure)

Experiments

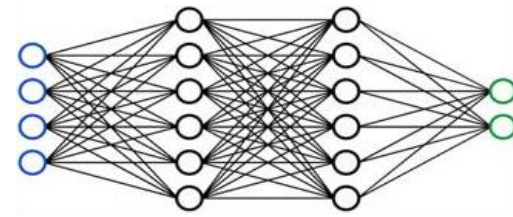
	Name	# Sims	Results
<input checked="" type="checkbox"/>	Experiment		1 Data
<input checked="" type="checkbox"/>	Experiment_1		1 Data

Overview x Result: Experiment_1 x

$$\begin{cases} \dot{x} = f(x, u) \\ y = g(x, u) \end{cases}$$



State Network (f)



Output Network (g)

AI-based ROM using Neural State Space (also known as Neural ODE)
Create Deep Learning-based nonlinear state-space models

Data Preparation **AI Modeling** Simulation & Test Deployment

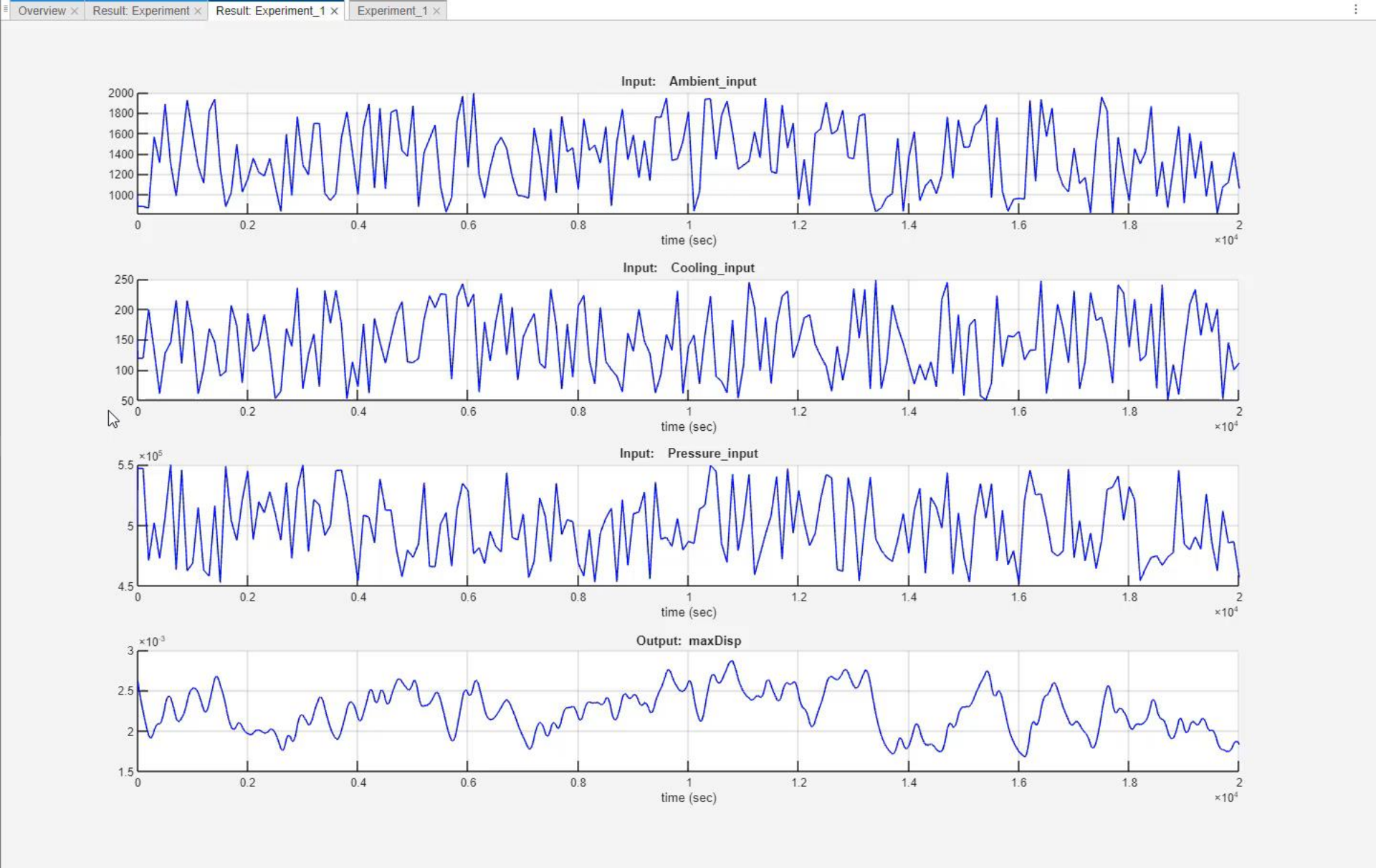
Session opened, engineBlade_ROMsession_results.ma

Inputs/Outputs

- ROM Input
 - JetEngineBlade/High-fidelity Model/First Order Hold:1(Ambient_input)
 - JetEngineBlade/High-fidelity Model/First Order Hold1:1(Cooling_input)
 - JetEngineBlade/High-fidelity Model/First Order Hold2:1(Pressure_input)
- ROM Output
 - JetEngineBlade/High-fidelity Model/MATLAB Function:1(maxDisp)
- Simulation Input
 - JetEngineBlade/Ambient Temperature:1(Ambient)
 - JetEngineBlade/Cooling Temperature:1(Cooling)
 - JetEngineBlade/Pressure:1(Pressure)

Experiments

	Name	# Sims	Results
<input checked="" type="checkbox"/>	Experiment		1 Data
<input checked="" type="checkbox"/>	Experiment_1		1 Data



Reduced Order Modeler-JetEngineBlade

REDUCED ORDER MODEL SIMULATION RESULT

New Session Open Session Save Session Edit Inputs/Outputs New Experiment Simulation Options Run Simulations Open Results LSTM Network **Nonlinear ARX** Neural State Space Export

Inputs/Outputs

ROM Input

- JetEngineBlade/High-fidelity Model/First Order Hold1:1(Ambient_input)
- JetEngineBlade/High-fidelity Model/First Order Hold1:1(Cooling_input)
- JetEngineBlade/High-fidelity Model/First Order Hold2:1(Pressure_input)

ROM Output

- JetEngineBlade/High-fidelity Model/MATLAB Function:1(maxDisp)

Simulation Input

- JetEngineBlade/Ambient Temperature:1(Ambient)
- JetEngineBlade/Cooling Temperature:1(Cooling)
- JetEngineBlade/Pressure:1(Pressure)

Experiments

	Name	# Sims	Results
<input checked="" type="checkbox"/>	Experiment		1 Data
<input checked="" type="checkbox"/>	Experiment_1		1 Data

input u

output y

Regressors
 $u(t), u(t-1), y(t-1), \dots$
Current input and past inputs and outputs

Output Function

Offset

Nonlinear Function

Linear Function

AI-based ROM using Nonlinear ARX

Extend linear models and model nonlinear behavior using flexible nonlinear functions

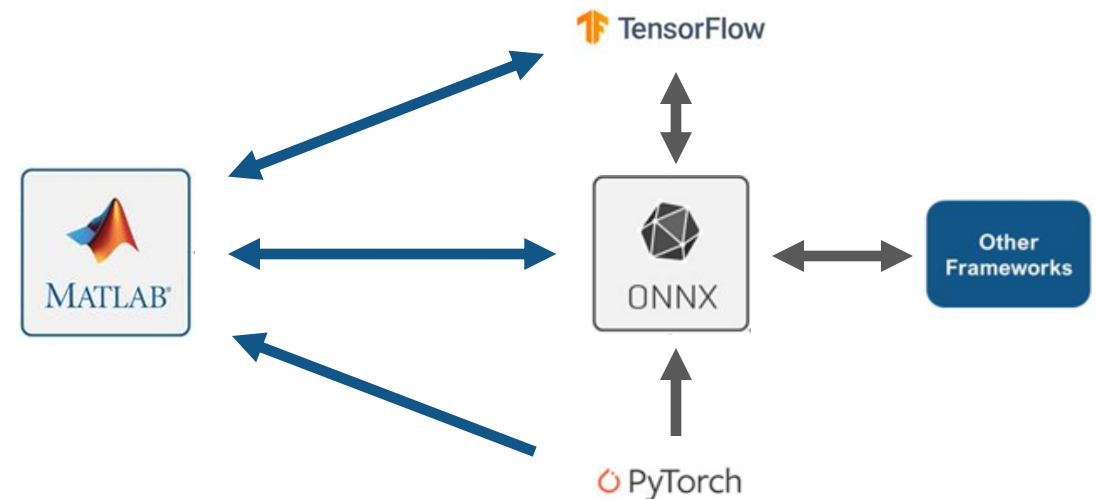
Data Preparation AI Modeling Simulation & Test Deployment

Session opened, engineBlade_ROMsession_results.ma

MATLAB interoperates with other frameworks

Framework interoperability bridges the gap between data science, engineering and production

TensorFlow-Keras Import	R2017b
ONNX Converter (Import & Export)	R2018a
TensorFlow Converter (Import)	R2021a
TensorFlow Converter (Export)	R2022b
PyTorch Converter (Import)	R2022b



AI libraries in Simulink are expanding to include more AI blocks for more applications

Specialized

Audio Toolbox

Computer Vision Toolbox

AI Core

Statistics and Machine Learning Toolbox

Deep Learning Toolbox

System Identification Toolbox

Integration of trained AI models into Simulink

The screenshot displays the MATLAB R2023b Live Editor interface. The top menu bar includes HOME, PLOTS, APPS, LIVE EDITOR, INSERT, and VIEW. The current folder is C:\Documents > ROMSeminar > SimulationAndCodeGeneration. The workspace shows variables: trainingOutput_lstm (1x1 struct) and trainingOutput_nss (1x1 struct). The code editor displays the following function definition:

```

1 function output = Experiment2_training1(params,monitor)
2
3
4
5
6 % TestSplit - For multiple data sets the percentage of data sets to use
7 % for testing, a double in range [0 100]. The test data sets are selected
8 % randomly from the available data sets.
9 testSplit = 20;
10
11 % BatchSize - The number of data points to use when converting signals into
12 % min-batches, i.e., collections of smaller signal segments.
13 batchSize = 20;

```

Data Preparation

AI Modeling

Simulation & Test

Deployment

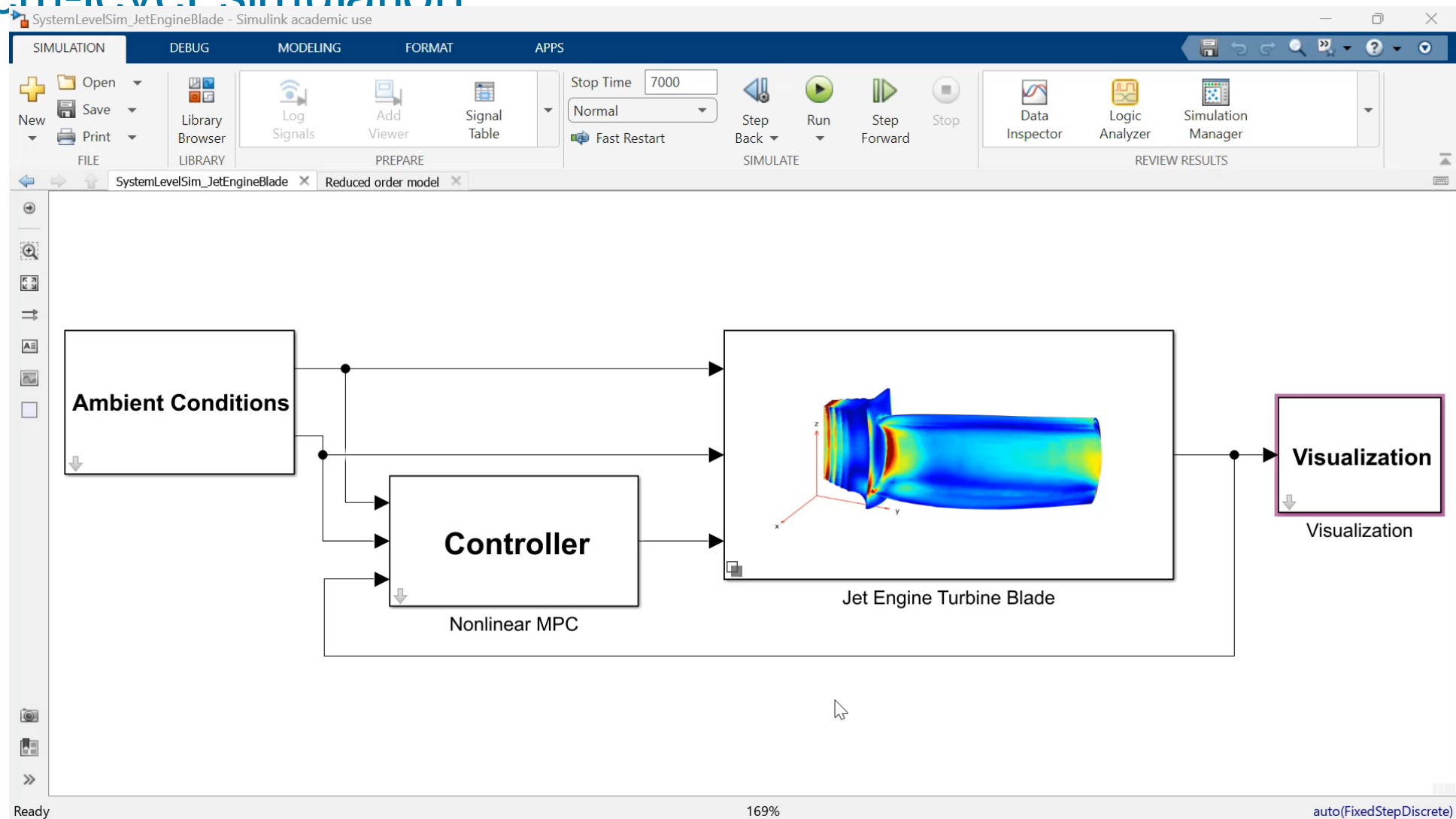
Integration of trained AI models into Simulink

Simulink Profiler

Path	Time Plot (Dark Band = Self Time)	Total Time (s)	Self Time (s)	Number of Calls
▼ JetEngineBlade_AI		17.207	1.807	2014
> LSTM		11.465	0.000	0
Scope1		3.895	3.895	1004
> Neural State Space Model		0.028	0.000	0
From Workspace1		0.008	0.008	1003
Ambient Temperature		0.002	0.002	1003
Cooling Temperature		0.001	0.001	1003
Pressure		0.001	0.001	1003
> Normalize1		0.000	0.000	0
> Denormalize1		0.000	0.000	0
> Denormalize		0.000	0.000	0
> Normalize		0.000	0.000	0

Neural state-space model is approximately 1e6x faster than the FEA model

System-level simulation



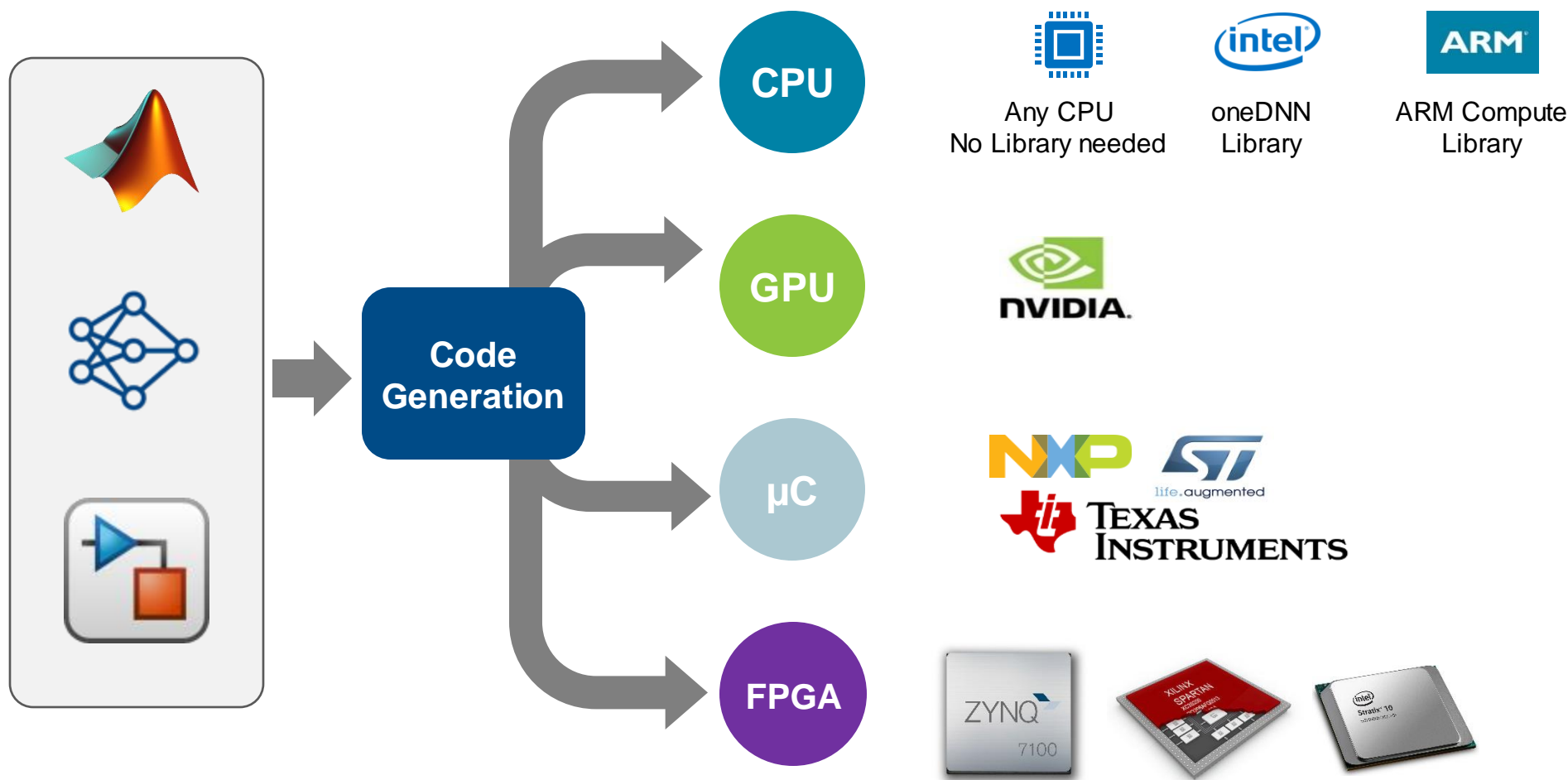
Data Preparation

AI Modeling

Simulation & Test

Deployment

Deploy to target with zero coding errors



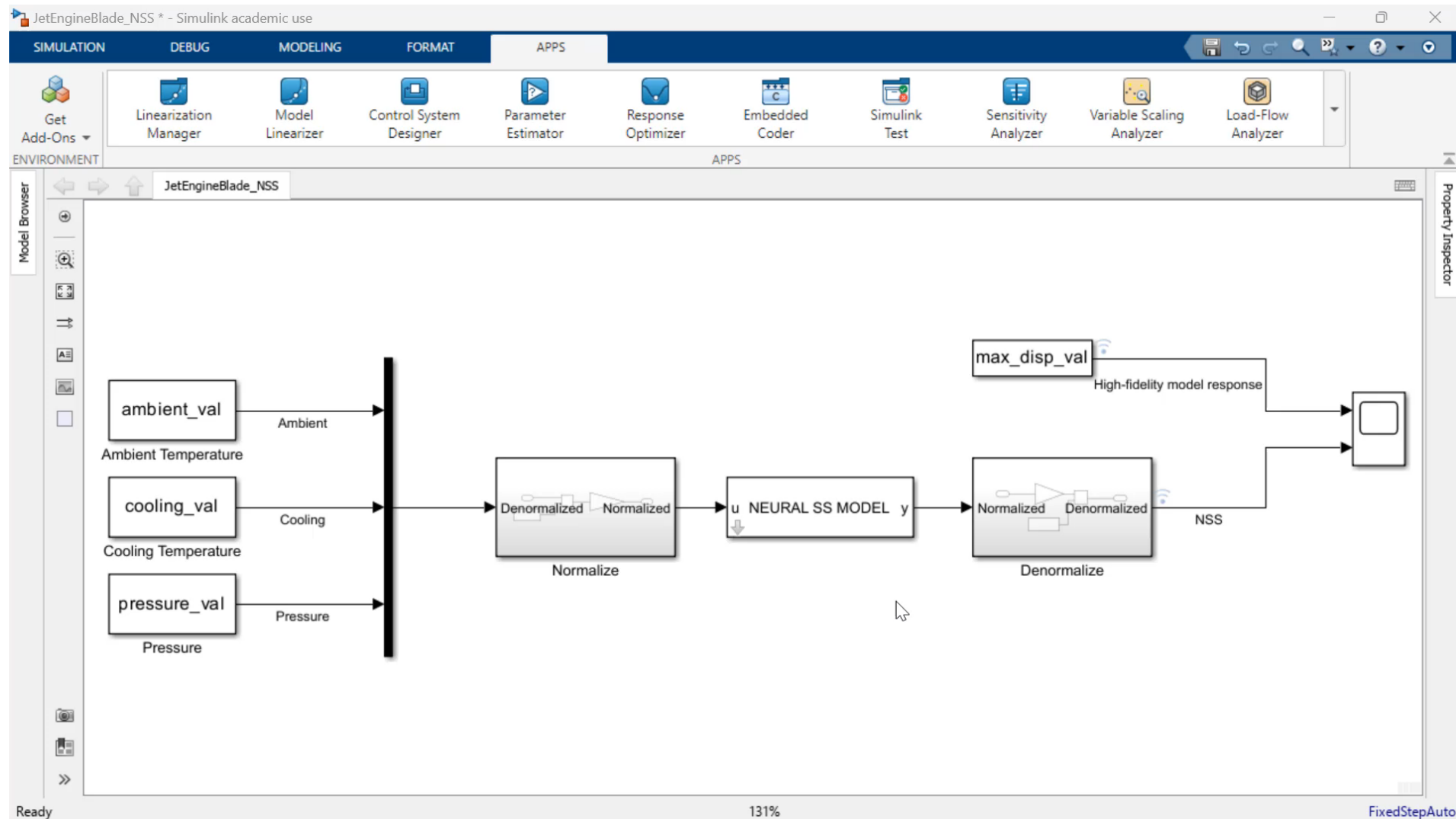
Data Preparation

AI Modeling

Simulation & Test

Deployment

Generate Library-Free C Code for Deep Learning Networks



Data Preparation

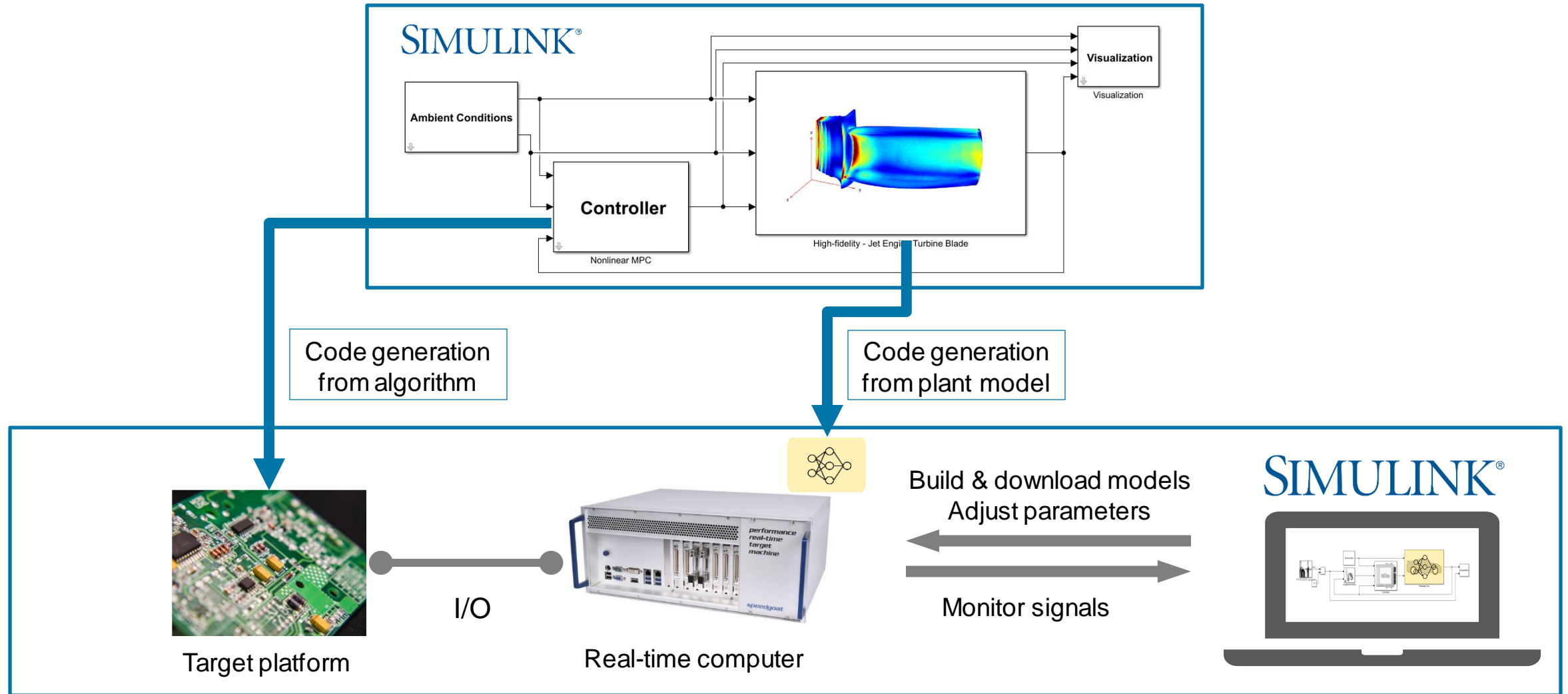
AI Modeling

Simulation & Test

Deployment

Hardware-in-the-loop simulation

System-level integration and test



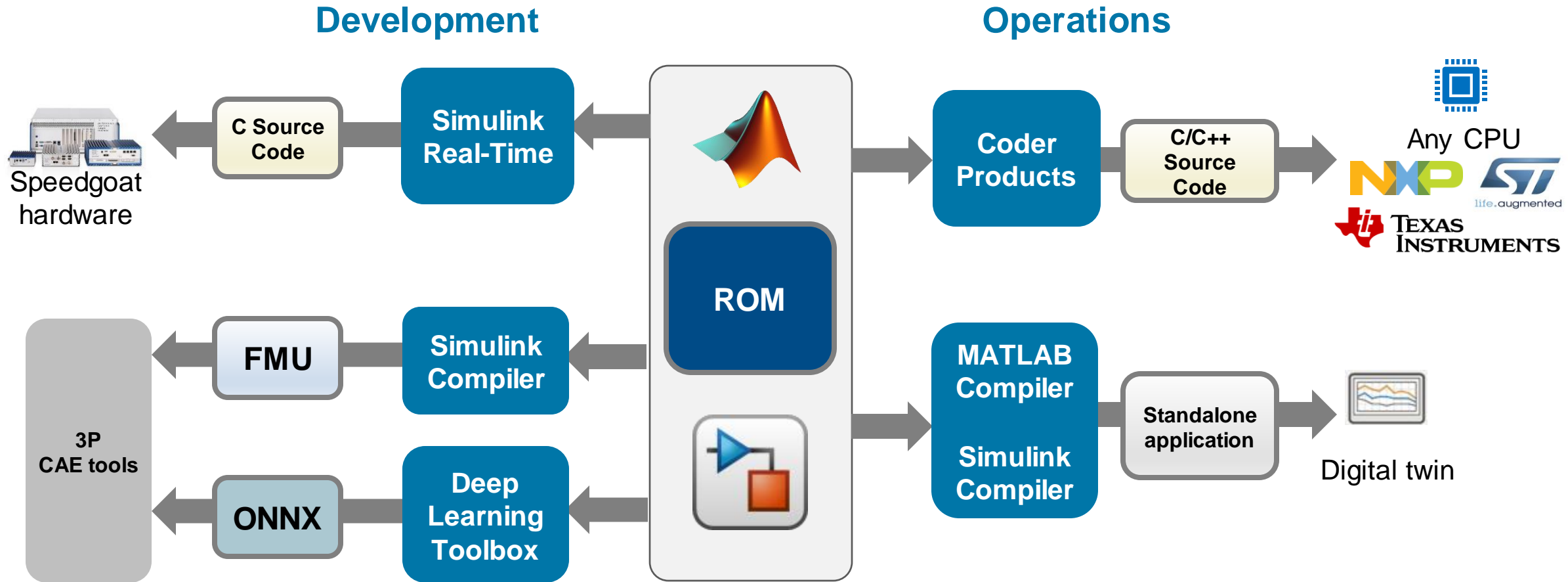
Data Preparation

AI Modeling

Simulation & Test

Deployment

Use ROMs outside of Simulink, for development and operation stages

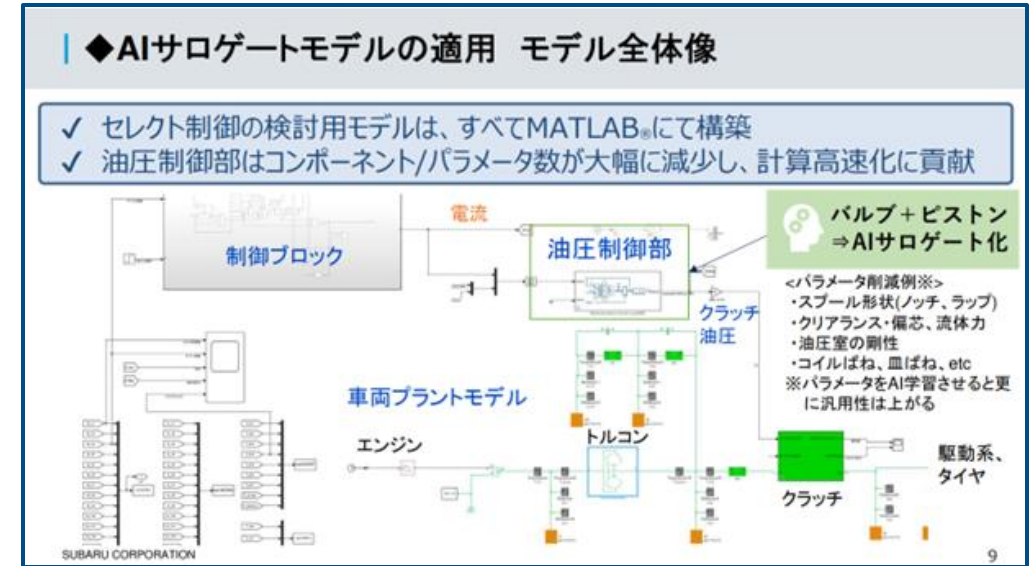


SUBARU Uses AI Surrogate Model to Reduce Transmission Control System Analysis Time

Using MATLAB, engineers at Subaru developed a surrogate AI model to optimize transmission hydraulic systems, achieving a 99% reduction in calculation times compared to the original third-party 1D model.

Key Outcomes/Advantages:

- Achieved a 99% reduction in calculation time compared to the original 1D model
- Constructed AI surrogate model in MATLAB that can reproduce waveforms with arbitrary current, oil temperature, and source pressure readings
- Accurately reproduced waveforms, even in oil temperature ranges where the model has not been trained



The AI surrogate model for studying selective control was built completely in MATLAB.

The AI model can now reproduce waveforms at any source pressure, oil temperature, and current. The calculation time can be significantly reduced while ensuring the accuracy of hydraulic waveforms.

MATLAB EXPO



© 2024 The MathWorks, Inc. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

