





Accelerating the Pace of Engineering and Science

We at MathWorks believe in the importance of engineers and scientists. They increase human knowledge and profoundly improve our standard of living.



We created MATLAB and Simulink to help them do their best work.

MATLAB EXPO

What's New in MATLAB and Simulink **R2020b**

Kevin Cohan
Ed Marquez



2,834



**MATLAB[®]
& SIMULINK[®]**



**Test and Verify
Share and Deploy**



**Deep
Solutions**



Test and Verify
Share and Deploy



Deep
Solutions

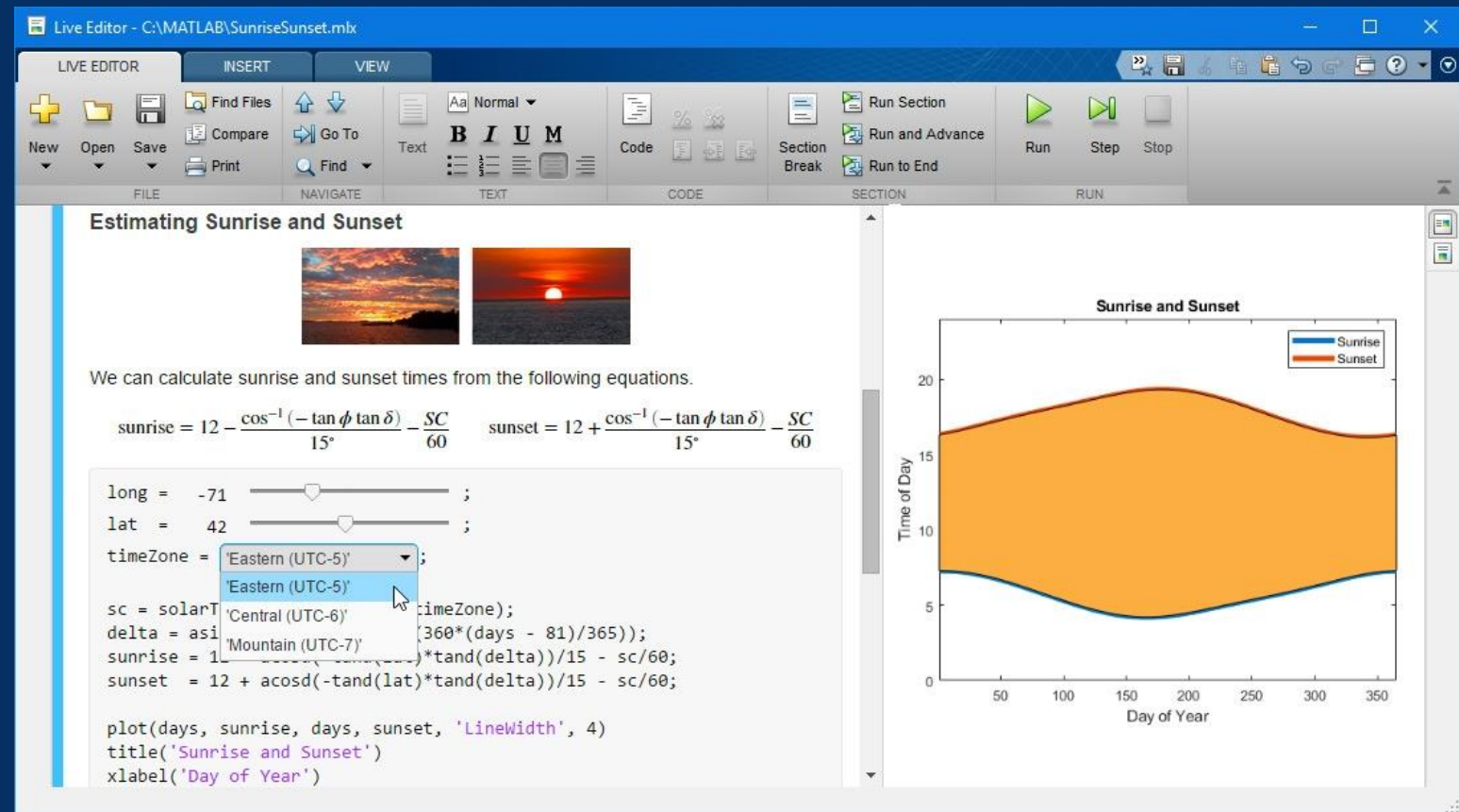
Create executable notebooks for sharing, presenting, teaching

Code + Output + Formatted Text = **Executable Notebook**

Contextual hints
while coding

View interactive outputs
next to the code

Add rich text formatting,
equations, images,
and hyperlinks



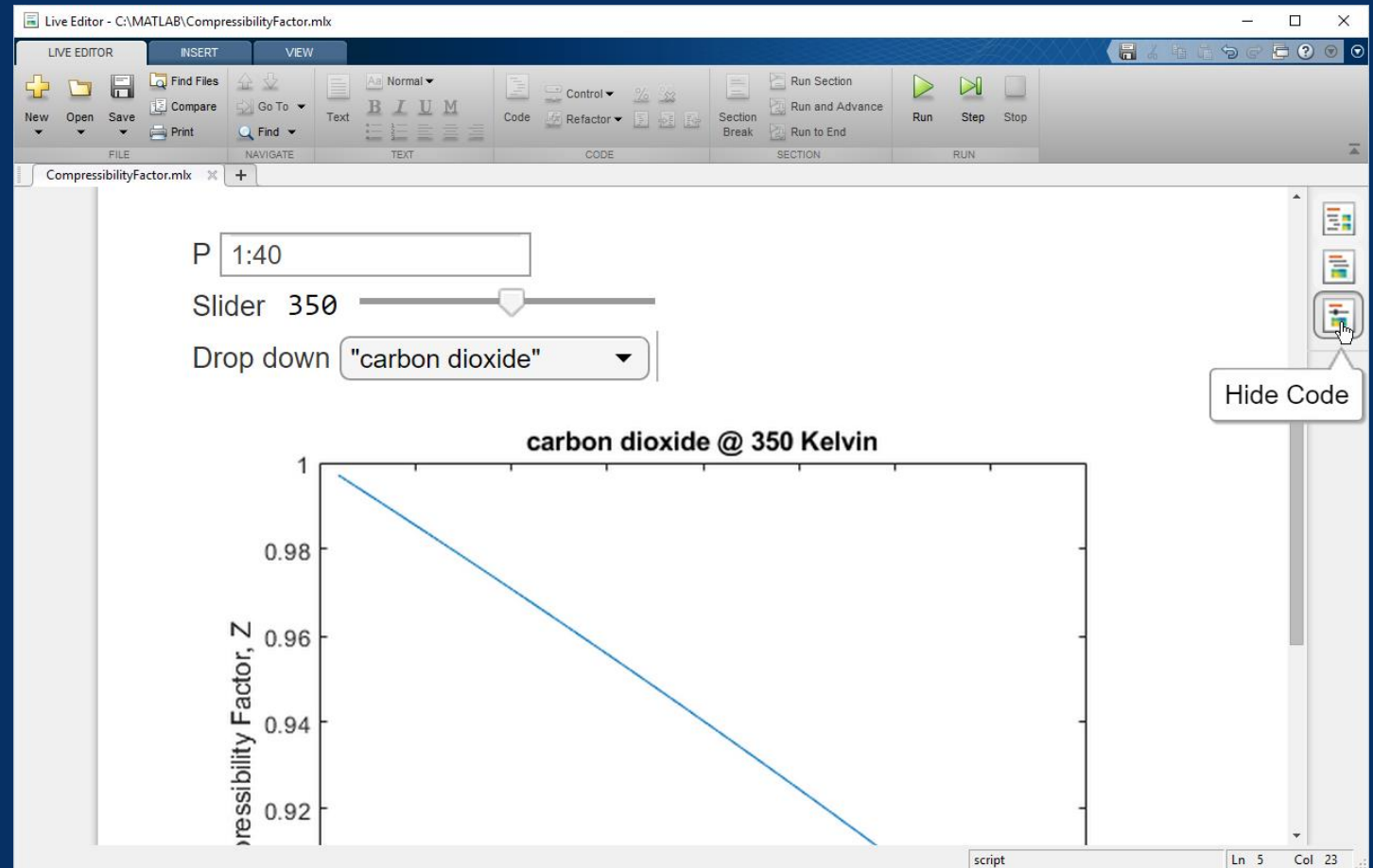
Live Editor

Turn a script into a simple app

Add **interactive controls** to modify script variables

- Numeric sliders
- Drop-down lists
- Edit fields

Hide the code to create simple applications and dashboards



Live Editor

Complete steps interactively

Use **tasks** to explore parameters and options

Automatically generate MATLAB code for the completed task

The screenshot displays the MATLAB Live Editor interface. The top toolbar includes tabs for LIVE EDITOR, INSERT, FIGURE, and VIEW. Below these are various toolbars for file operations, navigation, text formatting, code editing, and execution. The main workspace is divided into three sections:

- Task Configuration:** A panel on the left titled "Select data" with dropdowns for "Input data" (nyiso), "DUNWOD", "X-axis" (nyiso), and "Date". Below it, "Specify method" shows "Cleaning method" as "Fill missing" and "Linear interpolation". "Display results" has checkboxes for "Cleaned data" and "Filled missing entries", both of which are checked.
- Generated MATLAB Code:** A central code editor window containing the following MATLAB code:

```
% Fill missing data
[cleanedData,missingIndices] = fillmissing(nyiso.DUNWOD,'linear',...
    'SamplePoints',nyiso.Date);

% Display results
clf
plot(nyiso.Date,cleanedData,'Color',[0 114 189]/255,'Linewidth',1.5,...
    'DisplayName','Cleaned data')
hold on

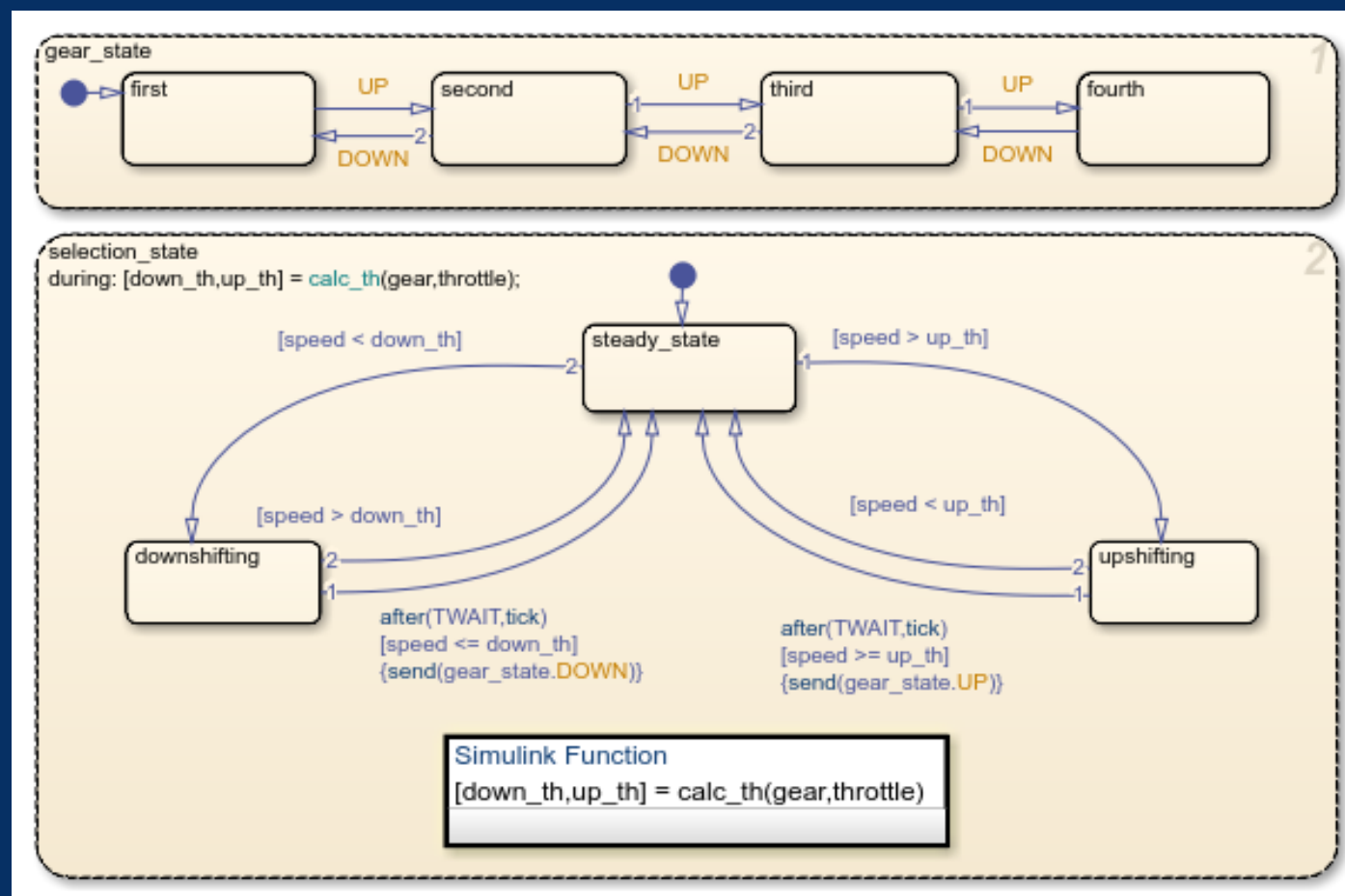
% Plot filled missing entries
plot(nyiso.Date(missingIndices),cleanedData(missingIndices),'.','MarkerSize',12,...
    'Color',[217 83 25]/255,'DisplayName','Filled missing entries')
title(['Number of filled missing entries: ' num2str(nnz(missingIndices))])

hold off
legend
clear missingIndices
```
- Results:** On the right, there is a table of data and a line plot. The table shows 8 rows of data with columns for date, value, and other metrics. The plot, titled "Number of filled missing entries: 4", shows a time series from July 2007 to April 2008. The legend indicates "Cleaned data" (blue line) and "Filled missing entries" (orange dots).

Live Editor

Design decision logic at a higher level of abstraction

Graphically program, debug
and execute state machines



Design decision logic at a higher level of abstraction – in MATLAB

The image displays the MATLAB Stateflow environment. The main window is divided into three panes:

- Editor:** Contains a script for a highway simulator. The script includes comments and code for resetting the random number generator, closing the scenario, opening the chart, creating the road environment, running the simulation, and cleaning up.
- Simulation Window:** A 3D visualization of a highway scenario with several cars (orange, blue, green) on a road.
- Symbols Table:** A table listing variables and their values.

Script Code:

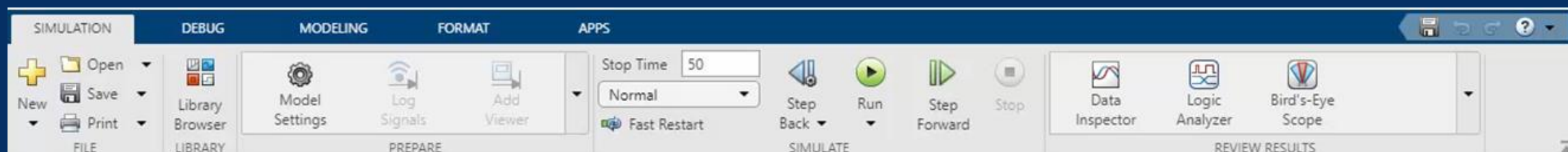
```
1 %% Highway Simulator
2
3 % Reset random number generator to default values.
4
5 rngPrev = rng(0);
6
7 % Close Highway Scenario from previous run
8
9 if ~isempty(findobj('Name','Highway Scenario'))
10     close 'Highway Scenario';
11 end
12
13 % Open chart if Stateflow license exists
14
15 if license('test','stateflow')
16     % edit sf_driver.sfx;
17 end
18
19 %% Create road environment
20
21 numCars = 20;
22
23 scenario = HighwayScenario(...
24     'NumCars',numCars,...
25     'NumLanes',5,...
26     'LaneWidth',3,...
27     'Dt',1e-1,...
28     'AnimateChart',true,...
29     'Plot',true,...
30     'AverageDesiredSpeed',20,...
31     'Sigma',3,...
32     'EgoDesiredSpeed',30,...
33     'NumUserControlledCars',0);
34
35 %% Run simulation
36
37 for i = 1:5000
38     scenario.step();
39 end
40
41 %% Clean up
42
43 for i = 1:numCars
44     delete(scenario.Drivers(i));
45 end
46
47 delete(scenario);
48
49 rng(rngPrev);
```

Symbols Table:

TYPE	NAME	VALUE
double	lWidth	3
double	target	-3
double	me	1
double	deltalane	0
double	laneCenters	-6 -3 0
double	myLane	4
double	myPos	106.49
double	width	0
double	zoneCar	1
double	isZoneOccupied	1
double	positions	zeros(1
double	frontF	0.9
double	errLane	0
double	topLane	2
double	isLaneChanging	0
double	delay	13
double	numCars	
double	maxSpeed	30
double	myVel	18.644
double	velocities	[1 0 0]
double	slowCar	0
function	checkZone	
function	getLane	
function	getVel	

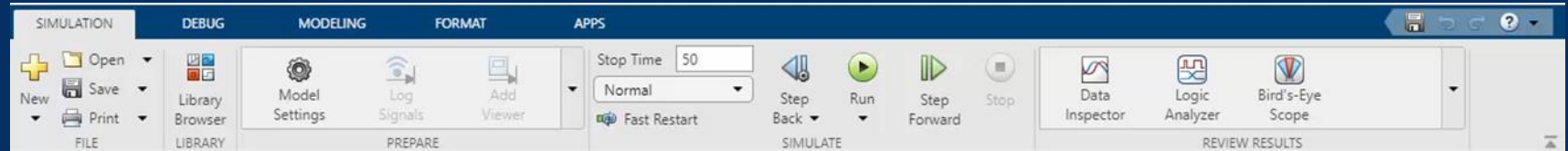
Enable **any engineer** at **any level** to model **any system**

User interfaces

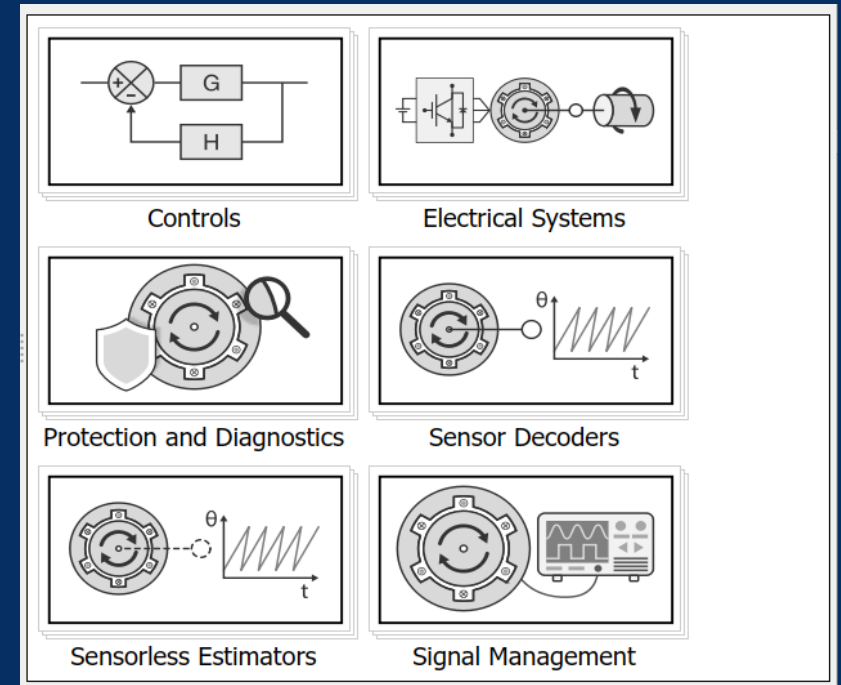
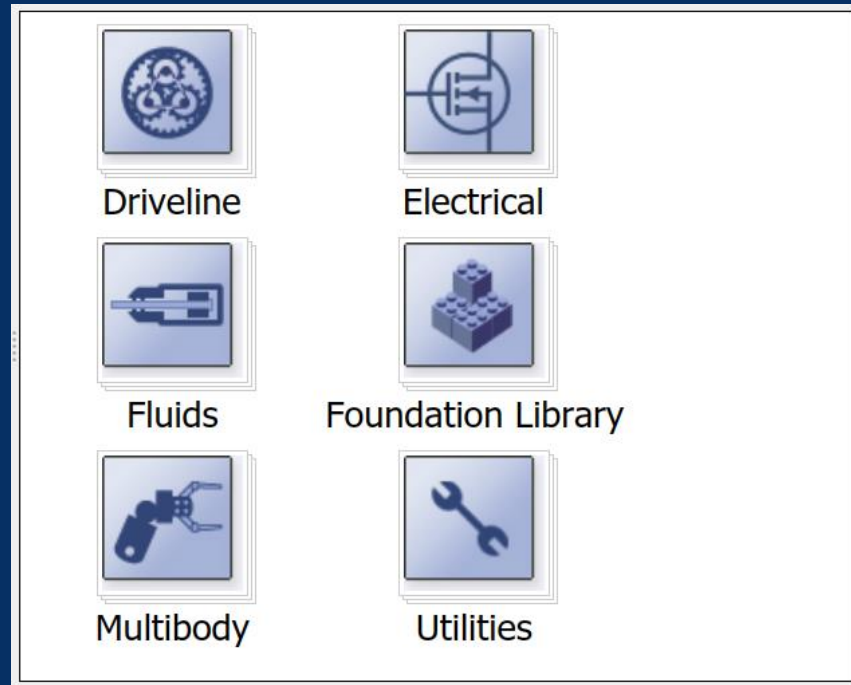


Enable **any engineer** at **any level** to model **any system**

User interfaces



Libraries

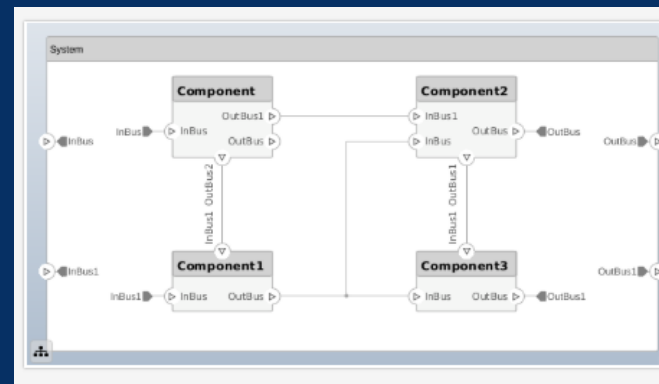
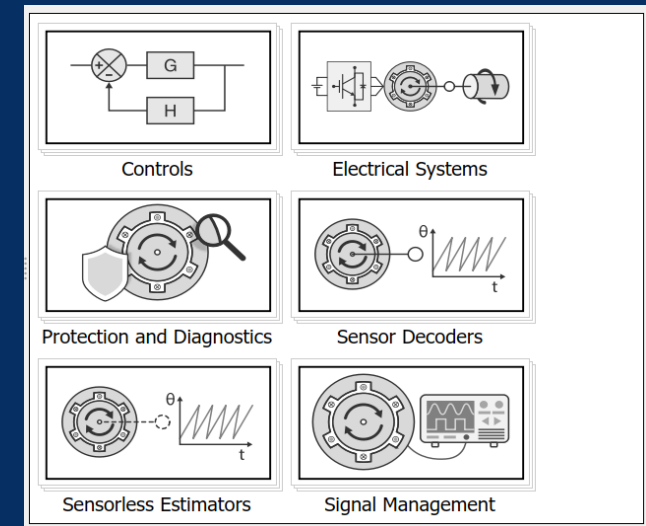
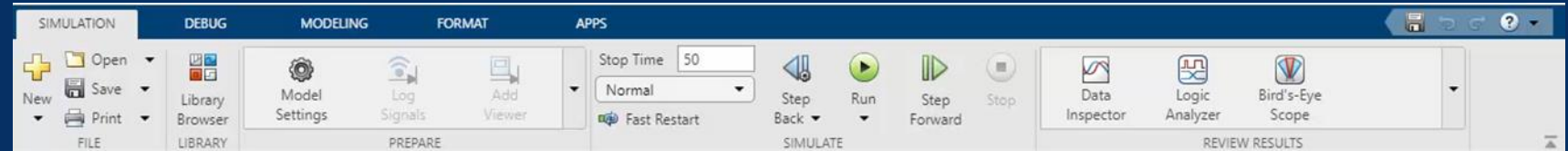


Enable **any engineer** at **any level** to model **any system**

User interfaces

Libraries

Systems engineering



Architecture Model

By The MathWorks, Inc.

Create Model

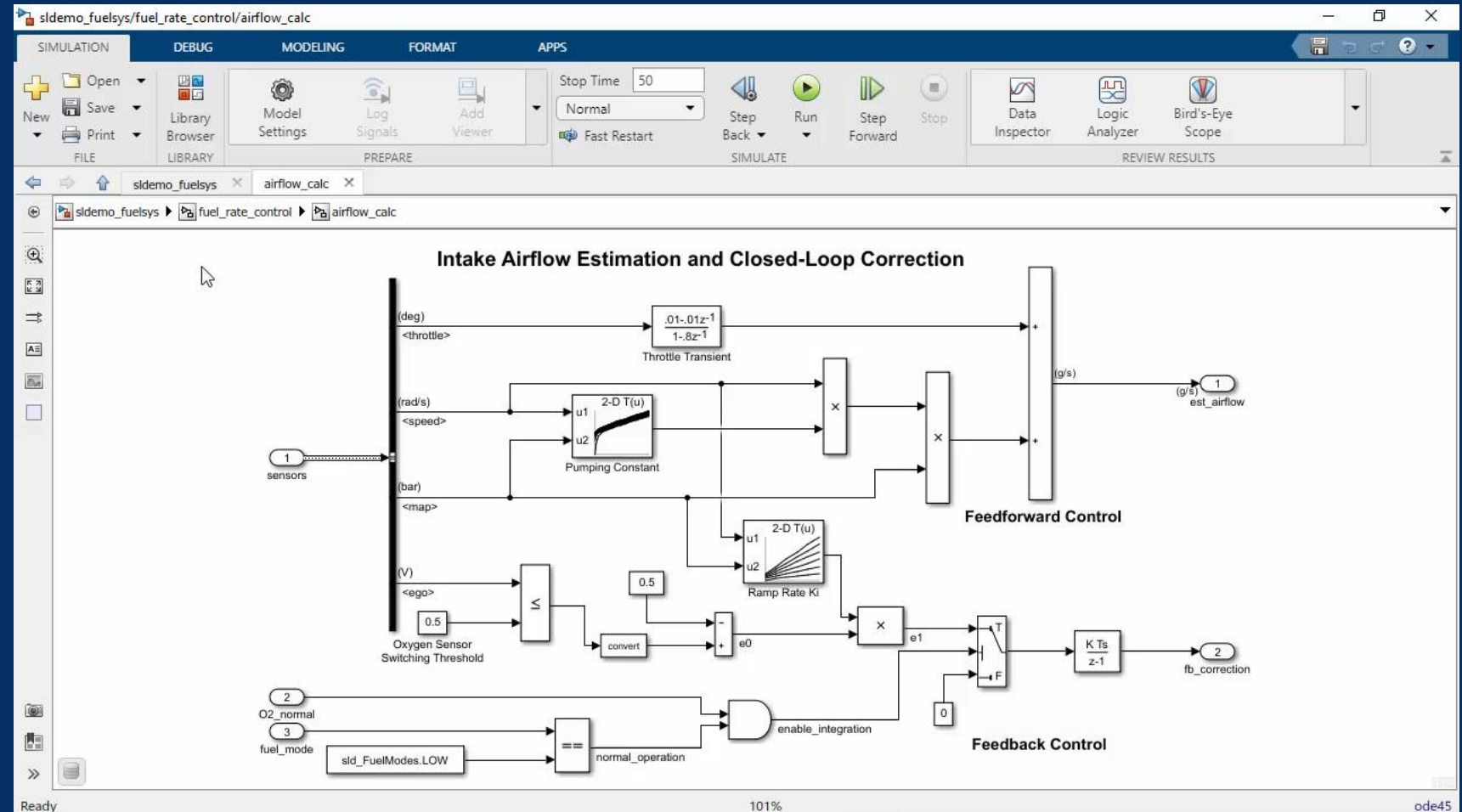
Create an architecture model. Model physical and logical architecture of a system. Create a visual representation with components, ports, and connectors. Specify information exchange between components with interfaces.

Access and discover Simulink capabilities when you need them

User interfaces

Libraries

Systems engineering



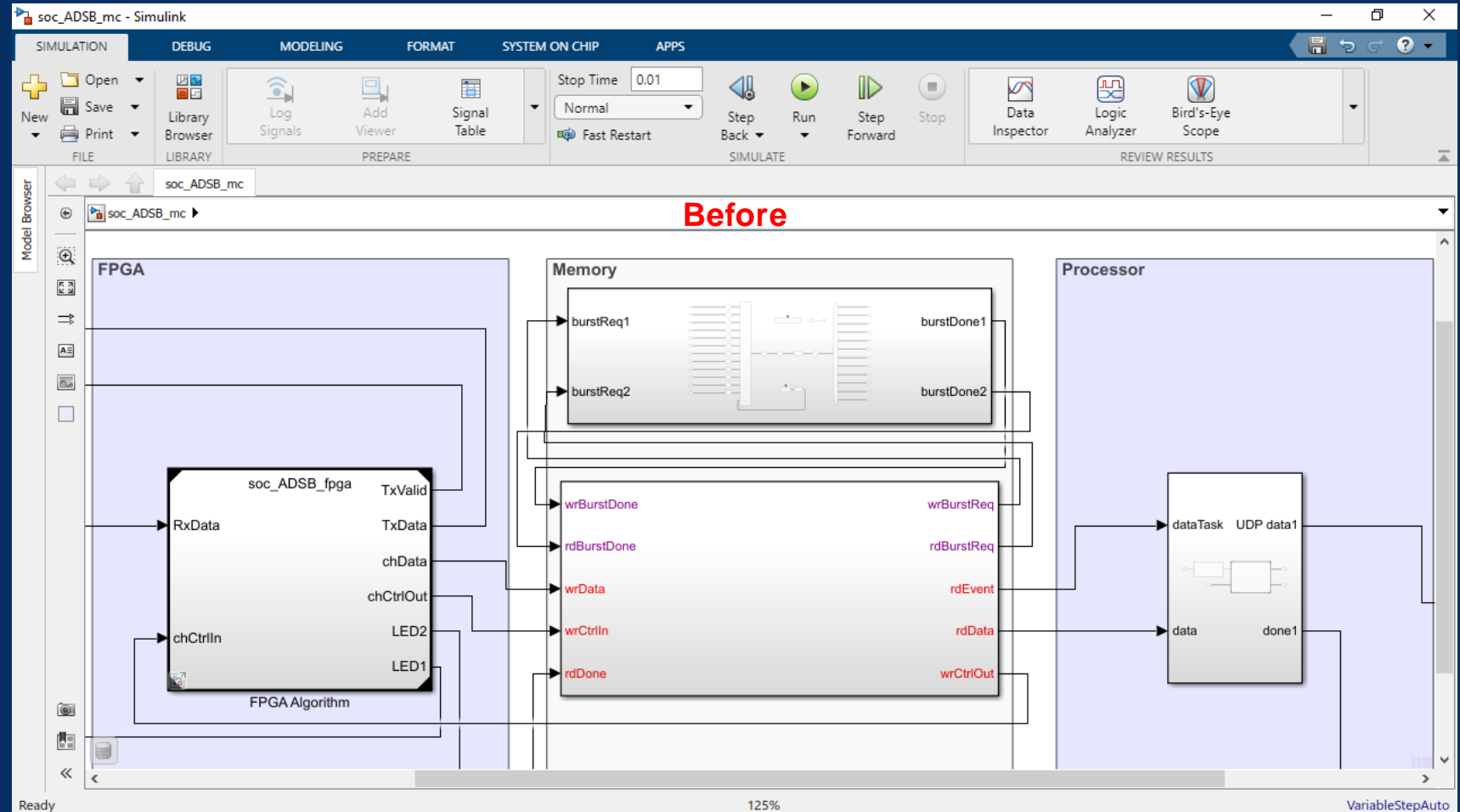
Simulink Toolstrip

Edit at the speed of thought

User interfaces

Libraries

Systems engineering

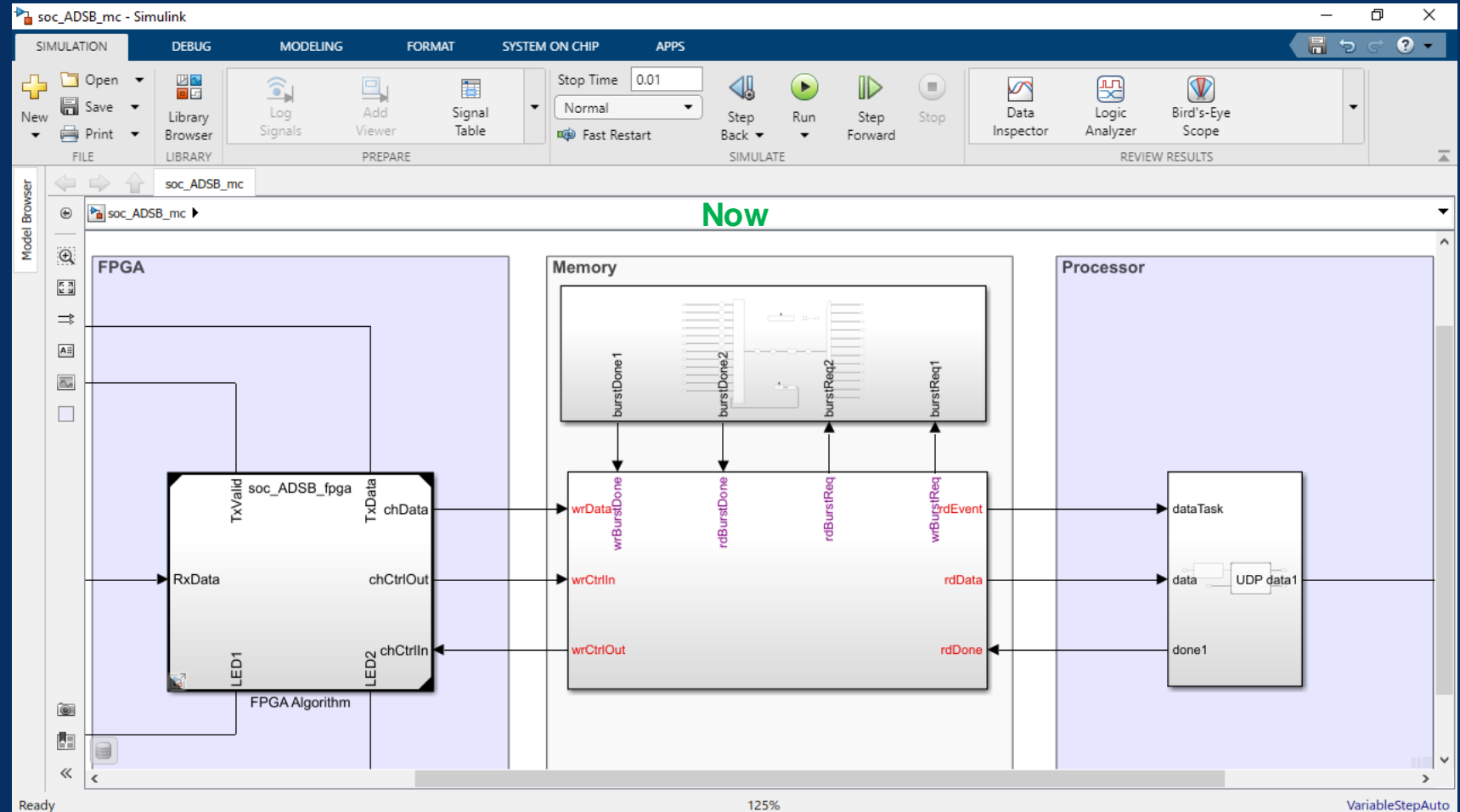


Edit at the speed of thought

User interfaces

Libraries

Systems engineering

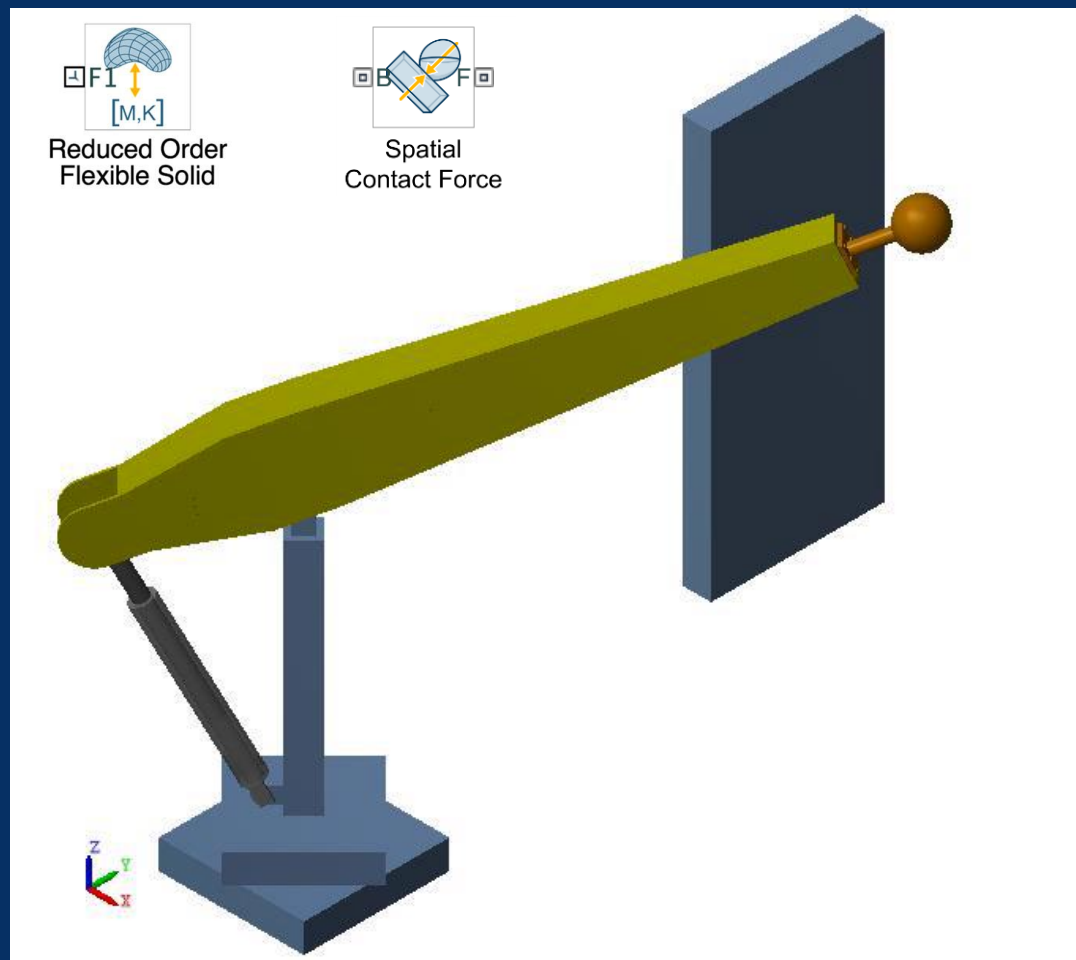


Model deformations and contact between bodies

User interfaces

Libraries – Physical modeling

Systems engineering

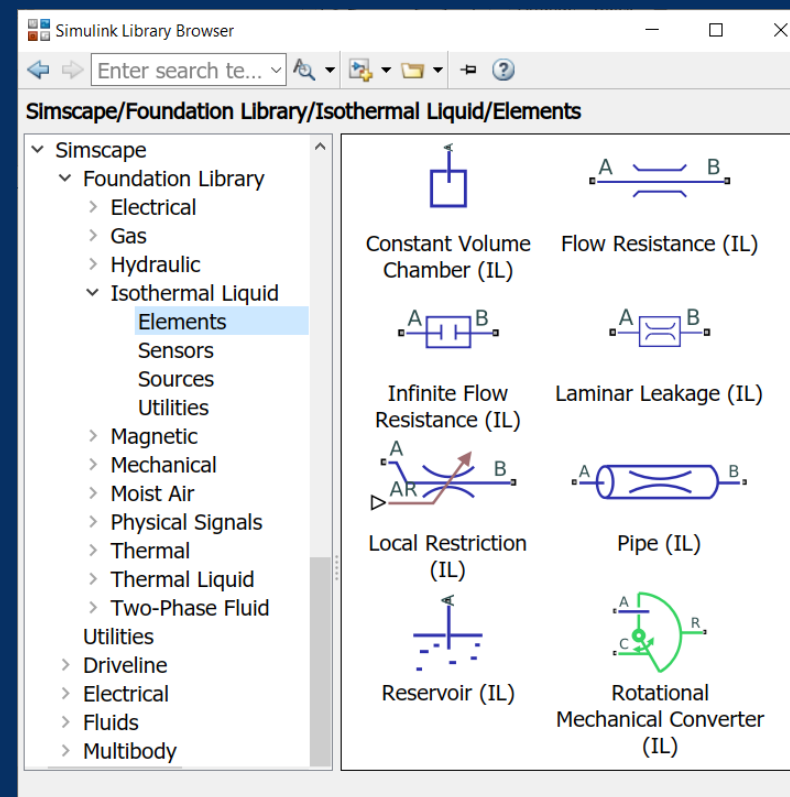


Model fluid power and transport applications

User interfaces

Libraries – Physical modeling

Systems engineering

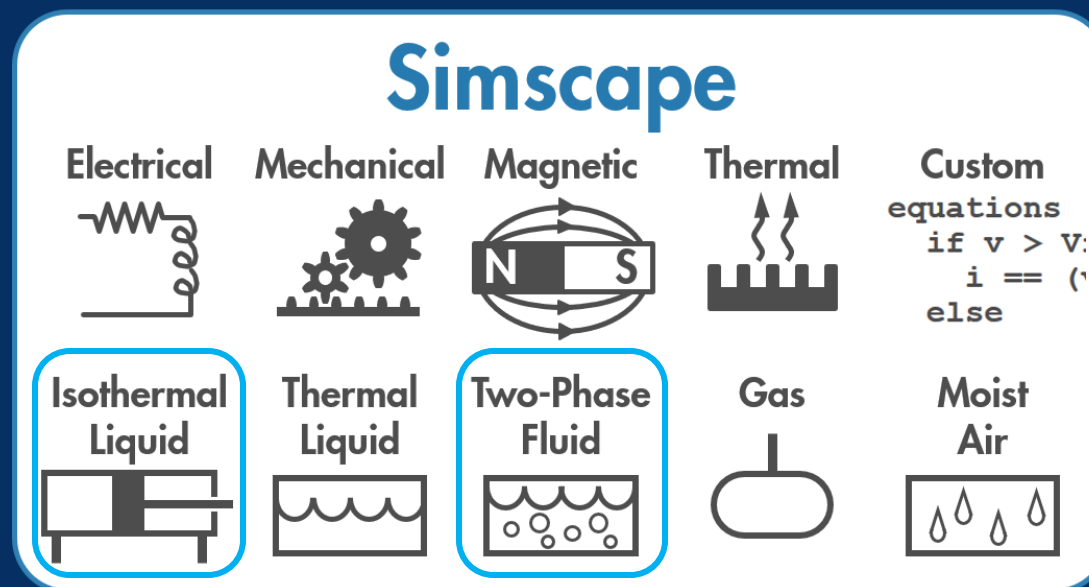


Model fluid power and transport applications

User interfaces

Libraries – Physical modeling

Systems engineering

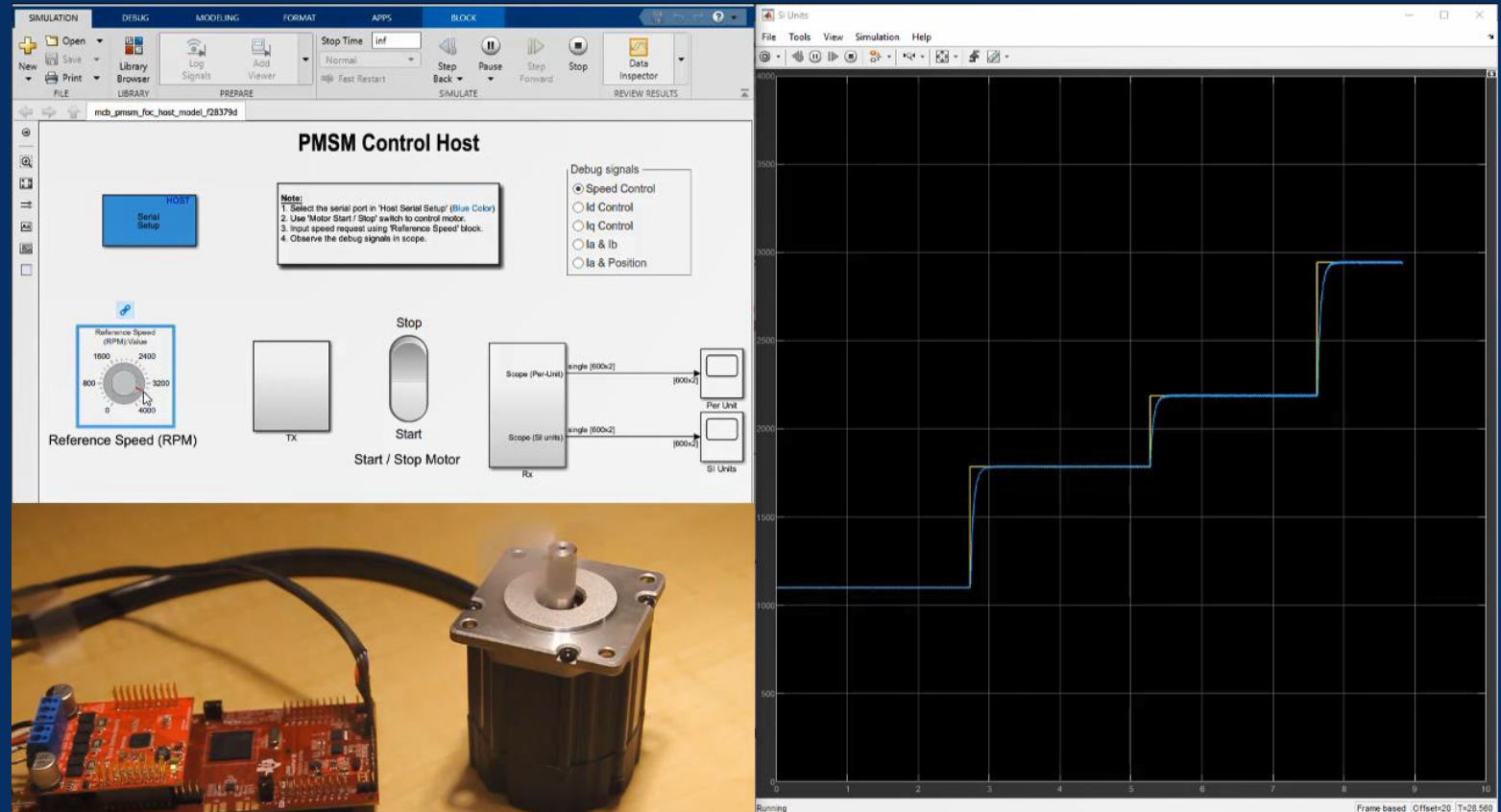


Generate motor control software with just a few clicks

User interfaces

Libraries – Motor control

Systems engineering

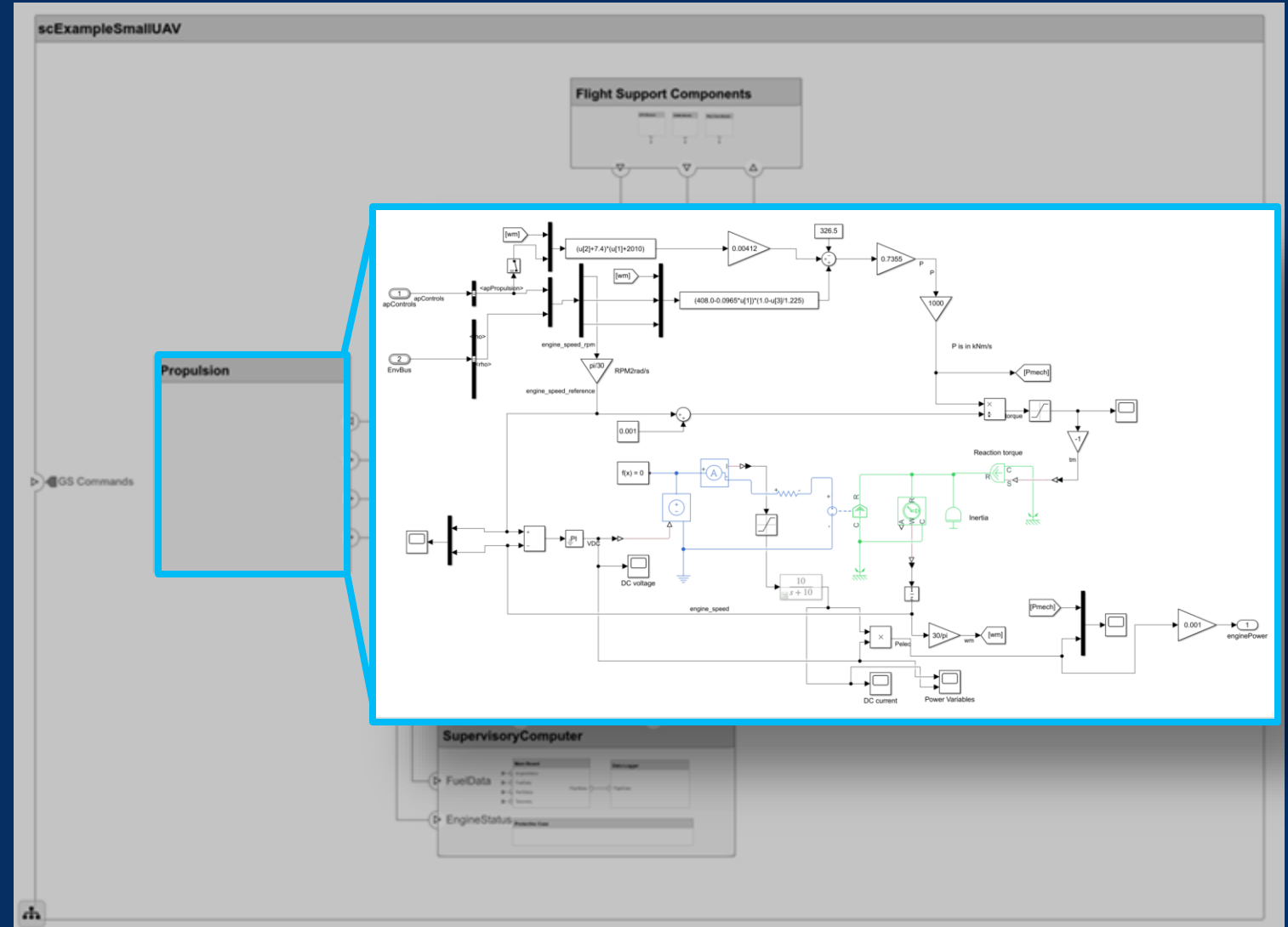


Design and analyze complex system and software architectures

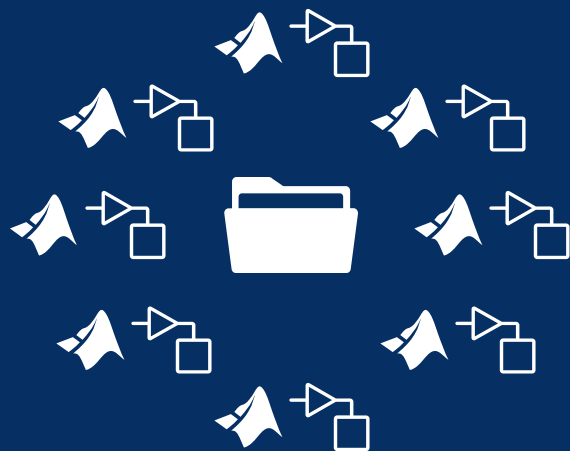
User interfaces

Libraries

Systems engineering



Manage system complexity



**Numerous
Files**



**Team
Collaboration**



**Environment
Configuration**

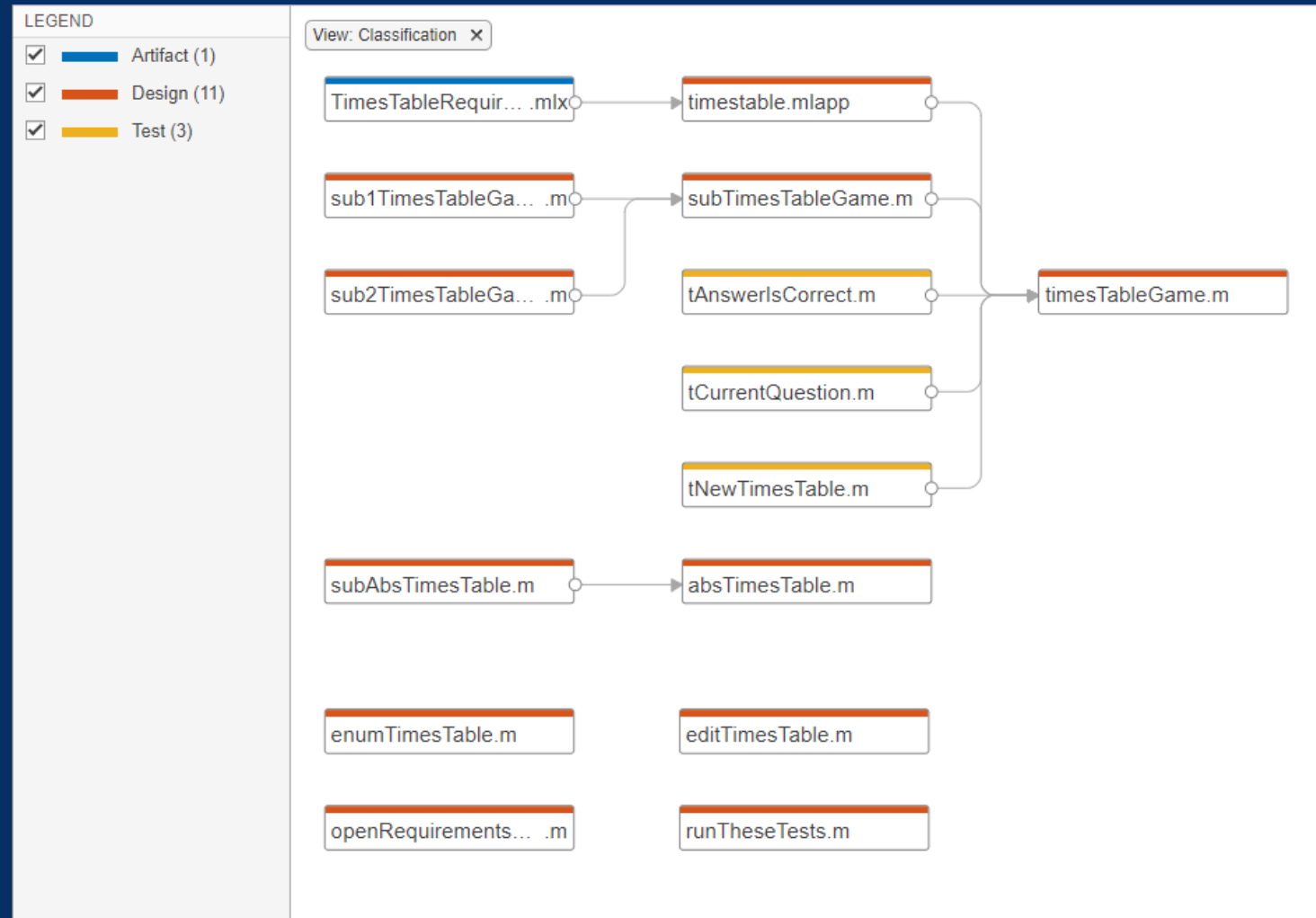
Manage system complexity with projects

Projects in MATLAB and Simulink help you to organize, manage, and share your code and models



Explore file dependencies and impact analysis

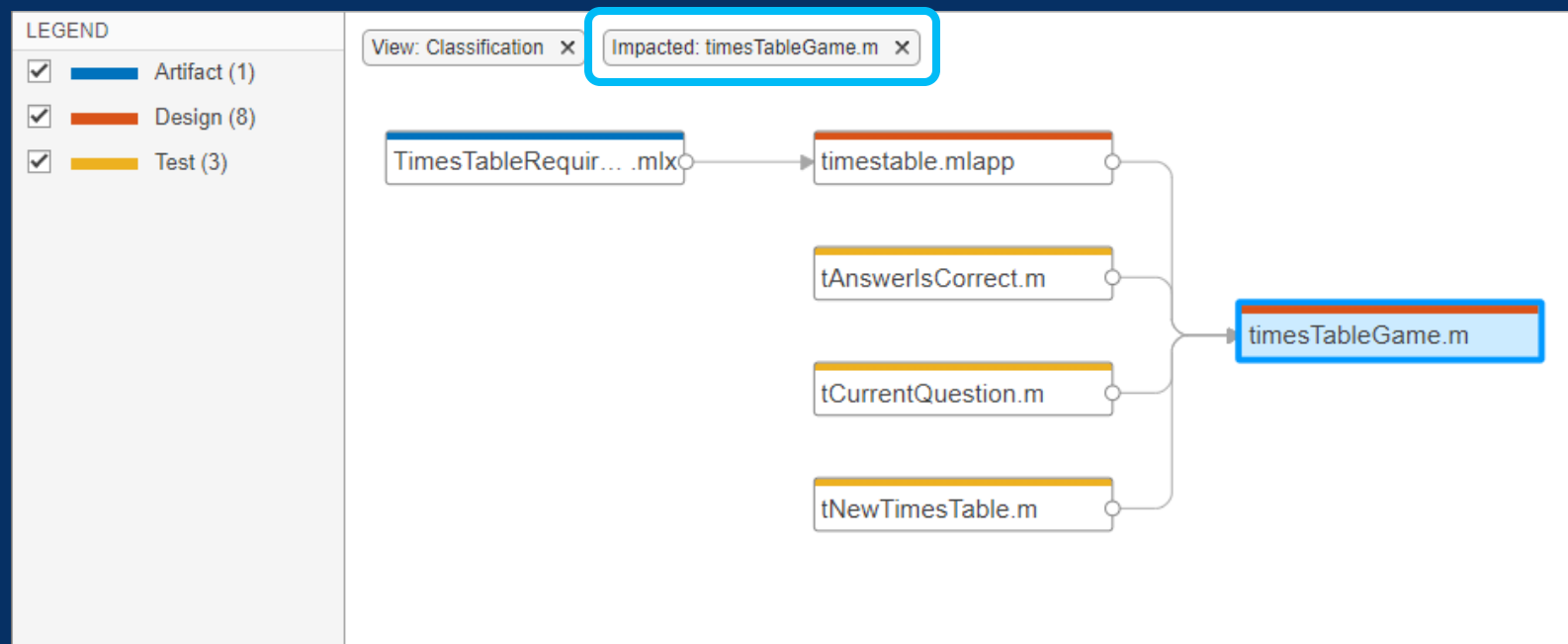
Explore and visualize project structure



Explore file dependencies and impact analysis

Explore and visualize
project structure

Assess how a change
affects other files

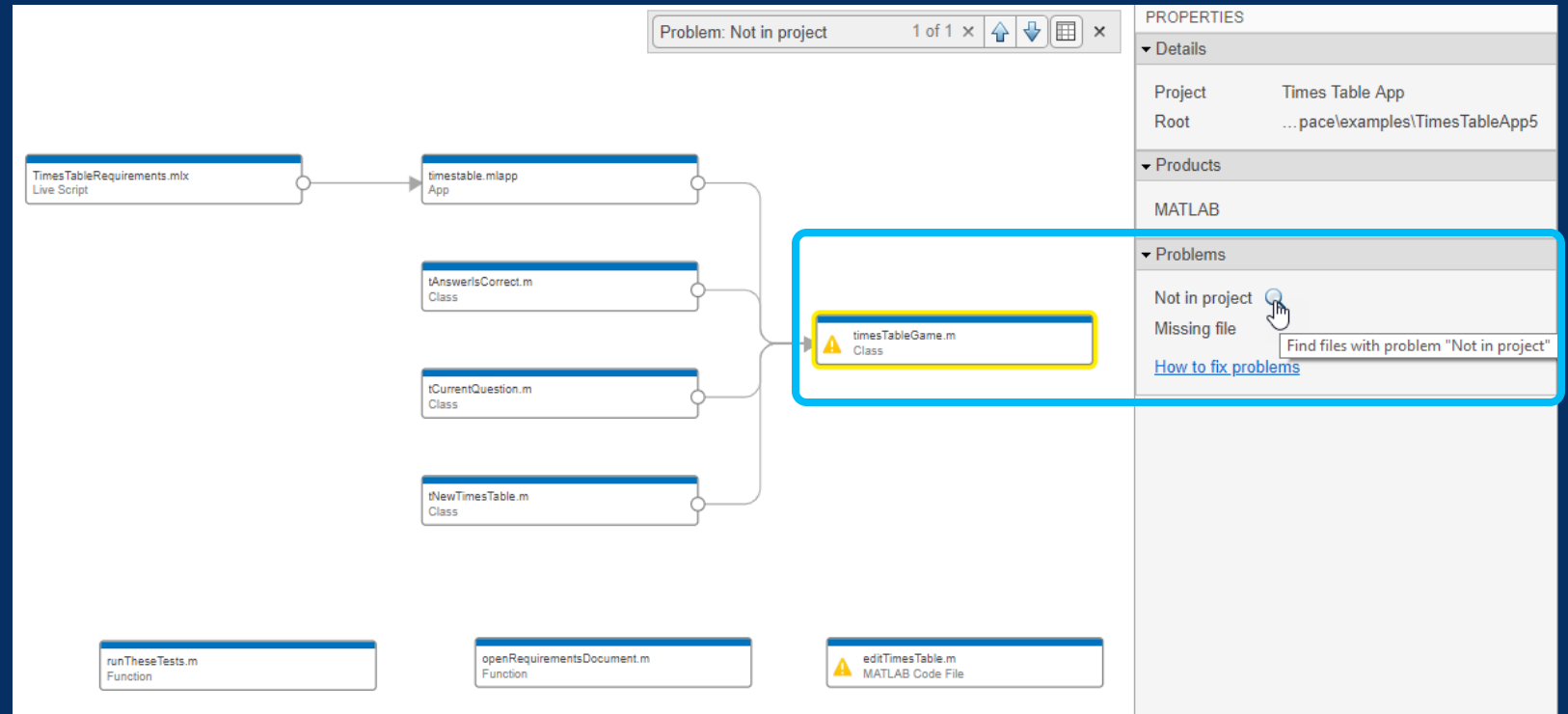


Explore file dependencies and impact analysis

Explore and visualize project structure

Assess how a change affects other files

Find and fix problems



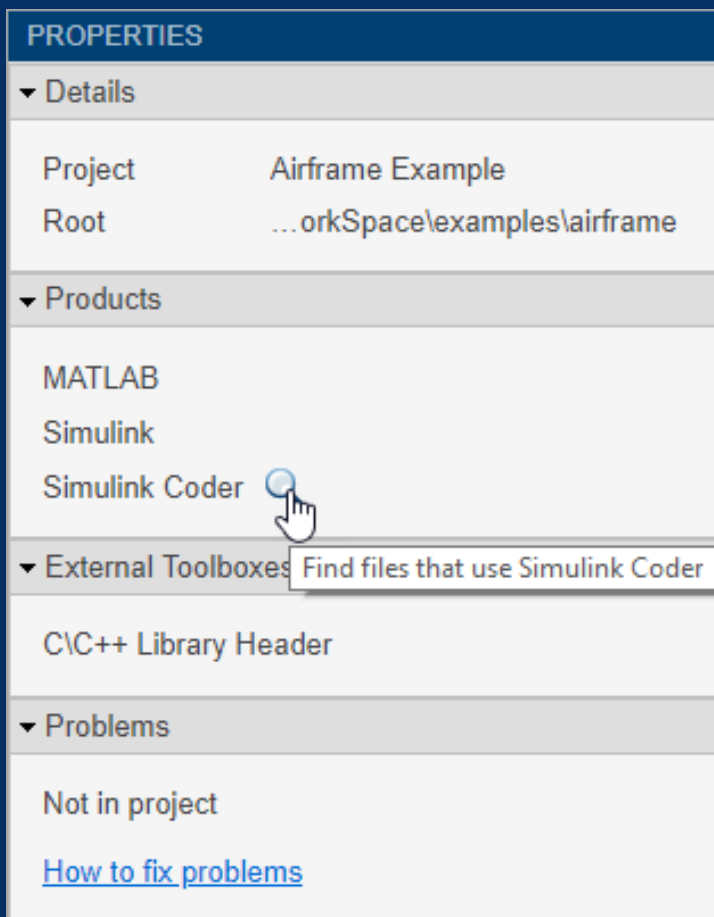
Explore file dependencies and impact analysis

Explore and visualize
project structure

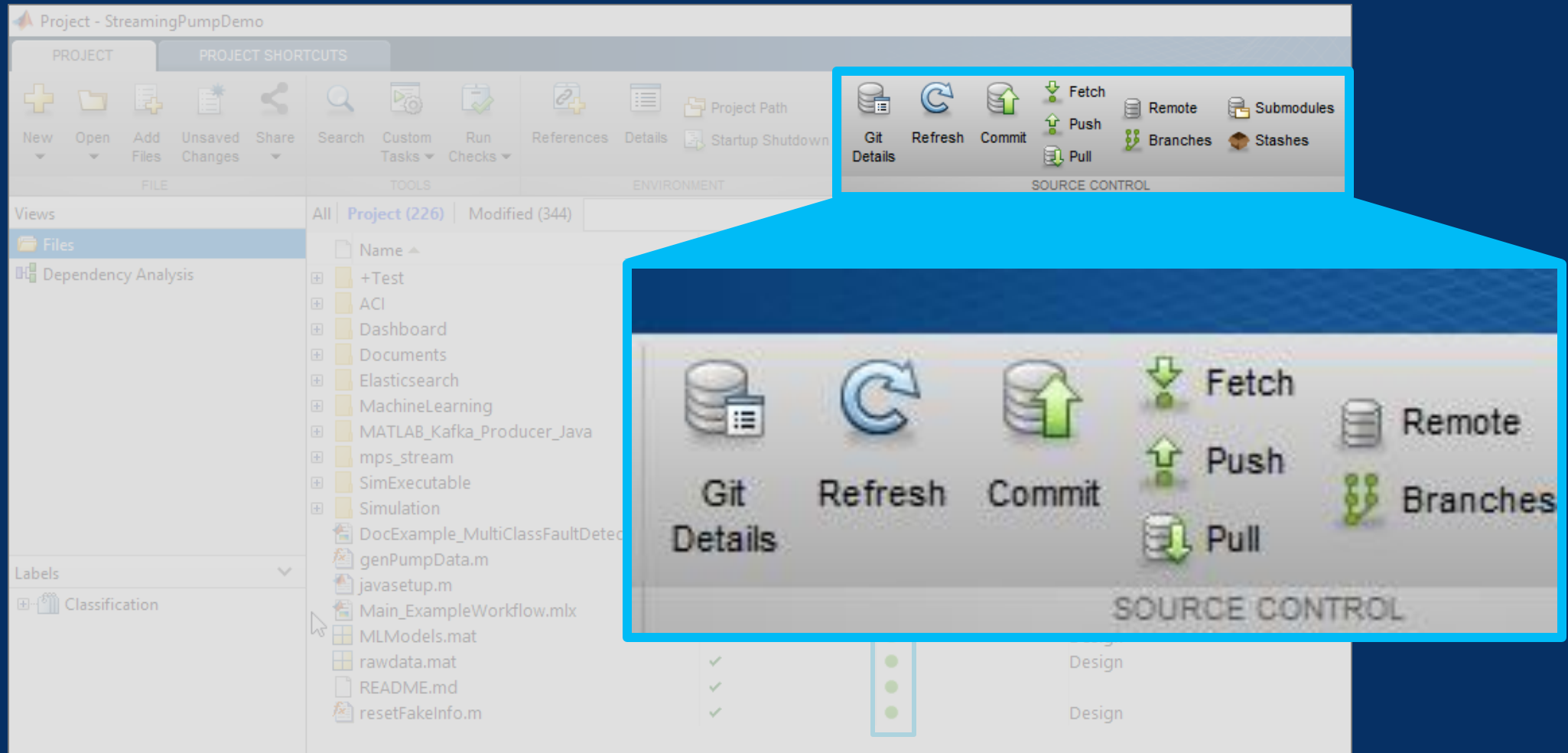
Assess how a change
affects other files

Find and fix problems

Identify required products
and toolboxes



Use source control systems (Git, Subversion) with projects



Access other languages and systems directly from MATLAB

Python

C/C++

Java

Fortran

COM components and ActiveX controls

RESTful, HTTP, and WSDL web services



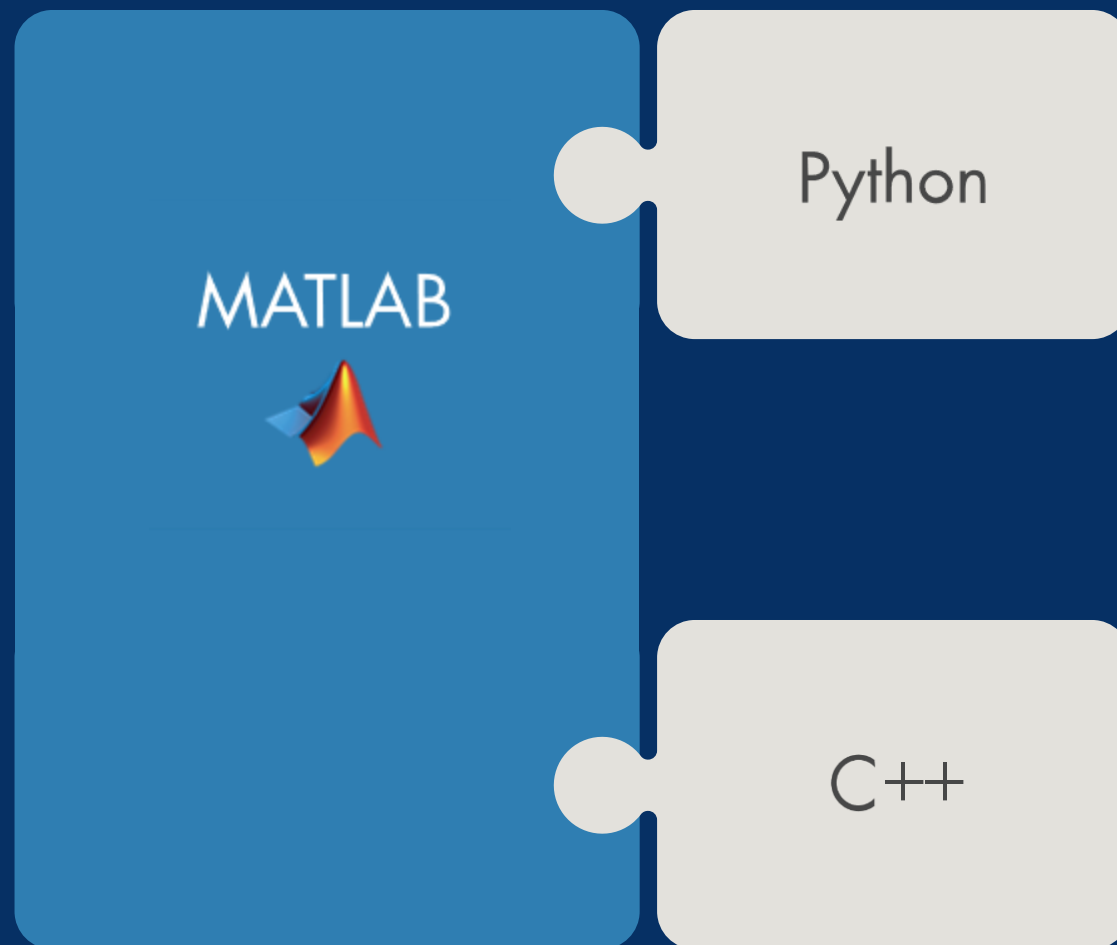
Access other languages and systems directly from MATLAB

Access Python functions out-of-process

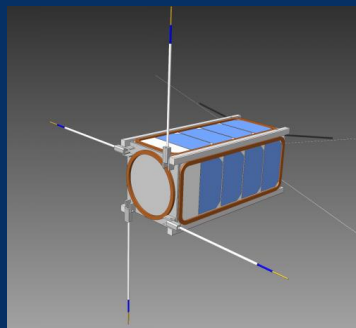
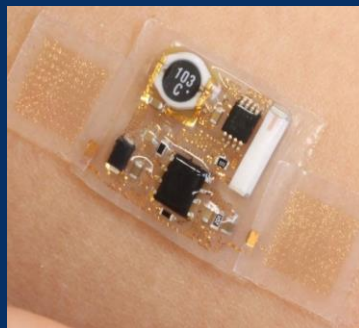
```
pyenv("ExecutionMode","OutOfProcess")  
wrapped = py.textwrap.wrap(T);  
terminate(pyenv)  
pyenv("Version","2.7");  
py.list; % Reload interpreter
```

Call C++ libraries directly from MATLAB

```
retVal = clib.libname.funcname(arg1, arg2, ...)
```



Simulink



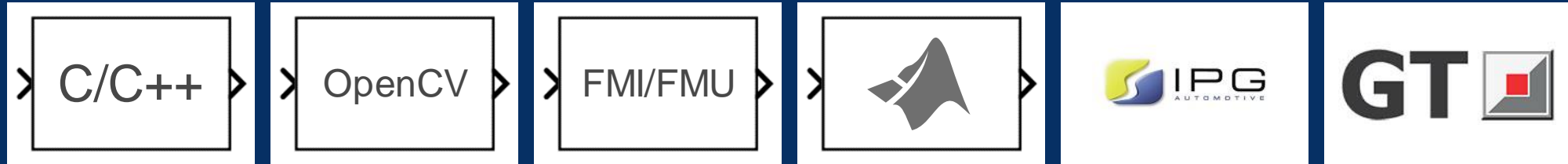
Simulink is the **simulation integration platform**



Simulink 



Simulink is the **simulation integration platform**



Simulink 





**MATLAB®
& SIMULINK®**



**Test and Verify
Share and Deploy**



**Deep
Solutions**

Test and verify your design

Review and analyze traceability between artifacts in one interface

The screenshot displays the Traceability Matrix tool, which is used for reviewing and analyzing traceability between artifacts in one interface. The interface includes a 'FILTER PANEL' on the left and a main table showing the traceability matrix.

Traceability Matrix

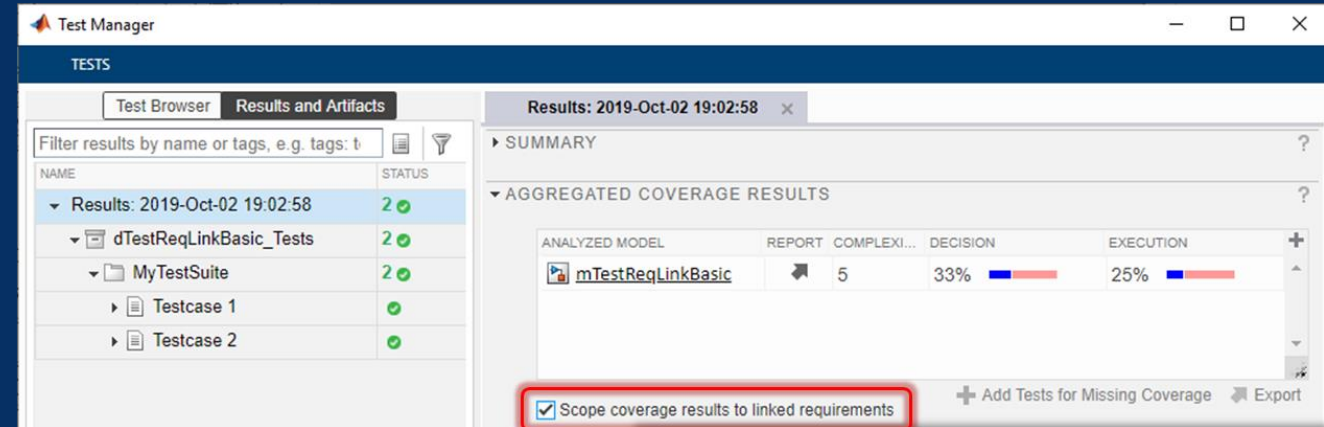
	scExampleSmallUAVModel	Supervisory Computer	Flight Support Components	Flight Computer	Main Board	Protective Case	Telemetry Antenna	Airframe	Payload	Imbal	Payload Retraction Sy	Payload Data Link	Propulsion	Power Module	Prop	Engine	Fuel System	Fuel Level Sensors	Fuel Pump	Fuel Tank	Pressure Regulator	Fuel Filter
scExampleSmallUAVModel																						
#1 Aircraft Capabilities																						
#3 Airworthiness																						
#11 Communications																						
#12 Flight Control																						
#13 Payload																						
#14 Payload Capabilities																						
#15 Construction																						
#20 Modularity																						

Traceability Matrix

Test and verify your design

Review and analyze traceability between artifacts in one interface

Scope model coverage to requirements-based tests (RBT)



MultiPortSwitch block "[MPSwitch1](#)"

Requirement Testing Details

Implemented Requirements	Verified by Tests	Associated Runs
Requirement 1	Testcase 1	T1

Metric	Coverage
Cyclomatic Complexity	2
Decision	33% (1/3) decision outcomes
Execution	100% (1/1) objective outcomes

Decisions analyzed

Decision	Coverage
truncated input value	33%
= 1 (output is from input port 1)	51/51 T1
= 2 (output is from input port 2)	0/51 T1
= *,3 (output is from input port 3)	0/51 T1

Hit by linked RBT -- **Satisfied**

Hit, but not by linked RBT -- **Unsatisfied**

Test and verify your design

Review and analyze traceability
between artifacts in one interface

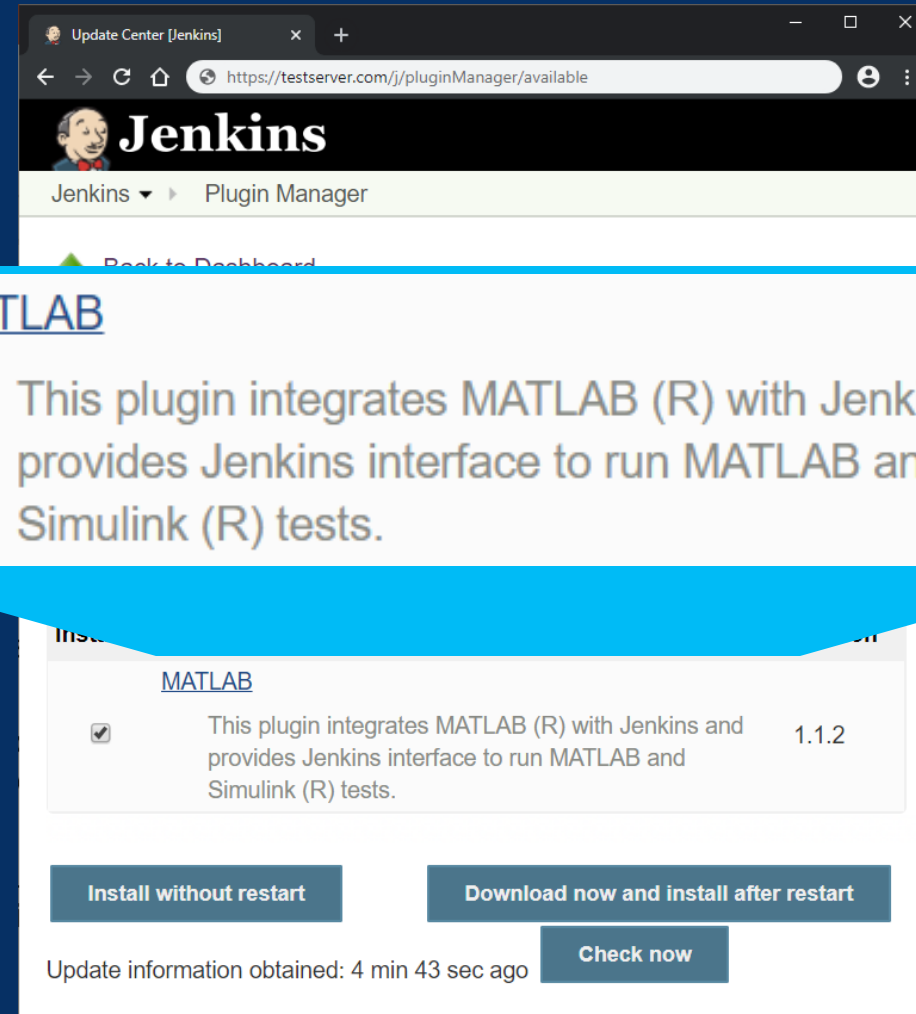
Scope model coverage to
requirements-based tests (RBT)

Use full physical RAM in target computer
with the 64-bit real-time operating system



Use Jenkins servers to automatically run and test your project

Install MATLAB Plugin for Jenkins directly from the Jenkins Plugin Manager



Code verification using Polyspace

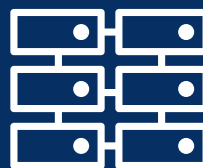


Desktop

Automate code verification using Polyspace



Desktop



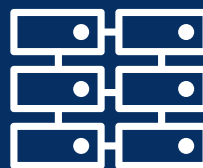
Server Computer



Automate code verification and share results using Polyspace



Desktop



Server Computer



Web Interface





**MATLAB®
& SIMULINK®**



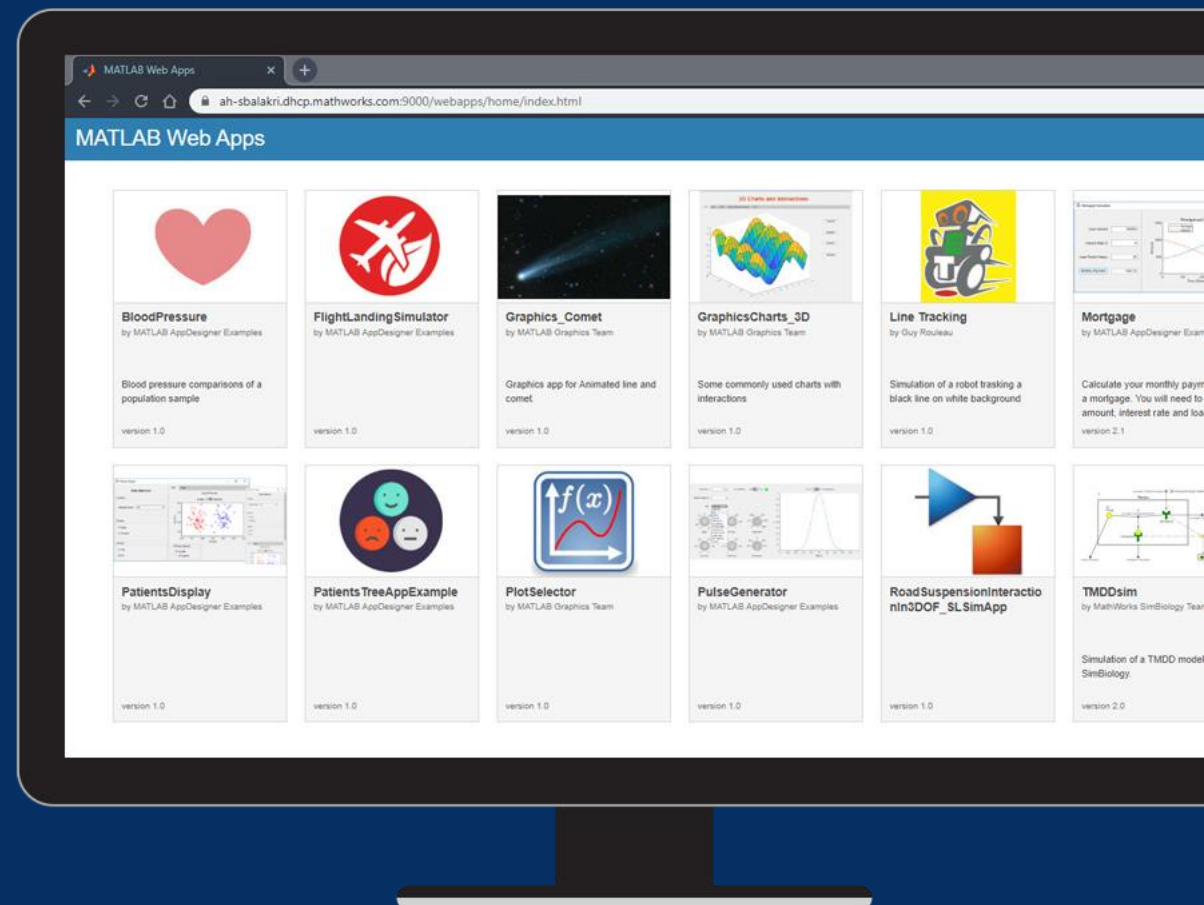
**Test and Verify
Share and Deploy**



**Deep
Solutions**

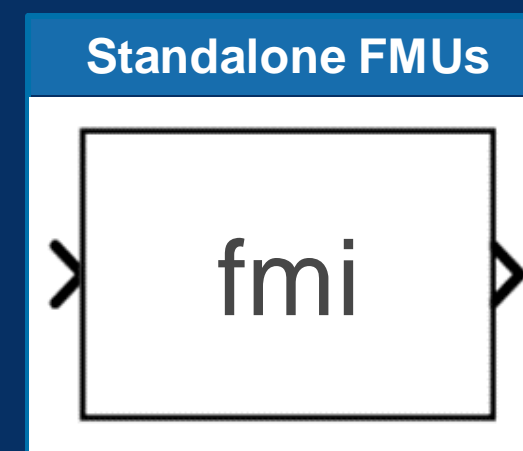
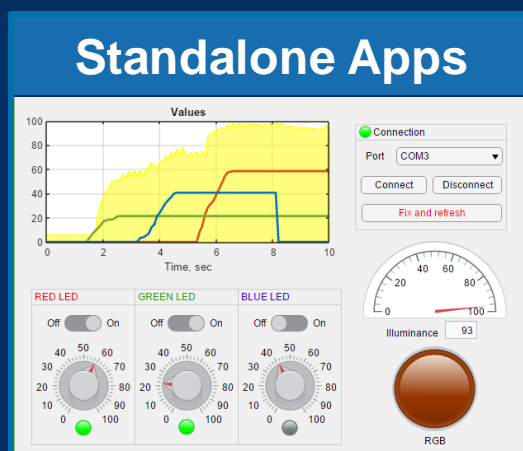
Share MATLAB apps as browser-based web apps

Create apps using App Designer
and host them using
MATLAB Web App Server



Share Simulink simulations – *where Simulink is not available*

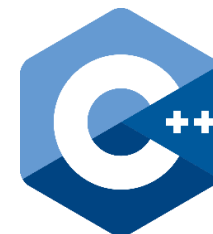
Package a compiled Simulink model with MATLAB code



Deploy algorithms with automatically generated code

C++ classes from MATLAB classes

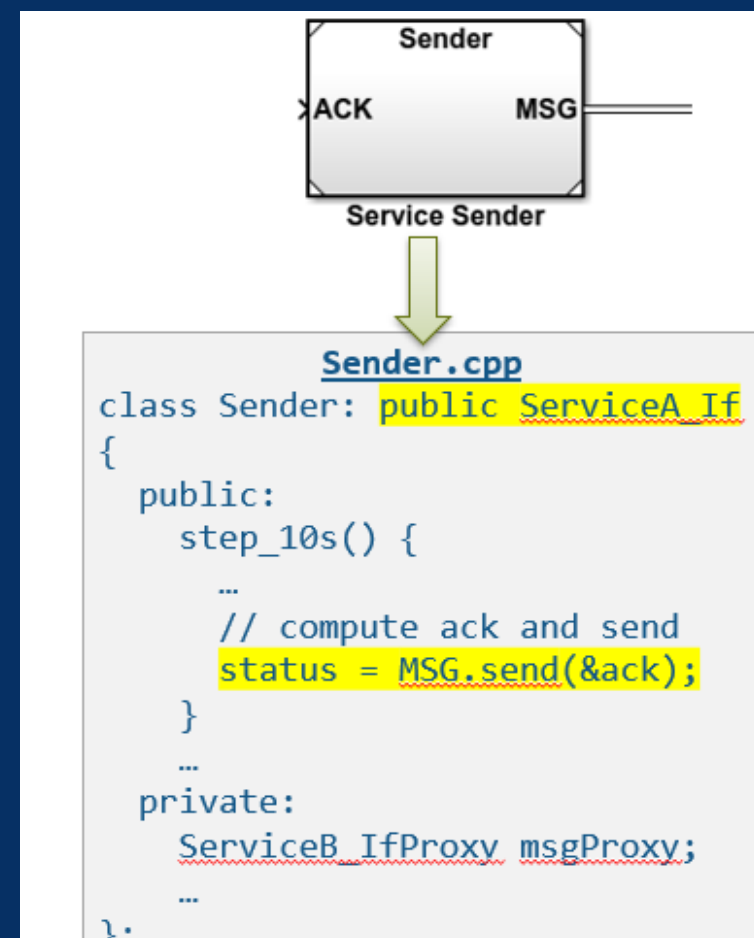
```
class MyClass
{
public:
    MyClass *init();
    void publicMethod(double value);
    static double doubleThisValue(double val);
    double calculateSomeValue() const;
private:
    MyClass *matlabCodegenHandle_init();
    MyClass *privateMethod(double value);
public:
    double publicProp;
private:
    double privateProp;
};
```



Deploy algorithms with automatically generated code

C++ classes from MATLAB classes

Code from software compositions
with message-based communication

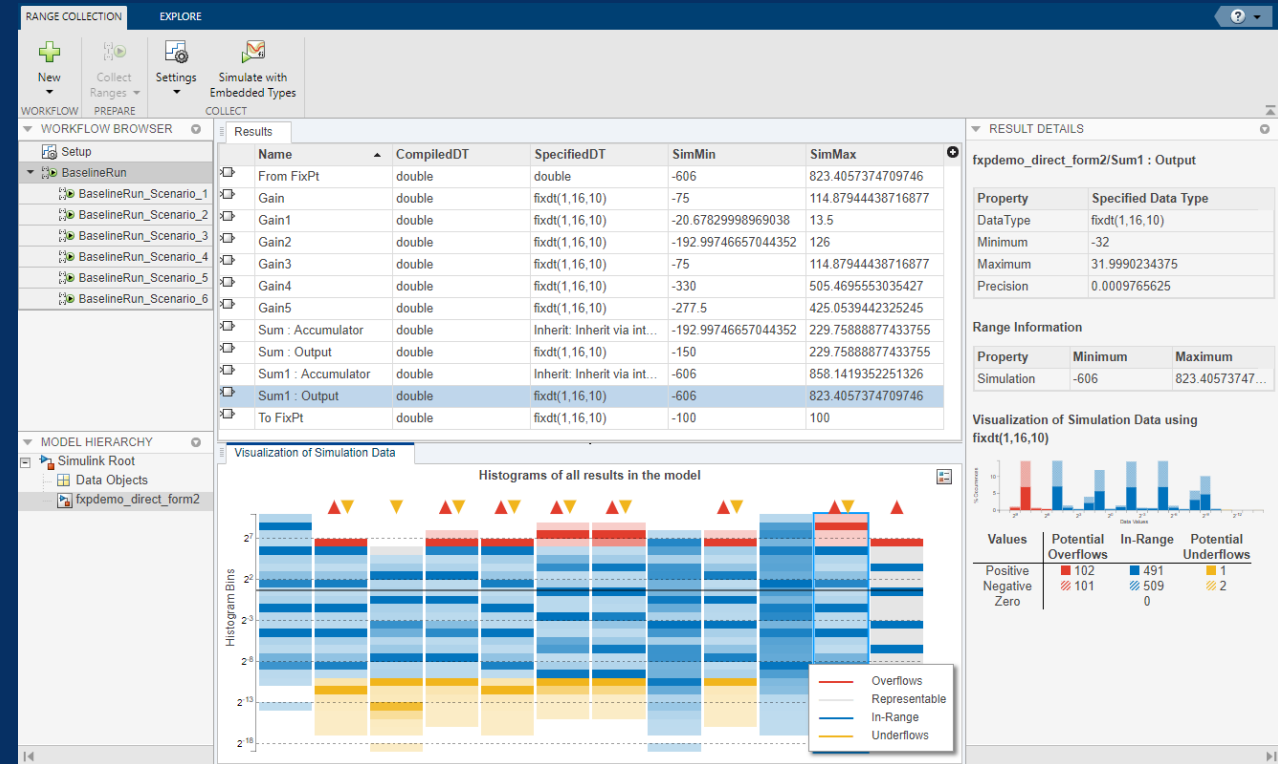


Deploy algorithms with automatically generated code

C++ classes from MATLAB classes

Code from software compositions
with message-based communication

Explore signal ranges in designs,
and data type optimization



Fixed-Point Tool



**MATLAB®
& SIMULINK®**



**Test and Verify
Share and Deploy**



**Deep
Solutions**



Artificial Intelligence (AI)

MATLAB

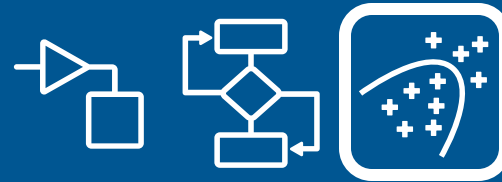
Access Data



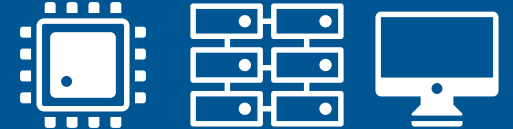
Preprocess Data



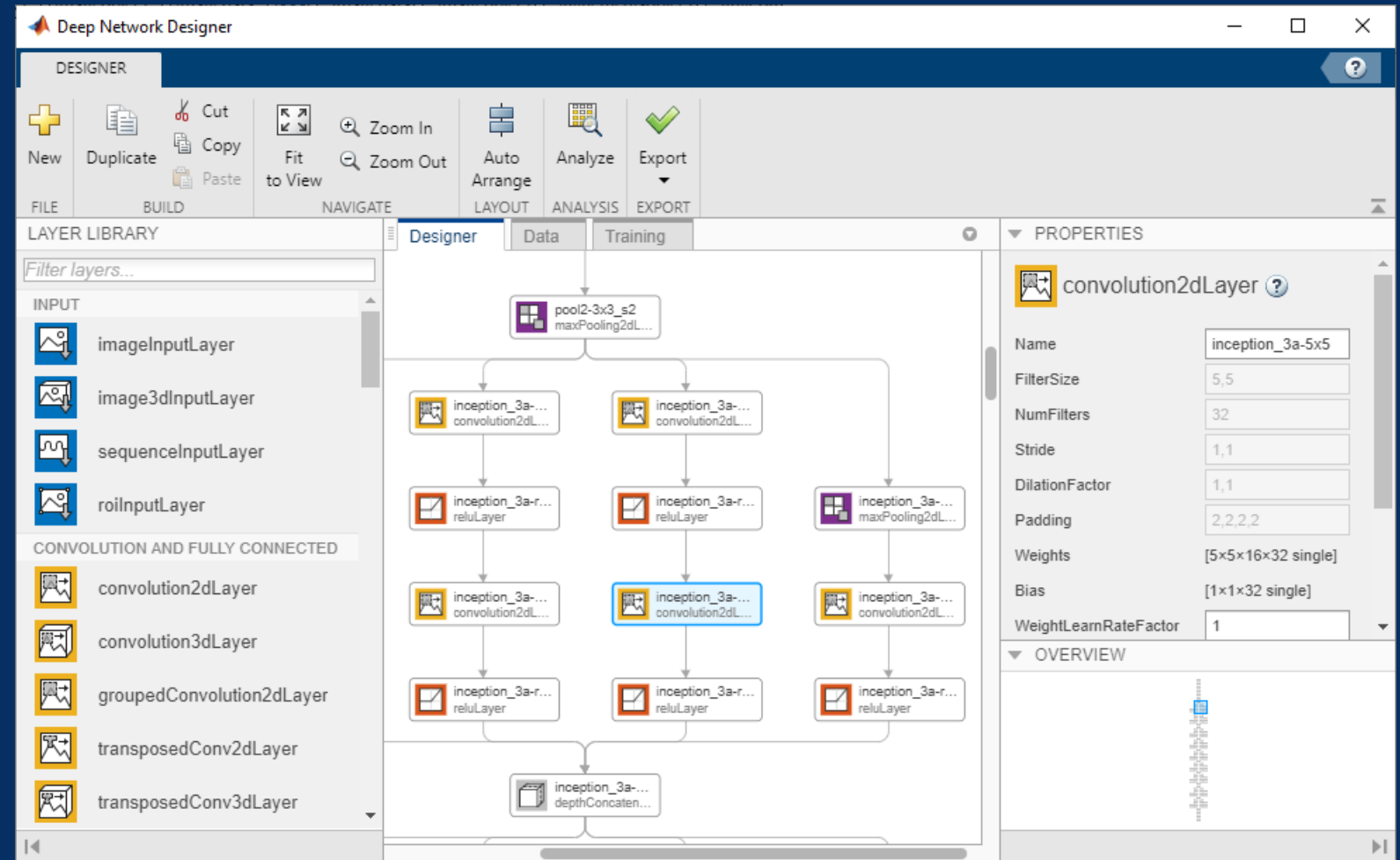
Develop



Deploy



Interactively access models, and develop and train networks

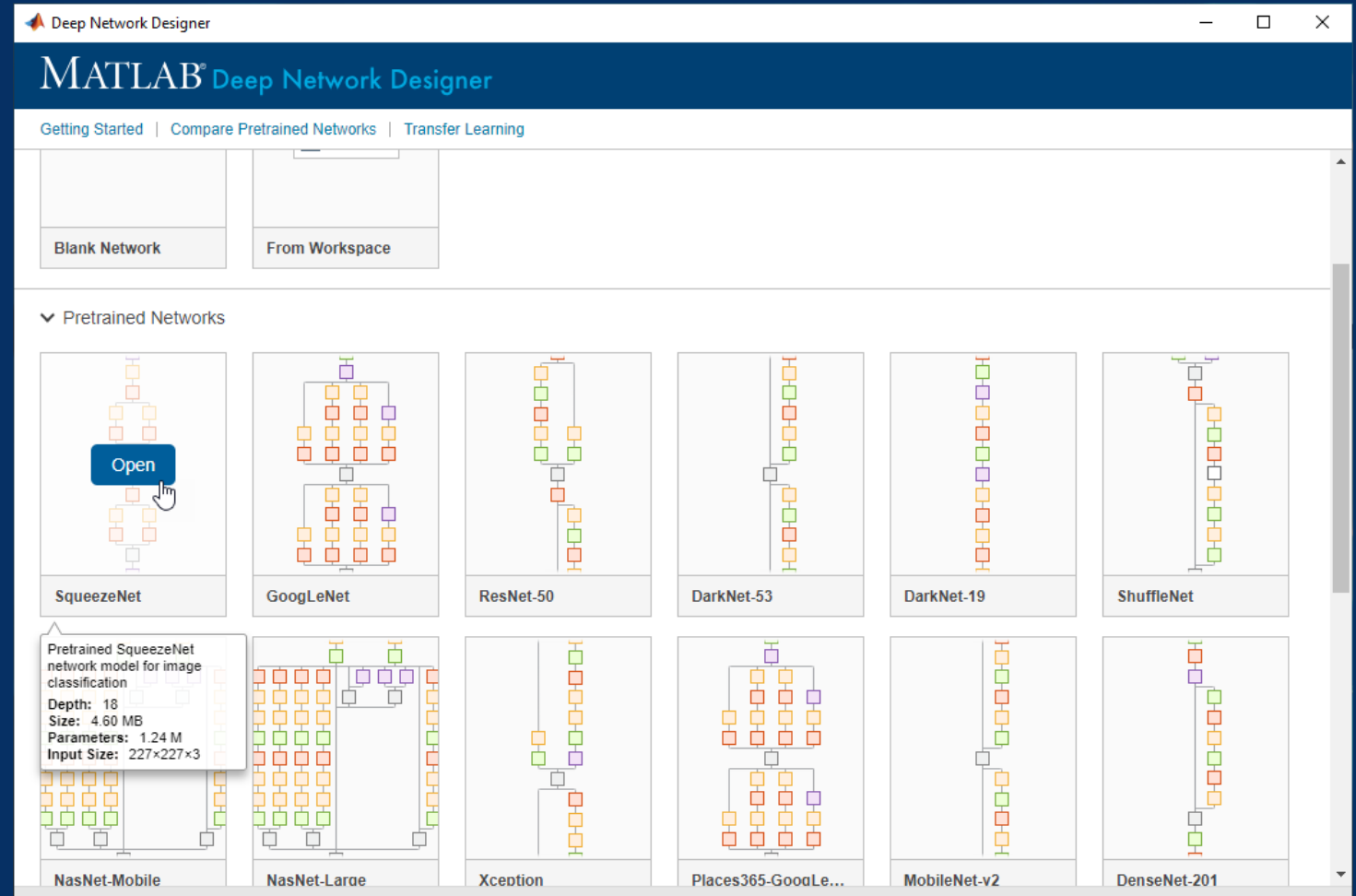


Deep Network Designer App

Interactively access models, and develop and train networks



Import pretrained
networks for transfer
learning



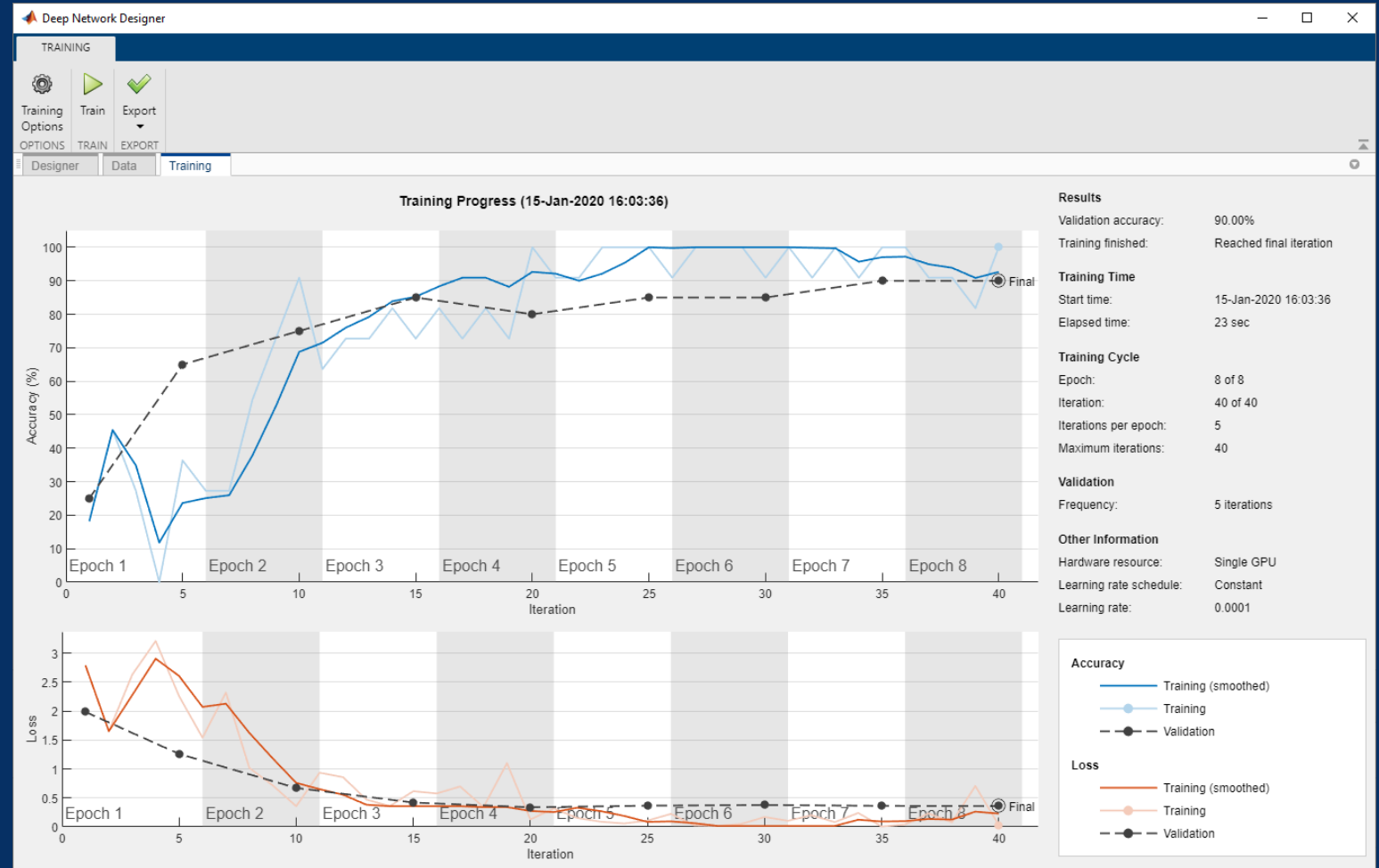
Deep Network Designer App

Interactively access models, and develop and train networks



Import pretrained networks for transfer learning

Train networks and generate MATLAB code



Deep Network Designer App

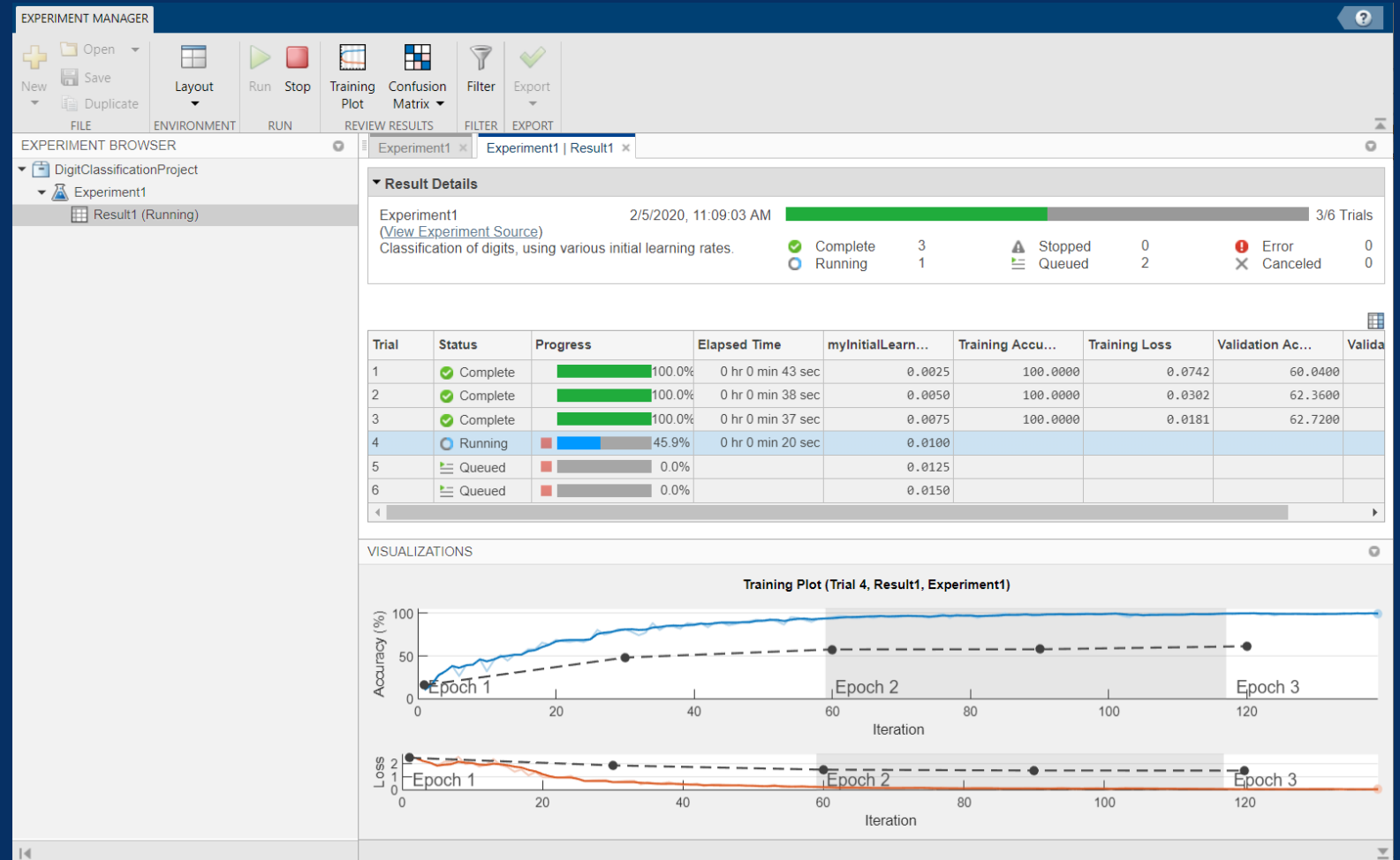


Manage multiple deep learning experiments

Keep track of training parameters

Reuse training data across multiple networks

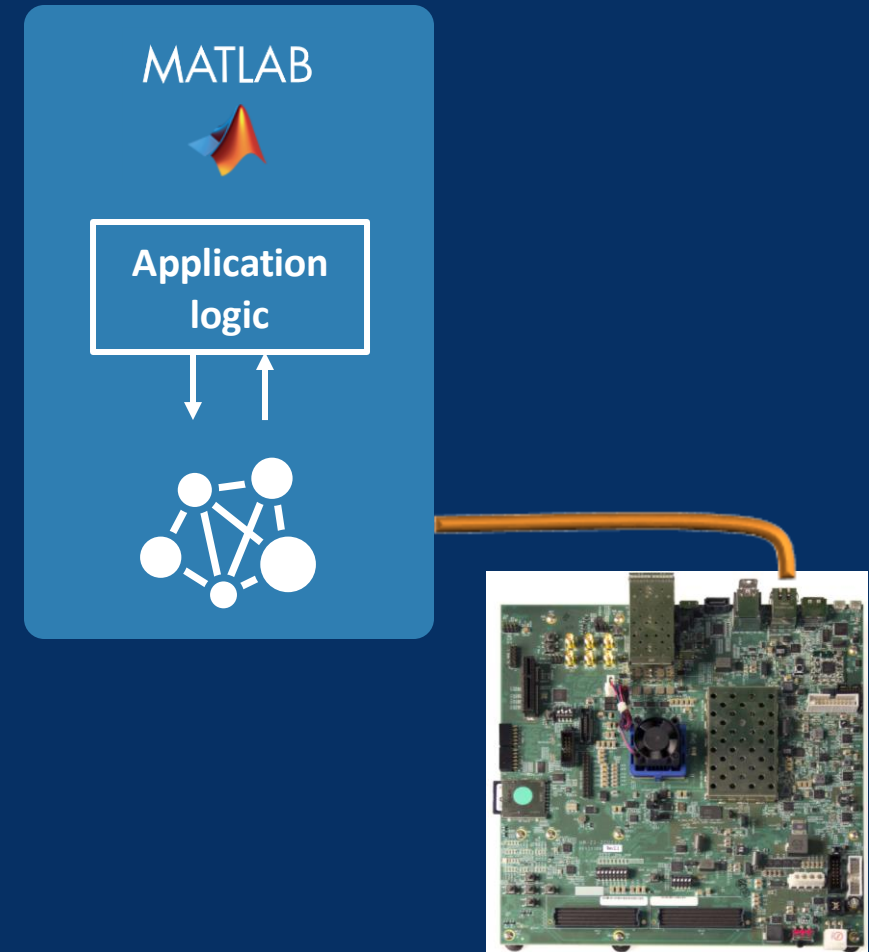
Analyze and compare results



Experiment Manager App

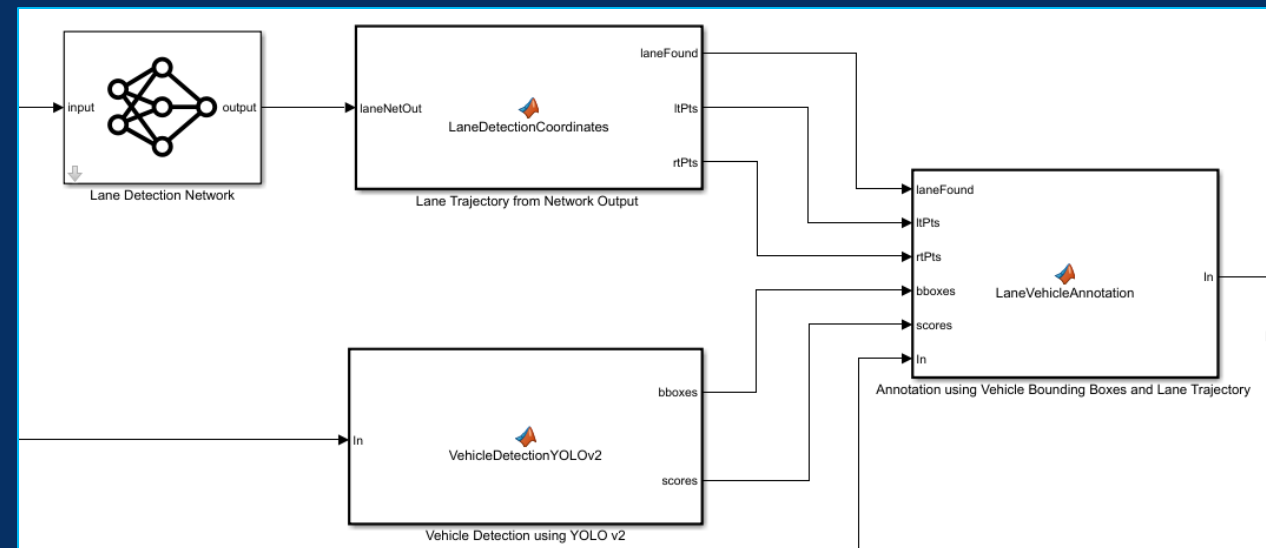
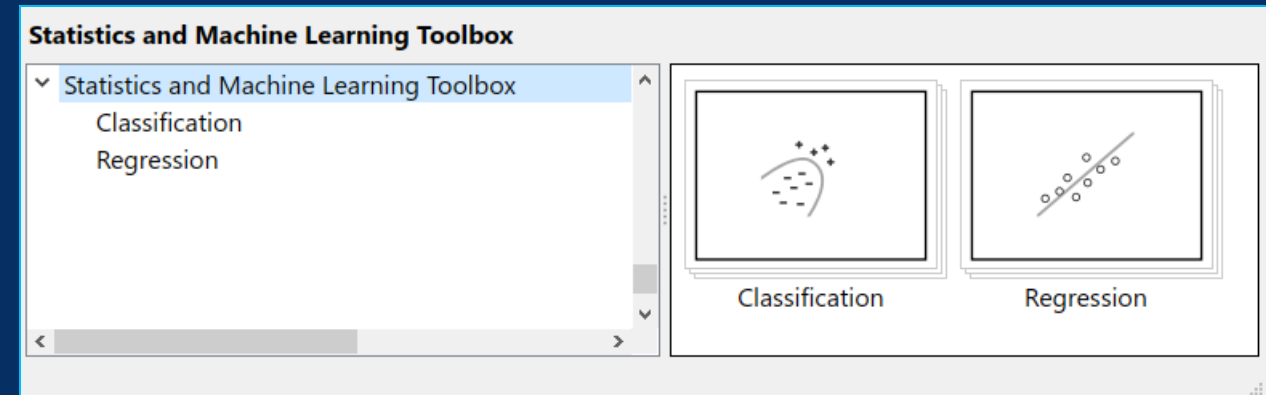
Prototype and deploy deep learning networks on FPGAs and SoCs

- Run deep learning inferencing on FPGAs directly from MATLAB
- Use pre-built bitstreams for running on supported Xilinx and Intel devices



Deep Learning in Simulink

- Simulate and generate code using native blocks for support vector machine (SVM) models
- Generate, build, and deploy deep learning networks in Simulink models to NVIDIA GPUs



Simulink
Statistics and Machine Learning Toolbox
MATLAB Coder
GPU Coder

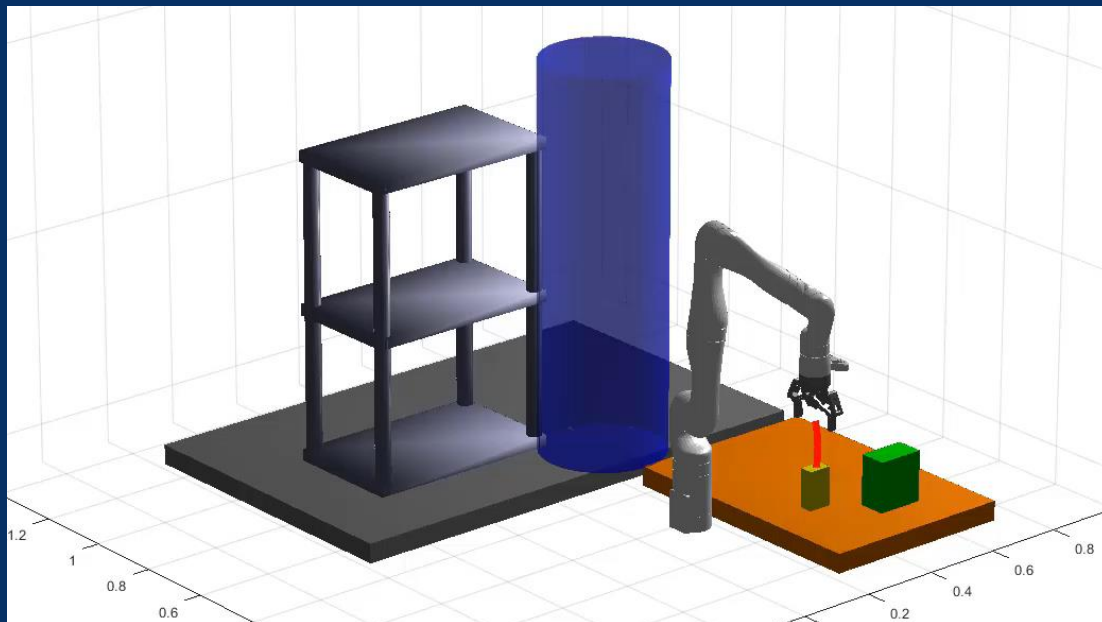


Robotics and Autonomous Systems





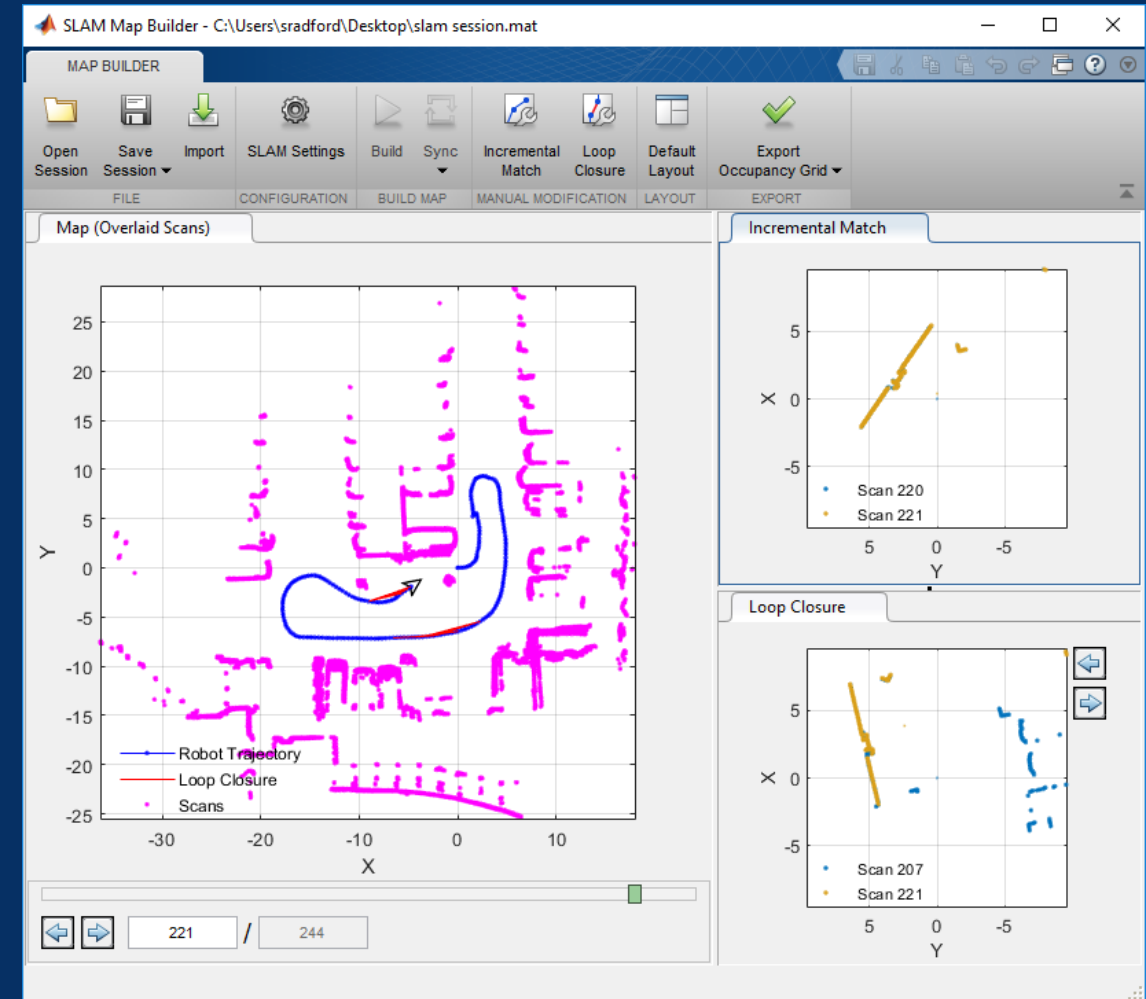
Simulate and visualize robot kinematics





Design algorithms for planning and navigation

- Create a map of the environment

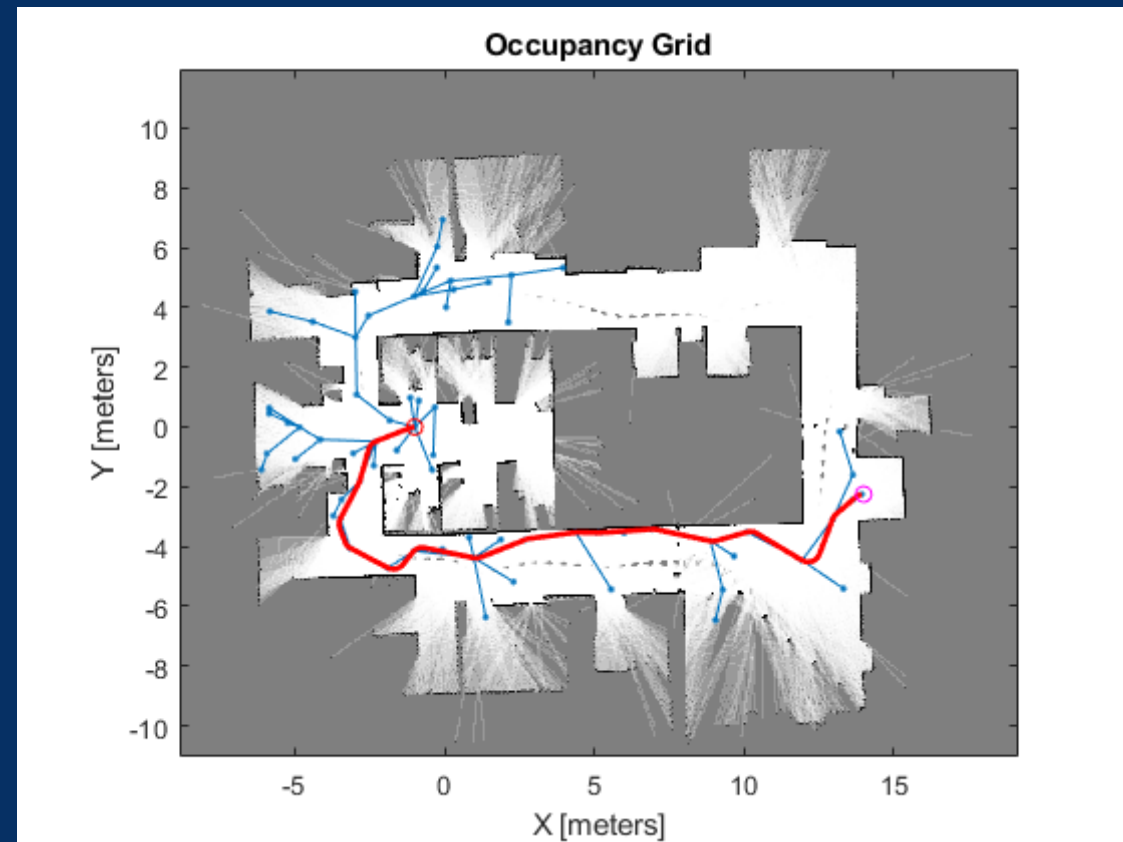


SLAM Map Builder



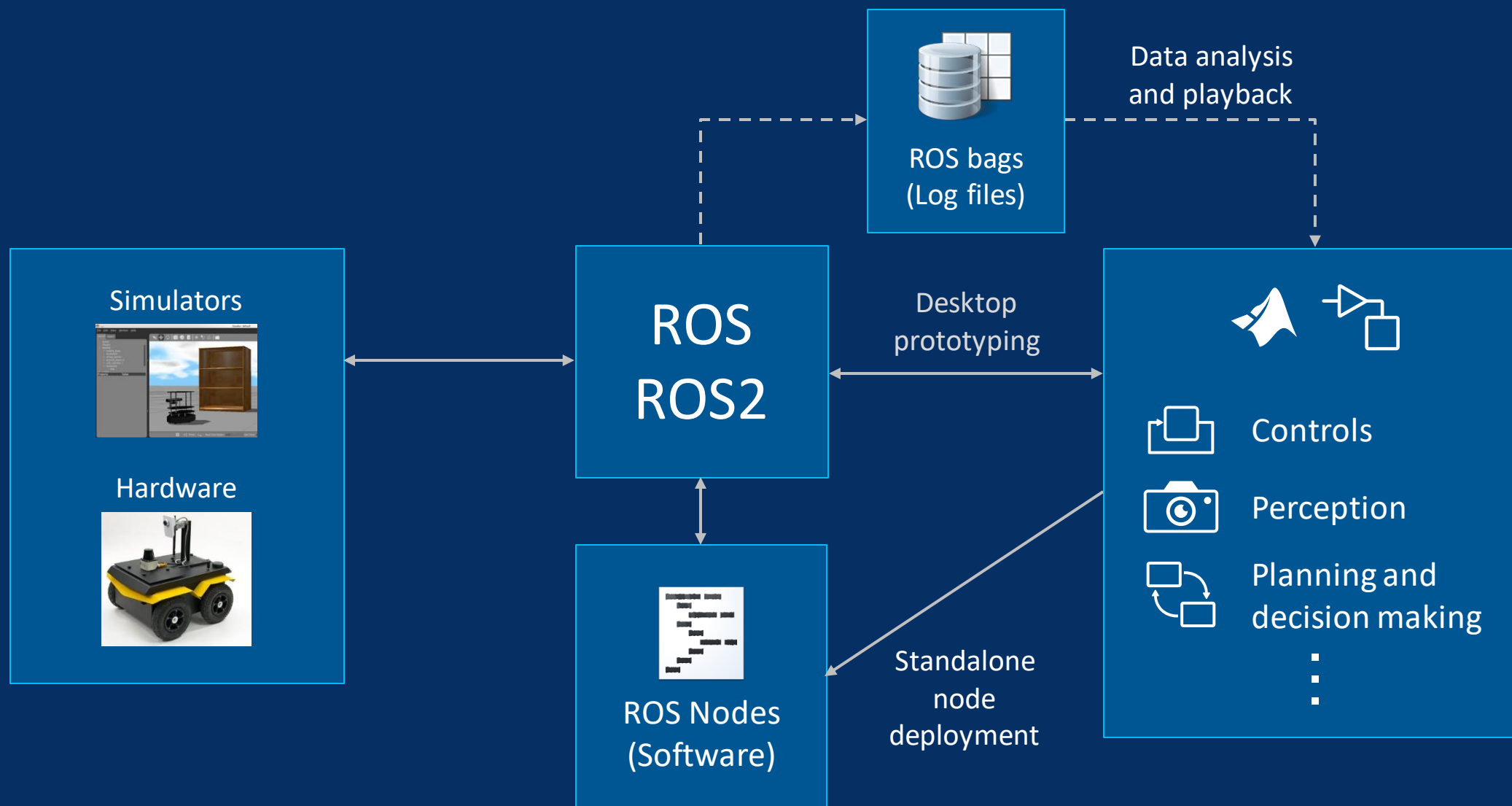
Design algorithms for planning and navigation

- Create a map of the environment
- Plan a path through a known map

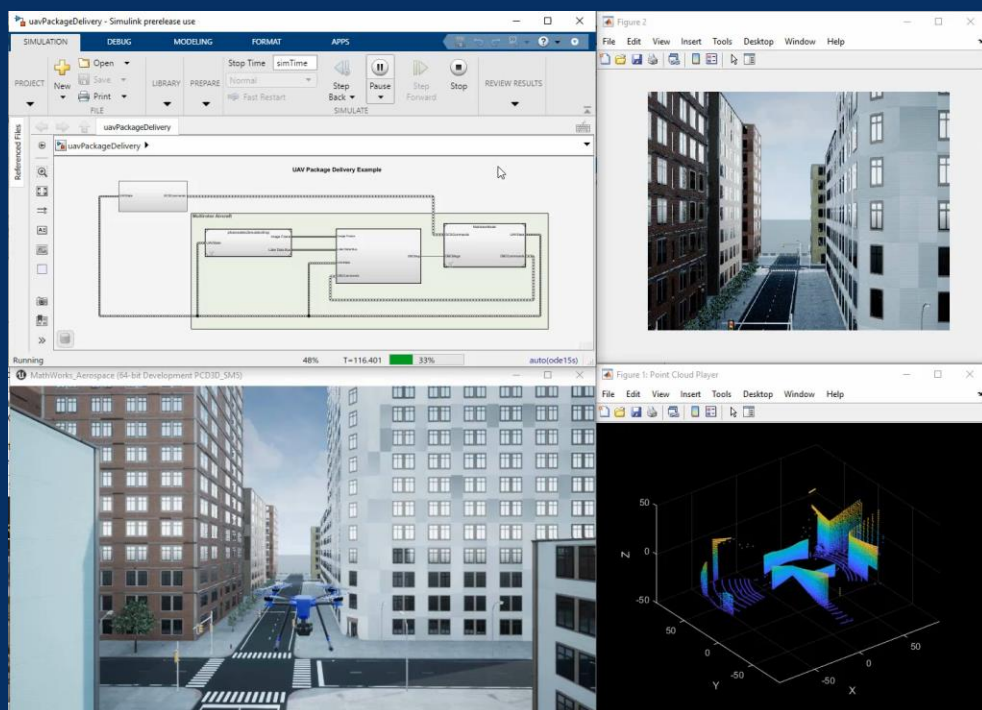




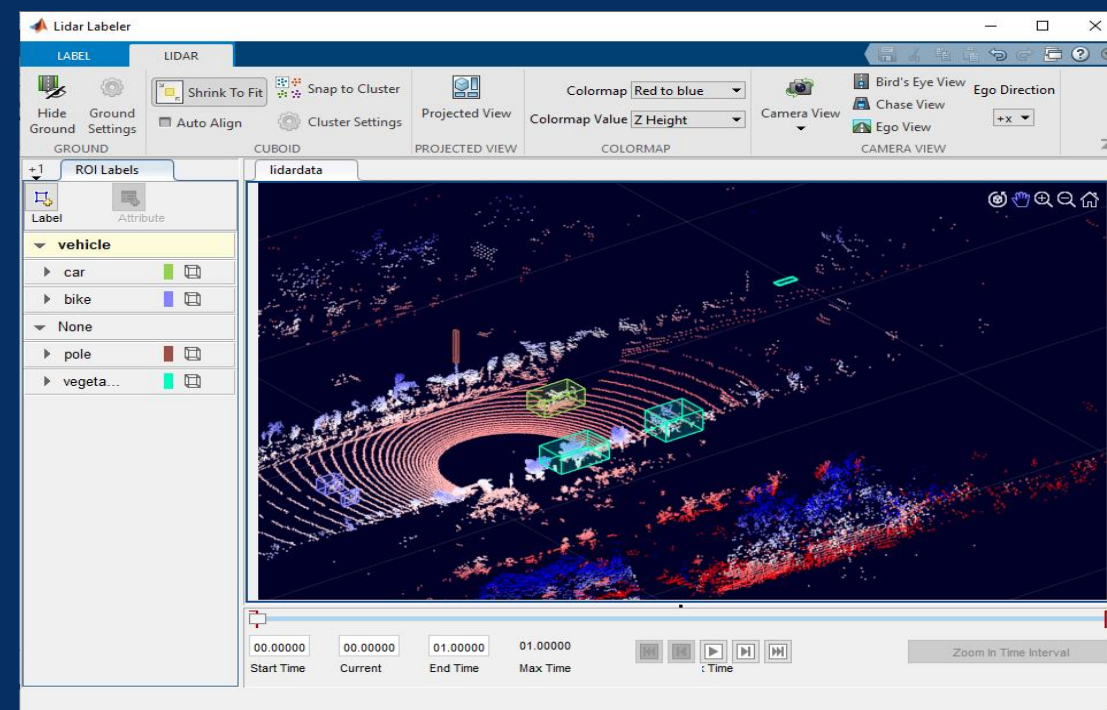
Design, simulate, and deploy ROS-based applications



Develop UAV applications and lidar processing systems



UAV Toolbox



Lidar Toolbox



Wireless Communications

RF & Antenna

RF Toolbox

RF Blockset

Antenna
Toolbox

Phased Array
Toolbox

Digital Baseband

Communications
Toolbox

Wireless HDL
Toolbox

Standards

5G Toolbox

LTE Toolbox

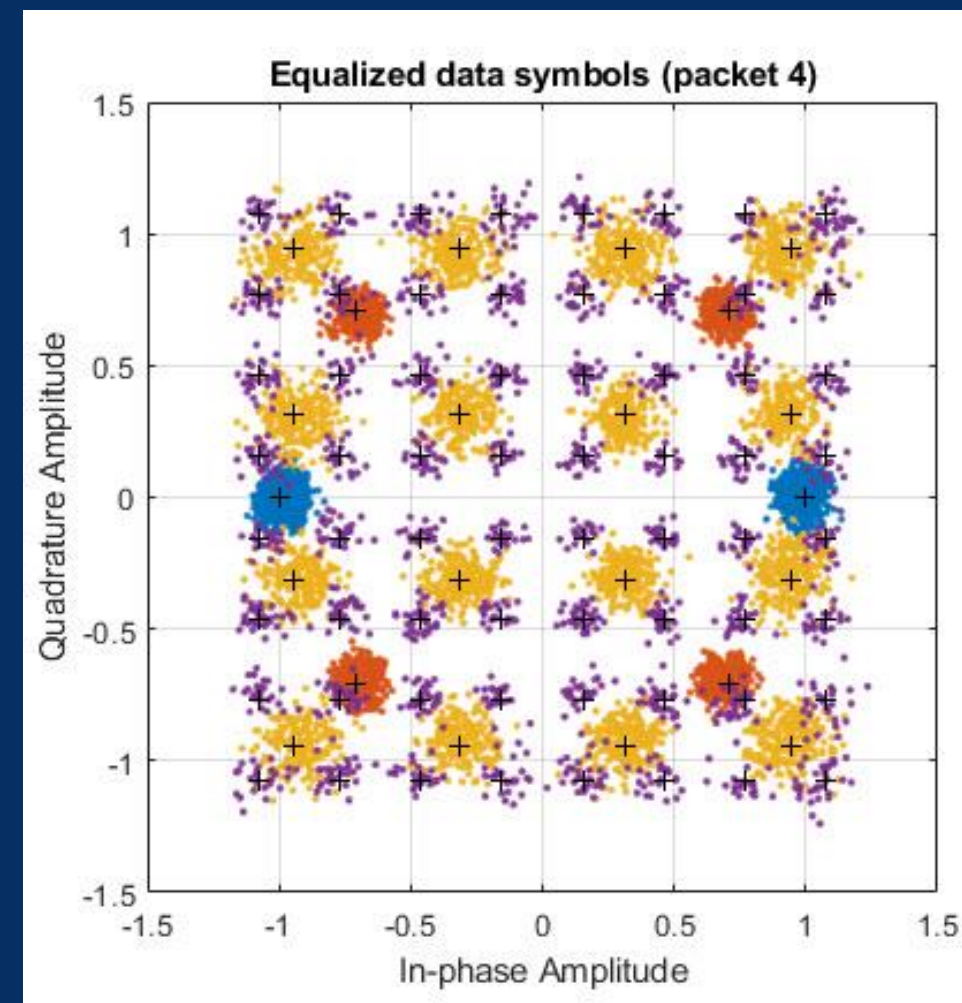
WLAN Toolbox



Model, simulate, and test Wi-Fi 6 systems

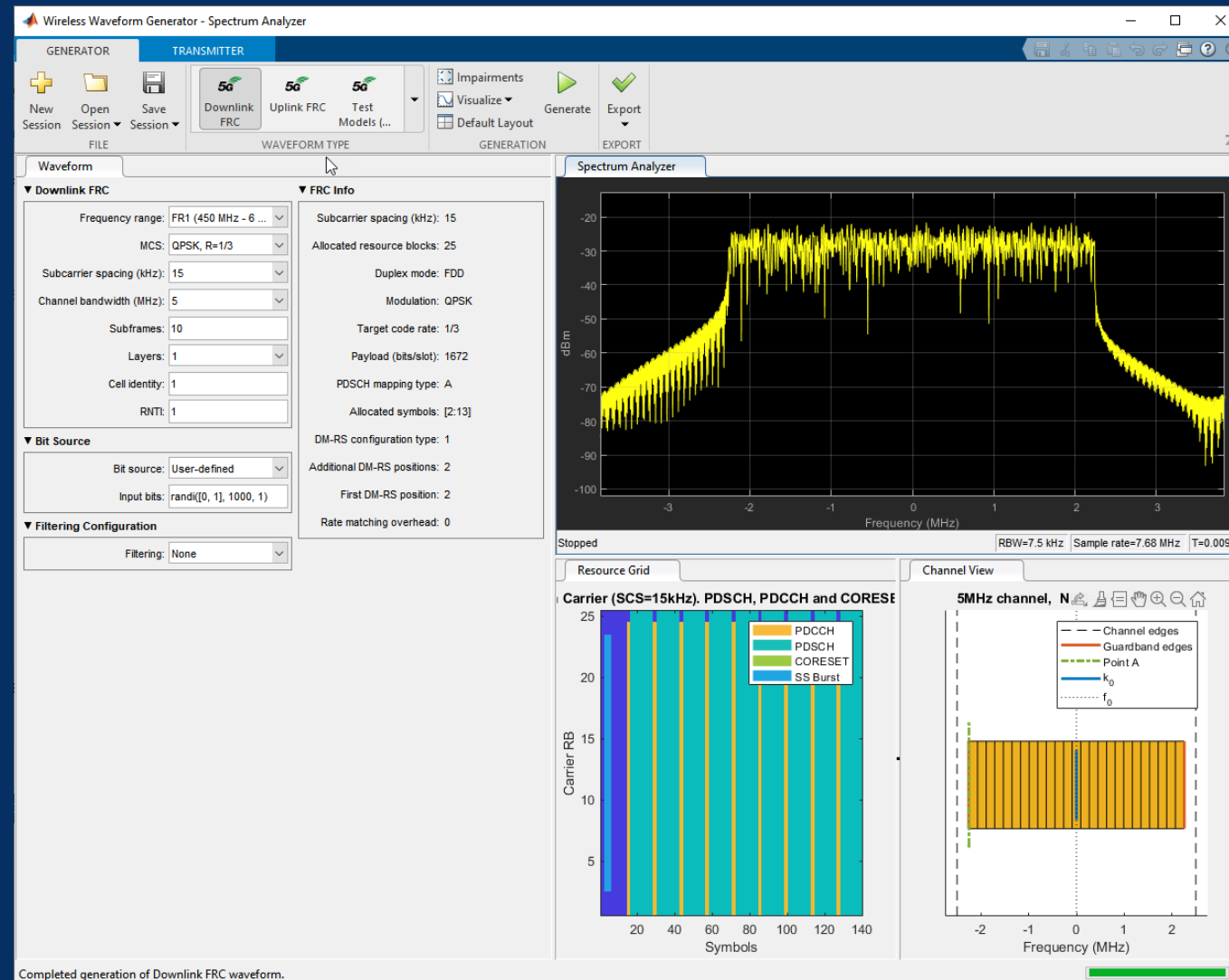
Generate P802.11ax™ Draft 4.1 waveforms

Link-level simulation of 802.11ax Trigger-Based Format





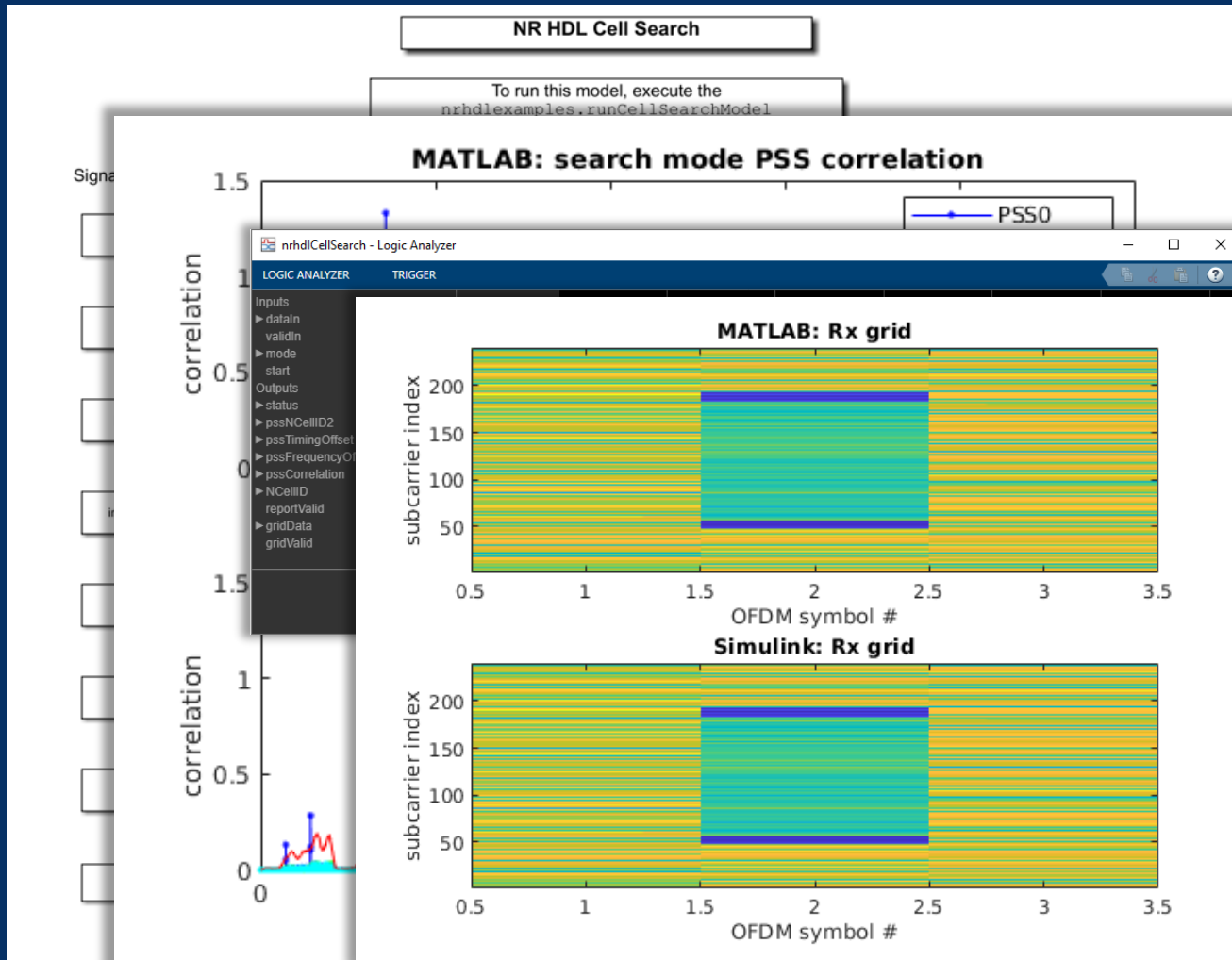
Interactively generate 5G waveforms for testing



Wireless Waveform Generator



Start with reference examples to implement your design



Standards-based IP

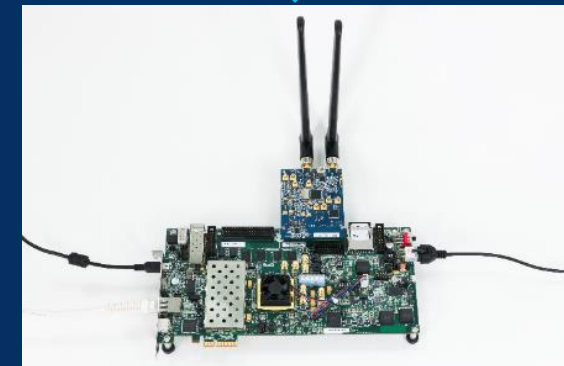
+

Your expertise

Wireless
HDL Toolbox

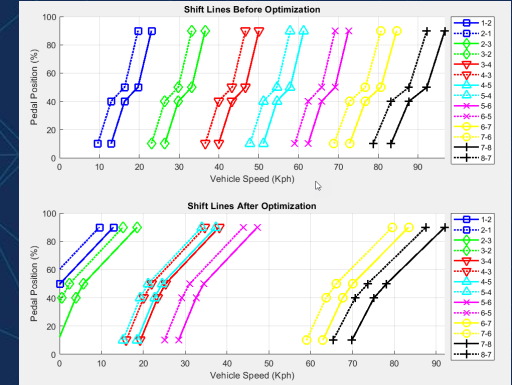
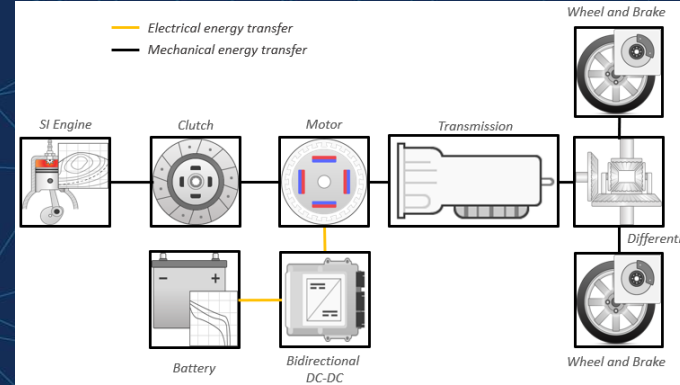
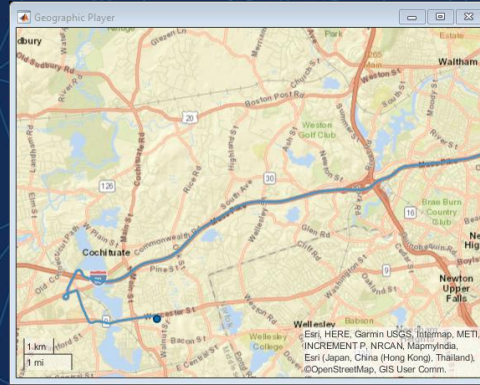
Your unique
application

HDL Coder





Automotive Systems





Accelerate development of automated driving systems

Develop driving algorithms in a 3D simulation environment

Test algorithms with prebuilt scenarios



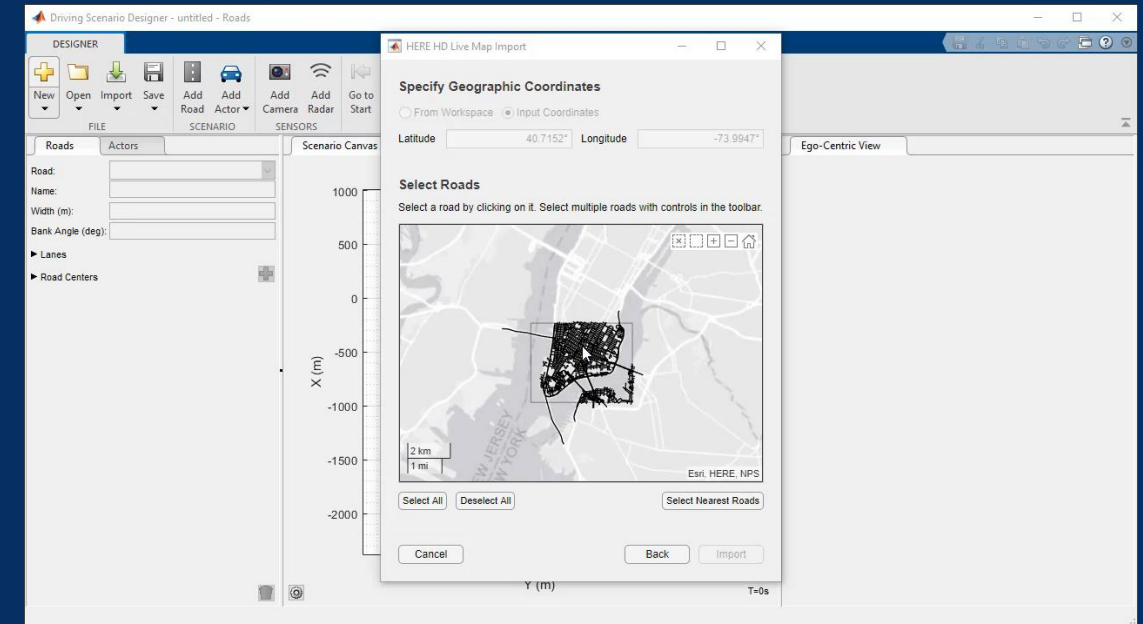


Accelerate development of automated driving systems

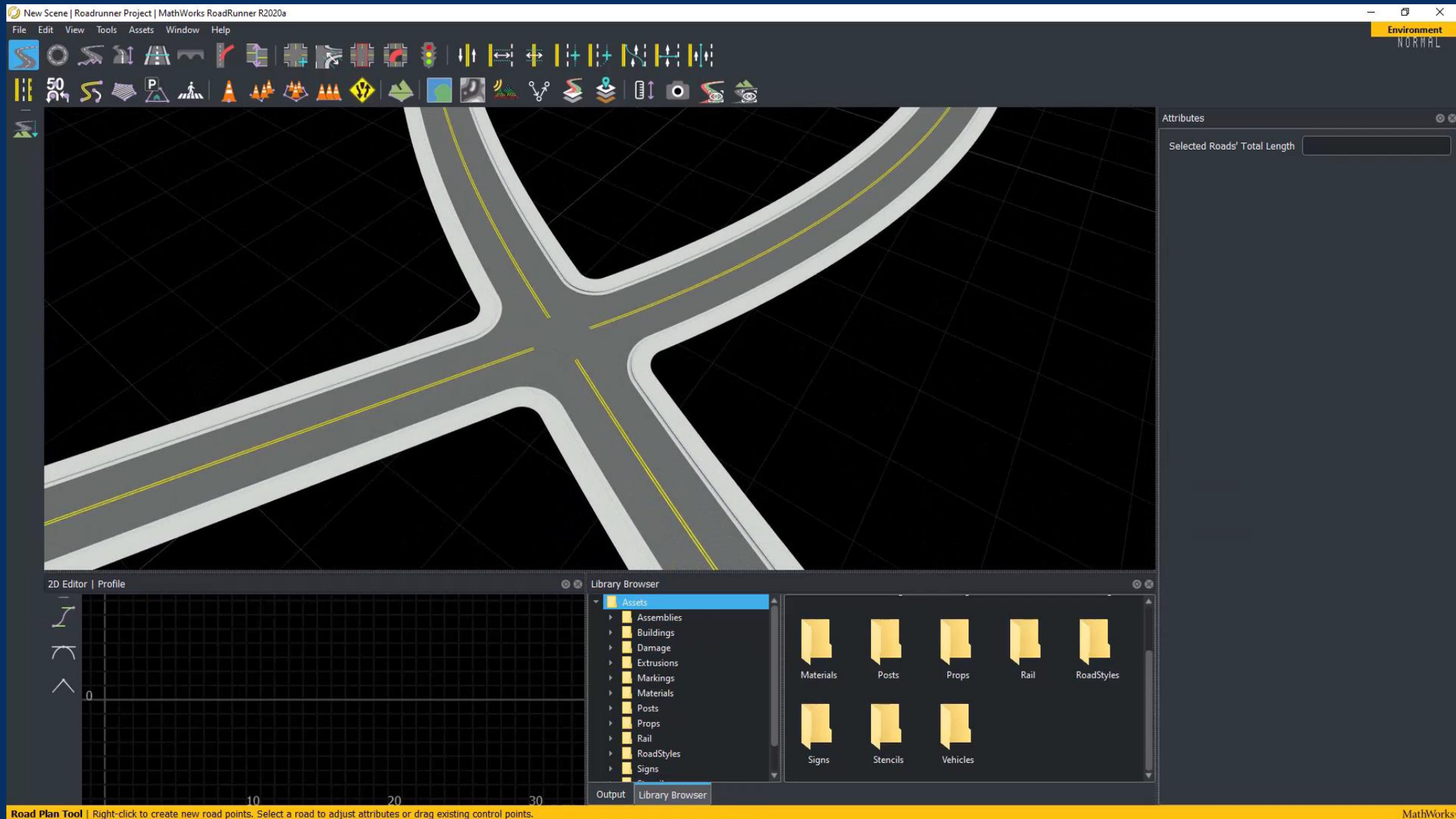
Develop driving algorithms in a 3D simulation environment

Test algorithms with prebuilt scenarios

Create driving scenarios using road data from high-definition maps

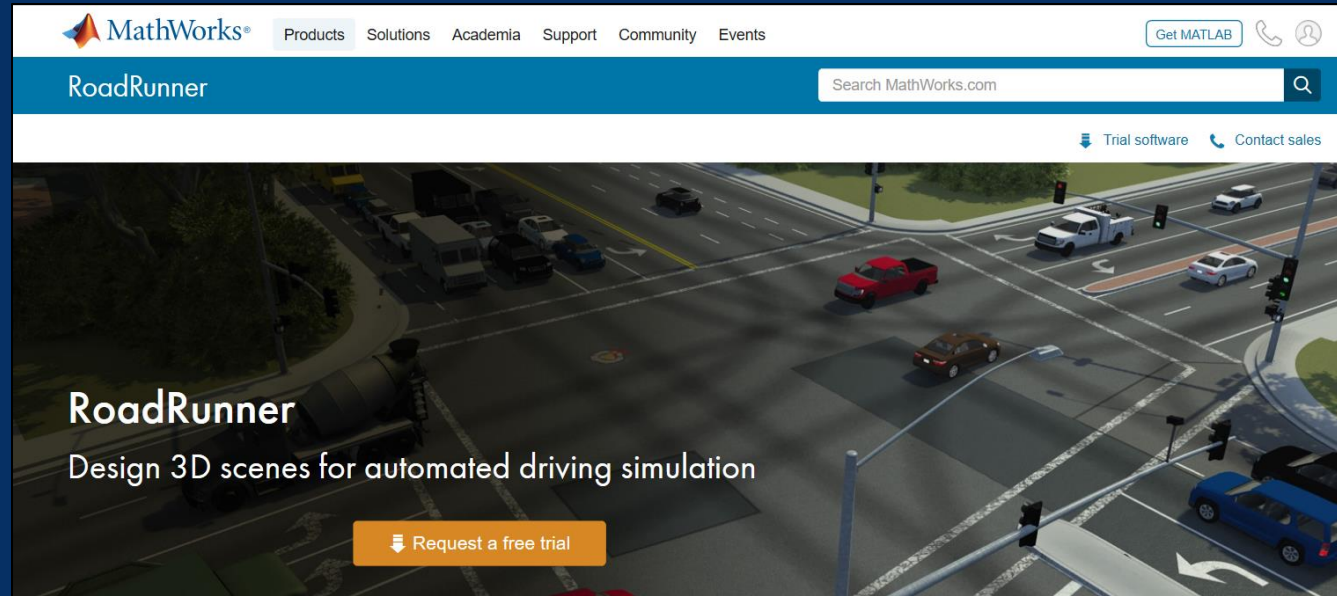


Accelerate development of automated driving systems

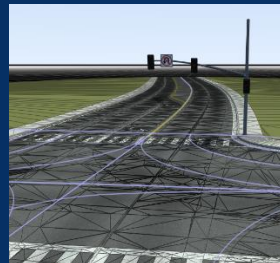
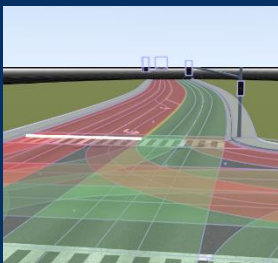




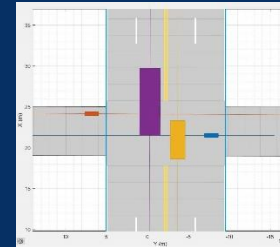
Accelerate development of automated driving systems



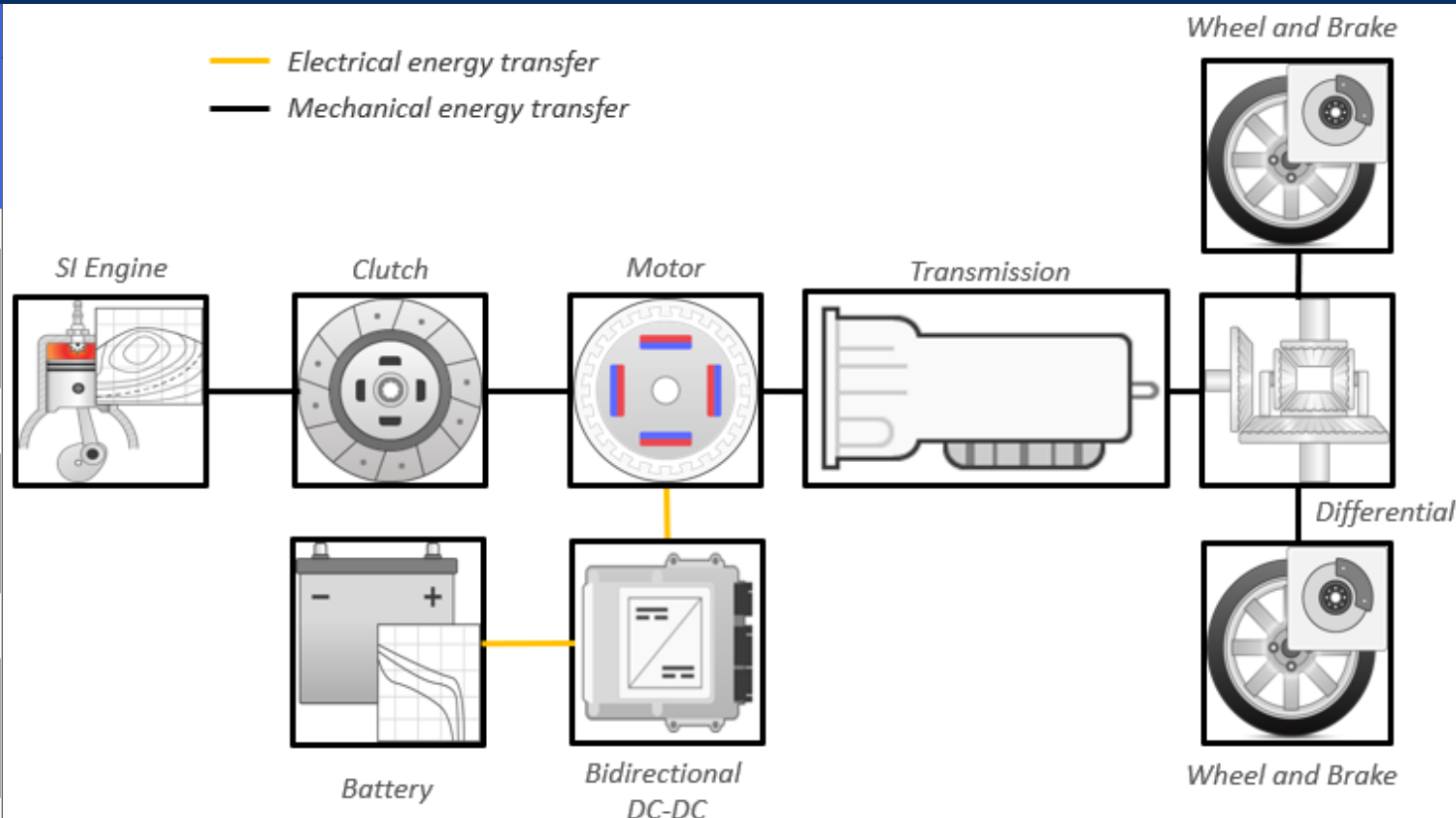
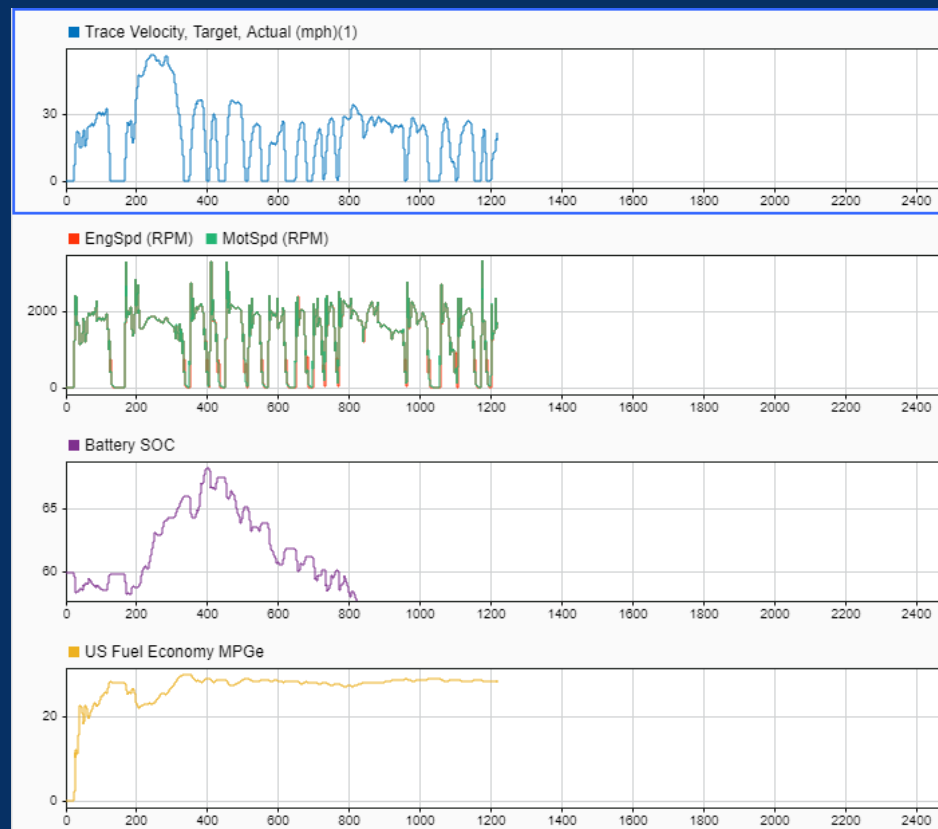
External Simulators



MATLAB & Simulink



Analyze fuel economy and performance for various architectures





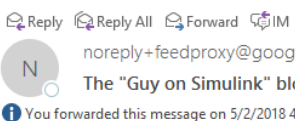
**MATLAB®
& SIMULINK®**



**Test and Verify
Share and Deploy**

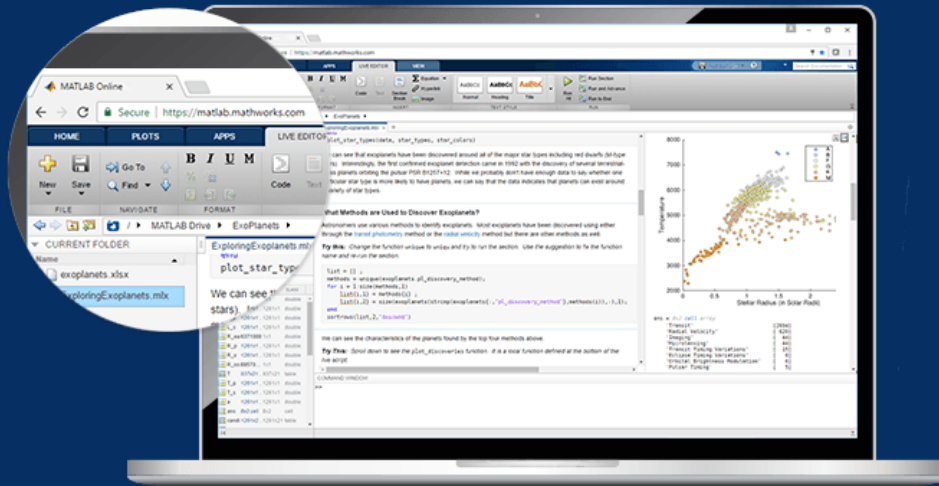


**Deep
Solutions**



R2020a at a Glance

Access and try the latest release with MATLAB Online

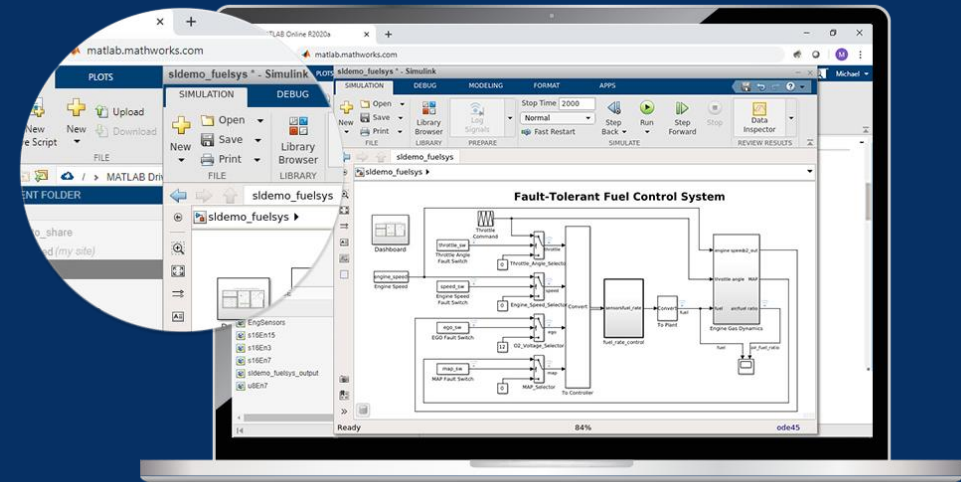


MATLAB Online

Access and try the latest release with MATLAB and Simulink Online



MATLAB Online



Simulink Online

Access and try the latest release with MATLAB and Simulink Online

MATLAB Online R2020a

matlab.mathworks.com

HOME sldemo_fuelsys - Simulink

SIMULATION DEBUG MODELING FORMAT APPS

Stop Time 2000

Normal

Fast Restart Step Back Run Step Forward Stop

Data Inspector

FILE LIBRARY PREPARE SIMULATE REVIEW RESULTS

NAME

Example

Public

Share

slprj

analysis

analysis

Myproj

sensor

WORKSPACE

Name

EngSensor

out

s16En15

s16En3

s16En7

u8En7

sldemo_fuelsys

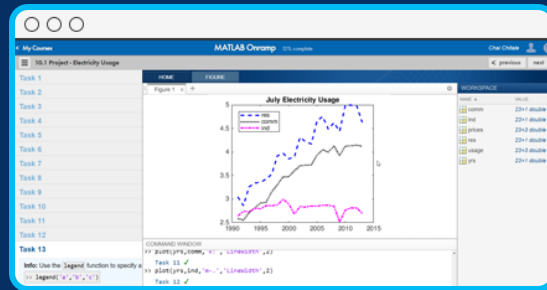
sldemo_fuelsys

Fault-Tolerant Fuel Control System

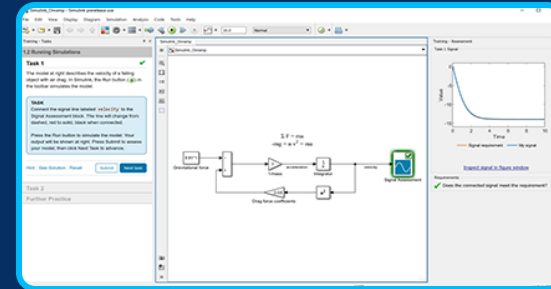
Open the Dashboard subsystem to simulate any combination of sensor f...

Ready 79% ode45

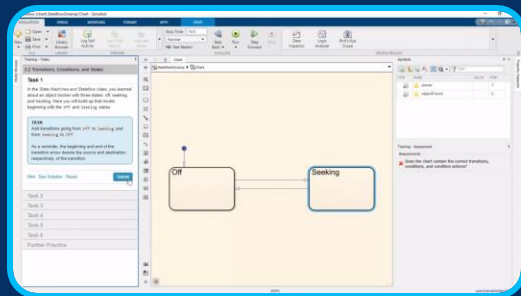
Get started and learn with Onramps



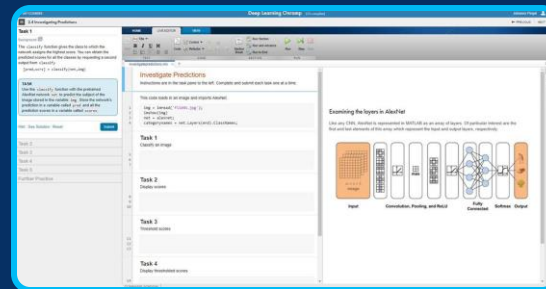
MATLAB Onramp



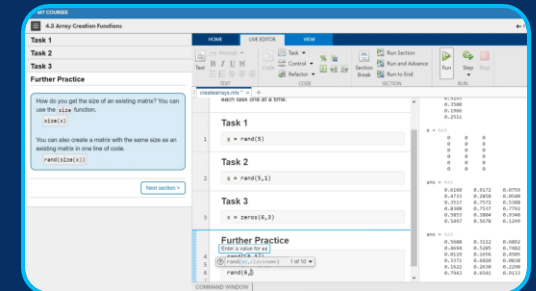
Simulink Onramp



Stateflow Onramp



Deep Learning Onramp



Machine Learning Onramp

MATLAB EXPO

